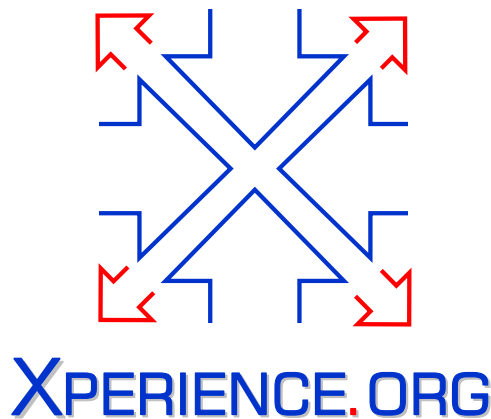




Project Acronym:	Xperience
Project Type:	IP
Project Title:	Robots Bootstrapped through Learning from Experience
Contract Number:	270273
Starting Date:	01-01-2011
Ending Date:	31-12-2015



Deliverable Number:	D2.1.2
Deliverable Title:	Sensorimotor Experience (II): Report or scientific publication on representation of motor actions, their learning, sequencing and their use to generate advanced motor behaviours with focus on discrete movements and grasping.
Type (Internal, Restricted, Public):	PU
Authors:	F. Wörgötter, T. Asfour, J. Piater, A. Ude, D. Teney, H. Xiong, D. Schiebener, N. Krüger, D. Kraft, G. Metta, R. Bevec and R. Dillmann
Contributing Partners:	UGOE, UIBK, SDU, IIT, KIT, JSI

Contractual Date of Delivery to the EC: 31-01-2014
Actual Date of Delivery to the EC: 18-03-2014

Contents

1	Executive Summary	3
1.1	Summary of the Results	3
1.2	Links to other Workpackages	3
2	Grasping	4
2.1	Context dependent Grasping	4
2.2	Learning Grasping Affordances with Features of Different Granularity, Order and Semantic	5
2.3	Grasping Unknown Objects Based on RGBD Images	6
2.4	Object Models	6
2.4.1	Object Recognition and Pose Estimation in 2D Images	6
2.4.2	3D Reconstruction	6
2.4.3	3D Shape Pose Estimation	7
2.4.4	Reduced, dense 3D Scene representation of RGB-D images	7
3	Physical Interaction for Object Learning	8
3.1	Learning Textured and Non-textured Object Representations	8
3.2	Acquiring Object Models by Foveated Vision	9
4	Action Recognition and Object Categorization	11

Chapter 1

Executive Summary

1.1 Summary of the Results

This deliverable focuses on methods that involve grasping actions developed in **WP2.1** and the application of motor actions to support visual perception. It consists of the following contributions:

- Chapter 2 describes our work on context-dependent transfer of grasps to different tasks (Sec. 2.1) and on learning grasping affordances based on feature-action associations with different granularity, order and semantics (Sec. 2.2). In Section 2.3 we describe an approach for grasping unknown objects based on RGBD images. Finally, in Section 2.4 we report on several techniques for acquiring object models needed for grasping. These range from object recognition and pose estimation (Sec. 2.4.1), 3D reconstruction (Sec. 2.4.2) to 3D shape pose estimation (Sec. 2.4.3) and dense 3D representations (Sec. 2.4.4).
- Chapter 3 deals with learning object representation through physical interaction. We describe the extension of learning object representations through pushing actions to deal with non-textured objects (Sec. 3.1) as well as to exploit foveated vision to improve the learning results (Sec. 3.2).
- Chapter 4 describe a machine learning approach for one-shot action learning and recognition based on RGBD images, optical flow and sparse coding techniques for action representation. The work is extended to object recognition.

1.2 Links to other Workpackages

Since discrete movements are mainly investigated in **WP2.2**, we report on our research regarding learning and sequencing of discrete movements in **D2.2.2**. Representations of discrete movements that involve cooperation with other agents are investigated in **WP4.1**. Progress regarding cooperative discrete movements is described in **D4.1.2**.

Chapter 2

Grasping

2.1 Context dependent Grasping

In [REK⁺ed], we have investigated the role the task context plays for grasping. It is a common practice that suitable grasps are computed off-line in simulation in a free floating environment (see top row of Figure 2.1). These grasps are then applied in a specific task context, e.g., a table environment (see bottom row of Figure 2.1). It remains unclear to what degree such a transfer is valid or reasonable. For example Figure 2.1 shows a grasp that is not successful in a free floating environment but is successful in a table environment.

We have investigated this transfer for two grippers – one of them being able to realize different grasp types – with three objects in three different tasks contexts (free-floating, standing on table and lying on table). Our investigations show that such a transfer is far from being straightforward. In particular, it is not only that many grasps which work in the free floating environment cannot be applied in the other two environments (where the trivial effect of collisions is ignored) but also that the table environment makes a large set of grasps feasible which are not valid grasps in the free floating environment. We then also evaluate the transfer from two simulated contexts to a real world context of picking an object. From our results the advantages and need of context-dependent simulation becomes apparent which is of importance for the proper use of simulators as a means to facilitate investigations of learning tasks by simulation as done in the Xperience project.

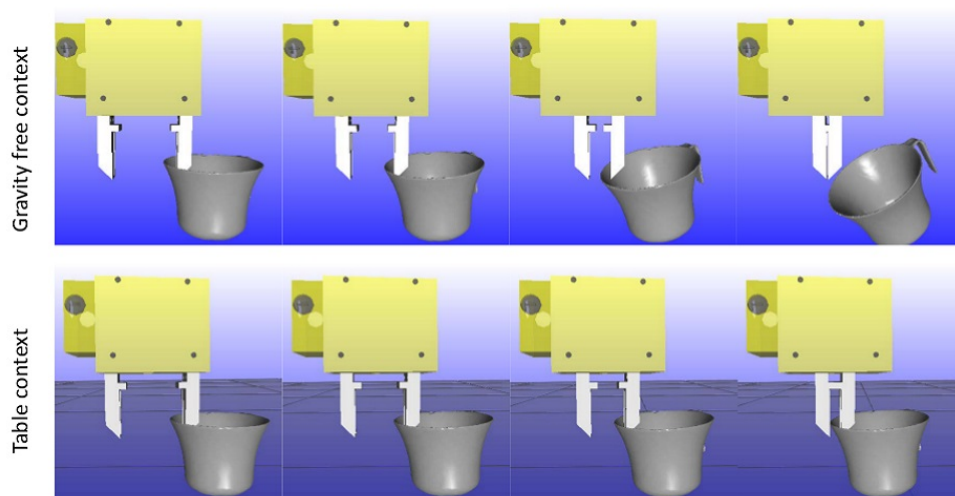


Figure 2.1: Two simulated grasp sequences of the same grasp in two different contexts. In the top sequence the grasp of the cup fails in the gravity-free environment while in the bottom sequence the cup is successfully grasped when placed on top of a table.

2.2 Learning Grasping Affordances with Features of Different Granularity, Order and Semantic

In [TKK14], we investigate feature-action association where features vary according to their granularity (spatial size, see Figure 2.2A), their order (first or second order spatial combination, see Figure 2.2B) and their abstraction (being a surface patch, a surface patch with border label or a surface patch with border label with additional direction information, see Figure 2.2C). These features correspond to different levels of the visual hierarchy as extensively discussed in [KJK⁺13]). We combine the perceptual features with a grasping action into particles and then apply a straightforward evaluation method based on neighborhoods in the feature space extracting success probability and support for each particle and store these particles in a data base. We then apply two learning mechanisms on the particles in the data base which allow for the extraction promising grasp affordances for a novel scene. As a main result, we show that the same learning algorithm leads to very different results depending on the structure of the perceptive features. The particles found as promising for affordance prediction are building blocks for structural bootstrapping processes on higher levels.

Note that [TKK14] is a significant extension of ideas already introduced in attachment [TK12] in deliverable 3.1.1. The work goes across WP 2.1, WP 2.3 and WP 3.1 since the interaction of feature extraction and learning as part of structural bootstrapping processes is investigated.

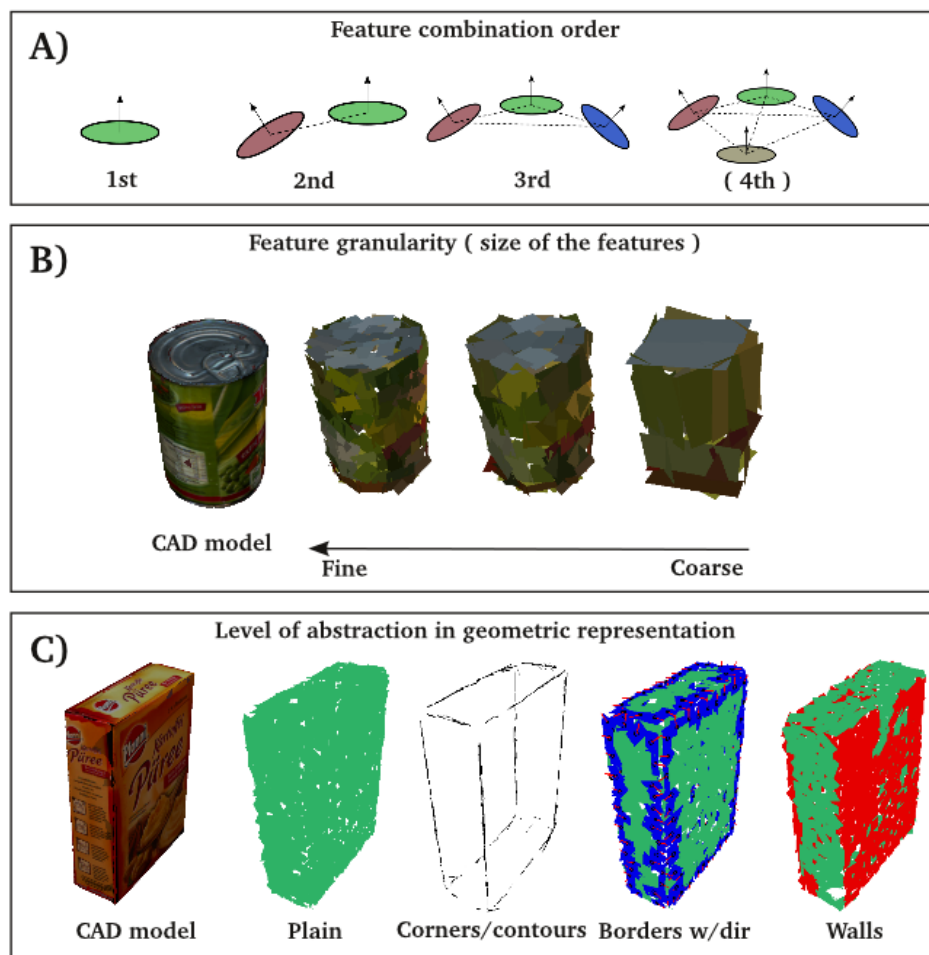


Figure 2.2: Different features varying in terms of A) order, B) granularity and C) semantic abstraction.

2.3 Grasping Unknown Objects Based on RGBD Images

In [GPTM13], we deal with the problem of grasping unknown objects. We use RGBD images to recover a point cloud representation of the object to be grasped (power grasp at the moment). Our method seeks object regions that match the curvature of the robots palm. The object point cloud is segmented in smooth surfaces. A score function measures the quality of the graspable points on the basis of the surface they belong to. A component of the score function is learned from experience and it is used to map the curvature of the object surfaces to the curvature of the robots hand. The user can further provide top-down information on the preferred grasping regions (e.g. a handle). We guarantee the feasibility of a chosen hand configuration by measuring its manipulability. We prove the effectiveness of the proposed approach by tasking the iCub to grasp a number of unknown real objects.

2.4 Object Models

2.4.1 Object Recognition and Pose Estimation in 2D Images

We developed a general framework for object recognition and pose estimation in 2D images, using 2D examples as training data (i.e. without any explicit 3D model). The training data therefore consists of a large number of annotated 2D pictures of the object of interest, observed from different viewpoints. We used a probabilistic approach for modeling both the training and the test data. Our formulation of the problem, as the maximization of the cross-correlation between the test view and the training images, departs from traditional approaches by not focusing on one specific type of image features. The proposed algorithm avoids relying on specific model-to-scene correspondences, allowing using similar-looking and generally unmatchable features. We effectively demonstrated this capability by applying the method to edge segments. The details of a first implementation were presented at DICTA 2012 [TP12a] and were already reported in Periodic Project Report of Y2. The proposed algorithm was designed to recover a complete description of the probability distribution of the pose of the object, in the 6 degree-of-freedom 3D pose space, thereby accounting for the inherent ambiguities in the 2D input data. This capability is of particular interest for our robotic tasks, where this ambiguity in the 3D pose of the object may be used in later stages of the overall application. We evaluated the capabilities of the method to perform pose estimation of known objects in cluttered images, and we obtained interesting results on both synthetic and real test images.

The overall framework has been extended to generalize the learned models to categories instead of instances; those recent developments have been presented in a follow-up paper [TP13a]. Enhanced performance has also been recently obtained by incorporating, into the visual object model, the actual deformations of the appearance between the discrete training viewpoints [TP13b].

We extended this method for the task of self-recognition of a robot arm [TSP13]. Here, models of the arm's links are learned or constructed individually. Then, a variant of the above method [TP13a] is used to produce pose hypotheses for all links, which are then jointly optimized using a message-passing scheme along the kinematic chain.

2.4.2 3D Reconstruction

We developed a method for performing 3D reconstructions of scenes or objects, using multiple calibrated 2D views thereof. Although considerable research has already been done on this fundamental topic over the years, we proposed an original approach to the problem. We used a probabilistic formulation of the problem in the 3D, reconstructed space that allows the use of features that cannot be matched one-to-one, or which cannot be precisely located, such as points along edges. The reconstructed scene, modelled as a probability distribution in the 3D space, is defined as the intersection of all reconstructions compatible with each available view. We introduce a method based on importance sampling to retrieve individual samples from that distribution, as well as an iterative method to identify contiguous regions of high density. This allows the reconstruction of continuous 3D curves compatible with all the given input views, without establishing specific correspondences and without relying on connectivity in the input images, while accounting for uncertainty in the input observations, due e.g. to noisy images and poorly calibrated cameras. The technical formulation is attractive in its flexibility and genericity. The implemented

system, evaluated on several very different publicly-available datasets, showed results competitive with existing methods, effectively dealing with arbitrary numbers of views, wide baselines and imprecise camera calibrations. This work [TP12b] was already mentioned in the Xperience Year-2 Periodic Progress Report.

2.4.3 3D Shape Pose Estimation

3D shape pose estimation is an essential component of robot perception. In our current stage, we estimate the pose of 3D shapes by using registration algorithms. Despite intensive study, 3D shape registration remains an open question. We developed a novel, highly efficient, robust and accurate 3D registration method for 3D point clouds, which is currently a popular and important data format in practice.

For most existing registration algorithms, there are many obstacles to practical use in fully autonomous systems. For instance, ICP usually requires manual assistance to provide a good initialization, and Gaussian Mixtures and SoftAssign are too expensive for real-time tasks. Therefore, our novel algorithm is developed to satisfy the practical need for better robustness and higher efficiency. Quite different from previous registration methods, instead of computing correspondence and alignment in 3D space, our algorithm first map points to a higher, possibly infinite-dimensional space by applying Kernel methods. Registration is subsequently performed within feature space by aligning corresponding principal components. The result is projected back into 3D pose space. The whole procedure is theoretically elegant and efficient. Kernel PCA is used to avoid explicit computation in feature space, and $SE(3)$ on-manifold optimization is employed to enhance the generality (flexible to any number of dimensions) of our registration algorithm. Empirical results demonstrate that our method [3] (attached to Deliverable D2.3.2) is quite accurate and robust to various challenging circumstances (e.g. large motion, presence of outliers), and remarkably, it is much faster than other state-of-the-art methods with comparable performance. Our immediate predecessor of this method [XSP13] used Gaussian mixtures and probability product kernel functions.

2.4.4 Reduced, dense 3D Scene representation of RGB-D images

3D from 2D is useful (see sections above) but may remain limited. For this reason we have introduced a novel, dense 3D representation called Super-Voxels for point cloud data, published in CVPR 2013 [PASW13]. The idea is based on the fact that unsupervised over-segmentation of an image into regions of perceptually similar pixels, known as superpixels, is a widely used preprocessing step in computer vision. Existing methods make use of projected color or depth information, but do not consider three dimensional geometric relationships between observed data points which can be used to prevent superpixels from crossing regions of empty space. To address this we have developed a novel over-segmentation algorithm which uses voxel relationships to produce over-segmentations which are fully consistent with the spatial geometry of the scene in three dimensional, rather than projective, space. This way consistent and much reduced 3D representations can be produced and use for all kinds of different tasks (e.g. pose estimation, object recognition, etc.). Additionally, as the algorithm works directly in 3D space, observations from several calibrated RGB+D cameras can be segmented jointly.

Chapter 3

Physical Interaction for Object Learning

In earlier work [2, 1] reported in the first two periods we have shown how humanoid robots can leverage their capability to physically interact with the world in order to support the autonomous visual segmentation, learning and grasping of unknown objects in a cluttered environment. To do so, the robot first generates object hypotheses based on its camera images, then pushes one of these possible objects and verifies or discards the hypothesis based on an analysis of its motion.

3.1 Learning Textured and Non-textured Object Representations

This work was however restricted to textured objects whose surface contains enough visually significant and unique points or regions that allow robust relocalization and estimation of the 3D motion that the object underwent. Following the same general idea, we were able to devise a more general approach that allows the interactive segmentation of textured as well as non-textured rigid objects.

Again, the first step is to generate object hypotheses based on the stereo images obtained from the cameras of the robot. Three different criteria are used for this step: Firstly, we look for regular geometric structures (planes, cylinders, spheres) amongst the 3D positions of Harris interest points, which is a strong indicator for underlying textured objects. Secondly, we look for larger unicolored regions which indicate a unicolored object or object part. Thirdly, we generate object hypotheses in image regions that are visually salient but not yet covered by hypotheses of the first two kinds.

One of these object hypotheses is selected based on being reachable and higher than its direct neighborhood, which indicates that it can probably be manipulated without collisions. The robot then tries to

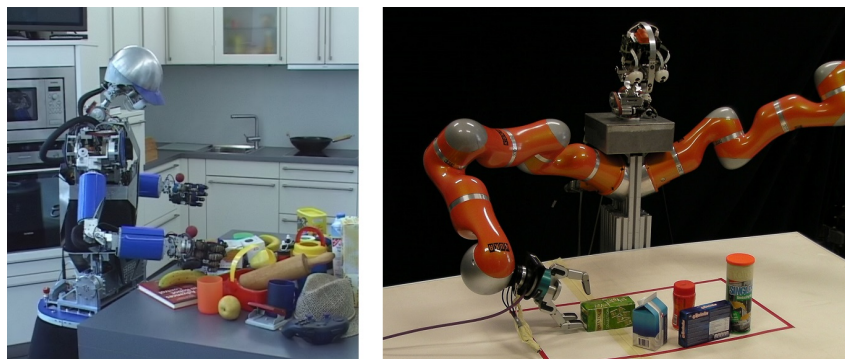


Figure 3.1: ARMAR-III and the JSI robot executing explorative pushes in order to segment unknown objects in slightly and very cluttered scenes.



Figure 3.2: Interactive segmentation of unknown objects in clutter: The left image in each row shows the initial object hypotheses, the following images show the segmentation results after one, two and three pushes respectively.

move this object with a careful push, using its force sensors to react to unexpected collisions with other objects or e.g. the table surface.

After the push, the object has to be re-localized. To this end, we generate a 3D point cloud from the images using dense stereo matching. Each point consists of its 3D position and color (this is usually called an RGBD point), and an object hypothesis is represented by the points lying in the region that it covers. We try to match the RGBD points belonging to each object hypothesis in the new point cloud obtained after the push. If the hypothesis can be matched well, has moved significantly and lies in an image region that changed due to the push, we consider it a verified object. With the same criteria we then verify or discard each individual point that belongs to the hypothesis, and add additional neighbouring points as candidates that are tested in the course of further pushes. After 2-3 pushes, the object hypotheses usually completely and correctly cover a real object. We showed that these segmentations allow to train a simple object recognizer that provides good recognition results [SUA14].

3.2 Acquiring Object Models by Foveated Vision

In another extension of this work we show how the advantages of foveated vision can be exploited to improve the learning of new object representations. The approach described in the attached paper [BU13] generates initial object hypotheses in the peripheral view. The initial hypotheses are more accurately investigated in the foveal view, which requires tightly integrated perception and motor control. Hypotheses are validated, corrected and extended after interactive manipulation of the object by a teacher or the robot itself. We compared different methods for validating the hypotheses in the foveal view and showed the advantages of foveal vision compared to the standard active stereo vision with a fixed field of view for object learning and recognition. A representation of accumulated features that is built through manipulation shows a particular advantage when an object is learned from several viewpoints.



Figure 3.3: A typical object learning / recognition procedure. The upper row respectively shows the peripheral and the bottom row the foveal view. The images in the first column contain the initial object hypotheses as the head is turned towards hypothesis "0". Each of the following columns shows the scene after moving the object, validating the initial hypothesis and accumulating the verified feature points for learning or recognition.

Chapter 4

Action Recognition and Object Categorization

In [FGMO13] we describe machine learning methods that can be applied to recognize actions performed by a human experimenter in front of a robot. We use RGBD images (either obtained through a Kinect device or from stereo cameras) and 3D optical flow (scene flow) to build visual features. We then select features using dictionary learning methods (sparsification). The main contribution of the paper is an effective real-time system for one-shot action modeling and recognition; the paper highlights the effectiveness of sparse coding techniques to represent 3D actions. We obtain very good results on three different data sets: a benchmark data set for one-shot action learning (the ChaLearn Gesture Data Set), an in-house data set acquired by a Kinect sensor including complex actions and gestures differing by small details, and a data set created for human-robot interaction purposes.

In [FNMO14], we extend the work presented in [FNMO14] to object recognition. Also in this case we use sparsity and a number of coding strategies to prepare data for classification. We follow the widely accepted coding-pooling recognition pipeline (often used as a model of biological vision). In this paper we show how to exploit ad-hoc representations in the coding and the pooling phases. We learn a dictionary for each object class and then use local descriptors encoded with the learned atoms to guide the pooling operator. We exhaustively evaluate the proposed approach in both single instance object recognition and object categorization problems. From the applications standpoint we consider a classical image retrieval scenario with the Caltech 101, as well as a typical robot vision task with data acquired by the iCub humanoid robot.

References

- [1] David Schiebener, Jun Morimoto, Tamim Asfour, and Aleš Ude. Integrating visual perception and manipulation for autonomous learning of object representations. *Adaptive Behavior*, 21(5):328–345, 2013.
- [2] Aleš Ude, David Schiebener, Hirokazu Sugimoto, and Jun Morimoto. Integrating visual processing and manipulation for autonomous learning of object representations. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1709–1715, Saint Paul, MN, 2012.
- [3] Hanchen Xiong, Sandor Szedmak, and Justus Piater. Efficient, General Point Cloud Registration With Kernel Feature Maps. In *Tenth Conference on Computer and Robot Vision*, pages 83–90, 5 2013.

Attached Articles

- [BU13] Robert Bevec and Aleš Ude. Object learning through interactive manipulation and foveated vision. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 234–239, Atlanta, Georgia, October 2013.
- [FGMO13] S.R. Fanello, I. Gori, G. Metta, and F. Odone. Keep it simple and sparse: Real-time action recognition. *Journal of Machine Learning Research*, pages 2617–2640, 2013.
- [FNMO14] S.R. Fanello, N. Noceti, G. Metta, and F. Odone. Dictionary based pooling for object categorization. In *International Conference on Computer Vision Theory and Applications (VIS-APP)*, Lisbon, Portugal, January 5–7, 2014.
- [GPTM13] I. Gori, U. Pattacini, V. Tikhanoff, and G. Metta. Ranking the good points: A comprehensive method for humanoid robots to grasp unknown objects. In *International Conference on Advanced Robotics*, Montevideo, Uruguay, November 25–29, 2013.
- [KJK⁺13] Norbert Krüger, Peter Janssen, Sinan Kalkan, Markus Lappe, Aleš Leonardis, Justus Piater, Antonio J. Rodríguez-Sánchez, and Laurenz Wiskott. Deep hierarchies in the primate visual cortex: What can we learn for computer vision? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1847–1871, 2013.
- [PASW13] Jeremie Papon, Alexey Abramov, Markus Schoeler, and Florentin Wörgötter. Voxel cloud connectivity segmentation - supervoxels for point clouds. In *CVPR*, pages 2027–2034, 2013.
- [REK⁺ed] Jimmy Alison Rytz, Lars-Peter Ellekilde, Dirk Kraft, Henrik Gordon Petersen, and Norbert Krüger. On transferability and contexts in data-driven grasping. *Robotica*, 2014 (submitted).
- [SUA14] D. Schiebener, A. Ude, and T. Asfour. Physical interaction for segmentation of unknown textured and non-textured rigid objects. In *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, 2014.
- [TKK14] Mikkel Tang Thomsen, Dirk Kraft, and Norbert Krüger. Identifying relevant feature-action associations for grasping unknown objects. Technical Report 2014–1, Cognitive and Applied Robotics Group, The Maersk Mc-Kinney Moller Institute, University of Southern Denmark, 2014.
- [TP12a] Damien Teney and Justus Piater. Generalized Exemplar-Based Full Pose Estimation from 2D Images without Correspondences. In *Digital Image Computing: Techniques and Applications*, 2012.
- [TP12b] Damien Teney and Justus Piater. Sampling-based Multiview Reconstruction without Correspondences for 3D Edges. In *3DimPVT*, 2012.
- [TP13a] Damien Teney and Justus Piater. Continuous Pose Estimation in 2D Images at Instance and Category Levels. In *Tenth Conference on Computer and Robot Vision*, pages 121–127, 5 2013.
- [TP13b] Damien Teney and Justus Piater. Modeling Pose/Appearance Relations for Improved Object Localization and Pose Estimation in 2D images. In *6th Iberian Conference on Pattern Recognition and Image Analysis*, volume 7887 of *LNCS*, pages 59–68, Berlin, Heidelberg, New York, 6 2013. Springer.
- [TSP13] Damien Teney, Dadhichi Shukla, and Justus Piater. Markerless Self-Recognition and Segmentation of Robotic Manipulator in Still Images. In *Mobile Manipulation Workshop on Interactive Perception*, 5 2013. Workshop at ICRA.

- [XSP13] Hanchen Xiong, Sandor Szedmak, and Justus Piater. A Study of Point Cloud Registration with Probability Product Kernel Functions. In *International Conference on 3D Vision*, pages 207–214, 6 2013.

Object Learning through Interactive Manipulation and Foveated Vision

Robert Bevec and Aleš Ude

Abstract—Autonomous robots that operate in unstructured environments must be able to seamlessly expand their knowledge base. To identify and manipulate previously unknown objects, a robot should continuously acquire new object knowledge even when no prior information about the objects or the environment is available. In this paper we propose to improve visual object learning and recognition by exploiting the advantages of foveated vision. The proposed approach first creates object hypotheses in peripheral stereo cameras. Next the robot directs its view towards one of the hypotheses to acquire images of the hypothetical object by foveal cameras. This enables a more thorough investigation of a smaller area of the scene, which is seen in higher resolution. Additional information that is needed to verify the hypothesis comes through interactive manipulation. A teacher or the robot itself induces a change in the scene by manipulating the hypothetical object. We compare two methods for validating the hypotheses in the foveal view and experimentally show the advantage of foveated vision compared to standard active stereo vision that relies on camera systems with a fixed field of view.

I. INTRODUCTION

To be able to successfully work in unstructured and uncontrolled environments, autonomous robots must have the ability to expand their library of known objects. Such robots must therefore be able to detect and learn new objects when no prior knowledge about them and the environment is available. Segmenting objects using only visual information has proved very difficult [1], [2]. However, perturbing the scene by for example pushing a hypothetical object introduces additional information that makes this task more feasible [3], [4], [5], [6], [7], [8], [9], [10], [11], [12].

In this paper we propose to improve object recognition in autonomous robots by learning and recognizing objects using foveated vision. In biological systems, the fovea is a part of the retina with a very high density of cone cells. It is responsible for color vision and color sensitivity. The density of cones slowly decreases toward the peripheral part of the retina. This layout provides sharp central vision and a relatively low average resolution over the entire field of view, therefore reducing the need for computational resources, but still achieving high precision vision in the fovea. Foveated stereo vision in robots can be accomplished using two cameras per eye with different focal lengths [13], [14],

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience, and from the Slovenian Research Agency under grant agreement no J2-4284.

R. Bevec and A. Ude are with Jožef Stefan Institute, Department of Automatics, Biocybernetics and Robotics, Humanoid and Cognitive Robotics Lab, Jamova 39, Ljubljana, Slovenia robert.bevec@ijs.si, ales.ude@ijs.si

[15], [16]. This arrangement enables capturing wide-angle peripheral and narrow-angle foveal images at the same time, but requires gaze control in order to acquire the area of interest in the foveal view. A practical advantage of such an arrangement is that a robot can simultaneously analyze the wide field of view of peripheral cameras – where it is easier to find and track objects – and the narrower field of view of foveal cameras – where objects images have higher resolution and are therefore more suitable for recognition.

Some of the recently proposed methods rely on accurate depth sensors to segment objects from the scene [7], [9], [17], [11], [12], [18]. We chose to rely solely on stereo foveated vision to learn object representations (Fig. 1) because such systems are more generally applicable and are also closer to human vision and depth perception.

In our previous work [10], [19] the robot learns and recognizes objects using standard active stereo vision. It generates hypotheses about the existence of objects and tries pushing them to look for changes in the scene and validate the object hypotheses. Object representations were obtained by accumulating the confirmed features over several snapshots and a bag-of-features type models [20] have been acquired. Here we propose to extend this object learning process by making use of foveated vision, thus adding the confirmed object features acquired in the foveal view. In our current experiments, the manipulation of objects has been realized through human interaction. However, the robot



Fig. 1. Karlsruhe Humanoid Head [13] and the object test set used in the experiments. The head is equipped with two cameras in each eye. One pair of cameras models human peripheral vision, the other pair foveal vision.

could also use its own manipulation capabilities to achieve the same result. The developed approach requires no prior knowledge about the objects or the environment and retains the assumptions that the objects contain some distinctive visual features and move as rigid bodies.

II. OVERVIEW

The following procedure is applied to learn new objects and generate their representations for recognition:

- **Generate object hypotheses in peripheral view:** Find smooth surfaces in the point cloud of stereo matched visual features.
- **Turn the head and eyes toward one hypothesis:** The centroid of the hypothesis should lie in the middle of the foveal images.
- **Generate an object hypothesis in foveal view:** The object takes up a large portion of the foveal images, therefore all visual features represent a hypothesis.
- **Generate data for hypothesis verification:** The robot requests a human teacher to move the object and validates which features belong to the object due to the resulting change in the scene. Additional features are added if they move concurrently with the object.
- **Turn the head and eyes toward the confirmed hypothesis:** The centroid of the manipulated object should lie in the middle of foveal images.
- **Validate the hypothesis in foveal view:** The robot validates which features belong to the object due to the resulting change in the scene.
- **Feature accumulation:** The last three steps above can be repeated several times to accumulate object features from different viewpoints.

Note that in this procedure, the object manipulation step by a human teacher could be replaced by robot manipulation as in our other work [10].

III. OBJECT HYPOTHESES

A. Peripheral view

Object hypotheses in the peripheral view are created within a point cloud generated by the peripheral stereo vision. Initial point correspondences are found by matching Harris interest points [21] and maximally stable extremal regions (MSER) [22] in each eye. The Harris interest points are found mostly on textured parts of the image. The MSER detector balances that by finding salient points in areas with less texture. The correct correspondences and precise 3-D point positions are obtained by using epipolar geometry and stereo calibration on an active camera system [23]. At each 3-D interest point a SIFT feature descriptor [24] is calculated, which has shown robustness to scale, rotation, translation and illumination changes.

The hypotheses are created by searching for smooth surface patches within this point cloud. As in our previous work [10], [19], the robot looks for planar, spherical and cylindrical surface patches using RANSAC [25]. This iterative, nondeterministic model fitting algorithm chooses a random subset of points, fits models of the possible surface

types and returns the parameters of the fitted surface that includes the largest number of points within a tolerance of that surface out of the entire point cloud. All of the points belonging to the fitted surface are then excluded from the point cloud and a search for a new hypothesis is started again. When no good fits are found anymore, the remaining points are clustered into groups of points lying close to each other using X-means algorithm [26]. Each of these clusters also represents a hypothesis if it retains enough points and has a sufficient point per volume ratio. This allows the robot to create a hypothesis for an object, even if no part of that object corresponds to a smooth surface.

In order to prevent surface hypotheses spanning over several objects, X-means clustering is applied to each hypothesis. The hypothesis is divided into several hypotheses if that is deemed appropriate by the algorithm. Erroneous splitting of hypotheses is not a problem since all features that move concurrently after interactive manipulation are later joined together as a validated object. An example of initial object hypotheses in the peripheral view is seen in Fig. 2. A detailed description about the detection of planar, spherical and cylindrical surface patches is given in [19].

B. Foveal view

Since a foveal image covers a much narrower field of view, the object of interest will cover a larger portion of the foveal than peripheral image. We can therefore assume that there is no need to search for smooth surface patches, like in the peripheral views, to create hypotheses. Instead, the entire point cloud is considered as an object hypothesis. The foveal camera pair naturally requires its own camera calibration to find interest point correspondences and calculate the 3-D position of points. An example object hypothesis in the foveal view is seen in Fig. 2.

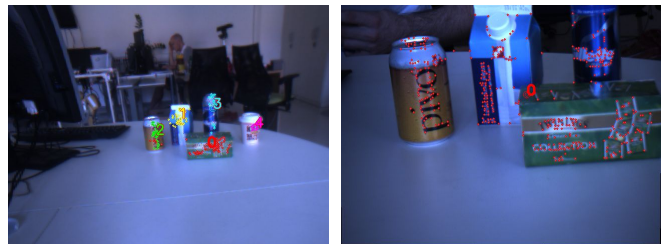


Fig. 2. Initial hypotheses in a typical scene with household objects. In the peripheral view, each hypothesis is represented by points of the same color. After the head has turned towards hypothesis "0", we see the initial hypothesis in the foveal view, where a hypothesis consists of all features.

IV. GAZE CONTROL

In order to acquire the object hypothesis in the foveal view, the robot must rotate the head and eyes so that the center of the hypothesis appears in the foveal cameras. Hypothesis i is first created in the peripheral view. It contains N_i points corresponding to a smooth surface patch. The 3-D centroid \mathbf{P}_i of the hypothesis in the peripheral view is calculated as follows

$$\mathbf{P}_i = \frac{\sum_{n=1}^{N_i} \mathbf{x}_n}{N_i} \quad (1)$$



Fig. 3. A typical object learning / recognition procedure. The upper row respectively shows the peripheral and the bottom row the foveal view. The images in the first column contain the initial object hypotheses as the head is turned towards hypothesis "0". Each of the following columns shows the scene after moving the object, validating the initial hypothesis and accumulating the verified feature points for learning or recognition.

Using direct kinematics equations, the robot calculates the positions of all the hypotheses in the global coordinate system. It then uses a virtual mechanism approach to calculate the proper joint configuration to center the chosen hypothesis in the foveal cameras and moves the head and eyes accordingly [27].

V. OBJECT VALIDATION

After the robot has identified the object hypotheses and turned its view towards one of them, it needs additional information to validate or discard the hypothesis. This additional information is provided by inducing motion on the object. Changes in the scene can then be analyzed for simultaneous feature motion, which is a very strong indicator of object existence in the object definition given by Gibson [28]. In our system, the robot requests a human to move the object hypothesis the robot is looking at. This is indicated by displaying the hypothetical object feature points in the acquired stereo image pair. Alternatively, the robot could also use its own manipulation capabilities to try and push the object hypothesis or perform some other manipulation. A typical learning or recognition procedure is seen in Fig. 3, where the initial hypotheses in the first column are validated after changes in the scene in each of the following columns.

After the manipulation action is completed, the robot recomputes the point cloud using Harris interest points and MSERs as described in Section III.

A. Peripheral view

Our basic assumption is that the object moves as a rigid body. The feature points contained in the hypothesis are matched in the peripheral images after the change using a SIFT descriptor. Due to occlusions or large rotations, some of the features may not be matched at all. If enough matches are found, we can test our assumption and find a rigid body motion that corresponds to the motion induced by manipulation. Since there will be false feature matches and

matches of points that didn't belong to the object in the first place, we use the RANSAC algorithm for finding the most probable object rotation and translation, thereby filtering out all feature correspondence matches not within the tolerance of the transformation. The parameters of the transformation can be estimated from three pairs of points before and after the change. Let \mathbf{x}_n be the position of a feature point before the change and \mathbf{x}'_n the position of the matched point after. If the new feature position corresponds to the transformation

$$\mathbf{x}'_n = \mathbf{R} \cdot \mathbf{x}_n + \mathbf{t}, \quad (2)$$

where \mathbf{R} is the rotation matrix and \mathbf{t} is the translation vector, it moved concurrently with this transformation. If more than a minimum amount of features correspond to this transformation, the hypothesis is considered as validated. All of the features that move according to the estimated rigid body transformation are considered confirmed object features. In the following we call this process a rigid body motion filter (RBMF).

All other feature matches from the point cloud, i.e. features from the point cloud that do not belong to the initial hypothesis, are also checked regarding the estimated rigid body transformation. If they move as the object features, these features are added as candidate features of the validated hypothesis. If they are matched and move together with the object also after the next manipulation, they are considered confirmed. The first row in Fig. 3 shows an example of successful object validation through several manipulations.

Even though in this paper we propose learning based on foveal views, it is still necessary to track the motion of the hypotheses in peripheral view as well. This is due to the fact that after manipulation the object usually disappears from the foveal view. The only way to get it back into the foveal view is to perform a saccade towards the new object position, which can be done by estimating the new object pose in the peripheral view.

B. Foveal view

The same process of hypothesis validation described for the peripheral views can be used in the foveal view as well. After the successful validation of the hypothesis in the peripheral view, the head and eyes are turned toward the hypothesis' centroid. The new point cloud is computed in the foveal view and matched to the one before the change using SIFT descriptors. Matches are verified with the rigid body motion filter described in Section V-A and the object hypothesis is validated or discarded. The second row in Fig. 3 shows an example of successful object validation through several manipulations.

We also suggest a simpler solution for use in the foveal views to reduce the computational complexity of the validation procedure. Since RANSAC is a statistical method, it needs many repetitions in order to provide a good solution with high certainty [25]. Being a nondeterministic algorithm, it does not guarantee the best solution even after an arbitrary number of repetitions. Instead, we propose to make use of the known movement of the head and eyes and assume that the surroundings of the hypothesis does not move when the human moves the object. We can therefore filter all static features in the peripheral views, i.e. features that moved according to the motion of the head-eye system, and confirm all other features as features belonging to the object. The assumption of static surrounding is much more justified in foveal than peripheral views because foveal views contain only a small portion of the scene.

Let \mathbf{x}_n be the position of a feature point before the change and \mathbf{x}'_n the position of the matched feature point after manipulation and head-eye rotation. Let \mathbf{R}_V be the rotation matrix and \mathbf{t}_V the translation vector describing the change of viewpoint from the previous gaze direction. We define threshold ϵ as a minimum displacement that implies motion. If statement (3) is true, the feature point has moved and belongs to the validated hypothesis:

$$\|\mathbf{x}'_n - \mathbf{R}_V \cdot \mathbf{x}_n + \mathbf{t}_V\| \geq \epsilon. \quad (3)$$

We call this method a static feature filter (SFF). Although SFF requires a partially static scene, it provides a valid alternative to the first approach. In Fig. 4 we can see how successful these two methods are at filtering feature matches and validating the hypotheses. In these examples, the initial object hypotheses in foveal views (recall that our assumption is that all feature points detected in foveal views belong to the object) contained a large number of features found in the background. The first row shows validation of the initial hypothesis with rigid body motion filter (RBMF) and the second row validation with the static feature filter (SFF). Both approaches succeeded in eliminating most of the spurious features, although SFF failed to discard a small number of false matches on the box in the background.

VI. OBJECT LEARNING AND RECOGNITION

As in our previous work, the visual appearance of objects is learned using a bag of features (BoF) model [20]. Firstly, a visual vocabulary is created by clustering SIFT feature



Fig. 4. In the first row, the initial hypothesis in the foveal view is validated with the rigid body motion filter and in the second row with the static feature filter. The rigid body motion filter discards all the background features and all false matches. The static feature filter requires a static background and cannot filter false SIFT matches.

descriptors extracted from training images. To compute the vocabulary, we use the SIFT feature descriptors extracted while learning different objects from different viewpoints. To represent an object, each SIFT descriptor that is confirmed to belong to the object in the current view, is matched with the closest descriptor in the vocabulary. A histogram of descriptors from the vocabulary corresponding to object features is built and later used for recognition.

To include color information, the robot also calculates a saturation-weighted hue histogram [29] within the ellipse spanned by the principal axes of the confirmed features. Both the BoF and hue histograms are calculated for individual viewpoints and for the accumulated representation after each successful validation in foveal and peripheral views. Combined, the histograms represent global color information and descriptors of salient area in all relevant views.

Object recognition is realized by calculating the corresponding histograms of the initial or validated object hypothesis as described in the previous paragraphs. A k-nearest neighbors decision is based on the distance measure between known objects in the database and the hypothesis. The distance measure is a weighted sum of normalized χ^2 histogram distances of the BoF and hue histograms as described in [10].

Some examples of object recognition for the initial hypothesis in the foveal view are shown in Fig. 5. Objects rich with features can be quite successfully recognized in this stage already, even though there are a lot of features belonging to the background included in the hypothesis (upper left image). Smaller objects with fewer features are sometimes classified as incorrect objects (unknown object in the background, upper right image) or as the object in the background itself (lower left image). When initial hypotheses do not include many features from the background, they are recognized rather successfully as shown later in the experimental evaluation (lower right image).



Fig. 5. Recognition of initial hypotheses in the foveal view depends greatly on the amount of features in the background. In the upper left image we have an unknown object in the background, but the object in the foreground is correctly recognized, since it is very rich with features. In the upper right image the object in the foreground has much fewer features and is therefore incorrectly recognized. For the same reason, the object in the lower left image is falsely recognized as the known object in the background. If there are few features in the background, as seen in the lower right image, the recognition of objects in the initial hypotheses can reach 93%.

VII. EXPERIMENTAL EVALUATION

We performed several experiments to evaluate the gain of using foveated vision for object learning and recognition. We also compared the proposed filters (RBMF and SFF) for validating hypotheses in the foveal images.

The robot first learned representations of 20 different typical household objects (Fig. 1), where a human teacher manipulated the objects. The objects were placed on a table in sets of 5 at a distance approximately 1 meter from the robot, as shown in Fig. 2. Using KUKA LWR manipulators, this distance would be well within reach of the robot if pushing was done autonomously. There were some occlusions present at times, but eventually each object was shown in full extent. The system created initial hypotheses about the objects and then learned a model for each. Each model was learned through 6 consecutive manipulations. The human teacher (the first author) moved the objects mainly laterally to the image plane with small rotations. This ensured good feature matching, but the objects were learned mainly from one viewpoint. For the foveal view different representations were learned, once using RBMF and once SFF.

For recognition, the sets of objects were randomly mixed and placed back on the table. The robot tried recognizing the initial hypotheses and then requested the human to move the object it was facing. It followed the object through 3 consecutive manipulations in the scene and tried recognizing it after each one. Table I shows the results of object recognition.

The results of recognition in peripheral views are significantly worse than in our previous work [10], where we used the same camera pair as a standard active stereo vision system. This is due to the increased distance of the objects from the robot. In our experiments the objects were placed

TABLE I

OBJECT RECOGNITION RATE IN THE PERIPHERAL VIEW, FOVEAL VIEW WITH RIGID BODY MOTION FILTER (RBMF) AND FOVEAL VIEW WITH STATIC FEATURE FILTER (SFF) FOR THE INITIAL HYPOTHESES AND THE FOLLOWING PUSHES.

	init. hyp.	1 push	2 pushes	3 pushes
Peripheral	51 %	52 %	60 %	62 %
Foveal w. RBMF	60 %	95 %	95 %	95 %
Foveal w. SFF	60 %	100 %	100 %	100 %

approximately 50 cm further away from the cameras. The recognition rate improves with each push, until it starts to converge toward approximately 65 % and doesn't improve even after more pushes. At such a distance few object features were found and even fewer matched. On average, only 32 features were accumulated in the recognition stage, describing each object after 3 pushes. A larger number of features allows for more robust recognition under partial occlusion in cluttered scenes [24].

Recognition rate using foveated vision varied a lot in the initial hypothesis. Depending on the amount of clutter in the view, features of the hypothesis might belong to the background. Singulated objects were correctly recognized 93 % of the time, while dense clutter reduced this rate down to 27 %. On average, the initial hypothesis recognition rate in our experiments was 60 %. Using the proposed approach, background features were filtered out after the first push and the recognition rate improved significantly.

Both of the proposed filters used the same initial hypotheses. Using RBMF, the validated hypothesis after the first push was successfully recognized 95 % of the time. This rate stayed stable with consequent pushes. It turned out that there was just one particular object, which was constantly recognized as another object from the database. These recognition rates are significantly higher compared to the rates in the peripheral view. On average, 181 features described each object after 3 pushes, which is significantly more than in the peripheral view.

Using SFF, the validated hypothesis after the first push was successfully recognized 100% of the time. This rate remained stable throughout the interactive recognition process. SFF is not able to discard false descriptor matches and therefore builds a richer representation than RBMF including false positives. On average, 227 points described each object after 3 pushes, but as it turns out, a richer representation including some false descriptor matches does not hurt the recognition rates, since the proportion of false features is low. Some of them were actually on the object itself and therefore benefited the representation. The results prove that both methods are valid options for hypothesis validation in foveal views. Overall, our results confirm the usefulness of foveal vision for object learning and recognition.

VIII. CONCLUSIONS

We developed a novel system for object learning and recognition by manipulation, which can exploit the advantages of foveal vision. Initial object hypotheses are generated



Fig. 6. Our method does not require a static tabletop scene. The system is able to learn new objects or recognize known objects in an arbitrary environment. In the pictures above, we can see the object learning through human interaction.

in the peripheral view and more accurately investigated in the foveal view by turning the head and eyes toward the hypothesis. Hypotheses are validated, corrected and extended after interactive manipulation by a teacher or the robot itself. We compared different methods for validating the hypotheses in the foveal view and showed the advantages of foveal vision compared to the standard active stereo vision with a fixed field of view for object learning and recognition.

A representation of accumulated features that is built through manipulation shows a particular advantage when an object is learned from several viewpoints. As it is evident in Fig. 6, our methods works in an arbitrary environment in cooperation with a human teacher and relies on only two assumptions: that the object moves as a rigid body and that it has distinctive visual features.

REFERENCES

- [1] P. Fitzpatrick and G. Metta, "Grounding vision through experimental manipulation," *Philosophical Transactions of the Royal Society A*, vol. 361, pp. 2165–2185, Oct. 2003.
- [2] G. Kootstra, J. Ypma, and B. de Boer, "Active exploration and keypoint clustering for object recognition," in *IEEE International Conference on Robotics and Automation*, (Pasadena, CA), pp. 1005–1010, 2008.
- [3] P. Fitzpatrick, "First contact: an active vision approach to segmentation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Las Vegas, Nevada), pp. 2161–2166, 2003.
- [4] J. Kenney, T. Buckley, and O. Brock, "Interactive segmentation for manipulation in unstructured environments," in *IEEE International Conference on Robotics and Automation*, (Kobe, Japan), pp. 1377–1382, 2009.
- [5] E. Stergaršek Kuzmič and A. Ude, "Object segmentation and learning through feature grouping and manipulation," in *2010 10th IEEE-RAS International Conference on Humanoid Robots*, (Nashville, Tennessee), pp. 371–378, 2010.
- [6] W. H. Li and L. Kleeman, "Segmentation and modeling of visually symmetric objects by robot actions," *The International Journal of Robotics Research*, vol. 30, no. 9, pp. 1124–1142, 2011.
- [7] T. Hermans, J. M. Rehg, and A. Bobick, "Guided pushing for object singulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Vilamoura, Portugal), pp. 4783–4790, 2012.
- [8] M. Rudinac, G. Kootstra, D. Kragic, and P. P. Jonker, "Learning and recognition of objects inspired by early cognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Vilamoura, Algarve, Portugal), pp. 4177–4184, 2012.

- [9] L. Chang, J. R. Smith, and D. Fox, "Interactive singulation of objects from a pile," in *IEEE International Conference on Robotics and Automation*, (St. Paul, Minnesota), pp. 3875–3882, 2012.
- [10] A. Ude, D. Schiebener, N. Sugimoto, and J. Morimoto, "Integrating surface-based hypotheses and manipulation for autonomous segmentation and learning of object representations," in *IEEE International Conference on Robotics and Automation*, (St. Paul, Minnesota), pp. 1709–1715, 2012.
- [11] D. Katz, M. Kazemi, J. A. Bagnell, and A. Stentz, "Clearing a Pile of Unknown Objects using Interactive Perception," in *IEEE International Conference on Robotics and Automation*, (Karlsruhe, Germany), pp. 154–161, 2013.
- [12] D. Katz, M. Kazemi, J. A. Bagnell, and A. Stentz, "Interactive Segmentation, Tracking, and Kinematic Modeling of Unknown 3D Articulated Objects," in *IEEE International Conference on Robotics and Automation*, (Karlsruhe, Germany), pp. 4988–4995, 2013.
- [13] T. Asfour, K. Welke, P. Azad, A. Ude, and R. Dillmann, "The Karlsruhe Humanoid Head," in *2008 8th IEEE-RAS International Conference on Humanoid Robots*, (Daejeon, Korea), pp. 447–453, 2008.
- [14] C. G. Atkeson, J. G. Hale, F. Pollick, M. Riley, S. Kotosaka, S. Schaul, T. Shibata, G. Tevatia, A. Ude, S. Vijayakumar, E. Kawato, and M. Kawato, "Using humanoid robots to study human behavior," *IEEE Intelligent Systems and their Applications*, vol. 15, no. 4, pp. 46–56, 2000.
- [15] H. Kozima and H. Yano, "A robot that learns to communicate with human caregivers," in *Proceedings of the First International Workshop on Epigenetic Robotics*, (Lund, Sweden), pp. 47–52, 2001.
- [16] T. Shibata, S. Vijayakumar, J. Conradt, and S. Schaal, "Biomimetic oculomotor control," *Adaptive Behavior*, vol. 9, pp. 189–207, 2001.
- [17] A. Aldoma, F. Tombari, J. Prankl, A. Richtsfeld, L. D. Stefano, and M. Vincze, "Multimodal Cue Integration through Hypotheses Verification for RGB-D Object Recognition and 6DOF Pose Estimation," in *IEEE International Conference on Robotics and Automation*, (Karlsruhe, Germany), pp. 2096–2103, 2013.
- [18] I. Lysenkov and V. Rabaud, "Pose Estimation of Rigid Transparent Objects in Transparent Clutter," in *IEEE International Conference on Robotics and Automation*, (Karlsruhe, Germany), pp. 162–169, 2013.
- [19] D. Schiebener, J. Morimoto, T. Asfour, and A. Ude, "Integrating visual perception and manipulation for autonomous learning of object representations," *Adaptive Behavior*, vol. 21, no. 5, 2013.
- [20] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *ECCV Workshop on statistical learning in computer vision*, (Prague, Czech Republic), 2004.
- [21] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the Fourth Alvey Vision Conference*, (Manchester, UK), pp. 147–151, 1988.
- [22] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, 2004.
- [23] A. Ude and E. Oztop, "Active 3-D vision on a humanoid head," in *2009 International Conference on Advanced Robotics (ICAR)*, (Munich, Germany), pp. 1–6, 2009.
- [24] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [25] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.
- [26] D. Pelleg, A. Moore, and Others, "X-means: Extending k-means with efficient estimation of the number of clusters," in *Proceedings of the Seventeenth International Conference on Machine Learning*, (Stanford, California), pp. 727–734, 2000.
- [27] D. Omrčen and A. Ude, "Redundancy control of a humanoid head for foveation and three-dimensional object tracking: A virtual mechanism approach," *Advanced Robotics*, vol. 24, no. 15, pp. 2171–2197, 2010.
- [28] J. J. Gibson, "The Ecological Approach to the Visual Perception of Pictures," *Leonardo*, vol. 11, no. 3, pp. 227–235, 1978.
- [29] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek, "Evaluating color descriptors for object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1582–1596, 2010.

Keep It Simple And Sparse: Real-Time Action Recognition

Sean Ryan Fanello*

Ilaria Gori*

Giorgio Metta

iCub Facility

Istituto Italiano di Tecnologia

Genova, Via Morego 30, 16163, Italia

SEAN.FANELLO@IIT.IT

ILARIA.GORI@IIT.IT

GIORGIO.METTA@IIT.IT

Francesca Odone

Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi

Università degli Studi di Genova

Genova, Via Dodecaneso 35, 16146, Italia

FRANCESCA.ODONE@UNIGE.IT

Editors: Isabelle Guyon and Vassilis Athitsos

Abstract

Sparsity has been showed to be one of the most important properties for visual recognition purposes. In this paper we show that sparse representation plays a fundamental role in achieving one-shot learning and real-time recognition of actions. We start off from RGBD images, combine motion and appearance cues and extract state-of-the-art features in a computationally efficient way. The proposed method relies on descriptors based on 3D Histograms of Scene Flow (3DHOFs) and Global Histograms of Oriented Gradient (GHOGs); adaptive sparse coding is applied to capture high-level patterns from data. We then propose a simultaneous on-line video segmentation and recognition of actions using linear SVMs. The main contribution of the paper is an effective real-time system for one-shot action modeling and recognition; the paper highlights the effectiveness of sparse coding techniques to represent 3D actions. We obtain very good results on three different data sets: a benchmark data set for one-shot action learning (the ChaLearn Gesture Data Set), an in-house data set acquired by a Kinect sensor including complex actions and gestures differing by small details, and a data set created for human-robot interaction purposes. Finally we demonstrate that our system is effective also in a human-robot interaction setting and propose a memory game, "All Gestures You Can", to be played against a humanoid robot.

Keywords: real-time action recognition, sparse representation, one-shot action learning, human robot interaction

1. Introduction

Action recognition as a general problem is a very fertile research theme due to its strong applicability in several real world domains, ranging from video-surveillance to content-based video retrieval and video classification. This paper refers specifically to action recognition in the context of Human-Machine Interaction (HMI), and therefore it focuses on whole-body actions performed by a human who is standing at a short distance from the sensor.

Imagine a system capable of understanding when to turn the TV on, or when to switch the lights off on the basis of a gesture; the main requirement of such a system is an easy and fast learn-

*. S.R. Fanello and I. Gori contributed equally to this work.

ing and recognition procedure. Ideally, a single demonstration suffices to teach the system a new gesture. More importantly, gestures are powerful tools, through which languages can be built. In this regard, developing a system able to communicate with deaf people, or to understand paralyzed patients, would represent a great advance, with impact on the quality of life of impaired people. Nowadays these scenarios are likely as a result of the spread of imaging technologies providing real-time depth information at consumer's price (as for example the Kinect (Shotton et al., 2011) by Microsoft); these depth-based sensors are drastically changing the field of action recognition, enabling the achievement of high performance using fast algorithms.

Following this recent trend we propose a *complete system based on RGBD video sequences*, which models actions *from one example only*. Our main goal is to recognize actions in real-time with high accuracy; for this reason we design our system accounting for good performance as well as low computational complexity. The method we propose can be summarized as follows: after segmentation of the moving actor, we extract two types of features from each image, namely, Global Histograms of Oriented Gradient (GHOGs) to model the shape of the silhouette, and 3D Histograms of Flow (3DHOFs) to describe motion information. We then apply a sparse coding stage, which allows us to take care of noise and redundant information and produces a compact and stable representation of the image content. Subsequently, we summarize the action within adjacent frames by building feature vectors that describe the feature evolution over time. Finally, we train a Support Vector Machine (SVM) for each action class.

Our framework can segment and recognize actions accurately and in real-time, even though they are performed in different environments, at different speeds, or combined in sequences of multiple actions. Furthermore, thanks to the simultaneous appearance and motion description complemented by the sparse coding stage, the method provides a one-shot learning procedure. These functions are shown on three different experimental settings: a benchmark data set for one-shot action learning (the ChaLearn Gesture Data Set), an in-house data set acquired by a Kinect sensor including complex actions and gestures differing by small details, and an implementation of the method on a humanoid robot interacting with humans.

In order to demonstrate that our system can be efficiently engaged in real world scenarios, we developed a real-time memory game against a humanoid robot, called "All Gestures You Can" (Gori et al., 2012). Our objective in designing this interaction game is to stress the effectiveness of our gesture recognition system in complex and uncontrolled settings. Nevertheless, our long term goal is to consider more general contexts, which are beyond the game itself, such as rehabilitation and human assistance. Our game may be used also with children with memory impairment, for instance the Attention Deficit/Hyperactivity Disorder (ADHD) (Comoldi et al., 1999). These children cannot memorize items under different conditions, and have low performances during implicit and explicit memory tests (Burden and Mitchell, 2005). Interestingly, Comoldi et al. (1999) shows that when ADHD children were assisted in the use of an appropriate strategy, they performed the memory task as well as controls. The game proposed in this paper could be therefore used to train memory skills to children with attention problems, using the robot as main assistant. The interaction with the robot may increase their motivation to maintain attention and help with the construction of a correct strategy.

The paper is organized as follows: in Section 2 we briefly review the state of the art. In Section 3 sparse representation is presented; Section 4 describes the complete modeling and recognition pipeline. Section 5 validates the approach in different scenarios; Section 6 shows a real application

in the context of Human Robot Interaction (HRI). Finally, Section 7, presents future directions and possible improvements of the current implementation.

2. Related Work

The recent literature is rich of algorithms for gesture, action, and activity recognition—we refer the reader to Aggarwal and Ryoo (2011) and Poppe (2010) for a complete survey of the topic. Even though many theoretically sound, good performing and original algorithms have been proposed, to the best of our knowledge, none of them fulfills at the same time *real-time*, *one-shot learning* and *high accuracy* requirements, although such requirements are all equally important in real world application scenarios.

Gesture recognition algorithms differ in many aspects. A first classification may be done with respect to the overall structure of the adopted framework, that is, how the recognition problem is modeled. In particular, some approaches are based on machine learning techniques, where each action is described as a complex structure; in this class we find methods based on Hidden Markov Models (Malgireddy et al., 2012), Coupled Hidden Semi-Markov models (Natarajan and Nevatia, 2007), action graphs (Li et al., 2010) or Conditional Markov Fields (Chatzis et al., 2013). Other methods are based on matching: the recognition of actions is carried out through a similarity match with all the available data, and the most similar datum dictates the estimated class (Seo and Milanfar, 2012; Mahbub et al., 2011).

The two approaches are different in many ways. Machine learning methods tend to be more robust to intra-class variations, since they distill a model from different instances of the same gesture, while matching methods are more versatile and adapt more easily to one-shot learning, since they do not require a batch training procedure. From the point of view of data representation, the first class of methods usually extracts features from each frame, whereas matching-based methods try to summarize all information extracted from a video in a single feature vector. A recent and prototypical example of machine learning method can be found in Malgireddy et al. (2012), which proposes to extract local features (Histograms of Flow and Histograms of Oriented Gradient) on each frame and apply a bag-of-words step to obtain a global description of the frame. Each action is then modeled as a multi channel Hidden Markov Model (mcHMM). Although the presented algorithm leads to very good classification performance, it requires a computationally expensive offline learning phase that cannot be used in real-time for one-shot learning of new actions. Among the matching-based approaches, Seo and Milanfar (2012) is particularly interesting: the algorithm extract a new type of features, referred to as *3D LSKs*, from space-time regression kernels, particularly appropriate to identify the spatio-temporal geometric structure of the action; it then adopts the Matrix Cosine Similarity measure (Shneider and Borlund, 2007) to perform a robust matching. Another recent method following the trend of matching-based action recognition algorithms is Mahbub et al. (2011); in this work the main features are standard deviation on depth (STD), Motion History Image (MHI) (Bobick and Davis, 2001) and a 2D Fourier Transformation in order to map all information in the frequency domain. This procedure shows some benefits, for instance the invariance to camera shifts. For the matching step, a simple and standard correlation measure is employed. Considering this taxonomy, the work we propose falls within the machine learning approaches, but addresses specifically the problem of one-shot learning. To this end we leverage on the richness of the video signal used as a training example and on a dictionary learning approach to obtain an effective and distinctive representation of the action.

An alternative to classifying gesture recognition algorithms is based on the data representation of gesture models. In this respect there is a predominance of features computed on local areas of single frames (local features), but also holistic features are often used on the whole image or on a region of interest. Among the most known methods, it is worth mentioning the spatio-temporal interesting points (Laptev and Lindeberg, 2003), spatio-temporal Hessian matrices (Willems et al., 2008), Gabor Filters (Bregonzio et al., 2009), Histograms of Flow (Fanello et al., 2010), Histograms of Oriented Gradient (Malgireddy et al., 2012), semi-local features (Wang et al., 2012), combination of multiple features (Laptev et al., 2008), Motion History Image (MHI) (Bobick and Davis, 2001), Space-Time shapes (Gorelick et al., 2007), Self-Similarity Matrices (Efros et al., 2003). Also, due to the recent diffusion of real-time 3D vision technology, 3D features have been recently employed (Gori et al., 2012). For computational reasons as well as the necessity of specific invariance properties, we adopt global descriptors, computed on a region of interest obtained through motion segmentation. We do not rely on a single cue but rather combine motion and appearance similarly to Malgireddy et al. (2012).

The most similar works to this paper are in the field of HMI as for example Lui (2012) and Wu et al. (2012): they both exploit depth information and aim at one-shot learning trying to achieve low computational cost. The first method employs a nonlinear regression framework on manifolds: actions are represented as tensors decomposed via Higher Order Singular Value Decomposition. The underlying geometry of tensor space is used. The second one extracts Extended-MHI as features and uses Maximum Correlation Coefficient (Hirschfeld, 1935) as classifier. Features from RGB and Depth streams are fused via a Multiview Spectral Embedding (MSE). Differently from these works, our approach aims specifically to obtain an accurate real-time recognition from one video example only.

We conclude the section with a reference to some works focusing on continuous action or activity recognition (Ali and Aggarwal, 2001; Green and Guan, 2004; Liao et al., 2006; Alon et al., 2009). In this case training and test videos contain many sequential gestures, therefore the temporal segmentation of videos becomes fundamental. Our work deals with continuous action recognition as well, indeed the proposed framework comprehends a novel and robust temporal segmentation algorithm.

3. Visual Recognition with Sparse Data

One-shot learning is a challenging requirement as the small quantity of training data makes the modeling phase extremely hard. For this reason, in one-shot learning settings a careful choice of the data representation is very important. In this work we rely on sparse coding to obtain a compact descriptor with a good discriminative power even if it is derived from very small data sets.

The main concept behind sparse coding is to approximate an input signal as a linear combination of a few components selected from a dictionary of basic elements, called atoms. We refer to *adaptive sparse coding* when the coding is driven by data. In this case, we require a *dictionary learning* stage, where the dictionary atoms are learnt (Olshausen and Fieldt, 1997; Yang et al., 2009; Wang et al., 2010).

The motivations behind the use of image coding arise from biology: there is evidence that similar signal coding happens in the neurons of the primary visual cortex (V1), which produces sparse and overcomplete activations (Olshausen and Fieldt, 1997). From the computational point of view the objective is to find an overcomplete model of images, unlike methods such as PCA, which

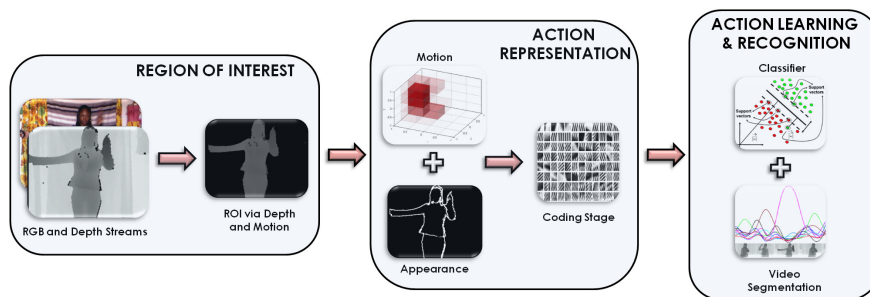


Figure 1: Overview of the recognition system, where video segmentation and classification are performed simultaneously.

aims at finding a number of components that is lower than the data dimensionality. Overcomplete representation techniques have become very popular in applications such as denoising, inpainting, super-resolution, segmentation (Elad and Aharon, 2006; Mairal et al., 2008b,a) and object recognition (Yang et al., 2009). In this work we assess their effectiveness also for gesture recognition. Let $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m] \in \mathbb{R}^{n \times m}$ be the matrix whose m columns $\mathbf{x}_i \in \mathbb{R}^n$ are the feature vectors. The goal of adaptive sparse coding is to learn a dictionary \mathbf{D} (a $n \times d$ matrix, with d the dictionary size and n the feature vector size) and a code \mathbf{U} (a $d \times m$ matrix) that minimize the reconstruction error:

$$\min_{\mathbf{D}, \mathbf{U}} \|\mathbf{X} - \mathbf{D}\mathbf{U}\|_F^2 + \lambda \|\mathbf{U}\|_1, \quad (1)$$

where $\|\cdot\|_F$ is the Frobenius norm. As for the sparsity, it is known that the L_1 -norm yields to sparse results while being robust to signals perturbations. Other penalties such as the L_0 -norm could be employed, however the problem of finding a solution becomes NP-hard and there is no guarantee that greedy algorithms reach the optimal solution. Notice that fixing \mathbf{U} , the above optimization reduces to a least square problem, whilst, given \mathbf{D} , it is equivalent to linear regression with the sparsifying norm L_1 . The latter problem is referred to as a feature selection problem with a known dictionary (Lee et al., 2007). One of the most efficient algorithms that converges to the optimal solution of the problem in Equation 1 for a fixed \mathbf{D} , is the *feature-sign search* algorithm (Lee et al., 2007). This algorithm searches for the sign of the coefficients \mathbf{U} ; indeed, considering only non-zero elements the problem is reduced to a standard unconstrained quadratic optimization problem (QP), which can be solved analytically. Moreover it performs a refinement of the signs if they are incorrect. For the complete procedure we refer the reader to Lee et al. (2007).

In the context of recognition tasks, it has been proved that a sparsification of the data representation improves the overall classification accuracy (see for instance Guyon and Elisseeff, 2003; Viola and Jones, 2004; Destrero et al., 2009 and references therein). In this case sparse coding is often cast into a *coding-pooling* scheme, which finds its root in the Bag of Words paradigm. In this scheme a *coding operator* is a function $f(\mathbf{x}_i) = \mathbf{u}_i$ that maps \mathbf{x}_i to a new space $\mathbf{u}_i \in \mathbb{R}^k$; when $k > n$ the representation is called overcomplete. The action of coding is followed by a pooling stage, whose purpose is to aggregate multiple local descriptors in a single and global one. Common pooling operators are the max operator, the average operator, or the geometric L_p -norm pooling operator (Feng et al., 2011). More in general, a pooling operator takes the codes located in S regions—for



Figure 2: Region of Interest detection. Left: RGB video frames. Center: depth frames. Right: the detected ROI.

instance cells of the spatial pyramid, as in Yang et al. (2009)—and builds a succinct representation. We define as Y_s the set of locations within the region s . Defining the pooling operator as g , the resultant feature can be rewritten as: $\mathbf{p}_{(s)} = g_{(i \in Y_s)}(\mathbf{u}_{(i)})$. After this stage, a region s of the image is encoded with a single feature vector. The final descriptor of the image is the concatenation of the descriptors \mathbf{p}_s among all the regions. Notice that the effectiveness of pooling is subject to the coding stage. Indeed, if applied on non-coded descriptors, pooling would bring to a drastic loss of information.

4. Action Recognition System

In this section we describe the versatile real-time action recognition system we propose. The system, depicted in Figure 1, consists of three layers, that can be summarized as follows:

- **Region Of Interest detection:** we detect a Region of Interest (ROI), where the human subject is actually performing the action. We use the combination of motion and depth to segment the subject from the background.
- **Action Representation:** each ROI within a frame is mapped into a feature space with a combination of 3D Histogram of Flow (3DHOF) and Global Histogram of Oriented Gradient (GHOG) on the depth map. The resultant 3DHOF+GHOG descriptor is processed via a sparse coding step to compute a compact and meaningful representation of the performed action.
- **Action Learning:** linear SVMs are used on frame buffers. A novel on-line video segmentation algorithm is proposed which allows isolating different actions while recognizing the action sequence.

4.1 Region Of Interest Segmentation

The first step of each action recognition system is to identify correctly where in the image the action is occurring. Most of the algorithms in the literature involve background modeling techniques

(Stauffer and Grimson, 1999), or space-time image filtering in order to extract the interesting spatio-temporal locations of the action (Laptev and Lindeberg, 2003). Other approaches require an *a priori* knowledge of the body pose (Lv and Nevatia, 2007). This task is greatly simplified in our architecture, since in human-machine interaction we can safely assume the human to stand in front of the camera sensors and that there is no other motion in the scene. For each video in the data set, we initially compute the frame differences within consecutive frames in a small buffer, obtaining the set P of pixels that are moving. Relying on this information, we compute the mean depth μ of the pixels belonging to P , which corresponds to the mean depth of the subject within the considered buffer. Thus, for the rest of the video sequence, we select the region of interest as $ROI(t) = \{p_{i,j}(t) : \mu - \varepsilon \leq d(p_{i,j}(t)) \leq \mu + \varepsilon\}$, where $d(p_{i,j}(t))$ is the depth of the pixel $p_{i,j}(t)$ at time t and ε is a tolerance value. In Figure 2 examples of segmentation are shown. We determined empirically that this segmentation procedure achieves better performance with respect to classic thresholding algorithms such as Otsu’s method (Otsu, 1979).

4.2 Action Representation

Finding a suitable representation is the most crucial part of any recognition system. Ideally, an image representation should be both *discriminative* and *invariant* to image transformations. A discriminative descriptor should represent features belonging to the same class in a similar way, while it should show low similarity among data belonging to different classes. The invariance property, instead, ensures that image transformations such as rotation, translation, scaling do not affect the final representation. In practice, there is a trade-off between these two properties (Varma and Ray, 2007): for instance, image patches are highly discriminative but not invariant, whereas image histograms are invariant but not discriminative, since different images could be associated to the same representation. When a lot of training data is provided, one could focus on a more discriminative and less invariant descriptor. In our specific case however, where only one training example is provided, invariance is a necessary condition in order to provide discriminant features; this aspect is greatly considered in our method.

From the neuroscience literature it is known that body parts are represented already in the early stages of human development (Mumme, 2001) and that certainly adults have prior knowledge on the body appearance. Many suggests that motion alone can be used to recognize actions (Bisio et al., 2010). In artificial systems this developmental-scale experience is typically not available, although actions can still be represented from two main cues: motion and appearance (Giese and Poggio, 2003). Although many variants of complex features describing human actions have been proposed, many of them imply computationally expensive routines. Differently, we rely on simple features in order to fulfill real-time requirements, and we show that they still have a good discriminative power. In particular we show that a combination of 3D Histograms of Flow (3DHOFs) and Global Histograms of Gradient (GHOGs) models satisfactorily human actions. When a large number of training examples is available, these two features should be able to describe a wide variety of actions, however in one-shot learning scenarios with noisy inputs, they are not sufficient. In this respect, a sparse representation, which keeps only relevant and robust components of the feature vector, greatly simplifies the learning phase making it equally effective.

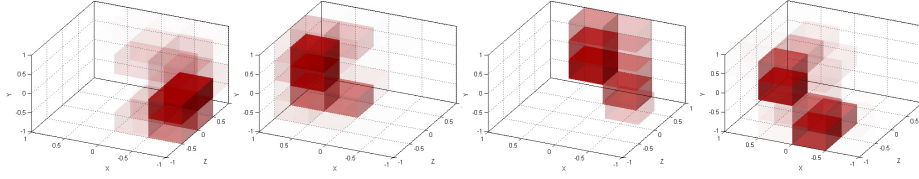


Figure 3: The figure illustrates high level statistics obtained by the proposed scene flow description (3D-HOFs). Starting from the left we show the histogram of the scene flow directions at time t , for a moving hand going on the *Right*, *Left*, *Forward*, *Backward* respectively. Each cuboid represents one bin of the histogram, for visualization purposes we divided the 3D space in $n \times n \times n$ bins with $n = 4$. Filled cuboids represent high density areas.

4.2.1 3D HISTOGRAM OF FLOW

Whereas 2D motion vector estimation has been largely investigated and various fast and effective methods are available today (Papenberg et al., 2006; Horn and Shunk, 1981), the scene flow computation (or 3D motion field estimation) is still an active research field due to the required additional binocular disparity estimation problem. The most promising works are the ones from Wedel et al. (2010), Huguet and Devernay (2007) and Cech et al. (2011); however these algorithms are computationally expensive and may require computation time in the range of 1.5 seconds per frame. This high computational cost is due to the fact that scene flow approaches try to estimate both the 2D motion field and disparity changes. Because of the real-time requirement, we opted for a simpler and faster method that produces a coarser estimation, but is effective for our purposes.

For each frame F_t we compute the 2D optical flow vectors $U(x, y, t)$ and $V(x, y, t)$ for the x and y components with respect to the previous frame F_{t-1} , via the Farneback algorithm (Farneback, 2003). Each pixel (x_{t-1}, y_{t-1}) belonging to the ROI of the frame F_{t-1} is reprojected in 3D space $(X_{t-1}, Y_{t-1}, Z_{t-1})$ where the Z_{t-1} coordinate is measured through the depth sensor and X_{t-1}, Y_{t-1} are computed by:

$$\begin{pmatrix} X_{t-1} \\ Y_{t-1} \end{pmatrix} = \begin{pmatrix} \frac{(x_{t-1} - x_0)Z_{t-1}}{f} \\ \frac{(y_{t-1} - y_0)Z_{t-1}}{f} \end{pmatrix},$$

where f is the focal length and $(x_0, y_0)^T$ is the principal point of the sensor. Similarly, we can reproject the final point (x_t, y_t) of the 2D vector representing the flow, obtaining another 3D vector $(X_t, Y_t, Z_t)^T$. For each pixel of the ROI, we can define the scene flow as the difference of the two 3D vectors in two successive frames F_{t-1} and F_t :

$$\begin{aligned} \mathbf{D} &= (\dot{X}, \dot{Y}, \dot{Z})^T = \\ &= (X_t - X_{t-1}, Y_t - Y_{t-1}, Z_t - Z_{t-1})^T. \end{aligned}$$

Once the 3D flow for each pixel of the ROI at time t has been computed, we normalize it with respect to the L_2 -norm, so that the resulting descriptors $\mathbf{D}_1, \dots, \mathbf{D}_n$ (n pixels of the ROI) are invariant to the overall speed of the action. In order to extract a compact representation we build a 3D Histogram

of Flow (3DHOF) $\mathbf{z}(t)$ of the 3D motion vectors, where $\mathbf{z}(t) \in \mathbb{R}^{n_1}$ and $\sqrt[3]{n_1}$ is the quantization parameter of the space (i.e., the bin size). In addition we normalize each 3DHOF $\mathbf{z}(t)$ so that $\sum_j z_j(t) = 1$; hence we guarantee that these descriptors are invariant to the subject of interest's scale.

Figure 3 shows that the movements toward different directions reveal to be linearly separable, and the main directions are accurately represented: each cuboid represents one bin of the histogram, and the 3D space is divided in $n \times n \times n$ bins with $n = 4$. It is possible to notice how, in the *Right* direction for example, all the filled bins lay on the semi-space defined by $x < 0$. Similar observations apply all cases.

4.2.2 GLOBAL HISTOGRAM OF ORIENTED GRADIENT

In specific contexts, motion information is not sufficient to discriminate actions, and information on the pose or appearance becomes crucial. One notable example is the American Sign Language (ASL), whose lexicon is based mostly on the shape of the hand. In these cases modeling the shape of a gesture as well as its dynamics is very important. Thus we extend the motion descriptor with a shape feature computed on the depth map. If we assume the subject to be in front of the camera, it is unlikely that the perspective transformation would distort his/her pose, shape or appearance, therefore we can approximately work with invariance to translation and scale. We are interested in characterizing shapes, and the gradient of the depth stream shows the highest responses on the contours, thus studying the orientation of the gradient is a suitable choice. The classical Histograms of Oriented Gradient (HOGs) (Dalal and Triggs, 2005) have been designed for detection purposes and do not show the above-mentioned invariance; indeed dividing the image in cells makes each sub-histogram dependent on the location and the dimension of the object. Furthermore, HOGs exhibit a high spatial complexity, as the classical HOG descriptor belongs to $\mathbb{R}^{(ncells \times nblocks \times n_2)}$. Since we aim at preserving such invariance as well as limiting the computational complexity, we employed a simpler descriptor, the Global Histogram of Oriented Gradient (GHOG). This appearance descriptor produces an overall description of the appearance of the ROI without splitting the image in cells. We compute the histogram of gradient orientations of the pixels on the entire ROI obtained from the depth map to generate another descriptor $\mathbf{h}(t) \in \mathbb{R}^{n_2}$, where n_2 is the number of bins. The scale invariance property is preserved normalizing the descriptor so that $\sum_j h_j(t) = 1$. Computing this descriptor on the depth map is fundamental in order to remove texture information; in fact, in this context, the only visual properties we are interested in are related to shape.

4.2.3 SPARSE CODING

At this stage, each frame F_t is represented by two global descriptors: $\mathbf{z}(t) \in \mathbb{R}^{n_1}$ for the motion component and $\mathbf{h}(t) \in \mathbb{R}^{n_2}$ for the appearance component. Due to the high variability of human actions and to the simplicity of the descriptors, a feature selection stage is needed to catch the relevant information underlying the data and discarding the redundant ones such as background or body parts not involved in the action; to this aim we apply a sparse coding stage to our descriptor.

Given the set of the previously computed 3DHOFs $\mathbf{Z} = [\mathbf{z}(1), \dots, \mathbf{z}(K)]$, where K is the number of all the frames in the training data, our goal is to learn one motion dictionary \mathbf{D}_M (a $n_1 \times d_1$ matrix, with d_1 the dictionary size and n_1 the motion vector size) and the codes \mathbf{U}_M (a $d_1 \times K$ matrix) that minimize the Equation 1, so that $\mathbf{z}(t) \sim \mathbf{D}_M \mathbf{u}_M(t)$. In the same manner, we define the equal optimization problem for a dictionary \mathbf{D}_G (a $n_2 \times d_2$ matrix) and the codes \mathbf{U}_G (a $d_2 \times K$ matrix) for

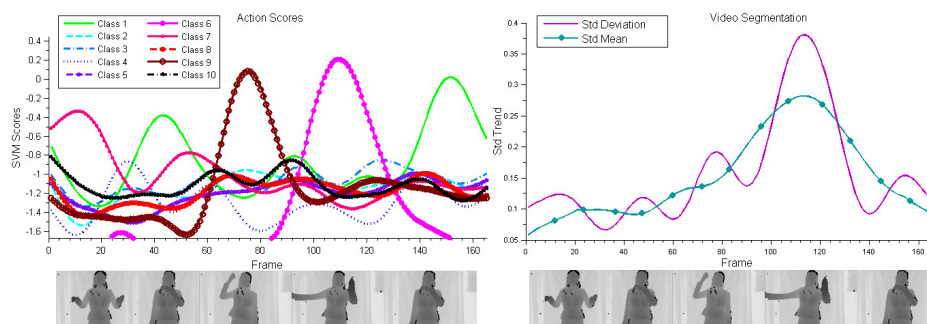


Figure 4: The figure illustrates on the left the SVMs scores (Equation 2) computed in real-time at each time step t over a sequence of 170 frames. On the right the standard deviation of the scores and its mean computed on a sliding window are depicted. The local minima of the standard deviation function are break points that define the end of an action and the beginning of another one. See Section 4.3.2 for details.

the set of GHOGs descriptors $\mathbf{H} = [\mathbf{h}(1), \dots, \mathbf{h}(K)]$. Therefore, after the Sparse Coding stage, we can describe a frame as a code $\mathbf{u}(i)$, which is the concatenation of the motion and appearance codes: $\mathbf{u}(i) = [\mathbf{u}_M(i), \mathbf{u}_G(i)]$.

Notice that we rely on global features, thus we do not need any pooling operator, which is usually employed to summarize local features into a single one.

4.3 Learning and Recognition

The goal of this phase is to learn a model of a given action from data. Since we are implementing a one-shot action recognition system, the available training data amounts to one training sequence for each action of interest. In order to model the temporal extent of an action we extract sets of sub-sequences from a sequence, each one containing T adjacent frames. In particular, instead of using single frame descriptors (described in Section 4.2), we move to a concatenation of frames: a set of T frames is represented as a sequence $[\mathbf{u}(1), \dots, \mathbf{u}(T)]$ of codes. This representation allows us to perform simultaneously detection and classification of actions.

The learning algorithm we adopt is the Support Vector Machine (SVM) (Vapnik, 1998). We employ linear SVMs, since they can be implemented with constant complexity during the test phase fulfilling real-time requirements (Fan et al., 2008). Additionally, recent advances in the object recognition field, such as Yang et al. (2009), showed that linear classifiers can effectively solve the classification problem if a preliminary sparse coding stage has previously been applied. Our experiments confirm these findings. Another advantage of linear SVMs is that they can be implemented with a linear complexity in training (Fan et al., 2008); given this property, we can provide a real-time one-shot learning procedure, extremely useful in real applications.

The remainder of the section describes in details the two phases of action learning and action recognition.

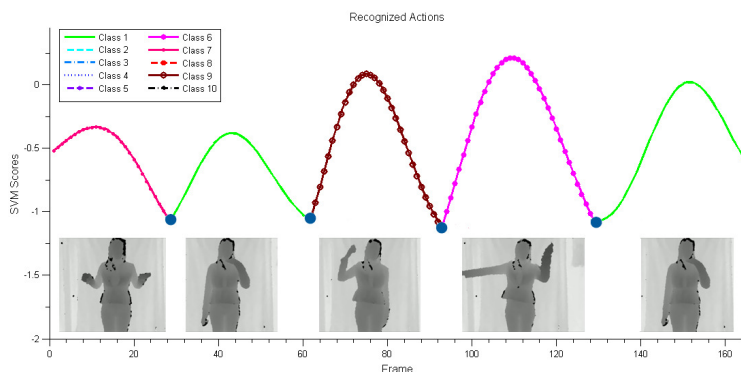


Figure 5: The figure illustrates only the scores of the recognized actions via the method described in Section 4.3.2. Blue dots are the break points computed by the video segmentation algorithm that indicate the end of an action and the beginning of a new one.

4.3.1 ACTION LEARNING

Given a video V_s of t_s frames, containing only one action A_s , we compute a set of descriptors $[\mathbf{u}(1), \dots, \mathbf{u}(t_s)]$ as described in Section 4.2. Then, action learning is carried out on a set of data that are descriptions of a frame buffer $\mathbf{B}_T(t)$, where T is its length:

$$\mathbf{B}_T(t) = (\mathbf{u}(t-T), \dots, \mathbf{u}(t-1), \mathbf{u}(t))^T.$$

We use a one-versus-all strategy to train a binary linear SVM for each class A_s , so that at the end of the training phase we obtain a set of N linear SVM classifiers $f_1(\bar{\mathbf{B}}), \dots, f_N(\bar{\mathbf{B}})$, where N is the number of actions. In particular, in this one-shot learning pipeline, the set of buffers

$$\mathbf{B}_s = [\mathbf{B}_T(t_0), \dots, \mathbf{B}_T(t_s)]$$

computed from the single video V_s of the class A_s are used as positive examples for the action A_s . All the buffers belonging to A_j with $j \neq s$ are the negative examples. Although we use only one example for each class, we benefit from the chosen representation: indeed, descriptors are computed per frame, therefore one single video of length t_s provides a number of examples equal to $t_s - T$ where T is the buffer size. Given the training data $\{\mathbf{B}, \mathbf{y}\}$ where \mathbf{B} is the set of positive and negative examples for the primitive A_s , $y_i = 1$ if the example is positive, $y_i = -1$ otherwise, the goal of SVM is to learn a linear function (\mathbf{w}^T, b) such that a new test vector $\bar{\mathbf{B}}$ is predicted as:

$$y_{pred} = \text{sign}(f(\bar{\mathbf{B}})) = \text{sign}(\mathbf{w}^T \bar{\mathbf{B}} + b).$$

4.3.2 ON-LINE RECOGNITION: VIDEO SEGMENTATION

Given a test video V , which may contain one or more known actions, the goal is to predict the sequence of the performed actions. The video is analyzed using a sliding window $\mathbf{B}_T(t)$ of size T . We compute the output score $f_i(\mathbf{B}_T(t))$ of the $i = 1, \dots, N$ SVM machines for each test buffer $\mathbf{B}_T(t)$ and we filter these scores with a low-pass filter W that attenuates noise. Therefore the new score at

time t becomes:

$$H_i(\mathbf{B}_T(t)) = W \star f_i(\mathbf{B}_T(t)) \quad i = 1, \dots, N, \quad (2)$$

where the \star is the convolution operator. Figure 4 depicts an example of these scores computed in real-time. As long as the scores evolve we need to predict (on-line) when an action ends and another one begins; this is achieved computing the standard deviation $\sigma(H)$ for a fixed t over all the scores H_i^t (Figure 4, right chart). When an action ends we can expect all the SVM output scores to be similar, because no model should be predominant with respect to idle states; this brings to a local minimum in the function $\sigma(H)$. Therefore, each local minimum corresponds to the end of an action and the beginning of a new one. Let n be the number of local minima computed from the standard deviation function; there will be $n + 1$ actions, and in particular actions with the highest score before and after each break point will be recognized. We can easily find these minima in real-time: we calculate the mean value of the standard deviation over time using a sliding window. When the standard deviation trend is below the mean, all the SVMs scores predict similar values, hence it is likely that an action has just ended. In Figure 5 the segmented and recognized actions are shown together with their scores.

5. Experiments

In this section we evaluate the performance of our system in three different settings:

- **ChaLearn Gesture Data Set.** The first experiment has been conducted on a publicly available data set, released by ChaLearn (, CGD2011). The main goal of the experiment is to compare our method with other techniques.
- **Kinect Data.** In the second experiment we discuss how to improve the recognition rate using all the functionalities of a real Kinect sensor. Gestures with high level of detail are easily caught by the system.
- **Human-Robot Interaction.** For the last experiment we considered a real HMI scenario: we implement the system on a real robot, the iCub humanoid robot (Metta et al., 2008), showing the applicability of our algorithm also in human-robot interaction settings.

For the computation of the accuracy between a sequence of estimated actions and the ground truth sequence we use the normalized Levenshtein Distance (Levenshtein, 1966), defined as:

$$TeLev = \frac{S + D + I}{M},$$

where each action is treated as a symbol in a sequence, S is the number of substitutions (misclassifications), D the number of deletions (false negatives), I the number of insertions (false positives) and M the length of the ground truth sequence. More specifically, this measure computes the minimum number of modifications that are required to transform a sequence of events in another one. It is widely used in speech recognition contexts, where each symbol represents an event. In action and gesture recognition, when sequences of gestures are to be evaluated, the Levenshtein Distance shows to be a particularly suitable metric, as it allows accounting not only for the single classifier accuracy, but also for the capability of the algorithm to accurately distinguish different gestures in a sequence (Minnen et al., 2006).

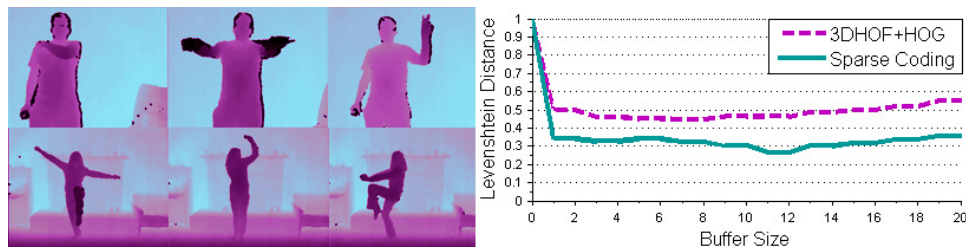


Figure 6: On the left examples of 2 different batches from the ChaLearn Data Set (, CGD2011). On the right the overall Levenshtein Distance computed in 20 batches with respect to the buffer size parameter is depicted for both 3DHOF+GHOG features and descriptors processed with sparse coding.

We empirically choose a quantization parameter for the 3DHOF, n_1 equal to 5, $n_2 = 64$ bins for the GHOG descriptor, and dictionary sizes d_1 and d_2 equal to 256 for both motion and appearance components. This led to a frame descriptor of size 189 for simple descriptors, which increases to 512 after the sparse coding processing. The whole system runs at 25fps on 2.4Ghz Core 2 Duo Processor.

5.1 ChaLearn Gesture Data Set

We firstly assess our method on the ChaLearn data set for the One-Shot Gesture Recognition Challenge (Guyon et al., 2012), see Figure 6. The data set is organized in batches, where each batch includes 100 recorded gestures grouped in sequences of 1 to 5 gestures arbitrarily performed at different speeds. The gestures are drawn from a small vocabulary of 8 to 15 unique gestures called *lexicon*, which is defined within a batch. For each video both RGB and Depth streams are provided, but *only one example* is given for the training phase. In our experiments we do not use information on the body pose of the human. We consider the batches from *devel_01* to *devel_20*; each batch has 47 videos, where L (the lexicon size) videos are for training and the remaining are used as test data.

The main parameter of the system is the buffer size T , however in Figure 6 it is possible to notice that the parameter offers stable performances with a buffer range of 1 – 20, so it does not represent a critical variable of our method. Furthermore, high performance for a wide buffer length range imply that our framework is able to handle different speeds implicitly. We compute the Levenshtein Distance as the average over all the batches, which is 25.11% for features processed with sparse coding, whereas simple 3DHOF+GHOG descriptors without sparse coding lead to a performance of 43.32%. Notably, each batch has its own lexicon and some of them are composed of only gestures performed by hand or fingers; in these cases, if the GHOG is computed on the entire ROI, the greatest contribution of the histogram comes from the body shape, whilst finger actions (see Figure 2, bottom row) represent a poor percentage of the final descriptor. If we consider batches where the lexicon is not composed of only hand/fingers gestures, the Levenshtein Distance reduces to 15%.

We compared our method with several approaches. First of all a Template Matching technique, where we used as descriptor the average of all depth frames for each action. The test video is split in

Method	TeLev	TeLen
Sparse Representation (proposed)	25.11%	5.02%
3DHOF + GHOG	43.32%	9.03%
Template Matching	62.56%	15.12%
DTW	49.41%	Manual
Manifold LSR (Lui, 2012)	28.73%	6.24%
MHI (Wu et al., 2012)	30.01%	NA
Extended-MHI (Wu et al., 2012)	26.00%	NA
BoVW (Wu et al., 2012)	72.32%	NA
2D FFT-MHI (Mahbub et al., 2011)	37.46%	NA
TBM+LDA (Malgireddy et al., 2012)	24.09%	NA

Table 1: Levenshtein Distance on the ChaLearn Gesture Data Set. For SVM classification we chose the appropriate buffer size for each batch according to the defined lexicon. TeLev is the Levenshtein Distance, TeLen is the average error (false positives + false negatives) made on the number of gestures (see text).

slices estimated using the average size of actions. In the recognition phase we classify each slice of the video comparing it with all the templates. The overall Levenshtein Distance becomes 62.56%. For the second comparison we employ Dynamic Time Warping (DTW) method (Sakoe and Chiba, 1978) with 3DHOF + GHOG features. We manually divided test videos in order to facilitate the recognition for DTW; nevertheless the global Levenshtein Distance is 49.41%. Finally we report the results presented in some recent works in the field, which exploit techniques based on manifolds (Lui, 2012), Motion History Image (MHI) (Wu et al., 2012), Bag of Visual Words (BoVW) (Wu et al., 2012), 2D FFT-MHI (Mahbub et al., 2011) and Temporal Bayesian Model (TBM) with Latent Dirichlet Allocation (LDA) (Malgireddy et al., 2012).

Table 1 shows that most of the compared approaches are outperformed by our method except for Malgireddy et al. (2012); however the method proposed by Malgireddy et al. (2012) has a training computational complexity of $O(n \times k^2)$ for each action class, where k is the number of HMM states and n the number of examples, while the testing computational complexity for a video frame is $O(k^2)$. Thanks to the sparse representation, we are able to use linear SVMs, which reduce the training complexity with respect to the number of training examples to $O(n \times d)$ for each SVM, where d is the descriptor size. In our case d is a constant value fixed a priori, and does not influence the scalability of the problem. Therefore we may approximate the asymptotic behavior of the SVM in training to $O(n)$. Similarly, in testing the complexity for each SVM is constant with respect to the number of training examples when considering a single frame, and it becomes $O(N)$ for the computation of all the N class scores. This allows us to provide real-time training and testing procedures with the considered lexicons.

Furthermore our on-line video segmentation algorithm shows excellent results with respect to the temporal segmentation used in the compared frameworks; in fact it is worth noting that the proposed algorithm leads to an action detection error rate $TeLen = \frac{FP+FN}{M}$ equal to 5.02%, where FP and FN are false positives and false negatives respectively, and M is the number of all test

gestures. Considering the final results of the ChaLearn Gesture Challenge (Round 1),¹ we placed 9th over 50 teams, but our method also fulfills real-time requirements for the entire pipeline, which was not a requirement of the challenge.

5.1.1 MOTION VS APPEARANCE

In this section we evaluate the contribution of the frame descriptors. In general we notice that the combination of both motion and appearance descriptors leads to the best results when the lexicon is composed of actions where both motion and appearance are equally important. To show this, we considered the 20 development batches from the ChaLearn Gesture Data Set. For this experiment, we used only coded descriptors, since we have already experienced that they obtain higher performance. Using only the motion component, the Levenshtein Distance is equal to 62.89%, whereas a descriptor based only on the appearance leads to an error of 34.15%. The error obtained using only the 3DHOF descriptors was expected, due to the nature of the lexicons chosen: indeed in most gestures the motion component has little significance. Considering instead batch *devel_01*, where motion is an important component in the gesture vocabulary, we have that 3DHOF descriptors lead to a Levenshtein Distance equal to 29.48%, the GHOG descriptors to 21.12% and the combination is equal to 9.11%. Results are consistent with previous findings, but in this specific case the gap between the motion and the appearance components is not critical.

5.1.2 LINEAR VS NON-LINEAR CLASSIFIERS

In this section we compare the performances of linear and non linear SVM for the action recognition task. The main advantage of a linear kernel is the computational time: non-linear SVMs have a worst case training computational complexity per class equal to $O(n^3 \times d)$ against the $O(n \times d)$ of linear SVMs, where n is the number of training examples, and d is the descriptor size. In testing, non linear SVMs show computational complexity of $O(n \times d)$ per frame, since the number of support vectors grows linearly with n . Moreover, non-linear classifiers usually require additional kernel parameter estimation, which especially in one-shot learning scenarios is not trivial. Contrarily, linear SVMs take $O(d)$ per frame. For this experiment we used coded features where both motion and appearance are employed. A non-linear SVM with RBF Kernel has been employed, where the kernel parameter and the SVM regularization term have been chosen empirically after 10 trials on a subset of the batches. The Levenshtein Distance among the 20 batches is 35.11%; this result confirms that linear classifiers are sufficient to obtain good results with low computational cost if an appropriate data representation, as the one offered by sparse coding, is adopted.

5.2 Kinect Data Set

In this section we assess the ability of our method to recognize more complex gestures captured by a Kinect for Xbox 360 sensor. In Section 5.1, we noted that the resolution of the proposed appearance descriptor is quite low and may not be ideal when actions differ by small details, especially on the hands, therefore a localization of the interesting parts to model would be effective. The simplest way to build in this specific information is to resort to a body part tracker; indeed, if a body tracker were available it would have been easy to extract descriptors from different limbs and then concatenate all the features to obtain the final frame representation. An excellent candidate to provide a reliable

1. The leaderboard website is: <https://www.kaggle.com>.

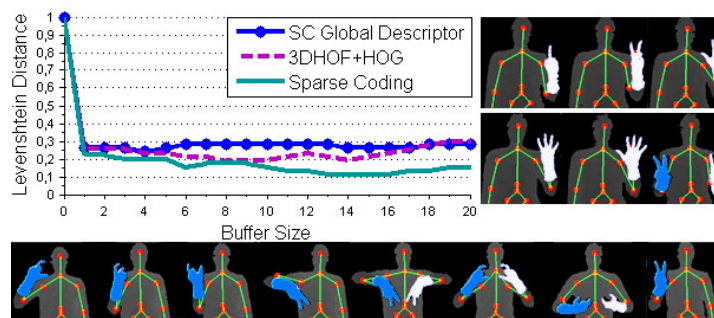


Figure 7: On the right and bottom the two vocabularies used in Section 5.2; these gestures are difficult to model without a proper body tracker, indeed the most contribution for the GHOG comes from the body shape rather than the hand. On the left the Levenshtein Distance.

body tracker is Microsoft Kinect SDK, which implements the method in Shotton et al. (2011). This tool retrieves the 20 principal body joints position and pose of the user’s current posture. Given these positions, we assign each 3D point of the ROI to its nearest joint, so that it is possible to correctly isolate the two hands and the body from the rest of the scene (see Figure 7). Then, we slightly modify the approach, computing 3DHOF and GHOG descriptors on three different body parts (left/right hand and whole body shape); the final frame representation becomes the concatenation of all the part descriptors. As for the experiments we have acquired two different sets of data (see Figure 7): in the first one the lexicon is composed of numbers performed with fingers, in the second one we replicate the lexicons *devel_3* of the ChaLearn Gesture Data Set, the one where we obtained the poorest performances. In Figure 7 on the left the overall accuracy is shown; using sparse coding descriptors computed only on the body shape we obtain a Levenshtein Distance around 30%. By concatenating descriptors extracted from the hands the system achieves 10% for features enhanced with sparse coding and 20% for normal descriptors.

We compared our method with two previously mentioned techniques: a Template Matching algorithm and an implementation of the Dynamic Time Warping approach (Sakoe and Chiba, 1978). The resulted Levenshtein Distance is respectively 52.47% and 42.36%.

5.3 Human-Robot Interaction

The action recognition system has been implemented and tested on the iCub, a 53 degrees of freedom humanoid robot developed by the RobotCub Consortium (Metta et al., 2008). The robot is equipped with force sensors and gyroscopes, and it resembles a 3-years old child. It mounts two Dragonfly cameras, providing the basis for 3D vision, thus after an offline camera calibration procedure we can rely on a full stereo vision system; here the depth map is computed following Hirschmuller (2008). In this setting the action recognition system can be used for more general purposes such as Human-Robot-Interaction (HRI) or learning by imitation tasks. In particular our goal is to teach iCub how to perform simple manipulation tasks, such as move/grasp an object. In this sense, we are interested in recognizing actions related to the arm-hand movements of the robot. We define 8 actions, as shown in Figure 8, bottom row, according to the robot manipulation capabilities.

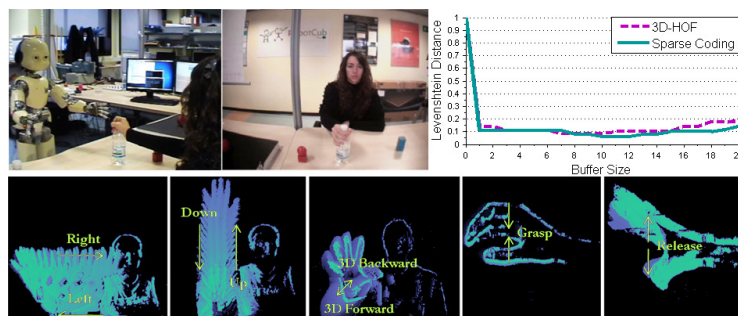


Figure 8: Accuracy for actions sequences (see bottom row). We evaluated the performance on more than 100 actions composed of sequences of 1 to 6 actions.

Each action is modeled using only the motion component (3DHOF), since we want the descriptor to be independent on the particular object shape used.

In Figure 8 we show the accuracy based on the Levenshtein Distance; this measure has been calculated on more than 100 actions composed of sequences of 1 to 6 actions. Notably the error is less than 10%; these good results were expected due to the high discriminative power of the 3DHOFs (Figure 3) on the chosen lexicon, which leads to a linearly separable set.

6. All Gestures You Can: a Real Application

As pointed out in the previous sections, our approach was designed for real applications where real-time requirements need to be fulfilled. We developed and implemented a “game” against a humanoid robot, showing the effectiveness of our system in a real HRI setting: “All Gestures You Can” (Gori et al., 2012), a game aiming at improving memory skills, visual association and concentration. Our game takes inspiration from the classic “Simon” game; nevertheless, since the original version has been often defined as “visually boring”, we developed a revisited version, based on gesture recognition, which involves a “less boring” opponent: the iCub (Metta et al., 2008). Both the human and the robot have to take turns and perform the longest possible sequence of gestures by adding one gesture at each turn: one player starts performing a gesture, the opponent has to recognize the gesture, imitate it and add another gesture to the sequence. The game is carried on until one of the two players loses: the human player can lose because of limited memory skills, whereas the robot can lose because the gesture recognition system fails. As described in the previous sections, the system has been designed for one-shot learning; however, Kinect does not provide information about finger configuration, therefore a direct mapping between human fingers and the iCub’s ones is not immediate. Thus we set a predefined pool of 8 gestures (see Figure 9, on the left). The typical game setting is shown in Figure 10: the player stays in front of the robot while performing gestures that are recognized with Kinect. Importantly, hand gestures cannot be learned exploiting the Skeleton Data of Kinect: the body tracker detects the position of the hand and it is not enough to discriminate more complicate actions,—for example, see gesture classes 1 and 5 or 2 and 6 in Figure 9.

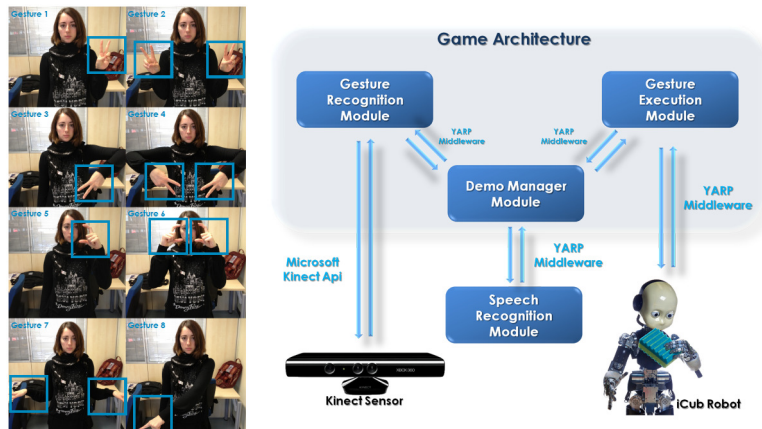


Figure 9: On the left the hand gestures. The vision system has been trained using 8 different actors performing each gesture class for 3 times. On the right the game architecture. There are three main modules that take care of recognizing the action sequence, defining the game rules and making the robot gestures.

The system is simple and modularized as it is organized in three components (see Figure 9) based on the iCub middleware, YARP (Metta et al., 2006), which manages the communication between sensors, processors, and modules. The efficiency of the proposed implementation is assured by its multithreading architecture, which also contributes to real-time performances. The software presented in this section is available in the iCub repository.²

The proposed game has been played by more than 30 different players during the ChaLearn Kinect Demonstration Competition at CVPR 2012.³ Most of them were completely naive without prior knowledge about the gestures. They were asked to play using a lexicon that had been trained specifically for the competition (Figure 9). After 50 matches we had 75% of robot victories. This result indicates that the recognition system is robust also to different players performing variable gestures at various speeds. 15% of the matches have been won by humans and usually they finished during the first 3-4 turns of the game; this always occurred when players performed very different gestures with respect to the trained ones. A few players (10% of matches) succeeded in playing more than 8 turns, and they won due to recognition errors. “All Gestures You Can” ranked 2nd in the ChaLearn Kinect Demonstration Competition.

7. Discussion

This paper presented the design and implementation of a complete action recognition system to be used in real world applications such as HMI. We designed each step of the recognition pipeline to function in real-time while maximizing the overall accuracy. We showed how a sparse action repre-

2. Code available at <https://svn.code.sf.net/p/robotcub/code/trunk/iCub/contrib/src/demoGestureRecognition>.

3. The competition website is <http://gesture.chalearn.org/>

A YouTube video of our game is available at http://youtu.be/U_JLoe_ft3I.

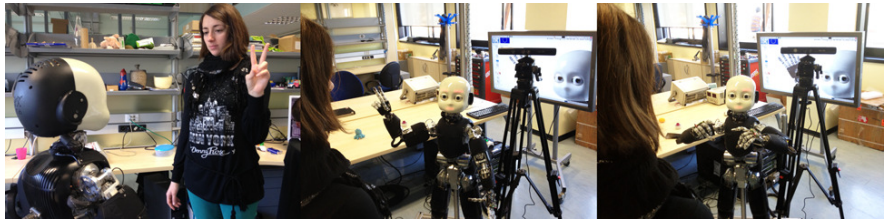


Figure 10: The first two turns of a match. Left: the human player performs the first gesture of the sequence. Center: iCub recognized the gesture and imitates it. Right: iCub adds a new random gesture to the sequence.

sensation could be effectively used for one-shot learning of actions in combination with conventional machine learning algorithms (i.e., SVM), even if the latter would normally require a larger set of training data. The comprehensive evaluation of the proposed approach showed that we achieve good trade-off between accuracy and computation time. The main strengths of our learning and recognition pipeline can be summarized as follows:

1. **One-Shot Learning:** one example is sufficient to teach an new action to the system; this is mainly due to the effective per-frame representation.
2. **Sparse Frame Representation:** starting from a simple and computationally inexpensive description that combines global motion (3DHOF) and appearance (GHOG) information over a ROI, subsequently filtered through sparse coding, we obtained a sparse representation at each frame. We showed that these global descriptors are appropriate to model actions of the upper body of a person.
3. **On-line Video Segmentation:** we propose a new, effective, reliable and on-line video segmentation algorithm that achieved a 5% error rate on action detection on a set of 2000 actions grouped in sequences of 1 to 5 gestures. This segmentation procedure works concurrently with the recognition process, thus a sequence of actions is simultaneously segmented and recognized.
4. **Real-time Performances:** the proposed system can be used in real-time applications, as it does require neither a complex features processing nor a computationally expensive training and testing phases. From the computational point of view the proposed approach scales well even for large vocabularies of actions.
5. **Effectiveness in Real Scenarios:** our method achieves good performances in a Human-Robot Interaction setting, where the RGBD images are obtained through binocular vision and disparity estimation. For testing purposes, we proposed a memory game, called “All Gestures You Can”, where a person can challenge the iCub robot on action recognition and sequencing. The system ranked 2nd at the Kinect Demonstration Competition.⁴

4. The competition website is <http://gesture.chalearn.org/>.

We stress here the simplicity of the learning and recognition pipeline: each stage is easy to implement and fast to compute. It is shown to be adequate to solve the problem of gesture recognition; we obtained high-quality results while fulfilling real-time requirements. The approach is competitive against many of the state-of-the-art methods for action recognition.

We are currently working on a more precise appearance description at frame level still under the severe constraint of real-time performance; this would enable the use of more complex actions even when the body tracker is not available.

Acknowledgments

This work was supported by the European FP7 ICT projects N. 270490 (EFAA) and N. 270273 (Xperience).

References

- J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 2011.
- A. Ali and J.K. Aggarwal. Segmentation and recognition of continuous human activity. *IEEE Workshop on Detection and Recognition of Events in Video*, 2001.
- J. Alon, V. Athitsos, Y. Quan, and S. Sclaroff. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.
- A. Bisio, N. Stucchi, M. Jacono, L. Fadiga, and T. Pozzo. Automatic versus voluntary motor imitation: Effect of visual context and stimulus velocity. *PLoS ONE*, 2010.
- A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2001.
- M. Bregonzio, S. Gong, and T. Xiang. Recognising action as clouds of space-time interest points. *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.
- M. J. Burden and D. B. Mitchell. Implicit memory development in school-aged children with attention deficit hyperactivity disorder (adhd): Conceptual priming deficit? In *Developmental Neuropsychology*, 2005.
- J. Cech, J. Sanchez-Riera, and R. Horaud. Scene flow estimation by growing correspondence seeds. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
- ChaLearn Gesture Dataset (CGD2011). <http://gesture.chalearn.org/data>, 2011.
- S.P. Chatzis, D. Kosmopoulos, and P. Doliotis. A conditional random field-based model for joint sequence segmentation and classification. In *Pattern Recognition*, 2013.
- C. Comoldi, A. Barbieri, C. Gaiani, and S. Zocchi. Strategic memory deficits in attention deficit disorder with hyperactivity participants: The role of executive processes. In *Developmental Neuropsychology*, 1999.

- N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.
- A. Destro, C. De Mol, F. Odone, and Verri A. A sparsity-enforcing method for learning face features. *IEEE Transactions on Image Processing*, 18:188–201, 2009.
- A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *International Conference on Computer Vision*, 2003.
- M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 2006.
- R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9, 2008.
- S. R. Fanello, I. Gori, and F. Pirri. Arm-hand behaviours modelling: from attention to imitation. In *International Symposium on Visual Computing*, 2010.
- G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian Conference on Image Analysis*, 2003.
- J. Feng, B. Ni, Q. Tian, and S. Yan. Geometric lp-norm feature pooling for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
- M. A. Giese and T. Poggio. Neural mechanisms for the recognition of biological movements. *Nature reviews. Neuroscience*, 2003.
- L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29, 2007.
- I. Gori, S. R. Fanello, F. Odone, and G. Metta. All gestures you can: a memory game against a humanoid robot. *IEEE-RAS International Conference on Humanoid Robots*, 2012.
- R.D. Green and L. Guan. Continuous human activity recognition. *Control, Automation, Robotics and Vision Conference*, 2004.
- I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *International Journal of Machine Learning Research*, 3:1157–1182, 2003.
- I. Guyon, V. Athitsos, P. Jangyodsuk, B. Hammer, and H. J. E. Balderas. Chalearn gesture challenge: Design and first results. In *Computer Vision and Pattern Recognition Workshops*, 2012.
- H. O. Hirschfeld. A connection between correlation and contingency. In *Mathematical Proceedings of the Cambridge Philosophical Society*, 1935.
- H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.
- B. K. P. Horn and B. G. Shunk. Determining optical flow. *Journal of Artificial Intelligence*, 1981.
- F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *International Conference on Computer Vision*, 2007.

- I. Laptev and T. Lindeberg. Space-time interest points. In *IEEE International Conference on Computer Vision*, 2003.
- I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
- H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Conference on Neural Information Processing Systems*, 2007.
- V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics - Doklady*, 1966.
- W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *Computer Vision and Pattern Recognition Workshops*, 2010.
- H.-Y.M. Liao, D.-Y. Chen, and S.-W. Shih. Continuous human action segmentation and recognition using a spatio-temporal probabilistic framework. *IEEE International Symposium on Multimedia*, 2006.
- Y. M. Lui. A least squares regression framework on manifolds and its application to gesture recognition. In *Computer Vision and Pattern Recognition Workshops*, 2012.
- F. Lv and R. Nevatia. Single view human action recognition using key pose matching and viterbi path searching. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.
- U. Mahbub, H. Imtiaz, T. Roy, S. Rahman, and A. R. Ahad. Action recognition from one example. *Pattern Recognition Letters*, 2011.
- J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008a.
- J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Transactions on Image Processing*, pages 53–69, 2008b.
- M. R. Malgireddu, I. Inwogu, and V. Govindaraju. A temporal Bayesian model for classifying, detecting and localizing activities in video sequences. *Computer Vision and Pattern Recognition Workshops*, 2012.
- G. Metta, P. Fitzpatrick, and L. Natale. YARP: Yet Another Robot Platform. *International Journal of Advanced Robotic Systems*, 2006.
- G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori. The icub humanoid robot: an open platform for research in embodied cognition. In *Workshop on Performance Metrics for Intelligent Systems*, 2008.
- D. Minnen, T. Westeyn, and T. Starner. Performance metrics and evaluation issues for continuous activity recognition. In *Performance Metrics for Intelligent Systems Workshop*, 2006.
- D. L. Mumme. Early social cognition: understanding others in the first months of life. *Journal of Infant and Child Development*, 2001.

- P. Natarajan and R. Nevatia. Coupled hidden semi markov models for activity recognition. In *Workshop Motion and Video Computing*, 2007.
- B. A. Olshausen and D. J. Fieldt. Sparse coding with an overcomplete basis set: a strategy employed by v1. *Vision Research*, 1997.
- N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 1979.
- N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert. Highly accurate optic flow computation with theoretically justified warping. *International Journal of Computer Vision*, 2006.
- R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 2010.
- H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1978.
- H. J. Seo and P. Milanfar. A template matching approach of one-shot-learning gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.
- J. W. Shneider and P. Borlund. Matrix comparison, part 1: Motivation and important issues for measuring the resemblance between proximity measures or ordination results. In *Journal of the American Society for Information Science and Technology*, 2007.
- J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
- C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, 1999.
- V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, Inc., 1998.
- M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *IEEE International Conference on Computer Vision*, 2007.
- P. Viola and M.J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57:137–154, 2004.
- J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.
- J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu. Robust 3d action recognition with random occupancy patterns. *European Conference on Computer Vision*, 2012.
- A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers. Stereoscopic scene flow computation for 3d motion understanding. *International Journal of Computer Vision*, 2010.
- G. Willems, T. Tuytelaars, and L. Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. *European Conference on Computer Vision*, 2008.

- D. Wu, F. Zhu, and L. Shao. One shot learning gesture recognition from rgb-d images. In *Computer Vision and Pattern Recognition Workshops*, 2012.
- J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

Dictionary Based Pooling for Object Categorization

Sean Ryan Fanello^{1,2}, Nicoletta Noceti², Giorgio Metta¹ and Francesca Odone²

¹*iCub Facility, Istituto Italiano di Tecnologia, Via Morego 30, 16163, GE, Italia*

²*DIBRIS, Università degli Studi di Genova, Via Dodecaneso 35, 16146, GE, Italia*

{*sean.fanello, giorgio.metta*}@iit.it, {*nicoletta.noceti, francesca.odone*}@unige.it

Keywords: Dictionary Based Image Pooling, Sparse Representation, Object Categorization, iCub, iCubWorld Data-Set.

Abstract: It is well known that image representations learned through ad-hoc dictionaries improve the overall results in object categorization problems. Following the widely accepted coding-pooling visual recognition pipeline, these representations are often tightly coupled with a coding stage. In this paper we show how to exploit ad-hoc representations both within the coding and the pooling phases. We learn a dictionary for each object class and then use local descriptors encoded with the learned atoms to guide the pooling operator. We exhaustively evaluate the proposed approach in both single instance object recognition and object categorization problems. From the applications standpoint we consider a classical image retrieval scenario with the Caltech 101, as well as a typical robot vision task with data acquired by the iCub humanoid robot.

1 Introduction

If, from a methodological point of view, image categorization is considered by many the very essence of computer vision, its applicative aspects are equally important. The possible application domains are countless and include industry, communications, entertainment, robotics, just to name a few. Not only object categorization is one of the hardest tasks of artificial intelligence, but also, in domains such automation and cognitive robotics, visual recognition is a cornerstone of very complex systems that include many other components — pose estimation, grasp, manipulation (Collet et al., 2011; Taylor and Kleeman, 2003; Ekvall et al., 2003; Gordon and Lowe, 2006). For these reasons, in the last decades the problem of designing effective visual representations for classification tasks has been given considerable attention. Since it is nowadays acknowledged recognition algorithms can be more effectively trained from examples than programmed, visual recognition has been tackled by both the computer vision and machine learning communities.

An important result of this joint effort are the so-called *hierarchical representations* which achieve remarkable performances in complex visual recognition tasks once they are used in combination with supervised learning algorithms – see for example (Lazebnik et al., 2006; Wang et al., 2010). Despite the good results obtained on benchmark and challenges,

the application of these approaches to real scenarios is still limited. The goal of this paper is to propose an effective image representation pipeline which is able to generalize to different contexts: from common computer vision datasets oriented to image retrieval, e.g. Caltech-101 (Fei-Fei et al., 2004), to real Human-Robot Interaction (HRI) scenarios (Fanello et al., 2013a).

A very influential method for representing the image content is the so-called Bag of Words (BoW) paradigm (Csurka et al., 2004) (also referred to as Bag of Keypoints) based on a vector quantization of local keypoints. This approach has been extended by the work of Lazebnik et al. (Lazebnik et al., 2006), which introduces the *Spatial Pyramid Representation* (SPR) to preserve the spatial configuration in images, and leads to a very popular framework within the visual recognition community.

In classification tasks, it is well known that the sparsity of data representations improves the overall classification accuracy (Fanello et al., 2013c; Viola and Jones, 2004; Huang and Aviyente, 2008; Destrero et al., 2009), therefore Yang et al. (Yang et al., 2009) improve the spatial pyramid pipeline by replacing the vector quantization procedure with a sparse coding step. Different extensions to (Yang et al., 2009) have been proposed in the recent literature (Boureau et al., 2010; Boureau et al., 2011; Feng et al., 2011; Jia et al., 2012; Russakovsky et al., 2012; Chen et al., 2012) — all these methods being based on an unsupervised

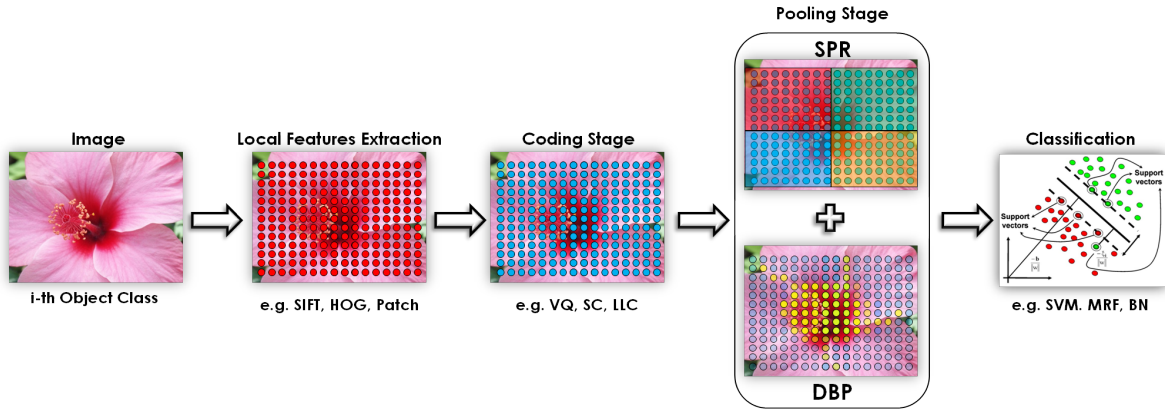


Figure 1: General pipeline for a visual recognition system. We contribute to the Pooling Stage, where we apply a Dictionary Based Pooling (DBP) operator.

learning of an ad hoc dictionary of atoms.

A recent line of research showed how dictionaries learned according to discriminative strategies may produce very effective image representations and should be used if labeled data are available. In (Kong and Wang, 2012; Fanello et al., 2013c) the discriminative strategies involve the coding stage of the pipeline. In this paper we show that discriminative dictionaries can be employed also during the pooling stage, yielding to image representations with an increased discriminative power. We start off from a low-level set of feature descriptors and we learn ad-hoc dictionaries in a discriminative manner. Then, we use these dictionaries to identify the region of the images belonging to a particular class of objects. The regions are then pooled together in order to obtain a compact and meaningful descriptor of the image.

The rest of the paper is organized as follows: in Section 2 we review the background of our work. In Section 3 we describe the method we propose; experiments, results and applications are presented in Section 4, while Section 5 is left to a final discussion.

2 Background

In this section we review a classification pipeline commonly used in literature for multi-class image recognition (Lazebnik et al., 2006; Yang et al., 2009; Boureau et al., 2011). This will set the basis to discuss the contributions of our approach.

2.1 Visual Recognition Pipeline

A general visual recognition pipeline based on the use of coding and pooling techniques can be divided in four main stages, as depicted in Fig. 1:

Local Features Extraction. The input image is first described with a set of local features $\{\mathbf{x}_i\}_{i=1}^M$. Very popular examples are image patches, SIFT (Lowe, 2004), or SURF (Bay et al., 2008) (either sparse or dense). Taking inspiration from (Fei-fei and Perona, 2005), in this work we compute SIFT descriptors on a regular grid of image locations, thus each image is represented with M descriptors $\mathbf{x}_i \in \mathbb{R}^d$, with $d = 128$.

Feature Coding. It is based on the use of a fixed or data-driven dictionary D of K atoms. The goal is to associate each image feature $\mathbf{x}_i \in \mathbb{R}^d$ with a code $\mathbf{u}_i \in \mathbb{R}^K$ estimated as:

$$\mathbf{u}_i = \arg \min_{\mathbf{u}} \|\mathbf{x}_i - D\mathbf{u}\|_F^2 + \lambda R(\mathbf{u}) \quad (1)$$

s.t. $C(\mathbf{u})$

where $\|\cdot\|_F$ is the Frobenius norm, and C is a (possible) constraint. Vector Quantization (VQ) (Lazebnik et al., 2006), Sparse Coding (SC) (Yang et al., 2009) and Locality-constrained Linear Coding (LLC) (Wang et al., 2010) are popular examples of coding methods, that mainly differ in the choice of regularization term $R(\mathbf{u})$ and constraints $C(\mathbf{u})$. Following (Fanello et al., 2013c), in this work we use Sparse Coding with ad-hoc dictionaries learned from the data (Sec. 2.2).

Feature Pooling. A common approach to overcome the locality of codes \mathbf{u}_i relies on the definition of a pooling operator g that combines the contributions of multiple image locations. Often, this operator takes the codes located at S overlapping regions (e.g. cells of the spatial pyramid), and for each region pools the information in a single vector $\phi_s \in \mathbb{R}^K$, $\phi_s = g_{(i \in Y_s)}(\mathbf{u}_i)$, where Y_s denotes the set of locations within the region s . The image is finally represented

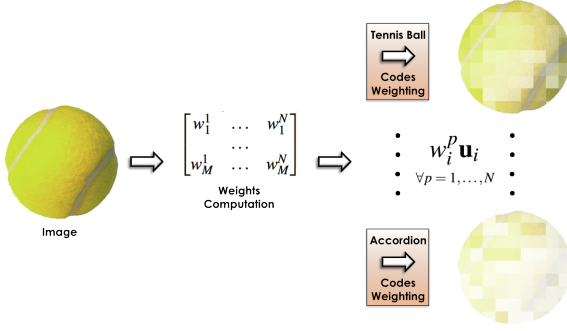


Figure 2: Visual intuition of the Dictionary Based Pooling operator. For each image we compute the weights of the N classes related to each code \mathbf{u}_i , $i = 1, \dots, M$. All the codes are weighted according to the considered class. The max pooling operator will select only relevant features for the considered image.

with a descriptor $\mathbf{z}_s \in \mathbb{R}^{K \times S}$ which is the concatenation of all ϕ_s . Examples of popular pooling operators are *average pooling* and *max pooling*. In this paper, we propose instead the use of a pooling operator which is guided by the discriminative dictionaries (Sec. 3).

Image Classification. The image descriptor is the input of a final classification step. It has been shown that sparse coding is very effective if combined with a linear classifier (Yang et al., 2009), leading to computationally efficient approaches. In what follows, we adopt a linear Support Vector Machines (Vapnik, 1998) following a one-vs-all strategy.

2.2 Discriminative Adaptive Sparse Coding

Our approach to sparse coding relies on learning discriminative dictionaries. We follow the method proposed in (Fanello et al., 2013c). In the remainder of this section we briefly recall the procedure, referring the interested reader to (Fanello et al., 2013c) for further details.

Let us consider a multi-class problem with N classes (objects) and let $\mathbf{X}^p = [\mathbf{x}_1, \dots, \mathbf{x}_{m^p}]$ be the $d \times m^p$ matrix whose columns are the training vectors of the p -th class. Also, let us define $\bar{\mathbf{X}}^p = [\mathbf{X}^1, \mathbf{X}^2, \dots, \mathbf{X}^{p-1}, \mathbf{X}^{p+1}, \dots, \mathbf{X}^N]$ to be the concatenation of the training matrices of all other classes $q \neq p$. Dictionary learning is based on the minimization of the functional:

$$E = \|\mathbf{X}^p - \mathbf{D}^p \mathbf{U}^p\|_F^2 + \|\bar{\mathbf{X}}^p - \mathbf{D}^p \bar{\mathbf{U}}^p\|_F^2 + \lambda \|\mathbf{U}^p\|_1 + \mu \|\bar{\mathbf{U}}^p\|_2 \quad (2)$$

with respect to \mathbf{D}^p , \mathbf{U}^p and $\bar{\mathbf{U}}^p$. \mathbf{D}^p is the $d \times K$ dictionary of class p , $\mathbf{U}^p \in \mathbb{R}^{K \times m^p}$ is the codes matrix of class p , while $\bar{\mathbf{U}}^p \in \mathbb{R}^{K \times \bar{m}^p}$, with $\bar{m}^p = \sum_{q=1}^N m^q$, $q \neq p$, are the coefficients related to all other classes. In essence, when learning the dictionary of class p , features belonging to it are constrained to have a sparse representation thanks to the l_1 -penalty term, while features of all other classes are forced to be associated with a more dense and smooth code vector. λ and μ are the regularization parameters allowing to control the importance of the two contributions.

As a consequence, features belonging to class p have a higher response if encoded with dictionary \mathbf{D}^p rather than any other dictionary \mathbf{D}^q , $q \neq p$, leading to a very discriminative representation.

3 Dictionary Based Pooling (DBP)

The use of feature dictionary is usually limited to the coding stage of the object classification pipeline. In this work, instead, we propose to extend their use to the pooling stage, exploiting their discriminative power.

Similarly to (Fanello et al., 2013c), we consider a global dictionary $\mathbf{D} = [\mathbf{D}^1, \dots, \mathbf{D}^N]$, of size $d \times K_G$, where $K_G = K \times N$, composed as the concatenation of all discriminative dictionaries previously computed. A feature $\mathbf{x} \in \mathbb{R}^d$ can be decomposed with respect to dictionary and codes as $\mathbf{x} \simeq \mathbf{D}\mathbf{u}$, with \mathbf{u} a K_G column vector.

We can interpret each element of code \mathbf{u} as the relevance of each atom in the linear combination. Since we know the correspondence between atoms of the dictionary and classes, \mathbf{u} can be seen as a concatenation of blocks, each one including the responses of a dictionary:

$$\mathbf{u}^T = [(\mathbf{u}^1)^T, \dots, (\mathbf{u}^N)^T]; \quad (3)$$

where \mathbf{u}^p is a K -ary vector representing the response of the p -th dictionary.

We evaluate the strength w^p of code \mathbf{u} with respect to class p as

$$w^p(\mathbf{u}) = \sum_{j=1}^K |u(j)^p| \quad (4)$$

where $u(j)^p$ denotes the j -th element of codes block of class p .

As observed in the previous section, highest values in \mathbf{u} , and consequently in w , should directly denote a particular affinity with the corresponding class. We thus adopt these measures as weights within a pooling operator working on a partition of the codes space $\{\mathbf{X}^p\}_{p=1}^N$, induced by the association of codes with



Figure 3: The iCubWorld 1.0 Dataset. Samples of the 7 classes collected for the *robot* (top strip) and *human* (bottom strip) datasets.

classes. Pooling is performed in each state of the partition according to the following

$$g_{(i \in X^p)}(\mathbf{u}_i) = \max_i (w_i^p(\mathbf{u}_i) \cdot \mathbf{u}_i) \quad \forall p = 1, \dots, N \quad (5)$$

The weight w_i^p represents a confidence measuring how likely is that the code \mathbf{u}_i has been observed in class p . Roughly speaking, they evaluate how much a given class is able to “see” in a particular image. Fig. 2 shows a visual representation of this principle. On the right, in particular, we report an image depicting a *tennis ball* as “seen” by its true class (above) and by the *accordion* class. Weights associated with the correct class are clearly higher.

For each image, we can finally build a representation $\mathbf{z}_n \in \mathbb{R}^{K_G \times N}$ that is the concatenation of all the weighted responses followed by the max pooling operator.

Combining the Spatial Layout and the Dictionary Based Pooling. The spatial pyramid representation leads to an image descriptor $\mathbf{z}_s \in \mathbb{R}^{K_G \times S}$, with S the number of the pyramid cells (see Sec. 2.1), while the proposed DBP generates a descriptor $\mathbf{z}_n \in \mathbb{R}^{K_G \times N}$. The final image representation \mathbf{z} will be the concatenation of the two vectors: $\mathbf{z} = [\mathbf{z}_s, \mathbf{z}_n] \in \mathbb{R}^{K_G \times (S+N)}$. It is common practice to normalize the data before classification, and as a consequence the descriptors become more peaky around zero. It has been experimentally observed the benefit of using a power normalization (Perronnin et al., 2010). Each component of both \mathbf{z}_s and \mathbf{z}_n are exposed to the following power normalization:

$$\begin{aligned} \mathbf{z}_s &= \text{sign}(\mathbf{z}_s) |\mathbf{z}_s|^\alpha \\ \mathbf{z}_n &= \text{sign}(\mathbf{z}_n) |\mathbf{z}_n|^\alpha \end{aligned} \quad (6)$$

where $0 \leq \alpha \leq 1$, in our experiments we set $\alpha = 0.5$. This is basically an explicit mapping to another feature space, where the highest code responses have less impact in the descriptor.

4 Experiments

In this section we validate the proposed dictionary based pooling method. We consider three datasets:

iCub World 1.0, iCubWorld Categorization¹ and a subset of the Caltech-101 (Fei-Fei et al., 2004). We compare our approach with state of the art methods (Yang et al., 2009; Fanello et al., 2013c), with the goal of showing that our pooling stage can improve the overall performances. We denote with:

- **SC** the method in (Yang et al., 2009).
- **SC + DASC** the approach proposed in (Fanello et al., 2013c).
- **DBP** (SC + DASC + Dictionary Based Pooling) the method described in Sec. 3.

4.1 Implementation Details

We provide here the details concerning the system parameters. As for the local feature extraction, we extract fixed-scale SIFT on patches of size 16×16 pixels, centered on a fixed grid every 8 pixels.

In the coding stage we set the global dictionary size K_G to 1024, while each class dictionary has $K = \frac{K_G}{N}$ atoms. In this way we ensure a fair comparison with the baseline methods, i.e. all the image representations have the same size. The regularization parameters λ and μ of Eq. 2, and the cost parameter C of SVMs have been selected with a 5-fold cross validation on the training set ($\mu = 0.15$ and $\lambda = 0.1$).

4.2 iCubWorld 1.0

We first evaluate the proposed method in a real Human-Robot Interaction (HRI) setting, where the goal is to recognize single instance of objects. The dataset we refer to has been acquired with the iCub humanoid robot (Metta et al., 2008), and is composed of 7 classes with 500 frames per class, for both the training and the test phase respectively.

Acquisitions have been made with respect to two different modalities, the **Robot Mode** and the **Human Mode** (Fanello et al., 2013a; Fanello et al., 2013b) (see Fig. 3). The Robot Mode dataset contains images acquired by iCub while handling an object of interest. The robot moves the arm in order to observe the object from multiple points of view. The Human Mode dataset contains images depicting a human actor holding one of the seven objects in his hand and showing it to the robot. The robot actively tracks the object, which is presented to the robot from multiple points of view.

The recognition has been performed per frame, temporal information is not used. The results we obtained

¹The iCubWorld 1.0 and iCubWorld Categorization Datasets can be downloaded from <http://www.iit.it/en/projects/data-sets.html>

Table 1: Accuracy results for the iCubWorld 1.0 Dataset, for both Robot Mode (RM) and Human Mode (HM). We show results when no pyramid is used (No SPM) and with 3-level pyramid (SPM).

	Method	Accuracy RM	Accuracy HM
No SPM	SC	70.65%	66.83%
	SC + DASC	76.00%	69.57%
	DBP	81.82%	77.57%
SPM	SC	84.11%	75.44%
	SC + DASC	84.33%	77.73%
	DBP	86.04%	80.97%



Figure 4: The iCubWorld Categorization dataset. It contains 10 classes acquired with a HRI scheme.

are summarized in Tab. 1 and show how, in this first robotics scenario, dictionary based pooling boosts the performances of the reference methods.

4.3 iCubWorld Categorization

For the second experiment we used a recent object categorization dataset acquired with a HRI setting (Fanello et al., 2013a). The modalities of the acquisition are similar to iCubWorld 1.0, but the focus is on object categorization. It comprehends 10 object categories of different complexity with respect to shape and textures. For each category 3 objects instances of 200 frames each are used for training and 200 frames are used for the testing phase for each new object instance. The particular complexity of this dataset is due to the presence of structured clutter, mean-

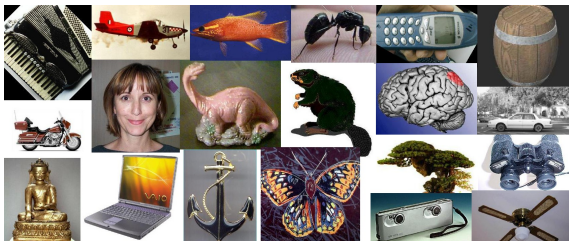


Figure 5: The selection of 20 classes from the popular Caltech-101 dataset, that we considered within the object categorization experiments.

Table 2: Accuracy results for the iCubWorld Categorization Data-Set. We show results when no pyramid is used (No SPM) and with 3-level pyramid (SPM).

	Method	Accuracy
No SPM	SC	38.07%
	SC + DASC	39.37%
	DBP	43.51%
SPM	SC	44.01%
	SC + DASC	44.89%
	DBP	49.28%

Table 3: Accuracy results for the 20 classes of the Caltech-101. We show results when no pyramid is used (No SPM) and with 3-level pyramid (SPM).

	Method	Accuracy
No SPM	SC	64.55%
	SC + DASC	66.81%
	DBP	73.62%
SPM	SC	76.95%
	SC + DASC	84.43%
	DBP	86.24%

ing that the context/background does not improve the recognition performances and it cannot be exploited as in standard image retrieval data-sets (Fanello et al., 2013a). In Tab. 2 we show the results for the categorization test ($T3$ test in (Fanello et al., 2013a)). Even in this challenging data-set the proposed approach outperforms the baseline methods.

4.4 Caltech-101

Finally we show that our method well generalizes also to standard computer vision dataset oriented to image retrieval problems. For this test we used a selection of 20 classes from the very popular *Caltech-101* dataset (Fei-Fei et al., 2004). The classes are the same used in (Fanello et al., 2013c) and are depicted in Fig. 5. We followed the standard evaluation procedure: for each class we used 30 of the available images as training set, while the others have been used for the test phase (max 50 per class). Again even in absence of the spatial pyramid, our method greatly improves the overall accuracy. With a 3-level pyramid combined with the DBP we obtain a substantial gain in the final accuracy. Tab. 3 summarizes the results.

5 Discussion

In this work we dealt with the widely accepted coding-pooling pipeline for visual recognition and

proposed a pooling method guided by the use of discriminative dictionaries. We considered a typical multi-class scenario and learned a dictionary for each object class. Then, we used local descriptors encoded with the learned atoms to guide the pooling stage: we designed a pooling operator making use of weights directly obtained from the coded descriptors. We performed an extensive evaluations of the method in both single instance object recognition and object categorization problems, and stressed the representation we proposed considering a classical image retrieval scenarios – using the very popular Caltech 101 – as well as on a typical robot vision task – with data acquired by the iCub humanoid robot. Results clearly speak in favor of our approach, showing that the dictionary based pooling strategy we proposed outperforms previous approaches. Our method is also computationally effective thanks to compactness of the description and usability with linear kernels.

ACKNOWLEDGEMENTS

This work was supported by the European FP7 ICT project No. 270490 (EFAA), project No. 270273 (Xperience) and project No. 288382 (Poeticon++).

REFERENCES

- Bay, H., Ess, A., Tuytelaars, T., and Vangool, L. (2008). Speeded-up robust features. *CVIU*, 110:346–359.
- Boureau, Y.-L., Bach, F., LeCun, Y., and Ponce, J. (2010). Learning mid-level features for recognition. In *CVPR*.
- Boureau, Y.-L., Le Roux, N., Bach, F., Ponce, J., and LeCun, Y. (2011). Ask the locals: multi-way local pooling for image recognition. In *ICCV*.
- Chen, Q., Song, Z., Hua, Z., Y., H., and Yan, S. (2012). Hierarchical matching with side information for image classification. In *CVPR*.
- Collet, A., Martinez, M., and Srinivasa, S. S. (2011). The MOPED framework: Object Recognition and Pose Estimation for Manipulation. *The International Journal of Robotics Research*.
- Csurka, G., Dance, C., Fan, L., Willamowski, J., and BrayLixin, C. (2004). Visual categorization with bags of keypoints. In *In Workshop on Statistical Learning in Computer Vision, ECCV*.
- Destrero, A., De Mol, C., Odone, F., and A., V. (2009). A sparsity-enforcing method for learning face features. *IP*, 18:188–201.
- Ekvall, S., Kragic, D., and Hoffmann, F. (2003). Object recognition and pose estimation using color cooccurrence histograms and geometric modeling. In *Image Vision Computing*.
- Fanello, S., Ciliberto, C., Santoro, M., Natale, L., Metta, G., Rosasco, L., and Odone, F. (2013a). icub world: Friendly robots help building good vision data-sets. In *CVPRW*.
- Fanello, S. R., Ciliberto, C., Natale, L., and Metta, G. (2013b). Weakly supervised strategies for natural object recognition in robotics. *ICRA*.
- Fanello, S. R., Noceti, N., Metta, G., and Odone, F. (2013c). Multi-class image classification: Sparsity does it better. *VISAPP*.
- Fei-Fei, L., Fergus, R., and Perona, P. (2004). Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *CVPRW*.
- Fei-fei, L. and Perona, P. (2005). A bayesian hierarchical model for learning natural scene categories. In *CVPR*, pages 524–531.
- Feng, J., Ni, B., Tian, Q., and Yan, S. (2011). Geometric lp-norm feature pooling for image classification. In *CVPR*, pages 2609–2704.
- Gordon, I. and Lowe, D. (2006). What and where: 3d object recognition with accurate pose. In *Lecture Notes in Computer Science*.
- Huang, K. and Aviyente, S. (2008). Wavelet feature selection for image classification. *IP*, 17:1709–1720.
- Jia, Y., Huang, C., and Darrell, T. (2012). Beyond spatial pyramids: Receptive field learning for pooled image features. In *CVPR*, pages 3370–3377.
- Kong, S. and Wang, D. (2012). A dictionary learning approach for classification: separating the particularity and the commonality. In *ECCV*.
- Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, volume 2, pages 2169–2178.
- Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110.
- Metta, G., Sandini, G., Vernon, D., Natale, L., and Nori, F. (2008). The icub humanoid robot: an open platform for research in embodied cognition. In *8th Work. on Performance Metrics for Intelligent Systems*. Website: <http://www.icub.org>.
- Perronnin, F., Sánchez, J., and Mensink, T. (2010). Improving the fisher kernel for large-scale image classification. In *ECCV*.
- Russakovsky, O., Lin, Y., Yu, K., and Fei-Fei, L. (2012). Object-centric spatial pooling for image classification. In *ECCV*.
- Taylor, G. and Kleeman, L. (2003). Fusion of multimodal visual cues for model-based object tracking. In *ACRA*.
- Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley and Sons, Inc.
- Viola, P. and Jones, M. (2004). Robust real-time face detection. *IJCV*, 57:137–154.
- Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., and Gong, Y. (2010). Locality-constrained linear coding for image classification. In *CVPR*.
- Yang, J., Yu, K., Gong, Y., and Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*.

Ranking the Good Points: A Comprehensive Method for Humanoid Robots to Grasp Unknown Objects

Ilaria Gori, Ugo Pattacini *IEEE Member*, Vadim Tikhonoff and Giorgio Metta *IEEE Senior Member*

Istituto Italiano di Tecnologia

Via Morego 30, 16163, Genova, Italy

{ilaria.gori, ugo.pattacini, vadim.tikhonoff, giorgio.metta}@iit.it

Abstract—We propose a grasping pipeline to deal with unknown objects in the real world. We focus on power grasp, which is characterized by large areas of contact between the object and the surfaces of the palm and fingers. Our method seeks object regions that match the curvature of the robot’s palm. The entire procedure relies on binocular vision, which provides a 3D point cloud of the visible part of the object. The obtained point cloud is segmented in smooth surfaces. A score function measures the quality of the graspable points on the basis of the surface they belong to. A component of the score function is learned from experience and it is used to map the curvature of the object surfaces to the curvature of the robot’s hand. The user can further provide top-down information on the preferred grasping regions. We guarantee the feasibility of a chosen hand configuration by measuring its manipulability. We prove the effectiveness of the proposed approach by tasking a humanoid robot to grasp a number of unknown real objects.

I. INTRODUCTION

Object manipulation is a crucial skill for humanoid robots as it often enables more complex tasks such as human-robot interaction or the development of service robots that work in unstructured environments. The problem of robot grasping has been widely studied in the literature and is still an active research field. Obtaining reliable grasps is hard because of the dimension of the configuration space, the difficulty in retrieving accurate visual priors as well as the difficulty in predicting contact forces. We aim at attacking some of these problems through a complete framework capable of relating the object visual properties to effective hand configurations. Behavioral studies have demonstrated the reliance of human grasping on 3D shape information [1]. Neurophysiologists have also postulated the existence of a behavioral vocabulary that connects 3D shape to action prototypes [2], highlighting the importance of 3D information to generate effective grasps. Inspired by these ideas, we decided to exploit stereo vision to retrieve 3D cues about the object before grasping. A full 3D model of the object is typically not available, as the robot can only estimate the shape from a single viewpoint unless we embark in relatively complicated active exploration procedures. Even when complete models of known objects are provided directly by the user, the robot may fail to recognize them or will perceive new ones while exploring the environment. In all these cases the robot will be able nevertheless to retrieve 3D information from a single view of the object. Therefore, to obtain reliable grasps on unknown objects, we have to deal

with incomplete 3D point clouds.

In Napier’s taxonomy [3], grasp actions are divided into power grasps and precision grasps. Power grasp is characterized by large areas of contact between the object and the surfaces of the palm and fingers, and by little or no ability to perform further movements with the fingers. Therefore if an object is meant to be grasped stably, power grasp is usually chosen. Differently, if sensitivity and dexterity are required, precision grasp is preferable, whereby the object is held with the tips of the fingers. When the task to perform is specific to the object, as e.g. in tool use, the fingers need to be placed in certain specific locations. In this case precision grasp is the most flexible choice. On the other hand, when the object is unknown, we cannot define complex object-specific tasks as in e.g. requiring the robot to use a tool as in hammering, cutting, pushing, etc. The only possible goal when dealing with unknown objects is therefore to grasp them efficiently under some stability condition. Since we are dealing with unknown objects, in the following we focus on power grasp only.

Most of the algorithms that tackle the grasping problem analyze and evaluate finger configurations to rank them as a function of the expected grasp quality. However it has been recently highlighted in [4] and showed in [5] that spreading the fingers enclosing the object against the palm allows the hand to grasp arbitrary objects, and that the thumb opposition is very important to obtain stable grasps. As we are interested in power grasp, we can exploit this suggestion restricting our search to the most suitable area of the object on which to place the palm. We carry out this analysis through vision, guaranteeing that the palm’s shape matches the selected object surface, so that even if the fingers close on an occluded region, the grasp is likely to be stable. To do so, we exploit information on the geometry of the object surface.

The relevant parameters of a successful grasp are the “where and how” to grasp the object. In order to select them, we exploit vision and kinematics information: vision serves to localize the most promising surfaces on the object (“where”), while kinematics determines the feasibility of a given configuration (“how”).

The main contribution of this work is the idea that matching surface curvature of the object with the curvature of the robot hand is sufficient to obtain effective grasps. We demonstrate this claim by implementing and validating a complete computational grasp pipeline on the iCub robot [6], in a scenario involving objects of different shape and size.

*This work was supported by the European FP7 ICT project No. 270490 (EFAA), project No. 270273 (Xperience) and project No. 288382 (POETI-CON+).

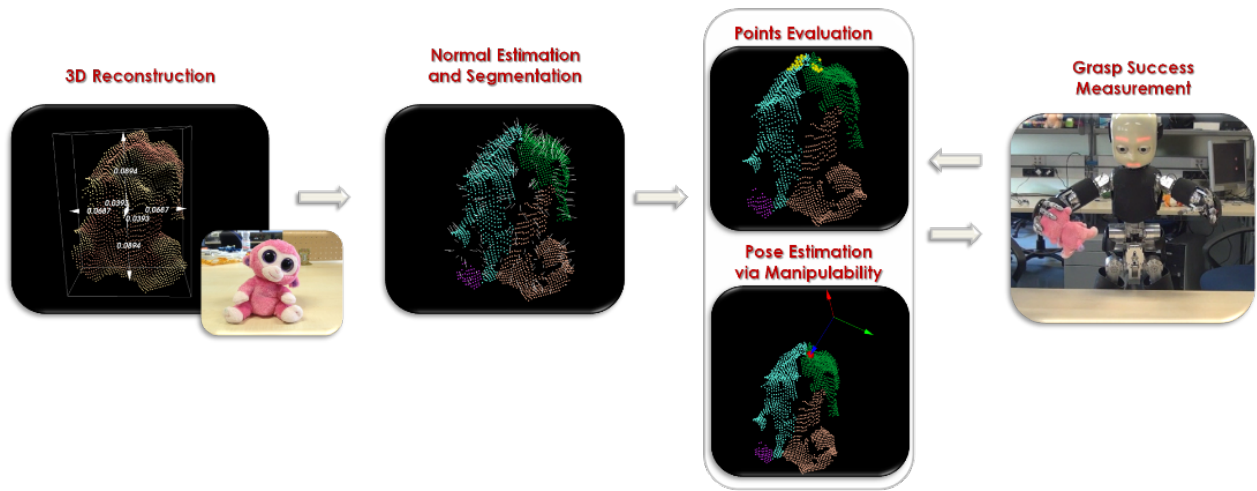


Fig. 1. The figure illustrates the entire pipeline. We reconstruct the object in 3D obtaining a point cloud, which we then segment. We extract surface normals and isolate connected smooth regions. We then rank the best points on the basis of a composite score function taking into account the object shape and size, and eventually including top-down information. Subsequently the best end-effector position and pose are estimated on the basis of the robot manipulability measure. If the grasp is successful, the score function is adjusted by updating the coefficients of an incremental Least-Square Support Vector Machine.

II. RELATED WORK

In general, robotic manipulation can be decomposed into a number of subtasks, such as the identification of the object, localization of the best grasping points on the basis of the object shape, inverse kinematics, hand preshape computation, force-closure fulfillment, adjustment of the position of the hand and finally the evaluation of exerted forces in relation to visual and haptic feedbacks. Making sure that all these subtasks are effective and they do not interact catastrophically is not easy. Not surprisingly, many of the systems in the literature focus on specific aspects of the problem, as for example, assuming that the contact points and normals are known [7] or by looking for “graspable” regions [8] without concerns of the feasibility of the hand’s configuration. Conversely, we provide a complete pipeline that spans the domain from data acquisition to the actual grasping procedure.

Especially in the past, the literature addressed the grasping problem only in terms of force-closure and form-closure, looking for specific conditions on the contact wrenches that assured a certain hand configuration to firmly hold any object; according to [9], these approaches are called ‘analytical’ [10] [11].

The analytical approaches though, usually assumed that contact point locations were given without explicitly relating the hand configuration to the object geometry. Lately in this regard, more recent applications attempt to get around the analytical approach limitations; these methods are called ‘empirical’ [9] and our method belongs to this category.

Among the most interesting empirical approaches, a few methods create a direct mapping between object shape and hand pose [12], [13]. Other techniques generate a certain number of grasp hypotheses on the basis of specific heuristics, and then evaluate them with machine learning algorithms such as Artificial Neural Networks [14] [15], simple maximum likelihood algorithms [16], or kernel density estimation methods [17]. Others assume a fixed number of possible grasp configurations, and associate them directly to specific shape

properties [18] [19] [20]. All of them look for a suitable hand configuration in terms of palm and fingers position. On the contrary we exploit the work in [5], and we limit our search to a suitable palm configuration (position and orientation). We select the best palm pose on the basis of both visual and kinematic cues.

Among the approaches focusing on power grasp, the most similar to our work are Detry et al. [18], Boularias et al. [21], Saxena et al. [8], Li et al. [22] and Roa et al. [4]. The first three mentioned methods ([18], [21] and [8]) are comparable to our framework as they retrieve a good end-effector position and orientation without taking into account the finger positions. However our approach exploits local surface curvatures to find the most suitable region for a specific robotic palm. Li et al. [22] starts from a similar concept by hypothesizing that the shape of an object has to match the shape of the hand; however Li and colleagues start from a fixed hand preshape, therefore they do not actively search for a suitable palm configuration with respect to the given object. Roa [4] uses the size of the hand as the main feature to match the object’s surfaces. This approach is usable only on complete 3D models, whereas our method has been thought of specifically to deal with incomplete 3D point clouds.

In the following, we thus describe a new method to match the local curvature of the object to the surface of the robot’s palm. We exploit the insight that, in order to obtain a reliable grasp, the palm has to adapt to the surface of the object with the most similar curvature. We further illustrate the complete grasp pipeline starting from data acquisition to the actual grasping procedure, exploiting along the way the incomplete 3D point clouds obtained from stereo vision. We show how we evaluate the appearance of the object to find candidate grasping points and subsequently retrieve a feasible hand configuration based on the robot’s kinematics. Not less importantly, we show that our method is fast and reliable, and can be easily applied to real world scenarios.

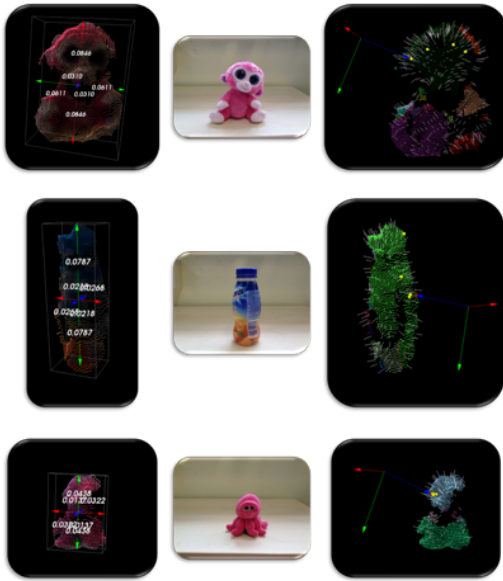


Fig. 2. Results for three different objects. Left column: 3D incomplete point clouds with the minimum enclosing bounding box. Middle: object. Right: segmented point cloud. The yellow dots are the best points after ranking, whereas the red dot is the selected point. The palm/hand orientation is shown where the red, green and blue arrows stand respectively for the x , y and z axis of the robotic hand (see Fig. 3).

III. EXTRACTING GRASPING POINTS

Before deciding how to grasp an object, we need to define where to grasp it. Usually the answer to this problem is not unique; in fact, if one has to lift an object, he can put his hand in several different positions. If we limit our analysis to the power grasp, then the number of possible locations gets smaller, but still, there is no a universally accepted rule on where to take an object. Several factors influence how a person performs a grasp [23]; some of them regard the object shape and dimension, others regard the weight of the object and its surface roughness as well as the task at hand. In our implementation of the grasp pipeline we take into account some of these factors in the process of extracting a set of significant points on the object surface. We first create a 3D point cloud of the visible part of the object (from a single viewpoint), using the stereo vision system of our humanoid robot – the iCub [6]. We subsequently compute a minimum bounding box enclosing the point cloud, estimating the approximate dimension and orientation of the object with respect to the robot’s root frame. Unsupervised learning techniques are employed to segment the reconstructed cloud in smooth regions. We finally look for the regions that best approximate the robotic palm’s curvature. As shown in [4] and [5], spreading the fingers and enclosing the object against the palm significantly helps in obtaining a stable grasp. Hence we limit our search to the most compatible surfaces under the criterion that they have to match the palm size and curvature. Firstly we guarantee that the hand lies in a visible region, therefore we select, among the obtained smooth regions, those large enough as compared to the size of the palm. We then apply a uniform sampling on the selected clusters of points, retrieving a smaller number of points along with their normals. Each point here represents the center of a planar region computed on the point’s neighborhood with an

area similar to the area of the robot’s palm. This set of points is ranked with the help of a score function, which takes into account the local shape properties around the points, as well as simple heuristics on the object dimension. This defines the best regions from where to extract grasping points. The user can also provide top-down information to bias the point selection process. Since vision is not enough to ensure a stable grasp, we select a set of N points that got the highest scores, and finally we pick a feasible hand configuration on the basis of the robot manipulability. The complete pipeline is schematized in Fig. 1.

A. Reconstructing and segmenting the point cloud

Three-dimensional information can significantly improve the quality of robotic grasps, as it enables a more precise estimation of cues such as surface curvatures and normals. We rely on stereo vision algorithms in order to retrieve such 3D information. We use the Hirschmuller algorithm [24] to estimate the depth map, and we project each pixel of the object in the 3D space. We then estimate a minimum bounding box enclosing the point cloud, in order to obtain the approximate dimension of the object. We employ a technique based on the convex hull of the point cloud, which is analyzed using rotating calipers algorithms [25]. The next step selects where to place the end effector. Here we would like to guarantee that the hand lies in a region that is large enough with curvature similar to that of the palm. We employ the *Region Growing Segmentation* [26] that segments the object point cloud into a set of smooth connected regions. This method starts with the computation of surface normals as an estimation of the normal of a plane tangent to the surface passing by each point. This is obtained through a least-square fitting on each point’s neighborhood [27]. Given a point p and its neighborhood P^k , the plane tangent to the surface can be defined as a couple (x, \bar{n}) , where x is a point of the plane and \bar{n} is the normal to the plane. We define the neighborhood P^k of a point p as the set of points that lies within a circular area having radius equal to the radius of the robotic palm. The distance between a point $p_i \in P^k$ and the fitting plane can be expressed as

$$dist_i = (p_i - x)\bar{n}; \quad (1)$$

in order to compute the plane parameters, we need to minimize the distance $dist_i$ for each point. If we impose that x is the centroid of the neighborhood (i.e. $x = \frac{1}{k} \sum_{i=1}^k p_i$), then the solution for \bar{n} can be calculated by analyzing the eigenvectors and eigenvalues of the covariance matrix:

$$C = \frac{1}{k} \sum_{i=1}^k (x - p_i)(x - p_i)^T. \quad (2)$$

Once the normals of all points have been computed, a seed point p is chosen and every point $p_i \in P^k$ is evaluated; p_i will be added to the current cluster only if it is locally connected to the seed p , and if the angle between the normals of p and p_i is smaller than a specified threshold, otherwise it is added to the list of potential seeds. The point cloud is thus subdivided into several regions having similar curvatures. Later in order to assure grasp stability, we select only the regions that contain a sufficient number of points. This way we effectively impose the condition that the hand is placed on a smooth and large enough surface.

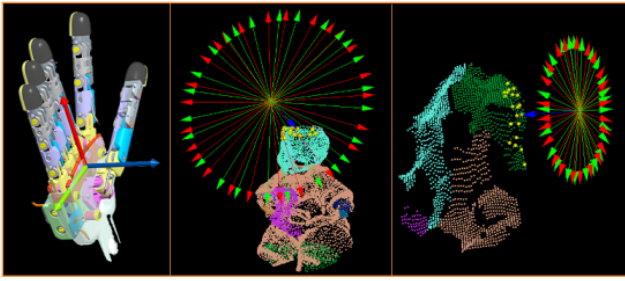


Fig. 3. Sampled orientations for a single point. The leftmost panel illustrates the iCub end effector coordinate system: red is x , green is y and blue is z . The same representation is employed in the middle and right panel: the hand z axis is coded by a blue arrow, and the red and green arrows represent the x and y hand axes sampled on the plane perpendicular to z . The middle panel shows a synthetic and complete point cloud as taken from the *KITObjectModels WebDatabase* [28], whereas the rightmost plot is an incomplete point cloud taken directly from the iCub stereo vision system.

B. Points evaluation

Appropriate end effector positions are ranked by means of a score function which biases those with specific characteristics. We choose the N points with the highest score as returned by a function that weighs the object shape and dimension:

$$s(p) = w_1 \cdot v(p) + w_2 \cdot m(p) \quad (3)$$

where $v(p)$ is an auto-adaptive function representing the evaluation of visual properties at the point p , and $m(p)$ is a fixed component that depends on the object dimension. $m(p)$ can integrate a user-defined task component. w_1 and w_2 are relative weights which are chosen empirically; they are particularly useful for balancing the two contributions.

1) *Visual Component*: The first part of the score function takes into account the shape of the object. In particular, we would like to grasp the object on a point which lies on a surface having a curvature similar to the curvature of the robot's palm, so that the hand can adapt on the object. Since good curvature values are not immediately computable, we use machine learning to approximate a relation between the robot's hand curvature and the curvature of a surface centered in a point p . In order to ensure that the hand will adapt on the object, the neighborhood of the evaluated point p will have the same area as the robot's hand. We relate the local curvature of the surface to a grasp success measure g , which is evaluated on the basis of the robot's own exploration. In particular, the grasp success measurement is a binary value (0 or 1), and is provided by a grasp detector mechanism. To achieve such detection we exploit the intrinsic elasticity of the iCub fingers; in particular we employ a technique that retrieves a measure of contact occurring on the distal phalanges by comparing the actual joint position θ_j with the prediction $\hat{\theta}_j$ provided by a linear model of the joint actuation, given as input the motor position $\theta_{m(j)}$ [29]. High discrepancy between the feedback and the prediction corresponds to increasing external pressures; in this case we assign $g = 1$, otherwise $g = 0$. In this respect, we were able to significantly improve the quality of the detection by replacing the simple linear representation of the map $\hat{\theta}_j = M(\theta_{m(j)})$ with a map learned using Least Square Support Vector Machine (LSSVM) [30]; the training phase was carried out over a set of input-output pairs acquired while the

joints can move freely in the space without any contact with the environment. This online evaluation leads to a cycle where the score function is continuously updated on the basis of the robot experience: at each grasp, the pair (c, g) composed of the curvature $c \in \mathbb{R}$ computed on the surface centered at the grasping point p and the grasp success measure $g \in \mathbb{R}$ is added to the training set. We use Least-Square Support Vector Machine [30] with Radial Basis Function Kernel to learn the map between the curvature and the success of a grasp.

2) *Modality Component*: The location of the grasp point on the object is an important element in choosing the grasping location. For example, if the object has one dimension much larger than the others, then selecting a point along the larger dimension increases the chances of a stable grasp. The rationale is that grasping an object that is too large for the hand is doomed to fail. On the contrary, if the object is too small, then it would be better to place the hand on the top of it. Following these considerations, we define three modality-specific biases, which thus assign higher scores to the points that respectively lie in the top, right, or left regions with respect to the robot's root frame.

Objects have also specific affordances, hence it is reasonable to assume that the grasping mode depends on the task at hand. Since we are dealing with power grasp of unknown objects, we cannot define complex object-specific affordances. We have to content ourselves with generic task biases as for instance taking an object to give it to a person, or taking the object to explore it as for learning tactile classification. To this aim, we can simply analyze the position of the point with respect to the rest of the object. For instance, if the task is to pass an object to a person waiting with her hand open, palm up, we can assume that the end-effector position is better located on the top part of the object. We leave this choice to a user-tunable parameter in our score function.

In addition, we would like the hand to reach far from the border of the visible portion of the object since the computation of the surface normals tends to be noisier at the borders. This condition is easily satisfied by privileging points that lie far from the corner points of the minimum bounding box. In summary, given π_j , $j = 1, \dots, nc = 8$ corner points of the minimum enclosing bounding box, the preference for points on the top part of the object can be formulated as follow:

$$t(p) = \sqrt{|p_z - c_z|/dim_z} + \sum_{j=1}^{nc} \frac{\|p - \pi_j\|}{nc}, \quad (4)$$

where p_z is the z component of the point p , c_z is the z component of the center of the object, and dim_z is the dimension of the object along the z axis with respect to the robot's root frame.

IV. GRASP PARAMETER ESTIMATION

Once a number of suitable candidate points has been computed as illustrated earlier, we have to determine the best grasping point and the corresponding pose of the robot's end effector. This last step has to take into account the robot kinematics in order to ensure a feasible grasp. For each candidate point, we evaluate a set of possible orientations in terms of their manipulability index; given this measure, the most suitable point and orientation of the end-effector is selected.

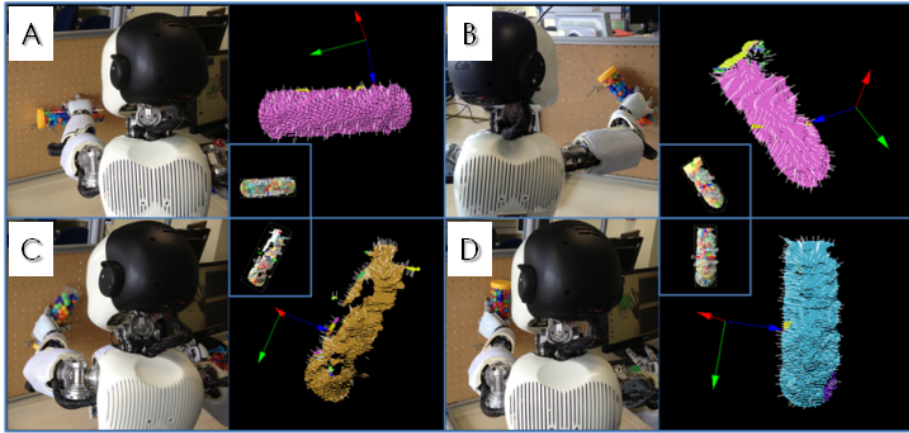


Fig. 4. Four different experiments with the same object. The object – a cylindrical container – is rotated respectively by 0 (A), -45 (B), 45 (C) and 90 (D) degrees. The iCub on each quadrant. The rightmost plot on each case shows the segmented object. The color coding correctly shows that a single region is detected. Grasping is successful in all cases.

We would like the hand to be parallel to the surface at the contact point: given the robot kinematics depicted in Fig. 3 (left), where the RGB color convention (i.e. red-green-blue) represents the x , y , and z axes respectively, we ask that the z axis of the end effector is parallel to the surface normal computed at the point under evaluation and directed opposite-wise. We then sample the plane determined by the z axis and passing through the point p by identifying n possible orientations for the x and y axes (see Fig. 3). We make use of *Ipopt* [31] to solve the inverse kinematics resulting in the joint configuration that satisfies the desired position and orientation of the hand using 10 degrees of freedom of the robot (7 for the arm and 3 for the torso). Notably, using the algorithm in [32], we find a reliable solution in only about 0.04 seconds, and can consequently explore hundreds of possible robot configurations in a handful of seconds. Each resulting joint configuration is evaluated using the standard manipulability measure [33]:

$$w(\boldsymbol{\theta}) = \sqrt{\det(J(\boldsymbol{\theta})J(\boldsymbol{\theta})^T)} \quad (5)$$

where $\boldsymbol{\theta}$ is the current joint configuration, and J is the Jacobian matrix. Such measure is further improved with a penalty term [34] that considers the distance from the joint limits:

$$P(\boldsymbol{\theta}) = 1 - \exp\left(-k \prod_{j=1}^{nj} \frac{(\theta_j - l_j^-)(l_j^+ - \theta_j)}{(l_j^+ - l_j^-)^2}\right) \quad (6)$$

where θ_j is the current position of the j -th joint, l_j^+ is the j -th joint upper limit, l_j^- is the j -th joint lower limit and k is a scaling factor that weights the behavior of the measure near joint limits. Summarizing, we aim at finding a suitable position and orientation of the end-effector, such that the associated joint configuration $\boldsymbol{\theta}$ maximizes the following quantity:

$$\arg \max_{\boldsymbol{\theta}} (w(\boldsymbol{\theta}) + P(\boldsymbol{\theta})). \quad (7)$$

The manipulability measure, combined with the direction of the normal, also defines the most suitable hand that should be used for a given grasp.

V. EXPERIMENTS

We validate our framework by conducting three different experiments. We start with a qualitative experiment, where we show that the same object, rotated by different amounts, can still be grasped reliably. Then we demonstrate that we can learn the relation between the curvature of an object region and its influence on a successful grasp. We finally perform a large number of grasp actions on several objects lying in different positions with respect to the robot, and we show that the robot can grasp them with a high success rate. All the experiments were conducted on the iCub, a 53 degrees of freedom humanoid robot developed by the RobotCub project [6]. In the context of these experiments we used 19 degrees of freedom (DOF) in total, considering the 3 DOFs of the torso along with the 7 DOFs of the arm and 9 DOFs of the hand.

A. Qualitative results - Rotating object

To demonstrate that our system is robust against objects rotations, we run a qualitative test using an elongated cylindrical container as shown in Fig. 4, placed at four different orientations with respect to upright direction: 0 , -45 , 45 , and 90 degrees. It turns out that the modality component of the score function correctly rewards points that are lateral to the cylinder principal axis in the tested cases. As result, iCub adapts the grasp action accordingly (see Fig. 4).

B. Quantitative results

1) *Evaluating the Learned Map:* The validation of the learning procedure and the resulting map between local surface curvature and the grasp success rate is carried out on a training set of 100 data points. These points are collected on the objects shown in Fig. 5. The 100 trials are performed imposing $w_2 = 0$, therefore only the curvature component is taken into account to evaluate grasp success. The trials are carried out choosing points with random curvature values in order to explore the whole function domain. Fig.5 shows the results of learning. The x and the y axes report, respectively, the explored curvatures normalized between 0 and 1, and the grasp success rate. From inspecting this function it is possible to

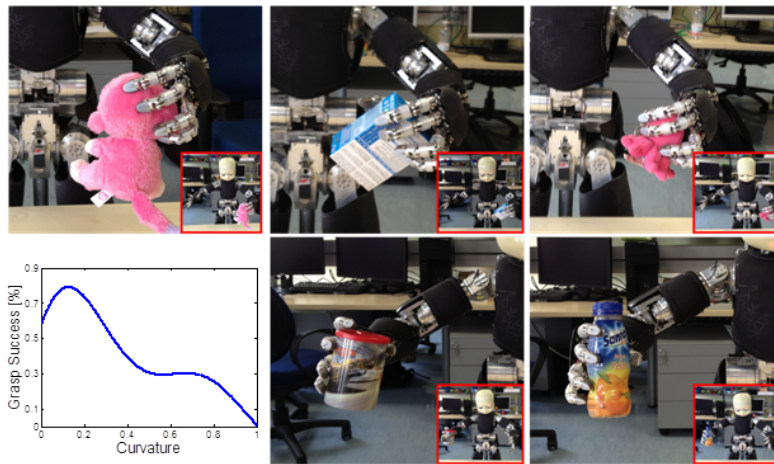


Fig. 5. Training set. Bottom-left: the learned map between object curvatures and success measure. Curvature is normalized between 0 and 1; these values represent respectively the minimum and the maximum curvatures explored by the robot. Points with curvatures in the range of 0.1 and 0.2 are preferable as they are likely to bring about successful grasps.

infer that surfaces with curvature values between 0.03 and 0.2 are suitable for the iCub’s palm, and usually lead to successful grasps. Conversely, surfaces too flat or with higher curvature tend to yield unsuccessful grasps. To verify whether the maximum of the function identifies critical curvature, we designed a dedicated experiment with two grasping sessions on the cylindrical container (Fig. 6), which presents both flat and curve surfaces. In the first session, we let the robot perform 30 grasps by choosing points with curvature close to the limits of unsuccessful grasp ($c > 0.3$ as in Fig. 5). In the second session, an additional set of 30 grasps is collected by rewarding points with curvature close to the maximum. We report a grasp success rate of 60% for the first session, and a significantly higher rate (90%) in the second session, proving that grasping performance considerably changes with respect to the curvature of the chosen point. Notably, a relatively small set of 100 samples is sufficient to learn the curvature map.

C. Evaluating the complete pipeline

The main objective of our work is that of endowing the iCub with grasping skills to operate in a real, generally unstructured, environments. This requires many practical adjustments to guarantee that the grasp actions are indeed reliable. To this end a quantitative evaluation of the overall system is needed to verify the performance of all components lumped together. As in [21], a grasp is considered as successful only if the object does not fall after being lifted. The experiment we present here also demonstrates that the curvature map (learned earlier) generalizes to novel objects. We execute 20 trials on each object in the test set (Fig. 6), i.e. 80 trials in total, achieving an overall success rate of 91.25%. Such accuracy (shown in details in Fig. 6) makes the framework suitable to be employed for robust manipulation tasks.

As grasping lacks a standardized benchmark, we compare our approach with a simple “top” grasp (grasping the object always from the top), which has been used e.g. in [35]. We tested this more stereotyped grasp on the same objects showed in Fig. 6 carrying out 20 trials per object as before. The success rate for the cylinder and the bottle was significantly lower (15% for both) quite obviously because of their elongated

shape that makes the top grasp unsuitable. We also achieved 65% for the dog and 80% for the cube. These results confirm the considerable performance gain of our complete grasping pipeline.

VI. CONCLUSIONS

We presented a complete framework that addresses the problem of grasping unknown objects in a humanoid robotic settings. We start from visual information obtained through stereo images. We exploit 3D information to acquire an incomplete point cloud, and we successively rank the best candidate grasping points of the cloud accounting for the object visual properties. In particular, we look for regions with curvature similar to that of the robot’s palm. A suitable score function is continuously updated with experience. We eventually choose the end effector position and orientation on the basis of the robot kinematics avoiding unsuitable configurations as defined by the robot’s manipulability index. We demonstrate that this score function accounts correctly for the visual and kinematic aspects of grasping. More importantly, the entire pipeline computes in a few seconds enabling its application in real scenarios. Numerical results support these conclusions.

REFERENCES

- [1] Y. Hu, R. Eagleson, and M. A. Goodale, “Human visual servoing for reaching and grasping: The role of 3-d geometric features.” *IEEE International Conference on Robotics and Automation*, 1999.
- [2] G. Rizzolatti and G. Luppino, “The cortical motor system,” *Neuron*, 2001.
- [3] J. Napier, “The prehensile movements of the human hand,” *The Journal of bone and joint surgery*, pp. 902–913, 1956.
- [4] M. Roa, M. Argus, D. Leidner, C. Borst, and G. Hirzinger, “Power grasp planning for anthropomorphic robot hands,” *IEEE International Conference on Robotics and Automation*, 2012.
- [5] M. Chalon, M. Grebenstein, T. Wimboeck, and G. Hirzinger, “The thumb: Guidelines for a robotic design,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.
- [6] G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori, “The icub humanoid robot: an open platform for research in embodied cognition,” in *8th Workshop on Performance Metrics for Intelligent Systems*, 2008.



Fig. 6. Objects used to test our method with their respective success rates over 20 trials.

- [7] C. Rosales, R. Suárez, M. Gabiccini, and A. Bicchi, "On the synthesis of feasible and prehensile robotic grasps," *IEEE International Conference on Robotics and Automation*, 2012.
- [8] A. Saxena, J. Driemeyer, and A. Ng, "Robotic grasping of novel objects using vision," *The International Journal of Robotics Research*, 2008.
- [9] A. Sahbani and S. El-Khouri, "An overview of 3D object grasp synthesis algorithms," *Robotics and Autonomous Systems*, 2011.
- [10] Y. Liu, "Qualitative test and force optimization of 3-D frictional form closure grasps using linear programming," *IEEE Transactions on Robotics and Automation*, 1999.
- [11] J. Li, H. Liu, and H. Cai, "On computing three-finger force-closure grasps of 2-D and 3-D objects," *IEEE Transactions on Robotics and Automation*, 2003.
- [12] R. Pelossof, A. Miller, P. Allen, and T. Jebara, "An SVM learning approach to robotic grasping," *IEEE International Conference on Robotics and Automation*, 2004.
- [13] A. Maldonado, U. Klank, and M. Beetz, "Robotic grasping of unmodeled objects using time-of-flight range data and finger torque information," *International Conference on Intelligent Robots and Systems*, 2010.
- [14] S. Geidenstam, K. Huebner, D. Banksell, and D. Kragic, "Learning of 2D grasping strategies from box-based 3D object approximations," *Robotics: Science and Systems Conference*, 2009.
- [15] L. Bodenhagen, D. Kraft, and M. Popović, "Learning to grasp unknown objects based on 3D edge information," *International Symposium on Computational Intelligence in Robotics and Automation*, 2009.
- [16] A. Saxena, L. Wong, and A. Ng, "Learning grasp strategies with partial shape information," *AAAI Conference on Artificial Intelligence*, 2008.
- [17] R. Detry, E. Başeski, M. Popović, Y. Touati, N. Krüger, O. Kroemer, J. Peters, and J. Piater, "Learning object-specific grasp affordance densities," *IEEE International Conference on Development and Learning*, 2009.
- [18] R. Detry, C. Ek, M. Madry, J. Piater, and D. Kragic, "Generalizing grasps across partly similar objects," *IEEE International Conference on Robotics and Automation*, 2012.
- [19] A. Miller, S. Knoop, H. Christensen, and P. Allen, "Automatic grasp planning using shape primitives," *IEEE International Conference on Robotics and Automation*, 2003.
- [20] M. Popović, D. Kraft, L. Bodenhagen, E. Başeski, N. Pugeault, D. Kragic, T. Asfour, and N. Krüger, "A strategy for grasping unknown objects based on co-planarity and colour information," *Robotics and Autonomous Systems*, 2010.
- [21] A. Boularias, O. Kroemer, and J. Peters, "Learning robot grasping from 3-D images with Markov Random Fields," *IEEE International Conference on Intelligent Robots and Systems*, 2011.
- [22] Y. Li, J. Saut, J. Cortés, T. Siméon, and D. Sidobre, "Finding enveloping grasps by matching continuous surfaces," *IEEE International Conference on Intelligent Robots and Systems*, 2011.
- [23] M. R. Cutkosky and R. D. Howe, "Dextrous robot hands," S. T. Venkataraman and T. Iberall, Eds. Springer-Verlag New York, Inc., 1990, ch. Human grasp choice and robotic grasp analysis.
- [24] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *TPAMI*, 2008.
- [25] G. Barequet and S. Har-Peled, "Efficiently approximating the minimum-volume bounding box of a point set in three dimensions," *J. Algorithms*, pp. 91–109, 2001.
- [26] T. Rabbani, F. van den Heuvel, and G. Vosselmann, "Segmentation of point clouds using smoothness constraint," *ISPRS Commission V Symposium 'Image Engineering and Vision Metrology'*, 2006.
- [27] B. R. Radu, "Semantic 3d object maps for everyday manipulation in human living environments," Ph.D. dissertation, Computer Science department, Technische Universität München, Germany, October 2009.
- [28] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models web database: An object model database for object recognition, localization and manipulation in service robotics," *The International Journal of Robotics Research*, 2012.
- [29] A. Schmitz, U. Pattacini, F. Nori, L. Natale, G. Metta, and G. Sandini, "Design, realization and sensorization of the dexterous icub hand," in *IEEE-RAS International Conference on Humanoid Robots*, 2010.
- [30] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. World Scientific Pub. Co., 2002.
- [31] A. Wächter and L. Bielger, "On the implementation of a primaldual interior point filter line search algorithm for large-scale nonlinear programming," in *Mathematical Programming*, 2006, pp. 25–57.
- [32] U. Pattacini, F. Nori, L. Natale, G. Metta, and S. G., "An experimental evaluation of a novel minimum-jerk cartesian controller for humanoid robots," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1668–1674, 2010.
- [33] T. Yoshikawa, "Manipulability of robotic mechanisms," *The International Journal of Robotics Research*, vol. 4(2), pp. 3–9, 1985.
- [34] M.-J. Tsai, "Workspace geometric characterization and manipulability of industrial robots," Ph.D. dissertation, Ohio State University, 1986.
- [35] L. S., P. U., and B. J. D. et al., "Towards a platform-independent cooperative human-robot interaction system: II. perception, execution and imitation of goal directed act," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

Deep Hierarchies in the Primate Visual Cortex: What Can We Learn For Computer Vision?

Norbert Krüger, Peter Janssen, Sinan Kalkan, Markus Lappe, Aleš Leonardis, Justus Piater, Antonio J. Rodríguez-Sánchez, Laurenz Wiskott

Abstract—Computational modeling of the primate visual system yields insights of potential relevance to some of the challenges that computer vision is facing, such as object recognition and categorization, motion detection and activity recognition or vision-based navigation and manipulation. This article reviews some functional principles and structures that are generally thought to underlie the primate visual cortex, and attempts to extract biological principles that could further advance computer vision research. Organized for a computer vision audience, we present *functional principles* of the *processing hierarchies* present in the primate visual system considering recent discoveries in neurophysiology. The hierarchical processing in the primate visual system is characterized by a sequence of different levels of processing (in the order of ten) that constitute a *deep hierarchy* in contrast to the *flat* vision architectures predominantly used in today's mainstream computer vision. We hope that the functional description of the deep hierarchies realized in the primate visual system provides valuable insights for the design of computer vision algorithms, fostering increasingly productive interaction between biological and computer vision research.

Index Terms—Computer Vision, Deep Hierarchies, Biological Modeling

1 INTRODUCTION

The history of computer vision now spans more than half a century. However, general, robust, complete satisfactory solutions to the major problems such as large-scale object, scene and activity recognition and categorization, as well as vision-based manipulation are still beyond reach of current machine vision systems. Biological visual systems, in particular those of primates, seem to accomplish these tasks almost effortlessly and have been, therefore, often used as an inspiration for computer vision researchers.

Interactions between the disciplines of “biological vision” and “computer vision” have varied in intensity throughout the course of computer vision history and have in some way reflected the changing research focuses of the machine vision community [32]. Without any doubt, the groundbreaking work of Hubel and Wiesel [72] gave a significant impulse to the computer vision community via Marr’s work on building visual hierarchies analogous to the primate visual system [109]. However, the insufficient computational resources that

- N. Krüger is first author since he initiated and organized the writing of this paper, all other authors are ordered alphabetically.
- N. Krüger is with the Maersk Mc-Kinney Moller Institute at the University of Southern Denmark.
- P. Janssen is with the Division of Neurophysiology at the KU Leuven.
- S. Kalkan is with the Dept. of Computer Engineering, Middle East Technical University, Turkey.
- M. Lappe is with the Institute for Psychology of the University of Muenster, Germany.
- A. Leonardis is with the Faculty of Computer and Information Science, University of Ljubljana, Slovenia, and with the Centre for Computational Neuroscience and Cognitive Robotics, University of Birmingham, United Kingdom.
- J. Piater and A. Rodríguez-Sánchez are with the Intelligent and Interactive Systems group at the University of Innsbruck, Austria.
- L. Wiskott is with the Institut für Neuroinformatik at the Ruhr-Universität Bochum, Germany.

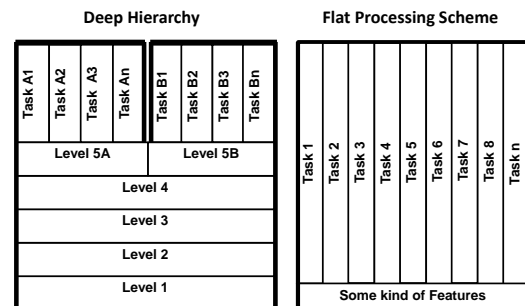


Fig. 1. Deep hierarchies and flat processing schemes

were available at that time and the lack of more detailed understanding of the processing stages in the primate visual system presented two insurmountable obstacles to further progress in that direction.

What followed was a reorientation of mainstream computer vision from trying to solve general vision problems to focusing more on specific methods related to specific tasks. This has been most commonly achieved in *flat processing schemes* (see figure 1, right) in which rather simple feature-based descriptors were taken as an input and then processed by the task-dependent learning algorithms. The ties with the biological vision faded, and if there were some references to biological-related mechanisms they were most commonly limited to individual functional modules or feature choices such as Gabor wavelets.

While the progress on some specialized machine vision problems and problem domains has been enormous (on some tasks, these systems can easily surpass human capabilities), artificial systems still lack the generality and robustness inherent in the primate visual system. As we are gaining

more and more insight into the functional mechanisms of the visual cortex (largely due to the advanced imaging techniques used in neuroscience), the time may be ripe to make a new attempt at looking at the mechanisms that could bring the capabilities of artificial vision, primarily in terms of generality and robustness, closer to those of biological systems. This may be a feasible enterprise also from the computational point of view, particularly in the light of new developments of computer architectures such as GPUs and multi-core systems.

In this paper, we will primarily focus on hierarchical representations and functional mechanisms of primates. We will look at different hierarchical levels of processing as well as different information channels (e.g., shape, color, motion) and discuss information abstractions that occur throughout the hierarchy.

It is known that around 55% of the neocortex of the primate brain is concerned with vision [44] and that there is a hierarchical organization of the processing pipeline that spans 8 to 10 levels (see figure 2). There is clear evidence that neurons in the early visual areas extract simple image features (e.g., orientation, motion, disparity, etc.) over small local regions of visual space and that this information is then transmitted to neurons in higher visual areas which respond to ever more complex features with receptive fields¹ covering larger and larger regions of the visual field. Such hierarchical structures, to which we refer as *deep hierarchies* (see figure 1, left), exhibit a number of computational advantages compared to the so-called *flat processing schema* (see figure 1, right).

Two important aspects are computational efficiency and generalization: As the hierarchical levels build on top of each other, they exploit the *shareability* of the elements to efficiently arrive at more complex information units. Such a design principle also contributes to common computations (both during learning and inference) which results in highly efficient processing as well as in lower storage demands. Moreover, reusing commonalities that exist among different visual entities and which are important perceptual building blocks for achieving different tasks leads to generalization capabilities and transfer of knowledge. For example, there is strong neurophysiological evidence that a generic description in terms of a variety of visual properties is computed in areas V1–V4 and MT, covering around 60% of the volume of visual processing in the primate neocortex (see [44] and figure 2 where visual areas are drawn proportionally to their actual sizes). These areas carry necessary information for a completion of a number of different tasks, such as object recognition and categorization, grasping, manipulation, path planning, etc.

It is also evident that in the visual system of primates there are separate (though highly inter-connected) channels that process different types of visual information (color, shape, motion, texture, 3D information), which contribute to the efficiency of representation (avoiding the combinatorial explosion of an integrated representation) and robustness (with respect to the available information). These advantages cover multiple

aspects and will be discussed in more detail in section 8.

However, although all neurophysiological evidence suggests that in the primate visual system quite a number of levels are realized, most existing computer vision systems are ‘flat’ and hence cannot make use of the advantages connected to deep hierarchies. Here in particular the generalization and scalability capabilities are crucial for any form of cognitive intelligence. In fact, there is overwhelming neurophysiological evidence that cognition and the concept of deep hierarchies are linked [178]. As a consequence, we see the issue of establishing deep hierarchies as one major challenge on our way towards artificial cognitive systems.

Bengio [9] discussed the potential of deep hierarchies as well as fundamental problems related to learning of deep hierarchies. In particular, he emphasizes the problem of the huge parameter space that has to be explored due the large number of hierarchical levels. This learning problem can be alleviated by (a) tackling intermediate representations as independent learning problems as well as (b) introducing bias in terms of basic connectivity structures expressed in the number of levels or the locality of connectivity of individual units of such deep structures. We believe that this paper can help to guide the learning process of deep hierarchies for vision systems by giving indications for suitable intermediate representations in the primate’s visual system. In addition, we believe that useful guidelines for connectivity patterns can be derived from the biological model in terms of appropriate receptive field sizes of neurons, number of levels being processed in the biological model as well as the number of units in a certain hierarchical level as indicated by area sizes in the primate’s visual cortex.

Despite the challenges connected to the learning of deep hierarchies, there exists a body of work in computer vision that made important contributions towards understanding and building hierarchical models. Due to lack of space, a more thorough review is outside the scope of this paper, and the following list is far from complete. From the computational complexity point of view, Tsotsos has shown that unbounded visual search is NP complete and that hierarchical architectures may be the most promising solution to tackle the problem [189]. Several works have shown that efficient matching can only be performed in several hierarchical stages, including Ettinger [42], Geman et al. [59], [60], Mel and Fiser [114], Amit [1] [2], Hawkins [69], Fidler et al. [45], Scalzo and Piater [153], Ullman and Epshtein [192], DiCarlo and Cox [31], Ommer and Buhmann [126], Serre and Poggio [157], Pugeault et al. [138], and Rodríguez-Sánchez [144]. Among the more known hierarchical models are the Neocognitron [54], HMAX [141], [158], LHOP [46], 2DSIL [145] and Convolutional Nets [101]. Recently, Bengio [9] published an exhaustive article on learning deep architectures for artificial intelligence.

In summary, in this article, we want to argue that deep hierarchies are an appropriate concept to achieve a general, robust, and versatile computer vision system. Even more importantly, we want to present relevant insights about the hierarchical organization of the primate visual system for computer vision scientists in an accessible way. We are aware that some of our abstractions are rather crude from the neurophysiological point of view and that we have left out important details of the

1. The *receptive field* of a neuron is the region where certain stimuli produce an effect on the neuron’s firing.

processes occurring at the different levels², but we hope that such abstractions and the holistic picture given in this paper will help to foster productive exchange between the two fields.

The paper is organized as follows: In section 2, we will touch upon the aspects of the primate visual system that are relevant to understand and model the processing hierarchy. The hierarchy in the primate vision system is then outlined from two perspectives. In the *horizontal perspective* (sections 3–6) we give a description of processing in the different areas indicated in figure 2. In section 7, we give a *vertical perspective* on the processing of different visual modalities across the different areas. In section 8, we then draw conclusions for the modeling and learning of artificial visual systems with deep hierarchical structures.

2 RELEVANT ASPECTS OF THE STRUCTURE OF THE VISUAL CORTEX

In section 2.1, we provide a basic overview of the deep hierarchy in the primate visual system. In section 2.2, we also give an intuition of basic (mostly biological) terms used in the following sections. Most data we present in the following were obtained from macaque monkeys since most neurophysiological knowledge stems from investigations on these.

While the primate brain consists of approximately 100 cortical areas, the human brain probably contains as many as 150 areas.³ There is a general consensus that the primary sensory and motor areas in the monkey are homologous to the corresponding areas in the human brain. Furthermore, several other cortical areas in the monkey have an identified homologue in the human (e.g. MT/MST, AIP). These areas can be viewed as landmarks which can be used to relate other cortical areas in the human to the known areas in the monkey.

It should be mentioned that a visual cortical area consists of six layers, which do not correspond to the layers in artificial deep models. In general, layer 4 is the input layer where the inputs from earlier stages arrive. The layers above layer 4 (layers 2 and 3) typically send feedforward connections to downstream visual areas (e.g. from V1 to V2), whereas layers 5 and 6 send feedback projections to upstream areas or structures (e.g. from V1 to the LGN and the Superior Colliculus – see also section 3.2). At higher stages in the visual hierarchy, the connectivity is almost always bidirectional. At present, detailed knowledge about the precise role of cortical microcircuits in these different layers is lacking.

2. For example, a heterogeneity of computations has been reported, including summation, rectification, normalization [19], averaging, multiplication, max-selection, winner-take all [150] and many others [89]. This is of great interest for addressing how neurons are inter-connected and the subject of much discussion but out of the scope of the present paper.

3. A region in the cerebral cortex can be considered to be an area based on four criteria: (1) cyto- and myeloarchitecture (the microscopic structure, cell types, appearance of the different layers, etc.), (2) the anatomical connectivity with other cortical and subcortical areas, (3) retinotopic organization, and (4) functional properties of the neurons. In far extrastriate cortex, where retinotopic organization is weak or absent, the specific functional properties of the neurons are an important characteristic to distinguish a region from the neighboring regions.

2.1 Hierarchical Architecture

Here we give a coarse and intuitive summary of the processing hierarchy realized in the primate visual system. A more detailed description can be found in sections 3 – 6. Basic data on the sizes of the different areas, receptive field sizes, latency, organization etc. are provided in table 1.

The neuronal processing of visual information starts in the retina of the left and right eye. Nearly all connections then project to a visual area called LGN before it reaches the visual cortex. We call these stages *precortical processing* and the processing in these areas is described in section 3. The visual cortex is commonly divided into three parts (figure 2 and table 1): the occipital part gives input to the dorsal and ventral streams. The occipital part covers the areas V1-V4 and MT. All areas are organized retinotopically, i.e., nearby neurons in the visual cortex have nearby receptive fields (see table 1, 6th column) and the receptive field size increases from V1 to V4 (see table 1, 3rd column). There are strong indications that these areas compute generic scene representations in terms of processing different aspects of visual information [84]. However, the complexity of features coded at the different levels increases with the level of the hierarchy as will be outlined in detail in section 4. Also it is worth noting that the size of the occipital part exceeds the other two parts occupying more than 62% of the visual cortex compared to 22% for the ventral and 11% for the dorsal pathway [44] (see table 1, 2nd column).⁴ In the following, we call the functional processes established in the occipital part *early vision* indicating that a generic scene analysis is performed in a complex feature structure.

The ventral pathway covers the areas TEO and TE which are involved in object recognition and categorization. The receptive field sizes are in general significantly larger than in the occipital part. There is a weak retinotopic organization in area TEO which is not observed in area TE. Neurons' receptive fields usually include the fovea (the central part of the retina with the highest spatial resolution). In the ventral path, the complexity of features increases up to an object level for specific object classes (such as faces) [127], however most neurons are responsive to features below the object level indicating a coding scheme that uses multiple of these descriptors to code objects and scenes [173].

The dorsal pathway consists of the motion area MST and the visual areas in posterior parietal cortex. The dorsal stream is engaged in the analysis of space and in action planning. Similar to the ventral stream, the receptive field sizes increase along the dorsal pathway and the complexity of stimulus features increases progressively (e.g. from simple motion in MT to more complex motion patterns in MST and VIP). Moreover, the relation of receptive fields to retinal locations weakens. Instead, higher areas encode the location of stimuli in spatial or head fixed coordinates.

Besides the division into two pathways (ventral and dorsal) it is worth noting that there are also two streams to be

4. These proportions are unknown for the human visual cortex because in both the temporal and the parietal lobe new areas have probably evolved in humans compared to monkeys.

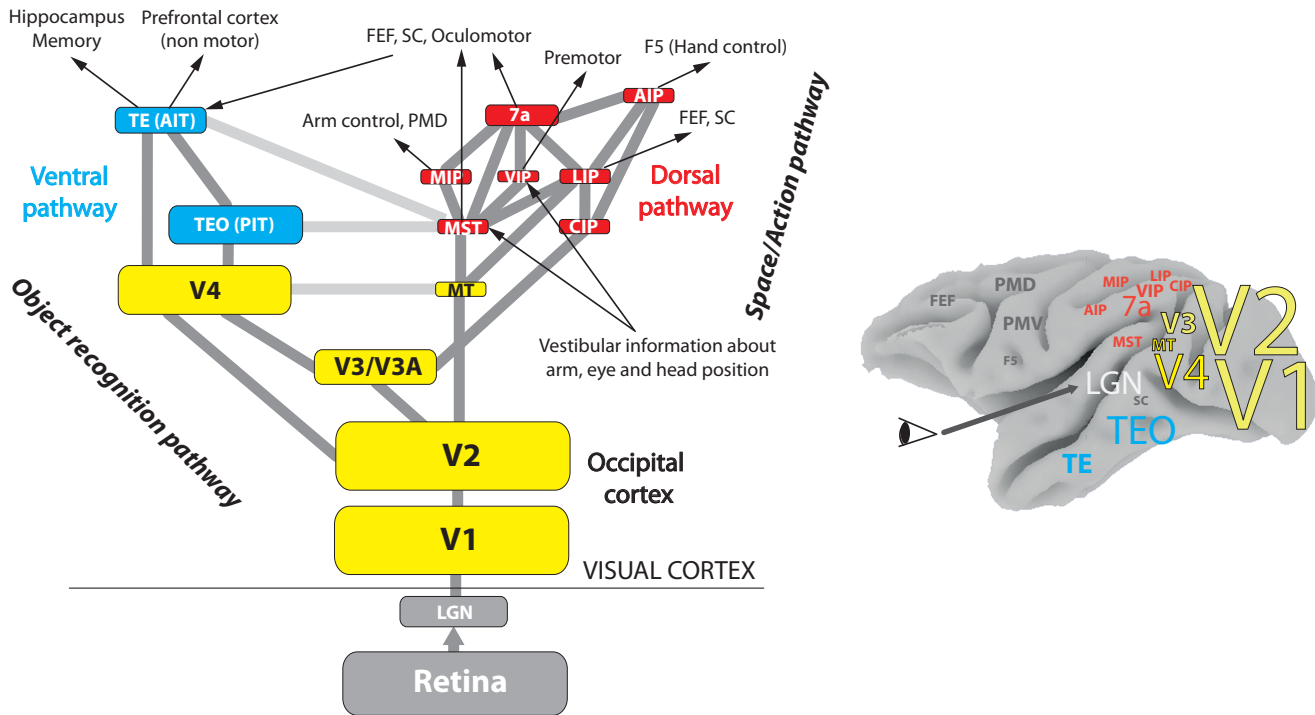


Fig. 2. Simplified hierarchical structure of the primate's visual cortex and approximate area locations (summarized from [44]). Box and font sizes are relative to the area size.

distinguished, the magnocellular (M-) and parvocellular (P-) stream [73]. This distinction is already present at the ganglion cell level, i.e., at the level of the output of the retina. P ganglion cells are color sensitive, have a small receptive field and are responsible for the high visual acuity in the central visual field. M ganglion cells have lower spatial but higher temporal resolution than P ganglion cells. The distinction between P and M cells carries through LGN to the whole visual cortex. To a first approximation, the P path is believed to be responsible for shape and object perception while the M path can account for the perception of motion and sudden changes [84]. Also the strongly space-variant resolution from the fovea to the visual periphery carries through most regions of the visual cortex.

It is worth noting that at every stage in the visual hierarchy, neurons also exhibit selectivities that are present at earlier stages of the hierarchy (e.g. orientation selectivity can be observed up to the level of TEO).

It is in general acknowledged that the influence of extrinsic information on the visual representations in the brain increases with its level in the hierarchy. For example, there is no report on any learning or adaptation processes in the retina and also quite some evidence on a high influence of genetic pre-structuring for orientation maps in V1 (see, e.g., [63]). On the other hand, it has also been shown that learning can alter the visual feature selectivity of neurons. However, the measurable changes at the single-cell level induced by learning appear to be much smaller at earlier levels in the visual hierarchy such as V1 [155] compared to later stages such as V4 [140] or IT [104].

2.2 Basic Facts on Different Visual Areas

Table 1 gives basic data on the different areas of the visual system. The first column indicates the name of the area, the second column the size in mm^2 (see also figure 2 where areas are drawn proportionally to their area size). The third column indicates the average receptive field size at 5 degrees of eccentricity. The fourth column indicates the latency to the first response to a stimuli at the retina.

Figure 3 provides a summary of most of the terms that follow in columns 5 through 7. The fifth column distinguishes between contra- and bilateral receptive fields. Contralateral (co in table 1) receptive fields only cover information from one hemifield while bilateral (bl in table 1) receptive fields cover both hemifields (figure 3b). The sixth column indicates different schemas of organization: Retinotopic organization (rt) indicates that the spatial arrangement of the inputs from the retina is maintained which changes every time we move our eyes, spatiotopic (st) indicates the representation of the world in real-world coordinates (see figure 3a), clustered organization (cl) indicates that there are larger subareas with similar functions, columnar organization (co) indicates that there is a systematic organization in columns according to some organizational scheme (mostly connected to visual features or retinotopy). The seventh column indicates different kinds of invariances (see figure 3c-f): cue invariance (CI) refers to the ability to obtain the same type of information from different cues, a cell that responds to an object independently of its size is called size invariant (SI), similarly for position

Area	Size (mm ²)	RFS	Latency (ms)	co/bi lat.	rt/st/cl/co	CI/SI/PI/OI	Function
Sub-cortical processing							
Retina	1018	0.01	20-40	bl	+/-/-	-/-/-	sensory input, contrast computation relay, gating
LGN		0.1	30-40	co	+/-/-	-/-/-	
Occipital / Early Vision							
V1	1120	3	30-40	co	+/-/+	-/-/-	generic feature processing
V2	1190	4	40	co	+/-/+	-/-/-	generic feature processing
V3/V3A/VP	325	6	50	co	+/-/+	-/-/-	generic feature processing
V4/VOT/V4t	650	8	70	co	+/-/+	+/-/-	generic feature processing / color
MT	55	7	50	co	+/-/+	+/-/+	motion
Sum	3340						
Ventral Pathway / What (Object Recognition and Categorization)							
TEO	590	3-5	70	co	(+)-/-/+	?/-/?	object recognition and categorization
TE	180	10-20	80-90	bl	-/+/+	+/+/+(-)	
Sum	770						
Dorsal Pathway / Where and How (Coding of Action Relevant Information)							
MST	60	>30	60-70	bl	+/-/+	I	optic flow, self-motion, pursuit 3D orientation of surfaces
CIP	?	?	?		+/-/?	+/?/?	
VIP	40	10-30	50-60	bl	-/+/-	I	optic flow, touch, near extra personal space Optic flow, heading
7a	115	>30	90	bl	(+)-/-/-	?/?+/?	
LIP	55	12-20	50	cl	+/-/-	?/-/-	salience, saccadic eye movements grasping
AIP	35	5-7	60	bl	?/+/?	?/+/?	
MIP	55	10-20	100	co	+/-/?	I	reaching
Sum	585						

TABLE 1

Basic facts on the different areas of the macaque visual cortex based on different sources [44], [28], [95], [142], [162] *First column:* Name of Area. *Second column:* Size of area in mm². '?' indicates that this information is not available. *Third column:* Average receptive field size in degrees at 5 degree of eccentricity. *Fourth column:* Latency in milliseconds. *Fifth Column:* Contra versus bilateral receptive fields. *Sixth Column:* Principles of organization: Retinotopic (rt), spatiotopic (st), clustered (cl), columnar (co). *Seventh Column:* Invariances in representation of shape: Cue Invariance (CI), Size Invariance (SI), Position Invariance (PI), Occlusion Invariance (OI). 'I' indicates that this entry is irrelevant for the information coded in these areas. *Eighth Column:* Function associated to a particular area.

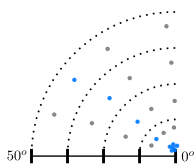
invariance (PI). Finally, a cell that responds similarly to an object irrespective of whether it is completely or partially present is invariant to occlusions (OI).

3 SUB-CORTICAL VISION

In this section, we describe the primate sub-cortical vision system. We begin with the retinal photoreceptors as the first stage of visual processing (section 3.1), and follow the visual signal from the eye through the Lateral Geniculate Nucleus (LGN) (section 3.2). For all areas, we first give a neurophysiological and then a functional perspective.

3.1 Base Level: Retinal Photoreceptors

The retina is located in the inner surface of the eye and contains photoreceptors that are sensitive only to a certain interval of the electromagnetic spectrum, as well as cells that convert visual information to neural signals. The pictogram on the left illustrates the space-variant retinal density of rods (gray) and cones (blue) as described below, as well as the uniformly-small receptive field sizes (around 0.01° of visual angle). Compare this to the corresponding pictograms we consistently give in the following sections.



Neurophysiological view: There are two kinds of photoreceptors, rods and cones. Rods have a high sensitivity to low levels of brightness (see icons at the left). Cones, on the other hand, require high levels of brightness. We can classify the cones as a function of their wavelength absorbency as S (short wavelength = blue), M (middle wavelength = green) and L (long wavelength = red) cones. These three cone types allow for the perception of color [13]. The resolution (i.e., the number of receptors per mm²) decreases drastically with the distance from the fovea. This holds for both rods and cones, except that there are no rods in the fovea. Most cones are concentrated in and around the fovea, while rods constitute the bulk of the photoreceptors at high eccentricities.

Functional view: Because only a small part of the retina has a high spatial resolution (the fovea), gaze control is required to direct the eyes such that scene features of interest project onto the fovea. Therefore, primates possess an extensive system for active control of eye movements (involving the FEF in the frontal lobe, LIP in the parietal lobe and the Superior Colliculus in the midbrain). It is influenced both by reflexive, signal-driven and by intentional, cognitively-driven attentional mechanisms, and involves the entire visual hierarchy. Attention models compute where to fixate [135], [143] and some work even addresses learning to control gaze, e.g., to minimize tracking uncertainty [6]. However, in computer vision cognitively-driven attentional mechanisms remain largely unexplored.

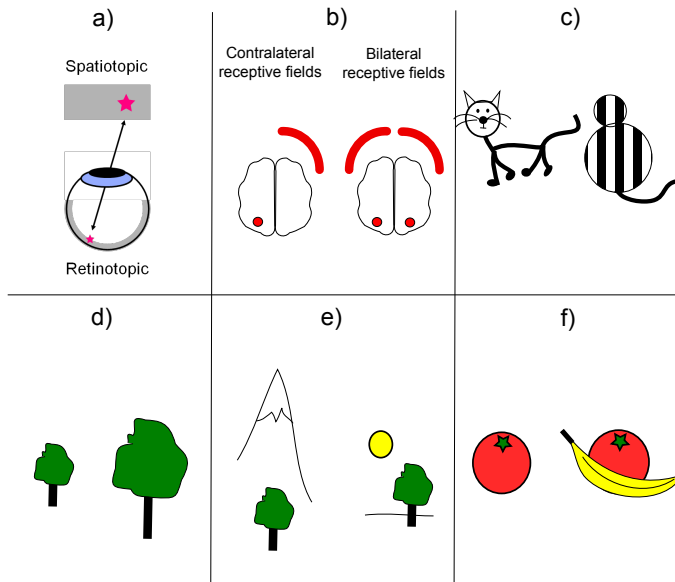



Fig. 3. Summary of table 1 concepts: a) Retinotopic (rt) and spatiotopic (st) organization; b) Contra- (co) versus bilateral (b) receptive fields; c) Cue Invariance (CI); d) Size Invariance (SI); e) Position Invariance (PI); f) Occlusion Invariance (OI)

3.2 Ganglion Cells and LGN

From the photoreceptors of the retina information is passed through ganglion cells and LGN to the primary visual cortex. The left LGN receives input of the right visual hemifield from both eyes, and the right LGN receives input of the left visual hemifield from both eyes. However, the information from the two eyes remains still entirely separate in six different neuronal layers (four P- plus two M-layers, three layers receive input from the left eye, the other three layers from the right eye) of the LGN; no binocular integration is done at this level. Regarding spatial analysis, there are no significant differences between retinal ganglion cells and their LGN counterparts (there is even almost a one-to-one correspondence between retinal ganglion and LGN cells [95]). In motion analysis, LGN ganglion cells have lower optimal temporal frequencies, 4–10 Hz vs. 20–40 Hz in retinal ganglion cells, which indicates the presence of some low-pass filtering over retinal ganglion cells [95]. The two prominent new features emerging at this level are center-surround receptive fields and color opponency. The visual cortex is also organized into layers, where most of the feedforward connections (i.e. connections to a higher stage in the hierarchy) originate from the superficial layers and most of the feedback connections originate from the deeper layers. However, virtually nothing is known about the role of these different cortical layers in stimulus processing.

3.2.1 Center-Surround Receptive Fields


 **Neurophysiological view:** Luminance sensitive cells with a center-surround receptive field come in two

types: on-center/off-surround cells are sensitive to a bright spot on a dark background; off-center/on-surround cells are sensitive to the inverse pattern. Both are insensitive to homogeneous luminance. These cells are magnocellular (M) neurons and are involved in the temporal analysis.

Functional view: Center-surround receptive fields can be modeled by a difference of Gaussians and resemble a Laplace filter as used for edge detection [68]. They thus emphasize *spatial change* in luminance. These cells are also sensitive to *temporal changes* and form the basis of motion processing. Notably, the transformation into a representation emphasizing spatial and temporal change is performed at a very early stage, immediately following the receptor level, before any other visual processing takes place.

Most of the current computer vision techniques also involve in the earliest stages gradient-like computations which are essential parts of detectors / descriptors such as SIFT, HOG/HOF, etc.

3.2.2 Single-Opponent Cells

 **Neurophysiological view:** Single-opponent cells are color sensitive and compute color differences, namely L-M (L for long wavelength and M for middle wavelength, symbol “-” stands for opponency) and S-(L+M) (S stands for short wavelength), thereby establishing the red-green and the blue-yellow color axes. They have a band-pass filtering characteristic for luminance (gray value) stimuli but a low-pass characteristics for monochromatic (pure color) stimuli. These cells are parvocellular (P) neurons and are somewhat slower but have smaller receptive fields, i.e. higher spatial resolutions, than the magnocellular neurons. They are particularly important for high acuity vision in the central visual field.

Functional view: Single-opponent cells can be modeled by a Gaussian in one color channel, e.g. L, and another Gaussian of opposite sign in the opposing color channel, i.e. -M. This results in low-pass filtering in each color channel. The color opponency provides some level of invariance to changes in brightness and is one step towards color constancy.

4 GENERIC SCENE REPRESENTATION IN THE OCCIPITAL CORTEX

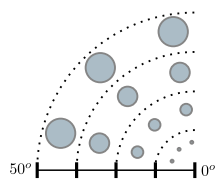
All areas in the occipital cortex (except MT) are organized retinotopically with orientation columns as basic units (see table 1, 6th column). MT is also organized retinotopically, but with depth and motion columns. Note that the visual system is not organized in a strictly sequential hierarchy but there are shortcuts between levels of the hierarchy. There is a stream $V1 \rightarrow V2 (\rightarrow V3^5) \rightarrow V4$ to the ventral pathway and another stream $V1 \rightarrow V2 \rightarrow MT$ to the dorsal pathway (figure 2). However, there also exist cross connections between V4 and MT.

The latency of the visual signal increases with each level by approximately 10 ms, and the receptive field sizes increase gradually (see table 1, 3rd and 4th column). In general, the

5. Not much is known about the role of V3, therefore we have not given any detailed information in this paper about V3.

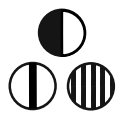
magnocellular pathway provides most of the input to the dorsal visual stream and the parvocellular pathway provides most of the information to the ventral pathway, but this is certainly not an absolute distinction.

4.1 Area V1



V1 is the first cortical area that processes visual information. Thus, the features it is sensitive to are more complex than in LGN but remain relatively simple: edges, gratings, line endings, motion, color, and disparity.

4.1.1 Edges, Bars, and Gratings



Neurophysiological view: V1 contains cells that respond preferentially to edges, bars, and gratings, i.e. linear oriented patterns. They are sensitive to the orientation of the patterns and, in case of gratings, to their spatial frequency (for a review, see [127]). Some cells are more sensitive to edges or single bars while others prefer gratings. There are two types of such cells, simple and complex cells. The former are sensitive to the phase of a grating (or exact position of a bar), the latter are not and have a larger receptive field.

Functional view: The original proposal by Hubel and Wiesel to achieve the phase-invariant orientation tuning characteristic of complex cells was simply to add the responses of simple cells along the axis perpendicular to their orientation, see [167] for a computational model. Later authors have attributed the behavior of complex cells to a MAX-like operation [48] (producing responses similar in amplitude to the larger of the responses pertaining to the individual stimuli – see, e.g., [141]) or to a nonlinear integration of a pool of unoriented LGN cells [115]. In computational models, simple cells can self-organize from natural images by optimizing a linear transformation for sparseness, i.e. only few units should respond strongly at any given time [125], or statistical independence [8] – however, it has been noted that linear models may not be sufficient for modeling simple cells [149]. Complex cells can be learned from image sequences by optimizing a quadratic transformation for slowness, i.e. the output of the units should vary as slowly over time as possible [38], [10]. On a more technical account it has been shown that Gabor wavelets are a reasonable approximation of simple cells while the magnitude of a Gabor quadrature pair resembles the response of complex cells [82]. Gabor wavelets have also been very successful in applications such as image compression [29], image retrieval [108], and face recognition [202]. In fact, it has been shown using statistics of images that Gabor wavelets (and the simple cells in V1) construct an efficient encoding of images [164].

4.1.2 Point Features



Neurophysiological view: V1 also contains cells that are sensitive to the end of a bar or edge or the border of a grating. Such cells are called end-stopped or hypercomplex [127].

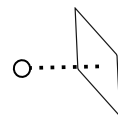
Functional view: In V1, end-stopped cells might help to solve the aperture problem the system is faced with in motion as

well as disparity processing (see section 4.1.3) since they can detect displacement also in the direction of an edge [127]. Like complex cells, hypercomplex cells can be learned from image sequences by optimizing slowness [10].

In computer vision, interest point detectors (which are not subject to the aperture problem due to the fact that they analyze local regions with occurrence of different orientations) of various kinds [107], [116] have been used since these features have turned out to be discriminative and stable, which is important for matching tasks and fundamental in many computer vision problems (pose estimation, object recognition, stereo, structure from motion, etc.). In this regard, it is interesting that V1 is dominated by detectors (simple and complex cells) for *linear* features (edges, bars, gratings). A possible reason might be that most meaningful features in natural scenes are actually edges which also allow for a complete reconstruction of the input signal (see, e.g., [39]).

The rather infrequent occurrence of neurons sensitive to point features at this low-level stage of visual processing suggests that primate vision does not necessarily rely on point features for bottom-up visual processing. Stereo and motion processing on the basis of edge and line features further suggests that the aperture problem is not solved by V1, but involves subsequent cortical layers for spatial integration.

4.1.3 Absolute Disparity



Neurophysiological view: V1 is the first area containing neurons that receive input from both eyes [84] (neurons in LGN are still monocular) and are able to compute disparity. In V1, this is still absolute disparity (i.e., the angular difference between the projections of a point onto the left and right retinas with reference to the fovea). Calculating disparity and thereby depth can be done in V1 without monocular contours in the image, as it is evident from our ease at interpreting random-dot stereograms [83]. There are also neurons in V1 that are sensitive to disparity in anticorrelated stereograms [26], in which the contrast polarity of the dots in one eye is reversed compared to the other eye. However, these neurons do not contribute to the depth perception and may have other functions.

Functional view: A prominent model for disparity estimation in V1 is the energy model, which is based on Gabor wavelets with slight phase or positional shifts [50]. Disparity is, of course, only one cue for depth perception, although an early one (in terms of processing and development, see [87]) and operational at close distance. On higher levels and at farther distances, cues such as occlusion, motion parallax etc. are used [84] which however are processed in higher-level areas of the primate brain's dorsal and ventral visual streams (see section 4.4). Also from a developmental perspective there are significant differences with pictorial depth cues developing only after approx. 6 months [87]. This is very much linked to the observation that statistics of natural scenes are linked to laws of perceptual organization, an idea first formulated by Brunswick [17] which has then later been confirmed computationally (see [200] for a review). This line of thought opens the perspective to formulate the problem of deriving

pictorial depth cues in computer vision systems as a statistical learning problem. Disparity is not only important for depth perception but also for gaze control [84], object grasping and object recognition. It has been shown that disparity tuned units can be learned from stereo images by maximizing mutual information between neighboring units, because depth is a feature that is rather stable across space [7].

In computer vision, stereo is a whole field of research, with many methods based on point features, which are convenient since their matches fix all degrees of freedom (see, e.g., [16]). However, there are approaches in computer vision that also use phase-differences of Gabor wavelets [49].

4.1.4 Local Motion

Neurophysiological view: Neurons in areas V1 and V2 are not only involved in static scene analysis but also in motion analysis. A fraction of simple and complex cells in V1 are direction selective, meaning that they respond only if the stimulus pattern (grating) moves in one direction and not the other [127]. However, only complex cells have spatio-temporal frequency tuning. The direction selective cells belong to the M-pathway and project mostly to area MT [118]. The aperture problem is not solved at that stage of processing.

Functional view: Estimating motion, or optic flow, is actually quite related to estimating disparity, since the latter can be viewed as a special case of the former with just two frames that are displaced in space rather than in time. The algorithms in computer vision as well as models of V1 are in general correspondingly similar to those discussed for estimating disparity (see section 4.1.3). For V1 (mainly simple cells), motion processing is usually conceptualized and modeled by spatiotemporal receptive fields [195], [179]. Complex cell-like units learned by optimizing slowness are motion direction selective, much like physiological neurons [10].

It is interesting to note that spatiotemporal features such as motion have been demonstrated to be the first features developmentally present in humans for recognizing objects (even sooner than color and orientation) [204].

4.1.5 Double-Opponent Cells

Neurophysiological view: About 5–10% of V1 cells are dedicated color-coding cells (for reviews see [23], [161]). In addition to single-opponent cells similar to those in LGN, which respond to local color (on a blue-yellow or red-green axis), V1 has double-opponent cells. These cells, whose existence used to be debated and is now supported with growing evidence (e.g., [22]), have a spatial-opponency structure within each color channel in addition to the opponency between different color channels. Such cells respond particularly well to a spot of one color on a background of its opponent color, and are thought to play a crucial role in perceptual color constancy. It is therefore not surprising that color contrast effects, i.e. a shift of perceived color of a stimulus away from the color of the background, have been observed in V1 [127]. The receptive fields of these cells are rarely circularly symmetric and therefore also show some

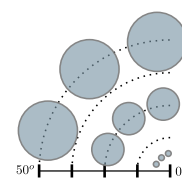
orientation tuning, but their spatial resolution is low. Some double-opponent cells are also orientation selective. On the other hand, simple and complex cells, although not considered as coding color, are often sensitive to the orientation of equiluminant stimuli, i.e. edges or gratings defined only by color contrast and not luminance contrast. This shows that they are sensitive to color, but they do not code the color polarity but only orientation. We therefore see that color and form processing are largely (but not completely) separated in V1.

Functional view: Double-opponent cells form the basis of color contrast and color constancy, because they allow the system to take the color context into account in determining the perceived color [23]. It is interesting that double-opponent receptive fields can be learned from natural color images by optimizing statistical independence [20], which suggests that they are organized by an information optimization process and are therefore functionally driven.

In contrast to low-level color normalization in computer vision, which is based primarily on operations applied the same way to each pixel (see, e.g., [47]), it is evident from human color perception that the achievement of color constancy involves local and global processes spanning all levels of the hierarchy, as already indicated by Helmholtz (see [199] and section 7.1).

4.2 Area V2

V2 is a retinotopically-organized area that mostly receives its input from V1. In V2, the segregation between M and P pathways is largely preserved although not complete [84]. Like V1, V2 contains cells tuned to orientation, color, and disparity. However, a fraction of V2 cells are sensitive to relative disparity (in contrast to absolute disparity arising in V1), which means that they represent depth relative to another plane rather than absolute depth. The main new feature of V2 is the more sophisticated contour representation including texture-defined contours, illusory contours, and contours with border ownership.



4.2.1 Texture-Defined and Illusory Contours

Neurophysiological view: Some V2 cells are sensitive to texture-defined contours, with an orientation tuning that is similar to that for luminance-defined contours [127]. V2 cells are also sensitive to illusory contours [84]. These can arise in various contexts, including texture or disparity discontinuities, or in relation to figure-ground effects such as the Kanizsa triangle (see icons at the left).⁶

Functional view: This is a step towards greater invariance of shape perception, since contours can be defined by a greater variety of cues.

⁶ V1 also responds to illusory contours but has longer latencies and might be driven by feedback from V2.

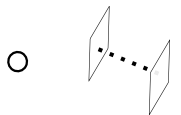
4.2.2 Border Ownership



Neurophysiological view: Borders (i.e., contours) are mostly formed by the projections of two or more surfaces that either intersect or have gap between them in 3D. In most cases, such borders belong only to one of the surfaces that meet at the border, and border ownership pertains to the assignment of which surface (or region) a border belongs to. Border ownership was already identified as an important visual information by [90], although with a different term, *belongingness*. Border ownership, which was largely neglected in computational approaches to vision, is especially crucial for diffusion and filling-in mechanisms with which missing and ambiguous visual information can be reduced and rectified to a great extent. Discovery of cells sensitive to border ownership was quite recent. In 2000, Zhou et al. [206] found that 18% of the cells in V1 and more than 50% of the cells in V2 and V4 (along the ventral pathway) respond or code according to the direction of the owner of the boundary. However, the mechanisms by which neurons determine the ownership is largely unclear.

Functional view: The fact that border ownership sensitive neurons differentiate the direction of the owner 10–25 ms after the onset of the response and that border ownership sensitivity emerges as early as V1 (although to a lesser extent) suggests that border ownership can be determined using local cues that can be integrated by lateral long-range interactions along a boundary. However, as shown recently by Fang et al. [43], the process might also be modulated or affected from higher-level cortical areas with attention.

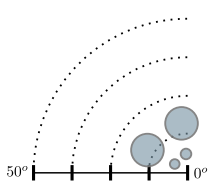
4.2.3 Relative Disparity



Neurophysiological view: V2 also includes disparity-sensitive cells. However, contrary to disparity-sensitive cells in V1, those in V2 are sensitive to relative disparity, which is the difference between the absolute disparities of two points in space. Relative disparity is for example the difference in disparity between a point at the fixation plane (zero disparity) and a point closer to the observer (near disparity). It is known that stereopsis relies mostly on the processing of relative disparity [130].

Functional view: With sensitivity to relative disparity in V2, it becomes possible to compare depth of objects and reason about their 3D spatial relationships.

4.3 Area V4



In contrast to MT (see section 4.4) which seems to be dominated by M-pathway input, V4 seems to combine input from the M as well as the P pathway since blocking either M or P pathway reduces activity of most cells in V4 [84].

V4 neurons respond selectively to orientation, color, disparity and simple shapes. They continue the process of integrating lower-level into higher-level responses and increasing invariances. For instance, V4 cells respond to contours defined by

differences in speed and/or direction of motion with an orientation selectivity that matches the selectivity to luminance-defined contours [127] (a few such cells are also found in V1 and V2 but with longer latencies, which again suggests that they are driven by feedback from V4). Prominent new features in V4 are curvature selectivity and luminance-invariant coding of hue.

4.3.1 Curvature Selectivity



Neurophysiological view: Some V4 cells are tuned to contours with a certain curvature (with a bias towards convex contours [131]) or vertices with a particular angle [127]. This selectivity is even specific to the position of the contour segment relative to the center of the shape considered, thus yielding an object centered representation of shape. V2 also has cells that respond to curves (contours that are not straight lines), but their response can be explained by their tuning to edges alone, which is not the case for V4 neurons.

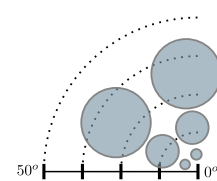
Functional view: Experiments in monkeys where area V4 was ablated showed that V4 is important for the perception of form and pattern/shape discrimination. V4 neuronal responses represent simple shapes by a population code that can be fit by a curvature-angular position function [131]. In this representation, the object's curvature is attached to a certain angular position relative to the object center of mass. Most V4 neurons represent individual parts or contour fragments.

4.3.2 Color Hue and Luminance Invariance



Neurophysiological view: Color coding cells in V4 differ from those in V2 in that they code for hue, rather than color opponency along the two principal color axes, and that the tuning to hue is invariant to luminance [24]. Even though specialized to color, many of these cells also show a prominent orientation tuning. **Functional view:** Luminance invariant tuning to hue is already a form of color constancy, and the orientation tuning of color coding cells indicates some level of integration between color and form perception, although V4 neurons are clearly segregated into two populations, one for color and one for form processing [177].

4.4 Area MT

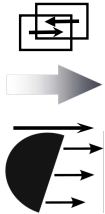


The middle temporal (MT) area is dedicated to visual motion and binocular depth processing. The vast majority of neurons in area MT are sensitive to moving stimuli. Neurons are tuned to direction and speed of motion [112]. Receptive fields are about 10 times larger than in V1 so that MT neurons integrate a set of motion signals from V1 over a larger area. The receptive fields show characteristic substructures of different motion sensitivity in different parts of the receptive field [106]. Many MT neurons are also sensitive to binocular disparity [30]. Activity in MT directly relates to perceptual motion [152] and depth [14] judgments. Area MT is retinotopically organized with motion and depth

columns similar to orientation and ocular dominance columns in V1.

MT is not only important for perception but also for motor control, particularly for smooth pursuit eye movements. MT together with MST provides the main velocity signal in the feedback control loop [37], [94] through output connections into oculomotor structures in the brain stem.

4.4.1 2D motion

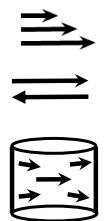


Neurophysiological view: MT neurons compute a mid-level representation of motion by combining inputs from V1 neurons that respond to local motion [165], [118]. Some MT cells solve the aperture problem and encode the direction of motion independent of the orientation of the moving stimulus [117]. MT cells encode the speed rather than spatiotemporal frequency as V1 cells

do [133]. In calculating motion signals, MT neurons follow a coarse-to-fine strategy in which responses to moving stimuli are fast, but imprecise, and become more refined over time [129].

Functional view: After initial measurements of local spatiotemporal energy (in V1), the combination of motion measurements is required to solve the aperture problem, derive 2D motion direction, and estimate speed. This results in a mid-level representation of motion in the visual field that is more faithful to the true motion and more robust against noise than earlier visual areas such as V1 and V2. The spatial smoothing that is inherent in the combination of motion over large receptive fields is partially reduced by disparity information in the combination of motion signals [99].

4.4.2 Motion Gradients and Motion-Defined Shapes



Neurophysiological view: Some MT cells are selective to higher order features of motion such as motion gradients, motion-defined edges, locally opposite motions, and motion-defined shapes [127]. These selectivities are aided by disparity sensitivity. Disparity helps to separate motion signals from objects at different distances, retain motion parallax and compute transparent motion

and three-dimensional motion surfaces.

Functional view: MT constructs a representation of motion-defined surfaces and motion on surfaces.

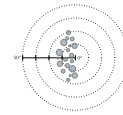
5 OBJECT RECOGNITION AND CATEGORIZATION: THE VENTRAL STREAM

Lesion studies have demonstrated that the ventral pathway is critical for object discrimination [193], whereas the posterior parietal cortex is important for spatial vision. The most widely used partitioning of the inferior temporal cortex (IT) is between the more posterior part, TEO, and the more anterior part, area TE, based on the presence of a coarse retinotopy in TEO but not in TE (see table 1) as well as a larger receptive field size of neurons in the latter area over the former.⁷ Two

7. Many more functional subdivisions have been proposed for IT, including separate regions encoding information about faces, color or 3D shape, but the correspondence with the anatomical subdivisions is unclear at present.

types of neurons have been identified in IT [174]: Primary cells respond to simple combinations of features and are a majority in TEO; Elaborate cells respond to faces, hands and complex feature configurations and have a high presence in area TE.

5.1 Area TEO

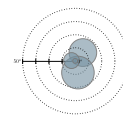


Neurophysiological view: TEO (also known as PIT for *Posterior IT*) neurons are orientation- and shape-selective. It has been shown that TEO neurons mostly respond to very simple shape elements. The main difference between TEO and TE is the coarse retinotopic organization in TEO, which is absent in TE. The receptive fields of TEO neurons are still relatively small (3-5 deg) and located around the fovea or in the contralateral hemifield.



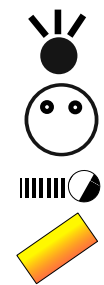
Functional view: TEO is responsible for medium complexity features and it integrates information about the shapes and relative positions of multiple contour elements. TEO integrates contour elements but with a higher degree of complexity over V4. This integration is non-linear and it includes inhibitory inputs (in addition to the excitatory ones). Shape tuning is position and size invariant, and it supports part-based shape theories [127].

5.2 Area TE



Neurophysiological view: Area TE (also known as AIT for *Anterior IT*) can be characterized by a marked increase in the complexity of the visual features that drive the neurons with respect to the previous areas in the ventral pathway (Sec. 4). It is suggested that shape-selective TE neurons integrate the output from the previous areas. The receptive fields of visual neurons in TE range from 10 to 20 degrees of visual angle, and the average response latencies are around 70–80 ms.

Although 2D shape is the primary stimulus dimension to which TE neurons respond, other object attributes are encoded in TE as well: color [183], disparity [183], texture [183], and 3D shape [81]. At least for color and 3D shape it has been demonstrated that the processing of these object properties is largely confined to specific subregions in TE [80], [184].



Tanaka and co-workers [174] made a critical contribution by developing the stimulus-reduction method (see figure 4). After having measured the responses of TE neurons to real-world objects, they systematically reduced the image of the most effective object in an effort to identify the critical feature to which the TE neurons were responding. For many TE neurons, the critical feature was moderately complex, i.e. less complex than the entire image but more complex than simple bars or spots (figure 4).

In some cases, the neurons driven by the critical features were clustered in what might be considered cortical columns [183]. These findings have led to the hypothesis that TE neurons do not explicitly code for entire objects but only for object parts. Therefore, the read-out of TE needs to combine

information from many TE neurons to build an explicit object representation.

Functional view: Many properties of TE neurons (e.g. invariances, see table 1, 7th column) correspond well with the properties of visual object recognition. Several studies have demonstrated that the trial-to-trial variations in the firing rate of TE neurons correlate with the perceptual report of rhesus monkeys in various tasks, including object recognition [119], color discrimination [111] and 3D shape discrimination [196].

A neural system capable of object recognition has to fulfill two seemingly conflicting requirements, i.e. selectivity and invariance. On the one hand, neurons have to distinguish between different objects in order to provide information about object identity (and object class in the case of categorization) to the rest of the system, by means of sensitivity to features in the retinal images that discriminate between objects. On the other hand, this system also has to treat highly dissimilar retinal images of the same object as equivalent, and must therefore be insensitive to transformations in the retinal image that occur in natural vision (e.g. changes in position, illumination, retinal size, etc.). This can be achieved by deriving invariant features that are highly robust towards certain variations by discarding certain aspects of the visual data (as, e.g., SIFT descriptors [107]). From a systematic point of view, it would be however advantageous to not discard information, but to represent the information such that the aspects that are invariant are separated from the variant parts such that both kinds of information can be used efficiently (see, e.g., [31] and section 8.2).

TE neurons generally show invariance of the shape preference to a large range of stimulus transformations (though in general not in the absolute response levels). The most widely studied invariances of TE neurons include invariance for position (PI, cf. table 1, 7th column) and size (SI), but other stimulus transformations can also evoke invariant shape preferences: the visual cue defining the shape (cue invariance CI; [183]), partial occlusion (occlusion invariance OI; [183]), position-in-depth [79], illumination direction [92] and clutter (overlapping shapes, [183]). Rotation in depth evokes the most drastic changes in the retinal image of an object, and also the weakest invariance in TE, since most TE neurons show strongly view-dependent responses even after extensive training. The only exception might be faces, for which both view-dependent and view-invariant responses have been documented [183].

TE neurons typically respond to several but not all exemplars of the same category, and many TE neurons also respond to exemplars of different categories [198]. Therefore object categories are not explicitly represented in TE. However, recent *readout* experiments have demonstrated that statistical classifiers (e.g. support vector machines) can be trained to classify objects based on the responses of a small number of TE neurons [183], [88]. Therefore, a population of TE neurons can reliably signal object categories by their combined



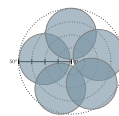
Fig. 4. TE neurons respond to critical features of objects that can be quite complex; more complex than edges or bars but less complex than objects [174].

activity.⁸ It is surprising that relatively little visual training has noticeable physiological effects on visual perception, on a single cell level as well as in fMRI [93]. For instance morphing objects into each other increases their perceived similarity, which is thought to be a useful mechanism for learning invariances [51].

6 VISION FOR ACTION: THE DORSAL STREAM

The dorsal visual stream (see Figure 2) contains a number of areas that receive visual information from areas such as MT and V3A, and project mostly to the premotor areas in the frontal lobe, bridging between the visual and motor systems. The areas located in the dorsal stream are functionally related to different effectors: LIP is involved in eye movements, MIP in arm movements, AIP in hand movements (grasping) and MST and VIP in body movements (self-motion).⁹

6.1 MST



Neurophysiological view: Area MST receives its major input from area MT (see figure 2). Like MT, MST has many neurons that respond to visual motion. Receptive fields in MST are much larger than those of MT, often covering substantial portions of the visual field without a clear retinotopic arrangement. Many MST neurons respond selectively to global motion patterns such as large-field expansions or rotations [176]. Thus, MST neurons integrate motion in different directions from within the visual field. The structure of the receptive fields, however, is very complex and often not intuitively related to the pattern selectivity [34]. MST neurons are tuned to the direction of self-motion, or heading, in an optic flow field [132], [100]. MST neurons carry disparity signals [148] and receive vestibular input [15], [67] both consistent with their involvement in self-motion estimation.

Area MST is also involved in smooth pursuit eye movement [37], where it employs non-visual (extraretinal) input [122]. Using this extraretinal information, some MST neurons cancel the retinal effects of eye movements and respond to motion in the world rather than to motion on the retina [41]. This is also seen in Area V3A [57].

Functional view: Area MST is concerned with self-motion, both for movement of the head (or body) in space and

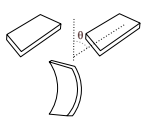
⁸ In contrast, an explicit category representation is present in the prefrontal cortex [53] and surprisingly also in the posterior parietal cortex (area LIP, [52]). Category information even occurs earlier, is stronger and more reliable in parietal cortex than in the prefrontal cortex [169].

⁹ Note that since not much is known about area 7a we have not discussed this area in detail.

movement of the eye in the head. The selectivity of MST neurons to optic flow patterns generates a population-based map of heading in MST [100]. Rather than representing the distribution of particular features in a retinotopic map of the visual field, as lower areas such as V1, V2, V4 or MT do, MST creates a new reference frame that represents self-motion in different directions in space. The organization is not retinotopic, but heading is represented in retinal coordinates, i.e., left or right with respect to the direction of gaze. The access to extraretinal eye movement information enables MST to estimate heading during combinations of body movement and eye movement.

The estimation of self-motion from optic flow is a common requirement in robotics. Solutions to this problem rely on the combination of many motion signals from different parts of the visual field as well as from non-visual areas relevant for heading estimation.

6.2 Caudal Intraparietal Area (CIP)

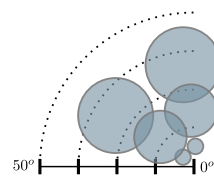


Neurophysiological view: CIP¹⁰ receives strong projections from area V3A and projects to LIP and AIP [121]. In [163], [170] it was reported that CIP neurons respond selectively to tilted planar surfaces defined by binocular disparity (first-order disparity). Some CIP neurons are also selective for the 3D orientation of elongated stimuli [151]. CIP neurons can show cue invariance for the tilt of planar surfaces, which means that the preference for a particular tilt is preserved when different depth cues signal the tilt (disparity, texture and perspective [190]). Results [188] suggest selectivity for zero-order disparity (position in depth) of CIP neurons. More recently, [86] also reported selectivity for curved surfaces (second-order disparity) in one monkey. CIP neurons do not respond during saccadic eye movements. No data exist on the size and shape of the CIP receptive fields nor on response latencies of CIP neurons.

Functional view: It is convenient to make a distinction between different orders of depth information from disparity [71]. Zero-order disparity refers to position-in-depth of planar surfaces (or absolute disparity, no disparity variation along the surface, see section 7.3) first-order disparity refers to inclined surfaces (tilt and slant, linear variations of disparity along the surface), and second-order disparity refers to curved surfaces (concave or convex, a change in the variation of disparity over the surface). CIP contains neurons that encode zero-, first- and possibly second-order disparities, which suggests that it is an important visual intermediate area that may provide input to visuomotor areas such as LIP and AIP. Not much is known about the internal organization of CIP.

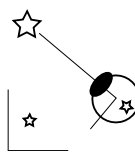
10. Note that since receptive field sizes of CIP neurons are unknown we have not drawn a corresponding figure as for the other regions.

6.3 Lateral Intraparietal Area (LIP)



Neurophysiological view: LIP is situated between visual areas and the motor system, receiving information from the dorsal and the ventral stream and projecting to other oculomotor control centers in the frontal lobe (FEF) and the superior colliculus [103]. LIP neurons respond before saccadic eye movements into the receptive field, and electrical microstimulation of LIP can evoke saccadic eye movements [181].

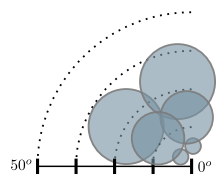
The visual responses in LIP are related to the salience of the stimulus [65], which led to the suggestion that LIP contains a salience map of the visual field, that guides attention and decides about saccades to relevant stimuli [11]. Moreover, LIP has been implicated in several other cognitive processes: decision formation [160], reward processing [136], timing [76] and categorization [52]. A more recent series of studies has also demonstrated that LIP neurons can respond selectively to simple two-dimensional shapes during passive fixation [156], a property that had been primarily allocated to the ventral visual stream.



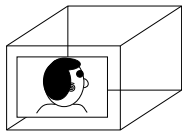
Functional view: The representation of space in LIP exemplifies several key properties of spatial processing in the dorsal stream. LIP neurons have visual receptive fields that represent locations on the retina, i.e. they represent stimuli in a retino-centric coordinate system. However, a few milliseconds before a saccadic eye movement, some LIP neurons become sensitive to stimuli at locations where their receptive field will be after the saccade [36]. This remapping of activity between the current and the future receptive field seems like a transient shift of the receptive field before a saccade. Moreover, although LIP receptive fields are basically in retino-centric coordinates, the activity of the cells is modulated by eye position, i.e., some cells respond more strongly to stimuli in their receptive field when the animal looks to the right than when it looks to the left, and vice versa [4]. The combination of retino-centric receptive fields and eye position modulation provides a population code in LIP that can represent the location of a stimulus in head-centric coordinates, i.e. can perform a coordinate transformation [207], [137]. This transformation allows, for example, for a combination of visual with auditory spatial input for the localization of sights and sounds [3].

LIP is one of the most studied areas in the dorsal stream. Despite more than two decades of single-cell studies, a considerable controversy exists with respect to the role of area LIP in high-level cognitive control processes such as motor planning, attention, decision formation, etc. However, LIP is believed to be a core area for spatial representation of behaviorally relevant stimuli. Visual (and auditory) input is transformed into a spatial representation in which each neuron uses eye-centered coordinates but in which the entire population forms a head-centric representation that encodes stimulus location even when the eye position changes. At the single neuron level, remapping of activity across saccades ensures continuity of the visual representation despite the eye movement.

6.4 Ventral Intraparietal Area (VIP)

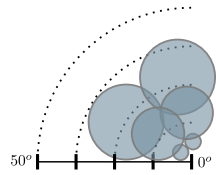


Neurophysiological view: Area VIP is connected with a wide range of visual, somatosensory, and premotor (mouth representation) areas. VIP neurons are multi-modal, in the sense that they can be activated by visual, tactile, vestibular and auditory stimulation, and smooth pursuit eye movements [21]. The tactile receptive fields are generally located on the skin of the head and face, and visual and tactile receptive fields frequently match in size and location: a neuron that responds to tactile stimulation of an area around the mouth will also respond to visual stimuli approaching the mouth. It has been proposed that VIP encodes near-extraperosnal space [21]. The receptive fields of VIP neurons vary from purely retinocentric to purely head-centered [35], including also receptive fields that are intermediate between retinocentric and head-centered. Furthermore, some VIP neurons respond to complex motion stimuli, such as the direction of heading in optic flow displays.



Functional view: Area VIP is likely to be involved in self-motion, control of head movements, and the encoding of near-extraperosnal (head-centered) space which link tactile and visual fields.

6.5 Medial Intraparietal Area (MIP)

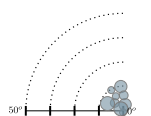


Neurophysiological view: MIP mainly projects to the dorsal premotor cortex (PMd). Neurons in this area typically respond selectively during a delayed reach task, in which monkeys are instructed to reach to a target on a touch screen after a certain time delay in order to receive a reward. MIP neurons will respond to particular reaching directions but not to others, and this neural selectivity is primarily eye-centered. When monkeys are free to choose the target, the MIP and PMd show increased spike-field coherence, suggesting direct communication between these brain areas [134].



Functional view: The activity of MIP neurons mainly reflects the movement plan towards the target, and not merely the location of the target or visual attention evoked by the target appearance [55]. MIP neurons also respond more when the animal chooses a reach compared to when the animal chooses a saccade towards a target, indicating that MIP encodes autonomously selected motor plans [25].

6.6 Anterior Intraparietal Area (AIP)



Neurophysiological view: The main inputs to AIP arise in LIP, CIP and the ventral pathway [12], whereas the output from AIP is directed towards the ventral premotor area F5, which is also involved in hand movements. Reversible inactivation of AIP causes a profound grasping deficit in the contralateral hand [56]. Sakata and co-workers showed that AIP neurons

frequently discharge during object grasping [151], with a preference for some objects over other objects. Some AIP neurons respond during object fixation and grasping, but not during grasping in the dark (visual-dominant neurons), other AIP neurons do not respond during object fixation but only when the object is grasped, even in the dark (motor-dominant neurons), whereas a third class of AIP neurons responds during object fixation and grasping, and during grasping in the dark (visuo-motor neurons, [120]). AIP encodes the disparity-defined 3D structure of curved surfaces [168]. However, experiments with monkeys indicate that the neural coding of 3D shape in AIP is not related to perceptual categorization of 3D shape [196]. In contrast, most 3D-shape-selective AIP neurons also respond during object grasping [180], suggesting that AIP represents 3D object properties for the purpose of grasping (i.e., grasping affordances).



Functional view: Neurons in AIP are sensitive to the 2D and 3D features of the object and shape of the hand (in a light or dark environment) relevant for grasping. In other words, area AIP might be involved in linking grasping affordances of objects with their 2D and 3D features. The extraction of grasping affordances from visual information is also currently a highly researched area in robotics since picking up unknown objects is a frequent task in autonomous and service robotics.

7 THE VERTICAL VIEW: PROCESSING OF DIFFERENT VISUAL MODALITIES

Based on the knowledge we gained in sections 3 – 6 on the brain areas involved in the processing of visual information, we can now summarize the processing of different visual modalities such as color (section 7.1), 2D and 3D shape (section 7.2 and 7.3), motion (section 7.4) as well as the processing for object recognition (section 7.5) and actions (section 7.6) in a 'vertical view', emphasizing the hierarchical aspects of processing of visual information. Figure 5 gives an overview of this vertical (per modality) as well as the horizontal (per area) view.

7.1 Color

Color can be an extremely informative cue and has always been used as one of the basic features in psychophysical visual search experiments. Efficient search can be performed with heterogeneous colors (up to nine distractors) as soon as they are widely separated in color space [203].

Neurophysiologically color processing is characterized by a steady progression towards color constancy (see figure 5, 3rd column). The three cones types (L, M, S) have a broad and largely overlapping wavelength tuning, and their firing rate is heavily affected by luminance. The single-opponent cells in LGN establish the two color axes red-green and blue-yellow, thereby sharpening the wavelength tuning and achieving some invariance to luminance. Double-opponent cells provide the means to take nearby colors into account for color contrast. In V4 hue is encoded, which spans the full color space. The final step is IT where there exists an association of color with form [205]. In TEO (closer to V4) most of the neurons are activated

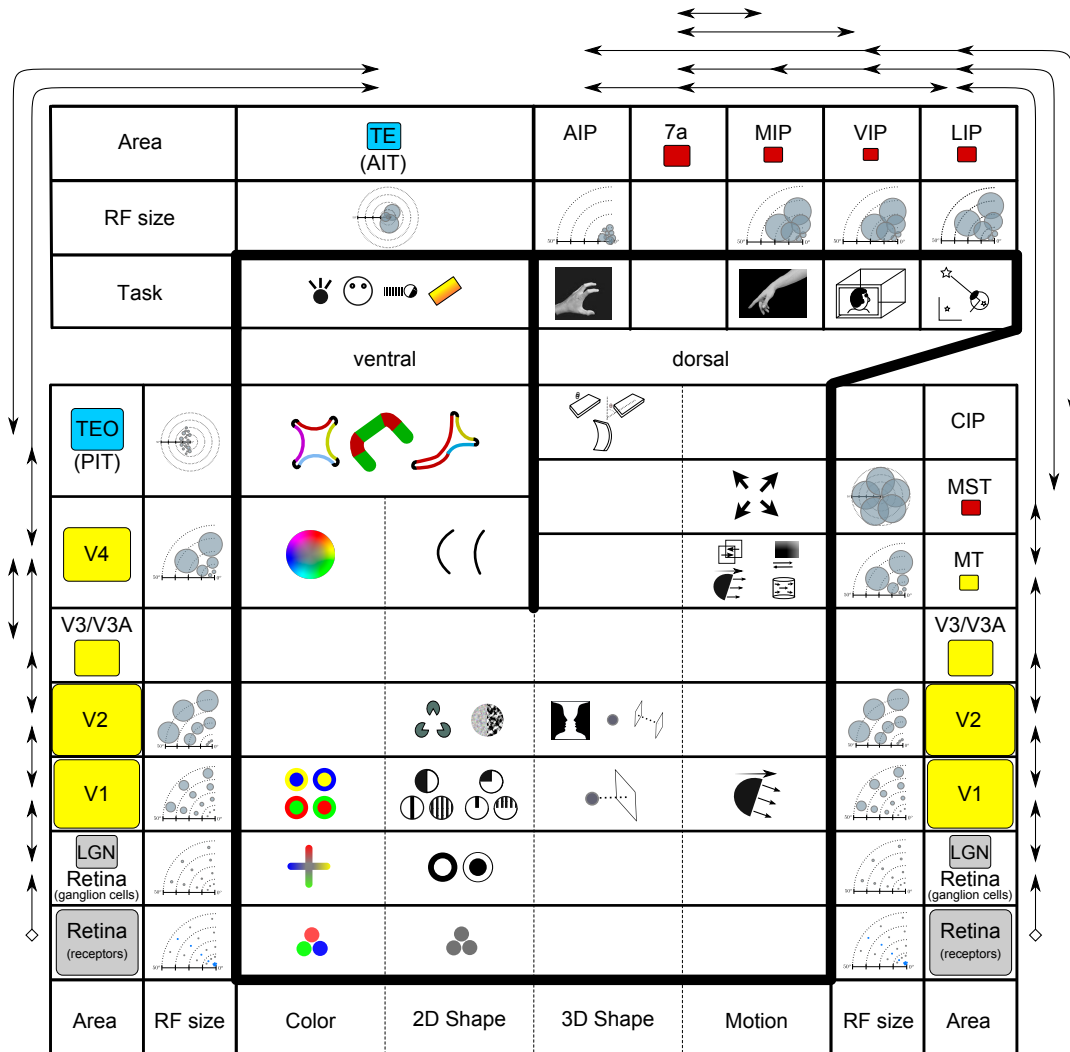


Fig. 5. Overview over the processing in the visual system. The icons of the text are arranged according to areas and modalities. The general layout follows figure 1, left. The yellow, blue and red rectangles have a size proportional to those of the corresponding areas. Connectivity is indicated in the vicinity, with the downwards arrowheads indicating the source area and the upwards arrowheads indicating the destination area. All but the retina → LGN connection are mirrored by feedback connections.

maximally by a simple combination of features such as bars or disks varying in size, orientation and color [175]. Elaborate cells (a majority in sub-area TE) respond to combinations of different features (shape and texture, shape and color, texture and color, texture and color and shape) [175].

There are a number of relevant insights that can be drawn from the neurophysiological evidence presented in the sections before. Color processing is taking place in a, to a large degree separated, pathway that only merges in general shape representations on the level of TE. Color is a cheap but also brittle feature for computer vision purposes. Its efficient use for object recognition depends on achieving color constancy which can still be seen as a challenge in computer vision applications. In the primate visual system, this is only achieved at rather late stages (V4 and beyond), hence involving a large part of the visual hierarchy. This is very different from

color normalization schemes on a pixel level predominant in computer vision. A hierarchical representation might be able to provide means to provide mid- and high level cues for achieving color constancy.

7.2 Two-Dimensional Shape

Processing of 2D shape is characterized throughout the visual system by increasing receptive field sizes, increasing complexity of relevant features, and increasing degree of invariance (see figure 5, 4th column).

The receptive field sizes are tiny in the retina and can be as large as half the visual field in IT (see Table 1, second column). But it is not only the size that increases, the receptive fields also get more complex and dynamic. In the early areas, receptive fields tend to show a linear response. Beginning in V1, cells have a non-classical receptive field, i.e. the response

of these cells is modulated by the surrounding, which implies contextual effects. In V4, strong attentional effects have been shown, resulting in different responses of a cell for identical stimuli, if the task and thereby the attentional state changes [84]. In IT, receptive fields are very large but the selectivity can be influenced by clutter and other objects in the scene [147]. For isolated objects on a blank background, receptive fields are very large; for an object on a cluttered background or among many distractors, receptive fields are relatively small which indicates a tight connection between detection and segmentation.

The features that drive cells in the visual system also gradually increase in complexity. They are simply spots in retina and LGN, primarily bars or edges in V1, particular curves in V4, then more complex patterns and object parts in TEO and TE. The general notion is that the more complex features are built up from the simpler ones, e.g. simple cells that are sensitive to bars can be combined from on- and off-center cells that are sensitive to spots.

Early on does the visual system try to make responses invariant to frequently occurring but irrelevant variations [201]. That starts already in the retina where several mechanisms are in place to achieve a high degree of luminance invariance, so that we can see in the darkness of the night and the brightness of a sunny day. Some position invariance is first achieved in V1 by the complex cells, which are sensitive to bars and edges of certain orientations, like simple cells, but which are less sensitive to the position of the stimuli. This position invariance increases throughout the visual system and in IT, objects can be moved around by 10 degrees or even more without degrading the selectivity of some of the IT cells [185], [75]. There is also increasing size invariance. In addition to invariances to illumination and to geometrical transformations, invariance is also achieved with respect to the cues used to define objects (see table 1, 7th column). Edges are the primary features used to represent objects, it seems. In V1 they are defined as boundaries between dark and light, or between different color hues; in V2 contours may also be defined by texture boundaries and these cells respond to illusory contours; in V4 contours may even be defined by differences in motion.

Representing and recognizing a 2D shape requires more than a collection of edges. The edges must be integrated somehow into one coherent percept. This is known in neuroscience as the binding problem [186], [84]. It is thought that there must be a mechanism that binds together the elementary features to one object, because otherwise one would mix the features of one object with those of another one and perceive something that is not there (this actually happens in humans in case of fast presentation times [187]). Possible solutions to the binding problem are tuning of cells to conjunctions of features, spatial attention, and temporal synchronization. The latter idea assumes that somehow the visual system manages to synchronize the firing of those neurons that represent the same object and desynchronize them from others [166], which could also explain the fairly limited number of objects we can process simultaneously. The binding problem is related to segmentation. Responses that represent also border ownership, like in V2, and responses that are specific to the relative

position of an edge with respect to the object center, like in V4, are probably relevant for both processes.

7.3 Three-Dimensional Shape

The brain computes the third dimension (depth) from a large number of depth cues. Binocular disparity is one of the most powerful depth cues. Importantly, only second-order disparities (see section 6.2) are independent of eye position (vergence angle) and distance [71], thereby constituting a very robust parameter to estimate the three-dimensional layout of the environment.

The neural representation of 3D shape emerges gradually in the visual system (see figure 5, '3D shape' column). A few general principles can be identified. First, at progressively higher stages in the visual system, the neurons become tuned to more complex depth features, starting with absolute disparity in V1 [27]. Along the ventral stream, new selectivity emerges for relative disparity in V2 [182], first-order disparity in V4 [70] and finally second-order disparity in IT [70], [80]. Along the dorsal stream areas V3 and V3A encode primarily absolute disparities [5], area MT encodes absolute, relative and first-order disparity [97], [191], [123], area CIP encodes primarily first-order disparity [190], and AIP second-order disparities [168]. As with every other visual feature representation, the receptive fields of the neurons become larger and the latencies become longer. Secondly, at every level in the hierarchy the neural selectivity of the previous level(s) is reiterated such that at the highest levels in the hierarchy (e.g. IT cortex) selectivity for zero-, first- and second-order disparities can be measured [81].

Thirdly, in the visual hierarchy there seems to be a considerable amount of parallel processing of 3D shape information. Thus the end-stage areas of both the ventral and the dorsal visual stream (area AIP), each contain a separate representation of 3D shape [79], [168]. These representations are distinct because the properties of 3D-shape selective neurons differ markedly between IT and AIP: the coding of 3D shape in AIP is faster (shorter latencies), coarser (less sensitivity to discontinuities in the surfaces), less categorical and more boundary-based (less influence of the surface information) compared to IT [180], [77]. Finally, the two neural representations become more tailored towards the behavioral goal that the two processing streams support: in IT the 3D-shape representation subserves categorization of 3D shapes [197], but in AIP most 3D-shape selective neurons also respond during grasping [180]. In contrast, selectivity for anticorrelated disparities (in which each black dot in one eye corresponds to a white dot in the other eye and no depth can be perceived) is present in V1 [26], MT [96] and MST [171], weak in V4 [172] but absent in IT [78] or AIP [180], presumably because the latter areas are not involved in eye movements, which are strongly modulated by anticorrelated disparity [110].

7.4 Motion

The pattern of motion that is induced on the retina when one moves through the environment provides information about one's own motion and about the structure of the environment

[61]. The motion pathway extracts this information from the optic flow.

The first steps of motion analysis in V1 involve the computation of local spatiotemporal motion energy from the dynamics of the retinal image [195], [179]. A mid-level representation in area MT computes basic motion features such as 2D direction and speed based on the V1 inputs [165]. This computation needs to solve several difficult problems. First, local motion energy calculation by spatiotemporal receptive fields in V1 measures only the direction normal to the orientation of a moving bar or grating (the aperture problem). Secondly, spatiotemporal receptive fields cannot calculate speed but only spatiotemporal frequency. Speed tuning corresponds to orientation in spatiotemporal frequency. Most V1 neurons respond to a specific combination of spatial and temporal frequency whereas truly speed-tuned neurons respond to a preferred speed v over a range of spatial and temporal frequency s.t.: $v = df/dt$. Both problems are solved in MT by combining signals from many different V1 cells [165], [133]. However some complex cells in V1 have also been found to already solve these problems [128].

As indicated in figure 5 (6th column), the spatial integration that is needed to perform this integration leads to larger receptive fields in MT and thus has the effect of spatially smoothing the motion pattern. However, this smoothing is well adapted to the structure of the optic flow and preserves self-motion information [18]. Different weighting of inputs within the MT receptive fields moreover allows new motion features to be computed such as differential motion (differences between motion directions at adjacent positions in the visual field), motion edges, and gradients of the motion field [127]. These higher order motion signals are directly related to properties of surfaces in the scene. An important signal that carries this information is motion parallax, i.e. the difference in speed of two objects at different distances from a moving observer. The sensitivity of MT neurons to motion edges and locally opposite motion can be used to extract motion parallax from the optic flow. Motion processing is combined with disparity analysis in MT in order to separate motion signals from different depths [99].

The extraction of information about self-motion is a function of area MST. MST neurons have very large, bilateral receptive fields and respond to motion patterns. The patterns include expansion, contraction, rotation, and more generally speaking, spirals [176], [34]. One way to look at MST is thus in terms of pattern analysis. However, MST is better understood in terms of self-motion analysis [100]. Self-motion describes the translation and rotation of the eye of the observer in space, i.e. the 6 degrees of freedom of any rigid body motion. Single MST neurons are tuned to particular self motions, i.e. to particular translation directions (e.g. forward or rightward) and to rotations as well as to combinations of rotation and translation [100], [67]. MST thus contains a representation of self-motion.

Motion processing is linked to smooth pursuit eye movements. When one tracks a moving target with the eyes, the target is stable on the retina while the background is sweeping across the retina. The target, however, is perceived

to move and the background is perceived as stable. Some cells in MST respond to motion in the world rather than motion on the retina, by combining visual information with extraretinal information about ongoing eye movements [41]. This combination of visual and extraretinal signals is also useful for self-motion analysis when one does not look in the direction of movement but fixates and tracks an object in the visual field [99]. Vestibular input about the state of self-motion is also combined with vision in MST [67].

In summary, the analysis of motion in the primate visual system proceeds in a hierarchy from V1 (local spatiotemporal filtering) to MT (2D motion) to MST (self-motion, motion in world coordinates). Along this hierarchy several computational problems are solved, the features become more complex, receptive fields become larger, and spatial integration of motion signals increases. The representation shifts from one of motion in the visual field (V1, MT) to one of motion in the world and motion of oneself in the world (MST). Also along this hierarchy, visual motion processing is combined with disparity (MT, MST), eye movement information (MST), and vestibular signals (MST). The representation becomes thus less tied to the image and more to the action of the body.

7.5 Object Recognition

Object recognition goes beyond simple 2D-shape perception in several aspects: integration of different cues and modalities, invariance to in-depth rotation and articulated movement, use of context. It is also important to distinguish between-class discrimination (object categorization) and within-class discrimination of objects.

Some integration of different cues is done already for 2D-shape perception. For instance, edges can be defined by luminance in V1, by textures in V2 and by differences in motion in V4. However, color and shape seems to be processed rather independently until high up in the hierarchy. Motion is processed early on, but it is used for object recognition in a different way than for shape perception. For instance, one can recognize familiar people from great distance by their characteristic gait. Other modalities, such as sound and odor, obviously also contribute to object recognition.

It appears that the units in IT pull together various features of medium complexity from lower levels in the ventral stream to build models of object parts. Precise granularity of these parts has not been established at present time, although there are indications that they span different sizes of receptive fields and are possibly tuned to different levels of feature invariance (abstraction) [174]. Computational models that can predominantly be described as compositional hierarchies (the hierarchical organization of categorical representations) define/learn units that are not inconsistent with these findings. For example, it has been shown that features that have been learned (in an unsupervised manner) at the level that roughly corresponds to IT contain sufficient information for reliable classification of object categories (this can be related to readout experiments [74]). Some of the related computational models could also help in making predictions regarding the need for massive feedback (from IT to LGN/V1) and alleviate the problems

with the stimulus-reduction method as these stimuli could be generated through a learning procedure [46].

Rotation in depth usually changes the shape of an object quite dramatically. However, a small fraction of IT neurons can exhibit some rotation invariance and speed of recognition of familiar objects does not depend on the rotation angle [79]. A particular case are face sensitive neurons which can show a rather large invariance to rotations in depth. Representations of the same object under different angles are presumably combined into a rotation invariant representation like simple cell responses might be combined into a complex cell response. Comparing unfamiliar objects from different perspectives seems to require mental rotation and requires extra time that is proportional to the rotation angle [154].

Context plays a major role in object recognition [124] and can be of different nature – semantic, spatial configuration or pose – and is, at least partially, provided by higher areas beyond IT. A simple example are the words ‘THE’ and ‘CAT’, which can be written with an identical character in the center with a shape somewhere between an ‘H’ and an ‘A’. We recognize this very same shape immediately in the appropriate way depending on the context of the surrounding two letters. But we are also faster to recognize a sofa in a living room than floating in the air or a street scene. Interestingly, objects also help to recognize the context and context may be defined on a crude statistical level [124].

Some people have perfectly good object recognition capabilities but cannot recognize faces, a deficit known as *prosopagnosia*, although they can recognize people by their clothes or voices. The FFA (*fusiform face area*) seems the brain structure for face recognition [85]. There is evidence that prosopagnosia not only affects face processing but that it is a deficit in telling apart instances from the same category. For instance bird-watcher with prosopagnosia cannot tell birds apart anymore and car experts cannot make fine car distinctions [58].

It is interesting that in human subjects highly selective neurons have been described that may support object recognition. For example, recordings from epileptic patients in the medial temporal lobe have shown that single neurons reliably respond to particular objects, like the tower of Pisa, in whatever image [139].

7.6 Action Affordances

To supply visual information to the planning and control of action, the visual system extracts specific action-relevant features in hierarchical processing along the occipital and dorsal pathways. This processing is characterized by successively increasing complexity, multi-sensory integration, and a shift from general visual representations to representation specific for particular effectors and actions. Moreover, this processing is to some degree independent of conscious perception, such that lesion patients may be able to interact correctly with objects they fail to recognize and vice versa [64].

Early stages in the dorsal stream hierarchy (V1, V2, MT) are concerned with visual feature extraction (location, orientation, motion,) and the estimation of action-relevant objects features, such as surface orientation, from different cues (motion: MT,

stereo: CIP). These features are encoded in a retinotopic frame of reference. Hierarchically higher areas encode information in spatiotopic or head-centric reference frames, sometimes at the single cell level (as in area VIP [35]) and often in a population code (areas MST, LIP, 7A, MIP) [137]. A major function of the dorsal stream thus lies in coordinate transformations.

These transformations are necessary because the planning of action with different effectors needs to consider targets in different reference frames. Eye movements are best encoded in a retinocentric representation but reach movements need a transformation to arm coordinates, and hence a representation of the target in space. It is not always clear what the best encoding for a particular action is, but the areas in the parietal cortex provide a number of parallel encodings for different tasks.

A further issue for these transformations lies in the combination of vision with other sensory or motor signals. Along the processing in the dorsal stream visual information is combined with vestibular (in MST, VIP), auditory (in LIP), somatosensory (in VIP), and proprioceptive or motor feedback signals (MST and VIP for smooth eye movements, LIP for saccades, MST/VIP/7A/MIP for eye position). Since these signals come in different sensory representations, the combination with vision requires extensive spatial transformations.

Eventually, higher areas in the dorsal stream construct spatial representations that are specialized to provide information for specific actions: LIP represents salience in the visual scene as a target signal for eye movements, MIP and AIP provide information for reaching (target signals) and grasping (shape signals). LIP and VIP provide information for the control of self-motion. Therefore, the processing of action-relevant visual information in the dorsal stream is characterized by a separation of functions, unlike processing in the ventral stream, which is focused on the perception of objects.

8 WHAT CAN WE LEARN FROM THE VISUAL SYSTEM FOR COMPUTER VISION?

What can we learn from the primate visual system for computer vision systems as well as the learning of deep hierarchies? We believe that there are at least four design principles of the former that could be advantageous also for the latter: hierarchical processing¹¹, separation of information channels, feedback and an appropriate balance between prior coded structure and learning.

8.1 Hierarchical Processing

One prominent feature of the primate visual system is its hierarchical architecture consisting of many areas that can roughly be ordered in a sequence with first a common early processing and then a split into two interacting pathways, see Figure 2 and 5. Each pathway computes progressively more complex and invariant representations. What are the possible advantages of such an architecture?

11. In Introduction, we listed several authors who have in various ways studied and demonstrated this principle.

Computational efficiency: The brain is a machine with an enormous number of relatively simple and slow processing units, the neurons. Thus, significant performance can only be achieved if the computation is distributed efficiently. A visual hierarchical network does this spatially as well as sequentially. The spatial partitioning results in localized receptive fields, and the sequential partitioning results in the different areas that gradually compute more and more complex features. Thus, computation is heavily parallelized and pipelined. On a PC, this is less of an issue because it has only one or few but very fast processing units. However, this might change with GPUs or other computer architectures in the future and then the high degree of parallelization of hierarchical networks might be a real plus.

Computational efficiency in the primate visual system also arises from the fact that a lot of processing is reused for several different purposes. The occipital part, which constitutes most of the visual cortex, provides a generic representation that is used for object recognition, navigation, grasping, etc. This saves a lot of computation.

Learning efficiency: Equally important as the computational efficiency during the inference process is the learning efficiency. Hierarchical processing helps in that it provides several different levels of features that already have proven to be useful and robust in some tasks. Learning new tasks can build on these and can be fast because appropriate features at a relatively high level are available already. For instance invariance properties can simply be inherited from the features and do not have to be learned again.

Hierarchical processing, in particular in conjunction with the progression of receptive field sizes (see Table 1, column 3), offers mechanisms that may alleviate the overfitting problem. Namely, small size receptive fields in the lower hierarchical layers limit the potential variability of the features inside the receptive fields and consequently confine the units to learn low dimensional features, which can be sampled with relatively few training examples [46]. The process is recursively applied throughout the hierarchy resulting in a controlled progression in the overall complexity of units on the higher layers. This corresponds to an implicit regularization.

It is important to note that biological visual systems mature in complexity and sophistication in an intertwined process of development (through growing neural substrate) and learning (tuning of neural units) in a sequence of stages. From the computational point of view, this has important implications that deserve more attention in the future.

The world is hierarchical: Even within the brain is the visual system extreme in that has such a deep hierarchy. This may have to do with the complexity of the vision problem or the importance vision has for us. But it might also be a consequence of the fact that the (visual) world around us is spatially laid out and structured hierarchically. Objects can be naturally split into parts and subparts, complex features and simple features, which makes hierarchical processing useful. Nearby points in the visual field are much more related than distant points, which makes local processing within limited receptive fields effective at lower levels.

8.2 Separation of Information Channels

Another prominent feature of the visual system is the separation of information channels. Color, motion, shape etc. are processed separately, even in separate anatomical structures, for quite some time before they are integrated in higher areas. Some of these features are even duplicated in the dorsal and the ventral pathway but with different characteristics and used for different purposes. We believe this has at least two reasons: availability of information and efficiency of representation.

Availability of information: Depending on the circumstances, some of the information channels may not be available at all times. If we look at a photograph, depth and motion are not available. If it is dark, color is not available. If it is foggy high resolution shape information is not available, and motion and color might be the more reliable cues. A representation that would integrate all cues at once would be seriously compromised if one of the cues is missing. Separating the information channels provides robustness with respect to the availability of the different information cues.

Efficiency of representation: Separating the information channels naturally results in a factorial code; an integrated representation would yield a combinatorial code, which is known to suffer from the combinatorial explosion and also does not generalize well to new objects. If we represent four colors and four shapes separately, we can represent 16 different object more efficiently, i.e. with fewer units, than if we would represent each object as a unique color/shape combination. And also if we have seen only a few of the 16 possible combinations, we still can learn and represent unseen combinations easily.

It has been suggested that the binding problem, which arises because different neurons process different visual features of the same object (e.g. color and shape), is solved by means of neuronal synchronization in the temporal domain [40], [146]. In this ‘binding by synchronization’ hypothesis, neurons throughout the cortex encoding features of the same object would show synchronous activity, which would act as a ‘label’ that would indicate that the different features belong to the same object. However, experimental support for the synchronization hypothesis has been mixed [98], [33], [159], and no experiment has unambiguously proven that synchrony is necessary for binding.

8.3 Feedback

While we have outlined in this paper a hierarchical feedforward view on visual processing, it is important to remember that within the visual cortex there are generally more feedback connections than forward connections. Also lateral connections play an important role. This hints at the importance of processes like attention, expectation, top-down reasoning, imagination, and filling in. Many computer vision systems try to work in a purely feed-forward fashion. However, vision is inherently ambiguous and benefits from any prior knowledge available. This may even imply that the knowledge of how the tower of Pisa looks influences the perception of an edge on the level of V1. It also means that a system should be able to

produce several hypotheses that are concurrently considered and possibly not resolved [102].

8.4 Development and Learning of Visual Processing Hierarchies

In this paper, we focused on a description of and lessons to be learned from the end product, the functional visual system of the adult primate. We do not have the space here to discuss what is known about the development [194] and learning of biological visual processing hierarchies (e.g., [105], [113]). However, there are some fairly obvious conclusions relevant to computer vision.

First, in contrast to most deeply hierarchical computer vision systems, the primate visual processing hierarchy does not consist of a homogeneous stack of similar layers that are trained either bottom-up or in a uniform fashion. Rather, it consists of heterogeneous and specialized (horizontal) layers and (vertical) streams that differ considerably in their functions. Thus, a conceptually simple, generic vision system may not be achievable. It may be that biology has instead chosen to optimize specialized functions and their integration into a perceptual whole. It remains to be seen however, whether the specialization of cortical areas is due to fundamentally different mechanisms or to differences in the input and the particular combination of a very small set of learning principles (see, e.g. [31], [91]).

An aspect of these heterogeneous layers and streams that should be of interest to computer vision is that these distinct functional units and intermediate representations provide structural guidance for the design of hierarchical learning systems. As discussed by Bengio [9], this constitutes a way of decomposing the huge end-to-end learning problem into a sequence of simpler problems (see also p. 2).

Secondly, biological vision systems arise due to interactions between genetically-encoded structural biases and exposure to visual signals. One might argue that this is precisely how today's computer vision systems are conceived: The computational procedure is designed by hand, and its parameters are tuned using training data. However, inhomogeneous processing hierarchies require dedicated learning methods at various stages. Mounting evidence for adult cortical plasticity suggests that the influence of learning on cortical processing is much more profound than the tuning of synaptic strengths within fixed neural architectures [66], [62].

9 CONCLUSION

We have reviewed basic facts about the primate visual system, mainly on a functional level relevant for visual processing. We believe that the visual system still is very valuable as a proof of principle and a source of inspiration for building artificial vision systems. We have in particular argued for hierarchical processing with a separation of information channels at lower levels. Moreover, concrete design choices which are crucial for or potentially facilitate the learning of deep hierarchies (such as the structure of intermediate representations, the number of layers and the basic connectivity structure between layers) can be motivated from the biological model. Main stream computer vision, however, seems to follow design principles that are quite different from what we know from primates. We hope that the review and the thoughts presented here help in reconsidering this general trend and encourage the development of flexible and multi-purpose vision modules that can contribute to a hierarchical architecture for artificial vision systems.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience. We would also like to thank Michael D'Zmura for fruitful discussions.

REFERENCES

- [1] Y. Amit. *2D Object Detection and Recognition: Models, Algorithms and Networks*. MIT Press, Cambridge, 2002.
- [2] Y. Amit and D. Geman. A computational model for visual selection. *Neural Comp.*, 11(7):1691–1715, 1999.
- [3] R. Andersen, A. Batista, L. Snyder, C. Buneo, and Y. Cohen. Programming to look and reach in the posterior parietal cortex. In M. Gazzaniga, editor, *The New Cognitive Neurosciences*, chapter 36, pages 515–524. MIT Press, 2 edition, 2000.
- [4] R. Andersen and V. B. Mountcastle. The influence of the angle of gaze upon the excitability of the light-sensitive neurons of the posterior parietal cortex. *J. Neurosci.*, 3(3):532–548, 1983.
- [5] A. Anzai, S. Chowdhury, and G. DeAngelis. Coding of stereoscopic depth information in visual areas v3 and v3a. *The Journal of Neuroscience*, 31(28):10270–10282, 2011.
- [6] L. Bazzani, N. Freitas, H. Larochelle, V. Murino, and J.-A. Ting. Learning attentional policies for tracking and recognition in video with deep networks. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 937–944, New York, NY, USA, June 2011. ACM.
- [7] S. Becker and G. E. Hinton. A self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, 355(6356):161–163, 1992.
- [8] A. J. Bell and T. J. Sejnowski. The ‘independent components’ of natural scenes are edge filters. *Vision Res.*, 37(23):3327–3338, 1997.
- [9] Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.
- [10] P. Berkes and L. Wiskott. Slow feature analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, 5(6):579–602, 2005. <http://journalofvision.org/5/6/9/>, doi:10.1167/5.6.9.
- [11] J. W. Bisley and M. E. Goldberg. Attention, intention, and priority in the parietal lobe. *Annu. Rev. Neurosci.*, 33:1–21, 2010.
- [12] E. Borra, A. Belmalih, R. Calzavara, M. Gerbella, A. Murata, S. Rozzi, and G. Luppino. Cortical connections of the macaque anterior intraparietal (aip) area. *Cerebral Cortex*, 18(5):1094, 2008.
- [13] J. Bowmaker and H. Dartnall. Visual pigments of rods and cones in a human retina. *The Journal of Physiology*, 298:501–511, 1980.
- [14] D. C. Bradley, G. C. Chang, and R. A. Andersen. Encoding of three-dimensional structure-from-motion by primate area MT neurons. *Nature*, 392:714–717, 1998.
- [15] F. Bremmer, M. Kubischik, M. Pekel, M. Lappe, and K.-P. Hoffmann. Linear vestibular self-motion signals in monkey medial superior temporal area. *Ann. N.Y. Acad. Sci.*, 871:272–281, 1999.
- [16] M. Brown, D. Burschka, and G. Hager. Advances in computational stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(8):993–1008, 2003.
- [17] E. Brunswik and J. Kamiya. Ecological cue–validity of ‘proximity’ and of other Gestalt factors. *American Journal of Psychology*, LXVI:20–32, 1953.
- [18] D. Calow, N. Krüger, F. Wörgötter, and M. Lappe. Biologically motivated space-variant filtering for robust optic flow processing. *Network*, 16(4):323–340, 2005.
- [19] M. Carandini and D. J. Heeger. Normalization as a canonical neural computation. *Nature neuroscience*, 13:51–62, 2012.
- [20] M. S. Caywood, B. Willmore, and D. J. Tolhurst. Independent components of color natural scenes resemble V1 neurons in their spatial and color tuning. *J. Neurophysiol.*, 91(6):2859–2873, Jun 2004.
- [21] C. Colby and M. Goldberg. Space and attention in parietal cortex. *Annual Review of Neuroscience*, 22(1):319–349, 1999.
- [22] B. Conway. Spatial structure of cone inputs to color cells in alert macaque primary visual cortex (v-1). *The Journal of Neuroscience*, 21(8):2768–2783, 2001.
- [23] B. R. Conway. Color vision, cones, and color-coding in the cortex. *Neuroscientist*, 15:274–290, Jun 2009.
- [24] B. R. Conway, S. Moeller, and D. Y. Tsao. Specialized color modules in macaque extrastriate cortex. *Neuron*, 56:560–573, Nov 2007.
- [25] H. Cui and R. Andersen. Posterior parietal cortex encodes autonomously selected motor plans. *Neuron*, 56(3):552–559, 2007.
- [26] B. Cumming and A. Parker. Responses of primary visual cortical neurons to binocular disparity without depth perception. *Nature*, 389(6648):280–283, 1997.
- [27] B. Cumming and A. Parker. Binocular neurons in v1 of awake monkeys are selective for absolute, not relative, disparity. *The Journal of Neuroscience*, 19(13):5602–5618, 1999.
- [28] C. Curcio and K. Allen. Topography of ganglion cells in human retina. *The Journal of Comparative Neurology*, 300(1):525, 1990.
- [29] J. Daugman. Complete discrete 2-d gabor transforms by neural networks for image analysis and compression. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 36(7):1169–1179, jul 1988.
- [30] G. C. De Angelis, B. G. Cumming, and W. T. Newsome. Cortical area MT and the perception of stereoscopic depth. *Nature*, 394:677–680, 1998.
- [31] J. J. DiCarlo and D. D. Cox. Untangling invariant object recognition. *Trends in Cognitive Sciences*, 11(8):333–341, 2007.
- [32] S. Dickinson. The evolution of object categorization and the challenge of image abstraction. In S. Dickinson, A. Leonardis, B. Schiele, and T. M., editors, *Object Categorization: Computer and Human Vision Perspectives*, pages 1–37. Cambridge Univ. Press Cambridge, UK, 2009.
- [33] Y. Dong, S. Mihalas, F. Qiu, R. von der Heydt, and E. Niebur. Synchrony and the binding problem in macaque visual cortex. *Journal of Vision*, 8(7), 2008.
- [34] C. J. Duffy and R. H. Wurtz. Sensitivity of MST neurons to optic flow stimuli. II. mechanisms of response selectivity revealed by small-field stimuli. *J. Neurophysiol.*, 65:1346–1359, 1991.
- [35] J. Duhamel, F. Bremmer, S. BenHamed, and W. Graf. Spatial invariance of visual receptive fields in parietal cortex neurons. *Nature*, 389(6653):845–848, 1997.
- [36] J. Duhamel, C. Colby, and M. Goldberg. The updating of the representation of visual space in parietal cortex by intended eye movements. *Science*, 255(5040):90, 1992.
- [37] M. R. Dürsteler and R. H. Wurtz. Pursuit and optokinetic deficits following chemical lesions of cortical areas MT and MST. *J. Neurophysiol.*, 60:940–965, 1988.
- [38] W. Einhäuser, C. Kayser, P. König, and K. P. Körding. Learning the invariance properties of complex cells from their responses to natural stimuli. *Euro J Neurosci*, pages 475–486, February 2002.
- [39] J. H. Elder. Are edges incomplete? *International Journal of Computer Vision*, 34:97–122, 1999.
- [40] A. Engel, P. Roelfsema, P. Fries, M. Brecht, and W. Singer. Role of the temporal domain for response selection and perceptual binding. *Cerebral Cortex*, 7(6):571–582, 1997.
- [41] R. G. Erickson and P. Thier. A neuronal correlate of spatial stability during periods of self-induced visual motion. *Exp. Brain Res.*, 86:608–616, 1991.
- [42] G. J. Ettinger. Hierarchical object recognition using libraries of parameterized model sub-parts. Technical report, MIT, 1987.
- [43] F. Fang, H. Boyaci, and D. Kersten. Border ownership selectivity in human early visual cortex and its modulation by attention. *The Journal of Neuroscience*, 29(2):460–465, 2009.
- [44] D. Felleman and D. V. Essen. Distributed hierarchical processing in primate cerebral cortex. *Cerebral Cortex*, 1:1–47, 1991.
- [45] S. Fidler, M. Boben, and A. Leonardis. Similarity-based cross-layered hierarchical representation for object categorization. In *CVPR*, 2008.
- [46] S. Fidler, M. Boben, and A. Leonardis. Learning hierarchical compositional representations of object structure. In S. Dickinson, A. Leonardis, B. Schiele, and T. M., editors, *Object Categorization: Computer and Human Vision Perspectives*, pages 196–215. Cambridge Univ. Press Cambridge, UK, 2009.
- [47] G. Finlayson and S. Hordley. Color constancy at a pixel. *J. Opt. Soc. Am. A*, 18:253–264, 2001.
- [48] I. M. Finn and D. Ferster. Computational diversity in complex cells of cat primary visual cortex. *Journal of Neuroscience*, 27(36):9638–9648, 2007.
- [49] D. J. Fleet, A. D. Jepson, and M. R. Jenkin. Phase-based disparity measurement. *CVGIP: Image Understanding*, 53(2):198–210, 1991.
- [50] D. J. Fleet, H. Wagner, and D. J. Heeger. Neural encoding of binocular disparity: energy models, position shifts and phase shifts. *Vision Res.*, 36(12):1839–1857, Jun 1996.
- [51] M. Franzius, N. Wilbert, and L. Wiskott. Invariant object recognition and pose estimation with slow feature analysis. *Neural Computation*, 23(9):2289–2323, 2011.
- [52] D. Freedman and J. Assad. Experience-dependent representation of visual categories in parietal cortex. *Nature*, 443(7107):85–88, 2006.
- [53] D. Freedman, M. Riesenhuber, T. Poggio, and E. Miller. Categorical representation of visual stimuli in the primate prefrontal cortex. *Science*, 291(5502):312–316, 2001.
- [54] K. Fukushima, S. Miyake, and T. Ito. Neocognitron: a neural network model for a mechanism of visual pattern recognition. *IEEE Systems, Man and Cybernetics*, 13(3):826–834, 1983.
- [55] A. Gail and R. Andersen. Neural dynamics in monkey parietal reach region reflect context-specific sensorimotor transformations. *The Journal of Neuroscience*, 26(37):9376–9384, 2006.

- [56] V. Gallese, A. Murata, M. Kaseda, N. Niki, and H. Sakata. Deficit of hand preshaping after muscimol injection in monkey parietal cortex. *Neuroreport*, 5(12):1525–1529, 1994.
- [57] C. Galletti, P. P. Battaglini, and P. Fattori. ‘real-motion’ cells in area v3a of macaque visual cortex. *Exp. Brain Res.*, 82:67–76, 1990.
- [58] I. Gauthier, P. Skudlarski, J. C. Gore, and A. W. Anderson. Expertise for cars and birds recruits brain areas involved in face recognition. *Nat Neurosci*, 3(2):191–197, 2000.
- [59] S. Geman. Hierarchy in machine and natural vision. In *Proceedings of the 11th Scandinavian Conference on Image Analysis*, 1999.
- [60] S. Geman, D. Potter, and Z. Chi. Composition systems. *Quarterly of Applied Mathematics*, 60(4):707–736, 2002.
- [61] J. Gibson. The perception of visual surfaces. *American Journal of Psychology*, 63(367–384.), 1950.
- [62] C. Gilbert and W. Li. Adult visual cortical plasticity. *Neuron*, 75(2):250–264, 7 2012.
- [63] I. Gödecke and T. Bonhoeffer. Development of identical orientation maps for two eyes without common visual experience. *Nature*, 379:251–255, 1996.
- [64] M. A. Goodale and A. D. Milner. Separate visual pathways for perception and action. *Trends Neurosci.*, 15(1):20–25, 1992.
- [65] J. P. Gottlieb, M. Kusunoki, and M. E. Goldberg. The representation of visual salience in monkey parietal cortex. *Nature*, 391:481–484, 1998.
- [66] E. Gould, A. Reeves, M. Graziano, and C. Gross. Neurogenesis in the neocortex of adult primates. *Science*, 286(5439):548–552, 1999.
- [67] Y. Gu, P. Watkins, D. Angelaki, and G. DeAngelis. Visual and nonvisual contributions to three-dimensional heading selectivity in the medial superior temporal area. *J. Neurosci.*, 26(1):73–85, 2006.
- [68] M. Hawken and A. Parker. Spatial properties of neurons in the monkey striate cortex. *Proceedings of the Royal Society of London, series B, Biological Sciences*, 231:251–288, 1987.
- [69] J. Hawkins and S. Blakeslee. *On Intelligence*. Times Books, 2004.
- [70] D. Hinkle, C. Connor, et al. Three-dimensional orientation tuning in macaque area v4. *Nature Neuroscience*, 5(7):665–670, 2002.
- [71] I. Howard and B. J. Rogers. *Seeing in depth, Vol. 1: Basic mechanisms*. University of Toronto Press, 2002.
- [72] D. Hubel and T. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex. *J. Physiology*, 160:106–154, 1962.
- [73] D. Hubel and T. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *Journal of Physiology*, 195(1):215–243, 1968.
- [74] C. Hung, G. Kreiman, T. Poggio, and J. DiCarlo. Fast readout of object identity from macaque inferior temporal cortex. *Science*, 310(5749):863–866, 2005.
- [75] M. Ito, H. Tamura, I. Fujita, and K. Tanaka. Size and position invariance of neuronal responses in monkey inferotemporal cortex. *Journal of Neurophysiology*, 73(1):218–226, Jan. 1995.
- [76] P. Janssen and M. Shadlen. A representation of the hazard rate of elapsed time in macaque area lip. *Nature Neuroscience*, 8(2):234–241, 2005.
- [77] P. Janssen, R. Vogels, Y. Liu, and G. Orban. Macaque inferior temporal neurons are selective for three-dimensional boundaries and surfaces. *The Journal of Neuroscience*, 21(23):9419–9429, 2001.
- [78] P. Janssen, R. Vogels, Y. Liu, and G. Orban. At least at the level of inferior temporal cortex, the stereo correspondence problem is solved. *Neuron*, 37(4):693–701, 2003.
- [79] P. Janssen, R. Vogels, and G. Orban. Macaque inferior temporal neurons are selective for disparity-defined three-dimensional shapes. *Proceedings of the National Academy of Sciences*, 96(14):8217, 1999.
- [80] P. Janssen, R. Vogels, and G. Orban. Selectivity for 3d shape that reveals distinct areas within macaque inferior temporal cortex. *Science*, 288(5473):2054–2056, 2000.
- [81] P. Janssen, R. Vogels, and G. Orban. Three-dimensional shape coding in inferior temporal cortex. *Neuron*, 27(2):385–397, 2000.
- [82] J. Jones and L. Palmer. An evaluation of the two dimensional Gabor filter model of simple receptive fields in striate cortex. *Journal of Neurophysiology*, 58(6):1223–1258, 1987.
- [83] B. Julesz. *Foundations of cyclopean perception*. U. Chicago Press, 1971.
- [84] E. R. Kandel, J. H. Schwartz, and T. M. Jessel, editors. *Principles of Neural Science*. McGraw-Hill, 4th edition, 2000.
- [85] N. Kanwisher, J. McDermott, and M. M. Chun. The fusiform face area: A module in human extrastriate cortex specialized for face perception. *The Journal of Neuroscience*, 17(11):4302–4311, 1997.
- [86] N. Katsuyama, A. Yamashita, K. Sawada, T. Naganuma, H. Sakata, and M. Taira. Functional and histological properties of caudal intraparietal area of macaque monkey. *Neuroscience*, 167(1):1–10, 2010.
- [87] P. Kellman and M. Arterberry. *The Cradle of Knowledge*. MIT-Press, 1998.
- [88] R. Kiani, H. Esteky, K. Mirpour, and K. Tanaka. Object category structure in response patterns of neuronal population in monkey inferior temporal cortex. *Journal of Neurophysiology*, 97(6):4296–4309, 2007.
- [89] C. Koch. *Biophysics of Computation: Information Processing in Single Neurons*. Oxford University Press, New York., 1999.
- [90] K. Koffka. *Principles of Gestalt Psychology*. New York, 1955.
- [91] P. König and N. Krüger. Perspectives: Symbols as self-emergent entities in an optimization process of feature extraction and predictions. *Biological Cybernetics*, 94(4):325–334, 2006.
- [92] K. Köteles, P. De Maziere, M. Van Hulle, G. Orban, and R. Vogels. Coding of images of materials by macaque inferior temporal cortical neurons. *European Journal of Neuroscience*, 27(2):466–482, 2008.
- [93] Z. Kourtzi and J. J. DiCarlo. Learning and neural plasticity in object recognition. *Curr. Opin. Neurobiol.*, 16:152–158, Apr 2006.
- [94] R. J. Krauzlis and S. G. Lisberger. A model of visually-guided smooth pursuit eye movements based on behavioral observations. *J. Comput. Neurosci.*, 1(4):265–283, 1994.
- [95] J. Kremers. *The primate visual system: a comparative approach*. Wiley, 2005.
- [96] K. Krug, B. Cumming, and A. Parker. Comparing perceptual signals of single v5/mt neurons in two binocular depth tasks. *Journal of Neurophysiology*, 92(3):1586–1596, 2004.
- [97] K. Krug and A. Parker. Neurons in dorsal visual area v5/mt signal relative disparity. *The Journal of Neuroscience*, 31(49):17892–17904, 2011.
- [98] V. Lamme, H. Spekreijse, et al. Neuronal synchrony does not represent texture segregation. *Nature*, 396(6709):362–366, 1998.
- [99] M. Lappe. Functional consequences of an integration of motion and stereopsis in area MT of monkey extrastriate visual cortex. *Neural Comp.*, 8(7):1449–1461, 1996.
- [100] M. Lappe, F. Bremmer, M. Pekel, A. Thiele, and K. P. Hoffmann. Optic flow processing in monkey STS: a theoretical and experimental approach. *J. Neurosci.*, 16(19):6265–6285, 1996.
- [101] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [102] T. Lee and D. Mumford. Hierarchical bayesian inference in the visual cortex. *Journal of the Optical Society of America*, 20(7):1434–1448, 7 2003.
- [103] J. Lewis and D. Van Essen. Corticocortical connections of visual, sensorimotor, and multimodal processing areas in the parietal lobe of the macaque monkey. *The Journal of Comparative Neurology*, 428(1):112–137, 2000.
- [104] N. Li and J. DiCarlo. Unsupervised natural visual experience rapidly reshapes size-invariant object representation in inferior temporal cortex. *Neuron*, 67(6):1062–1075, 2010.
- [105] S. Li, S. D. Mayhew, and Z. Kourtzi. Learning shapes spatiotemporal brain patterns for flexible categorical decisions. *Cerebral Cortex*, In Press, 2011.
- [106] M. S. Livingstone, C. C. Pack, and R. T. Born. Two-dimensional substructure of MT receptive fields. *Neuron*, 30(3):781–793, 2001.
- [107] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 2(60):91–110, 2004.
- [108] B. Manjunath and W. Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, 1996.
- [109] D. Marr. *Vision: A computational investigation into the human representation and processing of visual information*. Freeman, 1977.
- [110] G. Masson, C. Busetini, and F. Miles. Vergence eye movements in response to binocular disparity without depth perception. *Nature*, 389(6648):283–286, 1997.
- [111] T. Matsumora, K. Koida, and H. Komatsu. Relationship between color discrimination and neural responses in the inferior temporal cortex of the monkey. *Journal of Neurophysiology*, 100(6):3361–3374, 2008.
- [112] J. H. R. Maunsell and D. C. van Essen. Functional properties of neurons in middle temporal visual area of the macaque monkey. I. selectivity for stimulus direction, speed, and orientation. *J. Neurophysiol.*, 49(5):1127–1147, 1983.
- [113] S. D. Mayhew, S. Li, and Z. Kourtzi. Learning acts on distinct processes for visual form perception in the human brain. *J. Neurosci.*, 32(3):775–786, 2012.
- [114] B. W. Mel and J. Fiser. Minimizing binding errors using learned conjunctive features. *Neural Computation*, 12(4):731–762, 2000.

- [115] B. W. Mel, D. L. Ruderman, and A. K. A. Translation-invariant orientation tuning in visual “complex” cells could derive from intradendritic computations. *Journal of Neuroscience*, 18(11):4325–4334, 1998.
- [116] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.
- [117] J. A. Movshon, E. H. Adelson, M. S. Gizzi, and W. T. Newsome. The analysis of moving visual patterns. In C. Chagas, R. Gattass, and C. Gross, editors, *Pattern Recognition Mechanisms*, pages 117–151, New York, 1985. Springer.
- [118] J. A. Movshon and W. T. Newsome. Visual response properties of striate cortical neurons projecting to area mt in macaque monkeys. *Journal of Neuroscience*, 16(23):7733–7741, 1996.
- [119] R. Mruczek and D. Sheinberg. Activity of inferior temporal cortical neurons predicts recognition choice behavior and recognition time during visual search. *The Journal of Neuroscience*, 27(11):2825–2836, 2007.
- [120] A. Murata, V. Gallese, G. Luppino, M. Kaseda, and H. Sakata. Selectivity for the shape, size, and orientation of objects for grasping in neurons of monkey parietal area aip. *Journal of Neurophysiology*, 83(5):2580–2601, 2000.
- [121] H. Nakamura, T. Kuroda, M. Wakita, M. Kusunoki, A. Kato, A. Mikami, H. Sakata, and K. Itoh. From three-dimensional space vision to prehensile hand movements: the lateral intraparietal area links the area v3a and the anterior intraparietal area in macaques. *The Journal of Neuroscience*, 21(20):8174–8187, 2001.
- [122] W. T. Newsome, R. H. Wurtz, and H. Komatsu. Relation of cortical areas MT and MST to pursuit eye movements. II. differentiation of retinal from extraretinal inputs. *J. Neurophysiol.*, 60(2):604–620, 1988.
- [123] J. Nguyenkim and G. DeAngelis. Disparity-based coding of three-dimensional surface orientation by macaque middle temporal neurons. *The Journal of Neuroscience*, 23(18):7117–7128, 2003.
- [124] A. Oliva and A. Torralba. The role of context in object recognition. *Trends Cogn. Sci.*, 11:520–527, 2007.
- [125] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, June 1996.
- [126] B. Ommer and J. M. Buhmann. Learning the compositional nature of visual objects. In *CVPR*, 2007.
- [127] G. A. Orban. Higher order visual processing in macaque extrastriate cortex. *Physiol. Rev.*, 88:59–89, Jan 2008.
- [128] C. Pack, M. S. Livingstone, K. Duffy, and R. Born. End-stopping and the aperture problem: Two-dimensional motion signals in macaque v1. *Neuron*, 39:671680, 2003.
- [129] C. C. Pack and R. T. Born. Temporal dynamics of a neural solution to the aperture problem in visual area MT of macaque brain. *Nature*, 409(6823):1040–1042, 2001.
- [130] A. Parker. Binocular depth perception and the cerebral cortex. *Nature Reviews Neuroscience*, 8(5):379–391, 2007.
- [131] A. Pasupathy and C. Connor. Responses to contour features in macaque area v4. *Journal of Neurophysiology*, 82(5):2490, 1999.
- [132] M. Pekel, M. Lappe, F. Bremmer, A. Thiele, and K.-P. Hoffmann. Neuronal responses in the motion pathway of the macaque monkey to natural optic flow stimuli. *NeuroReport*, 7(4):884–888, 1996.
- [133] J. A. Perrone and A. Thiele. Speed skills: measuring the visual speed analyzing properties of primate MT neurons. *Nat. Neurosci.*, 4(5):526–532, 2001.
- [134] B. Pesaran, M. Nelson, and R. Andersen. Free choice activates a decision circuit between frontal and parietal cortex. *Nature*, 453(7193):406–409, 2008.
- [135] R. Peters, A. Iyer, L. Itti, and C. Koch. Components of bottom-up gaze allocation in natural images. *International Journal of Neural Systems*, 45(18):2397–2416, 2005.
- [136] M. Platt and P. Glimcher. Neural correlates of decision variables in parietal cortex. *Nature*, 400(6741):233–238, 1999.
- [137] A. Pouget and T. J. Sejnowski. Spatial transformations in the parietal cortex using basis functions. *J. Cog. Neurosci.*, 9(2):222–237, 1997.
- [138] N. Pugeault, F. Wörgöter, and N. Krüger. Visual primitives: Local, condensed, and semantically rich visual descriptors and their applications in robotics. *International Journal of Humanoid Robotics*, 7(3):379–405, 2010.
- [139] R. Q. Quiroga, L. Reddy, C. Koch, and I. Fried. Decoding visual inputs from multiple neurons in the human temporal lobe. *J Neurophysiol*, 98(4):1997–2007, 2007.
- [140] S. Raiguel, R. Vogels, S. Mysore, and G. Orban. Learning to see the difference specifically alters the most informative v4 neurons. *The Journal of Neuroscience*, 26(24):6589–6602, 2006.
- [141] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 11(2):1019–1025, 1999.
- [142] K. Rockland, J. Kaas, and A. Peters. *Cerebral Cortex: Extrastriate Cortex in Primates*, volume 12. Springer, 1997.
- [143] A. Rodríguez-Sánchez, E. Simine, and J. Tsotsos. Attention and visual search. *International Journal of Neural Systems*, 17(4):275–288, 2007.
- [144] A. J. Rodríguez-Sánchez and J. K. Tsotsos. The importance of intermediate representations for the modeling of 2d shape detection: Endstopping and curvature tuned computations. *Proc. IEEE Computer Vision and Pattern Recognition*, pages 4321–4326, 2011.
- [145] A. J. Rodríguez-Sánchez and J. K. Tsotsos. The roles of endstopped and curvature tuned computations in a hierarchical representation of 2d shape. *PLoS ONE*, 7(8):1–13, 2012.
- [146] P. Roelfsema. Solutions for the binding problem. *Zeitschrift für Naturforschung. C, Journal of biosciences*, 53(7-8):691, 1998.
- [147] E. Rolls, B. Webb, and M. C. A. Booth. Responses of inferior temporal cortex neurons to objects in natural scenes. *Society for Neuroscience Abstracts*, 26:1331, 2000.
- [148] J.-P. Roy and R. H. Wurtz. Disparity sensitivity of neurons in monkey extrastriate area MST. *J. Neurosci.*, 12(7):2478–2492, 1992.
- [149] N. Rust, O. Schwartz, J. Movshon, and E. Simoncelli. Spike-triggered characterization of excitatory and suppressive stimulus dimensions in monkey v1. *Neurocomputing*, 58:793–799, 2004.
- [150] U. Rutishauser, R. J. Douglas, and J. J. Slotine. Collective stability of networks of winner-take-all circuits. *Neural Computation*, 23(3):735–773, 2011.
- [151] H. Sakata, M. Taira, M. Kusunoki, A. Murata, and Y. Tanaka. The parietal association cortex in depth perception and visual control of hand action. *Trends in Neurosciences*, 20(8):350–357, 1997.
- [152] C. D. Salzman, K. H. Britten, and W. T. Newsome. Cortical microstimulation influences perceptual judgements of motion direction. *Nature*, 346(12):174–177, 1990.
- [153] F. Scalzo and J. H. Piater. Statistical learning of visual feature hierarchies. In *Workshop on Learning, CVPR*, 2005.
- [154] S. C. E. Schendan H. E. Mental rotation and object categorization share a common network of prefrontal and dorsal and ventral regions of posterior cortex. *Neuroimage*, 35:1264–1277, 2007.
- [155] A. Schoups, R. Vogels, N. Qian, and G. Orban. Practising orientation identification improves orientation coding in v1 neurons. *Nature*, 412(6846):549–553, 2001.
- [156] A. Sereno and J. Maunsell. Shape selectivity in primate lateral intraparietal cortex. *Nature*, 395(6701):500–503, 1998.
- [157] T. Serre and A. T. Poggio. Neuromorphic approach to computer vision. *Communications of the ACM (online)*, 53, 2010.
- [158] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):411–426, 2007.
- [159] M. Shadlen and J. Movshon. Synchrony unbound: review a critical evaluation of the temporal binding hypothesis. *Neuron*, 24:67–77, 1999.
- [160] M. Shadlen and W. Newsome. Motion perception: seeing and deciding. *Proceedings of the National Academy of Sciences*, 93(2):628, 1996.
- [161] R. Shapley and M. J. Hawken. Color in the cortex: single- and double-opponent cells. *Vision Res.*, 51(7):701–717, Apr 2011.
- [162] X. Shi, N. Bruce, and J. K. Tsotsos. Fast, recurrent, attentional modulation improves saliency representation and scene recognition. *CVPR Workshop on Biologically-Consistent Vision*, pages 1–8, 2011.
- [163] E. Shikata, Y. Tanaka, H. Nakamura, M. Taira, H. Sakata, et al. Selectivity of the parietal visual neurones in 3d orientation of surface of stereoscopic stimuli. *Neuroreport*, 7(14):2389–2394, 1996.
- [164] E. Simoncelli. Vision and the statistics of the visual environment. *Current Opinion in Neurobiology*, 13(2):144–149, 2003.
- [165] E. P. Simoncelli and D. J. Heeger. A model of neuronal responses in visual area MT. *Vis. Res.*, 38(5):743–761, 1998.
- [166] W. Singer. Consciousness and the binding problem. *Annals of the New York Academy of Sciences*, 929(1):123–146, 2001.
- [167] H. Spitzer and S. Hochstein. A complex-cell receptive-field model. *Journal of Neurophysiology*, 53(5):1266–1286, 1985.
- [168] S. Srivastava, G. Orban, P. De Mazière, and P. Janssen. A distinct representation of three-dimensional shape in macaque anterior intraparietal area: fast, metric, and coarse. *The Journal of Neuroscience*, 29(34):10613–10626, 2009.
- [169] S. Swaminathan and D. Freedman. Preferential encoding of visual categories in parietal cortex compared with prefrontal cortex. *Nat Neurosci*, 15:315–320, 2012.
- [170] M. Taira, K. Tsutsui, M. Jiang, K. Yara, and H. Sakata. Parietal neurons represent surface orientation from the gradient of binocular disparity. *Journal of Neurophysiology*, 83(5):3140, 2000.

- [171] A. Takemura, Y. Inoue, K. Kawano, C. Quaia, and F. Miles. Single-unit activity in cortical area mst associated with disparity-vergence eye movements: evidence for population coding. *Journal of Neurophysiology*, 85(5):2245–2266, 2001.
- [172] S. Tanabe, K. Umeda, and I. Fujita. Rejection of false matches for binocular correspondence in macaque visual cortical area v4. *The Journal of Neuroscience*, 24(37):8170–8180, 2004.
- [173] K. Tanaka. Neuronal mechanisms of object recognition. *Science*, 262:685–688, 1993.
- [174] K. Tanaka. Inferotemporal cortex and object vision. *Annual review of neuroscience*, 19(1):109–139, 1996.
- [175] K. Tanaka, H. Saito, Y. Fukada, and M. Moriya. Coding visual images of objects in the inferotemporal cortex of the macaque monkey. *Journal of Neurophysiology*, 66(1):170–189, 1991.
- [176] K. Tanaka and H.-A. Saito. Analysis of motion of the visual field by direction, expansion/contraction, and rotation cells clustered in the dorsal part of the medial superior temporal area of the macaque monkey. *J. Neurophysiol.*, 62(3):626–641, 1989.
- [177] H. Tanigawa, H. D. Lu, and A. W. Roe. Functional organization for color and orientation in macaque V4. *Nat. Neurosci.*, 13:1542–1548, Dec 2010.
- [178] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman. How to grow a mind: Statistics, structure, and abstraction. *Science*, 331:1279–1285, 2011.
- [179] F. E. Theunissen, S. V. David, N. C. Singh, A. Hsu, W. E. Vinje, and J. L. Gallant. Estimating spatio-temporal receptive fields of auditory and visual neurons from their responses to natural stimuli. *Network: Computation in Neural Systems*, 12(3):289–316, 2001.
- [180] T. Theys, S. Srivastava, J. van Loon, J. Goffin, and P. Janssen. Selectivity for three-dimensional contours and surfaces in the anterior intraparietal area. *Journal of Neurophysiology*, 107(3):995–1008, 2012.
- [181] P. Thier and R. Andersen. Electrical microstimulation suggest two different kinds of representation of head-centered space in the intraparietal sulcus of rhesus monkeys. *Proc. Natl. Acad. Sci.* 93, pages 4962–4967, 1996.
- [182] O. Thomas, B. Cumming, and A. Parker. A specialization for relative disparity in v2. *Nature Neuroscience*, 5(5):472–478, 2002.
- [183] T. Tompa and G. Sary. A review on the inferior temporal cortex of the macaque. *Brain Research Reviews*, 62(2):165–182, 2010.
- [184] R. Tootell, K. Nelissen, W. Vanduffel, and G. Orban. Search for color center(s) in macaque visual cortex. *Cerebral Cortex*, 14(4):353–363, 2004.
- [185] M. J. Tovee, E. T. Rolls, and P. Azzopardi. Translation invariance in the responses to faces of single neurons in the temporal visual cortical areas of the alert macaque. *Journal of Neurophysiology*, 72(3):1049–1060, Sept. 1994.
- [186] A. Treisman. The binding problem. *Current Opinion in Neurobiology*, 6(2):171 – 178, 1996.
- [187] A. Treisman and H. Schmidt. Illusory conjunctions in the perception of objects. *Cognitive Psychology*, 14(1):107 – 141, 1982.
- [188] D. Tsao, W. Vanduffel, Y. Sasaki, D. Fize, T. Knutsen, J. Mandeville, L. Wald, A. Dale, B. Rosen, D. Van Essen, et al. Stereopsis activates v3a and caudal intraparietal areas in macaques and humans. *Neuron*, 39(3):555–568, 2003.
- [189] J. K. Tsotsos. Analyzing vision at the complexity level. *Behavioral and Brain Sciences*, 13(3):423–469, 1990.
- [190] K. Tsutsui, H. Sakata, T. Naganuma, and M. Taira. Neural correlates for perception of 3d surface orientation from texture gradient. *Science*, 298(5592):409–412, 2002.
- [191] T. Uka and G. DeAngelis. Linking neural representation to function in stereoscopic depth perception: roles of the middle temporal area in coarse versus fine disparity discrimination. *The Journal of neuroscience*, 26(25):6791–6802, 2006.
- [192] S. Ullman and B. Epshtein. Visual classification by a hierarchy of extended features. In *Towards Category-Level Object Recognition*. Springer-Verlag, 2006.
- [193] L. G. Ungerleider and M. Mishkin. Two cortical visual systems. In D. J. Ingle, M. A. Goodale, and R. J. W. Mansfield, editors, *Analysis of Visual Behavior*, pages 549–586. MIT Press, Cambridge, MA, 1982.
- [194] C. van den Boomen, M. J. van der Smagt, and C. Kemner. Keep your eyes on development: The behavioral and neurophysiological development of visual mechanisms underlying form processing. *Front Psychiatry*, 16(3), 2012.
- [195] J. H. van Hateren and D. L. Ruderman. Independent component analysis of natural image sequences yields spatio-temporal filters similar to simple cells in primary visual cortex. *Proc. Biological Sciences*, 265(1412):2315–2320, 1998.
- [196] B. Verhoef, R. Vogels, and P. Janssen. Contribution of inferior temporal and posterior parietal activity to three-dimensional shape perception. *Current Biology*, 20(10):909–913, 2010.
- [197] B. Verhoef, R. Vogels, and P. Janssen. Inferotemporal cortex subserves three-dimensional structure categorization. *Neuron*, 73:171–182, 2012.
- [198] R. Vogels. Categorization of complex visual images by rhesus monkeys. part 2: single-cell study. *European Journal of Neuroscience*, 11(4):1239–1255, 1999.
- [199] H. von Helmholtz, editor. *Handbuch der physiologischen Optik*. Hamburg & Leipzig: Voss, 1866.
- [200] J. Wagemans, J. Elder, M. Kubovy, S. Palmer, M. Peterson, M. Singh, and R. von der Heydt. A century of gestalt psychology in visual perception: I. perceptual grouping and figure-ground organization. *Psychological Bulletin*, in press.
- [201] L. Wiskott. How does our visual system achieve shift and size invariance? In J. L. van Hemmen and T. J. Sejnowski, editors, *23 Problems in Systems Neuroscience*, chapter 16, pages 322–340. Oxford University Press, New York, 2006.
- [202] L. Wiskott, J.-M. Fellous, N. Kruger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(7):775–779, July 1997.
- [203] J. M. Wolfe. Visual search. In H. Pashler, editor, *Attention*. University College London Press, London, UK, 1998.
- [204] F. Xu and S. Carey. Infants’ metaphysics: The case of numerical identity. *Cognitive Psychology*, 30(2):111–153, APR 1996.
- [205] S. Zeki, S. Aglioti, D. McKeefry, and G. Berlucchi. The neurobiological basis of conscious color perception in a blind patient. *Proceedings of the National Academy of Sciences*, 96:14124–14129, 1999.
- [206] H. Zhou, H. Friedman, and R. Von Der Heydt. Coding of border ownership in monkey visual cortex. *The Journal of Neuroscience*, 20(17):6594–6611, 2000.
- [207] D. Zipser and R. A. Andersen. A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. *Nature*, 331(6158):679–684, 1988.



Norbert Krüger is a professor at the Mærsk McKinney Møller Institute, University of Southern Denmark. He holds a M.Sc. degree from the Ruhr-Universität Bochum, Germany and his Ph.D. degree from the University of Bielefeld. He leads the Cognitive Vision Lab which focuses on computer vision and cognitive systems, in particular the learning of object representations in the context of grasping.



Ales Leonardis is a Chair of Robotics and a Co-Director of the Centre of Computational Neuroscience and Cognitive Robotics at the University of Birmingham. He is also a full professor at the Faculty of Computer and Information Science, University of Ljubljana, and an adjunct professor at the Faculty of Computer Science, Graz University of Technology.



Peter Janssen is professor of neurophysiology in the Laboratorium voor Neuro- en Psychofysiologie at the KU Leuven, Belgium. He holds an MD degree, a masters degree in psychology, and a PhD degree in Biomedical Sciences from the KU Leuven. His research interests are the processing of three-dimensional shape, object analysis in the dorsal visual stream, functional interactions between cortical areas, and the ventral premotor cortex.



Antonio J. Rodríguez-Sánchez is currently a senior research fellow in the department of Computer Science at the University of Innsbruck, Austria. He completed his Ph.D. at York University, Toronto, Canada on the subject modeling attention and intermediate areas of the visual cortex. He is part of the Intelligent and Interactive Systems group and his main interests are computer vision and computational neuroscience.



Sinan Kalkan received his M.Sc. degree in Computer Engineering from Middle East Technical University, Turkey in 2003, and his Ph.D. degree in Informatics from the University of Göttingen, Germany in 2008. He is currently an assistant professor at the Dept. of Computer Engineering, Middle East Technical University. Sinan Kalkan's research interests include biologically motivated Computer Vision and Cognitive Robotics.



Justus Piater is a professor of computer science at the University of Innsbruck, Austria. He holds a M.Sc. degree from the University of Magdeburg, Germany, and M.Sc. and Ph.D. degrees from the University of Massachusetts Amherst, USA. He leads the Intelligent and Interactive Systems group that works on visual perception and inference in dynamic and interactive scenarios, including applications in autonomous robotics and video analysis.



Markus Lappe received a PhD in physics from the University of Tübingen, Germany. He worked on computational and cognitive neuroscience of vision at the Max-Planck Institute of Biological Cybernetics in Tübingen, the National Institutes of Health, Bethesda, USA, and the Department of Biology of the Ruhr-University Bochum, Germany. In 1999 he was awarded the BioFuture prize of the German Federal Ministry of Education and Research. Since 2001 he is full professor of Experimental Psychology at the University of Muenster. He is also a member of the Otto Creutzfeldt Center for Cognitive and Behavioral Neuroscience at the University of Muenster.



Laurenz Wiskott is full professor at the Ruhr-Universität Bochum, Germany. He holds a Diploma degree in Physics from the Universität Osnabrück and a PhD from the Ruhr-Universität Bochum. The stages of his career include The Salk Institute in San Diego, the Institute for Advanced Studies in Berlin, and the Institute for Theoretical Biology, Humboldt-Universität Berlin. He has been working in the fields of Computer Vision, Neural Networks, Machine Learning and Computational Neuroscience.

Voxel Cloud Connectivity Segmentation - Supervoxels for Point Clouds

Jeremie Papon

Alexey Abramov

Markus Schoeler

Florentin Wörgötter

Bernstein Center for Computational Neuroscience (BCCN)

III Physikalisches Institut - Biophysik, Georg-August University of Göttingen

{jpapon, abramov, mschoeler, worgott}@physik3.gwdg.de

Abstract

Unsupervised over-segmentation of an image into regions of perceptually similar pixels, known as superpixels, is a widely used preprocessing step in segmentation algorithms. Superpixel methods reduce the number of regions that must be considered later by more computationally expensive algorithms, with a minimal loss of information. Nevertheless, as some information is inevitably lost, it is vital that superpixels not cross object boundaries, as such errors will propagate through later steps. Existing methods make use of projected color or depth information, but do not consider three dimensional geometric relationships between observed data points which can be used to prevent superpixels from crossing regions of empty space. We propose a novel over-segmentation algorithm which uses voxel relationships to produce over-segmentations which are fully consistent with the spatial geometry of the scene in three dimensional, rather than projective, space. Enforcing the constraint that segmented regions must have spatial connectivity prevents label flow across semantic object boundaries which might otherwise be violated. Additionally, as the algorithm works directly in 3D space, observations from several calibrated RGB+D cameras can be segmented jointly. Experiments on a large data set of human annotated RGB+D images demonstrate a significant reduction in occurrence of clusters crossing object boundaries, while maintaining speeds comparable to state-of-the-art 2D methods.

1. Introduction

Segmentation algorithms aim to group pixels in images into perceptually meaningful regions which conform to object boundaries. While they initially only considered low-level information from the image, recent semantic segmentation methods take advantage of high-level object knowledge to help disambiguate object borders. Graph-based approaches, such as Markov Random Field (MRF) and Condi-

tional Random Field (CRF), have become popular, as they merge relational low-level context within the image with object level class knowledge. While the use of such techniques have met with significant success, they have the drawback that the computational cost of inference on these graphs generally rises sharply with increasing number of nodes. This means that solving graphs with a node for every pixel quickly becomes intractable, which has limited their use in applications which require real-time segmentation.

The cost of solving pixel-level graphs led to the development of mid-level inference schemes which do not use pixels directly, but rather use groupings of pixels, known as superpixels, as the base level for nodes [9]. Superpixels are formed by over-segmenting the image into small regions based on local low-level features, reducing the number of nodes which must be considered for inference. While this scheme has been successfully used in many state-of-the-art algorithms [4, 15], it suffers from one significant disadvantage; mistakes in the over-segmentation which creates the superpixels generally cannot be recovered from and will propagate to later steps in the vision pipeline.

Due to their strong impact on the quality of the eventual segmentation [5], it is important that superpixels have certain characteristics. Of these, avoiding violating object boundaries is the most vital, as failing to do so will decrease the accuracy of classifiers used later - since they will be forced to consider pixels which belong to more than one class. Additionally, even if the classifier does manage a correct output, the final pixel level segmentation will necessarily contain errors. Another useful quality is regular distribution over the area being segmented, as this will produce a simpler graph for later steps.

In this paper, we present a novel method, Voxel Cloud Connectivity Segmentation (VCCS), which takes advantage of 3D geometry provided by RGB+D cameras to generate superpixels which conform to object boundaries better than existing methods, and which are evenly distributed in the actual observed space, rather than the projected image plane. This is accomplished using a seeding method-

ology based in 3D space and a flow-constrained local iterative clustering which uses color and geometric features. In addition to providing superpixels which conform to real geometric relationships, the method also can be used directly on point clouds created by combining several calibrated RGB+D cameras, providing a full 3D supervoxel (the 3D analogue of superpixels) graph at speeds sufficient for robotic applications. Additionally, the method source code is freely distributed as part of the Point Cloud Library [11] (PCL)¹.

The organization of the paper is as follows: first, in Section 2 we give an overview of existing methods. In Section 3 we present the 3D supervoxel segmentation algorithm. In Section 4 we present a qualitative evaluation of the method segmenting 3D point clouds created by merging several cameras. In Section 5 we use standard quantitative measures on results from a large RGB+D semantic segmentation dataset to demonstrate that our algorithm conforms to real object boundaries better than other state-of-the-art methods. Additionally, we present run-time performance results to substantiate the claim that our method is able to offer performance equivalent to the fastest 2D methods. Finally, in Section 6 we discuss the results and conclude.

2. Related Work

There are many existing methods for over-segmenting images into superpixels. These can be generally classified into two subsets - graph-based and gradient ascent methods. In this section, we shall briefly review recent top-performing methods.

Graph-based superpixel methods, similar to graph-based full segmentation methods, consider each pixel as a node in a graph, with edges connecting to neighboring pixels. Edge weights are used to characterize similarity between pixels, and superpixel labels are solved for by minimizing a cost function over the graph. Moore *et al.* [8] produce superpixels which conform to a regular lattice structure by seeking optimal paths horizontally and vertically across a boundary image. This is done using either a graph cuts or dynamic programming method which seeks to minimize the cost of edges and nodes in the paths. While this method does have the advantage of producing superpixels in a regular grid, it sacrifices boundary adherence to so, and furthermore, is heavily dependent on the quality of the pre-computed boundary image.

The Turbopixels [7] method of Levinstein *et al.* uses a geometric flow-based algorithm based on level-set, and enforces a compactness constraint to ensure that superpixels have regular shape. Unfortunately, it is too slow for use in many applications; while the authors claim complexity linear in image size, in practice we experienced run times

over 10 seconds for VGA-sized images. Veksler *et al.* [13], inspired by Turbopixels, use an energy minimization framework to stitch together image patches, using graph-cuts to optimize an explicit energy function. Their method (referred to here as GCb10) is considerably faster than Turbopixels, but still requires several seconds even for small images.

Recently, a significantly faster class of superpixel methods has emerged - Simple Linear Iterative Clustering[1] (SLIC). This is an iterative gradient ascent algorithm which uses a local k-means clustering approach to efficiently find superpixels, clustering pixels in the five dimensional space of color and pixel location. Depth-Adaptive Superpixels[14] recently extended this idea to use depth images, expanding the clustering space with the added dimensions of depth and point normal angles. While DASP is efficient and gives promising results, it does not take full advantage of RGB+D data, remaining in the class of 2.5D methods, as it does not explicitly consider 3D connectivity or geometric flow.

For the sake of clarity, we should emphasize that our method is not related to existing “supervoxel” methods [1, 8, 13], which are simple extensions of 2D algorithms to 3D volumes. In such methods, video frames are stacked to produce a structured, regular, and solid volume with time as the depth dimension. In contrast, our method is intended to segment actual volumes in space, and makes heavy use of the fact that such volumes are not regular or solid (most of the volume is empty space) to aid segmentation. Existing “supervoxel” methods cannot work in such a space, as they generally only function on a structured lattice.

3. Geometrically Constrained Supervoxels

In this Section we present Voxel Cloud Connectivity Segmentation (VCCS), a new method for generating superpixels and supervoxels from 3D point cloud data. The supervoxels produced by VCCS adhere to object boundaries better than state-of-the-art methods while the method remains efficient enough to use in online applications. VCCS uses a variant of k-means clustering for generating its labeling of points, with two important constraints:

1. The seeding of supervoxel clusters is done by partitioning 3D space, rather than the projected image plane. This ensures that supervoxels are evenly distributed according to the geometry of the scene.
2. The iterative clustering algorithm enforces strict spatial connectivity of occupied voxels when considering points for clusters. This means that supervoxels strictly cannot flow across boundaries which are disjoint in 3D space, even though they are connected in the projected plane.

First, in 3.1 we shall describe how neighbor voxels are calculated efficiently, then in 3.2 how seeds are generated and filtered, in 3.3 the features and distance measure used

¹<https://github.com/PointCloudLibrary/pcl/>

for clustering, and finally in 3.4 how the iterative clustering algorithm enforces spatial connectivity. Unless otherwise noted, all processing is being performed in the 3D point-cloud space constructed from one or more RGB+D cameras (or any other source of point-cloud data). Furthermore, because we work exclusively in a voxel-cloud space (rather than the continuous point-cloud space), we shall adopt the following notation to refer to voxel at index i within voxel-cloud V of voxel resolution r :

$$V_r(i) = \mathbf{F}_{1..n}, \quad (1)$$

where \mathbf{F} specifies a feature vector which contains n point features (*e.g.* color, location, normals).

3.1. Adjacency Graph

Adjacency is a key element of the proposed method, as it ensures that supervoxels do not flow across object boundaries which are disconnected in space. There are three definitions of adjacency in a voxelized 3D space; 6-, 18-, or 26-adjacent. These share a face, faces or edges, and faces, edges, or vertices, respectively. In this work, whenever we refer to adjacent voxels, we are speaking of 26-adjacency.

As a preliminary step, we construct the adjacency graph for the voxel-cloud. This can be done efficiently by searching the voxel kd-tree, as for a given voxel, the centers of all 26-adjacent voxels are contained within $\sqrt{3} * R_{voxel}$. R_{voxel} specifies the voxel resolution which will be used for the segmentation (for clarity, we shall simply refer to discrete elements at this resolution as voxels). The adjacency graph thus constructed is used extensively throughout the rest of the algorithm.

3.2. Spatial Seeding

The algorithm begins by selecting a number of seed points which will be used to initialize the supervoxels. In order to do this, we first divide the space into a voxelized grid with a chosen resolution R_{seed} , which is significantly higher than R_{voxel} . The effect of increasing the seed resolution R_{seed} can be seen in Figure 2. Initial candidates for seeding are chosen by selecting the voxel in the cloud nearest to the center of each occupied seeding voxel.

Once we have candidates for seeding, we must filter out seeds caused by noise in the depth image. This means that we must remove seeds which are points isolated in space (which are likely due to noise), while leaving those which exist on surfaces. To do this, we establish a small search radius R_{search} around each seed, and delete seeds which do not have at least as many voxels as would be occupied by a planar surface intersecting with half of the search volume (this is shown by the green plane in Figure 1). Once filtered, we shift the remaining seeds to the connected voxel within the search volume which has the smallest gradient in the

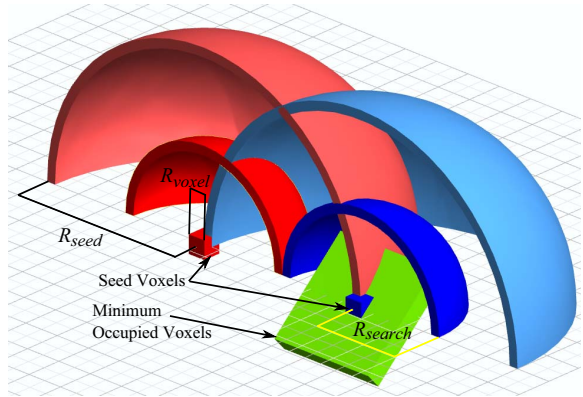


Figure 1. Seeding parameters and filtering criteria. R_{seed} determines the distance between supervoxels, while R_{voxel} determines the resolution to which the cloud is quantized. R_{search} is used to determine if there are a sufficient number of occupied voxels to necessitate a seed.

search volume. Gradient is computed as

$$G(i) = \sum_{k \in V_{adj}} \frac{\|V(i) - V(k)\|_{CIELab}}{N_{adj}}; \quad (2)$$

we use sum of distances in CIELAB space from neighboring voxels, requiring us to normalize the gradient measure by number of connected adjacent voxels N_{adj} . Figure 1 gives an overview of the different distances and parameters involved in seeding.

Once the seed voxels have been selected, we initialize the supervoxel feature vector by finding the center (in feature space) of the seed voxel and connected neighbors within 2 voxels.

3.3. Features and Distance Measure

VCCS supervoxels are clusters in a 39 dimensional space, given as

$$\mathbf{F} = [x, y, z, L, a, b, \text{FPFH}_{1..33}], \quad (3)$$

where x, y, z are spatial coordinates, L, a, b are color in CIELab space, and $\text{FPFH}_{1..33}$ are the 33 elements of Fast Point Feature Histograms (FPFH), a local geometrical feature proposed by Rusu *et al.* [10]. FPFH are pose-invariant features which describe the local surface model properties of points using combinations of their k nearest neighbors. They are an extension of the older Point Feature Histograms optimized for speed, and have a computational complexity of $O(n \cdot k)$.

In order to calculate distances in this space, we must first normalize the spatial component, as distances, and thus their relative importance, will vary depending on the seed resolution R_{seed} . Similar to the work of Achanta *et al.*, [1] we have limited the search space for each cluster so that it

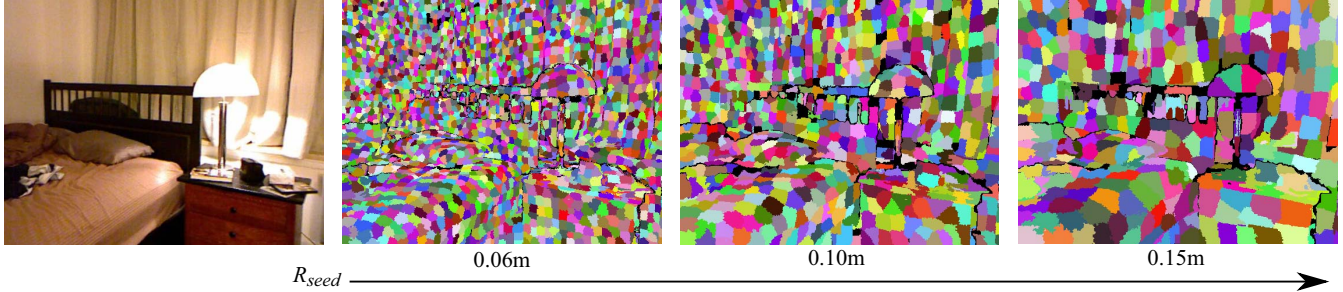


Figure 2. Image segmented using VCCS with seed resolutions of 0.1, 0.15 and 0.2 meters.

ends at the neighboring cluster centers. This means that we can normalize our spatial distance D_s using the maximally distant point considered for clustering, which will lie at a distance of $\sqrt{3}R_{seed}$. Color distance D_c , is the euclidean distance in CIELab space, normalized by a constant m . Distance in FPFH space, D_f , is calculated using the Histogram Intersection Kernel [2]. This leads us to a equation for normalized distance D :

$$D = \sqrt{\frac{\lambda D_c^2}{m^2} + \frac{\mu D_s^2}{3R_{seed}^2} + \epsilon D_{HiK}^2}, \quad (4)$$

where λ , μ , and ϵ control the influence of color, spatial distance, and geometric similarity, respectively, in the clustering. In practice we keep the spatial distance constant relative to the other two so that supervoxels occupy a relatively spherical space, but this is not strictly necessary. For the experiments in this paper we have color weighted equally with geometric similarity.

3.4. Flow Constrained Clustering

Assigning voxels to supervoxels is done iteratively, using a local k-means clustering related to [1, 14], with the significant difference that we consider connectivity and flow when assigning pixels to a cluster. The general process is as follows: beginning at the voxel nearest the cluster center, we flow outward to adjacent voxels and compute the distance from each of these to the supervoxel center using Equation 4. If the distance is the smallest this voxel has seen, its label is set, and using the adjacency graph, we add its neighbors which are further from the center to our search queue for this label. We then proceed to the next supervoxel, so that each level outwards from the center is considered at the same time for all supervoxels. We proceed iteratively outwards until we have reached the edge of the search volume for each supervoxel (or have no more neighbors to check).

This amounts to a breadth-first search of the adjacency graph, where we check the same level for all supervoxels before we proceed down the graphs in depth. Importantly, we avoid edges to adjacent voxels which we have already checked this iteration. The search concludes for a super-

voxel when we have reached all the leaf nodes of its adjacency graph or none of the nodes searched in the current level were set to its label. This search procedure, illustrated in Figure 3, has two important advantages over existing methods:

1. Supervoxel labels cannot cross over object boundaries that are not actually touching in 3D space, since we only consider adjacent voxels, and
2. Supervoxel labels will tend to be continuous in 3D space, since labels flow outward from the center of each supervoxel, expanding in space at the same rate.

Once the search of all supervoxel adjacency graphs has concluded, we update the centers of each supervoxel cluster by taking the mean of all its constituents. This is done iteratively; either until the cluster centers stabilize, or for a fixed number of iterations. For this work we found that the supervoxels were stable within a few iterations, and so have simply used five iterations for all presented results.

4. Three Dimensional Voxel Segments

The proposed method works directly on voxelized point clouds, which has advantages over existing methods which operate in the projected image plane. The most important of these is the ability to segment clouds coming from many sensor observations - either using multiple cameras [3] or accumulated clouds from one [6]. Computationally, this is advantageous, as the speed of our method is dependent on the number of occupied voxels in the scene², and not the number of observed pixels. As observations will have significant overlap, this means that it is cheaper to segment the overall voxel cloud than the individual 2D observations. For instance, the scene in Figure 5 comes from 180 Kinect observations (640x480), and yet the final voxel cloud (with $R_{voxel} = 0.01m$) only contains 450k voxels.

Additionally, while VCCS will become more accurate as cloud information is filled in by additional observations, 2D methods must necessarily segment them independently and therefore cannot make use of the added information. Most

² We should note that while the initial voxelization of the cloud does take more time with a larger cloud, it remains insignificant overall

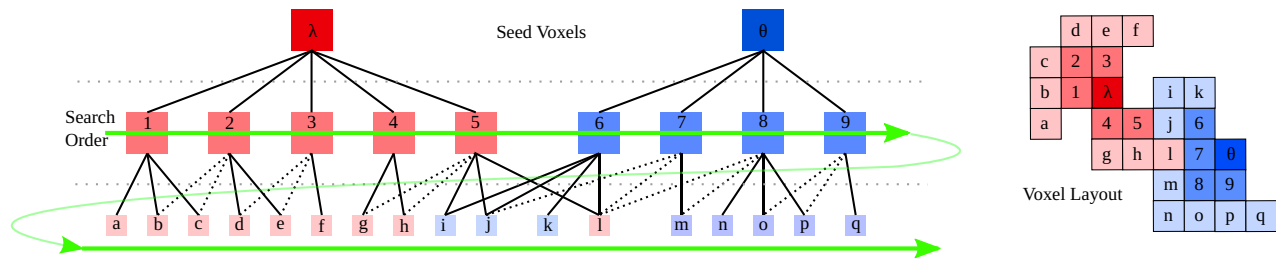


Figure 3. Search order for the flow constrained clustering algorithm (shown in 2D for clarity). Dotted edges in the adjacency graph are not searched, as the nodes have already been added to the search queue.

importantly, even with methods that use depth information, such as that of Weikersdorfer *et al.* [14], it is not clear how one would combine the multiple segmented 2d images, as superpixels from sequential observations will have no relation to each other and will have conflicting partitionings of space in the merged cloud.

5. Experimental Evaluation

In order to evaluate the quality of supervoxels generated by VCCS, we performed a quantitative comparison with three state-of-the-art superpixel methods using publicly available source code. We selected the two 2D techniques with the highest published performance from a recent review [1]: a graph based method, GCb10 [13]³, and a gradient ascent local clustering method, SLIC [1]⁴. Additionally, we selected another method which uses depth images, DASP[14]⁵. Examples of over-segmentations produced by the methods are given in Figure 6.

5.1. Dataset

For testing, we used the recently created NYU Depth Dataset V2 semantic segmentation dataset of Silberman *et al.* [12]⁶. This contains 1449 pairs of aligned RGB and depth images, with human annotated densely labeled ground truth. The images were captured in diverse cluttered indoor scenes, and present many difficulties for segmentation algorithms such as varied illumination and many small similarly colored objects. Examples of typical scenes are shown in Figure 6.

5.2. Returning to the Projected Plane

RGB+D sensors produce what is known as an organized point cloud- a cloud where every point corresponds to a pixel in the original RGB and depth images. When such a

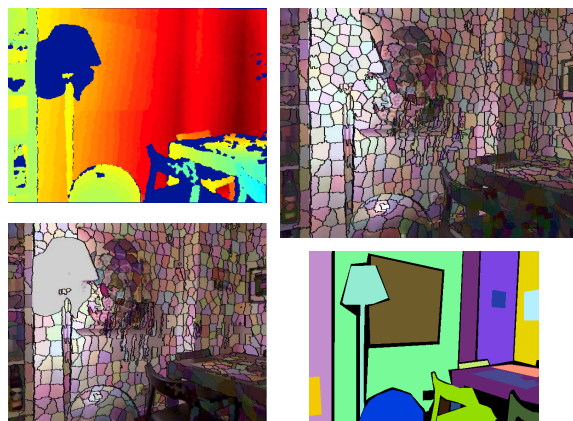


Figure 4. Example of hole-filling for images after returning from voxel-cloud to the projected image plane. Depth data, shown in the top left, has holes in it, shown as dark blue areas (here, due to the lamp interfering with the Kinect). The resulting supervoxels do not cover these holes as shown in the bottom left, since the cloud has no points in them. To generate a complete 2D segmentation, we fill these holes in using the SLIC algorithm, resulting in a complete segmentation, seen in the top right. The bottom right shows human annotated ground truth for the scene.

cloud is voxelized, it necessarily loses this correspondence, and becomes an unstructured cloud which no longer has any direct relationship back to the 2D projected plane. As such, in order to compare results with existing 2D methods we were forced to devise a scheme to apply supervoxel labels to the original image.

To do this, we take every point in the original organized cloud and search for the nearest voxel in the voxelized representation. Unfortunately, since there are blank areas in the original depth image due to such factors as reflective surfaces, noise, and limited sensor range, this leaves us with some blank areas in the output labeled images. To overcome this, we fill in any large unlabeled areas using the SLIC algorithm. This is not a significant drawback, as the purpose of the algorithm is to form supervoxels in 3D space, not superpixels in the projected plane, and this hole-filling is only needed for comparison purposes. Additionally, the hole fill-

³<http://www.csd.uwo.ca/~olga/Projects/superpixels.html>

⁴http://ivrg.epfl.ch/supplementary_material/RK_SLICSuperpixels/index.html

⁵<https://github.com/Danvil/dasp>

⁶http://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html

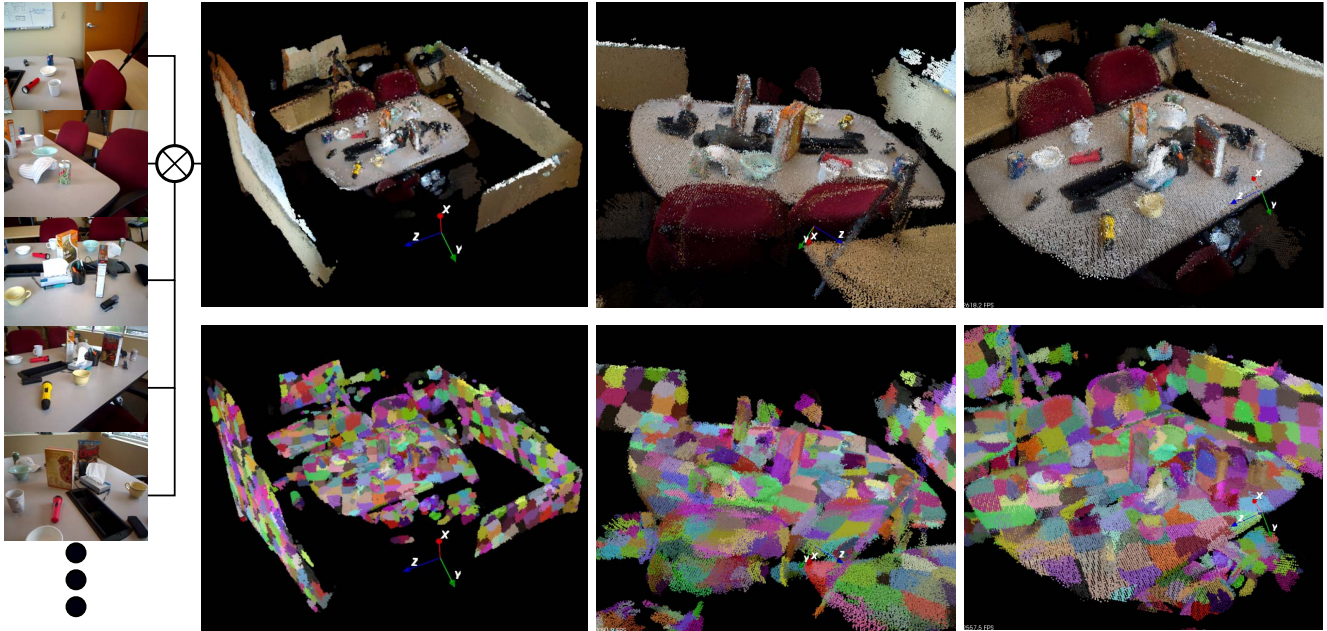


Figure 5. Over-segmentation of a cloud from the RGB-D scenes dataset[6]. The cloud is created by aligning 180 kinect frames, examples of which are seen on the left side. The resulting cloud has over 3 million points, which reduces to 450k points at $R_{voxel} = 0.01m$ and 100k points with $R_{voxel} = 0.02m$. Over-segmentation of these take 6 and 1.5 seconds, respectively (including voxelization).

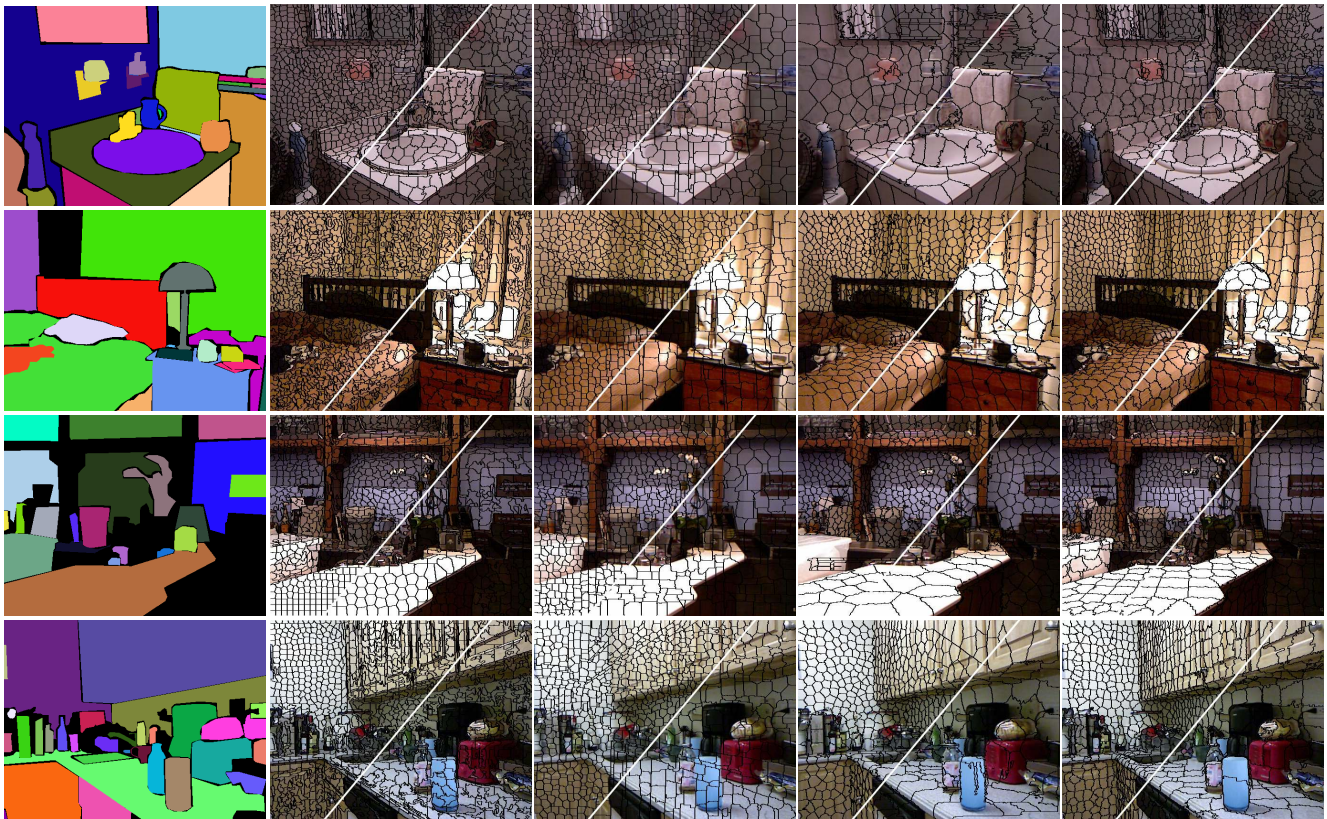


Figure 6. Examples of under-segmentation output. From left to right- ground truth annotation, SLIC, GCb10, DASP, and VCCS. Each is shown with two different superpixel densities.

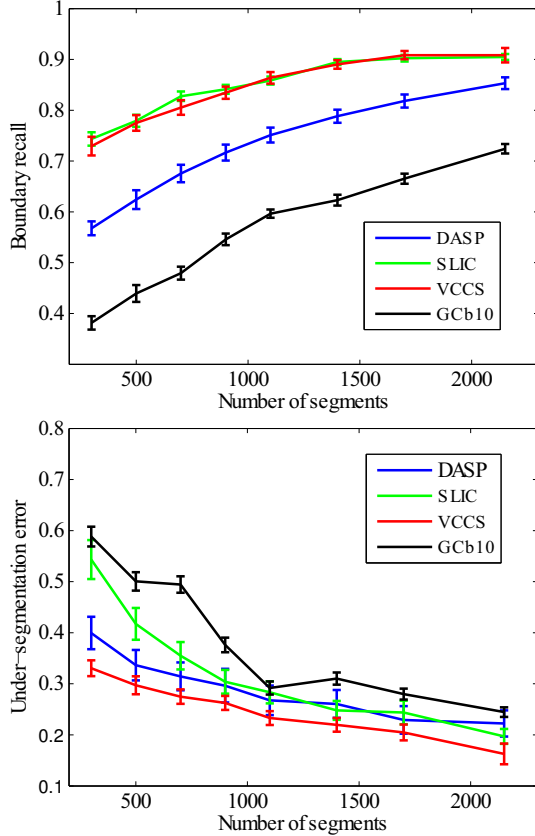


Figure 7. Boundary recall and under-segmentation error for SLIC, GCb10, DASP, and VCCS.

ing actually makes our results worse, since it does not consider depth, and therefore tends to bleed over some object boundaries that were correctly maintained in the supervoxel representation. An example of what the resulting segments look like before and after this procedure are shown in Figure 4.

5.3. Evaluation Metrics

The most important property for superpixels is the ability to adhere to, and not cross, object boundaries. To measure this quantitatively, we have used two standard metrics for boundary adherence- boundary recall and under-segmentation error[7, 13]. Boundary recall measures what fraction of the ground truth edges fall within at least two pixels of a superpixel boundary. High boundary recall indicates that the superpixels properly follow the edges of objects in the ground truth labeling. The results for boundary recall are given in Figure 7. As can be seen, VCCS and SLIC have the best boundary recall performance, giving similar results as the number of superpixels in the segmentation varies.

Under-segmentation error measures the amount of leak-

age across object boundaries. For a ground truth segmentation with regions g_1, \dots, g_M , and the set of superpixels from an over-segmentation, s_1, \dots, s_K , under-segmentation error is defined as

$$E_{useg} = \frac{1}{N} \left[\sum_{i=1}^M \left(\sum_{s_j | s_j \cap g_i} |s_j| \right) - N \right], \quad (5)$$

where $s_j | s_j \cap g_i$ is the set of superpixels required to cover a ground truth label g_i , and N is the number of labeled ground truth pixels. A lower value means that less superpixels violated ground truth borders by crossing over them. Figure 7 compares the four algorithms, giving under-segmentation error for increasing superpixel counts. VCCS outperforms existing methods for all superpixel densities.

5.4. Time Performance

As superpixels are used as a preprocessing step to reduce the complexity of segmentation, they should be computationally efficient so that they do not negatively impact overall performance. To quantify segmentation speed, we measured the time required for the methods on images of increasing size (for the 2D methods) and increasing number of voxels (for VCCS). All measurements were recorded on an Intel Core i7 3.2Ghz processor, and are shown in Figure 8. VCCS shows performance competitive with SLIC and DASP (the two fastest superpixel methods in the literature) for voxel clouds of sizes which are typical for Kinect data at $R_{voxel} = 0.008m$ (20-40k voxels). It should be noted that only VCCS takes advantage of multi-threading (for octree, kd-tree, and FPFH computation), as there are no publicly available multi-threaded implementations of the other algorithms.

6. Discussion and Conclusions

We have presented VCCS, a novel over-segmentation algorithm for point-clouds. In contrast to existing approaches, it works on a voxelized cloud, using spatial connectivity and geometric features to help superpixels conform better to object boundaries. Results demonstrated that VCCS produces over-segmentations which perform significantly better than the state-of-the-art in terms of under-segmentation error, and equal to the top performing method in boundary recall. This is fortunate, as we consider under-segmentation error to be the more important of the two measures, as boundary recall does not penalize for crossing ground truth boundaries- meaning that even with a high boundary recall score, superpixels might perform poorly in actual segmentation. We have also presented timing results which show that VCCS has run time comparable to the fastest existing methods, and is fast enough for use as a pre-processing step in online semantic segmentation applications such as robotics.

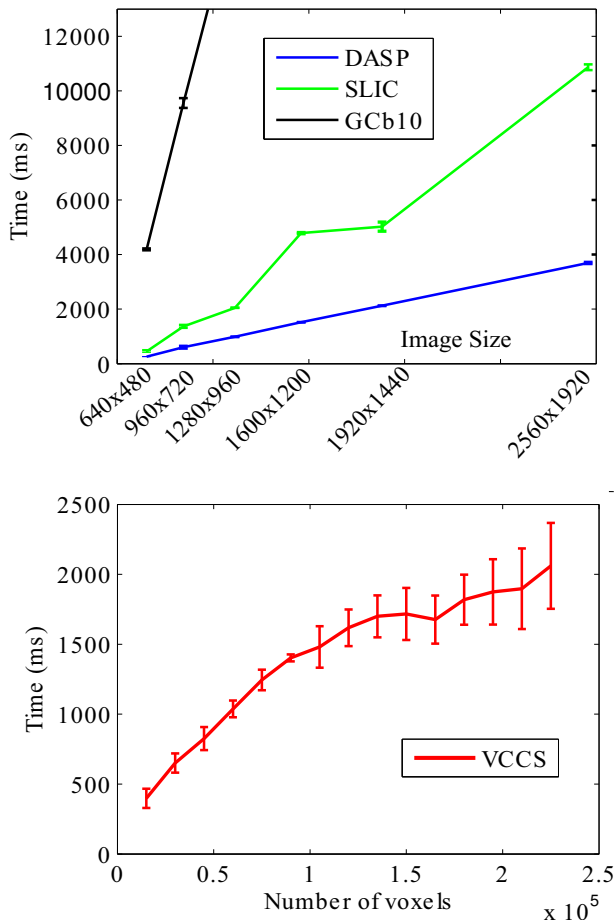


Figure 8. Speed of segmentation for increasing image size and number of voxels. Use of GCb10 rapidly becomes unfeasible for larger image sizes, and so we do not adjust the axes to show its run-time. The variation seen in VCCS run-time is due to dependence on other factors, such as R_{seed} and overall amount of connectivity in the adjacency graphs.

We have made the code publicly available as part of the popular Point Cloud Library, and intend for VCCS to become an important step in future graph-based 3D semantic segmentation methods.

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience and grant agreement no. 269959, Intellact.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art

superpixel methods. *IEEE Trans. Pattern Anal. Machine Intell.*, 34(11):2274–2282, nov. 2012. 2, 3, 4, 5

[2] A. Barla, F. Odone, and A. Verri. Histogram intersection kernel for image classification. In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, pages III–513–16 vol.2, sept. 2003. 4

[3] D. A. Butler, S. Izadi, O. Hilliges, D. Molyneux, S. Hodges, and D. Kim. Shake'n'sense: reducing interference for overlapping structured light depth cameras. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems, CHI '12*, pages 1933–1936, New York, USA, 2012. 4

[4] B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 670–677, oct. 2009. 1

[5] A. Hanbury. How do superpixels affect image segmentation? In *Progress in Pattern Recognition, Image Analysis and Applications*, Lecture Notes in Computer Science, pages 178–186. 2008. 1

[6] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824, may 2011. 4, 6

[7] A. Levinstein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE Trans. Pattern Anal. Machine Intell.*, 31(12):2290–2297, dec. 2009. 2, 7

[8] A. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones. Superpixel lattices. In *Computer Vision and Pattern Recognition, 2008 (CVPR). IEEE Conference on*, pages 1–8, june 2008. 2

[9] X. Ren and J. Malik. Learning a classification model for segmentation. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 10–17 vol.1, oct. 2003. 1

[10] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Robotics and Automation, 2009 (ICRA). IEEE International Conference on*, pages 3212–3217, may 2009. 3

[11] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *Robotics and Automation, 2011 (ICRA). IEEE International Conference on*, Shanghai, China, may 2011. 2

[12] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgb-d images. In *Computer Vision ECCV 2012*, volume 7576 of *Lecture Notes in Computer Science*, pages 746–760. 2012. 5

[13] O. Veksler, Y. Boykov, and P. Mehrani. Superpixels and supervoxels in an energy optimization framework. In *Computer Vision ECCV 2010*, volume 6315, pages 211–224. 2010. 2, 5, 7

[14] D. Weikersdorfer, D. Gossow, and M. Beetz. Depth-adaptive superpixels. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 2087–2090, 2012. 2, 4, 5

[15] Y. Yang, S. Hallman, D. Ramanan, and C. Fowlkes. Layered object detection for multi-class segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3113–3120, june 2010. 1

On Transferability and Contexts in Data-Driven Grasping

Jimmy A. Rytz, Lars-Peter Ellekilde, Dirk Kraft, Henrik G. Petersen, Norbert Krüger

December 10, 2013

Abstract

It has become a common practice to use simulation to generate large databases of good grasps for grasp planning in robotics research. However, the existence of a generic simulation context that enables the generation of high quality grasps that can be used in several different contexts such as bin-picking or picking objects from a table, has to our knowledge not yet been discussed in the literature.

In this paper, we investigate how well the quality of grasps simulated in a commonly used "generic" context transfers to a specific context, both, in simulation and in the real world.

We generate a large database of grasp hypothesis for several objects and grippers, which we then evaluate in different dynamic simulation contexts e.g., free floating (no gravity, no obstacles), standing on a table and lying on a table.

We present a comparison on the intersection of the grasp outcome space across the different contexts and quantitatively show that to generate reliable grasp databases, it is important to use context specific simulation.

We furthermore evaluate how well a state of the art grasp database transfers from two simulated contexts to a real world context of picking an object from a table and discuss how to evaluate transferability into non-deterministic real world contexts.

1 Introduction

For more than a decade, data-driven grasp planning approaches have been used in the research community and the main focus has been on how to online select good grasps from a grasp-database generated using heuristics or simulation [1–5]. Only little attention has been paid to the simulated context in which these databases were generated and how well the generated grasps actually perform in contexts which are different from the simulated context, e.g., grasps that have been generated in a free floating environment and then are applied to an object placed on a table.

Furthermore, object pose uncertainties due to model and sensor limitations may negatively influence the grasp execution. Pose uncertainties can be compensated for in the offline calculation of the grasps [6–8] by evaluating either the quality of neighboring grasps or the quality of the neighborhood grasp contacts, where a neighboring grasp can be simulated by applying a small displacement to the object before executing the target grasp in simulation. The basic idea is that a high average quality of grasps in the neighborhood of the target grasp reflects a high robustness toward uncertainties in the execution of the target grasp.

The above approach of calculating the robustness of a grasp target by evaluating the outcome of the neighborhood using small perturbations, assumes that the calculated neighborhood of grasps—whether done in simulation or with another heuristic—captures the same neighborhood as if the grasps were executed in the real world. This is problematic since the success of a grasp executed in the real world can be dependent on the environmental context, e.g., when the object is standing on a table, lying in water or leaning against a wall. This is illustrated in Fig. 1 where the same grasp is executed in two different environments, in one it succeeds and in the other it fails. It is obvious that changes in the environment can result in collisions between gripper and environment and therefore some grasps that are successful in one environment will fail

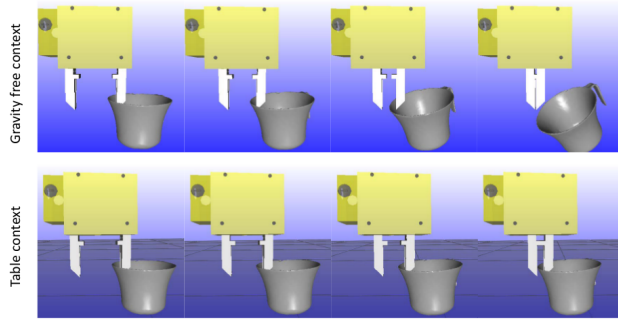


Figure 1: Two simulated grasp sequences of the same grasp but in two different contexts. The top sequence fails to grasp the cup in the gravity free environment. In the bottom sequence the cup is successfully grasped when placed on top of a table.

in another due to collision. This specific form of grasp failure can be effectively handled during the online selection of a grasp by testing gripper geometry for collision against a virtual model or the sensor information available. Thus, changes in grasp success space due to collisions are not of great importance in respect to predicting grasp success outcome.

However, the interaction forces between the object to be grasped and the environment may differ substantially in different contexts, which may result in different outcomes of attempting the same grasp. The reason is that pose uncertainties imply that the fingers will not be synchronously placed at the target contacts on the object, which will cause the object to move during the grasp. The movement will be constrained by the environment and hence different environments or contexts may influence the success of a specific grasp attempt.

In this paper, we investigate the transferability of grasp successes from both a simulated context to another simulated context as well as the transfer from a simulated context to a real world context. Our motivation is not only to understand how the simulated context affects the success of a given grasp, but also to understand how the success neighborhood of a grasp is affected when transferred to a new context. The influence on the neighborhood when transferring a grasp is important for several reasons:

1. *Grasp quality estimation*: the neighborhood is used to calculate the robustness quality of a grasp, which describes how much pose uncertainties affect the success likelihood of a grasp.
2. *Continuous grasp representation*: some grasp database representations such as grasp densities [9] represent not a single successful grasp, but a complete continuous grasp success space which include the grasp neighborhood successes.
3. *Uncertainties in execution*: when executing a grasp from a grasp database, then the actual executed grasp will likely be a grasp from the neighborhood of the selected grasp due to pose uncertainties.

We investigate the transferability of the grasp neighborhood by randomly generating grasp configurations that are simulated in different environments. Comparing the outcome of hundreds of thousands of grasps simulated in different environments enables us to get insight into the nature of the transferability of a grasp database, which allows us to qualitatively show the importance of using context specific simulation when computing grasp databases for data-driven grasp planners. In addition, we will present a quality measure for the context transferability of a grasp database and use it to compute the transferability for several chosen objects and contexts.

We furthermore provide real world results that show that context-aware generated grasp databases have a better grasp success transferability/predictability than context-unaware databases. We discuss and present

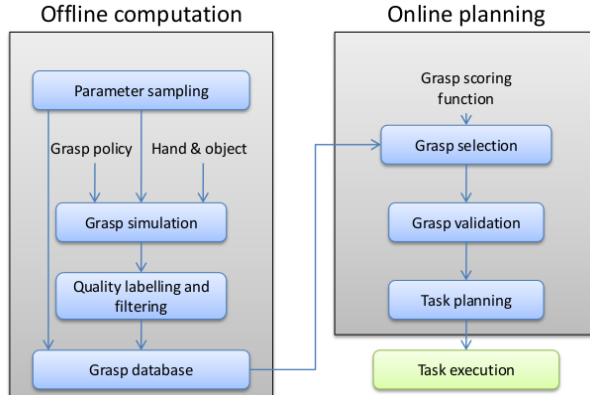


Figure 2: Typical data-driven grasp planning approach.

a quality measure for transferability to real world contexts which differ from simulated contexts by being non-deterministic and noisy.

In Section 2, we introduce related work within data-driven grasping and how it is commonly used in the community. Then in Section 3, we discuss transferability and how we can measure it, firstly when considering transfer between simulated contexts and secondly when considering transfer between simulated and real contexts. We introduce the setup, objects, grippers and control strategies in Section 4 which is followed by a description of how we compute the grasps in Section 5. In Section 6 the results are presented and a detailed analysis is followed in Section 7.

2 Related Work

State of the art grasp planning algorithms often rely on offline computed databases of feasible grasps generated using kinematic and/or dynamic simulation [1–5]. These grasp databases are used in online grasp/motion planning where the best grasp is chosen to be used in a specific context. The main steps in the generation of a grasp-database and its usage is outlined in Fig. 2. In the following, we first address the offline phase depicted on the left side of Fig. 2 followed by the online phase depicted to the right.

In the offline computation a sampling strategy is chosen to sample the parameter space of the system. This is defined by at least the pose of the gripper, the configuration of the gripper fingers (the grasp preshape) and the maximum joint torques. Additional parameters such as max finger velocity or the trajectory of the base of the gripper can also be used depending on the policy used to perform the actual grasping execution. The sampling can be selected arbitrarily, it can use grasp preshapes [10] and directly sample the base of the gripper in $SE(3)$ or use more biased sampling approaches that generate or plan samples based on the object geometry and/or gripper kinematics [11, 12].

The sampled parameters are used to initiate the grasp simulation which consist of the gripper and an object. When a single grasp has been generated, the simulator is started and a grasping policy is used to control the fingers into a grasp. This policy is typically simple and often dependent on which type of simulation is used e.g., kinematic or dynamic. In this work we use the dynamic rigid body grasp simulator in RobWork [13].

The simulation may be terminated due to a number of different criteria, which themselves involve the determination of whether the grasp was successful or not. In a kinematic simulation, analytic stability measures such as the maximum enclosing ball in the grasp wrench space [14], are computed based on contact points and used to describe if a grasp is stable. These measures also apply to dynamic simulations, but here the stability might also be determined by performing a lifting motion with the gripper and object as in [15]. If the object is not dropped after moving the gripper, then a stable grasp has been computed. Notice,

that in a dynamic simulation, the object may be lost during lifting due to forces from the environment, e.g., gravity or contacts with other objects.

After simulating sampled grasps the grasps are labeled and filtered before they are added to the grasp database, see bottom left in Fig. 2. Intuitively, only successful grasps of a certain quality should be added to the grasp database. However, for some applications it is important to include low quality grasps in order to maintain enough coverage of the object [16]. The grasp quality might be based on a grasp wrench space (GWS) analysis [14], on tactile based quality metrics [17] or any other form of quality measure [18] that can be computed offline. The main importance is to store grasps that will maximize the likelihood of success when used in an actual application.

For the simulated-to-simulated transfer experiments done in this paper, no quality other than success and failure is used. These are defined by trying to move the object after it is grasped. If the grasp can withstand the accelerations during movement then the grasp is successful.

In the online computation (the right block in Fig. 2), the context is changed. We have a robot with a limited reach, we have obstacles that limit the grasping possibilities, sensors that are imperfect and other constraints imposed by the task, that we wish the robot to perform. The main problem is then to select a grasp from the grasp database which is feasible in spite of these limitations and constraints. In general the online selection of a grasp from a grasp database should maximize the likelihood of success. In practice, grasps with highest offline computed qualities are typically used. A more advanced online grasp selection approach was presented in [2], where an online scoring function that favors grasp configurations that are furthest away from obstacles was used. The scoring function reduced unwanted collisions with other objects in the environment enabling safer grasping in cluttered environments.

Several learning approaches to grasping [19,20] use simulated grasp databases as training data. The main goal is typically to use learning to infer a feasible grasp when presented with some form of input, which could be point cloud data, images, or simply the geometry of an object.

In [4], a grasp database “the Columbia Grasp Database” is presented and it is used for grasp planning in [?]. The database is generated using dynamic simulation of grasps in an obstacle free environment and the grasp configurations were calculated by the authors’ eigen-based grasp planner.

In [1], the grasp database is generated based on a kinematic simulation of closing the gripper fingers around the object in an obstacle free environment. The authors do not comment on the context independence of the database but point out that a gripper with tactile feedback is necessary to execute grasps generated with their approach. In [2], the same approach is used to generate the grasp database, but here the authors do not mention the use of tactile feedback in their experimental setup.

The use of tactile sensors to online guide the grasp stability has been investigated in several papers [21–23] and shows great promise. However, the performance of tactile sensor hardware is far from ideal and typical issues are drift, low sensitivity range, low durability, detection of sliding, detection of measuring normal and measuring sheer forces. Reliable and general reactive grasping therefore still remains a challenge.

However, all above approaches rely on databases generated in a context independent simulation. To rely on the quality of those grasps it is necessary to have either a perfectly calibrated setup with no uncertainties or an online grasp execution that rely on tactile feedback to correct for uncertainties. Both of these properties in a setup are non-trivial and the latter is still not solved and is actively researched.

This paper is an extension on the conference paper [24] presented at the conference RAAD13. The original paper addressed the issues concerning the transfer of grasps that are computed in contexts that differ from the contexts they are used in. Simulated experiments were provided to document this transfer issue. In this paper we have extended the simulated experiments and provided results on transferring grasps from two different simulated contexts to a real world context. We furthermore show that context-aware generated grasp databases have a significantly better transferability than context-unaware databases.

3 On Quantifying Transferability

In this section we discuss how to quantify transferability both from simulated to simulated contexts and from simulated to real world contexts. Simulation to simulation quantification is easier due to ground truth

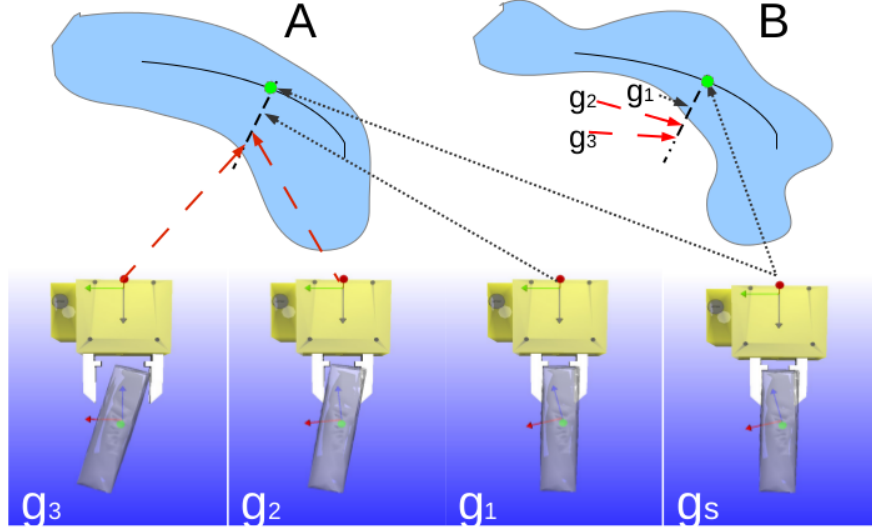


Figure 3: Grasp sequence demonstrating the difference between successful grasp configurations (the blue area) and stable grasp configurations (grasps on the black area). The two blue areas, depict the artificial grasp success space in two different environments A and B. The grasp sequence demonstrate the convergence of a successful grasp configuration into a stable grasp configuration in environment A. The success space of environment B suggests that grasps g_3 and g_2 should result in failure in environment B and are therefore not transferable from A to B.

knowledge of the virtual world and the deterministic way of simulation. In Section 3.1 we define a measure that exploits the deterministic behavior of simulation to evaluate the same grasps in two different contexts. Quantifying transferability from simulation to real world is more difficult due to the non-deterministic way of sensor and process noise in the real world. Hence, in Section 3.2 we describe a quantification method that is based on a continuous success evaluation instead of the binary success-failure outcome used in simulation.

3.1 Quantifying Transferability in Simulation

The transferability measure that we present in this section requires a set of grasps that have been executed in multiple contexts. The binary outcomes (success, failure) of these grasp experiments then define the transferability.

This is illustrated in Fig. 3 where the neighborhood (blue area) of successful grasp experiments in one context A is different from the neighborhood of successful grasp experiments from another context S. The green point illustrates the target grasp g_s which belongs to the set of stable grasps G_{stable} (black line/area) and the arrows illustrate three grasps (g_1, g_2, g_3) chosen from the neighborhood of g_s . In context B grasps (g_2, g_3) are no longer part of the neighborhood of successful grasps (indicated as red arrows) and therefore fail. We define the transfer of a grasp from one context to another to be successful in the case of g_1 , but unsuccessful in the case of (g_2, g_3).

If we think of transferability as a measure that determines how well we can predict outcomes of grasp experiments in one context, given the knowledge of the outcomes of the same experiments from another context, then we can borrow metrics from the machine learning literature and pose the measure of transferability as a classification problem. If we consider the execution of the same grasp in two different contexts then we may have four possible outcomes: (success, success), (failure, success), (success, failure) and (failure, failure). These are illustrated in a confusion matrix in Table 1 as true positive (TP), false negative (FN), false positive (FP) and true negative (TN). In the above example g_1 thus belongs to TP and g_2, g_3 belong to FP.

Table 1: A confusion matrix element that illustrate the outcome of transfer between two contexts s_{float} and s_1 . High values in TP and TN indicate high transferability of grasp results from one scenario to the other.

		Results in s_{float}	
		success	failure
Results in s_1	success	TP=74 (true positives), the number of grasp simulations that where successful in both scenarios	FN=113 (false negatives), the number of grasp simulations that failed in s_{float} but succeeded in s_1
	failure	FP=539 (false positives), the number of grasp simulations that succeeded in s_{float} but failed in s_1	TN=2507 (true negatives), the number of grasp simulations that failed in both s_{float} and s_1

The probability that a selected successful grasp from s_{float} will also be successful in s_1 is then $TP/(FP + TP)$.

We use the Matthews correlation coefficient (MCC) [25] as the transferability measure. It is based solely on the confusion matrix and produce a value between -1 and 1 , where 1 indicate perfect prediction, 0 indicate a random prediction and -1 indicate an inverse prediction of a binary classification. It can be calculated directly from the confusion matrix as shown in (1).

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (1)$$

3.2 Quantifying transferability in real experiments

As discussed in the previous section we think of transferability as a measure that determines how well we can predict the outcome of a grasp executed in one context, based on the knowledge of the outcome of a grasp executed in another typically simulated context. In the real world the outcome is a continuous success probability rather than a binary success/failure outcome. Hence, we need to compute a prediction of the success probability in the simulated context which, if good, would correlate with the real world success probability.

There exist several grasp quality metrics that describe the quality of a grasp as a continuous value. Some try to estimate success probability of a grasp by evaluating successes or quality over the neighborhood of grasps [6, 7, 16], others use heuristics typically based on grasp stability in relation to the grasp wrench space [14, 18] to determine grasp quality. Common for all quality metrics should be that a higher grasp quality correlates with a higher success probability. The Spearman rank correlation coefficient is capable of capturing this relationship between two variables. It essentially measure if the increase of the value on one variable does also reflect an increase (any increase) in the other variable. It is defined as the Pearson correlation coefficients over the ranked variables:

$$\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}} \quad (2)$$

where x_i, y_i are the ranked values and \bar{x}, \bar{y} are the mean of the rank variables. If there are no correlation then ρ yields 0 . If there is a increasing monotonic trend in the data then ρ yields 1 and when there is a

decreasing monotonic trend then ρ yields -1.

For quality metrics that directly estimate success probability, we would expect a linear relationship between simulated grasp quality and the real world success probability. It would therefore be more accurate to use the Pearson correlation coefficient in such scenarios since this captures the linear dependency between two variables. The Pearson correlation coefficient is defined in Equation 2 but with x_i, y_i as sampled values and \bar{x}, \bar{y} as the sampled means.

4 System Description

In this section we present the objects, grippers and the different simulated and real contexts used in this paper.

4.1 Objects and grippers

Three objects (see Fig. 4) were selected from the KIT Object-Models Web Database¹. The objects are common household objects which are sufficiently different to provide interesting comparisons.



Figure 4: Object from the KIT object database. From left to right: corny object, cup-object and tomato object.

The grippers used to grasp these objects are the Schunk parallel gripper (PG 70) and the Schunk Dexterous Hand (SDH-2), which are both shown in Fig. 5. The PG 70 gripper has two fingers coupled into one Degree Of Freedom (DOF), that is, one DOF moves both fingers. The fingers can move up to 6.8 cm apart and the contact surface is approximately 2×3 cm and covered by rubber.

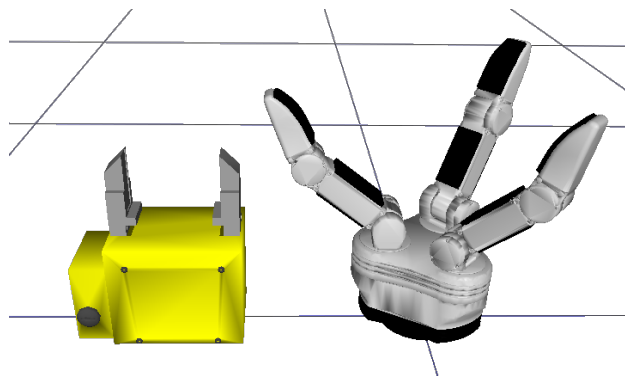


Figure 5: PG 70 gripper (left) and the SDH-2 (right).

The SDH-2 is a 3-fingered dexterous hand with two DOF per finger and one coupled DOF to control the base rotation of two of the three fingers. The SDH-2 has six contact surfaces covered with rubber, each of

¹<http://www.iam.ira.uka.de/ObjectModels>

them measuring approximately 2×3 cm. However, for precision grasps only the three contact surfaces on the distal joints are normally used.

For the PG 70, we choose only one preshape with the maximum distance of 6.8 cm between its jaws. Four preshapes were chosen for the SDH-2 which are shown in Fig. 6. These different preshapes enables different grasping options, and as such are important when characterizing the grasp affordances of the gripper.

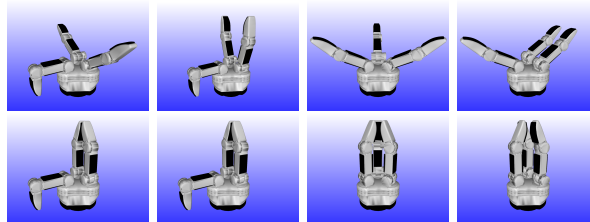


Figure 6: Four preshapes used with the SDH-2 dexterous hand. From left to right: c_{par} , $c_{parsmall}$, c_{ball} , c_{cyl} . The top row is the initial preshape before grasping, the bottom row is the target configuration that the hand will move to during grasping.

We use preshapes because:

- Using preshapes is a simple and direct way of providing multiple ways of grasping an object. The grasping process reduces to the sequence: *(open hand, move towards object, close hand)*, which is already supported by the software of the SDH-2.
- Preshapes do not require additional parametrization. This property is important when sampling random grasps since the dimensionality of the search space is not extended by any gripper parameters such as the individual joint configurations.

4.2 Contexts

The context describe the environment and task in which grasps are being executed. We have evaluated grasps in four different contexts where the first three are simulated environments and the last is a real environment:

- *float* - the object is placed in a simulated gravity free environment with no obstacles.
- *table* - a simulated environment with gravity is used and the object is placed on a planar surface in one of its canonical poses.
- *full* - the object is placed randomly on a table in an environment that closely models the real environment and task.
- *real* - the object is placed randomly on a table in a real world environment and picked up by a robot with a dexterous gripper.

5 Computation of Grasp Affordances

We compute complete grasp affordances by evaluating randomly sampled gripper poses in the neighborhood of an object. Each of these sampled poses, combined with a preshape of the gripper, represents a grasp hypothesis, which can be evaluated in simulation. The outcome of the simulation may be one of $\{success, failure\}$, where *success* represents a successful grasp of the object (indicated by the fact that the gripper is still in contact with the object after grasping it) and *failure* represents a grasp where the gripper has no contact with the object after trying to grasp or lift it; *collision* represents a grasp, where the gripper is in collision with the object or the environment in the initial state of the simulation.

The evaluation of a grasp hypothesis is performed using a dynamics grasp simulator from RobWork [13]. The main simulation process of a single grasp is:

1. Set the initial scene configuration, e.g., gravity, friction, poses of obstacles and objects.
2. Place the gripper in a sampled pose relative to the object and set the gripper configuration to one of the preshapes.
3. Test if the gripper collides with object or environment.
4. Start the simulation and set the target gripper configuration (preshape dependent) of the gripper controller.
5. If a grasp is obtained, lift the object and compute the success criteria.

We shall now discuss these steps in more detail.

5.1 Initial scene configuration

The initial scene configuration for generating the grasp-databases use a free floating environment, in which no gravity or obstacles are present. In such an environment the object can freely slide into or out of a grasp.

To investigate the transfer of grasp affordances of an object from the free floating environment we repeat the grasp simulation, but with the object placed on a table. Two canonical poses of each object are used to create two different table environments per object.

5.2 Object specific sampling strategy

The sampling of the gripper configuration and the choice of sampling strategy necessarily influence the resulting set of grasp affordances and the overall success probability. Typically, the primary goal of grasp planning is to maximize the grasp success probability by exploiting knowledge of gripper, object and environment. This tends to generate grasp databases that only represent a small subset of the complete set of successful grasps.

In this work, we need an approximation of the complete set of grasp affordances that are possible on a specific object. Thus, an unbiased sampling strategy that explores $SE(3)$ fully is preferred. Uniform random sampling in $SE(3)$ would therefore be ideal but it is also impractical because of the large number of simulations necessary to cover $SE(3)$. Instead a sampling strategy that is biased toward the object geometry is used. This effectively reduces the number of required simulations without reducing the success space too much.

Beside geometric models of the object the sampling also requires the approach vector of the gripper to be placed in the same direction as the positive z axis of the gripper Tool Center Point (TCP) frame. The sampling effectively encapsulates the idea that the gripper needs to point toward some part of the object geometry before it is able to successfully grasp the object.

First a random point \mathbf{p} is selected from a uniform distribution over the surface of the object. Then an orientation \mathbf{R} is selected from a uniform distribution in $SO(3)$ and used to define the temporary target pose (\mathbf{p}, \mathbf{R}) . The pose is then translated along the z axis by a randomly generated value d in the interval $[-0.04 \text{ m}; 0.04 \text{ m}]$. 4 cm was determined from the size of the fingers of the biggest gripper. If the tool center point was placed more than 4 cm away from the triangle surface then the chance of generating a valid grasp configuration is very low. Hence, a larger value would only increase the computation time without increasing the number of generated grasps. The final pose is therefore:

$$\mathbf{T}_{\text{pose}} = (\mathbf{p} - (\mathbf{R} \cdot [0, 0, 1]^T) \cdot d, \mathbf{R}) \tag{3}$$

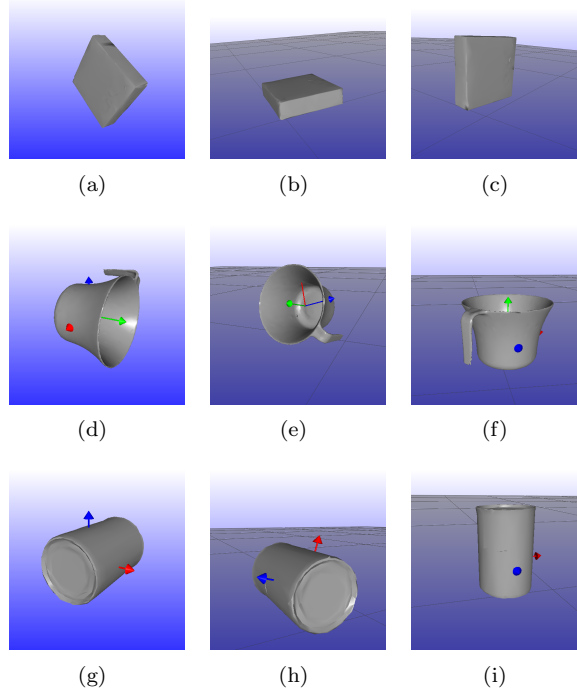


Figure 7: The floating pose is illustrated on the left, followed by two canonical poses on the table (side and upright). Top row: Corny object. Middle row: Cup object. Bottom row: Tomato object.

5.3 Collision filtering and labeling

For each sampled grasp configuration a collision detection between gripper and object/environment is performed before doing the actual simulation. If a grasp is colliding with the object then it will not be added to any database. If a grasp is colliding with the environment then it will be added and labeled *Colliding*.

In a grasp-planning context, all colliding grasps will be left out of the database, but for this work we need to evaluate the transfer of success between grasp-databases and the colliding grasps will be needed to create meaningful statistics. That is any successful or failed grasp from database A that are labeled as colliding in database B should be left out when computing the transfer statistics between A and B.

5.4 Grasp simulation

The final step before adding a non-colliding grasp to the database is the grasp simulation. Both pose and configuration of the gripper have been sampled and no collisions with environment or object was detected. The simulation is initialized with the scene configuration and the sampled parameters. When the simulation starts, a penalty based grasp controller guides the fingers toward a closing configuration. Closing configurations for the four preshapes of the SDH-2 are shown in Fig. 6. A grasp is successful if the object gets caught between the fingers and if it stays there under a 10 cm movement with a maximum acceleration of $3 m/s^2$ in the positive z -axis of the world coordinate frame (gravity works in the direction of the negative z -axis). This is termed a lifting operation, which only makes sense in the table environments where the objects needs to be lifted free of the table.

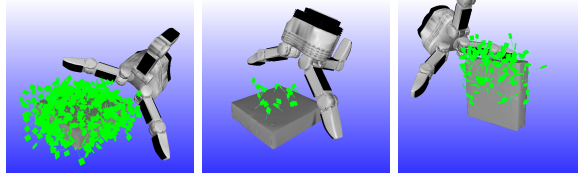


Figure 8: The successes of grasping the Corny object in the same 5000 grasp simulations but performed in different contexts. From left to right: free-float, on table (side pose) and on table (upright pose).

6 Experimental Results

In this section, we first provide experimental results on the transfer between two simulated contexts in Section 6.1 and secondly, we provide results on the transfer from simulated contexts to a real world context in Section 6.2.

6.1 Simulation Results

We provide simulation results in terms of complete sets of grasp affordances for three objects (see Fig. 4) using the PG 70 parallel gripper and SDH-2 hand in two different contexts, namely the rather artificial free-floating context shown in left column of Fig. 7 and the more application-oriented context, where the object is placed on a table (see the middle and right column of Fig. 7). The table environments have a gravity of $9.81 m/s^2$ and the viscous Coulomb friction between table and object is set to $0.3 N/(m/s)$. All simulated environments for all objects are illustrated in Fig. 7.

Fig. 8 illustrates the successful outcomes of the same 5000 grasp hypotheses in the three different contexts of the corny object grasped with the SDH-2 using the preshape c_{par} . It is clear that the added table constraint significantly reduces the number of successful grasps. However, it is not clear if successes from the constrained environments (center and right image) will also be successes in the floating environment. In the following, we use the confusion matrices introduced in Section 3.1 to evaluate how well successes and failures in the floating simulations transfer to successes and failures in the table environments, and vice versa.

Table 2: Success percentages of the simulated outcomes. In each major column, the left sub-column shows the percentages of success if collisions are not included, and the right column shows the success percentages if collisions are included.

		SDH-2								PG 70	
		c_{par}		$c_{parsmall}$		c_{ball}		c_{cyl}		c_0	
O_{corny}	s_{float}	43.1%	25.0%	45.5%	5.2%	70.3%	38.0%	52.9%	28.0%	12.3%	0.7%
	s_{side}	5.8%	0.4%	0.0%	0.0%	5.8%	0.2%	5.3%	0.3%	0.1%	0.0%
	$s_{upright}$	42.1%	7.2%	41.5%	1.9%	70.9%	9.3%	51.9%	8.9%	14.4%	0.4%
O_{cup}	s_{float}	54.5%	42.6%	47.7%	6.0%	79.8%	61.1%	61.7%	45.7%	41.5%	3.7%
	s_{side}	47.0%	5.8%	48.4%	1.7%	72.5%	5.4%	47.3%	6.0%	44.0%	1.6%
	$s_{upright}$	45.0%	4.7%	65.4%	2.3%	81.9%	4.9%	47.7%	4.9%	70.4%	3.5%
O_{tomato}	s_{float}	48.5%	24.9%	30.4%	5.5%	84.9%	72.0%	72.3%	59.9%	3.8%	0.3%
	s_{side}	22.7%	2.3%	10.9%	0.3%	34.4%	1.9%	32.8%	3.4%	2.4%	0.1%
	$s_{upright}$	43.4%	6.6%	23.1%	1.4%	75.3%	7.5%	63.1%	10.4%	17.2%	0.6%

Multiple datasets were generated for each gripper. For the SDH-2 the datasets are characterized by a triple (o_i, s_j, c_k) , where o_i is the object, s_j is the specific scene (free floating or on table with different poses), and c_k is the grasp strategy, which includes the number of fingers and the preshape used. The parallel gripper is simpler, and only one grasp strategy is used. Hence we describe datasets generated with the PG 70 by a pair (o_i, s_j) .

Table 3: Confusion matrices of successes and failures from floating environment and the specific table environment ($s_{side}, s_{upright}$). See Table 1 for a detailed explanation of a single cell.

		SDH-2								PG 70	
		c_{par}		$c_{parsmall}$		c_{ball}		c_{cyl}		c_0	
O_{corny}	s_{side}	159	221	0	0	74	113	97	222	0	1
	$s_{upright}$	933	5298	1	1495	539	2507	721	4991	0	1499
O_{cup}	s_{side}	6271	861	1777	100	8586	761	6296	830	327	46
	$s_{upright}$	1660	8183	284	2379	1127	2712	1587	5017	34	2182
O_{tomato}	s_{side}	4876	916	1548	126	1459	407	893	314	1568	81
	$s_{upright}$	1903	4621	241	1546	303	406	474	871	175	1924
O_{corny}	s_{side}	3483	1219	2093	207	1574	657	917	540	3119	396
	$s_{upright}$	1683	4059	155	1061	195	297	449	1146	78	1399
O_{cup}	s_{side}	1019	228	246	31	445	112	1063	278	30	35
	$s_{upright}$	1310	2923	214	2045	570	494	1212	1531	43	2568
O_{tomato}	s_{side}	1518	119	547	82	2167	91	2019	112	113	434
	$s_{upright}$	538	1591	219	1874	261	477	607	602	53	2584

Table 4: Quality estimates based on the MCC correlation coefficient of the confusion matrices in Table 3.

		SDH-2				PG 70
		c_{par}	$c_{parsmall}$	c_{ball}	c_{cyl}	c_0
O_{corny}	$s_{side, MCC}$	0.17	-	0.13	0.12	-
	$s_{upright, MCC}$	0.70	0.83	0.64	0.65	0.87
O_{cup}	$s_{side, MCC}$	0.55	0.79	0.34	0.39	0.86
	$s_{upright, MCC}$	0.45	0.78	0.25	0.35	0.79
O_{tomato}	$s_{side, MCC}$	0.43	0.64	0.26	0.33	0.42
	$s_{upright, MCC}$	0.67	0.72	0.67	0.52	0.32

For each floating environment experiment (o_i, s_{float}, c_k) 100.000 grasp simulations² were generated using the sampling approach presented in Section 5.2. The overall success probabilities of the simulations are available in Table 2. These indicate the size of the success space of the grasp datasets in the individual contexts. Two success probabilities are given. The first shows the percentage of grasp successes from all grasps that were not initially in collision, the second shows the percentage of grasp successes from all grasps including the colliding ones. The latter is only interesting for evaluating how many grasps out of the database can actually be applied. It cannot be used to state anything on transferability since collisions are easily predicted online using collision detection between gripper geometry and a virtual model of the environment or sensor data.

In the same table the success probabilities of the simulations of the table environments ($s_{side}, s_{upright}$) are also shown. These simulations use the same grasp hypotheses as in the floating environment, but performed in the specific contexts (e.g., the object was placed upright on the table).

That the exact same grasps have been executed in the different contexts enables us to calculate the transferability measure presented in Section 3.1. Table 3 show the confusion matrices from which the transferability is calculated and the transferability quality of each confusion matrix is presented in Table 4.

6.2 Real world experiments

The real world experiments were performed using the *Rump* object illustrated in Fig. 9. The *Rump* object weighs approximately 600 grams and was designed as a typical industrial object with a size to match the

²For the tomato object the actual number of samples is slightly lower.

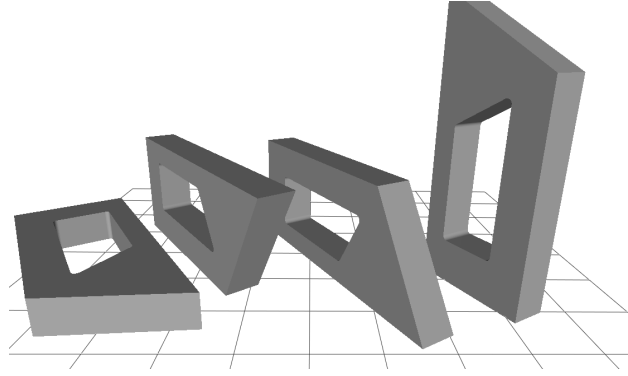


Figure 9: The Rump object used in the real world experiments.

SDH gripper.

The grasping setup is illustrated in Fig. 10 where the SDH gripper is mounted on a universal robot arm UR-6-85-5-A that has a reach of approximately 1 m and a payload of 5 kg. In-between gripper and robot end-effector a Schunk 6D force torque sensor is mounted that is used for collision detection.

A single grasp database consisting of 434 grasps of the rump object using the SDH was used for the real world experiments. These grasps were generated using a state of the art filtering method [16] which not only keeps high quality grasps but also keeps grasps that increase the number of directions that the object can be reached from. For each grasp in the database, three different simulated quality metrics were computed. One used the radius of the maximum enclosing ball in the grasp wrench space [14]. The other two were based on directly computing the grasp success probability in two different contexts, namely a floating context and the full context.

The grasp database was used in a repetitive process that first located the Rump object on the table using a CAD based object localization algorithm. The localization used RGB-D information captured from a Microsoft Kinect camera which was converted into a point cloud and thereafter all points below and all points above 50cm the largest plane was filtered away. The remaining points was clustered and an Iterative Closest Point (ICP) algorithm was used to fit the CAD model of the Rump to each cluster. The best fit was assumed to be the correct match.

After detecting the pose of the object, a suitable grasp from the database was chosen to be executed. If grasping was successful, then a new random drop pose above the table was calculated and the robot would execute motions to reach the drop pose. Finally, the grasp of the object would be released. If the grasp was unsuccessful then another grasp would be selected for execution. If all grasp attempts for a specific object pose would fail, a user would intervene. It should be noted that only grasps that were kinematically feasible would be selected. Meaning that for a grasp hypothesis to be selected it should have valid inverse kinematics (reachable), it should not collide with object or environment and there should exist a collision free path to reach the grasping configuration and to lift the object up from the table.

For each grasp execution attempt, the outcome (success/failed) was saved and used to later compute a success probability of each grasp. These success probabilities were then compared and correlated with the three different quality metrics.

A total of 2100 grasp experiments were performed in the real setup using 164 grasps out of the 434 grasps in the database. From these 164 grasps only 44 grasps have been used in more than 13 experiments. The grasp used most often had been used 106 times. Our correlations are based on these 44 grasps and the data is visualized in the graphs in Fig. 11.

The transferability measure based on the Spearman and Pearson correlation coefficients have also been computed for these two datasets. These are available in the Table 5. The table shows that the full context generated success probabilities correlate significantly better than the float based method, measured with both Pearson and Spearman correlation coefficients.



Figure 10: The UR-6-85-5-A robot arm mounted with a Schunk Dexterous Hand (SDH). The object in the lower right corner is the Rump object.

Table 5: Correlation between simulated grasp quality in two different contexts and the actual success probability achieved in real world use of the grasps.

	float context	full context
Pearson	0.30	0.67
Spearman	0.18	0.41

7 Analysis of results

In this section, we first discuss the results on transferring grasp outcomes from one simulated context to another in Section 7.1 thereafter we discuss the transfer of the continuous grasp outcome from simulated context to real world experiments in Section 7.2.

7.1 Simulation to simulation results

The MCC metric is used directly on the confusion matrices of Table 3 (see Table 1 for explanation). Note that in Table 3 collisions with the table are filtered away. The results are shown in the rows of Table 4.

The results show that the quality of the prediction depends highly on both the gripper and the object/context that is used. In about half the contexts the MCC is around 0.80, indicating a fairly good transfer. In the other half the quality ranges from 0.1 to 0.5, indicating a positive correlation, but a much weaker transfer.

It is important to mention that in certain contexts quite a number of grasps that are not successful in the free-floating environment are successful in the constrained environment. This implies that in certain scenarios, context-specific control strategies need to be taken into account. For example, the movement of a flat object on a table when touched by one of the fingers might be utilized in the grasping process (see Fig. 1). These context-specific constraints cannot be accounted for in a free-floating scenario and need to be learned in the specific context.

In summary, the results show that grasp simulation in free-floating scenarios gives often a fairly reliable but also frequently a rather unreliable prediction of grasp success in more constrained scenarios. In certain

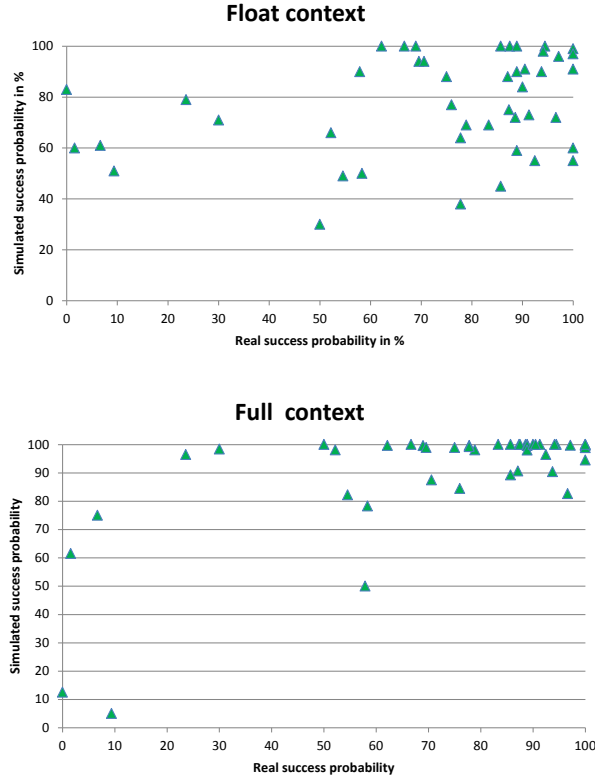


Figure 11: (top) Correlation between float based robustness quality and success probability in the real world. (bottom) Correlation between success probability achieved in a simulation of the full context and the success probability in the real world.

contexts $(C_{par}, O_{tomato,side})$, $(C_{par}, O_{Corny,side})$ and $(C_{ball}, O_{Corny,side})$ the prediction is so bad that the chances of choosing a successful grasp from s_{float} which will also be successful in s_1 is less than 20%. This is calculated by the relationship between TP and FP values of the confusion matrices in Table 3. Hence, grasp databases should ideally be generated in a context-specific fashion.

7.2 Simulation to real world results

The correlation results in Table 5 show that the "full context" simulation has the best transferability. This was expected since the "full context" should model the real setup to a larger detail compared to the "float context".

Fig. 11(bottom) provides further insight into the transferability for the "full context" where there is a tendency that samples are located in the upper left area of the correlation space, which suggest a systematic overestimation of success probability. The result of the float context is shown in Fig. 11, where there is a tendency in regard to over estimation for grasps with a very low success probability.

8 Conclusion

Data-driven grasp planning has become increasingly popular and some grasp databases are available for public download [3,26]. When using a grasp target g_s from such a database, in systems with uncertainties,

then it is most probable that the executed grasp will lie in the neighborhood of g_s . Hence, it is important to incorporate the transferability of the neighborhood of a grasp when using grasp database.

Moreover, in this paper we showed that it is quite likely that grasps performed on objects in different contexts have different outcomes. Hence, one should take care when using a grasp database in a context other than the context in which it was generated.

We specifically investigated the transfer between an unconstrained, free-floating environment and more constrained environments, which is important for applying learned grasp knowledge in novel contexts. For several different simulated contexts, we showed that the grasp success likelihood depends highly on the context that the object is embedded in and we used the Mathews correlation metric to evaluate how good this transfer is.

We also discussed how to evaluate transferability between simulated and real contexts and computed the transferability between two different simulated contexts to a real world context using a state of the art grasp database. These results should motivate the use of the detailed context in the simulation of grasps.

Further investigations on grasp transfer could include more real world experiments and a deeper analysis on the discrepancies between simulation and real world. Also evaluating the transferability of state of the art grasp quality metrics for different contexts, grippers and objects could provide further insights in how to choose a suitable grasp quality metric.

9 Acknowledgment

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience.

References

- [1] A. Morales, T. Asfour, P. Azad, S. Knoop, and R. Dillmann. Integrated grasp planning and visual object localization for a humanoid robot with five-fingered hands. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5663–5668, oct. 2006.
- [2] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner. Grasp planning in complex scenes. In *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*, pages 42–48, 29 2007-dec. 1 2007.
- [3] C. Goldfeder, M. Ciocarlie, J. Peretzman, Hao Dang, and P.K. Allen. Data-driven grasping with partial sensor data. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 1278–1283, oct. 2009.
- [4] Corey Goldfeder, Matei T. Ciocarlie, Hao Dang, and Peter K. Allen. The columbia grasp database. In *ICRA*, pages 1710–1716. IEEE, 2009.
- [5] Corey Goldfeder and PeterK. Allen. Data-driven grasping. *Autonomous Robots*, 31(1):1–20, 2011.
- [6] Junggon Kim, K. Iwamoto, J.J. Kuffner, Y. Ota, and N.S. Pollard. Physically-based grasp quality evaluation under uncertainty. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3258–3263, 2012.
- [7] Jonathan Weisz and Peter K. Allen. Pose error robust grasping from contact wrench space metrics. In *ICRA*, pages 557–562. IEEE, 2012.
- [8] H. Kruger, E. Rimon, and A.F. van der Stappen. Local force closure. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4176–4182, may 2012.

- [9] R. Detry, D. Kraft, O. Kroemer, L. Bodenhagen, J. Peters, N. Krger, and J. Piater. Learning grasp affordance densities. *Paladyn*, 2(1):1–17, June 2011.
- [10] David Wren and Robert B. Fisher. Planning dextrous hand precision grasps from range data, using preshaping and finger trajectories. In *Proc. Symp. Intelligent Robotic Systems (SIRS95)*, 1995.
- [11] K. Huebner, S. Ruthotto, and D. Kragic. Minimum volume bounding box decomposition for shape approximation in robot grasping. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1628–1633, 2008.
- [12] Andrew T. Miller, Steffen Knoop, Henrik I. Christensen, and Peter K. Allen. Automatic grasp planning using shape primitives. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 2, page 18241829, 2003.
- [13] J.A. Jørgensen, L.P. Ellekilde, and H.G. Petersen. Robworksim - an open simulator for sensor based grasping. In *Proceedings of Joint 41st International Symposium on Robotics (ISR 2010) and the 6th German Conference on Robotics*, Munich, 2010.
- [14] C. Ferrari and J. Canny. Planning optimal grasps. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 2290 –2295 vol.3, may 1992.
- [15] Gert Kootstra, Mila Popovi, JimmyAlison Jrgensen, Danica Kragic, HenrikGordon Petersen, and Norbert Krger. VisGraB: a benchmark for vision-based grasping. *Paladyn*, 3(2):54–62, 2012.
- [16] Dirk Kraft, Lars-Peter Ellekilde, and Jimmy Alison Jørgensen. Automatic grasp generation and improvement for industrial bin-picking. In Florian Röhrbein, Germano Veiga, and Ciro Natale, editors, *Gearing up and accelerating crossfertilization between academic and industrial robotics research in Europe:*, volume 94 of *Springer Tracts in Advanced Robotics*, pages 155–176. Springer International Publishing, 2014.
- [17] J.A. Jørgensen and H.G. Petersen. Grasp Synthesis for Dextrous Hands Optimised for Tactile Manipulation. In *Proceedings for the joint conference of ISR 2010 (41st International Symposium on Robotics) und ROBOTIK 2010 (6th German Conference on Robotics)*. VDE-Verlag, June 2010.
- [18] Raúl Suárez, Máximo Roa, and Jordi Cornella. Grasp quality measures. Technical report, Technical University of Catalonia, 2006.
- [19] R. Pelossof, A. Miller, P. Allen, and T. Jebara. An svm learning approach to robotic grasping. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 4, pages 3512 – 3518 Vol.4, 26-may 1, 2004.
- [20] Noel Curtis, Jing Xiao, and Senior Member. Efficient and effective grasping of novel objects through learning and adapting a knowledge base. In *in ICRA*, 2008.
- [21] J.A. Jørgensen and H. Petersen. Usage of simulations to plan stable grasping of unknown objects with a 3-fingered schunk hand. In *Workshop on robot simulators (IROS08)*, 2008.
- [22] A. Morales, M. Prats, P. Sanz, and A. P. Pobil. An experiment in the use of manipulation primitives and tactile perception for reactive grasping. In *Robotics: Science and Systems, Workshop on Robot Manipulation: Sensing and Adapting to the Real World, Atlanta, USA*, 2007.
- [23] K. Hsiao, S. Chitta, M. Ciocarlie, and E. G. Jones. Contact-reactive grasping of objects with partial shape information. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 10, page 2010, 2010.
- [24] J. A. Jrgensen, L. P. Ellekilde, D. Kraft, H. G. Petersen, and N. Kruger. On transferability of grasp-affordances in data-driven grasping. In *Proceedings of the RAAD 2013 22nd International Workshop on Robotics in Alpe-Adria-Danube Region*, 2013.

- [25] B.W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442 – 451, 1975.
- [26] Alexander Kasper, Zhixing Xue, and Rüdiger Dillmann. The KIT object models database: An object model database for object recognition, localization and manipulation in service robotics. *The International Journal of Robotics Research*, 31(8):927–934, July 2012.

Physical Interaction for Segmentation of Unknown Textured and Non-textured Rigid Objects

David Schiebener, Aleš Ude and Tamim Asfour

Abstract—We present an approach for autonomous interactive object segmentation by a humanoid robot. The visual segmentation of unknown objects in a complex scene is an important prerequisite for e.g. object learning or grasping, but extremely difficult to achieve through passive observation only. Our approach uses the manipulative capabilities of humanoid robots to induce motion on the object and thus integrates the robots manipulation and sensing capabilities to segment previously unknown objects. We show that this is possible without any human guidance or pre-programmed knowledge, and that the resulting motion allows for reliable and complete segmentation of new objects in an unknown and cluttered environment.

We extend our previous work, which was restricted to textured objects, by devising new methods for the generation of object hypotheses and the estimation of their motion after being pushed by the robot. These methods are mainly based on the analysis of motion of color annotated 3D points obtained from stereo vision, and allow the segmentation of textured as well as non-textured rigid objects. In order to evaluate the quality of the obtained segmentations, they are used to train a simple object recognizer. The approach has been implemented and tested on the humanoid robot ARMAR-III, and the experimental results confirm its applicability on a wide variety of objects even in highly cluttered scenes.

I. INTRODUCTION AND RELATED WORK

The ability of a humanoid robot to adapt to situations that it has not explicitly been programmed for is crucial for its usefulness in future assistive tasks in human-centered environments. Many of these not-yet-experienced situations for a robot will arise due to the appearance of objects that it has not encountered before but now needs to deal with. In such situations, the robot needs to autonomously make itself familiar with these new objects. The first two crucial steps in this process, whatever outcome may be expected, are to locate and segment the new objects. Once they are segmented, a visual descriptor can be learned that allows later recognition, and essential information for grasping and manipulation is provided.

The focus of this work is to present our approach for the autonomous, interactive discovery and segmentation of textured and non-textured unknown objects in a cluttered environment by a humanoid robot. To demonstrate its usefulness, we use the obtained segmentations to learn visual descriptors of the new objects and show that they allow reliable recognition.

D. Schiebener and T. Asfour are with the Institute for Anthropomatics and Robotics, High Performance Humanoid Technologies Lab (H²T), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany. A. Ude is with the Humanoid and Cognitive Robotics Lab, Jožef Stefan Institute, Ljubljana, Slovenia.

schiebener@kit.edu, asfour@kit.edu, ales.ude@ijs.si

The segmentation of unknown objects from a complex unknown background has turned out to be very difficult, if not impossible, if a robot is restrained to passive observation. On the other hand, individual motion of an object is a strong cue that usually dissolves any visual ambiguities, manifests clear object borders and thus vastly facilitates segmentation. Usually, such helpful motion does not happen on its own when needed, therefore the robot has to create it itself. This fundamental idea has been pioneered by the authors of [1] who detect the sudden spread of optical flow from the hand of a robot when it touches and starts to move another object. The pushing motion is pre-programmed, and the obtained segmentation is not used for anything.

In [3], an articulated object is pushed to explore its kinematic properties, i.e. joints and solid parts, exploiting the observed relative 2D motion of local visual features. Again, the robot motion is pre-programmed. In [4], an object is pushed and segmented, which allows for the learning of a visual descriptor. Yet this approach is restricted to symmetric objects in simple scenes. [5] focuses on the singulation of individual objects from a pile by pushing them systematically, and [6] sorts colored bricks from clutter, strongly leveraging physical interaction for separating them. In [7] and [5] heuristics are proposed for systematically pushing clusters of objects in order to separate them.

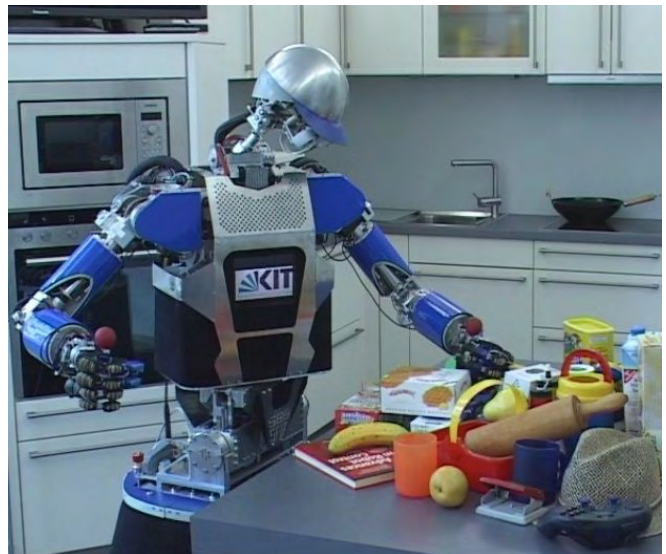


Fig. 1: Interactive object segmentation performed by the humanoid robot ARMAR-III [2]. By pushing unknown objects, they can be segmented from the environment based on the induced motion.

In our previous work (see [8], [9], [10], [11]) we used local visual features (SIFT[12] and later also color MSER[13]) to create initial object hypotheses. Those features are grouped based on their lying on a common regular geometric 3D structure (planes, later cylinders and spheres) as well as spatial proximity. One of these hypothetical objects is then pushed, and by observing the 3D motion of the local visual features, an object hypothesis can be verified by checking if it moves as a rigid body. This also permits to analyze each single local feature for concurrent motion and thus verify the individual features of the hypothesis. Other features that move consistently with the hypothesis are added and thus after two or three pushes a complete object segmentation in terms of the contained local features is achieved. We also demonstrated that this allows for the creation of object descriptors for recognition. In [14], we extended this concept by using the obtained object detection and segmentation to initialize a reactive grasping approach that enables the robot to grasp the formerly unknown object using tactile and haptic feedback without the need for a good 3D model for grasp planning.

While these results are very encouraging, our approach was always restricted to objects which have a sufficiently textured surface that offers enough distinctive local visual features to relocalize the object after it has been pushed. Most of the related approaches are also based on local visual features, with the exception of [15], where unicolored cylinder- and box-shaped objects are segmented interactively, tracking their edges in the image and depth data obtained from a Kinect sensor.

Based on the idea of interactive segmentation that we followed in our earlier work, we have now developed a different approach that enables the segmentation of textured as well as non-textured rigid objects, which we present in this paper. The only remaining restrictions are that the object can be moved by the robot, that it is not completely transparent or looks exactly like the background, and of course that it has an appropriate size with relation to the field of view and resolution of the cameras of the robot.

The following section will give an overview of our approach, which will be explained in detail in sections III and

IV. In section V we present the results of our experiments on the humanoid robot ARMAR-III, and section VI concludes the paper.

II. PHYSICAL INTERACTION FOR SEGMENTATION

Physical interaction enables a humanoid robot to overcome the problems that usually arise if an unknown object is to be segmented in a complex scene that causes visual ambiguities. If the robot is e.g. confronted with a heap of unknown objects, there is probably no certain and infallible criterion to tell two objects apart that can be analyzed by observation only (at least none has been discovered yet). However, if an object moves, it can in principle be distinguished clearly from its environment.

To cause such helpful motion, the robot needs to induce it on the object somehow. The most simple and foolproof way to do so is to carefully push the object. This requires an idea about the existence and location of the object, which we can not take for granted when dealing with unknown objects in an unknown local environment. Consequently, the robot needs the ability to discover possible objects and estimate their location before being able to examine them. Our approach for generating object hypotheses is described in section III-A.

When such an object candidate has been pushed by the robot, there are two possible outcomes: If it moved, the robot can segment it, learn its appearance and examine it further. If it did not move, we have to assume that the robot did not actually push an object but something else that does not move. Thus, we implicitly define an "object" as a physical entity that can be moved (and seen) by the robot. The problem of determining the motion of the object after it has been pushed is not trivial and has only been solved for special cases until now; we present our new and more general solution in section IV-A.

When the motion of the object has been determined, it can be exploited to acquire a complete and certain segmentation of the object in the camera image. We showed in our previous work [9] that if the object motion is known, it is simple to check for each local visual feature if it moved concurrently. But we do not want to rely on the existence of local features (i.e. texture), and we want an actual segmentation that tells for each pixel of the camera images whether it belongs to the object or not. Section IV-B describes how this is achieved.

III. INITIAL OBJECT DISCOVERY

A. Generation of object hypotheses

The first step in our approach for interactive segmentation is to create object hypotheses, i.e. to analyze the camera images of the robot for possible unknown objects. One of these hypotheses is then chosen for pushing and subsequent verification. A criterion for finding object candidates that has proven to be useful in our previous work is grouping of local features that lie on a common regular geometric structure like a plane, cylinder or sphere. Such a structure frequently indicates an underlying object. Another indication for promising candidates are unicolored regions of a

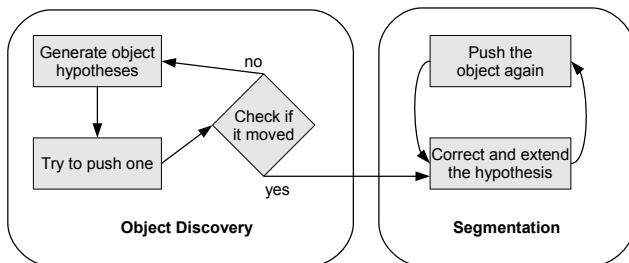


Fig. 2: System overview: Our approach can be divided into two main phases. First, the robot generates object hypotheses and tries to verify one of them by moving it. If an object has been discovered, the segmentation is improved and different views can be learned in the course of several further pushes.

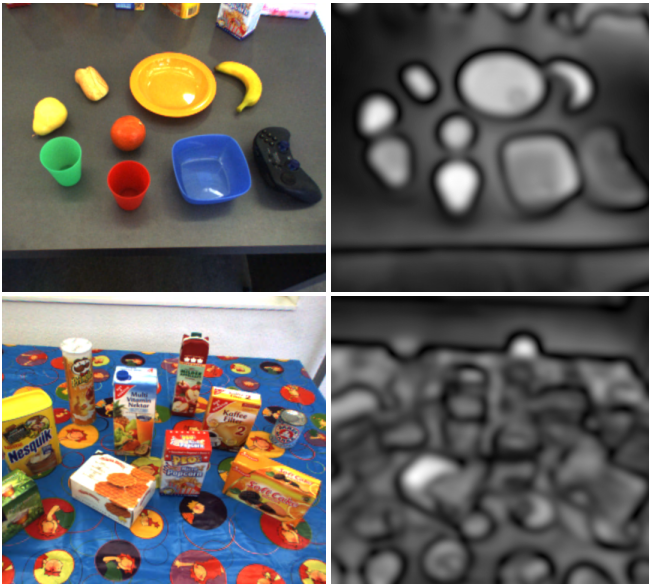


Fig. 3: A relatively simple and a confusing scene with their respective saliency images. As can be seen, the algorithm for saliency computation is not of much use in scenes where objects and background are equally rich in colors and contrasts.

size within the dimensions we would expect an object to have (about 5-50 cm in diameter). While these two criteria are certainly useful, we want to be able to detect objects independently of their appearance, therefore we complement these criteria with the generic concept of visual saliency.

Saliency is a bottom-up trigger for attention, a psychological concept that has been applied in computer vision to support other tasks by restraining the analysis of images to regions that "stand out" in a certain respect (cf. [16] [17]). We use the saliency detector proposed in [18] to calculate a saliency map for the whole camera image. In that work, saliency is defined as the difference of an image region to its neighborhood, which is calculated at different scales using band-pass filters. The filters are realized using a Difference of Gaussian (DoG) filter $G(x, y, \sigma_j) - G(x, y, \sigma_k)$ with $\sigma_j > \sigma_k$. Summing up all edge images at different scales is equivalent to using a filter that is the sum of all filters, which can be simplified as follows:

$$\sum_{n=1}^N G(x, y, \sigma_n) - G(x, y, \sigma_{n+1}) = G(x, y, \sigma_1) - G(x, y, \sigma_N)$$

Thus the resulting saliency image is calculated as $S = |G(\sigma_1) * \text{Img} - G(\sigma_N) * \text{Img}|$, i.e. the difference of the image after being filtered with a Gaussian kernel with the lowest and highest desired standard deviation. This is done for all three color channels of the RGB image, and the results are added. We choose $\sigma_1 = 80px$ which limits the size of detected regions to a size that corresponds to the maximal extent we expect objects to have in the image, and $\sigma_N = 10px$ which smooths out the fine textures that are already accounted for by the hypotheses generation for textured objects.

The resulting salient image regions that are not yet occupied by object hypotheses from the first two criteria (unicolored regions, and local features lying on a regular geometric structure) are used to generate additional object hypotheses. In practice, the first two criteria covered most of the objects we tried, but for those which do not clearly fall into one of the two categories the saliency detection turned out to be a useful complement.

Figure 3 shows the saliency map calculated for different images. As can be seen, in simple scenes it does indeed yield the regions occupied by actual objects. In contrast, if the scene has a rather confusing background, the saliency detection is clearly overburdened and not helpful anymore. The two criteria based on local features and unicolored regions also return very many hypotheses in such a scene. In general, in a nontrivial image the separate use of all three criteria will usually yield a large number of initial object hypotheses.

This is not a fatal problem, as the robot could just systematically try all hypotheses, including those that result e.g. from the tablecloth. But it would save a lot of time to filter the hypotheses beforehand. As we are only interested in things that can be pushed, an additional criterion can be applied in order to keep only those hypotheses that seem to allow pushing. A simple heuristic for estimating if this is the case is to check whether a candidate object is higher than its direct neighborhood. We calculate a dense depth map from the stereo camera images of the robot using semi-global block matching (SGBM) [19]. The resulting 3D points are transformed into world coordinates. The camera image is subdivided into regular bins for which we calculate the average height of the contained points and compare them to their eight direct neighbor cells. Doing this at different scales and adding up the results, we obtain a map that gives a value for the relative local height of the image regions. This map is used to filter the object hypotheses and keep only those that lie in a region which is higher than its direct surroundings. Figure 4 illustrates this relative local height map and its effect on the hypothesis generation.

As we do not want to rely on the existence of local visual features, we use color and shape to describe the object hypotheses. To this end, we calculate a dense depth map from the stereo camera images and annotate the resulting 3D points with their color in the image. This kind of point cloud is usually referred to as RGBD (RGB+depth) data. After the initial object hypotheses have been generated, each one is represented by the RGBD points in the image region that it occupies. These point clouds will be used throughout the rest of the paper.

B. Pushing for Verification

One of the initial hypotheses is chosen to be pushed in order to verify that it is indeed an object and, in case of success, to segment it. We choose the hypothesis that is closest to an optimal location in front of the robot that allows flexible manipulation by both arms, has at least a minimal size, and is higher than its direct surroundings. This is a

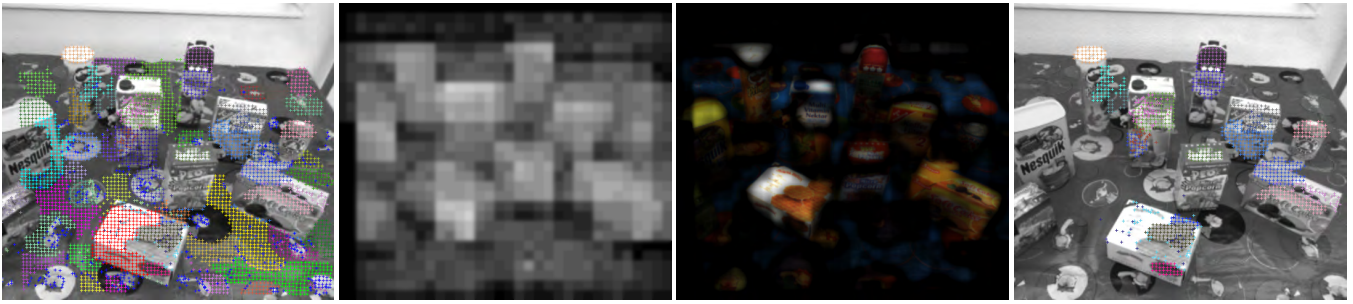


Fig. 4: Suppression of object hypotheses that do not lie in a region that is higher than its direct surroundings. The first image shows a complex scene that leads to the creation of very many initial object hypotheses. The second image displays the map quantifying the relative local height of the image regions. The third image demonstrates the selective effect of applying this criterion: the original image has been multiplied with the height map, thus the high regions are highlighted while lower regions appear dark. The right image shows the remaining initial hypotheses that lie in high regions.

pragmatical choice if the robot does not have any other intention than exploring the objects in front of it. If the object is to be grasped later, it is particularly reasonable to choose one that is higher than its local neighborhood. If the robot is interested in a specific kind of object, other criteria may be appropriate.

The push is planned in such a way that the object is kept in front of the robot and within the camera images. To this end, a central point in front of the robot is defined towards which the object is pushed over a fixed distance to ensure sufficient motion. The motion has to be significant enough to be distinguishable from noise, and as the object extent is unknown, the actual outcome is hard to predict. Therefore the intended motion length should not be too small: values in the range of 10-20 cm turned out to work reliably.

The push that is better suited to execute this push is chosen based on a reachability analysis [20]. The hand approaches the object on a trajectory significantly above it to avoid collisions with other objects. It is then lowered besides the object, and the force-torque sensor in the wrist is used to react to unplanned collisions during that phase (for details see [14] or [11]). The object is pushed, the hand is lifted again and moved out of sight. Afterwards, we analyze if the object has moved and determine its translation and rotation.

C. Detection and Analysis of Change in the Scene

Now we have to find the object that moved by comparing the point clouds before and after the push, which is the most important and most difficult subtask within our segmentation approach. This is due to the fact that (besides the general difficulty of the matching of point cloud subsets) we do not know which part of the point cloud is the object, neither for the cloud before nor the one after the push. Thus, we have to use the difference between them to determine both the subset constituting the object and the transformation that it underwent.

As a first step, we determine which part of the point cloud changed due to the push. This can easily be achieved by comparing the old and new camera images and calculating the difference image. Yet that is only possible if the camera

pose before and after pushing is virtually the same. On our robot ARMAR-III, the precision and repeat accuracy of the joints is high enough to allow that; we only need to shift the new image by up to four pixels in all directions when comparing it with the old one, and choose the modified position that causes a minimal difference. On other robots such a precise motion might not be possible, in which case an alternative is to align the two point clouds and find the points that are far away from their nearest neighbor or have a different color. Both methods yield comparable results and enable us to divide the old and new point cloud into a part that is unchanged and a part where a change occurred.

A first result we get immediately from this difference is an answer to the question if anything happened at all. If nothing changed in the scene, the robot was evidently unable to move the potential object or did not hit anything at all. In this case, the robot tries pushing another object candidate. If a change in the scene is detected, all initial object hypotheses are analyzed on whether they lie in image regions that changed. Each object hypothesis is represented by a set of RGBD points, and if more than half of them lie in a region that changed due to the push, the hypothesis will be analyzed for having moved; otherwise it is discarded. In addition to the initial hypotheses, we create new ones from the points that changed. This is done by determining 2-5 clusters¹ amongst these points using x-means, a variant of k-means that automatically chooses the number of clusters [21]. These new hypotheses frequently match the actual object better than the initial ones, although usually not perfectly either.

IV. OBJECT SEGMENTATION

A. Estimation of the Object Motion

All the hypotheses that lie in parts of the scene which changed may correspond to the object (or one of several objects) that moved, and therefore they are examined further.

¹There have to be at least two clusters, as a moving object causes change in the image regions of its old and new position (which may overlap though). More clusters may be appropriate if several objects move, or if there are false foreground regions due to errors in the background subtraction.

Each hypothesis consists of a set of 3D points with associated color information from the point cloud recorded before the push and has to be relocalized within the new point cloud. The probably most popular approach for matching (also referred to as *registration*) of 3D point clouds is the Iterative Closest Point (ICP) algorithm [22]. To register a point cloud with another, two steps are repeated iteratively:

- The nearest neighbor of every point of the first point cloud is determined in the second point cloud
- Based on these correspondences, the 3D transformation that minimizes the mean squared distance between all the pairs is calculated and applied to the first point cloud

These two steps are repeated iteratively until the improvement, i.e. the relative reduction of the mean square distance, lies below a threshold, or a maximal number of iterations has been executed. The algorithm reduces the mean square distance between the point sets in each step and converges to a local minimum.

In our implementation, we define the distance between two points as the weighted sum of their cartesian distance and their distance in normalized RGB space. The weighting is such that the maximal possible color distance is equivalent to a cartesian distance of 10 cm.² As we use both shape and color information, we avoid the problem of mismatching in case of similar shapes which would otherwise occur frequently, as the shapes of artificial household objects are mostly dominated by planar surfaces.

When trying to determine the transformation that a hypothetical object underwent during the push, we first try to register the hypothesis with the new point cloud by initializing ICP with its original pose (= position and orientation). If a good match is found, i.e. the resulting (cartesian + RGB) distance is small and the determined transformation indicates that the hypothesis did not move significantly, we consider it to be unchanged. If the determined transformation indicates that the object has moved, or only a bad match was found, it has to be relocalized. The one serious disadvantage of ICP is that it converges to a local optimum, therefore its initialization is decisive for finding the correct match of the object hypothesis after a push. Starting the registration at the original position frequently fails in complex scenes if the object moved over a large distance.

Thus, we execute ICP several times with different initial estimates of the new object pose, and keep the resulting transformation that yields the best match. As the object may have been moved over a large distance, finding it again requires an appropriate choice of the initial poses for ICP. To this end, we detect image regions that resemble the hypothesis in terms of color histogram similarity and initialize the alignment there. If the object surface contains stable local visual features, those can be used to get an

²This parameter allows to balance the relative importance of color and shape matching. The weight of the color component should not be too small to avoid mismatching due to similar shapes. If it is set too high, the risk of mismatches due to similar color rises. Empirically, values between 5 and 30 cm produced reasonable behavior. The choice may also depend on the precision of the 3D sensor and the sampling density.

initial estimation of the motion, too. The necessary number of different initial positions can be reduced by taking into account the direction of the push, which must not be done in a too restrictive manner as the caused object motion is rather unpredictable.

The best transformation returned by the differently initialized registration attempts is refined by another execution of ICP on a reduced point set where all those points are left out that still have a large distance to their nearest neighbor. The resulting final transformation is used to decide whether the estimated object motion is accepted, and if this is the case, to determine the object segmentation.

B. Verification, Correction and Extension of the Segmentation

After the motion of an object hypothesis has been estimated, the robot needs to decide whether the determined match and transformation are plausible. A hypothesis is only accepted, i.e. considered to correspond to an actual object, if it meets the following three criteria: First, the estimated motion has to be large enough to be sure that it is not due to noise or a slight mismatching³. Secondly, the match must be good, i.e. the average distance of the hypothesis points to their respective nearest neighbors in cartesian and normalized RGB space must be below a threshold. Thirdly, the relocalized hypothesis must lie mostly in image regions that have changed. This removes mismatches where by pure chance a good alignment to some part of the scene could be found, e.g. a part of the table surface that was matched to another part of the table after the object has been moved onto it.

The remaining hypotheses do most likely belong to an actual object that has been moved by the robot. But of course we must assume that they do not cover the object completely, and that they also contain points that do not belong to the object. We remove the latter ones by checking each point of the hypothesis: After applying the estimated object motion, a point must match its nearest neighbor in the scene point cloud well with respect to cartesian and color distance. It also has to lie in a region that changed due to the push. If both of these criteria are met, the point is considered to be verified, otherwise it is removed from the hypothesis.

After removing the false points, we try to extend the hypothesis to cover the whole object. To this end, we add all those points to it as candidates that lie close to the verified points and within the image region that changed. By pushing the object again and repeating the steps described before, these new candidate points can be verified or discarded, and new candidates can be added. Depending on the object size and the quality of the initial hypothesis, it usually takes two or three pushes until the whole object is contained in the hypothesis and thus segmented completely.

Usually, more than one object hypothesis is verified by the first push and the subsequent analysis. This happens

³Given the precision of our stereo calibration and a distance of 50-80 cm between camera and object, a threshold of 3 cm turned out to be definitely safe.



Fig. 5: Examples of object segmentations in different scenes. The first image in each row shows the initial object hypotheses, the second to fourth images show the verified hypothesis after one, two and three pushes.

in particular when several actual objects are moved. We choose the hypothesis containing the maximal number of confirmed points for the second push. After that, we discard the hypotheses that did not move again, and from the remaining ones we keep only the one with the maximal number of confirmed points and continue examining it as long as desired. If the robot did indeed move several objects, all of them can be segmented, but for the sake of simplicity we only observed one in our experiments. As long as the objects undergo different 3D transformation, they can easily be separated based on their different motion. It may happen though that two objects move exactly alike, in which case they are subsumed in one hypothesis. Most likely they are separated when pushed several times from different directions. Heuristics for systematical pushing to this end have been proposed in [5] and [7]. When two objects contained in one hypothesis are separated, the hypothesis will follow the object that is matched better after the motion, which is usually the bigger one.

Pushing an object several times will reveal different sides of it, thus the creation of a multi-view object descriptor is possible, although some sides will probably never be observed. In section V-D we demonstrate that the obtained segmentations are well suited to train an object descriptor that allows for reliable recognition.

V. EXPERIMENTAL EVALUATION

A. System Setup

We have implemented and tested our approach on the humanoid robot ARMAR-III [2]. The video accompanying this paper shows an interactive object segmentation executed by it. The robot has an active stereo camera system in its head, and its arms have seven degrees of freedom each and are equipped with force-torque sensors in the wrists. The cameras provide color images with a resolution of 640×480 pixels. About 85% of the stereo images overlap, and after calculating the dense depth map we use only every second pixel in x and y direction for the point cloud, thus we obtain around 65000 RGBD points that we work with.

The computational effort is dominated by the relocalization of the object hypotheses using the ICP algorithm, in which the computational complexity is proportional to $n \log(m)$, with n being the number of points of the object hypothesis and m the overall number of points in the scene. On a 3 year old standard PC with a quadcore processor, the computations after each push took between 2 and 5 seconds, depending on the size and number of moved objects.

An important aspect in comparison to some related work is that in our case the robot itself executes the object pushing, and we do not use an artificial setup where the camera always has an undisturbed view of the object. This is the reason why we do not try to track the object during the push, as the robot's hand frequently occludes large parts of it.

B. General Observations

Our approach aims at making it possible to segment rigid objects independently of their appearance or shape, thus we

tested it with a large variety of items. They can roughly be classified by their visual appearance as being strongly textured, sparsely or partially textured, multicolored but (almost) non-textured, unicolored, reflective (e.g. polished metallic objects or mirrors), or transparent. As far as we know, the related work in this field (including ours) has so far either depended on local features, i.e. texturedness, on unicoloredness, or on a certain shape.

It turned out that our segmentation approach works very well for all kinds of objects except the very reflective and the transparent ones. This is due to the fact that they appear to change their color when moved, and also tend to cause problems when trying to obtain depth information. All other objects were segmented successfully by our approach; for the transparent and reflective ones a special treatment might be necessary. Although we were able to tune the parameters of the background subtraction and the matching so that the segmentation worked for most of them, it does not function reliably and the chosen parameters depend strongly on the lighting conditions, thus we do not claim that our approach can handle this kind of objects.

In contrast, the shape of the examined objects did not seem to make an observable difference. While distinctive shape features are necessary for algorithms that match point clouds solely based on 3D data, the fact that we use color helps to overcome ambiguities that might arise otherwise. The combined use of shape and color information usually allows a good alignment of the object hypothesis with the object after it has been pushed. An exception here are symmetric unicolored objects, but in that case it actually does not matter if the orientation around the axis of symmetry is met correctly as long as the match is good. The only case in which problems occurred was when a flat, unicolored object was placed on a table of the same color.

C. Assessing the Segmentation Quality

We examined the performance of our interactive segmentation approach by testing it with 30 objects of different shape, size and visual appearance type (as defined above), which have been segmented twice each. To measure the quality of the obtained segmentations, two metrics are determined: First, the object should be segmented as completely as possible, i.e. in an optimal case the point cloud forming the object hypothesis should fully cover the object. The second metric is the size of the falsely segmented area, i.e. the part of the scene that is segmented but does not belong to the object. This happens when the object hypothesis includes points that belong to the background or other objects.

Figure 6 shows these two values depending on the number of pushes executed. As can be seen, after the first push the object is usually not covered completely, but already to a large part. After two to three pushes, the hypothesis contains almost the complete object, with the exception of small patches that newly appeared due to object rotation or that were discarded from the hypothesis due to a change in their appearance (e.g. reflections or bad depth estimation). After four or more pushes, the coverage does not improve

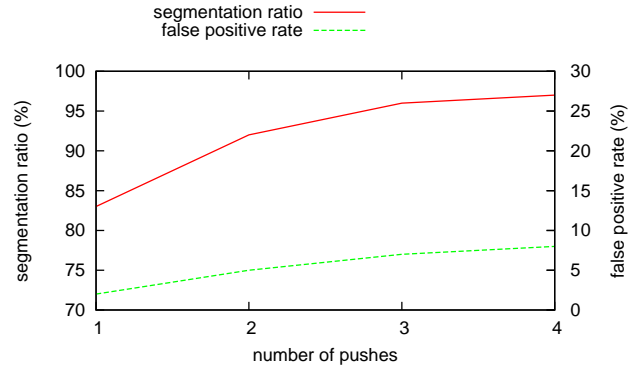


Fig. 6: The average segmentation quality depending on the number of pushes that were executed. The red line shows the segmentation ratio, i.e. the percentage of the object that is included in the segmentation. The dashed green line depicts the false positive rate, i.e. the fraction of the segmentation that does not belong to the actual object.

further, but different parts of the object may become visible, thus more information can still be gained.

The ratio of falsely segmented image regions compared to the whole object is always quite small. It seems to grow a bit from the first to the second push, but not any further afterwards. Such false positives occur when the shadow cast by the object leads to neighboring image regions being considered to have changed, and some of them look alike before and after the push, which frequently happens on unicolored table surfaces. In this case, the part of the table on which the object casts a shadow appears to belong to the object itself. We are not sure whether there is a theoretically sound solution for this specific ambiguity; it is probably necessary to grasp and lift the object to dissolve it.

D. Learning of an Object Descriptor

To demonstrate that the obtained segmentations are sufficiently complete and correct, we use them to train a simple object recognition system. The available information we can use are the image region that contains the object hypothesis, i.e. the segmentation, as well as the 3D and color information contained in the hypothesis point cloud itself. After each push, the object hypothesis and thus the segmentation are different, therefore we could generate several descriptors for each object from different perspectives. For the sake of simplicity, we just use the segmentation obtained after the second push for each object, which usually yields a good coverage, and generate only one descriptor.

To detect the learned objects in new images, we train a color histogram based descriptor using the image region that is occupied by the object hypothesis. The descriptor uses Receptive Field Cooccurrence Histogram (RFCH) features [23], [24] which are based on histograms of the colors and their first and second derivatives in the segmented image area.

These features allow to find image regions that have the same color distribution as the learned object. We then try to

TABLE I: Object recognition rates.

similar point of view	different point of view	partly occluded	false positive rate
98.5 %	70.6 %	67.2 %	3.8 %

match the learned RGBD point cloud in those areas using Iterative Closest Point (ICP) as in the motion estimation step of our segmentation approach. The localization is accepted if the resulting average point distance in Cartesian and color space is below an equivalent of 1 cm (with the maximal possible color distance being equivalent to 10 cm in Cartesian space).

Table I displays the recognition results for our set of autonomously learned objects. They are placed in potentially confusing scenes comparable to those shown in figure 5. When the object is seen from approximately the same point of view as during learning, the recognition rate is almost 100%. If the object has a significantly different orientation with relation to the camera, or if it is partly occluded by other objects, the recognition rate drops to around 70%. This can be improved by using object descriptors generated from different views, as we did in [11]. The false positive rate is about 4%, which is entirely due to two small unicolored objects in our test set that are sometimes fitted into blobs of similar color. These solid recognition results demonstrate the usefulness and quality of the segmentations obtained by the robot following our approach.

VI. CONCLUSIONS

We have presented a new approach for interactive object segmentation exploiting the manipulation capabilities of a humanoid robot. The proposed method enables it to discover and segment unknown rigid objects in an unknown, complex scene by pushing them and analyzing the motion of color-annotated 3D points obtained from the robot's stereo vision system. We have demonstrated that the provided segmentation results are of excellent quality and allow to train a well performing object recognition system. As already shown in [14], it is also possible to subsequently grasp the discovered objects for further examination or manipulation.

In contrast to our previous work in this direction, the approach proposed here works with almost any kind of rigid object except those which are transparent, highly reflective or impossible for the robot to move. We therefore believe that it is a small but important step for increasing the adaptability and autonomy of humanoid robots that will frequently have to deal with new, unknown objects in realistic scenarios.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement 270273 (Xperience).

REFERENCES

[1] G. Metta and P. Fitzpatrick, "Grounding vision through experimental manipulation," *Philosophical Transactions of the Royal Society: Mathematical, Physical and Engineering Sciences*, vol. 361, no. 1811, 2003.

[2] T. Asfour, K. Regenstein, P. Azad, J. Schröder, A. Bierbaum, N. Vahrenkamp, and R. Dillmann, "ARMAR-III: An integrated humanoid platform for sensory-motor control," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2006.

[3] D. Katz and O. Brock, "Manipulating articulated objects with interactive perception," in *IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, CA, 2008.

[4] W. H. Li and L. Kleeman, "Segmentation and modeling of visually symmetric objects by robot actions," *International Journal of Robotics Research*, vol. 30, no. 9, 2011.

[5] L. Chang, J. Smith, and D. Fox, "Interactive singulation of objects from a pile," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 3875–3882.

[6] M. Gupta and G. Sukhatme, "Using manipulation primitives for brick sorting in clutter," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 3883–3889.

[7] T. Hermans, J. Rehg, and A. Bobick, "Guided pushing for object singulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2012.

[8] E. S. Kuzmič and A. Ude, "Object segmentation and learning through feature grouping and manipulation," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2010.

[9] D. Schiebener, A. Ude, J. Morimoto, T. Asfour, and R. Dillmann, "Segmentation and learning of unknown objects through physical interaction," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Bled, Slovenia, 2011.

[10] A. Ude, D. Schiebener, N. Sugimoto, and J. Morimoto, "Integrating surface-based hypotheses and manipulation for autonomous segmentation and learning of object representations," in *IEEE International Conference on Robotics and Automation (ICRA)*, St. Pauls, Minnesota, 2012.

[11] D. Schiebener, J. Morimoto, T. Asfour, and A. Ude, "Integrating visual perception and manipulation for autonomous learning of object representations," *Adaptive Behavior*, vol. 21, no. 5, pp. 328–345, 2013.

[12] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. Int. Conf. Computer Vision*, Corfu, Greece, 1999.

[13] P. Forssten, "Maximally stable colour regions for recognition and matching," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[14] D. Schiebener, J. Schill, and T. Asfour, "Discovery, segmentation and reactive grasping of unknown objects," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2012.

[15] K. Hausman, F. Balint-Benczedi, D. Pangercic, Z. Marton, R. Ueda, K. Okada, and M. Beetz, "Towards tracking-based interactive segmentation of textureless objects," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[16] S. Frintrop, E. Rome, and H. Christensen, "Computational visual attention systems and their cognitive foundations: A survey," in *ACM Transactions on Applied Perception* 7, 2010.

[17] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998.

[18] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk, "Frequency-tuned salient region detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 1597–1604.

[19] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, Feb. 2008.

[20] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Efficient inverse kinematics computation based on reachability analysis," *International Journal of Humanoid Robotics*, vol. 9, no. 4, 2012.

[21] D. Pelleg and A. Moore, "X-means: Extending k-means with efficient estimation of the number of clusters," in *Proc. 17th Int. Conf. Machine Learning*, San Francisco, CA, 2000.

[22] P. Besl and N. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[23] K. Welke, "Memory-based active visual search for humanoid robots," Ph.D. dissertation, Karlsruhe Institute of Technology (KIT), Computer Science Faculty, Institute for Anthropomatics (IFA), 2011.

[24] S. Ekvall and D. Kragic, "Receptive field cooccurrence histograms for object detection," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2005, pp. 84–89.

Generalized Exemplar-Based Full Pose Estimation from 2D Images without Correspondences

Damien Teney

University of Liège, Belgium
Damien.Teney@ulg.ac.be

Justus Piater

University of Innsbruck, Austria
Justus.Piater@uibk.ac.at

Abstract—This paper addresses the problem of full pose estimation of objects in 2D images, using registered 2D examples as training data. We present a general formulation of the problem, which departs from traditional approaches by not focusing on one specific type of image features. The proposed algorithm avoids relying on specific model-to-scene correspondences, allowing using similar-looking and generally unmatchable features. We effectively demonstrate this capability by applying the method to edge segments. Our algorithm uses successive histogram-based and probabilistic evaluations, which ultimately recover a complete description of the probability distribution of the pose of the object, in the 6 degree-of-freedom 3D pose space, thereby accounting for the inherent ambiguities in the 2D input data. Furthermore, we propose, in a rigorous framework, an efficient procedure for fusing multiple sources of evidence, such as multiple registered 2D views of the same scene. The proposed method is evaluated qualitatively and quantitatively on synthetic and real test images. It shows promising results under challenging conditions, including occlusions and heavy clutter, while being capable of handling objects with little texture and detail.

I. INTRODUCTION AND RELATED WORK

Estimating the pose of a known object in a single 2D image is a fundamental problem in computer vision that has attracted a lot of attention over the years. The task is closely related to the problem of object recognition. However, state-of-the-art object recognition methods usually aims at identifying object *classes*, allowing small variability in appearance among different objects of the same class. We rather focus here on specific *instances* of objects, where such small changes in appearance are actually used as cues for determining the precise pose (3D position and orientation) of the object in a new scene.

The pose estimation task has many direct applications, such as robotic interaction and grasping, augmented reality, or the visual tracking of objects. Methods have been developed that make use of a 3D, explicit geometric model of the object of interest [1], [2], [3]. Those thus require precise a-priori knowledge of the 3D shape of the object, to be provided by external methods such as stereo vision or range sensors. In this paper, we rather present a 2D view-based, or exemplar-based method, which simply uses 2D views of the object as training data, in which the object appears in known poses. Those methods present the advantage of being easily trainable, *directly* using 2D visual data. Further motivation for the exemplar-based approach is brought by the human visual system, which was shown to exhibit properties of a

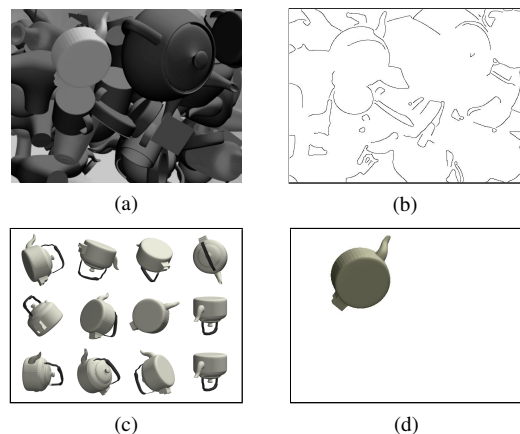


Fig. 1. Pose estimation in a single image, using 2D training examples; (a) test image; (b) edge map used as input; (c) sample training views; (d) rendering of a model of the object in the best pose found by the algorithm, note that the correct pose is recovered despite heavy clutter and missing observations.

view-based lookup function when recognizing objects, being robust to changes of about 20° around trained viewpoints [4]. Unfortunately, current, state-of-the-art methods following this approach present serious limitations, often relying on specific types of images features, or being suited to only particular types of objects, and are thus able to operate only under limited ranges of conditions. This led us to the reformulation of the problem in more general, probabilistic terms, and to the development of a novel method, that we will introduce after reviewing related work.

Early work in the field of exemplar-based methods used the appearance of the object as a whole. These so-called *appearance-based* methods [5], [6], [7] generally assumed a successful prior detection of the object in the test image and generally offered poor resistance to clutter and occlusions, or did not handle the full 6 degree-of-freedom pose space as needed in practical applications. More recent work, by contrast, focused on the use of individual, precisely located observations (such as *SIFT* features [8]) extracted in the 2D views of the object. These *feature-based* methods [9], [10] then rely on establishing matches, using their appearance, between observations in the test view and in the stored training examples. The limitations of this approach are obviously those of the extraction and matching of image features, which practically works best on texture-rich images, but can perform

poorly on scenes with mostly homogeneous surfaces or little detail.

The method proposed in this paper bridges a gap between the two approaches mentioned in the previous paragraph. It makes use of individual features extracted from the images, thereby offering the potential robustness of feature-based methods, e.g. against lighting changes, but does not rely on the matching of specific observations between the test and training views. Practically, this allows using similar-looking types of features. Although the method is generally applicable to different types of observations, we chose to demonstrate this key ability through the use of local edge segments. These correspond to points extracted in the images along the lines of maximum gradient, and they thus carry little appearance information individually. The result of our implementation is a pose estimation method readily trainable with 2D visual data, intrinsically robust to clutter and occlusions, and able to handle previously-problematic objects with little texture and detail.

The identification of the object of interest in a new image resembles the traditional problem of object recognition and localization. A number of successful methods have been developed that specifically make use of edges as image features. The classical measures of chamfer distance [11] and Hausdorff distance [12] evaluate the fit of a template over a test image; their initial formulations were refined in different ways to provide practical algorithms capable of finding such a template (a training image of the object of interest) in a cluttered scene [13], [14]. One key addition proved to be the use of the orientation of the edges, as we also do in the proposed method. Other state-of-the-art methods include the work of Ferrari *et al.* In [15], they use descriptors of simple edge groupings to train an SVM classifier, capable of recognizing object classes, then using a traditional sliding window over the test image. In [16], they focus on the learning of shape models from unsegmented training views, and then use a soft-matching procedure of those shapes to recognize objects in new images. The purpose of those two methods is however to specifically handle intra-class variations of appearance. The work presented in this paper differs from the cited methods in 3 important ways: (i) we present a generally-applicable method not bound to one specific type of image features; it offers the flexibility to use additional characteristics (e.g. edge curvature) or other features (e.g. interest points); (ii) we do not seek to identify objects or object classes, but rather to determine their pose, using the small changes of appearance as clues to this end; (iii) we go beyond a simple localization in the image (e.g. as 2D bounding box), as we directly consider the full 6-degree-of-freedom pose of the object in the 3D space, of which we recover a probability distribution, and not a single maximum.

The method proposed in this paper is based on a probabilistic representation of both the test and the training data. Such a representation has been used in the slightly different context of pose estimation using 3D models and observations [17], [3], and this work can be seen as their extension to the case of 2D data. In addition to modelling the uncertainty inherent in the

input data, the probabilistic approach leads to the definition of the pose of the observed object as a probability distribution in the 3D pose space, of which we want to identify the peaks. This is justified by the uncertainty in the pose estimation problem arising from the 3D-to-2D projection ambiguities. Intuitively, a given 2D view may often be explained by several 3D poses of the object of interest, and we are generally interested in recovering *all* these potentially correct results. Our probabilistic approach, as will be demonstrated, is able to address this objective. Another contribution of this paper is the introduction of successive histogram- and probabilistic-based evaluations that seek to identify *all* significant modes in the distribution of interest. The aforementioned references, which had to deal with less complex distributions, employed approximations such as Monte Carlo methods [18], which generally recovered only a unique solution. This would have been insufficient in the present case, due to the particular ambiguities mentioned above.

Finally, we propose an efficient method for fusing multiple sources of evidence in the same probabilistic framework. This information may be available e.g. through multiple 2D views of the scene, observed under different viewpoints, but the same principle can also serve to jointly handle multiple types of features extracted from a same image. Viksten *et al.* [10] proposed another method for combining such multiple sources of information through a simple clustering step on top of several instances of existing methods. This however lacks the genericity offered by the rigorous approach proposed here. We make full use of the probabilistic nature of the problem, combining the different sources of information in a Markov random field, on which inference is performed using non-parametric belief propagation. The power of the technique is demonstrated through the use of two 2D views of the same scene, thereby increasing the accuracy of the pose estimation process. A comparable approach was used by Toshev *et al.* [19] for tracking of the pose of an object over time in a video. Other methods for handling multiple views with a 2D pose estimation method have been proposed [1], [20], but with the underlying process based on feature matching, as opposed to the more generic approach proposed here.

II. PROBABILISTIC REPRESENTATION OF POSE AND APPEARANCE

In this section, we introduce a rigorous formulation of the pose estimation problem, using a probabilistic representation of the input data. As mentioned above, the proposed method is not specific to one particular type of image features, but the general formulation is illustrated with local edge segments. Those correspond to points extracted from the images along the lines of maximum gradient (see Section V).

A. Representation of test data

Let us first consider the test data, which consists of a single 2D image, from which we extract features x_i . They form the set of *observations* $\mathcal{O} = \{x_i\}_{i=1}^N$, where $x_i \in \mathcal{A}$, the space on which is defined the appearance of our observations. In the

case of local edge segments, an observation is characterized by its 2D position in the image, and by its orientation (without direction, i.e. an element on the semicircle). Therefore, we have $\mathcal{A} = \mathbb{R}^2 \times S_1^+$. Considering another case where each observation would be a texture patch extracted around an interest point, the appearance space \mathcal{A} would then encompass the position of that point, and a description of the texture itself.

As proposed in [3], such a set of observations can be used to define a continuous probability density ϕ on \mathcal{A} . This distribution is defined in a non-parametric fashion, using Kernel Density Estimation (KDE), directly using the elements of \mathcal{O} as supporting particles. The probability density function of ϕ is then given by

$$\phi(x) = \frac{1}{N} \sum_{x_i \in \mathcal{O}} K_1(x_i, x), \quad (1)$$

where $x \in \mathcal{A}$, and $K_1(\cdot, \cdot)$ a kernel function on \mathcal{A} . This formulation allows modelling the uncertainty that may be present in the observations, e.g. due to image noise or to other artifacts occurring during image formation and processing. The kernels used will depend on the appearance space considered [3]. In our application, using edge segments, we found that using kernels allowing only a small deviation on the position and on the orientation was sufficient, as our edge detection algorithm could provide results of good accuracy (see Section V). In practice, the narrow bandwidth of the chosen kernels implies that sampling from $\phi(x)$ amounts to selecting random points x_i from \mathcal{O} , with only small variations (see Fig. 2b).

B. Representation of training data

The training data is composed of a number of pre-segmented 2D images, in which the object of interest appears in known poses. Each of those images is processed, in a similar way as the test image, to extract image features. Each observation x_i is then associated with the pose w_i of the image it was extracted from, thereby forming a set of *pose/appearance pairs* $\mathcal{T} = \{(w_i, x_i)\}_{i=1}^M$, where $x_i \in \mathcal{A}$, the appearance space of our observations, and $w_i \in \text{SE}(3)$, the space of 3D poses. Similarly to the observations, these points are used to support a KDE, therefore defining a probability distribution on the joint pose/appearance space. This distribution, called ψ , represents the probability of observing an image feature of a given appearance when the object is in a given pose. Formally, ψ is defined by its density function

$$\psi(w, x) = \frac{1}{N} \sum_{(w_i, x_i) \in \mathcal{T}} K_2((w_i, x_i), (w, x)), \quad (2)$$

where $w \in \text{SE}(3)$, $x \in \mathcal{A}$ and $K_2(\cdot, \cdot)$ is a kernel function on $\text{SE}(3) \times \mathcal{A}$. The use of kernels on the training data can be seen here as a smoothing over the available training points, effectively yielding a continuous distribution and allowing us to interpolate, to some extent, the value of ψ over regions not covered by the training data. Practical details on the use of kernels in $\text{SE}(3)$ are discussed e.g. by Detry and Piater [18].

In addition to the training data, a number of possible transformations in the pose/appearance space are usually known.

For example, under orthographic projection¹, the camera intrinsic parameters dictate how a translation (in pose space) parallel to the camera image plane relates to a translation of the observations in the image (in appearance space). In our case, with edge segments, we chose to hard-code three such transformations, namely the translation and rotation in the image plane, and the change of depth along the camera projection rays which give identical projections on the image plane. Formally, we represent these transformations via a single function f , parameterized by a vector of parameters $p \in \mathcal{P}$, such that

$$f((w, x), p) = (w', x') \quad (3)$$

with (w, x) and (w', x') being pose/appearance pairs, equivalent through the hard-coded transformations under the parameters p . Those transformations allow us to extend our definition of ψ to larger regions of the pose/appearance space than with the training points alone. To that effect, we substitute \mathcal{T}' for \mathcal{T} in Eq. 2, where

$$\mathcal{T}' = \mathcal{T} \cup \left\{ (w', x') : \exists (w, x) \in \mathcal{T}, p \in \mathcal{P} : \right. \\ \left. f((w, x), p) = (w', x') \right\}. \quad (4)$$

This *augmented* training set \mathcal{T}' complements \mathcal{T} with all transformations of its elements that can be obtained using f . As we will see in Section III however, our implementation does not require an explicit representation of \mathcal{T}' , and, in practice, only a small subset of its elements will have to be identified.

For practical purposes, we remark that the definition of ψ (Eq. 2) presents the problem of making its value dependent on the density of training examples in the corresponding region. For example, including two identical views of the object in the training data, in the same pose, would simply double the density of ψ in the corresponding regions, which is not desirable. This effect is alleviated by using the maximum value of the neighbouring kernels (see Fig. 2c) instead of a summation over their values. This leads to the alternative definition

$$\psi(w, x) = \frac{1}{C} \max_{(w_i, x_i) \in \mathcal{T}'} K_2((w_i, x_i), (w, x)), \quad (5)$$

where C is a normalization constant.

C. Probability distribution of 3D pose

The probabilistic representations of the test and the training data, given respectively as ϕ and ψ , are now used together to model the pose of the object in the test image. The pose is modelled as a random variable $W \in \text{SE}(3)$, and its distribution is simply given by

$$p(w) = \int_{\mathcal{A}} \psi(w, x) \phi(x) dx. \quad (6)$$

¹Our implementation of the method assumes an orthographic or near-orthographic projection, which in practice is easily satisfied with a camera of sufficient focal length relative to the scene depth (see Section V).

This expression, in effect, measures the compatibility of a pose w with the whole distribution of features observed in the image. Another interpretation is to see it as the cross-correlation of the distribution ϕ of observations in the test image with the distribution $\psi(w, \cdot)$ of training points at a given pose. Note that this formulation of $p(w)$ is similar to that proposed in [18], [3] for the use of 3D models and observations.

III. POSE INFERENCE

This section presents a practical method for solving the pose estimation problem as formulated in Section II. The method is based on two key observations, presented below, which allow an approximate evaluation of $p(w)$.

First, the value of the integral in Eq. 6 can be approximated using Monte Carlo integration [21], [18]. This method, which involves a random exploration of the integration domain, gives

$$p(w) \approx \frac{1}{n} \sum_i^n \psi(w, x_i) \quad \text{where } x_i \sim \phi(x). \quad (7)$$

The evaluation of $p(w)$ (see Fig. 2a–d) thus amounts to successive evaluations of $\psi(w, x_i)$ for different values of x_i , drawn from the distribution of observations in the test image (ϕ).

Importantly, and this is our second key observation, each of these evaluations of $\psi(w, x_i)$ only requires a small number of elements of the *augmented* training set \mathcal{T}' . For a fixed x_i , using the hard-coded transformations (in-plane rotation and translation), any original training pair $(w, x) \in \mathcal{T}$ can be transformed into a pair $(w', x_i) \in \mathcal{T}'$ of appearance x_i . Those pairs will have the strongest influence on the value of $\psi(w, x_i)$ (Eq. 5), and its evaluation can therefore be limited in practice to the use of those pairs, which formally correspond to the following subset of \mathcal{T}' :

$$\left\{ (w', x_i) : \exists (w, x) \in \mathcal{T}, p \in \mathcal{P} : f((w, x), p) = (w', x_i) \right\} \subset \mathcal{T}' \quad (8)$$

The practical consequence of this property is that an explicit and complete representation of \mathcal{T}' is not required, and that only a fraction of its elements have to be identified.

A. Exhaustive search algorithm

The two properties we just presented make the evaluation of $p(w)$ possible for any pose w . Various methods can then in principle be used to identify the main modes of this distribution, such as a Monte Carlo-type search as proposed in [18], [3]. However, the purpose of such methods is generally to identify the global maximum of the density. As argued above, the particular ambiguities in the 2D input data are likely to induce a very complex distribution, potentially presenting multiple weak modes that we wish to identify. We therefore devised an algorithm to exhaustively explore the relevant parts of the 3D pose space. This task is particularly challenging [22] due to the high dimensionality of $SE(3)$. We propose a two-stage process that first relies on a histogram-based

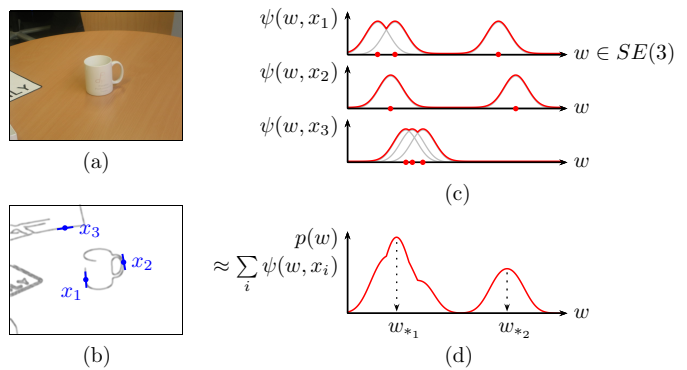


Fig. 2. Proposed method for pose estimation, using edge segments as image features. (a) Test image; (b) distribution of edges in test image, denoted $\phi(x)$ in the text, and three samples x_{1-3} (blue oriented points) of that distribution; (c) distribution (red curve) of poses compatible with each observation x_i (Eq. 5), made up of individual kernels (gray curves) supported by a small subset of poses $w' \in \mathcal{T}'$ (Eq. 8, red dots); (d) distribution of poses compatible with all observations x_i (Eq. 7) and local maxima w_{*i} , as recovered by our method.

approximation, in order to pre-select regions of interest in $SE(3)$. This serves to discard those bins of the histogram that correspond to areas of low density, dramatically reducing the amount of data used at the second stage. It is then possible to perform a full-scale kernel-based evaluation of the density (Eq. 5,7), limited to the pre-selected regions of the pose space. The algorithm returns a set of poses \mathcal{R} where the density exceeds a certain threshold.

The computational complexity of this algorithm is proportional to the number of training points (M , Section II-B), multiplied by the number of samples used from the observations (n , in Eq. 7), itself chosen as a fraction of the total number of observations (N , Section II-A). Note also that it is not mandatory to process all possible combinations of observation samples and training points, but a stochastic approach can rather make use of the probabilistic representation of the training data ϕ , and use a limited number of random samples thereof. This scheme was previously used in the related problem of 3D models and observations [17], [3].

B. Post processing of pose estimates

As a post-processing step, one may want to identify the actual peaks of each mode. This could be accomplished by a traditional gradient-ascent method, such as mean shift [23]. In our case, this procedure would be costly due to the complexity of the pose space. Fortunately, in practice, the proposed algorithm usually returns poses in the close neighbourhood of the actual peaks. A simple *non-maximum suppression* step therefore proves sufficient. In this method, an element is discarded if it lies in the close neighbourhood of an element of greater density, the neighbourhood being defined by a fixed radius in the pose space. This procedure, efficiently implemented by processing the poses of \mathcal{R} in order of decreasing density, therefore selects the poses that are the closest to the peaks of the distribution (Fig. 2d).

IV. EXTENSION TO MULTIPLE SOURCES OF EVIDENCE

The method presented above uses a single source of information as input data, i.e. a single 2D image, to evaluate the most probable poses of the object. However, it is sometimes desirable to use several sources of information to disambiguate the result, or make it more precise. Such extra information could be available, e.g. as multiple images of the same scene, observed under different viewpoints, or as several types of image features, extracted from one same image. This section proposes a rigorous method for fusing the results produced by each different cue, thereby determining globally consistent poses. The method is presented in the concrete context of multiple views, but it directly extends to other scenarios, e.g. with multiple types of image features.

We represent by the random variable $W \in SE(3)$ the pose of the object, and by $X_i \in \mathcal{A}$ the distribution of observations in the i th view. The dependency between these random variables can be represented by a pairwise Markov random field [24], [17], organized in a tree structure, W being the root node (see Fig. 3). The *compatibility potential functions* parameterizing the relationship between W and a X_i are called ψ_i . These are identical to the ψ introduced in Section II, apart from now taking into account the actual viewpoint of the corresponding view. Each node X_i is moreover connected to its corresponding observed variable, Y_i , their relationship being parameterized by ϕ_i , defined similarly to the ϕ of Section II. To determine the marginal density of the top node W , inference on such a graphical model can be performed using Non-parametric Belief Propagation (NBP), as proposed in [24]. The application of the NBP algorithm on a model as simple as that considered here allows many simplifications. In particular, the distribution of W is simply given by

$$p(w) = \prod_{i=1}^q m_i(w), \quad (9)$$

with a *message* $m_i(w)$, conceptually sent from a node X_i to the root node W (see Fig. 3), and expressing its *belief* about the state of W , being defined as

$$m_i(w) = \int_{\mathcal{A}} \psi_i(w, x) \phi_i(x) dx. \quad (10)$$

Note that this definition of $m_i(w)$ is identical to Eq. 6, but is now indexed on the source of the observations. Practically, each $m_i(w)$ can be independently evaluated, using the method of Section II-C. This method returns a set of poses in the most dense regions of $SE(3)$, which can directly be used to represent the distribution $m_i(w)$ in a non-parametric fashion, using KDE, weighting each of them with its evaluated probability density. Fusing the results from all sources of evidence, via Eq. 9, then amounts to computing the product, or intersection, of all of these non-parametric representations of densities on $SE(3)$. In practice, the representation of each $m_i(w)$ is usually quite compact, and the evaluation of $p(w)$ for a given w can thus be performed at a reasonable computational cost. We therefore identify the maxima of $p(w)$ with a Markov

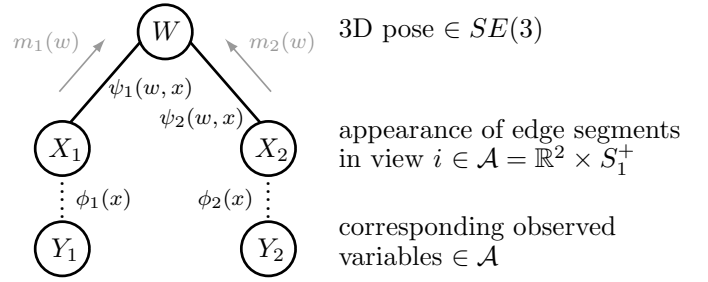


Fig. 3. Markov tree representing the integration of multiple source of evidence, in this case 2 views from which edge segments are extracted. The messages $m_i(w)$ represent the belief about the state of W sent from each node X_i ; they are fused to determine the values of W globally consistent with the two views.

Chain Monte Carlo (MCMC) type search, using a simple random walk scheme [18]. This local optimization process is performed from several different starting points, selected from the supporting particles of the $m_i(w)$. Using this process, the output of the algorithm is finally the set of poses corresponding to the local maxima of $p(w)$ as defined by Eq. 9, i.e. the poses globally the most consistent with all available sources of evidence.

V. EVALUATION

The evaluation of a pose estimation method is not trivial, due to the difficulty of obtaining the ground truth 3D poses themselves, especially in realistic scenes. We considered using various existing datasets, as reviewed below, but finally decided to produce new datasets, with synthetic images and thus known ground truth, which allowed performing a rigorous quantitative evaluation. Practically, the image features were extracted using the well-known Canny edge detector, followed by a smoothing and subsampling step to reduce the noise in the observations (Fig. 1b). All images used were 640×480 pixel grayscale images, and all the parameters of the algorithm were set to identical values for all the tests (with both synthetic and real images).

Among candidates public datasets, we considered the *ETHZ Shape dataset* [15], which features shape-based object classes in various cluttered scenes. It is however specifically targeted at class recognition algorithms, designed to handle variations in shape, as opposed to our method, which actually uses those slight changes in appearance as clues for estimating the 3D pose of the object. The dataset does not include any suitable training data or any ground truth for 3D pose. The *NORB dataset* [25] is made up of images of toy objects in different poses, and of artificial compositions of such images proposed as cluttered scenes. In addition to being evaluated only with class recognition methods (as far as we are aware), the very-low-resolution images prevent any reliable use of edge features, as our method requires. The *RGB-D dataset* [26] is made up of household objects on a turntable, viewed at 3 different elevations, thus in a fairly limited range of poses. We also argue that the basic evaluation methodology proposed for those sequences, which is basically to use every

other image for training and test alternatively, in the absence of clutter and object translations, is overly simplistic and of limited diagnostic value. The capture setup (e.g. constant-speed turntable) is also acknowledgedly imprecise and ruled out this dataset as an interesting candidate for a rigorous evaluation.

A. Quantitative evaluation on synthetic images

The synthetic datasets were produced with manually designed 3D models and rendered with ray tracing software. The training examples (Fig. 1c) correspond to different views of the object of interest on a uniform background; the poses of the object in the training set are chosen uniformly in the orientation space. The amount of clutter in a test image is measured as the ratio of the number of observations *not* belonging to the object of interest over the total number of observations in the image. For example, a clutter ratio of 0% corresponds to absence of clutter, whereas a clutter ratio of 80% means that about 4/5 of the observations are actually noise. We measure the success rate as the ratio of experiments that returned a *correct* pose in the first k results (the algorithm returns a list of poses sorted by decreasing probability density). This aspect is important, as the ambiguities the 2D input data often prevent one from distinguishing between different 3D poses that have very similar appearances on the image plane. The threshold for considering a pose as *correct* was set in accordance to the typical dimensions of a scene: considering our objects are of a size of 100–200 mm and distant from the camera of 1000–2000 mm, this threshold was set to a translation error of 20 mm parallel to the image plane (XY), 100 mm in depth (Z), and a maximum rotation error of 20° . The greater tolerance on the Z translation is justified by the fact that the use of a single 2D image makes the determination of depth very difficult. Note however that this error threshold remains a small fraction of the actual depth of the scenes. Using these conventions, the success rates of the algorithm for various conditions are reported in Fig. 5a. Please also note that relaxing the threshold discussed above does not necessarily lead to better quantitative results, as we also report, in Fig. 5c, the mean error of the first *correct* result returned by each run of the algorithm. The reported average numbers were computed over 30 runs of the algorithm for each of the 6 objects considered (Fig. 5b), each scene being generated at random, with clutter made up of different objects disposed randomly in the background. The measure of the error in orientation for the cylindrically symmetric objects (e.g. the bottle) naturally takes only their relevant degrees of freedom into account.

Systematic test cases including occlusions are hard to design, as the amount of occlusion is difficult to quantify: masking one half or the other of an object can have dramatically different effects due to different levels of detail. We are however confident in the ability of the system to cope with significant occlusion, since this is actually simulated by a common large fraction of missing observations (Fig. 4), due to background clutter preventing a good extraction of edges.

The algorithm presents very good success rates under common amounts of clutter (Fig. 5a). This success rate even remains acceptable as the amount clutter is raised to very challenging values (Fig. 4). Increasing the number of training views for each object was not found to have a significant impact on the success rate, but increased the accuracy of the results (Fig. 5c). Similarly, the amount of clutter did not have a significant influence on the precision of the results (Fig. 5c), but only makes harder the identification of the modes of the distribution. In general, the erroneous results can be attributed to two sources (see Fig. 4, last row). First, the edge segments we restrict ourselves to cannot always be extracted consistently. For example, in an image of the kettle, if the edges of the handle are extracted on one of its sides but not on the other, this side may be “matched” with any of the two sides of a training view, potentially leading to a large error on the orientation of the recovered pose – despite both being globally good matches with the 2D input view. Second, using the 2D projections of any 3D object introduces inevitable ambiguities. For example, it may be very difficult to differentiate between a cylindrical object pointing away and towards the camera (Fig. 4, bottom left); this effect is particularly true for our objects consisting of mostly homogeneous surfaces.

We used a similar protocol to evaluate the use of multiple views of a same scene, as proposed in Section IV. In those experiments, we used, instead of a single 2D image, 2 images of the scene from viewpoints spaced by 45° . Such a wide baseline is generally too large to be handled by traditional stereo methods, and thus demonstrates one of the interests of our approach. The success rate was generally not noticeably affected by the use of two views over one, but the error was almost always substantially decreased, as reported in Fig. 5c. Using a second view helps the algorithm disambiguating between the different possible orientations of the object, and also provides much better clues for determining the actual depth of the scene (Z translation).

B. Real test images

The method was evaluated on real test images. For practical reasons, we relied here again on computer-generated images as training data. We used 128 training views of each object, that were produced as explained above (Fig. 1c), through ray tracing with manually-designed 3D models. In a realistic application, such images are to be acquired, e.g. by a robotic agent taking pictures of the real object under various viewpoints [27]. This alternative option was chosen purely for practical reasons, but added an additional challenge as the models used for generating the training images inevitably did not match the real objects perfectly. The test images were taken with a handheld camera at about 1000–2000 mm from the scenes.

We performed many experiments on typical household scenes with common objects. We purposely chose objects presenting large homogeneous surfaces with little texture and details, on which classical feature-based pose estimation would likely fail. We present, in Fig. 6, typical results of both successful and failed experiments. As the ground truth

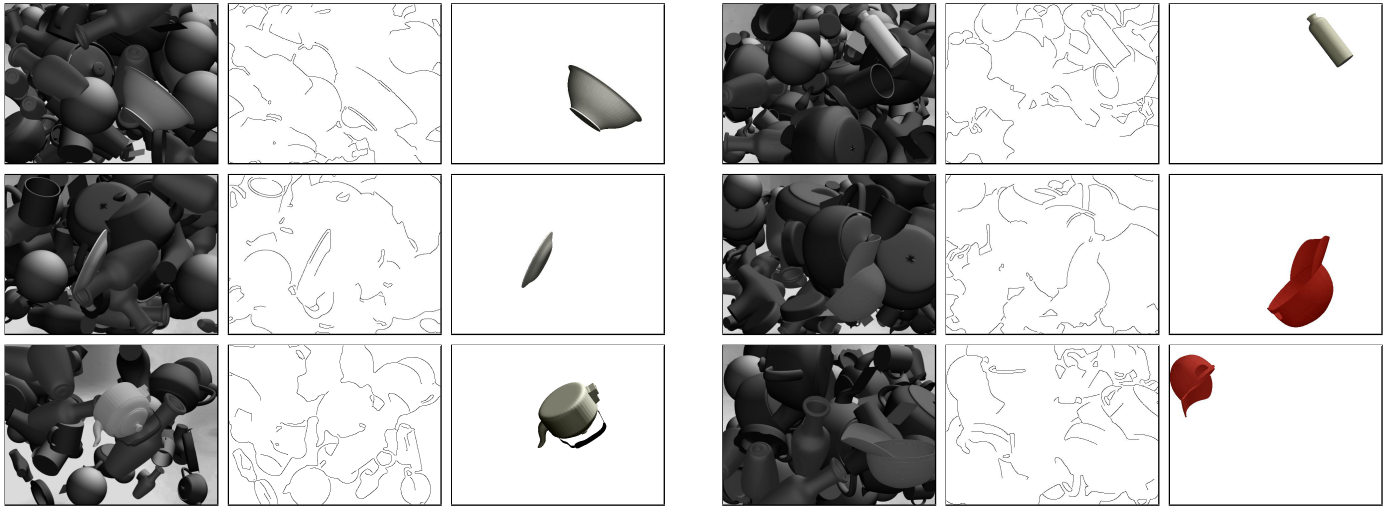


Fig. 4. Sample results of our quantitative evaluation (for each: test image, edge map used as input, rendering of object model in the first pose proposed by the algorithm); these tests used a single test view, 128 training views per object, and clutter=80%. The last row shows typical incorrect results: although a close match is found with the given edges, the 3D pose is incorrect.

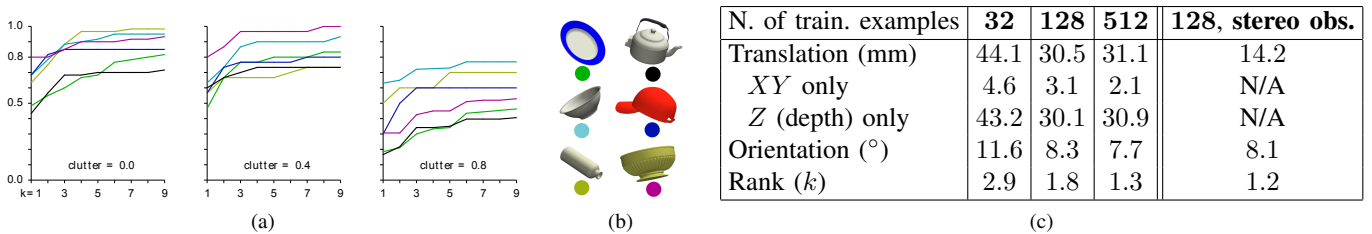


Fig. 5. Quantitative results on synthetic images (see text for details); (a) for each object, success rate of having a correct result among the first k ones (128 training examples), in scenes of no/medium/heavy clutter; (b) test objects used; (c) average error of first correct result.



Fig. 6. Sample results on real images (similar conditions as Fig. 4); for visualization, we render, in yellow, the outline of artificial models set in the first pose found. The last two images show common failures, typically due to uncertainty in the limited input data used (edges): the mitten identified in background clutter, and the rim of the plate matched onto its shadow.

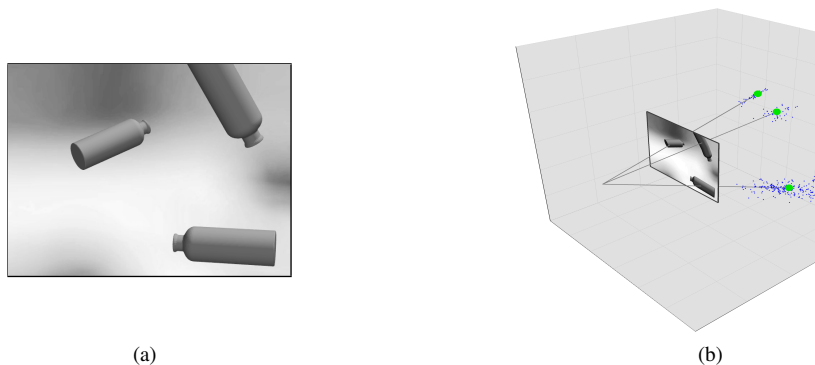


Fig. 7. Recovery of probability distribution of 3D pose; (a) input image; (b) plot of 3D position as a non-parametric description (blue points), and local maxima (green points). Each occurrence of the object in the image correctly generates one mode in the distribution.

pose is not available, measuring the errors is not possible. Instead, we visualize the results by rendering, onto the input images, synthetic models of the objects in the poses found by the algorithm. One can observe good matches with the input images, demonstrating the good use made of the 2D information available. As discussed before, the use of 2D observations, especially edge segments alone, often makes it hard to distinguish between different poses that may appear similar in one image. The first pose returned by the algorithm may thus correspond to an erroneous result, but the correct result will often be found in the other poses proposed by the algorithm (identified with slightly lower probability). The actual disambiguation is thus to be left to the end application, which may best make use of this uncertainty in the results.

C. Retrieval of full pose distribution

One key capability that we propose is to recover a *distribution* of 3D poses, rather than a single optimum. We illustrate this in Fig. 7: the pose of a bottle is evaluated in an image containing several occurrences of the object. The distribution is recovered in a non-parametric fashion as a collection of particles, of which we plot the 3D position. One mode is correctly identified for each occurrence of the object, the main uncertainty remaining unsurprisingly in the depth dimension, extending along the camera projection axis.

VI. CONCLUSIONS AND FUTURE WORK

We presented a novel method for exemplar-based pose estimation in single images. Relying on a general, probabilistic formulation of the problem, the method avoids establishing specific correspondences between training and test views, thus allowing similar-looking types of images features. The pose of the object is treated as a probability density over the 3D pose space, from which we identify the different modes, thereby accounting for the ambiguities of 2D input data. We also proposed an elegant way of fusing evidence from multiple sources, such as several views of the same scene, or different types of image features. A first validation of the overall approach showed promising results. Further developments will mainly focus on the use of other types of image features within this framework, extending its applicability further to more types of scenes, objects and imaging conditions.

Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience. Damien Teney is supported by a research fellowship of the Belgian National Fund for Scientific Research.

REFERENCES

- [1] A. Collet and S. S. Srinivasa, "Efficient multi-view object recognition and full pose estimation," in *IEEE Int. Conf. on Rob. and Autom.*, 2010, pp. 2050–2055.
- [2] I. Gordon and D. G. Lowe, "What and where: 3D object recognition with accurate pose," in *Toward Category-Level Object Recognition*, 2006, pp. 67–82.
- [3] D. Teney and J. Piater, "Probabilistic Object Models for Pose Estimation in 2D Images," in *DAGM*, ser. LNCS, vol. 6835. Springer, 2011, pp. 336–345.
- [4] S. Edelmann and H. Bülthoff, "Modeling human visual object recognition," in *Int. Joint Conf. on Neural Networks*, 1992, pp. 37–42.
- [5] S. Ekvall, F. Hoffmann, and D. Kragic, "Object recognition and pose estimation for robotic manipulation using color cooccurrence histograms," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2003.
- [6] P. Mittrapiyanuruk, G. N. DeSouza, and A. C. Kak, "Calculating the 3D-pose of rigid-objects using active appearance models," in *IEEE Int. Conf. on Rob. and Autom.*, 2004, pp. 5147–5152.
- [7] A. R. Pope and D. G. Lowe, "Probabilistic models of appearance for 3D object recognition," 2000.
- [8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comp. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
- [9] R. Soderberg, K. Nordberg, and G. Granlund, "An invariant and compact representation for unrestricted pose estimation," in *Patt. Rec. and Im. Anal.*, J. Marques, N. Prez de la Blanca, and P. Pina, Eds. Springer, 2005, vol. 3522, pp. 489–500.
- [10] F. Vikstén, P.-E. Forssén, B. Johansson, and A. Moe, "Comparison of local image descriptors for full 6 degree-of-freedom pose estimation," in *IEEE Int. Conf. on Rob. and Autom.*, 2009, pp. 2779–2786.
- [11] G. Borgefors, "Hierarchical chamfer matching: A parametric edge matching algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, no. 6, pp. 849–865, 1988.
- [12] D. P. Huttenlocher, G. A. Klanderman, and W. A. Rucklidge, "Comparing images using the hausdorff distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 9, pp. 850–863, 1993.
- [13] M.-Y. Liu, O. Tuzel, A. Veeraraghavan, and R. Chellappa, "Fast directional chamfer matching," in *Conf. on Comp. Vis. and Patt. Rec.*, 2010, pp. 1696–1703.
- [14] C. F. Olson and D. P. Huttenlocher, "Automatic target recognition by matching oriented edge pixels," *IEEE Trans. Im. Proc.*, pp. 103–113, 1997.
- [15] V. Ferrari, T. Tuytelaars, and L. Van Gool, "Object detection by contour segment networks," in *European Conf. on Comp. Vis.*, 2006.
- [16] V. Ferrari, F. Jurie, and C. Schmid, "From images to shape models for object detection," *Int. J. Comp. Vis.*, vol. 87, no. 3, pp. 284–303, 2010.
- [17] R. Detry, N. Pugeault, and J. Piater, "A probabilistic framework for 3D visual object representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1790–1803, 2009.
- [18] R. Detry and J. Piater, "Continuous surface-point distributions for 3D object pose estimation and recognition," in *Asian Conf. on Comp. Vis.*, 2010.
- [19] A. Toshev, A. Makadia, and K. Daniilidis, "Shape-based object recognition in videos using 3d synthetic object models," in *Conf. on Comp. Vis. and Patt. Rec.*, 2009, pp. 288–295.
- [20] F. Viksten, R. Soderberg, K. Nordberg, and C. Perwass, "Increasing pose estimation performance using multi-cue integration," in *IEEE Int. Conf. on Rob. and Autom.*, 2006, pp. 3760–3767.
- [21] R. Caflisch, "Monte carlo and quasi-monte carlo methods," *Acta Numerica*, vol. 7, pp. 1–49, 1998.
- [22] R. C. Nelson and A. Selinger, "Large-scale tests of a keyed, appearance-based 3D object recognition system," *Vis. Res.*, vol. 38, pp. 38–15, 1998.
- [23] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 790–799, 1995.
- [24] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky, "Nonparametric belief propagation," in *Comp. Vis. and Patt. Rec.*, 2003.
- [25] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Conf. on Comp. Vis. and Patt. Rec.*, 2004, pp. 97–104.
- [26] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *IEEE Int. Conf. on Rob. and Autom.*, 2011.
- [27] K. Welke, J. Issac, D. Schiebener, T. Asfour, and R. Dillmann, "Autonomous acquisition of visual multi-view object representations for object recognition on a humanoid robot," in *IEEE Int. Conf. on Rob. and Autom.*, 2010, pp. 2012–2019.

Sampling-based Multiview Reconstruction without Correspondences for 3D Edges

Damien Teney
University of Liège, Belgium
Damien.Teney@ulg.ac.be

Justus Piater
University of Innsbruck, Austria
Justus.Piater@uibk.ac.at

Abstract—This paper introduces a novel method for feature-based 3D reconstruction using multiple calibrated 2D views. We use a probabilistic formulation of the problem in the 3D, reconstructed space that allows using features that cannot be matched one-to-one, or which cannot be precisely located, such as points along edges. The reconstructed scene, modelled as a probability distribution in the 3D space, is defined as the intersection of all reconstructions compatible with each available view. We introduce a method based on importance sampling to retrieve individual samples from that distribution, as well as an iterative method to identify contiguous regions of high density. This allows the reconstruction of continuous 3D curves compatible with all the given input views, without establishing specific correspondences and without relying on connectivity in the input images, while accounting for uncertainty in the input observations, due e.g. to noisy images and poorly calibrated cameras. The technical formulation is attractive in its flexibility and genericity. The implemented system, evaluated on several very different publicly-available datasets, shows results competitive with existing methods, effectively dealing with arbitrary numbers of views, wide baselines and imprecise camera calibrations.

I. INTRODUCTION AND RELATED WORK

The problem of 3D scene reconstruction using multiple 2D images from different viewpoints is fundamental in computer vision. The variety of applications, from robotic interaction to phototourism or reverse engineering, has led to the development of numerous methods over the years. These can be broadly classified into two categories: (i) intensity-based multiview stereo methods, which produce *dense* surface reconstructions, and (ii) feature-based methods, which recover *sparse* 3D models of geometric features. Although many of these methods have proven successful in select fields of application, their typical requirements and limitations in operating conditions motivated the development a novel, feature-based method, particularly suited to the use of hard-to-match features. This method, which we successfully applied to the particular problem of 3D curve reconstruction, will be introduced after reviewing related literature.

Methods of the first category mentioned above typically aim at producing detailed 3D reconstructions of objects, enforcing photometric consistency and surface continuity constraints to recover a dense shape description. However,

those methods can typically only operate in precisely controlled settings, usually only with Lambertian surfaces, and with large numbers of precisely calibrated cameras. Those typical requirements for controlled acquisition conditions often prove impractical for general applications (see [1] for a review). While dense reconstructions can offer visually striking results, there are many applications where sparse reconstructions are sufficient, as argued below.

Methods of the second category aim at reconstructing sparse 3D models, made up of isolated geometric features, such as points or edges. Such methods are particularly interesting as they provide more expressive and efficient representations than dense surfaces, typically at a fraction of the computational cost. The classical methods rely on the detection of interest points in the individual 2D views, and then use their local appearance (e.g. using SIFT descriptors [2]) to propose likely matches between observations from different views. The geometric consistency between pairs or triples of points can then be enforced using the well-known epipolar or trifocal constraints [3], effectively leading to the reconstruction of a 3D point cloud compatible with the observations. The first limitations of this approach are obviously those of the extraction and matching of image features, which works best on texture-rich images, but can perform poorly on scenes with mostly homogeneous surfaces or little detail [4]. Moreover, the matching of local appearance descriptors is made harder as the baseline between the considered viewpoints increases [5], practically limiting this approach to the consideration of close pairs of views at a time.

Other methods of the second category make use of image curves, or edges, extracted in the available 2D views [4], [6]–[9]. Reconstructions made up of edge segments convey more geometric information than point clouds [6] and offer greater invariance to changes in illumination and viewpoint. Edge-based reconstructions have moreover proved directly useful for practical applications like pose estimation [10], [11], or the prediction of grasping points of objects [12]. The classical approach, described above, of matching observations between different views (now lines or curves) is however a non-trivial problem [6], exacerbated by the variability in the extraction of said edges from the 2D images. Li *et al.* [4]

reviewed various schemes, e.g. using extended projective geometry [13] or differential geometry [4], or restricting the problem to closed curves [14]. Common drawbacks are strong requirements for precisely calibrated camera [4], [9], [13] and limitations to pairs or triples of views at a time [13]. In [15], Kaess *et al.* focuses on the subproblem of fitting parametric curves to contours identified in several images, using a Monte Carlo-type search as we do. They do not however consider the reconstruction of entire scenes with several objects and the inevitable uncertainty in the input observations. Kahl *et al.* [7] present an approach that also avoids establishing correspondences between views, but delivers results only on simple scenes, reconstructing only small numbers of short curve fragments. We present results on arguably more challenging datasets and in much more varied conditions (see Section IV).

Multiview reconstruction is part of the larger problem of simultaneous localization and mapping (SLAM). In contrast to SLAM, this paper assumes calibrated views and does not make use of core assumptions made by most SLAM methods, most importantly the abundance of input views and feature tracking across views. Some SLAM methods are nevertheless relevant to the current discussion. Klein *et al.* [16] use edges as image features and show how complementary they are to interest points. They focus on the localization problem, and do not deliver convincing results for reconstruction of said edges. Civera *et al.* [17] propose, as we will do, an alternative probabilistic formulation to the classical Gaussian measurement uncertainty, but also focus on localization. [18] goes beyond precisely localizable features by tracking surface patches under photometric constraints to provide a dense reconstruction, but is based on frame-to-frame tracking.

The method proposed in this paper aims at reconstructing a sparse 3D model of geometric features. The key principle is the definition, using each available 2D view, of a probability density in the 3D reconstructed space, which is *compatible* with the view considered. This distribution thus encompasses all backprojected 3D features that could have produced the considered image. Considering all available views, the intersection, or product, of those distributions is then proposed as the distribution of 3D features of the reconstructed scene. We present in Section II an efficient algorithm for obtaining individual samples from that distribution, effectively yielding a set of 3D features (edge fragments in our implementation) describing the reconstructed scene. A second algorithm is proposed that iteratively identifies contiguous regions of high density in the 3D space, which links such samples together, forming continuous 3D curves.

The strength of the proposed approach is to handle non-precisely localizable features, which cannot be matched one-to-one, or which present uncertainty in some dimension of the observation (like a point along an edge). The resulting curve reconstruction method therefore does not rely on

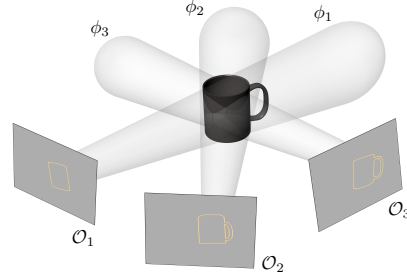


Figure 1. The proposed method uses the observations of each input view, \mathcal{O}_k , to define probability distributions ϕ_k in the 3D, reconstructed space; the reconstructed model lies at the intersection of those distributions.

connectivity in input images, effectively accounting for the variability in the extraction of edges from the images. Other reconstruction methods have been designed to handle uncertainty in the input data, often by relaxing the matching and geometry constraints. For example, Fabbri *et al.* [6] implemented a two stage process, where an initial robust reconstruction is used to optimize the calibration of the cameras, to then obtain a finer reconstruction in the second stage. That approach, which can be traced back to the classical RANSAC algorithm, proved robust, but, in addition to being arguably computationally inefficient, lacks the genericity and flexibility of the formulation presented below. Note finally that similar probabilistic models of objects and image observations have been used in the past [10], [11], and this work can be seen as their extension to the problem of 3D reconstruction.

We must finally remark that reconstruction without correspondences is not new. A basic formulation of the problem was presented in [19]. In [20], Dellaert *et al.* used expectation-maximization to recover the structure of a scene, handling however only precisely localized features, and only presented results on toy examples under several unrealistic assumptions. More recently, [21] showed how to recover the camera transformation between pairs of views using the radon transform, but without considering the 3D structure of the scene at all.

II. PROBABILISTIC RECONSTRUCTION FROM 2D VIEWS

We now present the proposed method, first in a general formulation, then applied to the use of edge segments. Those features correspond, in the input images, to points extracted along lines of maximum gradient, and characterized by their position and orientation on the image plane (see Section III-A). In the reconstructed model, they correspond to oriented 3D points, that we typically represent by short, fixed length, 3D line segments (see Fig. 4b for example); they can be connected together to form continuous curves (e.g. 4c).

A. Probability distributions from image observations

The key idea of the method is to define, from each available 2D view, a probability density over the reconstructed 3D space, which is *compatible* with the observations in that view (1). In other words, it describes the distribution of backprojected 3D features that could have produced the considered image, given the uncertainty present in that image, and in the available estimation of the camera parameters. Formally, each view $k \in [1, N]$ is described by a set of image features, or *observations*

$$\mathcal{O}_k = \{y_i\}_{i \in [1, M_k]}, \quad (1)$$

where $y_i \in \mathbb{R}^2 \times \mathcal{A}$ are the image features, characterized by their position in the image, and some descriptor in an appearance space \mathcal{A} . In the case of edge segments, which have an orientation but no direction, the appearance descriptor is an element on the semicircle (i.e. an angle in $[0, \pi]$), and $\mathcal{A} = S_1^+$. Considering instead more classical interest points, described by their position in the image and their local appearance, the space \mathcal{A} would then contain normalized texture descriptors. The 3D, reconstructed model, is to be defined on a corresponding space $\mathbb{R}^3 \times \mathcal{A}'$. With edge segments, then characterized by a 3D orientation, we have $\mathcal{A}' = S_2^+$.

We will now define, for a view k , a probability distribution ϕ_k on the reconstructed space, using kernel density estimation (KDE). Each element y_i of the considered view is associated with an element of the reconstructed 3D space, $y'_i \in \mathbb{R}^3 \times \mathcal{A}'$. This element can simply be obtained by setting a normalized value for the extra dimensions; e.g., the depth and 3D orientation of our edge segments can be fixed to lie on the image plane in the 3D world (see Fig. 2). This now allows us, using KDE, to define the distribution ϕ_k by its probability density function

$$\phi_k(x) = \frac{1}{M_k} \sum_{i=1}^{M_k} K_i(y'_i, x), \quad (2)$$

where K_i are kernel functions on $\mathbb{R}^3 \times \mathcal{A}'$. Intuitively, one kernel $K_i(y'_i, x)$ models the distribution of all reconstructed features that could have produced the observation y_i . The details, which will depend on the type of features used, are straightforward in the case of edge segments. Looking at the position only, it represents a constant probability density along the backprojected ray (see Fig. 2). Formally, we measure the distance between a given y'_i and x by 3 scalars:

- i. d_1 , the closest distance in position between x and the line defined by y'_i (the backprojected ray),
- ii. d_2 , the depth of x , relative to the camera center,
- iii. d_3 , the difference in orientation between x , and the plane corresponding to the backprojection of the orientation of y'_i .

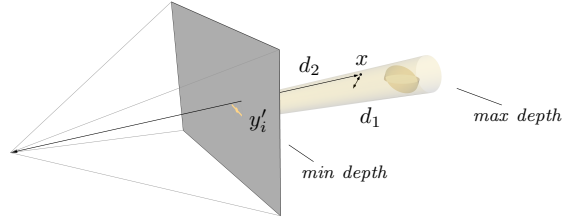


Figure 2. Illustration of an observation y'_i (an oriented point on the image plane) and its associated kernel $K_i(y'_i, \cdot)$ in the reconstructed, 3D space, both for the position and the orientation (surfaces of equidensity in transparent orange). The kernel, evaluated at a point x , uses the distances d_1 and d_2 , resp. to the axis of the backprojected cone and to the camera center (see text for details); d_3 is not represented.

We then define our kernel function $K_i(y'_i, x)$ as the product of 3 independent kernels that make use of those distance measures: a Gaussian kernel on (d_1/d_2) (inducing a conical surface of equidensity for the position, see Fig. 2), a box kernel on d_2 , and a von Mises-Fisher kernel on d_3 (which is a Gaussian-like distribution on orientations [22]). Note that the effect of the box kernel on the depth only corresponds to fixing a hard threshold on the distance to the reconstruction. Indeed, the only assumption that can generally be made here is that a reconstructed point must lie in front of the camera, and within a realistic depth range.

The geometric meaning of our definition of a kernel is quite intuitive, and is illustrated in Fig. 2. For example, the surfaces of equidensity for the position in the 3D space correspond to truncated cones, extending along the camera’s projection rays. The selection of the bandwidth of the kernels is discussed in Section III-B.

The definition of the kernels could be extended to other types of image features, or to include edge curvature for example. We propose another minor extension that takes into account the uncertainty along the orientation of an edge, thereby “flattening” the cone of Fig. 2. For this purpose, the distance d_1 is separated in 2 components d'_1 and d''_1 , respectively aligned and orthogonal to the orientation of the edge; they are then simply evaluated as (d'_1/d_2) and (d''_1/d_2) in Gaussian kernels of respectively large and small variance, thus allowing more slack along the orientation of the edge (see specific results in Section IV-A).

Finally, as a side note, let us remark that defining a probability distribution over the *reconstructed* space, as we did, differs from the classical formulation of the problem, where the reconstructed model is compared, once reprojected in the image space, against the 2D input observations. We will remark that, under certain parameterizations, the two approaches can be rendered equivalent. Our formulation was however chosen in this presentation, as it offers a more intuitive formulation of the sampling-based reconstruction methods that we will propose below.

B. 3D Reconstruction of individual points

The probability distributions ϕ_k we have defined make each use of one single view. We now combine them to produce another distribution ψ in the reconstructed space that is globally consistent with all available views. It is given by its probability density function

$$\psi(x) = \frac{1}{C} \prod_{k=1}^N (\phi_k(x) + \varepsilon), \quad (3)$$

where C is a normalization constant, and ε is a fudge constant, small relative to the scale of $\phi_k(x)$. This definition practically uses the intersection of the ϕ_k , relaxed by the constant ε . This allows observations that appear in some but not *all* input images to produce a nonzero density region in the reconstructed space. This proves necessary in practice, to handle e.g. self-occlusions and missing observations.

Equation (3) gives a formal definition of the 3D reconstruction of the scene. The main goal however is to obtain an explicit and practical representation of this model. Sampling directly from ψ is generally not feasible, but we propose an approximate method based on importance sampling (see for example [10], [23]). Importance sampling (IS) allows one to sample a target distribution $p(x)$, assuming one can evaluate $p(x) = \bar{p}(x)/Z$ up to some normalization constant Z , by using samples x^ℓ from a *proposal distribution* p' , ideally similar to p . IS accounts for the difference between the target and proposal distributions by assigning to each sample x^ℓ a weight given by

$$w^\ell = \bar{p}(x^\ell) / p'(x^\ell). \quad (4)$$

The collection of weighted samples $\{(x^\ell, w^\ell)\}_{\ell=1}^L$ is then, under mild assumptions, asymptotically consistent with the target distribution. This procedure is obviously most efficient as the proposal distribution is close to the target distribution. In practice, the collection of weighted samples is then generally resampled, to a smaller set of $L' (< L)$ *unweighted* samples.

The proposal function used here is given by

$$\psi'(x) = \frac{1}{C'} \sum_{\substack{(k_1, k_2) \\ \in \text{pairs}(1, N)}} \phi'_{k_1}(x) \phi'_{k_2}(x), \quad (5)$$

where C' is a normalization constant, and $\text{pairs}(1, N)$ denotes the list of all unique pairs of indices between 1 and N . Each density function ϕ'_k is a variation of the ϕ_k defined above, in which the kernels used are all box functions. Intuitively, ψ' simply corresponds to all the intersections of pairs of views. Sampling from $\psi'(x)$ is easily done by choosing two arbitrary views k_1 and k_2 , and triangulating two random observations y_1 and y_2 from each, the kernels of which intersect at least by a small amount (i.e. the 3D projections of which intersect each other within a small threshold). The bandwidth of the box kernels of ϕ' will

be chosen so that they extend up to a reasonable cutoff threshold of the exact kernels of ϕ . This ensures that the proposal distribution ψ' will generate samples in all of the most interesting regions of the target distribution ψ . The weights assigned to the proposal samples of ψ' are then simply computed using (4). They can then be resampled to obtain a set of non-weighted points.

C. 3D Reconstruction of continuous curves

The method presented above reconstructs individual points as samples from a probability distribution in the 3D space. Some interesting parts of the scene may however correspond to regions of lower density (e.g. due to missing observations in one or several views), but which can however still be identified as local maxima. Moreover, in the particular case of curve reconstruction, one wants to reconstruct continuous curves, and not individual points. Those two objectives can be met through the iterative procedure described below, which uses the individual samples as starting points for a stochastic exploration of the reconstructed space.

For each reconstructed curve, the procedure starts with a sample $x_0 \in \mathbb{R}^3 \times S_2^+$. It then iterates, searching at each step for a point x_{i+1} along a ridge of locally maximum probability density. Formally, local proposals are generated from a point $x = (p, \theta)$ of position $p \in \mathbb{R}^3$ and orientation $\theta \in S_2^+$ (a unit 3-vector), as a set of L samples:

$$\text{proposals}(x) = \left\{ (p + \Theta_\kappa(\theta) * \Gamma_{(\alpha, \beta)}, \Theta_\kappa(\theta))_j \right\}_{j \in [1, L]}, \quad (6)$$

where Γ is a gamma distribution that generates the distance in position to a proposal, and Θ is a Von Mises-Fisher distribution used to randomize the orientation. This uses the assumption that the next point of the curve is most likely in the direction of the current point. The parameters κ, α, β define how “spread out” the proposals are from an exactly straight line. The likelihood of each proposal is evaluated (Eq. 3), and the best one is selected as the new point x_{i+1} of the curve. The procedure is repeated, unless the likelihood of all L proposals fall below a threshold, indicating the probable end of the curve. That threshold is fixed beforehand as fraction of the mean density of a batch of samples of the whole scene. The procedure is comparable to the classical Canny algorithm, which, likewise, follows ridges of local optima until falling below a predefined threshold. Note that the use of a purely random walk scheme for selecting the neighbours in our method — as opposed to estimating local derivatives of a likelihood function (as could be done using differential geometry) — is motivated by the genericity of the procedure, which we plan to apply to other types of image features as future work. Finally, as the scene is being reconstructed, we “prune” ψ , removing the kernels that have significant overlap with the curves already reconstructed. This helps reconstructing parts of the scene of low probability density, initially masked out by regions of

higher density, and also avoids reconstructing several times the same portions of a scene.

III. IMPLEMENTATION

A. Edge detection in input images

The image features we use are oriented 2D points, identified along the edges in the images. We selected the method of [24], which is a simple method based on image gradients that extracts the orientation of the edges significantly better than the traditional method, which simply uses the direction orthogonal to the gradient. That method was chosen instead of more sophisticated ones which take texture or global segmentation into account, as they can be extremely slow and are thus not an option for many applications of 3D reconstruction. This also ensured a fair comparison with other published methods which used basic gradient-based edges as well.

B. Choice of parameters of the reconstruction

The kernels associated with the observations are parametrized by their bandwidth in position and orientation. This size should reflect the estimated uncertainty in the input data, and can be set according to a small fraction to the estimated scale of the scene. Our experiments showed however that the method was not particularly sensitive to the choice of those parameters. For example, in the experiments (with both small and large camera calibration errors) of Section IV-D, with 640×480 pixel images, the size of the kernels was set to allow a corresponding maximum deviation in the images of about 12 pixels and 20° .

The parameters used for local proposals (Eq. 6) are also to be set relatively to the scale of the scene. For example, the scene of Section IV-D, measuring about 1000 mm in diameter, used local proposals corresponding to a spacing of 5 mm and a deviation in orientation of 15° on average, with $L = 50$.

Finally, the running time of the iterative procedure grows linearly with the number of reconstructed points. The cost associated with the reconstruction of a point mostly corresponds to the evaluation of ψ (Equation 3) for the proposals. One evaluation involves the processing of every kernel of every input view, and is thus $O(N\bar{M})$, where N is the number of views, and \bar{M} the average number of observations per view. We currently use this basic implementation. However, a cleverer implementation could efficiently preselect the few kernels likely to be relevant to the evaluation of a given point, using an ordered data structure. Since the influence of a kernel in its distribution drops below insignificant values past some distance, one could, in this way, restrict the evaluation of the kernels to a small fraction of them.

IV. EXPERIMENTAL RESULTS

The proposed method was evaluated on 4 very different datasets. It is notoriously hard to produce ground truth

reconstructions for evaluating feature-based methods, due to the ambiguous selection of the features to reconstruct. Datasets for benchmarking dense reconstruction methods have been produced; however, the ground truth model is not necessarily made public [1], and the selection of actual edges from continuous surfaces [25] or 2.5D models makes it hard to design a meaningful quantitative evaluation of a method like ours. Competing methods for curve reconstruction faced a similar situation, which explains why no extensive qualitative evaluations were published. [6] made an exception, but they only evaluate their ability to match correct curve fragments between views, using a set of manually labelled ground truth correspondences — which was unfortunately not made public.

Practically, our prototype software was implemented in Matlab. Running times of such an implementation (especially of an iterative method) have little meaning, as the only switch to a compiled language offers potentially enormous room for improvement. Bearing this in mind, we report, as a base point, that a reconstruction as shown in Fig. 4 or Fig. 6a currently takes about 2 to 5 minutes on a standard laptop without multithreading. Most recent competing methods do not discuss the issue of efficiency; Fabbri *et al.* [6] report running times in the order of minutes on scenes like the dinosaur (see below). Let us note moreover that most parts of our algorithm are straightforward to parallelize.

A. Synthetic toy example

The lack of datasets with proper ground truth motivated the use of a synthetic toy example, in order to evaluate and demonstrate basic properties of the proposed method. The scene, pictured in Fig. 3a, contains curves of various lengths and shapes. Their exact 3D shape is used to directly generate the 2D edge maps used as input to the reconstruction method. This bypasses the stage of edge extraction from 2D images, focusing this evaluation on the reconstruction process alone. To simulate realistic conditions and missing observations, random parts of the curves are masked when generating those edge maps. The scene itself measures about 500 mm in diameter; we use 7 views from different viewpoints around the scene, at a distance of approximately 900 mm.

We compare reconstructions and ground truth using the accuracy/completeness metrics proposed in [1]. To obtain accuracy, we measure the Euclidean distance from each reconstructed point to the closest ground truth curve. The accuracy is then defined as the distance so that 90% of the points fall below that threshold. To obtain the completeness, we consider a number of points sampled uniformly along the ground truth curves, and count the ratio of them that have a part of the reconstruction within a reasonable distance (15° in orientation, and $5/8$ mm in position for scenes without/with noise). The exact choice of those thresholds is not relevant here, since we use them to compare different

methods and not to obtain absolute performance values. We report accuracy/completeness scores for 4 different reconstruction methods: (i) a baseline method where we perform random triangulations (Eq. 5), and keep a fixed number (1000) of points with a probability density (computed as in Eq. 3, but without orientation) above a threshold; this corresponds approximately to the basic approach where one simply imposes a maximum 2D distance between the re-projected reconstruction and the input observations; (ii) our sampling method (Section II-B) used to recover the same number (1000) of points; (iii) our iterative method for curve reconstruction (Section II-C); (iv) the same method accounting for uncertainty specifically along the orientation of edges (Section II-A).

Each method is run with different lower thresholds on the probability density of reconstructed points, setting the tradeoff to be made between accuracy and completeness. We report results in Fig. 3b, with and without noise on camera calibrations (in the form of added Gaussian perturbations of $\sigma = 4$ mm on the camera positions). We also plot, in Fig. 3c, the accuracy and the local probability density (Eq. 3) of a number of random samples (Eq. 5). This allows verifying that there is indeed a correlation between the probability density obtained through our definition, and the actual correctness of a reconstructed point. Reconstructions showing good accuracy can however sometimes correspond to low probability densities, which explains why our sampling method alone cannot recover the entire scenes, as opposed to the iterative method. Moreover, we also verify that the correlation between accuracy and probability density still holds when adding noise (as above) to the camera calibrations.

B. Dinosaur

The “dinosaur” dataset is standard for the evaluation of dense reconstruction methods [1]; we use the version made of 16 views from a circle around the object. We show in Fig. 4b a reconstruction made of individual points, obtained using our sampling method. These samples are drawn mostly in the regions of high probability density of the reconstructed space, with more samples in the regions the most precisely defined, e.g. along the crest on the back of the animal (such sharp edges correspond to well-defined edges in the 2D images). In Fig. IV-Bc, we show a reconstruction of continuous curves of the same scene; those curves are correctly identified along ridges of local maxima of the probability density function, yielding a high quality reconstruction of the object. Those results, directly comparable with those presented in [6], show a clear advantage, particularly in the level of noise in the reconstruction.

C. High-resolution building

Strecha *et al.* [25] produced a dataset of high resolution pictures of buildings for evaluating dense reconstruction

methods. We chose to evaluate our method on one of those scenes (“Herz-Jesu-P8”) as it represents a very different type of input data than our other evaluations. The images are of high resolution, but the nature of the scene (very textured surfaces and lots of fine details) renders the extraction of stable edges from the 2D images a difficult problem already. The reconstructed 3D model (Fig. 5a-b) exhibits missing parts, which are a direct consequence of this problem (corresponding to missing observations in the input data). The 8 viewpoints span only a small arc, roughly in the same plane, leaving a great deal of uncertainty in the *depth* dimension, in particular for the edges parallel to that plane. This can be observed when viewing the reconstructed model from the top (Fig. 5b), as some supposedly straight edges meander in this dimension. The same curves however, when re-projected on an input image, always closely match the input images (Fig. 5c). [25] uses a particular distance measure to evaluate dense reconstruction methods, requiring non-public information (calibration uncertainty), which prevented direct performance comparisons.

D. Office desk

We finally consider an indoor scene, containing typical household items with little texture (see Fig. 6a), shot from 12 different viewpoints around them. This represents the type of scenes that motivated our approach, in the context of robotic applications, where a robot would take the pictures using an arm-mounted camera. The extrinsic calibration of the camera would thus be known with a precision corresponding to the accuracy of the robotic arm. In this evaluation however, and for purely practical reasons, we used a checkerboard pattern in the scene with standard calibration software. We obtained visually excellent reconstructions (Fig. 6). A challenging part of the scene is the checkerboard, as it contains many lines close together, in both similar and different orientations. We then intentionally added noise to the positions of the 12 cameras, to verify the influence on the reconstruction (see Fig. 6b-e for details). The highest tested level of additional noise, drawn from a Gaussian distribution of $\sigma = 8$ mm, introduces corresponding translation errors as large as 10 pixels on the 640×480 pixel images. Experiments show that a reconstruction is still possible; some regions of the reconstructed space now receive a probability density lower than the allowed threshold, explaining missing parts in the reconstruction. Those results are representative of the performance we obtained on many experiments of similar nature.

V. CONCLUSIONS AND FUTURE WORK

We presented a novel method for feature-based 3D reconstruction from multiple calibrated views. We introduced a probabilistic formulation that admits hard-to-match features particularly suited to edge segments. The reconstructed scene is modelled as a probability density in the 3D space,

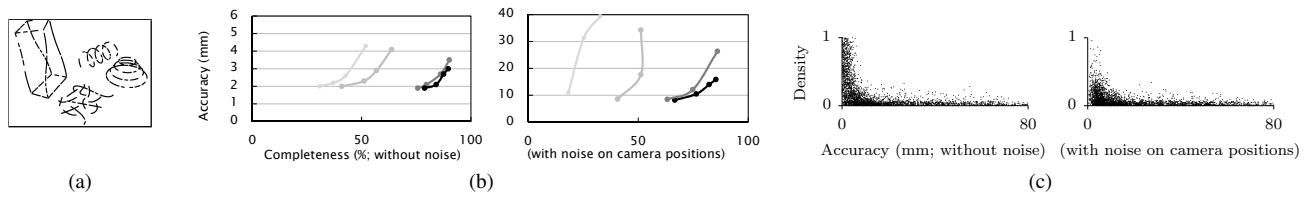


Figure 3. Evaluation on synthetic scene: (a) example input edge map, note missing observations; (b) accuracy/completeness scores for (light to dark) random sampling based on position only, our sampling method using orientation, our iterative method with conical kernels, then with “flattened” kernels (accounting for uncertainty along the edge orientations); (c) density/accuracy of random samples (density evaluated up to a multiplicative constant).

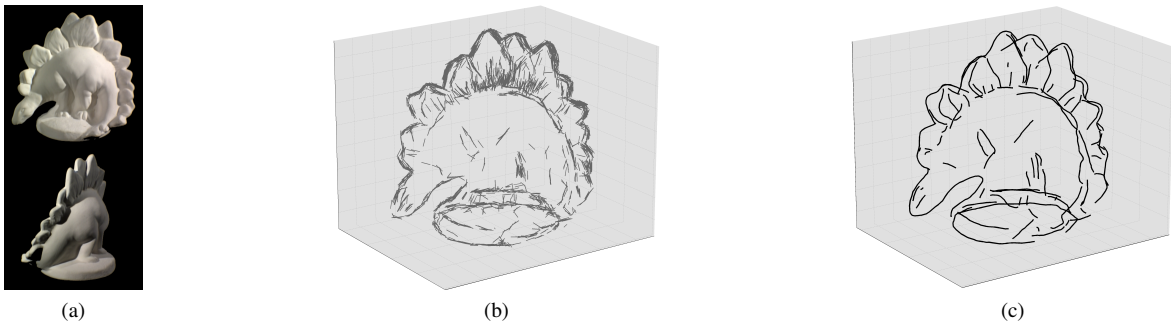


Figure 4. (a) Example images of the dinosaur dataset; (b) individual reconstructed 3D points obtained through our sampling method; (c) reconstruction of continuous curves.

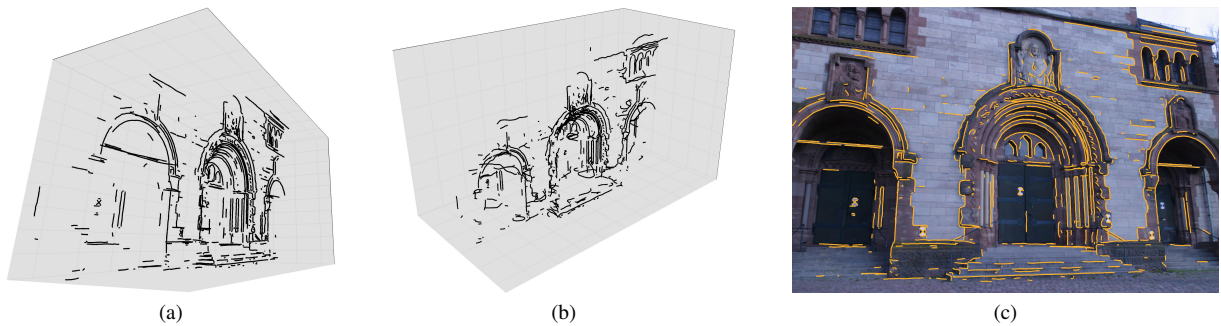


Figure 5. (a) Reconstruction of the building dataset, missing parts are mostly due to missing observations, difficult to extract from the input images; (b) other view of the reconstruction, showing the imprecisions in depth, as the input viewpoints span only a small arc in front of the building; (c) reconstructed edges, reprojected on an input image, match however closely.

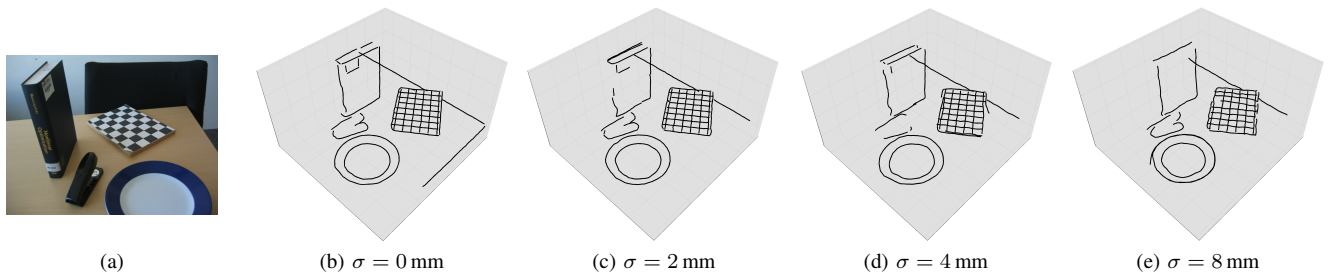


Figure 6. Reconstruction of scene with error in camera calibration; one input image (a); 3D reconstructions (rendered from a novel viewpoint) with original estimated camera calibration (b) and with added perturbation on camera position from Gaussian noise of variance σ (c-e); significant levels of error still allow reconstruction, at the price of some imprecisions (plate, checker board) and missing edges (book, lower edge of the table).

from which we can draw individual samples. Those are then used as starting points to reconstruct continuous 3D curves. The effectiveness of the approach was demonstrated on existing and new datasets, and showed competitive results with an existing method, while exhibiting more technical flexibility and genericity in its formulation. An important direction for future work is the evaluation of this method on features other than edges.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience. Damien Teney is supported by a research fellowship of the Belgian National Fund for Scientific Research.

REFERENCES

- [1] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2006, pp. 519–528. [1](#), [5](#), [6](#)
- [2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004. [1](#)
- [3] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004. [1](#)
- [4] G. Li, Y. Gene, and S. Zucker, "Multi-view edge-based stereo by incorporating spatial coherence," in *3-D Digital Imaging and Modeling, 2007, Sixth International Conference on*, 2007, pp. 341–348. [1](#), [2](#)
- [5] P. Moreels and P. Perona, "Evaluation of features detectors and descriptors based on 3d objects," *International Journal of Computer Vision*, vol. 73, pp. 263–284, 2007. [1](#)
- [6] R. Fabbri and B. Kimia, "3D curve sketch: Flexible curve-based stereo reconstruction and calibration," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 1538–1545. [1](#), [2](#), [5](#), [6](#)
- [7] F. Kahl and J. August, "Multiview reconstruction of space curves," in *IEEE International Conference on Computer Vision (ICCV)*, 2003, pp. 1017–1024 vol.2. [1](#), [2](#)
- [8] S. Liu, K. Kang, J.-P. Tarel, and D. B. Cooper, "Free-form object reconstruction from silhouettes, occluding edges and texture edges: A unified and robust operator based on duality," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 1, pp. 131–146, 2008. [1](#)
- [9] K. Potsch and A. Pinz, "3D geometric shape modeling by '3D contour cloud' reconstruction from stereo videos," in *Computer Vision Winter Workshop 2001*, Mitterberg, Austria, 2001. [1](#), [2](#)
- [10] R. Detry, N. Pugeault, and J. Piater, "A probabilistic framework for 3D visual object representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, pp. 1790–1803, 2009. [1](#), [2](#), [4](#)
- [11] D. Teney and J. Piater, "Probabilistic Object Models for Pose Estimation in 2D Images," in *DAGM*, ser. LNCS, vol. 6835/2011. Heidelberg: Springer, 2011, pp. 336–345. [1](#), [2](#)
- [12] M. Popović, D. Kraft, L. Bodenhagen, E. Başeski, N. Pugeault, D. Kragic, T. Asfour, and N. Krüger, "A strategy for grasping unknown objects based on co-planarity and colour information," *Robotics and Autonomous Systems*, 2010. [1](#)
- [13] C. Schmid and A. Zisserman, "The geometry and matching of lines and curves over multiple views," *International Journal of Computer Vision*, vol. 40, no. 3, pp. 199–233, 2000. [2](#)
- [14] R. Berthilsson, K. Astrom, and A. Heyden, "Reconstruction of general curves, using factorization and bundle adjustment," *International Journal of Computer Vision*, vol. 41, pp. 171–182, 2001. [2](#)
- [15] M. Kaess, R. Zboinski, and F. Dellaert, "Mcmc-based multi-view reconstruction of piecewise smooth subdivision curves with a variable number of control points," in *European Conference on Computer Vision (ECCV)*. Springer, 2004, pp. 329–341. [2](#)
- [16] G. Klein and D. Murray, "Improving the agility of keyframe-based slam," in *European Conference on Computer Vision (ECCV)*, 2008. [2](#)
- [17] J. Civera, A. J. Davison, and J. M. M. Montiel, "Unified inverse depth parametrization for monocular slam," in *Robotics: Science and Systems*, 2006. [2](#)
- [18] R. Newcombe, S. Lovegrove, and A. Davison, "DTAM: Dense Tracking and Mapping in Real-Time," in *IEEE International Conference on Computer Vision (ICCV)*, 2011. [2](#)
- [19] Y. qing Cheng, R. T. Collins, A. R. Hanson, and E. M. Riseman, "Triangulation without correspondences," in *DARPA Image Understanding Workshop*, 1994, pp. 993–1000. [2](#)
- [20] F. Dellaert, S. Seitz, C. Thorpe, and S. Thrun, "Structure from motion without correspondence," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000. [2](#)
- [21] A. Makadia, C. Geyer, and K. Daniilidis, "Correspondence-free structure from motion," *International Journal of Computer Vision*, vol. 75, no. 3, pp. 311–327, 2007. [2](#)
- [22] R. A. Fisher, "Dispersion on a sphere," in *Proc. Roy. Soc. London Ser. A.*, 1953. [3](#)
- [23] E. B. Sudderth, "Graphical models for visual object recognition and tracking," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, 2006. [4](#)
- [24] A. Tamrakar and B. B. Kimia, "No grouping left behind: From edges to curve fragments," in *IEEE International Conference on Computer Vision (ICCV)*, 2007, pp. 1–8. [5](#)

- [25] C. Strecha, W. von Hansen, L. J. V. Gool, P. Fua, and U. Thoennessen, "On benchmarking camera calibration and multi-view stereo for high resolution imagery," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 5, 6

Continuous Pose Estimation in 2D Images at Instance and Category Levels

Damien Teney
 University of Liège, Belgium
 Damien.Teney@ULg.ac.be

Justus Piater
 University of Innsbruck, Austria
 Justus.Piater@UIBK.ac.at

Abstract—We present a general method for tackling the related problems of pose estimation of known object instances and object categories. By representing the training images as a probability distribution over the joint appearance/pose space, the method is naturally suitable for modeling the appearance of a single instance of an object, or of diverse instances of the same category. The training data is weighted and forms a generative model, the weights being based on the informative power of each image feature for specific poses. Pose inference is performed through probabilistic voting in pose space, which is intrinsically robust to clutter and occlusions, and which we render tractable by treating separately the least interdependent dimensions. The scalability of category-level models is ensured during training by clustering the available image features in the joint appearance/pose space. Finally, we show how to first efficiently use a category-model, then possibly recognize a particular trained instance to refine the pose estimate using the corresponding instance-specific model. Our implementation uses edge points as image features, and was tested on several existing datasets. We obtain results on par with or superior to state-of-the-art methods, on both instance- and category-level problems, including for generalization to unseen instances.

I. INTRODUCTION AND RELATED WORK

The problem we focus on is the localization and the estimation of the precise 3D pose of objects in a new scene, given a single image of that scene, and multiple images of the objects as training examples. This is a central problem in computer vision, and there exists a wealth of literature on the topic, especially when dealing with specific object *instances*, e.g. a particular car or a particular coffee mug. The classical methods rely on the use discriminative image features and descriptors (such as SIFT or Geometric Blur), matched between the test view and the training examples. Such features are sometimes stored together with a rigid explicit 3D model of the object [1], [2], which brings viewpoint-invariance to the model. Other techniques have been proposed to encode viewpoint-invariant models, especially in the context of object recognition, e.g. by linking the observed features across different viewpoints [3], [4], [5], or modeling the object as a collection of planar parts [4]. Those methods however were used mainly with the goal or *localizing* and *recognizing* those objects in the images, but without recovering their 3D pose explicitly. One exception is the work of Savarese *et al.* [4], but the recovered pose is only a rough identification, such as “frontal view” or “side view”. This limitation is present in many other methods [6], [7], [4], [8] which use discretized pose values, treated

as separate classes, with different classifiers tuned to each of them. There exist however methods, often presented in the robotics community (with applications such as object grasping in mind), which can provide accurate pose estimates [9], [10], but they are mostly limited to specific object instances.

One particular aspect we are interested in is to provide the capability for pose estimation at the *category* level. There is an increased interest for this more challenging task; the goal is for example to train the system with a set of different mugs, then to recognize the pose of a new, unseen mug. The categories in such a scenario are defined implicitly by the training instances used as examples.

Previous work on object recognition does acknowledge the close link between handling the variability of object appearance as a function of pose and due to the diversity of objects within a category. Gu and Ren [11] showed how to solve for instance and *discrete* (coarse) pose recognition at the same time. Lai *et al.* [12] did so as well, using a tree of classifiers tuned for the different tasks. However, they use presegmented views of the objects, without any clutter or occlusions, and provide modest results on the accuracy of the retrieved pose. The methods mentioned in the previous paragraphs, while modeling the change of appearance due to different viewpoints, generally cannot directly handle the variability within *categories* of objects [3], [5]. One way this capability has been provided is by encoding — in addition to a rough 3D model — the possible variations in appearance [13], [14]; one limitation however is that no shape variability is possible. Our model, on the contrary, is purely appearance-based, and naturally accommodates variability in shape as well as in appearance. The traditional models of rigid geometrical constraints and highly discriminative features [2] are not adequate for encoding within-category variations. One exception to most methods here is again the model of Savarese *et al.* [4], which is specifically designed to provide viewpoint-invariance while handling within-category differences — but still provides only coarse pose estimates.

Recently, some methods have been introduced that can handle category variability and perform localization together with *precise* pose estimation. Glasner *et al.* [15] uses structure-from-motion to reconstruct accurate 3D models from the training images. They then account for within-category variability simply by merging multiple exemplars in their non-parametric model, in a fashion very similar to us. They perform pose infer-

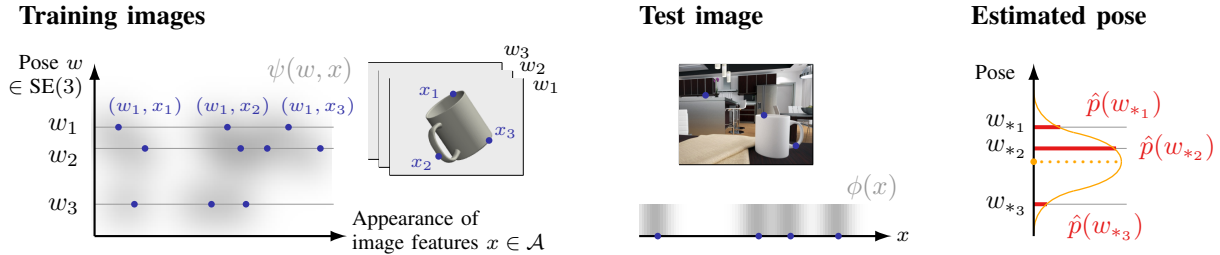


Fig. 1: Proposed method for representing training/test data and for pose estimation. Images features (blue points) are extracted from training and images; their appearance descriptor (in the case of our edge points, a position and orientation in the image) is defined on the generic space \mathcal{A} . Training/test observations define, using KDE, continuous probability distributions, respectively ψ and ϕ (gray shaded areas). Our pose inference algorithm (Fig. 2) returns approximations (red bars) of the pose likelihood function $p(w)$ at some discrete poses w_{*i} . Finally, we locally fit, on those approximations, a simple distribution in the pose space (orange curve), and keep its mean as our final, precise pose estimate (orange dot).

ence through probabilistic voting in the 6D pose space, again in a similar way as we do, thereby solving for localization and viewpoint identification together. However, the reconstruction of such dense 3D models relies on the initial availability of a large number of views. By contrast, the appearance-based model used in this paper can use an arbitrary number of views and can be incrementally updated as more views become available. In a very different approach, Torki and Elgammal [16] learn a regression from local image features to the pose of the object. They recover a precise pose, but cannot handle significant clutter or occlusions, and the accurate pose estimation depends on the (supervised) enforcement of a one-dimensional manifold constraint (corresponding to the 1D rotation of the object in the training examples). It is not clear how that approach would extend to the estimation of the full 3D pose of an object. On the contrary, our method is framed from the start in the context of the full 3D pose.

Our method can accommodate different types of image features, but we chose to use very basic points along edges (combined with their tangent orientation) as opposed to more elaborate features such as SIFT descriptors. Recognition by matching such descriptors, while easier with specific instances, does not easily extend well to object categories. We differ from most edge-based shape recognition methods (e.g. [17], among many others) by avoiding intermediate representations such as contour fragments, and leveraging the simplicity of low-level features — in our implementation, simple points along edges. These simple features provide invariance to viewpoint and to within-category changes of appearance. Using such non-discriminative features for recognition however raises an additional challenge, since no matching is possible. This motivated the use of the framework proposed by Teney and Piater [18] for pose estimation in 2D images, which does not rely on correspondences between the test and the training data. Like [4], this model is generative and does not include any discriminative aspects, but has however been shown to be useful for localization and recognition in the presence of heavy clutter and occlusions [18]. Compared to that work, (1) we use a more efficient method for pose inference that does not need

to consider the whole 6D pose space at once, (2) we introduce a weighting scheme of the training features which, as we will show, enhances significantly the performance of the system, and (3) we extend the methodology from instance-specific to category-level models.

The capabilities of the approach proposed in this paper differ from existing work by (1) handling, within the same framework, *instance*-specific models and *category*-specific models of objects, in the latter case allowing variations in shape and appearance, (2) performing *continuous* (precise) 3D pose estimation using those models, as opposed to viewpoint classification and coarse pose estimates, and (3) using such models to solve pose estimation *and* image localization together, as opposed to competing methods that do not handle clutter or occlusions. In addition, we present how to use category- and instance-level models successively, for optimal accuracy and efficiency: the category-model is used first to recover an initial pose estimate, which then allows one to possibly recognize a particular trained instance, so that the corresponding instance-specific model can be used to refine the pose estimate. Finally, in Section IV, the performance of our approach is compared to the most closely related methods [7], [4], [16]; we obtained promising results, on par with or superior to published data.

II. POSE ESTIMATION OF SPECIFIC OBJECT INSTANCES

A. Probabilistic representation of input data

The method we use is based on a probabilistic representation of both the training and the test data. This approach can be seen as a smoothing over the available data, providing continuous distributions of features and interpolating, to some extent, between the available data points (see Fig. 1, left and middle). Practically, the training examples are a set of K images of the object to learn, each annotated with the 3D pose of the object, $w_k \in \text{SE}(3)$ with $k = 1, \dots, K$. We extract, from each training image, features x_i , which are edge points (see Section IV) with their tangent orientation, and which are thus defined on $\mathbb{R}^2 \times S_1^+$ (accounting for the position in the image, plus an orientation without direction). In the general case, we will call this space the *appearance* space, \mathcal{A} . We then

pair all features x_i of a view k with the pose w_k , so that we obtain a set of *pose/appearance pairs* $(x_i, w_k)_i$. Considering the whole training set, the pairs from all example images are concatenated to form our full training set $\mathcal{T} = \{(w_i, x_i)\}_{i=1}^M$, with $x_i \in \mathcal{A}$, and $w_i \in \text{SE}(3)$.

The elements of our training set are then simply used to define a continuous probability distribution ψ on the pose/appearance space, in a non-parametric manner, with kernel density estimation:

$$\psi(w, x) = \frac{1}{M} \sum_{(w_i, x_i) \in \mathcal{T}} K_1(w, w_i) K_2(x, x_i), \quad (1)$$

where $w \in \text{SE}(3)$ and $x \in \mathcal{A}$. The kernel functions $K_1(\cdot, \cdot)$ and $K_2(\cdot, \cdot)$ handle respectively the pose and the appearance spaces. Details on suitable kernels can be found, e.g. in [18], [19]; the first is an isotropic kernel allowing small deviations in both position and orientation, and the second, similarly, allows small variations in the location in the image and tangent orientation of the image feature.

The test data, which is a single 2D image of a new scene, is handled in a similar fashion as the training data. We extract the same type of image features, which we store as a set of *observations* $\mathcal{O} = \{x_i\}_{i=1}^N$, where $x_i \in \mathcal{A}$. This set is then used to define the continuous probability density ϕ on \mathcal{A} :

$$\phi(x) = \frac{1}{N} \sum_{x_i \in \mathcal{O}} K_2(x, x_i). \quad (2)$$

As noted in [18], the transformations in the pose/appearance space corresponding to in-plane rotations/translations/scale changes are known from the camera calibration; those trivial transformations (e.g. a change in depth corresponds to a change of scale) are thus hard-coded. This allows us, when using ψ as a generative model, to extend its definition to parts of the pose space not explicitly covered by the training data.

B. Pose inference

The pose of the object of interest in the test scene is modeled as random variable $W \in \text{SE}(3)$, the distribution of which is given by the likelihood function

$$p(w) = \int_{\mathcal{A}} \psi(w, x) \phi(x) dx, \quad (3)$$

This expression simply measures the compatibility of the training data at a pose w , with the distribution of features observed in the test image. The objective is to identify the main modes and peaks of the distribution of W , which was accomplished in [18] by a probabilistic voting scheme on the 6D pose space. This procedure is extremely costly in memory and processing [15], [18] due to the high dimensionality of the pose space. We now propose an approximation of that method that handles different dimensions of the pose space in different ways. Formally, a pose $w \in \text{SE}(3)$ can be decomposed as a concatenation of 3 simpler entities, such that $w = w^3 \circ w^2 \circ w^1$. The first, w^1 , corresponds to the “viewpoint”, i.e. which side of the object is facing the camera; w^2 is a combination of an in-plane rotation and scale change, and w^3 corresponds to a pure

Input: training pairs $\mathcal{T} = \{(w_i, x_i)\}_i$ defining ψ
test observations $\mathcal{O} = \{x_i\}_i$ defining ϕ
Output: set \mathcal{R} of approximations of the pose likelihood function
 $\mathcal{R} = \{(w_{*i}, \hat{p}(w_{*i}))\}_i$

Procedure:

$\mathcal{R} \leftarrow \emptyset$

For each discrete w^1 in \mathcal{T} (viewpoint)

For each discrete step of w^2 (in-plane rotation and scale)

Considering pose $w' = w^2 \circ w^1$,

find best w^3 (image translation) between $\psi(w', x)$ and $\phi(x)$:

Get samples: $(w_i^\psi, x_i^\psi) \sim \psi(w', x)$

$x_j^\phi \sim \phi(x)$

Each possible pairing (x_i^ψ, x_j^ϕ) cast a vote in space of w^3 of weight $\text{wt}(w_i^\psi, x_j^\phi)$

Keep highest density peak in vote space: w_*^3 of vote score s

$\mathcal{R} \leftarrow \mathcal{R} \cup (w_*, s)$ with $w_* = w_*^3 \circ w^2 \circ w^1$

Fig. 2: Pose inference algorithm

translation parallel to the image plane. The main supporting observation for our proposed method is that a significant peak in the distribution of W will most likely appear as a peak in the distribution corresponding to the dimensions of w^3 alone. Indeed, an object of the test scene in any specific pose w will appear at a *precisely defined* image location (dimensions of w^3). This leads to the algorithm presented in Fig. 2, which iterates over discretized values for the dimensions of w^1 and w^2 , and uses probabilistic voting only on the dimensions of w^3 (the 2D localization in the image). The peaks in those last two dimensions are thus identified by the algorithm for discrete viewpoints, scale and in-plane rotation values. This formulation is reminiscent of the classical Hough voting scheme used extensively for object localization [20]. The main advantage over [15], [18] is to avoid considering the entire pose space at once.

We also propose an additional step for refining the pose estimate, beyond the precision of the discretized pose values. As illustrated in Fig. 1 (right), we use the peaks identified by the algorithm in the pose space, together with their score value, as approximations of the likelihood function $p(w)$ (Eq. 3) at some discrete “probing” points. We simplistically assume that the main modes in the underlying distribution of W must *locally* approximate a simple isotropic distribution in the pose space. We therefore locally fit such a distribution (isotropic Gaussian and von Mises-Fisher distributions [19]) on the main peaks of $p(w)$, using non-linear least squares. The mean of the fitted distribution is then retained as the peak of that particular mode of the distribution (Fig. 1, right). This provides a much more accurate estimate of the optimal pose(s) compared to the above algorithm (as demonstrated in Section IV-A), at a very small additional computational cost.

C. Weighting of training data

We now present a way of weighting the available training data. The model we use does not include any discriminative

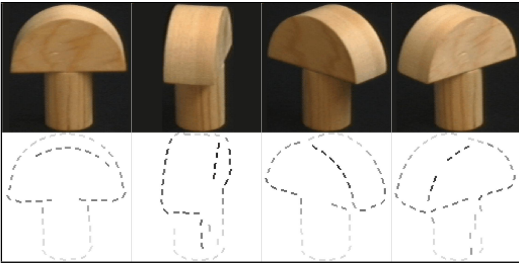


Fig. 3: Visualization of the weights attributed to each image feature (edge fragments) on a toy example; darker colors correspond to heavier weights. The parts looking similar in different views (e.g. the cylindrical base) receive lower weights, while the image features that can unambiguously determine a precise pose (e.g. non-silhouette edges) receive high weights.

aspects per se, and this weighting proved to significantly enhance the overall performance of the method (see Section IV). Appropriately weighting training data in the context of object recognition was previously shown to increase performance e.g. in [21], [22], [23], [24]. The formulation proposed here is different, suited to our non-discriminative low-level image features, and does not rely on massive amounts of training examples. The idea is to weight each image feature, depending on how informative it actually is for determining a specific pose. As detailed in the algorithm of Fig. 2, a training feature (w, x) is allowed to cast a vote of weight $\text{wt}(w, x)$, given by

$$\text{wt}(w, x) = 1 - \left[\frac{1}{K} \sum_{w':(w', \cdot) \in \mathcal{T}} \psi'(w', x)(1 - K'_1(w, w')) \right] \quad (4)$$

with ψ' and K'_1 being variants of ψ and K_1 with maximum values of 1. This definition yields numerically-convenient weights in the range $[0, 1]$.

In Eq. 4, the expression in square brackets measures, for an image feature x observed in a training pose w , how likely this feature would be in poses very different than w . The weight is then defined using the opposite of that value. This effectively corresponds to the *specificity* of that feature x for the pose w (see also Fig. 3).

III. LEARNING OBJECT CATEGORY MODELS

The model and methods presented above naturally extend to *category-level* models. In that case, the training images include different objects, which together implicitly define the category. This capability of our model is due both to the fact that we can use very simple, non-discriminative image features (points along edges), which often generalize well across different objects of a same category, and by the non-parametric representation of the training data, which can naturally handle variability in the training data, in this case coming from several object instances.

Formally, each object instance $\ell \in [1, L]$ used for training produces a training set \mathcal{T}_ℓ , as defined in Section II-A. A

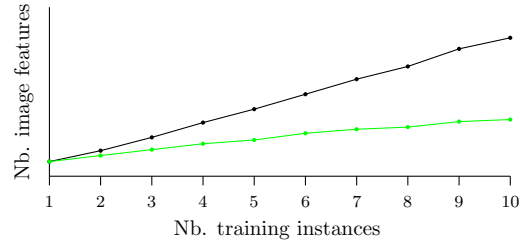


Fig. 4: Size of the category-level model of rotating cars built using different numbers of training instances: without (black) and with (green) the pruning of features by clustering. The proposed approach ensures a sublinear growth of the model.

category-level model is then simply created using all features of all example instances, $\mathcal{T} = \bigcup_\ell \mathcal{T}_\ell$.

A. Pruning of training features by clustering

The above formulation uses a linearly growing number of training points (pose/appearance pairs) as more object instances are used to learn a given category. This correspondingly increases the computational requirements of using the model. Fortunately, object instances within a category often share common appearance traits, and the elements of \mathcal{T} can thus be pruned at a very small cost of the representative capabilities of the model (as shown in Section IV). Practically, the elements of \mathcal{T} are grouped using a simple agglomerative clustering scheme on the joint pose/appearance space, and only the cluster centers are retained. A maximum variance is enforced within the clusters, both in pose and appearance, which determines the amount of discarded training points. Note that the clustering procedure is most efficiently performed after normalizing the training examples from different instances for in-plane translation, rotation and scale, using the hardcoded transformations mentioned in Section II-A.

B. Recognizing a particular trained instance

The clustering of training features limits the size of a category model for efficiency. To compensate for lost accuracy, after identifying an initial pose estimate w_* with this category model, one can determine whether the recognized object corresponds to a specific trained instance. We measure the score of each trained instance ℓ at the pose w_* with

$$p_\ell(w_*) = \int_{\mathcal{A}} \psi_\ell(w_*, x) \phi(x) dx, \quad (5)$$

where ψ_ℓ is defined as in Eq. 1, but using only the elements \mathcal{T}_ℓ of the instance ℓ . The value is easily approximated [18] with

$$p_\ell(w_*) \approx \frac{1}{n} \sum_i^n \psi_\ell(w_*, x_i) \quad \text{where } x_i \sim \phi(x). \quad (6)$$

If the value of $p_\ell(w_*)$ is significantly higher for a certain ℓ , the corresponding model of that instance ℓ (using all training data available for that instance) is then used to obtain a new, more accurate pose estimate (Section IV-A).

IV. EXPERIMENTAL EVALUATION


We now evaluate the proposed method under various conditions, using publicly-available datasets. We first analyze the incremental improvements in performance due to the individual ideas proposed in this paper. We then compare our results to existing, competing methods. The image features used are simple points identified along image edges, extracted with the classical Canny detector (see the examples in Fig. 3). Each of those points is characterized by its position in the image, and by the local orientation (smoothed for stability) of the edge at that point (an angle in $[0, \pi]$). As a ballpark figure of efficiency, on a standard laptop, our Matlab implementation of the method takes 20-30 seconds to process an image of the dataset of Section IV-B.

A. COIL Dataset

We first evaluate our method on the classical COIL dataset [25]. This dataset has been used in a variety of contexts, but not in the particular conditions we were interested in. The purpose of this part of our evaluation is to demonstrate the merits of the proposed method, by highlighting the incremental improvements brought by each proposed key point.

We selected a few objects from the original dataset, which correspond to reasonable categories (rectangular boxes, toy cars, flat bottles; see Fig. 5). Most other objects of the dataset were not suitable for estimating their pose (e.g. bell peppers, cylindrical cans) or could not be grouped into categories (e.g. duck toy). The dataset contains 72 images of each object undergoing a full rotation around a single (vertical) axis, with a fixed elevation. The estimated pose is thus similarly limited to this degree of freedom. For training, we use 18 images of each object (thus 20° apart), and the others for testing. We report the error as the median and mean (over all test images) of the absolute error of the estimated orientation. The rectangular boxes and the flat bottles present a 180° rotational symmetry, the error is accordingly evaluated on the half-circle.

1) *Seen instances*: The first series of tests uses 4 instances of each object (2 for the bottles) for training category models, and those same objects for testing. The basic method (algorithm of Fig. 2 without weighting the training data) already provides accurate results (see Fig. 5), with a median error of 5° which is the best achievable for the nearest-neighbour classification of the algorithm (Fig. 2) iterating on the discrete viewpoint values of the training data. The *mean* error decreases as we use the weights on the training data, as a few ambiguous test images are now better classified, which indicates the superior discriminability between different poses when using those weights. Interestingly, the fitting of a distribution on the pose space over the discrete approximations of the likelihood function (Section II-B) reduces the error significantly, as this allows accuracy beyond the resolution of the nearest-neighbour classification mentioned above. Finally, we refine the pose using the procedure proposed in Section III-B: the pose estimate obtained with the category model is used to efficiently check the resemblance with a particular trained instance. If one trained instance receives a significantly higher



	Toy cars		Boxes		Flat bottles	
Seen instances						
Without weights	5.0	12.1	5.0	13.5	5.0	7.9
With weights on training data	5.0	10.1	5.0	11.6	5.0	8.3
Weights + pruning of train. data	5.0	8.7	5.0	11.0	5.0	6.8
Weights + pruning + fitting of dist.	2.9	7.2	3.4	10.2	5.5	6.1
Refined w/ instance-specific model	2.0	5.8	3.2	9.4	4.2	5.3
Unseen instances						
Without weights	10.0	36.8	10.0	14.4	25.0	28.2
With weights on training data	5.0	39.7	10.0	11.0	30.0	31.2
Weights + pruning of train. data	10.0	44.6	10.0	11.8	15.0	25.5
Weights + pruning + fitting of dist.	2.9	41.8	4.3	8.8	16.8	23.6

Fig. 5: Results of category-level pose estimation with objects from the COIL dataset. Image top row: objects used for training and as *seen* test instances; image bottom row: objects used as *unseen* test instances. We report median (black) and mean (gray) error in degrees; large mean error is caused by (near-)symmetries which often induce errors of 90° and 180° .

likelihood than the others (Eq. 6), its corresponding *instance-specific* model is used to perform a (hopefully) more accurate estimation; this is indeed the case as reported in Fig. 5. This procedure thus makes use of both the category- and instance-models for best efficiency without sacrificing accuracy.

2) *Unseen instances*: The second series of tests uses the same category models, but with a test set of other, *unseen* objects (Fig. 5, second row). The purpose is to verify the generalization capability of the category models. The results, as reported in Fig. 5, show accurate pose estimation results in all of the 3 tested categories, even though the test objects vary in shape, appearance and proportions from the training instances. This is made possible by the combination of different appearance traits of different training instances, which is possible in our non-parametric representation of the model. The flat bottles however yielded slightly worse results, which indicate the difficulty of generalizing the appearance of such objects on the category level. A test view of a novel instance could equally correspond to a wide bottle seen from its side, or to a front-facing thin one.

B. Rotating cars

We evaluated our method using the “Multiview car dataset” used by [7] and [16]. It includes about 2000 images of 20 very different rotating cars filmed at a motor show. The dataset is very challenging due to clutter, changing lighting conditions, high within-class variance in appearance, shape and texture, and highly symmetric side views, or similar front and rear views, which are sometimes hard to discriminate even for a human. The dataset was used in [7] for pose classification in 16 discrete bins, and in [16] for continuous pose estimation.

Number of training examples	15	30	40
Baseline comparison: Torki and Elgammal [16]	5.47	1.93	1.84
Without weights	6.75	3.83	2.94
With proposed weights on training data	6.68	3.81	2.91
Weights + fitting of pose distribution	4.42	1.62	1.49

Fig. 6: Results of pose estimation on a single car; mean error in degrees.

We first evaluated our method, as in [16], on the first car of the dataset, using thus an instance-specific model. We select 15, 30 or 40 equally-spaced images of the sequence as training images, and use all other images (spaced about 3–4° apart) for testing. Using all the key techniques proposed in this paper, we obtain superior results to [16] (see Fig. 6 for details). We then performed an evaluation the “10/10 split”, where the first 10 cars of the dataset are used for training, and the other 10 for testing. We obtain again accurate pose estimation results. As highlighted in Fig. 8, most estimated poses are very accurate, while a number have an error of about 180°. This is caused by the symmetric aspects of some cars in the side views, as well as to confusion between front- and rear-facing views. This explains the seemingly large error reported as the mean in Fig. 7, even though the median error is clearly better than the results reported by [16]. In this case, the median as an evaluation metric better reflects the actual precision of the pose estimates, focusing on all the “successful” test cases.

We tested again the generalization capabilities of our model. As proposed in [7], we used the model trained on the cars at the motor show for testing on the database of Savarese *et al.* [4]. The cars appear here in natural environments with more clutter and in very diverse conditions; nevertheless, we obtained interesting results, of which we show some representative examples in Fig. 9. This again demonstrates the good capability of our system to generalize category-level models to conditions very different from those trained for. Note that, unfortunately, no quantitative results for these particular test conditions (proposed in [7]) — that we could compare to — were previously reported.

As a side note, let us mention that we tested our method on this same dataset [4] under the conditions of [8], i.e. training the model with 5 instances of that dataset. We obtained performance on pose estimation of the same order of magnitude as [8], but we missed some information for an exact quantitative comparison (which instances to use for training, and whether or not to include pose estimation results of inaccurate detections). Those experimental conditions were also evaluating coarse pose classification, whereas we focus on continuous pose estimation.

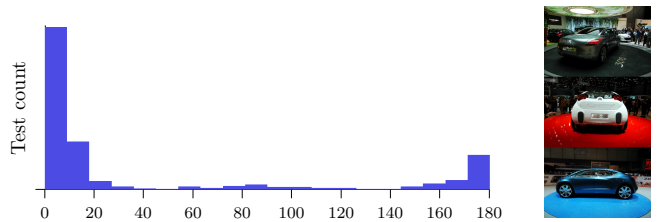


Fig. 8: Histogram showing distribution of error (in degrees) during experiments on multiple cars and sample test images that yielded an error of about 180°, due to ambiguous appearance.



Fig. 9: Detection and pose estimation results on the database of [4], using the model trained with Fig. 7. Boxes indicate the localization of the object as identified by our system, and the roses in the upper-left corners indicate the orientation of the front of the car as seen from the top (as in [7]). The last column contains failure cases, often due to the symmetrical appearance of the cars, or to too much clutter in the background.

V. CONCLUSIONS AND FUTURE WORK

We presented a framework for representing the appearance of object instances or categories, together with its mechanisms to perform object localization and pose estimation in 2D images. The training examples are represented by a probability distribution, stored in a non-parametric manner, in the joint pose/appearance space. This approach can naturally represent a single object, or a whole object category by including different training exemplars of that category. The localization and identification of the pose of the object in a new scene is accomplished via probabilistic voting in the pose space, intrinsically robust to background clutter and occlusions. The overall approach was shown to be competitive or outperform comparable methods. As future work, it will be interesting to evaluate the method in the context of robotic applications,



	Median	Mean 90%ile	Mean	Error<22.5°	Error<45°	Used training features
Baseline comparison: Ozuysal <i>et al.</i> [7]	–	–	46.5	41.7%	71.2%	
Baseline comparison: Torki and Elgammal [16]	11.3	19.4	34.0	70.3%	80.7%	
Without weights on training data	9.3	33.1	47.4	65.1%	70.0%	100%
With weights and fitting of distribution	5.8	23.7	39.0	78.1%	79.7%	100%
Same + moderate pruning of features	6.1	25.8	41.0	77.0%	78.7%	54%
Same + aggressive pruning of features	9.4	32.4	46.8	67.1%	70.0%	30%

Fig. 7: Results of pose estimation on multiple cars; instances 1–10 used for training (top), 11–20 for testing (bottom). Errors of 180° are common (e.g. on instances 16 and 19) and explain the greater mean but smaller median error compared to [16].

with training sets spanning the whole viewing sphere around the objects to learn.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Community’s Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience. Damien Teney is supported by a research fellowship of the Belgian National Fund for Scientific Research.

REFERENCES

[1] V. Ferrari, T. Tuytelaars, and L. J. V. Gool, “Simultaneous object recognition and segmentation from single or multiple model views,” *Int. J. Comp. Vis. (IJCV)*, vol. 67, no. 2, pp. 159–188, 2006. 1

[2] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce, “3D object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints,” *Int. J. Comp. Vis. (IJCV)*, vol. 66, no. 3, pp. 231–259, 2006. 1

[3] A. Kushal, C. Schmid, and J. Ponce, “Flexible object models for category-level 3D object recognition,” in *IEEE Int. Conf. on Comp. Vis. and Patt. Rec. (CVPR)*, 2007. 1

[4] S. Savarese and L. Fei-Fei, “3D generic object categorization, localization and pose estimation,” in *IEEE Int. Conf. on Comp. Vis. (ICCV)*, 2007. 1, 2, 6

[5] A. Thomas, V. Ferrar, B. Leibe, T. Tuytelaars, B. Schiel, and L. Van Gool, “Towards multi-view object class detection,” in *IEEE Int. Conf. on Comp. Vis. and Patt. Rec. (CVPR)*, 2006. 1

[6] J. Liebelt, C. Schmid, and K. Schertler, “Viewpoint-independent object class detection using 3D feature maps,” in *IEEE Int. Conf. on Comp. Vis. and Patt. Rec. (CVPR)*, 2008. 1

[7] M. Ozuysal, V. Lepetit, and P. Fua, “Pose estimation for category specific multiview object localization,” in *IEEE Int. Conf. on Comp. Vis. and Patt. Rec. (CVPR)*, 2009. 1, 2, 5, 6, 7

[8] M. Sun, H. Su, S. Savarese, and L. Fei-Fei, “A multi-view probabilistic model for 3D object classes,” in *IEEE Int. Conf. on Comp. Vis. and Patt. Rec. (CVPR)*, 2009. 1, 6

[9] M. Martinez Torres, A. Collet Romea, and S. Srinivasa, “MOPED: A scalable and low latency object recognition and pose estimation system,” in *IEEE Int. Conf. on Rob. and Autom. (ICRA)*, 2010. 1

[10] F. Viksten, R. Soderberg, K. Nordberg, and C. Perwass, “Increasing pose estimation performance using multi-cue integration,” in *IEEE Int. Conf. on Rob. and Autom. (ICRA)*, 2006. 1

[11] C. Gu and X. Ren, “Discriminative mixture-of-templates for viewpoint classification,” in *IEEE Europ. Conf. on Comp. Vis. (ECCV)*, 2010. 1

[12] K. Lai, L. Bo, X. Ren, and D. Fox, “A scalable tree-based approach for joint object and pose recognition,” in *Conf. on Artificial Intelligence (AAAI)*, 2011. 1

[13] D. Hoiem, C. Rother, and J. M. Winn, “3D LayoutCRF for multi-view object class recognition and segmentation,” in *IEEE Int. Conf. on Comp. Vis. and Patt. Rec. (CVPR)*, 2007. 1

[14] P. Yan, S. M. Khan, and M. Shah, “3D model based object class detection in an arbitrary view,” in *IEEE Int. Conf. on Comp. Vis. (ICCV)*, 2007. 1

[15] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich, “Viewpoint-aware object detection and continuous pose estimation,” *Image and Vision Computing*, 2012. 1, 3

[16] M. Torki and A. M. Elgammal, “Regression from local features for viewpoint and pose estimation,” in *IEEE Int. Conf. on Comp. Vis. (ICCV)*, 2011. 2, 5, 6, 7

[17] A. Opelt, A. Pinz, and A. Zisserman, “Learning an alphabet of shape and appearance for multi-class object detection,” *Int. J. Comp. Vis. (IJCV)*, 2008. 2

[18] D. Teney and J. Piater, “Generalized Exemplar-Based Full Pose Estimation from 2D Images without Correspondences,” in *Digital Image Computing: Techniques and Applications (DICTA)*, 2012. 2, 3, 4

[19] R. Detry and J. Piater, “Continuous surface-point distributions for 3D object pose estimation and recognition,” in *Asian Conf. on Comp. Vis. (ACCV)*, 2010. 3

[20] B. Leibe, A. Leonardis, and B. Schiele, “Robust object detection with interleaved categorization and segmentation,” *Int. J. Comp. Vision (IJCV)*, vol. 77, no. 1-3, pp. 259–289, May 2008. 3

[21] A. Frome, Y. Singer, F. Sha, and J. Malik, “Learning globally-consistent local distance functions for shape-based image retrieval and classification,” in *IEEE Int. Conf. on Comp. Vis. (ICCV)*, 2007. 4

[22] C. Gu, J. J. Lim, P. Arbelaez, and J. Malik, “Recognition using regions,” in *IEEE Int. Conf. on Comp. Vis. and Patt. Rec. (CVPR)*, 2009. 4

[23] S. Maji and J. Malik, “Object detection using a max-margin hough transform,” in *IEEE Int. Conf. on Comp. Vis. and Patt. Rec. (CVPR)*, 2009. 4

[24] P. Yarlagadda and B. Ommer, “From meaningful contours to discriminative object shape,” in *IEEE Europ. Conf. on Comp. Vis. (ECCV)*, 2012. 4

[25] S. A. Nene, S. K. Nayar, and H. Murase, “Columbia object image library COIL-100,” Columbia University, Tech. Rep., 1996. 5

Modeling Pose/Appearance Relations for Improved Object Localization and Pose Estimation in 2D images

Damien Teney

Justus Piater

University of Liège, Belgium
Damien.Teney@ULg.ac.be

University of Innsbruck, Austria
Justus.Piater@UIBK.ac.at

Abstract. We propose a multiview model of appearance of objects that explicitly represents their variations of appearance with respect to their 3D pose. This results in a probabilistic, generative model capable of precisely synthesizing novel views of the learned object in arbitrary poses, not limited to the discrete set of trained viewpoints. We show how to use this model on the task of localization and full pose estimation in 2D images, which benefits from its particular capabilities in two ways. First, the generative model is used to improve the precision of the pose estimate much beyond nearest-neighbour matching with training views. Second, the pose/appearance relations stored within the model are used to resolve ambiguous test cases (e.g. an object facing towards/away from the camera). Here, changes of appearance as a function of incremental pose changes are detected in the test scene, using a pair or triple of views, and are then matched with those stored in the model. We demonstrate the effectiveness of this method on several datasets of very different nature, and show results superior to state-of-the-art methods in terms of accuracy. The pose estimation of textureless objects in cluttered scenes also benefits from the proposed contributions.

1 Introduction and related work

We focus on the problem of 3D pose estimation of known objects in 2D images, using multiple registered images of the objects as training examples. Pose estimation, which is closely coupled to the related tasks of object recognition and localization, is a fundamental problem in computer vision and has naturally received great interest over the years. The main contribution of this paper is to explicitly include, in an existing multiview model of appearance [14], the possible changes of appearance undergone by the object as its pose varies between the trained viewpoints. With the exception of [9], this is, to our knowledge, the only work to include such information within a model of appearance in the context of pose estimation. We make use of this additional information in two different ways to improve the precision and accuracy of pose estimation. In the following we relate our approach to related work.

Multiview models of appearance. The traditional methods for object recognition using 2D images alone, known as appearance-based, typically use

specific models for individual viewpoints, e.g. a model for cars seen from the front, and another for cars seen from the side. Recent contributions in object recognition have introduced more and more models of appearance that include different viewpoint and that are also relevant to pose estimation. Some methods still treat those different viewpoints somewhat independently [14,18], while others try to match and link features across viewpoints [5,10,16]. Savarese *et al.* [10], for example, model an object as a collection of planar parts that can appear in different views. We follow an intermediate approach, by storing independently the image features that make up the different views, but we also store, along with every each image feature, how its appearance varies with respect to the pose of the object. The multiview models mentioned above were mainly used on the task of localization and recognition, without recovering a 3D pose explicitly, or only as rough estimate such as “frontal view” or “side view”. We rather focus on *continuous* pose estimation, to recover precise 3D position and orientation, as is needed, e.g. for robotic applications [7,18].

Continuous pose estimation. The classical approach to pose estimation using 2D training examples is to match highly discriminative features between the test and training views. These matches then vote for the most similar training example, yielding a nearest-neighbour classification of limited precision. Some authors have proposed averaging [18] and probabilistic smoothing [14,15] schemes to increase precision beyond the resolution of the training examples on the viewing sphere. While these procedures basically perform some averaging between trained viewpoints, we rather explicitly detect, and include in the model, the deformations and the transformations of appearance between the discrete viewpoints seen during training. We then use this information in our generative model to finely optimize the 3D pose, starting from a rough nearest-neighbour estimate. Another, radically different approach was proposed by Toriki and El-gammal [17], who learn a regression from local image features to the pose of the object. This original approach recovers a precise pose, but cannot handle significant clutter or occlusions, and the accurate pose estimation depends on the (supervised) enforcement of a one-dimensional manifold constraint (corresponding to the 1D rotation of the object in the training examples). It is not clear how that approach would extend to the estimation of the full 3D pose of an object.

New view synthesis. Our approach uses dense optical flow to identify the deformations between pairs of neighbouring training views. Only those parts of this dense information are then retained that correspond to the sparse image features actually stored in the model. This information can then be used in a generative manner, to synthesize the appearance of the object in a new, unseen pose, by transforming the image features of nearby trained viewpoints according to those stored deformations. The problem of new view synthesis has been studied in the field of computer graphics through the technique of *morphing* [1,2,11]. Most methods only consider pairs or triples of views, whereas we are interested in modeling and using transformations over the whole viewing sphere. Morphing algorithms also often rely on established correspondences between specific

image features of the input views [11], whereas we use dense optical flow to identify deformations between neighbouring views, before applying them to sparse features. As an advantage, our approach readily applies to difficult-to-match features (as opposed to the competing method of Savarese *et al.* [9]). This practically allows handling non-textured objects containing little detail. Although some global consistency in the detected deformations is enforced by optical flow algorithm, each image feature independently stores its possible deformations. This does not limit the model to a particular class of overall transformations. On the contrary, Savarese *et al.* [9] specifically models affine transformations of object parts, assuming that large planar parts can be identified (which is not a universal property of objects). Note also that we use a sparse set of training views (typically spaced about 20° apart on the viewing sphere) and do not require videos or dense sequences of images to track features between frames, as opposed to Sun *et al.* [13]. One may also note a similarity in spirit with the classical active appearance models used mainly for object tracking; they are however based on and limited by point-wise matches of specific landmarks.

Active vision. In addition to the generative model we use to refine pose estimates, we show how to use the deformations stored in the model to resolve ambiguous test cases (Section 4). In such scenes, the 2D appearance of the object can equally correspond to several 3D poses (see Fig. 4 for an example). We propose to also identify the changes of appearance with respect to the pose in the *test scene*. The camera is therefore allowed to move slightly in two orthogonal directions on the viewing sphere (around the test scene). The changes of appearance are detected — as with the training views — and used as extra dimensions in the descriptor of the image features. The features of the test scene can then be matched more discriminatively with those of the training data, and effectively identify the single correct pose unambiguously. This procedure, which can prove crucial in robotic applications, has been proposed in the field of active vision [3,12], but not integrated, to our knowledge, in such a straightforward manner within a pose estimation method. It resembles the way humans themselves resolve such perception ambiguities (in addition to stereo vision) by moving around the scene. Note that a similar effect can be obtained by fusing the result of independent pose estimations from multiple 2D images [14,18]. Our integrated procedure is however arguably more efficient, the displacement (change of camera position) does not need to be precisely know, and can also be very small (theoretically infinitesimal small, even though image noise and resolution dictate minimum displacements in practice).

2 Representation of training and test views

2.1 Notations for image features and object poses

The contributions of this paper are integrated with the method proposed in earlier work [14]. That method performs object recognition and pose estimation in 2D images, and is applicable to various types of image features. This includes features that cannot easily be matched between training and test views, such as

IV

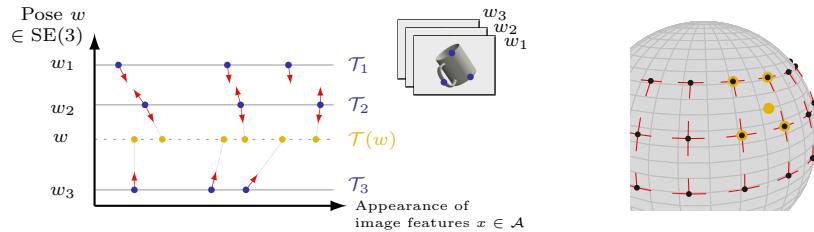


Fig. 1: Left: schematic representation of training data, in the dimensions of pose and appearance. Image features (blue points) are extracted from training images, and their possible changes of appearance (red arrows) are identified between neighbouring views. The appearance of the model at a novel view w (orange) is generated by adjusting the features of close-by views, according to those stored deformations. Right: representation of a set of training viewpoints (black dots) on the viewing sphere. The changes of appearance are detected between pairs of neighbouring views (red links). The appearance of a novel view (orange dot) is generated using the closest training viewpoints, four in this case (orange circles).

the edge points we use in our implementation. We will first review the notation for representing the test and training views, followed by the generative model, capable of synthesizing novel views.

The test data corresponds to a single 2D image of a scene, from which we extract image features. In general, an image feature x is defined by its localization in the image x^p , and an optional appearance descriptor x^a . Together, they are defined on the *appearance* space \mathcal{A} . Practically, we use points identified along edges, combined with the local orientation of the edge (see Fig. 3), so that $\mathcal{A} = \mathbb{R}^2 \times S_1^+$ (the 2D localization plus an orientation without direction). The image features from the test view form a set of *observations* $\mathcal{O} = \{x_i\}_i$, with $x_i \in \mathcal{A}$. The training data correspond to a series of K images of the object of interest, in different poses $w_k \in \text{SE}(3)$ with $k = 1, \dots, K$. Image features are extracted from each training view k to form a set $\mathcal{T}_k = \{x_i\}_{i=1}^{M_k}$, with $x_i \in \mathcal{A}$.

A pose $w \in \text{SE}(3)$, which defines a 3D location together with a 3D orientation, conveniently decomposes into separate sets of dimensions w^v and w^t . We call w^v the *viewpoint transformations* (defining which side of the object is facing the camera, i.e. an element of S^2) and w^t the set of *in-plane transformations* (i.e. translations and rotations parallel to the image plane, and depth/scale changes). In-plane and viewpoint transformations are considered separately, since the changes of appearance induced by the former are fixed by the calibration of the camera. The calibration is assumed to be known, and those transformations can thus be formally hard-coded in the function $\text{transformInPlane}_{w^t}(\mathcal{T}) = \mathcal{T}'$, which transforms a set of image features \mathcal{T} according to the in-plane transformations w^t . Without loss of generality, the following discussion will assume that the training views have been normalized for in-plane transformations, that is, centered and set to a similar scale/rotation¹.

¹ Formally, $\mathcal{T}_k \leftarrow \text{transformInPlane}_{w_k^t}(\mathcal{T}_k)$ and $w_k^t \leftarrow 0, \forall k$.

2.2 Generative model of training data

The training data, as presented above, defines the appearance of the object of interest at a set of discrete trained viewpoints. The goal of our generative model is to fill in the gaps between those viewpoints. Although it may be possible to establish explicit correspondences between image features of nearby training images, this approach may not always be reliable, and it does not generalize to dense or non-discriminative image features such as our edge points. Therefore, we choose to identify *dense* deformations between pairs of adjacent training views, using an optical flow algorithm. Those deformations will then be combined to deform the image features of the training images into the novel viewpoint.

More precisely, for an arbitrary viewpoint w^v , we identify its closest training viewpoints $\text{nb}(w^v) = \{k : d(w^v, w_k^v) \leq t\}$, where $d(\cdot, \cdot)$ measures the angular distance between two viewpoints. The threshold t is chosen similar to the typical angular distance between neighbouring viewpoints in the training data. We also identify the set of all neighbouring training viewpoints as $\text{NB} = \{(k, k') : k' \in \text{nb}(w_k^v), k \neq k'\}$. During an off-line training phase, an optical flow algorithm [6] is applied on all pairs of views $(k, k') \in \text{NB}$. Each pair produces a dense flow map $\text{UV}_{k \rightarrow k'}(x)$ that corresponds, in our case, to the local deformation (translation in the image plane) undergone at an image location x when moving from viewpoint k to k' . Although we compute a *dense* optical flow, we only need to store the actual values of the maps UV for the positions of the few image features of each view. We can now define our generative model $\mathcal{T}(w)$, which produces a set of image features corresponding to the appearance of the object in an arbitrary pose w . Its definition combines the image features of all nearby training views, individually translated using the stored deformations, then adjusted for in-plane transformations. Formally,

$$\mathcal{T}(w) = \bigcup_{k \in \text{nb}(w^v)} \text{transformInPlane}_{w_k^v \rightarrow w^v} \left(\text{deform}_{w_k^v \rightarrow w^v}(\mathcal{T}_k) \right). \quad (1)$$

The functions $\text{transformInPlane}()$ and $\text{deform}()$ adjust a set of image features respectively for in-plane and out-of-plane transformations. While the definition of the former is trivial (it just applies the translation, scaling and rotation of its parameter), the latter is more complex. It uses a linear combination of two available deformations to translate each image feature. We denote those two deformations by the indices of the viewpoints that generated them, and call them (k, k') and (k, k'') . They are chosen from NB so that the novel viewpoint can be reached (on the viewing sphere) by a positive linear combination of them. Consequently, $\exists \alpha, \beta \in \mathbb{R}^+ : w^v = w_k^v + \alpha(w_{k'}^v - w_k^v) + \beta(w_{k''}^v - w_k^v)$. Practically, this means that the viewpoints k, k' and k'' cannot be collinear on the viewing sphere. With training viewpoints spaced on a grid, as in our experiments, we simply choose k' and k'' respectively along the changes in azimuth and elevation. It is now straightforward to define the $\text{deform}()$ function that combines those two chosen deformations:

$$\begin{aligned} \text{deform}_{w_k^v \rightarrow w^v}(\mathcal{T}_k) = \{x'_i : x_i^p = x_i^p + \alpha \text{UV}_{k \rightarrow k'}(x_i^p) + \beta \text{UV}_{k \rightarrow k''}(x_i^p) \\ \text{and } x_i^a = x_i^a, \forall x_i \in \mathcal{T}_k \}. \end{aligned} \quad (2)$$

3 Refinement of pose with generative model

3.1 Method

The proposed generative model readily integrates with the method proposed in [15]. That method for pose estimation relies on continuous distributions of image features in the appearance space to represent the training and test views, using kernel density estimation. These distributions are simply built using the elements of \mathcal{O} and $\mathcal{T}(w)$ as particles, giving respectively $\phi_{\mathcal{O}}(x) = \frac{1}{|\mathcal{O}|} \sum_{x_i \in \mathcal{O}} \mathbf{K}(x, x_i)$ and $\phi_{\mathcal{T}(w)}(x) = \frac{1}{|\mathcal{T}(w)|} \sum_{x_i \in \mathcal{T}(w)} \mathbf{K}(x, x_i)$, with $\mathbf{K}(\cdot, \cdot)$ a kernel on \mathcal{A} .

We reuse the base method proposed in [15] to obtain initial proposals for the 3D pose of the object in the new scene. That method iterates over the training viewpoints and some discrete values of scale and in-plane rotation, then uses a probabilistic voting scheme between matching training and test features to identify the most probable image location. The peaks with high voting scores are then retained as initial pose estimates. This method basically corresponds to a nearest-neighbour identification of the training views in the test scene, and gives us initial estimates to refine by local optimization.

Using the two distributions of image features presented above, and a cross-correlation between them as a measure of similarity [14,15], we now have a likelihood function that can be used to evaluate any arbitrary pose w :

$$p(w) = \int_{\mathcal{A}} \phi_{\mathcal{O}}(x) \phi_{\mathcal{T}(w)}(x) dx, \quad \text{approximated with} \quad (3)$$

$$p(w) \approx \frac{1}{n} \sum_i^n \phi_{\mathcal{O}}(x_i) \quad \text{where } x_i \sim \phi_{\mathcal{T}(w)},$$

using Monte Carlo integration, which gives a convenient expression, relatively inexpensive to evaluate. This function $p(w)$ constitutes the objective function that we seek to maximize when optimizing a pose estimate. In practice, it is generally smooth in the neighbourhood of the global optimum, but no assumption can be made about its convexity, and its definition on the 6-dimensional pose space $\text{SE}(3)$ makes the evaluation of its gradient difficult. Fortunately, our initial pose estimate can be assumed to be a close approximation of the global optimum. All those conditions motivated the use of a simple hill-climbing algorithm. We iteratively optimize pairs of dimensions at a time, namely the 2 viewpoint angles, the image location, then the scale and in-plane rotation. We empirically observed that a close approximation of the global optimum can be reached in this way after only a few iterations (see Fig. 5, bottom right).

3.2 Results

Rotating car. We first evaluate our method on the first sequence of the “rotating car” dataset [8]. It consists of 118 images of a car on a rotating platform, shot at a motor show. Although it only includes a single degree of freedom (the rotation around the vertical axis), this dataset is interesting as it was shot in real

conditions, features a highly textured object of complex structure and allows a comparison of precision with two state-of-the-art methods. We report in Fig. 2 the precision (error of the estimated rotation angle of the car) of our initial pose estimate and of the pose estimates optimized using our generative model (Section 3.1). We show a clear advantage, with different sizes of training sets (which contain uniformly spaced images from the sequence).

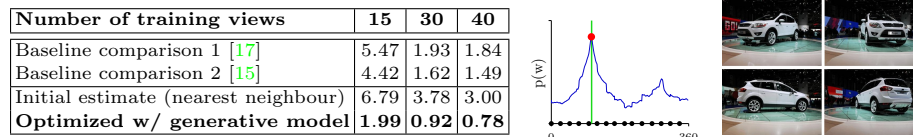


Fig. 2: Results of pose estimation on the rotating car dataset; mean error in degrees. Center: for one test image, we verify that our objective function (blue; evaluated over the whole range of the 1D rotation for demonstration) presents its global optimum near the ground truth (green line). Also represented: training poses (black dots) and our optimized result (red dot). Right: samples test images.

3D pose dataset. We now evaluate our method using the “3D pose dataset” of Viksten *et al.* [4,19]. It is one of the few public datasets available with views spanning more than a 1D rotation, and precisely annotated (in this case with the azimuth/elevation angles of the viewpoint). We use the only object (Volvo car) that was evaluated individually [4], using the same experimental conditions. This allows a comparison with a classical method [4], which uses discriminative image descriptors with a voting and averaging scheme; this constitutes the classical approach for robust 3D pose estimation. The small and large training sets contain views spaced respectively 20° and 10° apart (on both azimuth and elevation angles), with test views in between. With the larger training set, we obtain results superior to [4] in terms of accuracy (Fig. 3). The smaller training set is more challenging for detecting deformations between views, reaching the limits of the optical flow algorithm we use. Our large mean error is caused by two views that yield wrong initial pose estimates, with a large error of almost 180° , due to the similar aspect of the front and rear of the car.

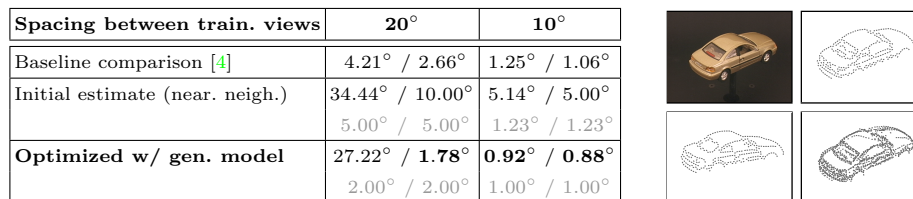


Fig. 3: Results on 3D pose dataset; mean (median) error of azimuth/elevation angles. Clockwise: one test image, its image features, features produced by our generative model at the optimized pose, and features of closest training view; the generative model closely approximates the appearance of the unseen view.

4 Matching pose/appearance relations in ambiguous test scenes

4.1 Method

We propose to make use of the pose/appearance relations identified and stored within the model in a second manner, as extra dimensions of the descriptor of the image features. In this context, the same changes of appearance with respect to the pose are identified in the test view, using additional images obtained by moving the camera slightly around the test scene (which effectively changes the relative pose between the camera and the object of interest). Those additional images are only used to identify the deformations (as in Section 2.2); only the features of the original test image are actually used. Each image feature $x \in \mathcal{A}$ of both the training set and the test image is then complemented with an extra information x^d , a first-order approximation of the derivative of its position in the image with respect to the viewpoint, i.e. $x^d \approx \frac{\partial x^p}{\partial w^v}$ ($\in \mathcal{R}^2 \times \mathcal{R}^2$). This conveniently constitutes a compact representation of the local deformations. In practice, considering an image feature x of a training view k , we approximate x^d by averaging the deformations identified with the neighbouring views:

$$x^d = \text{average}_{k' \in \text{nb}(k)} \left(\frac{\text{UV}_{k \rightarrow k'}(x^p)}{w_{k'}^v - w_k^v} \right). \quad (4)$$

Using azimuth and elevation angles to parametrize a viewpoint on the 2-sphere, this expression gives us two vectors (each $\in \mathcal{R}^2$), corresponding to the translation in the image plane relative resp. to azimuth and elevation changes of the viewpoint. These extra feature descriptors are similarly extracted in the test view. We then use them, both when matching observations between the training and test views for voting for an initial pose estimate, and when measuring the similarity between the test view and a generated view (Section 3.1). In both cases, we set a hard threshold for classifying two features x_1 and x_2 as similar²:

$$\text{angle}(x_1^d, x_2^d) < 135^\circ \quad \text{or} \quad \|x_1^d\| < t' \quad \text{or} \quad \|x_2^d\| < t'. \quad (5)$$

The threshold t' on the magnitude of the deformations discards small and insignificant deformations, which cannot be identified reliably. The function $\text{angle}(\cdot, \cdot)$ measures the difference in direction between the two deformations. The maximum value of 135° discards matches of *truly opposite* directions (as is the case with the ambiguous situations we are interested in), while still keeping most (even uncertain) matches (maybe simply due to noise), which is important in the voting algorithm [15] for the initial pose estimate.

4.2 Results

We finally evaluate the second proposed use of image deformations, for resolving ambiguous test cases by matching them. No dataset suitable for this very

² To shorten notations, we express the condition on a single vector, but it must be verified by both parts of x^d .

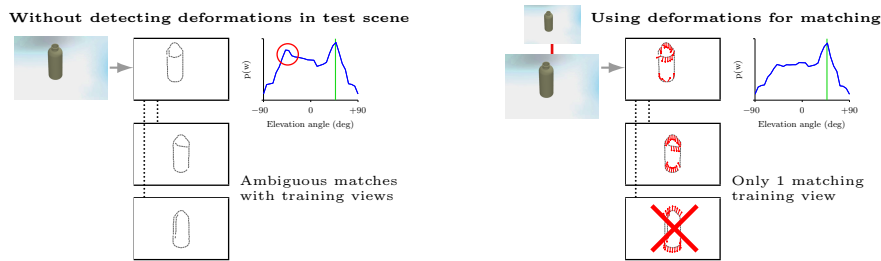


Fig. 4: Left: ambiguous test scene. The extracted edges correspond equally well to training images of the bottle facing towards/away from the camera, and our pose likelihood function $p(w)$ presents two strong peaks (incorrect one in red, ground truth in green). Right: using a second image taken after moving the camera slightly to the top, we detect deformations of image features (red arrows), and match them with the training examples; only the correct peak of $p(w)$ remains.

Objects					
No deform. + near. neigh.	55%	80%	68%	69%	51%
	6.3°	5.0°	6.5°	5.2°	16.7°
Match. def. + near. neigh.	66%	82%	81%	79%	60%
	6.2°	16.6°	6.2°	5.4°	17.5°
Match. def. + gen. model	70%	82%	92%	77%	60%
	5.0°	14.0°	4.6°	1.4°	14.7°

Fig. 5: Left: results on synthetic scenes without/with matching of deformations; success rate (localization error < 20 px and pose error $< 20^\circ$) and mean pose error of successes. Right: sample test images with localization results as bounding boxes. Bottom right: typical evolution of our objective function after successive optimizations for viewpoint, localization and scale/in-plane rotations.

particular problem is currently available, and we resorted to synthetic images, featuring simple objects. Although simple in appearance, they actually prove challenging for pose estimation due to their lack of detail and texture. The test data is now a “central” 2D image, complemented by 2 additional views obtained by moving the camera slightly to the right and to the top. This allows recovering the changes of appearance of the scene with respect to the pose, and matching them with the training data (Fig. 4). As reported in Fig. 5, this eases the localization and improves the precision of the pose estimation.

5 Conclusions

We integrated, within a multiview model of appearance, the explicit transformations undergone by the image features between the training viewpoints. The deformations between example images are detected with dense optical flow and stored for the discrete image features. First, we used this information in a generative model, to refine an initial estimate of the 3D pose of the object in a

new scene. Second, we showed how to match deformations between training and test data, in order to resolve the pose in ambiguous test images. We clearly demonstrated the advantage of those contributions on several datasets, and over existing methods. As future work, it will be interesting to integrate and evaluate these principles within the context and practical conditions of robotic applications.

Acknowledgments The research leading to these results has received funding from the European Community’s Seventh Framework Programme under grant agreement no. 270273, Xperience. Damien Teney is supported by a research fellowship of the Belgian National Fund for Scientific Research.

References

1. Avidan, S., Shashua, A.: Novel view synthesis in tensor space. In: CVPR (1997) **II**
2. Chen, S.E., Williams, L.: View interpolation for image synthesis. In: SIGGRAPH (1993) **II**
3. Chen, S., Li, Y., Kwok, N.M.: Active vision in robotic systems: A survey of recent developments. *Int. J. of Rob. Res.* 30(11), 1343–1377 (2011) **III**
4. Johansson, B., Moe, A.: Patch-duplets for object recognition and pose estimation. In: CRV (2005) **VII**
5. Kushal, A., Schmid, C., Ponce, J.: Flexible object models for category-level 3D object recognition. In: CVPR (2007) **II**
6. Liu, C.: Beyond Pixels: Exploring New Representations and Applications for Motion Analysis. Ph.D. thesis, MIT (2009) **V**
7. Martinez Torres, M., Collet Romea, A., Srinivasa, S.: MOPED: A scalable and low latency object recognition and pose estimation system. In: ICRA (2010) **II**
8. Ozuysal, M., Lepetit, V., Fua, P.: Pose estimation for category specific multiview object localization. In: CVPR (2009) **VI**
9. Savarese, S., Fei-Fei, L.: View synthesis for recognizing unseen poses of object classes. In: ECCV. Marseille, France (2008) **I, III**
10. Savarese, S., Fei-Fei, L.: 3D generic object categorization, localization and pose estimation. In: IEEE Int. Conf. on Comp. Vis. (2007) **II**
11. Seitz, S.M., Dyer, C.R.: Toward image-based scene representation using view morphing. In: Int. Conf. on Patt. Rec. pp. 84–89 (1996) **II, III**
12. Sipe, M.A., Casasent, D.: Best viewpoints for active vision classification and pose estimation. In: Intelligent Robots and Comp. Vis. pp. 382–393 (1997) **III**
13. Sun, M., Su, H., Savarese, S., Fei-Fei, L.: A multi-view probabilistic model for 3D object classes. In: CVPR (2009) **III**
14. Teney, D., Piater, J.: Generalized Exemplar-Based Full Pose Estimation from 2D Images without Correspondences. In: DICTA (2012) **I, II, III, VI**
15. Teney, D., Piater, J.: Continuous pose estimation in 2D images at instance and category levels. Submitted (2013) **II, VI, VII, VIII**
16. Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiel, B., Van Gool, L.: Towards multi-view object class detection. In: CVPR (2006) **II**
17. Torki, M., Elgammal, A.M.: Regression from local features for viewpoint and pose estimation. In: ICCV (2011) **II, VII**
18. Vikstén, F., Soderberg, R., Nordberg, K., Perwass, C.: Increasing pose estimation performance using multi-cue integration. In: ICRA (2006) **II, III**
19. Vikstén, F., Forssén, P.E., Johansson, B., Moe, A.: Comparison of local image descriptors for full 6 degree-of-freedom pose estimation. In: ICRA (2009) **VII**

Markerless Self-Recognition and Segmentation of Robotic Manipulator in Still Images

ICRA 2013 Mobile Manipulation Workshop on Interactive Perception

Damien Teney
University of Liège, Belgium

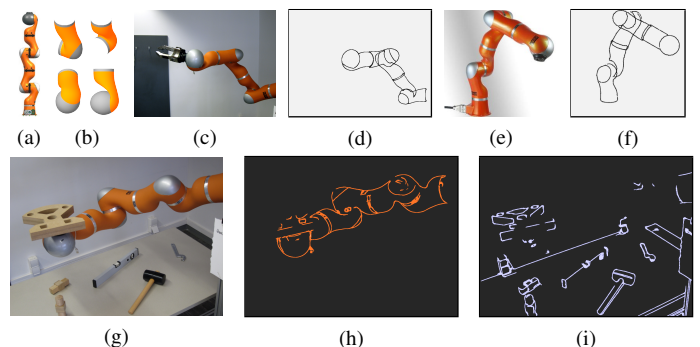
Dadhichi Shukla
University of Burgundy, France

Justus Piater
University of Innsbruck, Austria

Vision is a crucial capability for enabling robots to perceive and interact with their environment, e.g. manipulating or grasping objects. A current trend is bringing closer the aspects of interaction and perception, on the one hand by integrating visual information directly in the control process, and on the other hand, using interaction itself to help perception, allowing robots to explore their environment. In the context of manipulation, physical parts of the robot are then likely to appear in the observations, and an important capability emerges as the recognition of those parts, in order to separate the observations of the scene from those of the robot itself. Identifying the robot's own body parts in input images has been used before in different ways, helping obstacle avoidance or control directly (through visual servoing [1]). However, this is usually performed via indirect methods, tracking fiducial markers purposely attached to the robot [2], which imposes undesirable (e.g. visibility) constraints. Some recent work addresses the pose estimation of a robot manipulator *directly* [3], [4], but these methods focus on tracking the manipulator between consecutive frames, whereas the initial recognition is considered as the harder part. We propose a method for markerless, monocular recognition and pose estimation of an articulated robot arm, dealing with single images without initialization, allowing its use with unknown hand-eye calibration, imprecise kinematics or missing position feedback.

We use an existing appearance-based recognition method [5], that relies on object edges and contours, allowing the recognition of objects with few characteristic visual features (i.e. non-textured). The system is trained separately for each articulated link of the robot arm, using synthetic images of that link in known poses. The recognition of those elements proves extremely challenging, as their appearance may not offer very distinguishable visual clues, and because of the typical unstructured environment (background clutter) and possible (self-)occlusions. The initial recognition produces a set of candidate poses for each link, which are then combined, enforcing the known kinematic constraints between the links. These constraints are modeled as pairwise compatibility functions in a classical Markov random field, with a node for each link. Inference is carried out with an algorithm inspired by non-parametric belief propagation. We efficiently limit the evaluation of densities in the pose space to the discrete points proposed by the initial recognition step, thereby ensuring adequate efficiency. The algorithm ultimately recovers the pose of all the elements (links) of the arm. We can then classify the image features of the input scene as belonging to the scene or to the robot manipulator itself, simply by measuring their similarity with the training templates of the links in the identified poses. The poses of the links can also be used to recover the angles at each joint, together with the cartesian position of each element of the robot relative to the camera.

The system was implemented and tested with a Kuka Lightweight Robot arm. We considered the five internal links, of which four are completely identical in appearance, which constitutes an additional challenge. The four joints (revolute, both axial and hinge-like) are specified by the alignment of the joint axes of the adjacent links. Training images of the two types of links were generated with CAD software, at viewpoints about 30° apart. Although the recognition of individual links cannot be relied upon for any practical purposes, the pose ultimately recovered for the whole arm by the proposed algorithm was correct during most of our tests. The probabilistic inference can handle missing detections of links to some degree, as can happen with (self-)occlusions. The classification of the image features of the test image as robot/non-robot parts, as mentioned above, proved effective, and superior to using intermediate segmentation masks. Indeed, our procedure can handle occlusions, for example when the robot is manipulating an object. The capabilities offered by the whole system should help and make more robust the subsequent processing of visual data in the context of joint perception/manipulation scenarios. Future work will aim at relaxing the assumptions of known link and joint geometries, e.g. reusing existing work on autonomous learning of articulated models [6].



(a) The Kuka LWR used in our experiments (b) Synthetic training images of individual links (c, e) Test images and (d, f) rendering of the training templates as recognized for each link (g-i) After recognition of the robot arm, image features are classified as “robot” (orange) and “non-robot” (blue).

Research supported through European Commission's FP7 projects Xperience and IntellAct, and by the Belgian National Fund for Scientific Research.

- [1] M. Prats, P. Martinet, A. del Pobil, and S. Lee, “Robotic execution of everyday tasks by means of external vision/force control,” *Intelligent Service Robotics*, vol. 1, no. 3, pp. 253–266, 2008.
- [2] K. Welke, J. Issac, D. Schiebener, T. Asfour, and R. Dillmann, “Autonomous acquisition of visual multi-view object representations for object recognition on a humanoid robot,” in *ICRA*, 2010.
- [3] J. J. Sorribes, M. Prats, and A. Morales, “Visual tracking of a jaw gripper based on articulated 3D models for grasping,” in *ICRA*, 2010.
- [4] X. Gratal, J. Romero, and D. Kragic, “Virtual visual servoing for real-time robot pose estimation,” in *IFAC*, 2011.
- [5] D. Teney and J. Piater, “Continuous pose estimation in 2D images at instance and category levels,” in *Comp. and Rob. Vis.*, 2013.
- [6] D. Katz and O. Brock, “Extracting planar kinematic models using interactive perception,” in *Unifying Perspectives In Computational and Robot Vision*, no. 8. Springer Verlag, May 2008, pp. 11–23.

A Study of Point Cloud Registration with Probability Product Kernel Functions

Hanchen Xiong Sandor Szedmak Justus Piater
Institute of Computer Science, University of Innsbruck
Technikerstr.21a A-6020, Innsbruck, Austria

{hanchen.xiong, sandor.szedmak, justus.piater}@uibk.ac.at

Abstract

3D point cloud registration is an essential problem in 3D object and scene understanding. In many realistic circumstances, however, because of noise during data acquisition and large motion between two point clouds, most existing approaches can hardly work satisfactorily without good initial alignment or manually marked correspondences. Inspired by the popular kernel methods in machine learning community, this paper puts forward a general point cloud registration framework by constructing kernel functions over 3D point clouds. More specifically, Gaussian mixtures based on the point clouds are established and probability product kernel functions are exploited for the registration. To enhance the generality of the framework, $SE(3)$ on-manifold optimization scheme is employed to compute the optimal motion. Experimental results show that our registration framework works robustly when many outliers are presented and motion between point clouds is relatively large, and compares favorably to related methods.

1. Introduction

The past two decades have witnessed great development in 3D point cloud acquisition and usage, as increasingly more state-of-the-art stereo vision reconstruction algorithms were developed and affordable range-finder devices emerged. However, there still exist many obstacles to full exploitation of 3D point clouds, among which 3D registration plays a fundamental role. Simply speaking, 3D point cloud registration is the problem of moving a model point cloud \mathcal{M} to achieve the best possible alignment with a fixed scene point cloud \mathcal{S} by minimizing a certain distance function between them. Mathematically, the motion \mathbf{P} from \mathcal{M} to \mathcal{S} is a rigid transformation in \mathbb{R}^3 , which is composed of a 3D rotation \mathbf{R} and a translation \mathbf{t} . Given f as a distance function between two point clouds, the problem can be formulated as $\{\mathbf{R}^*, \mathbf{t}^*\} = \arg \min_{\mathbf{R}, \mathbf{t}} f(\mathcal{M}, \mathcal{S}; \mathbf{R}, \mathbf{t})$. One

can easily form f as the sum of squared distances between all corresponding points from two clouds if the correspondence information is provided. Unfortunately, however, there is no existing method which can find the perfectly accurate correspondence between two point clouds. Iterative Closest Point (ICP) [1], the most popular method for 3D registration so far, assumes correspondence based on the closest distance criterion and then computes the transformation that best aligns the putative correspondences. Two steps are implemented alternately and iteratively so that computing either of them will improve the other.

Different from the iterative method, our algorithm avoids point-wise correspondence search. Instead, we consider point clouds in their entirety, and compute them as set-format data in kernel method. In other words, the distance function f is formulated at the cloud level. First, continuous parametric probabilistic density functions (PDFs) are established as the representations of point clouds by applying kernel density estimation (KDE) with Gaussian kernels. Then, with KDE representations, an expected likelihood kernel function is employed to compute the affinity between two distributions, which corresponds to the similarity between two point clouds. Finally, the optimal pose is computed to maximize the kernel function, which is equivalent to minimizing the distance function f between two point clouds.

The key contributions of our work can be summarized in two points: First, we express 3D point cloud registration in a kernel method framework with a special case study on probability product kernel functions. Although the scope of study in this paper is limited, it can indicate a new perspective of the registration problem with more potential kernel functions explored and studied in the future. Secondly, we exploit the $SE(3)$ on-manifold optimization scheme to provide an elegant solution to compute the optimal motion. Since the generality of kernel method and $SE(N)$ optimization, the resulting registration algorithm can be easily extended to any dimension cases (although we only concern 3D case here).

2. Related Work

Since originally proposed [1], ICP and its variants almost dominate the research literature of 3D point cloud registration. As briefly mentioned in section 1, at each iteration of ICP, corresponding pairs are assumed to be found according to the nearest-neighbor criterion, based on which the optimal motion is subsequently computed. It has been realized that in practice this naive correspondence-search-assumption can fail if the motion between two point clouds is large or there exists a large amount of noise. Thus, many improvements to ICP methods were proposed. All ICP variants can be explained as iterative cycles of six steps [20]: (1) *Select subsets of two points clouds*: in most cases, all points are used. However, a subset can be randomly sampled to reduce the computational burden if the number of points is too large. (2) *Match corresponding pairs*: this is a key step which has drawn much attention to achieve improvement. The original version [1] matches pairs by closest distance. Many other studies tried to improve the correspondence accuracy by making use of more information such as normal vectors [7], curvature [23], and color [12]. Even some more sophisticated persistent point feature descriptors [21] were developed to find the exact matches. (3) *Weight the corresponding pairs*: Weighting alleviates the influence of poor correspondence matching. To each pair should be assigned a weight proportional to the likelihood of the correspondence, which can be computed as the compatibility of normals or colors [8]. (4) *Reject pairs*: To some degree, rejection is equivalent to weighting by only assigning binary weights $w = \{0, 1\}$, so normal orientation and color compatibility can be computed in the same way; then a threshold is set to decide which pairs can be accepted. Some other geometric properties such as inter-point distance consistency and collinearity consistency [15] can be also used to filter out weak corresponding pairs. Other methods reject pairs in which two points do not bi-uniquely correspond to each other [24], or use weaker notions of consistency such as ϵ -reciprocal correspondence [17]. (5) *Compute an error metric*: The error metric is usually designed as the sum of squared distances between corresponding points. To enhance the robustness to outliers of correspondences obtained from previous steps, Trimmed ICP was developed [4] by using trimmed squares, which is only composed of square distances with relatively small values. (6) *Minimize the error metric*: This is an optimization step with respect to 3D rotation and translation. Usually unit quaternion and the dual number quaternion methods [25] is used to compute rotation. Although there have been various improvements to ICP, its applicability is still limited to the scenario in which two point clouds can be fairly closely aligned in advance, and noise is low. Therefore, when used in practical applications, some manual assistance is usually required, which makes ICP methods barely suitable within fully automatic

systems.

Besides the progress of ICP methods, some other novel approaches have been proposed from different perspectives. A notable contribution is *spectral correspondence* [3], where graphs are constructed based on point clouds, and the structural properties are extracted with spectral graph theory to find matching point patterns. Another influential work is SoftAssign [9], which establishes one-to-many correspondences with different weights, and the registration is solved as a joint optimization over the transformation and correspondence matrix.

Prior to our work, a related algorithm was proposed by Jian *et al.* in [11]. In Jian’s method, each point cloud is similarly modelled globally and probabilistically with Gaussian mixtures, and the optimal motion is computed to minimize the L2 distance between corresponding Gaussian mixtures. However, we go beyond their work in several ways: first, instead of computing L2 distance between two fitted distributions, we consider the registration problem in a more general kernel-based framework, which results in more flexibility and extensibility. Secondly, in contrast to the identical and spherical covariance, we use more elaborately estimated bandwidths in KDE with Gaussian kernels to capture more local structural information of point clouds. Finally, instead of using unit quaternions, we exploit $SE(3)$ on-manifold optimization to achieve optimal motion estimation, which yields a rather general registration algorithm.

3. Kernel Methods for Point Clouds

Kernel methods have achieved remarkable success in many machine learning applications by enabling various linear models to exploit nonlinear data patterns, such as SVM and kernel PCA [22]. The kernel function is originally designed as a trick to efficiently compute the inner product (similarity) between the images of two data in a mapped (higher dimension) Hilbert feature space:

$$K(x, z) = \langle \Phi(x), \Phi(z) \rangle \quad (1)$$

where $x, z \in \chi$ and $\Phi(\cdot)$ is mapping function from χ to the feature space. A strength of the kernel method is that with properly designed kernel functions it can work with very general types of data, including strings, trees or graphs [22], without explicit feature maps. As a matter of fact, any symmetric similarity measurement between two data sets of certain types can be a kernel function as long as it satisfies positive semi-definiteness, and data types are implicitly embedded into an induced feature space. In this paper, we are focusing on point clouds, on which kernel methods can work in the exact same way to compute the similarity by mapping them into a Hilbert feature space.

3.1. Kernel methods for unordered sets of data

Point clouds can be treated as unordered sets of vectors, on which different kernel functions have been already developed and studied [14, 16]. The basic methodology is to consider all elements in a set as i.i.d. samples from an underlying probability distribution, and thus the distance between two sets can be computed as the discrepancy between two corresponding distributions. One can model two sets probabilistically by fitting them to two distributions whose PDFs are of a certain parametric form, and then instead of developing new kernel functions on point clouds, existing kernel functions on probabilities can be directly used. Kondor and Jebara developed a Bhattacharyya kernel based on the Bhattacharyya affinity used in the statistics literature [14]:

$$K_{Bha}(p, q) = \int \sqrt{p(x)}\sqrt{q(x)} dx \quad (2)$$

where p and q are two distributions. In [16], an expected likelihood kernel, which behaves as computing the expectation of either distribution under the other, is applied as

$$K_{exp}(p, q) = \int p(x)q(x) dx \quad (3)$$

Both the Bhattacharyya kernel and the expected likelihood kernel are two special cases of probability product kernels [10] $K(p, q) = \int p(x)^\rho q(x)^\rho dx$ with $\rho = 0.5$ and $\rho = 1$. Assuming point clouds \mathcal{M} and \mathcal{S} are fitted to two distributions ϕ and ψ respectively, the kernel function on point clouds can be defined as

$$K(\mathcal{M}, \mathcal{S}) = \int \phi(x)^\rho \psi(x)^\rho dx, \quad (4)$$

and the induced Hilbert feature space corresponds to the fitted distribution (infinite-dimensional feature space) when $\rho = 1$, and to the square root of the distribution when $\rho = 0.5$.

3.2. Probabilistic modeling of point clouds

In order to apply (4), one has to model point clouds in a probabilistic form. Because (4) can be efficiently computed without explicit integration if the PDFs of the two distributions are in the exponential family, the Gaussian mixture model is usually used [14, 16]. In this paper, to simplify selecting the number of components, KDE (Kernel Density Estimation) with Gaussian kernel is applied to construct a Gaussian mixture with Gaussians constructed on all points. This approach has been followed in other successful work [5, 6, 11] to establish a probabilistic form of point clouds. However, one weakness of KDE in this other work is that only identical and isotropic covariance is used to construct corresponding Gaussian kernels, which limits the representational power to exploit geometric details of

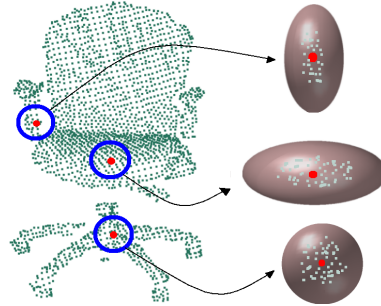


Figure 1. The covariance of the kernel associated with each point is locally determined from its neighbourhood and thus well captures local structure.

the point clouds. Therefore, in our algorithm, in order to capture the local structural information as much as possible, a full covariance is estimated with the neighboring region of each point (Figure 1). Given a model point cloud $\mathcal{M} = \{\lambda_{\mathcal{M}}^{(i)} \in \mathbb{R}^3\}_{i=1}^m$, a Gaussian kernel can be established with its mean equal to the 3D position of each point and covariance estimated with the surroundings of the point. Thus the KDE representation of the model point cloud can be written as

$$\phi(x) = \frac{1}{m_{\mathcal{M}}} \sum_{i=1}^{m_{\mathcal{M}}} \mathcal{N}_{\mathbb{R}^3} \left(x; \lambda_{\mathcal{M}}^{(i)}, \Sigma_{\mathcal{M}}^{(i)} \right) \quad (5)$$

where $\mathcal{N} \left(x; \lambda_{\mathcal{M}}^{(i)}, \Sigma_{\mathcal{M}}^{(i)} \right)$ denotes the normal distribution with mean $\lambda_{\mathcal{M}}^{(i)}$ and covariance $\Sigma_{\mathcal{M}}^{(i)}$, and $m_{\mathcal{M}}$ is the size of model point cloud. Similarly, the KDE representation of the scene point cloud \mathcal{S} can be constructed by the same procedure.

3.3. A kernel function on two point clouds

To put all pieces together, a kernel function on point clouds can be formulated by substituting (5) into (4). For the sake of clarity in describing the framework, here we only study the case of expected likelihood kernel function ($\rho = 1$) although other kernel functions can be analogously derived and explored. Thus (4) can be rewritten as:

$$K(\mathcal{M}, \mathcal{S}) = \frac{1}{m_{\mathcal{M}}} \frac{1}{m_{\mathcal{S}}} \sum_{i=1}^{m_{\mathcal{M}}} \sum_{j=1}^{m_{\mathcal{S}}} \mathcal{G}_{\mathcal{M}^i \mathcal{S}^j} \quad (6)$$

$$\begin{aligned} \mathcal{G}_{\mathcal{M}^i \mathcal{S}^j} &= \int \mathcal{N}_{\mathbb{R}^3} (x; \lambda_{\mathcal{M}}^{(i)}, \Sigma_{\mathcal{M}}^{(i)}) \mathcal{N}_{\mathbb{R}^3} (x; \lambda_{\mathcal{S}}^{(j)}, \Sigma_{\mathcal{S}}^{(j)}) dx \\ &= \frac{1}{(2\pi)^{3/2} |\Sigma^*|^{1/2}} \exp \left\{ -\frac{1}{2} \lambda^{*'} \Sigma^{*-1} \lambda^* \right\} \end{aligned} \quad (7)$$

where $|\cdot|$ denotes the determinant, and $\lambda^* = \lambda_{\mathcal{M}}^{(i)} - \lambda_{\mathcal{S}}^{(j)}$ and $\Sigma^* = \Sigma_{\mathcal{M}}^{(i)} + \Sigma_{\mathcal{S}}^{(j)}$ according to Gaussian identities [19].

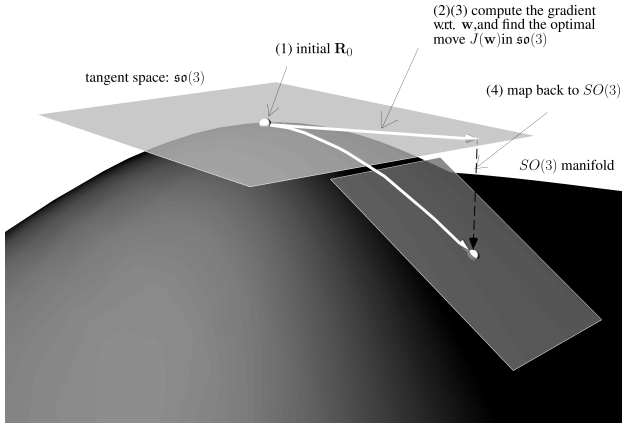


Figure 2. The $SO(3)$ manifold and its optimization scheme: (1) start from a rotation matrix \mathbf{R}_0 ; (2) use equation (10) as the local parametrization of the manifold at point \mathbf{R}_0 , and compute the gradient \mathbf{g} with respect to \mathbf{w} ; (3) compute the best move in $\mathfrak{so}(3)$ by mapping $J(\mathbf{w})$; (4) map back to $SO(3)$: $\mathbf{R}_0 \leftarrow \exp(J(\mathbf{w}))\mathbf{R}_0$; (5) repeat step (2)(3)(4) until convergence

4. Optimal Motion Estimation

A motion \mathbf{P} can be mathematically represented as a rigid transformation (the combination of a rotation \mathbf{R} and a translation \mathbf{t}) in \mathbb{R}^3 , which corresponds to an element of the Lie group $SE(3)$ (Special Euclidean group). Similar to [11], gradient type method is used in this paper to iteratively adjust rotation and translation parameters. Due to the orthogonality constraint of rotation matrices, unit quaternions are used as the representations of 3D rotations in [11]. However, the applicability of unit quaternions is rather limited because they can only be used in 3D cases. In our algorithm, instead, Lie group $SO(3)$ and its associated Lie algebra $\mathfrak{so}(3)$ is exploited to provide a novel solution of optimal rotation estimation. One of the virtue of $SO(3)$ representation of rotation is that it can be easily extended to any n -dimension case by analogously using $SO(n)$. $SE(3)$ on-manifold optimization can be straightforwardly achieved by combining 3D translation and $SO(3)$ rotation. The study of $SE(3)$ on-manifold optimization in this paper is very brief, and of course cannot cover the whole field but only the necessary scope used in this paper. Readers can refer to [2] for more details on $SE(3)$ and related optimization.

4.1. $SO(3)$ and associated Lie algebra

A rotation matrix \mathbf{R} within \mathbf{P} is an element of Lie group $SO(3) = \{\mathbf{R} \in \mathbb{R}^{3 \times 3} : \mathbf{R}^T \mathbf{R} = \mathbf{I}, |\mathbf{R}| = 1\}$, which is referred to as Special Orthogonal group because of its orthogonal characteristic. It has been always an obstacle in rotation-related optimization problems due to the orthogonality constraint. However, recent studies of $SO(3)$ in computer vision and related literature [2][18] reveal that the

$SO(3)$ on-manifold optimization can be used to find appropriate rotation matrices efficiently and elegantly without worrying about the orthogonality constraint. At first, since $SO(3)$ is a Lie group, it should fulfill the associated conditions (and $SE(3)$ as well) [2], and one of them is *closure*: $\forall \mathbf{R}_1, \mathbf{R}_2 \in SO(3), \mathbf{R}_1 \mathbf{R}_2 \in SO(3)$. Secondly, $SO(3)$ is a smooth manifold embedded in \mathbb{R}^3 , which is a topological space wherein all elements are rotation matrices (Figure 2). For each point \mathbf{R}_i on the $SO(3)$ manifold, there exists a tangent space, and fortunately, the tangent space of $SO(3)$ happens to be its associated Lie algebra $\mathfrak{so}(3)$. In other words, $\forall \mathbf{R}_i \in SO(3), \exists \Lambda_i \in \mathfrak{so}(3)$. Intuitively, the tangent space of the $SO(3)$ manifold can be understood as a vector space of the derivative of the manifold at point \mathbf{R}_i (Figure 2). The mathematical connection between $\mathfrak{so}(3)$ and $SO(3)$ is:

$$\mathfrak{so}(3) \rightarrow SO(3) : \mathbf{R} = \exp(\Lambda), \quad \Lambda \in \mathfrak{so}(3), \mathbf{R} \in SO(3) \quad (8)$$

Lie algebra $\mathfrak{so}(3)$ is the collection of anti-symmetric matrices, which can be mapped from \mathbb{R}^3 with a skew operator $J(\cdot)$ defined as:

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \rightarrow J(\mathbf{w}) = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \quad (9)$$

Thus, any point that lies within the infinitesimally small vicinity of a certain point \mathbf{R}_0 on the $SO(3)$ manifold can be represented as:

$$\mathbf{R}(\mathbf{w}) = \exp(J(\mathbf{w}))\mathbf{R}_0 \quad (10)$$

which provides a mapping from vectors in \mathbb{R}^3 to a local neighboring region of \mathbf{R}_0 on the $SO(3)$ manifold, and in which the exponential term can be computed using Rodrigues formula:

$$\exp(J(\mathbf{w})) = \mathbf{I} + J(\mathbf{w}) \frac{\sin(\|\mathbf{w}\|)}{\|\mathbf{w}\|} + J(\mathbf{w})^2 \frac{1 - \cos(\|\mathbf{w}\|)}{\|\mathbf{w}\|^2} \quad (11)$$

Finally, in unconstrained optimization problems, gradient type method is popularly used by iteratively updating the solution. Thanks to the local parametrization of $SO(3)$ in (10), we can transform the update of $SO(3)$ to the one with respect to \mathbb{R}^3 without worrying about the orthogonality constraint. Meanwhile, different from usual cases, instead of computing incremental updates within the same space, in the $SO(3)$ manifold optimization, after every update of \mathbf{w} , it needs to be projected back to $SO(3)$. Then, the gradient is computed within the local parametrization of the corresponding neighboring region. The on-manifold optimization scheme of $SO(3)$ is demonstrated in Figure 2.

4.2. $SE(3)$ on-manifold optimization

Since the translation corresponds to vectors in \mathbb{R}^3 , with the map from \mathbb{R}^3 to $SO(3)$ in (10), we can straightforwardly

establish a map Ω from \mathbb{R}^6 to $SE(3)$ manifold as:

$$\begin{aligned} \Omega : [\mathbf{w}, \mathbf{v}]' &\rightarrow \{\exp(J(\mathbf{w})\mathbf{R}_0), \mathbf{t}_0 + \mathbf{v}\} \\ \text{s.t. } \mathbf{w}, \mathbf{v}, \mathbf{t}_0 &\in \mathbb{R}^3, \mathbf{R}_0 \in SO(3) \end{aligned} \quad (12)$$

Based on the kernel method constructed in section 3, each 3D point cloud is represented by KDE, which is a distribution in the form of Gaussian mixtures. Thus the rotation and translation on point clouds correspond to rotating and shifting distributions. On one hand, both rotation and translation can affect means of Gaussians; on the other hand, since the covariance is invariant to translation, only rotation will be taken into account. With SVD decomposition of the covariance matrix, we have $\Sigma = USU'$, where S is a diagonal matrix with eigenvalues as diagonal entries, and U is a matrix composed of eigenvectors, so the rotation will only change the orientation of eigenvectors but preserve the magnitude of their corresponding eigenvalues. Thus the KDE of the model point cloud after the Euclidean transformations \mathbf{P} is:

$$\begin{aligned} \overline{\phi(x)} &= \mathbf{P} \star \phi(x) \\ &= \frac{1}{m_{\mathcal{M}}} \sum_{i=1}^{m_{\mathcal{M}}} \mathcal{N}(x; \mathbf{P}\lambda_{\mathcal{M}}^{(i)}, (\mathbf{R}U)S(\mathbf{R}U)') \\ &= \frac{1}{m_{\mathcal{M}}} \sum_{i=1}^{m_{\mathcal{M}}} \mathcal{N}(x; \mathbf{R}\lambda_{\mathcal{M}}^{(i)} + \mathbf{t}, \mathbf{R}\Sigma_{\mathcal{M}}^{(i)}\mathbf{R}') \end{aligned} \quad (13)$$

Here we abuse notation $\lambda_{\mathcal{M}}^{(i)}$ and $\mathbf{P}\lambda_{\mathcal{M}}^{(i)}$ for both original and homogeneous coordinates because there is no ambiguity. By substituting (13) into (5)(6), the objective function is the kernel function between \mathcal{M} transformed with $\{\mathbf{R}, \mathbf{t}\}$ and \mathcal{S} :

$$\begin{aligned} K(\mathcal{M}_{\mathbf{R},\mathbf{t}}, \mathcal{S}) &= K_{exp}(\overline{\phi(x)}, \psi(x)) \\ &= \frac{1}{m_{\mathcal{M}}} \frac{1}{m_{\mathcal{S}}} \sum_{i=1}^{m_{\mathcal{M}}} \sum_{j=1}^{m_{\mathcal{S}}} \mathcal{G}_{\mathcal{M}_{\mathbf{R},\mathbf{t}}^i \mathcal{S}^j} \\ \mathcal{G}_{\mathcal{M}_{\mathbf{R},\mathbf{t}}^i \mathcal{S}^j} &= \frac{1}{(2\pi)^{3/2} |\mathbf{G}_{ij}|^{1/2}} \exp\left(-\frac{1}{2} \Delta_{ij}' \mathbf{G}_{ij}^{-1} \Delta_{ij}\right) \end{aligned} \quad (14)$$

$$\quad (15)$$

where $\Delta_{ij} = \lambda_{\mathcal{S}}^{(j)} - \mathbf{R}\lambda_{\mathcal{M}}^{(i)} - \mathbf{t}$ and $\mathbf{G}_{ij} = \mathbf{R}\Sigma_{\mathcal{M}}^{(i)}\mathbf{R}' + \Sigma_{\mathcal{S}}^{(j)}$. Then the optimal motion can be estimated by maximizing the kernel function (14):

$$\{\mathbf{R}^*, \mathbf{t}^*\} = \arg \max_{\mathbf{R}, \mathbf{t}} \underbrace{K(\mathcal{M}_{\mathbf{R},\mathbf{t}}, \mathcal{S})}_{\mathbf{O}} \quad (16)$$

By substituting (12) into (16), the objective function \mathbf{O} is dependent only on \mathbf{w} and \mathbf{v} . The gradient can be computed as (check supplementary material for detailed derivation):

$$\frac{\partial \mathbf{O}}{\partial \mathbf{v}} = \frac{1}{m_{\mathcal{M}} m_{\mathcal{S}}} \sum_{i=1, j=1}^{m_{\mathcal{M}}, m_{\mathcal{S}}} \mathcal{G}_{\mathcal{M}_{\mathbf{R},\mathbf{t}}^i \mathcal{S}^j} \Delta_{ij}' \mathbf{G}_{ij}^{-1} \quad (17)$$

$$\begin{aligned} \frac{\partial \mathbf{O}}{\partial \mathbf{w}} &= \sum_{i=1, j=1}^{m_{\mathcal{M}}, m_{\mathcal{S}}} \left\{ \left(\frac{1}{2} (\Delta_{ij}' \mathbf{G}_{ij}^{-1}) \otimes (\Delta_{ij}' \mathbf{G}_{ij}^{-1}) - \text{vec}(\mathbf{G}_{ij}^{-1})' \right) \right. \\ &\quad \left. \cdot \frac{\partial \mathbf{G}_{ij}}{\partial \mathbf{w}} - \Delta_{ij}' J(\mathbf{R}_n \lambda_m^{(i)}) \right\} \mathcal{G}_{\mathcal{M}_{\mathbf{R},\mathbf{t}}^i \mathcal{S}^j} \end{aligned} \quad (18)$$

Where $\text{vec}(\cdot)$ operator vectorizes a matrix by stacking its columns, and $\frac{\partial \mathbf{G}_{ij}}{\partial \mathbf{w}}$ is:

$$\frac{\partial \mathbf{G}_{ij}}{\partial \mathbf{w}} = (\mathbf{I}_9 + T_{3,3})(\mathbf{R}\Sigma_{\mathcal{M}}^{(i)}\Sigma' \otimes \mathbf{I}_3) \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} \quad (19)$$

where \mathbf{I}_n is a $n \times n$ identity matrix, $T_{3,3}$ is a permutation matrix which satisfies $T_{3,3}\text{vec}(\mathbf{A}) = \text{vec}(\mathbf{A}')$ for any 3×3 matrix, $\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}}$ can be easily computed according to the definition (9).

5. Experiments

With the Jacobian vector $\{\nabla_{\mathbf{w}}, \nabla_{\mathbf{v}}\}$ computed in (17)(18), a gradient descent method for point cloud registration can be summarized as Algorithm 1.

Algorithm 1 3D Point Clouds Registration

Input: two 3D point clouds: $\mathcal{M} = \{\lambda_{\mathcal{M}}^{(i)}\}_{i=1}^{m_{\mathcal{M}}}$ and $\mathcal{S} = \{\lambda_{\mathcal{S}}^{(j)}\}_{j=1}^{m_{\mathcal{S}}}$;
Output: the optimal motion estimation $\{\mathbf{R}^*, \mathbf{t}^*\}$

- 1: compute the covariance on each point based on its neighborhood;
- 2: start from initial rotation \mathbf{R}_0 and translation \mathbf{t}_0
- 3: **while** 1 **do**
- 4: compute the gradient $\{\nabla_{\mathbf{w}}, \nabla_{\mathbf{v}}\}$ with current \mathbf{R}_n and \mathbf{t}_n according to (17)(18);
- 5: **if** both $\nabla_{\mathbf{w}}$ and $\nabla_{\mathbf{v}}$ are small enough **then**
- 6: **return** \mathbf{R}_n and \mathbf{t}_n ;
- 7: **end if**
- 8: map the update of \mathbf{w} back to $SO(3)$: $\mathbf{R}_{n+1} \leftarrow \exp(J(\nabla_{\mathbf{w}}))\mathbf{R}_n$;
- 9: direct update translation: $\mathbf{t}_{n+1} \leftarrow \mathbf{t}_n + \nabla_{\mathbf{v}}$;
- 10: set $n \leftarrow n + 1$
- 11: **end while**

5.1. Qualitative Experiments

At first, we test our algorithm qualitatively on KIT 3D database[13]. Since each 3D object model in the database is in triangulated mesh format, we generate the corresponding point cloud by first sampling a triangle with the probability proportional to its area and then uniformly sampling a point within the selected triangle. For each point cloud (red points in Figure 3), a random motion is generated and applied on it. In addition, either random outliers are added or random part of object are removed to generate a synthetic target point cloud (blue points in Figure 3). Some test results of our registration algorithm is displayed in Figure 3. We can see that the proposed algorithm can work qualitatively well in many challenging cases (registration results in green frames). However, the algorithm will also fail when the rotation angle is bigger than 90° or due to the overlage

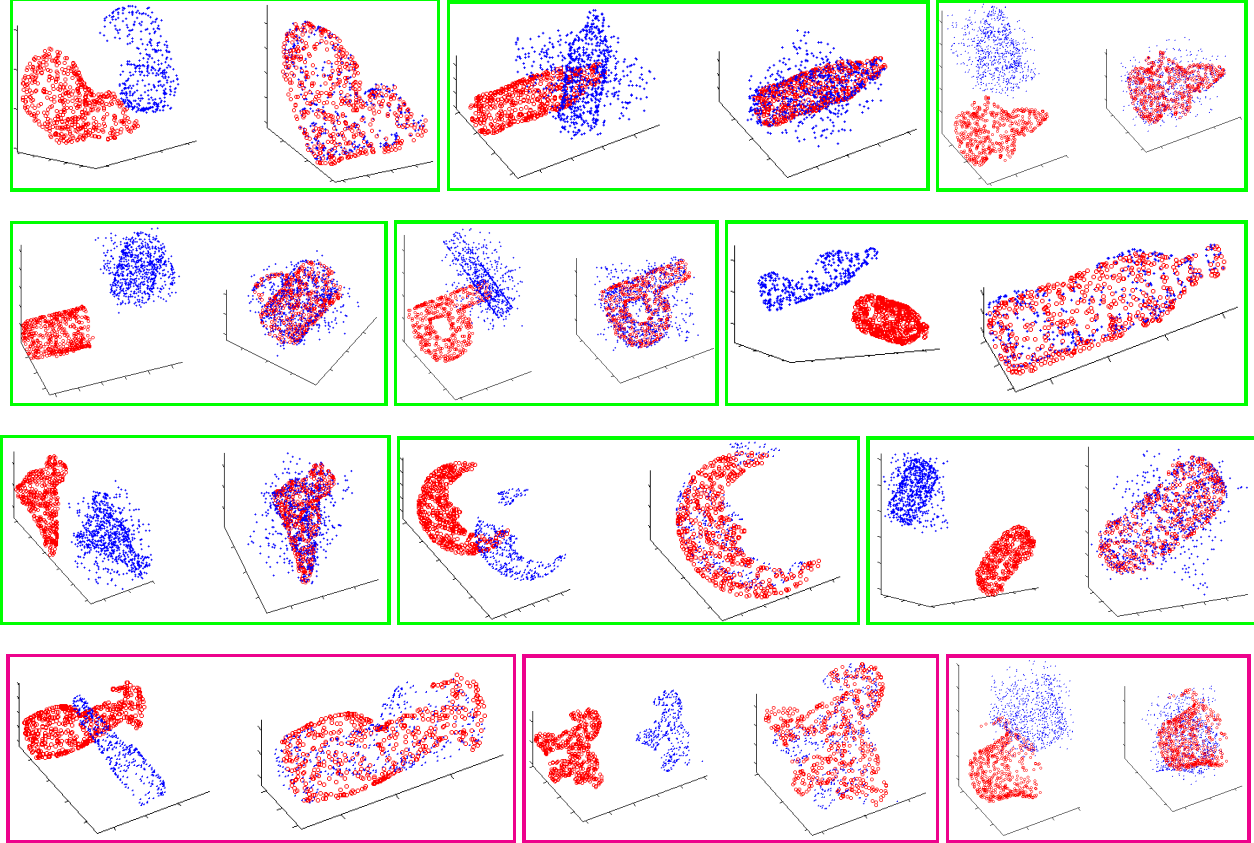


Figure 3. A sample set of qualitative test of the proposed algorithm on KIT database.

amount of outliers or missing points (registration results in magenta frames). The instability of the performance stems from the fact that the algorithm is likely to stuck into local optimum.

5.2. Quantitative Experiments

To obtain a more precise evaluation of the proposed algorithm, we conduct several quantitative experiments with different motion scales, outliers and missing portions for test. In addition, we also implement and run ICP and Jian’s method [11] for comparison. To ensure the fairness of the comparison, the same $SE(3)$ optimization strategy and stop criterion are used for their corresponding objective functions. KIT database is used as well here for quantitative evaluation and the point clouds are generated in the same way as section 5.1.

First, we test the robustness of three algorithms on different scales of motions. In this experiment, for motion scale i , the rotation angles of yaw, pitch and roll are $i \times [20^\circ, 4^\circ, 4^\circ]$, and translations are $i \times [S_x, S_y, S_z]$, where $[S_x, S_y, S_z]$ are standard deviations of point clouds in three axes. Different motions are applied on the point cloud of each object (points cloud is sampled with size 1000) to generate a tar-

get point cloud to align with. Since we know the correspondence between the original and target point clouds, the error for each registration is computed as the average distance between every pair of corresponding points in two point clouds. The test result on the database is plotted in Figure 4(a). We can see that ICP method is very sensitive to the initial motion. Actually average error of ICP method is monotonic increasing with respect to motion scale. Jian’s method and our method work in almost the same pattern. Both method work rather accurately when the motion scale $i \leq 4$ (rotation angle is smaller than 90°). However, both methods will fall into the local minimum (reflectional symmetry pose) when the motion scale increase ever since.

Secondly, we test the robustness of three algorithms by adding different portion (the percentage of point cloud size) of outliers. Outliers are generated by randomly sampling within the space around objects. For portion i , the number of added outliers is $i \times 10\%$ of the point-cloud size. The generated outliers are concatenated into the original point clouds, so the correspondence ground-truth is still available and the registration error is computed in the same way as in the motion experiment. To avoid the effect of large motion, a relatively small motion (motion scale $i = 1$) is ap-

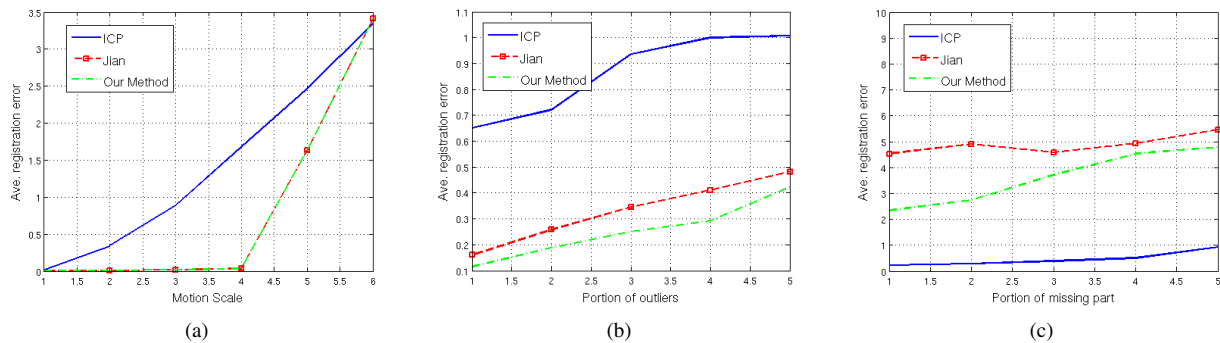


Figure 4. Performance comparison of three registration methods on the KIT database with different motion scales, portion of outliers and portion of missing part respectively.

plied on point clouds. The test result is plotted in Figure 4(b), from which we can see that ICP method is very fragile when the outliers are present even the amount is small. By contrast, since structure of point clouds are globally modelled in Jian’s method and our method, they are much more robust to the presence of outliers. In addition, due to the local estimated covariance in our method, it can outperform Jian’s method for all portion i .

Finally, part missing is an usual case because of occlusion. Here we also compare the performance of three algorithms on different portion of missing part. A missing part is selected by first randomly picking a point and then removing all points which lie within the neighbourhood of certain range. For portion i , the number of eliminated outliers is $i \times 6\%$ of the point-cloud size. Since the seed point is randomly selected, the correspondence ground truth will not be available between the original point cloud and target partial point cloud. To measure the registration accuracy, the full target point cloud is preserved before a random part is removed, and the registration error is computed as the distance between full original point cloud and full target point cloud. Similarly, a relatively small motion (motion scale $i = 1$) is applied on point clouds to avoid the effect of large motion. The test result is displayed in Figure 4(c). It can be seen that the performance of ICP method is most stable in missing cases. As for Jian’s method and our method, unfortunately, the global structure will be greatly changed when a random part is removed, neither of their performance is acceptable although our method is better than Jian’s method by making use of local structure information embedded in the local estimated covariance. However, it should be reminded that ICP method can outperform two others in missing cases only under the condition that the initial motion is small.

CONCLUSION

We present a full study on how probability product kernel function can be used for 3D point cloud registration. A

general registration framework is developed by incorporating $SE(N)$ on manifold optimization strategy. According to empirical test, the proposed registration algorithm is accurate and robust in many challenging cases. However, even with the help of local estimated covariance, the performance of our algorithm is still unacceptable when certain part of point cloud is missing, which points to the future direction of our research.

References

- [1] P. J. Besl and H. D. McKay. A method for registration of 3-D shapes. *PAMI*, 14(2):239–256, 1992. 1, 2
- [2] J. L. Blanco. A Tutorial on $SE(3)$ Transformation Parameterizations and on-Manifold Optimization. Technical report, University of Malaga, Sept. 2010. 4
- [3] M. Carcassoni and E. R. Hancock. Spectral Correspondence for Point Pattern Matching. *Pattern Recognition*, 36(1):193–204, 2003. 2
- [4] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek. The Trimmed Iterative Closest Point Algorithm. In *ICPR*, pages 545–548, 2002. 2
- [5] R. Detry and J. Piater. Continuous surface-point distributions for 3D object pose estimation and recognition. In *ACCV*, volume 6494 of *LNCS*, pages 572–585, Heidelberg, 2010. Springer. 3
- [6] R. Detry, N. Pugeault, and J. Piater. A Probabilistic Framework for 3D Visual Object Representation. *PAMI*, 31(10):1790–1803, 10 2009. 3
- [7] J. Feldmar, N. Ayache, and F. Betting. 3D-2D projective registration of free-form curves and surfaces. *Computer Vision and Image Understanding*, 65:403–424, 1997. 2
- [8] G. Godin, M. Rioux, and R. Baribeau. Three-dimensional registration using range and intensity information. In *Proceedings of the SPIE: Videometrics III*, volume 2350, pages 279–290, Boston, Massachusetts, USA, Nov. 1994. 2
- [9] S. Gold, A. Rangarajan, C. ping Lu, and E. Mjolsness. New Algorithms for 2D and 3D Point Matching: Pose Estimation and Correspondence. *Pattern Recognition*, 31:957–964, 1997. 2

- [10] T. Jebara, R. Kondor, and A. Howard. Probability Product Kernels. *Journal of Machine Learning Research*, 5:819–844, July 2004. [3](#)
- [11] B. Jian and B. C. Vemuri. Robust Point Set Registration Using Gaussian Mixture Models. *PAMI*, 33(8):1633–1645, 2011. [2](#), [3](#), [4](#), [6](#)
- [12] A. Johnson and S. B. Kang. Registration and Integration of Textured 3-D Data. In *Image and Vision Computing*, pages 234–241, 1996. [2](#)
- [13] A. Kasper, Z. Xue, and R. Dillmann. The KIT object models database: An object model database for object recognition, localization and manipulation in service robotics. *The International Journal of Robotics Research*, May 2012. [5](#)
- [14] R. Kondor and T. Jebara. A kernel between sets of vectors. In *ICML*, 2003. [3](#)
- [15] Y. Liu, L. Li, and B. Wei. 3D shape matching using collinearity constraint. In *ICRA*, pages 2285–2290, 2004. [2](#)
- [16] S. Lyu. A Kernel Between Unordered Sets of Data: The Gaussian Mixture Approach. In *ECML*, pages 255–267, 2005. [3](#)
- [17] T. Pajdla and L. V. Gool. Matching of 3-D Curves using Semi-differential Invariants. In *ICCV*, pages 390–395. IEEE Computer Society Press, 1995. [2](#)
- [18] M. D. Plumbley. Lie group methods for optimization with orthogonality constraints. In *Int. Conf. on Independent Component Analysis and Blind Signal Separation*, pages 1245–1252. Springer, 2004. [4](#)
- [19] C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. [3](#)
- [20] S. Rusinkiewicz and M. Levoy. Efficient Variants of the ICP Algorithm. In *3DIM*, pages 145–152, 2001. [2](#)
- [21] R. B. Rusu, Z. C. Marton, N. Blodow, and M. Beetz. Persistent Point Feature Histograms for 3D Point Clouds. In *Int. Conf. on Intelligent Autonomous Systems*, page 477. 2008. [2](#)
- [22] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, 2004. [2](#)
- [23] R. Yang and P. K. Allen. Registering, Integrating, and Building CAD Models from Range Data. In *ICRA*, pages 3115–3120, 1998. [2](#)
- [24] L. Zhang, S.-I. Choi, and S.-Y. Park. Robust ICP Registration Using Biunique Correspondence. In *3DIMPVT*, pages 80–85. IEEE, 2011. [2](#)
- [25] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994. [2](#)