Project no.:	**027657**
Project full title:	**Perception, Action & Cognition through learning of Object-Action Complexes**
Project Acronym:	**PACO-PLUS**
Deliverable no.:	**D4.1.5**
Title of the deliverable:	**Report on learning of object parts and part-action associations**

Contractual Date of Delivery to the CEC: 31.1.2010
Actual Date of Delivery to the CEC: 31.1.2010
Organisation name of lead contractor for this deliverable: SDU/ULg
Author(s): Leon Bodenhagen, Norbert Krüger, Justus Piater, Renaud Detry, Emre Baseski, Pascal Haazebroek, Bernhard Hommel, Saskia van Dantzig, Florentin Wörgötter, Kai Huebner, Danica Kragic, Alexander Bierbaum, Tamim Asfour, Rüdiger Dillmann
Participant(s): UL, SDU, ULg, BCCN, KTH, UniKarl
Work package contributing to the deliverable: WP1,WP2,WP4
Nature: R
Version: Final
Total number of pages: 11
Start date of project: 1st Feb. 2006 **Duration:** 48 month

Project co-funded by the European Commission within the Sixth Framework Programme (2002–2006)
Dissemination Level

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Abstract:

We report on work on the learning and utilization of the decomposition of objects into parts performed by the PACO-PLUS consortium in WP4.1 in months 37 – 48. We address the tasks object recognition, affordance learning (in particular grasping and haptic exploration) as well as semantic scene interpretation of action sequences. In this context, we have also developed a biological model to associate feature codes (parts) to actions. This deliverable covers 9 publications [A, B, C, D, E, F, G, H, I] (some of them in the status submitted).

Keyword list: Object-Part Associations, Grasping

Table of Contents

1. Introduction

The decomposition of scenes and objects into parts is important to formalize generalization processes across objects in contexts such as object recognition and affordance learning. Here parts are understood as re-occuring structures of lower complexity than objects but higher complexity than individual entities such as individual edges, contours or surfaces. They allow for structuring the scene and to generalize actions associated to parts across objects. Also we note that parts are not necessarily only visual entities but the concept of parts can also be applied to tactile or auditory information.

We have approached the part learning problem on multiple levels. For object recognition, we have realised an incremental learning process in which a multi-part, viewpoint-invariant model is generated from a set of roughly segmented, unregistered views, by sequentially registering and fusing the views with the incremental model (Section 2.1). Moreover, we have made use of parts based on contours related by their 3D geometry and appearance to perform object recognition (Section 2.2).

Based on grasping data gained by exploration (see D4.1.4 and [2]), we have made statistical investigations on regularities of grasp-part associations (Section 3). For this, we again made use of contour relations as used also for object recognition in Section 2.2. Complementarily to a contour based approach, we also made use of 3D surface primitives (box decomposition) for affordance learning (Section 4). Haptic part grasp associations are learned by detailed physical simulations as described in Section 5.

The decomposition of complex action sequences in terms of spatial-temporal events is done in Section 6. Re-occuring spatial-temporal events are then used to interpret action sequences and find commonalities between those. Finally, in Section 7, two computational models are presented which are based on cognitive theories of feature-related learning. The first model focuses on learning how actions are associated with the perceptual features of their effects (action-effect learning). The second model focuses on how experiences are stored into long-term memory to form mental concepts that represent particular categories of objects or actions (concept learning).

2. Using Parts for Object Recognition

For the task object recognition parts are learned based on two different feature types (point clouds in Section 2.1 and multi–modal contours in Section 2.2).

2.1 Probablistic Part Learning

In [G] we present a 3D, probabilistic object-surface model, along with mechanisms for constructing a model from unregistered 2.5D views and segmenting model instances in cluttered scenes. We model object parts probabilistically with smooth surface-point distributions obtained through kernel density estimation on 3D point clouds. A multi-part, viewpoint-invariant model is learned incrementally from a set of roughly segmented, unregistered views, by sequentially registering and fusing the views with the incremental model (examples are shown in Figure 1). Registration is conducted by nonparametric inference of maximum-likelihood model parameters, using Metropolis-Hastings Markov chain Monte Carlo methods (MCMC) with simulated annealing. This mechanism is robust to clutter, and avoids direct model-to-scene correspondences. The learning of viewpoint-invariant models and the applicability of our method to pose estimation, object detection, and object recognition is demonstrated on 3D-scan data, providing qualitative, quantitative and comparative evaluations.

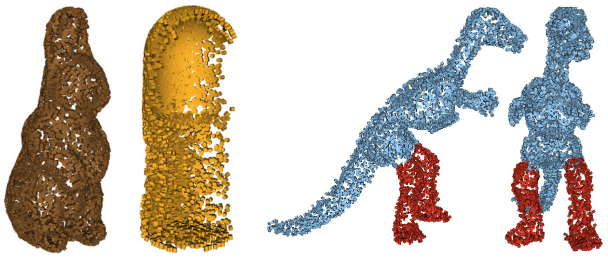


Figure 1: Object points obtained from the registration of sequences of 66 views. Color indicates learned parts; the bunny and deodorant bottle are made up of a single part, whereas the dinosaur yields a two-part model (legs and body).

2.2 Using feature relations for Object Recognition

In [C] the use of second-order relations between local and semi-global 3D edge features for object identification in visual scenes is investigated. Relations between 3D features have the inherent property of being pose-invariant, and therefore allow for direct comparison without knowledge of object pose. We define histograms of relations such as coplanarity, distance, and color difference between local 3D edge descriptors and semi-global groups thereof, and use them to encode higher–level object structure corresponding to parts (see Figure 2). Histogram intersection is then used to identify a set of low-textured objects that are poorly described by classical feature descriptors. Moreover, the relation histograms are shown to describe structural properties of the objects, and the histogram similarities between objects reflect structural similarities between them.

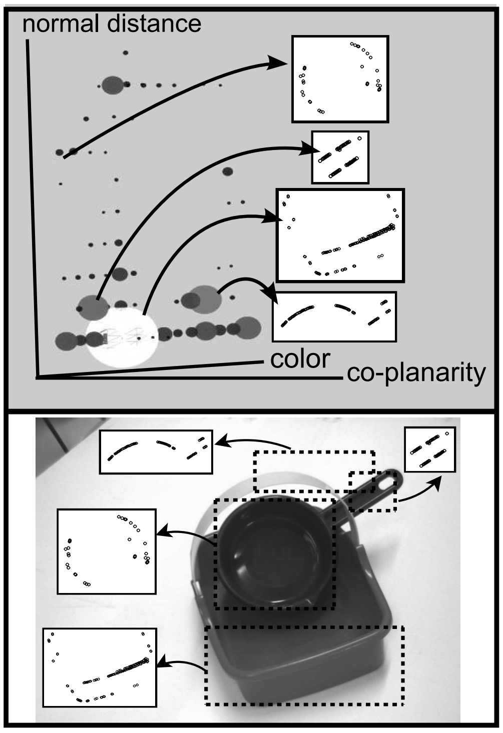


Figure 2: An example for a 3-dimensional histogram of primitive relations (top) and the primitives that created specific bins of the histogram (bottom). The blobs represent the locations of histogram bins. Their brightness and size are proportional to the value stored in the bins (a high count is represented as a big and bright blob).

3. Learning feature relations Grasp Association

In [E] we investigate what feature relations trigger grasps which are likely to be successful (P1) and how a complete set of grasping affordances *for a concrete object* can be learned in a fast way during an active exploration process making use of *generic grasp knowledge* (P2). The two problems stated above are closely related since knowledge about feature-relations and associated grasp success likelihoods can be used in an active exploration process in which knowledge about concrete grasping experience is available. For both problems we utilize the evaluated grasps which have been recorded in [2]. This provides us with large sets of evaluated grasps for three different objects for which accumulated representations are available.

The first problem (P1) is addressable in a relatively straightforward way. Given two visual features F_1, F_2, we define a predicted grasp: $predictGrasp(F_1, F_2)$. For all combinations of features (3D contours in this context) grasps are predicted and their likelihood for being successful is estimated by comparing them with the tried grasps — we assume that when the predicted grasp has a similar pose like a tried one, it is likely to be successful. Subsequently it is investigated if certain relations between the contours give an indication of successful predictions.

The second problem (P2) is more complex as it involves the investigation of the impact of a variation of the 6D pose of a grasp on its likelihood for being successful. Therefore we define two subproblems, P2.1 and P2.2. In P2.1 we do initial investigations of the impact of small changes of the pose of a tried grasp. This may lead to the replacement of the isotropic kernel used for the learning of object specific grasp densities in [1] with an anisotropic, learned one.

In P2.2 we compute the global impact of grasping experiences depending on concrete relations between the 3D contours of the object model. We address this problem by transferring a physically tried grasp from its associated contour to another contour. Subsequently we investigate which relations between contours can indicate if such a transferred grasp is still likely to be successful.

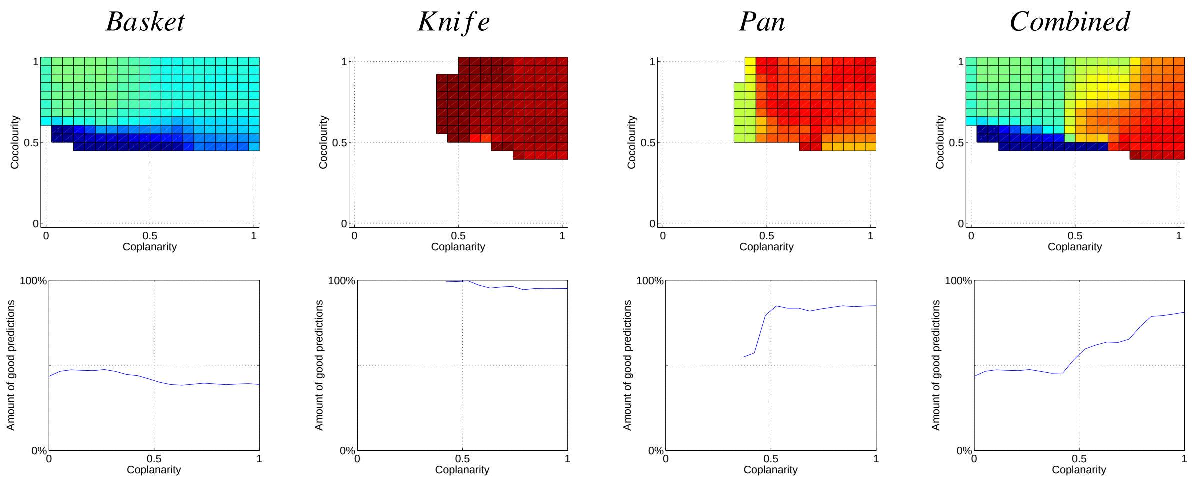


Figure 3: Top row shows amount of good predictions with respect to coplanarity and cocolourity, bottom row only for coplanarity. The fourth column is the sum over all objects, giving each object the same overall impact.

First results give already an indication that an anisotropic kernel is suitable for grasp densities (P1) and that certain relations, e.g. coplanarity, are important (Figure 3). At the same time it has been discovered that cocolourity for instance is problematic in the context of object representations which have been accumulated over multiple frames (see [6]). Still these results also clearly show that more data is required in order to cover the space defined by the relations. We therefore plan to acquire additional grasp data using a dynamics simulator.

4. Part Action Association using box primitives

It has been observed in the literature that the approximation of 3D data by shape primitives, e.g. spheres, boxes or cones, is a very valuable step for the purpose of grasping. In this context, a focus on a simple and efficient box approximation technique has further proven to be meaningful for connecting such shape information with pre-grasp configurations, as proposed in [4]. The output of such an approximation can be interpreted as a part-description of an object. While its geometric simplicity — a collection of boxes — not only allows for a tremendous reduction of possible grasp configurations on the object through heuristic reasoning, it also enables the connection of such representations to successful grasps by learning of contact-level grasp qualities in the force domain. We use simulation as a helpful tool for learning and evaluating the stability of grasps. The system embodiment may lead to different strategies to grasp different parts of the object. Figure 4 exemplifies this: a spherical grasp allows grasps on the slim part of a screwdriver, while a parallel jaw grasp is more reasonable for grasps on the handle part. A more extensive description of the whole framework can be found in a confidential journal submission attached to this report [I].

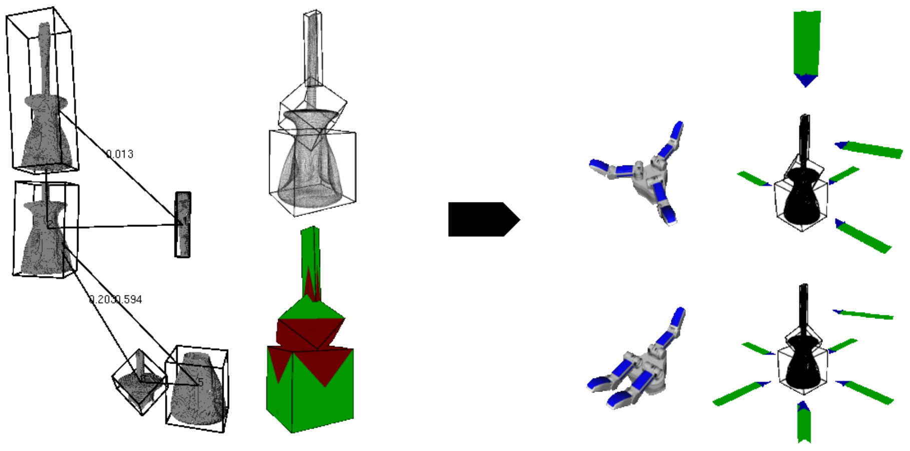


Figure 4: Left: Hierarchical box approximation and decomposition of dense 3D data. Boxes correspond to intuitive parts, and facets of boxes will spawn grasp hypotheses (grasp approach and orientation). Geometrical heuristics reduce the hypotheses set (green = valid; red = invalid). Right: Dependent on the pre-shape, a different set of grasp hypotheses is found to be stable force-closure grasps.

5. Haptic Part–Grasp associations

In [D] we address the problem of tactile exploration and subsequent extraction of grasp hypotheses for unknown objects with a multi-fingered anthropomorphic robot hand. We present extensions on our tactile exploration strategy for unknown objects based on a dynamic potential field approach resulting in selective exploration in regions of interest. In the subsequent feature extraction, faces found in the object model are considered to generate grasp affordances. Candidate grasps are validated in a four-stage filtering pipeline to eliminate impossible grasps. To evaluate our approach, experiments were carried out in a detailed physics simulation using models of the five-finger hand and the test objects. One example for the simulation results is shown in Figure 5.

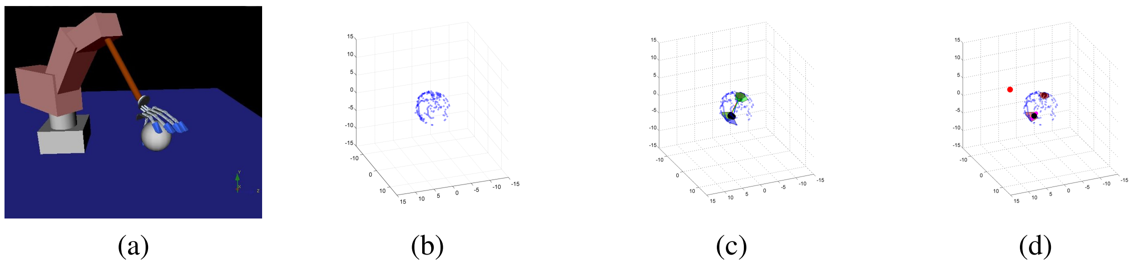


Figure 5: Typical simulation results for a sphere object. Column (a) shows a virtual scene snapshot during exploration, (b) final point cloud, (c) grasp affordances, (d) best grasp and grasping points.

6. Object–Action Relations from Semantic Scene Graphs

In [A, B] we introduce a novel real-time framework for model-free stereo-video segmentation and stereo-segment tracking, combining real-time optical flow and stereo with image segmentation running separately on two GPUs. The computed stereo segments are used to construct 3D segment graphs, from which main graphs, representing a relevant change in the scene, are extracted via an exact graph-matching technique, thus providing an event table of the action scene, which allows for the extraction of object-action relations. The central novelty of this framework is that it is model free and does not need an a-priori representation neither for objects nor actions. Essentially actions are recognized without requiring prior object knowledge and objects are categorized solely based on their exhibited role within an action sequence. The method has potential applications for recognition and categorization of human-activities for example in imitation learning, in developmental and cognitive robotics reaching beyond PACO-PLUS.

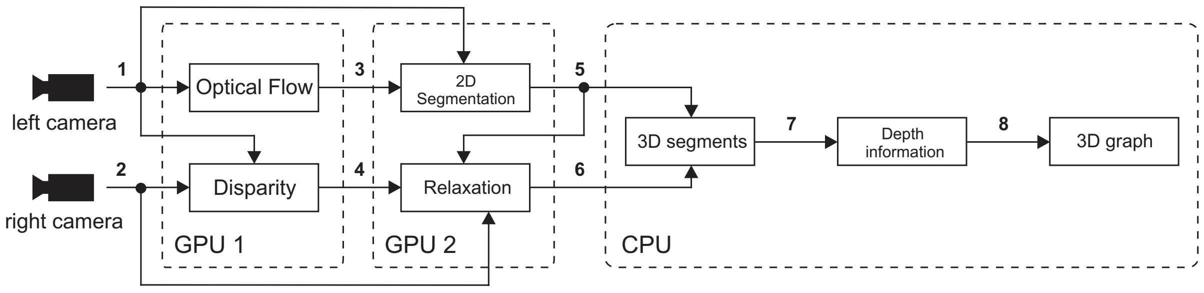


Figure 6: The architecture of the 3D segment tracking framework.

The architecture of the 3D segment tracking framework is shown in Figure 6. It consists of a stereo camera, two GPUs, one CPU, and various processing components that are connected by channels in the framework. Output data of all components can be accessed from any component in the framework. The left and right images from a stereo camera enter into the framework via channels 1 and 2 respectively. Optical flow and disparity are computed on GPU 1 using a real-time algorithm, and the results are accessible from channels 3 and 4, respectively. For the images from the left camera, the labels from a previous segmentation are warped to the current frame using optical flow (channel 3). The new label configuration is used as an initialization for the real-time segmentation algorithm running on GPU 2. This way, the required time for label relaxation can be reduced, and, even more importantly, a consistent labeling of the frames can be achieved, i.e. segments describing the same object part are likely to carry the same label (segment tracking). The results of the segmentation can be accessed from channel 5 and used to compute the label initialization for the segmentation of the right frame. This time, the labels are warped using phase-based disparity information obtained from channel 4. The segmentation result of the right image, which is now consistently labeled with respect to the images from the left camera, is stored in channel 6. Segments larger than a predefined threshold are extracted and stored in channel 7. The stereo-segment correspondences are used to find the 3D structure of the segments and stored in channel 8. Once 3D segments and depth information are extracted, we represent the scene by undirected and unlabeled semantic graphs. The graph nodes are the segment labels and plotted at the center of each segment. The nodes are then connected by an edge if the segments are neighbors and their depth differences are below a predefined threshold value. We also define a 3D field of view boundary and ignore segments whose depth value exceeds this boundary. With this method we create a focus of attention and ignore objects outside the 3D boundary. For 3D segmentation and graph results of a real image sequence, see Figure 7.

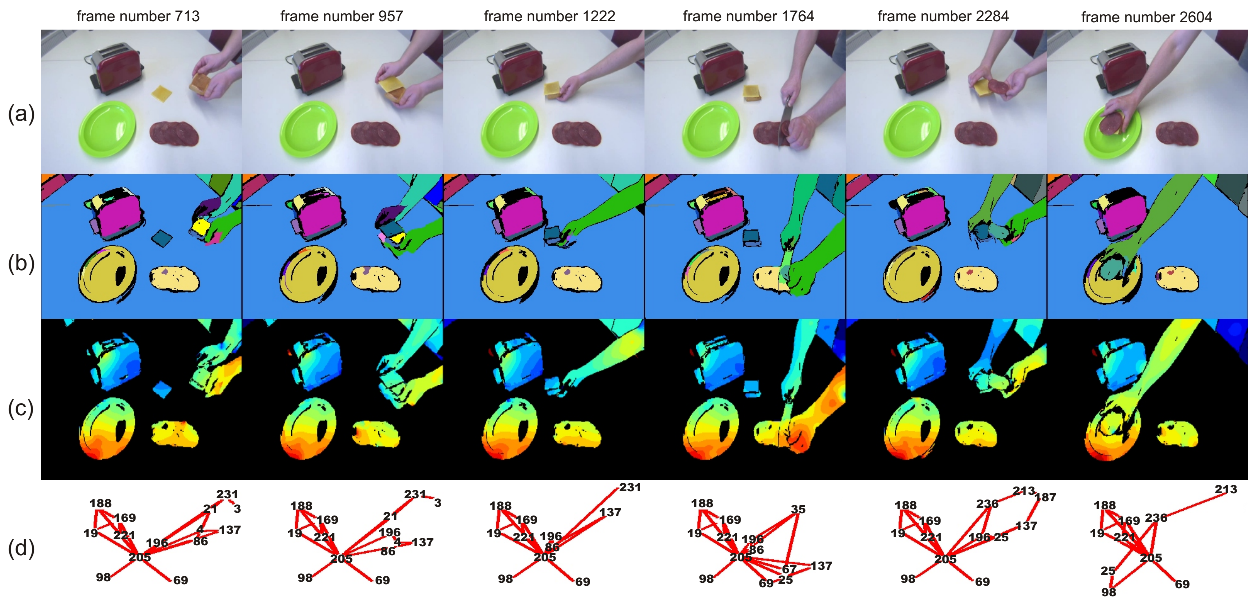


Figure 7: Results for the interleaved sample actions, i.e. “Cutting a salami”, “Making a sandwich”, and “Putting on a plate”. (a) Original frames from the left image sequence. (b) Extracted segments for frames of the left sequence. (c) The dense disparity maps obtained for extracted stereo segments. The disparity values are color-coded from blue (small) to red (large). Areas of low confidence are colored black, i.e., the uniform and untextured area of the table, for which only poor disparity results could be obtained. (d) Final 3D semantic scene graphs, representing action primitives.

In the temporal domain, 3D scene graphs represent spatial relations between nodes. Unless spatial relations change, the scene graphs remain topologically the same. The only changes in the graph structures are the node positions or the edge lengths depending on the object trajectory and speed. Consequently, any change in the spatial relation between nodes corresponds to a change in the main structure of the scene graphs. Therefore, those changes in the graphs can be employed to define action primitives. Considering this fact, we apply an exact graph-matching method in order to extract the main graphs by computing the eigenvalues and eigenvectors of the adjacency matrices of the 3D graphs. A change in the eigenvalues or eigenvectors then corresponds to a structural change of the graph. The temporal order by which those main graphs follow each other defines an “event table”. An event signifies that something has happened in the scene which caused a true topological change in the graph. This method allows classifying object-action relations by calculating the similarity between event tables from different scenes. Furthermore, nodes playing the same role in a classified action sequence can be identified and then be used to categorize objects by returning to the signal level via image segments.

The developed framework is applied to a real stereo-image sequence that consists of interleaved chained actions, i.e., “Cutting a salami”, “Making a sandwich”, and “Putting on a plate” (see Figure 7(a)). In this sequence two arms are first taking bread from a toaster, putting a piece of a cheese on it, and then cutting off a slice of salami with a knife. After putting the salami on top of the cheese, the sandwich is being placed on a plate and the arms are leaving the scene. Image segments of some sample frames from the left sequences are given in Figure 7(b). Dense disparity maps obtained for extracted stereo segments are given in Figure 7(c). The low-confidence-value area of the table segment is depicted with a black color in the dense disparity maps. Figure 7(d) illustrates some 3D semantic main graphs. The graphs show that all relevant object parts in the scene can be represented by unique labels and tracked during the whole image sequence. Moreover, each graph defines a topological change in the scene.

7. Feature-related learning

According to the Theory of Event Coding (TEC, proposed in [3]) perceptual stimuli and motor actions are represented by distributed patterns of features, so called event files. Importantly, percepts and actions are coded in a common representational domain. In other words, the same features can be used to represent perceptual stimuli and motor actions. This common coding principle allows for direct interactions between perceptual and action related processes, which may also be mediated by the task context.

In our work, we do not focus on the question how features are extracted from the raw sensory data. Instead, we focus on how feature codes (parts) are associated to actions (action-effect learning) and how event files may be stored in long term memory to form concepts that represent particular categories of objects or actions (concept learning).

7.1 Action-effect learning

In [H], we have developed HiTEC, a computational model based on TEC. In this model (see Figure 8) we differentiate between modality specific sensory codes (e.g., visual shape or color), abstract motor representations (e.g., coding for particular motor actions), common feature codes coding for amodal cognitive features (e.g., the generic feature left), and task codes that code for different decision alternatives. By associating feature codes with task codes, we encode the task rules. Crucially, we assume that motor actions get their cognitive meaning through their perceptual effects. Therefore, we first let the model perform random actions and associate them with their perceptual effects. In the experimental stage, we let the model respond to stimuli (using the associations between feature codes and task codes) by anticipating the desired effect and by propagating the activation (using the learned associations between feature codes and motor codes) towards the motor codes. Associations between feature codes and task codes are set by hand, following the task instruction. Associations between feature codes and motor codes are learned during training trials.

7.2 Concept learning

In [F] we have also developed CONCAT, a connectionist model that forms hierarchical representations of concepts and contexts. These representations are learned by extracting the statistical regularities over repeated experiences. Regularities can be detected at different temporal and spatial scales. At a small scale, features that co-occur in time and space are bound into concept representations. At a coarser scale, concepts that co-occur in time and space are bound into context representations. Learning of concepts and contexts is achieved by two Categorizing and Learning Modules (CALM; proposed by [5]). One important feature of CALM modules is that they display novelty-dependent learning. When a novel input pattern is presented, elaboration learning takes place, which is characterized by a high learning rate and strong competition between the uncommitted nodes (those nodes that do not yet represent a pattern). When a

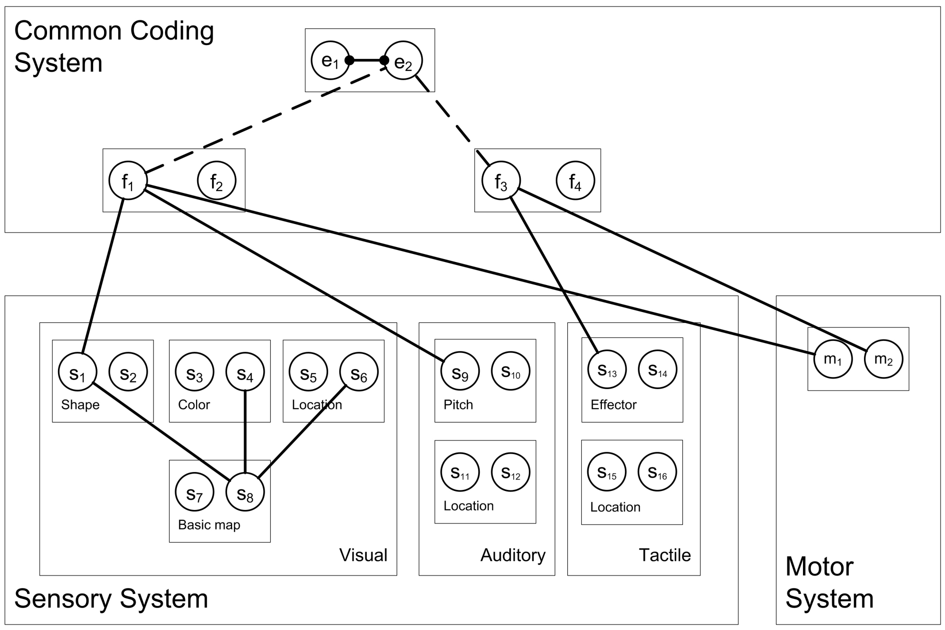


Figure 8: HiTEC architecture, based on the Theory of Event Coding.

familiar input pattern is presented, activation learning takes place, characterized by a low learning rate and weak competition between nodes.

8. Links to other Workpackages

Deliverable D4.1.5 is linked to and makes use of work made in a number of workpackages. It is linked to the software and hardware integration issues dealt with in WP1 and the sub-modules developed in WP2. Since it describes meta–learning processes, results presented here are only indirectly used in WP8.

Attached Papers

[A] Alexey Abramov, Eren Erdal Aksoy, Johannes Dörr, Karl Pauwels, Babette Dellen, and Florentin Wörgötter. 3D Semantic Representation of Actions from efficient stereo-image-sequence segmentation on GPUs. In *Fifth International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, (submitted).

[B] Eren Erdal Aksoy, Alexey Abramov, Florentin Wörgötter, and Babette Dellen. Categorizing object-action relations from semantic scene graphs. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2010 (accepted).

[C] Emre Başeski, Nicolas Pugeault, Justus Piater, and Norbert Krüger. 3D edge and contour relation histograms for object structure encoding and identification. *Pattern Recognition Letters*, (submitted).

[D] A. Bierbaum, M. Rambow, T. Asfour, and R. Dillmann. Grasp affordances from multi-fingered tactile exploration using dynamic potential field. In *IEEE/RAS International Conference on Humanoid Robots*, Humanoids 2009.

[E] Leon Bodenhagen, Emre Baseski, Dirk Kraft, Justus Piater, and Norbert Krüger. Initial investigation of local and global grasp transferability utilizing visual feature relations. Technical report, Robotics Group, Maersk Institute, University of Southern Denmark, 2010.

[F] S. Van Dantzig, A. Raffone, and B. Hommel. Grounded concept and context learning; a connectionist approach. In *(Under review)*, 2010.

[G] Renaud Detry and Justus Piater. Continuous surface-point distributions for 3D object pose estimation and recognition. In *IEEE Computer Science Conference on Computer Vision and Pattern Recognition (CVPR)*, (submitted).

[H] P. Haazebroek and B. Hommel. *Anticipatory Behavior in Adaptive Learning Systems*, chapter Anticipative control of voluntary action: Towards a computational model, pages 31–47. Springer-Verlag Berlin Heidelberg, 2009.

[I] Kai Huebner and Danica Kragic. Grasping by parts: Robot grasp generation from 3D box primitives. *Submitted to Advanced Robotics*.

References

[1] Renaud Detry, Emre Başeski, Norbert Krüger, Mila Popović, Younes Touati, and Justus Piater. Autonomous learning of object-specific grasp affordance densities. In *Approaches to Sensorimotor Learning on Humanoid Robots (Workshop at the IEEE International Conference on Robotics and Automation)*, 2009.

[2] Renaud Detry, Dirk Kraft, Anders Glent Buch, Norbert Krüger, and Justus Piater. Refining grasp affordance models by experience. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2010. (accepted).

[3] B. Hommel, J. Müsseler, G. Aschersleben, and W. Prinz. The theory of event coding (TEC): a framework for perception and action planning. *The Behavioral and brain sciences*, 24:849–878, 2001.

[4] Kai Huebner and Danica Kragic. Selection of robot pre-grasp using box-based shape approximation. In *IEEE Conference on Intelligent Robots and Systems*, pages 1765–1770, 2008.

[5] J.M.J. Murre, R. H. Phaf, and G. Wolters. CALM: Categorizing and learning module. *Neural Networks*, 5:55–82, 1992.

[6] N. Pugeault, F. Wörgötter, and N. Krüger. Accumulated visual representation for cognitive vision. In *BMVC*, 2008.

3D Semantic Representation of Actions from efficient stereo-image-sequence segmentation on GPUs

Alexey Abramov Eren Erdal Aksoy Johannes Dörr
Florentin Wörgötter
Georg-August University, BCCN Göttingen, III Physikalisches Institut
Göttingen, Germany
{abramov,eaksoye,jdoerr,worgott}@bccn-goettingen.de

Karl Pauwels
Laboratorium voor Neuro- en Psychofysiologie
K.U.Leuven, Belgium
karl.pauwels@med.kuleuven.de

Babette Dellen
BCCN Göttingen, Max-Planck-Institute for Dynamics and Self-Organization
Göttingen, Germany
Institut de Robòtica i Informàtica Industrial (CSIC-UPC)
Barcelona, Spain
bkdellen@bccn-goettingen.de

Abstract

A novel real-time framework for model-free stereo-video segmentation and stereo-segment tracking is presented, combining real-time optical flow and stereo with image segmentation running separately on two GPUs. The stereo-segment tracking algorithm achieves a frame rate of 23 Hz for regular videos with a frame size of 256×320 *pixels and nearly real time for stereo videos. The computed stereo segments are used to construct 3D segment graphs, from which main graphs, representing a relevant change in the scene, are extracted, which allow us to represent a movie of e.g. 396 original frames by only 12 graphs, each containing only a small number of nodes, providing a condensed description of the scene while preserving data-intrinsic semantics. Using this method, human activities, e.g., handling of objects, can be encoded in an efficient way. The method has potential applications for human-activity recognition and learning, and provides a vision-front end for applications in cognitive robotics.*

1. Introduction

Movies contain abundant information about the visual scene, which, if choosing the raw signals for representations, render the application of any logic or learning scheme intractable. This problem occurs for example if we want to understand and learn both courses and consequences of manipulations from visual data. A reduced representation of the scene is urgently required to make such problems tractable by algorithms operating on a small number of abstract descriptors (symbols). Finding this reduced representation without prior knowledge on the data (model free) thus represents a major challenge in cognitive-vision applications – this problem is also known as the signal-symbol gap [12].

In this paper, we present a novel framework for bridging this gap. We aim at creating a condensed description of stereo movies by computing the 3D relations between tracked image segments for generating 3D semantic graphs, which, in the future, will allow us to encode human actions in an efficient way. To achieve this goal, three main problems need to be solved: (i) Stereo images need to be segmented in real time in a consistent way and image segments need to be tracked along the movie, i.e., segments representing the same part of an object should carry the same label

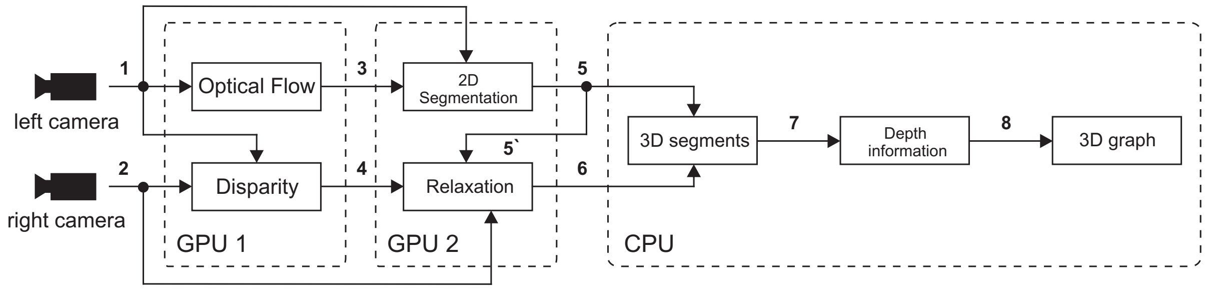


Figure 1. The architecture of 3D segment tracking framework.

all the time. Note that image segmentation represents a logical step towards our goal because redundant information is thereby grouped and condensed into higher level representational entities (segments). The method should be entirely data driven. (ii) The 3D relations of the segments need to be derived with sufficient accuracy, and graphs need to be constructed. Then, only the graphs representing a relevant cha nge in the scene should be extracted, further removing redundant information. (iii) The algorithms should run in real time or close to real-time[1] to allow the framework to be used for robotic applications.

Several approaches for video segmentation have been proposed in the past, where some methods rely on segmenting each frame independently, followed by a segment matching step based on their low-level features [5, 20, 23, 8], while other methods use motion projection to link segments [16, 28, 26, 14]. It was shown recently that image segments can be tracked along the frames of a movie in a model-free way, i.e. without assuming a data model of some kind, using the method of superparamagnetic clustering of data [6].

Real-time requirements render the video segmentation algorithms currently inadequate for most robotic applications. Furthermore, stereo movies have not been treated by any of these works. To overcome these limitations, we developed real-time model-free image segmentation on GPUs based on a novel parallel method, combined with real-time phase-based optical flow [17] and stereo [19], executed on GPU as well, for segment tracking.

The framework will potentially be applicable to a wide range of problems in the field of action and manipulation recognition and learning, and may serve as vision-front end in cognitive robots.

[1]By real-time we understand processing of a full frame at 25Hz or faster.

2. 3D segment tracking framework

2.1. Overview

The architecture of the 3D segment tracking framework is shown in Fig. 1. It consists of a stereo camera, two GPUs, one CPU, and various processing components that are connected by channels in the framework. Output data of all components can be accessed from any component in the framework. The left and right images from a stereo camera enter into the framework via channels 1 and 2 respectively. Optical flow and disparity are computed on GPU 1 using a real-time algorithm [17], and the results are accessible from channels 3 and 4 respectively (see Section 2.3). For the images from the left camera, the labels from a previous segmentation are warped to the current frame using optical flow (channel 3). The new label configuration is used as an initialization for the real-time segmentation algorithm running on GPU 2 (see Section 2.2-2.4). This way, the required time for label relaxation can be reduced, and, even more importantly, a consistent labeling of the frames can be achieved, i.e. segments describing the same object part are likely to carry the same label (segment tracking) (see Section 2.4). The results of the segmentation can be accessed from channel 5 and used to compute the label initialization for the segmentation of the right frame via channel $5'$ (see Section 2.5). This time, the labels are warped using phase-based disparity information obtained from channel 4. The segmentation result of the right image, which is now consistently labeled with respect to the images from the left camera, is stored in channel 6. Segments larger than a predefined threshold are extracted and stored in channel 7. The stereo-segment correspondences are used to find the 3D structure of the segments using a recent stereo method on the CPU [7] and stored in channel 8 (see Section 2.6). In a final step, relevant information about the 3D structure of the segments is extracted from the computed disparity map and used together with the segmentation results to construct an abstract 3D description of the scene, which is represented by undirected graphs in which nodes and edges represent 3D segments and their neighborhood relations (see Section 2.7).

2.2. Image segmentation algorithm

The method of superparamagnetic clustering solves the segmentation problem by finding the equilibrium states of the energy function of a ferromagnetic Potts model in the superparamagnetic phase [18, 11, 25, 9, 3, 15, 24]. The Potts model [18] describes a system of interacting granular ferromagnets or spins that can be in q different states, characterizing the pointing direction of the respective spin vectors. Three phases, depending on the system temperature, i.e. disorder introduced to the system, are observed: the paramagnetic, the superparamagnetic, and the ferromagnetic phase. In the ferromagnetic phase, all spins are aligned, while in the paramagnetic phase the system is in a state of complete disorder. In the superparamagnetic phase regions of aligned spins coexist. Blatt et al. (1998) applied the Potts model to the image segmentation problems in a way that in the superparamagnetic phase regions of aligned spins correspond to a natural partition of the image data [3]. Finding the image partition corresponds to the computation of the equilibrium states of the Potts model.

The equilibrium states of the Potts model have been approximated in the past using the Metropolis-Hastings algorithm with annealing [10] and methods based on cluster updating, which are known to accelerate the equilibration of the system by shortening the correlation times between distant spins, such as Swendsen-Wang [22], Wolff [27], and energy-based cluster updating (ECU) [15, 24]. All of these methods obey detailed balance, ensuring convergence of the system to the equilibrium state. Here we achieve efficient performance using the Metropolis algorithm with annealing [10], which can be easily parallelized and implemented on a GPU. The method further has the advantage that results from a previous segmentation can be utilized by the algorithm to find a consistent segmentation of the next frame within a small number of Metropolis updates only, drastically reducing computation time.

The real-time image segmentation algorithm proceeds as follows. In the Potts model, a spin variable σ_k, which can take on q discrete values $v_1, v_2, \dots, v_q$, called spin states, is assigned to each pixel of the image. The energy of the system is described by

$$E = -\sum_{<ij>} J_{ij}\delta_{ij} \quad , \tag{1}$$

with the Kronecker sign

$$\delta_{ij} = \begin{cases} 1 & \text{if } \sigma_i = \sigma_j, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

where σ_i and σ_j are the respective spin variables of two neighboring pixels i and j. The function

$$J_{ij} = 1 - |\mathbf{g_i} - \mathbf{g_j}|/\overline{\Delta} \tag{3}$$

is a coupling constant, determining the interaction strength, where $\mathbf{g_i}$ and $\mathbf{g_j}$ are the respective color vectors of the pixels, and

$$\overline{\Delta} = \alpha \cdot (\sum_{<i,j>} |\mathbf{g_i} - \mathbf{g_j}| / \sum_{<i,j>} 1) \tag{4}$$

computes the averaged color vector difference of all neighbors $< i, j >$. The factor $\alpha \in [0, 10]$ is a system parameter.

The Metropolis algorithm allows generating spin configurations S which obey the Boltzmann probability distribution [4]

$$P(S) \sim \exp\left[-\beta E(S)\right] \quad , \tag{5}$$

where $\beta = 1/kT$, T is the temperature parameter, and k is the Boltzmann constant.

Initially, values are assigned randomly to all spin variables. According to the Metropolis algorithm, each spin-update procedure consists of the following steps [13]:

1. The system energy E_A of the current spin configuration S_A is computed according to Eq. 1.

2. A pixel i with spin variable σ_i in spin state v_l is selected and for each possible move to a new spin state $\sigma_i \neq v_l$ the energy E_B of the resulting new spin configuration S_B is computed according to Eq. 1. The number of possible moves is $(q-1)$.

3. Among all new possible configurations we find the configuration with the minimum energy

$$E_{new} = \min(E_1, E_2, \dots, E_{q-1}) \quad , \tag{6}$$

and compute the respective change in energy

$$\Delta E = E_{new} - E_A \quad . \tag{7}$$

4. If the total energy of the configuration is decreased by this move, i.e. $\Delta E < 0$, the move is always accepted.

5. If the energy increased, i.e. $\Delta E > 0$, the probability that the proposed move will be accepted is given by

$$P_{A \to B} = \exp\left(-\frac{|\Delta E|}{kT_n}\right) \quad , \tag{8}$$

and

$$T_{n+1} = \gamma T_n \qquad \gamma < 1 \quad , \tag{9}$$

where γ is the annealing coefficient. We draw a number ξ randomly from a uniform distribution in the range of $[0, 1]$. If $\xi < P_{A \to B}$, the move is accepted.

Each spin update involves only the nearest neighbors of the considered pixel. Hence, spin variables of pixels that are not neighbors of each other can be updated simultaneously [2]. Therefore the Metropolis algorithm fits very well to the GPU architecture. For the first frame of the image sequence, a short-cut via a pre-segmentation step, is used to allow sufficiently fast segmentation. More details can be found in [1]. In the experiments we will use a large number of spins, which then serve as a segment label. Note that these wordings are thus equivalent here.

2.3. Phase-based optical flow and disparity

Since fast processing is a very important issue in our work, we use the GPU-based real-time optical flow algorithm proposed by Pauwels et al. (2008) [17]. This algorithm belongs to the class of phase-based techniques, which are highly robust to changes in contrast, orientation and speed. This particular algorithm integrates the temporal phase gradient (extracted from five subsequent frames) across orientation and gradually refines its estimates by traversing a Gabor pyramid from coarser to finer levels. In our framework optical flow is computed for the left video stream only. The algorithm provides a vector, at each pixel indicating its motion

$$\mathbf{u}(\mathbf{x},\mathbf{y}) = (u_x(x,y), u_y(x,y)) \quad , \tag{10}$$

This allows us to link pixels of two subsequent frames $\{t\}$ and $\{t+1\}$ (see Fig. 2).

In order to link pixels of correspondent left and right frames displacements for correspondent left and right pixels have to be estimated. For this purpose the disparity algorithm can be used. Our system relies on rectified images and therefore only horizontal disparities need to be determined. The disparity algorithm that is used in our framework is also phase-based and operates on phase differences between the left and right image as opposed to temporal phase gradients in optical flow algorithm. It is described in more detail in [19].

2.4. Image-sequence segmentation using optical flow based label warping

We obtain information about pixel correspondences between subsequent frames via a phase-based optical flow algorithm applied to the frame sequence, as described in 2.3. Since we are using a local algorithm, optical flow cannot be estimated everywhere, for example not in weakly-textured regions. For pixels in these regions, vertical and horizontal flows, i.e. u_y and u_x, do not exist. Thus we make an assumption that these pixels did not change position between frames $\{t\}$ and $\{t+1\}$, i.e. $u_y = 0$ and $u_x = 0$. We find the new label configuration by translating all labels according to the derived flow maps.

Suppose frame $\{t\}$ is segmented and S_t is its final label configuration (see Fig. 2(d)). The labels can be transferred from frame $\{t\}$ to frame $\{t+1\}$ according to

$$S_{t+1}(x_{t+1}, y_{t+1}) = S_t(x'_t, y'_t) \quad , \tag{11}$$

where

$$x'_t = x_{t+1} - u_x(x_t, y_t) \tag{12}$$

$$y't = y_{t+1} - u_y(x_t, y_t) \quad . \tag{13}$$

Labels which did not obtain an initialization via Eq. 11 are then given a randomly chosen label between 1 and q. Once frame $\{t+1\}$ is initialized (see Fig. 2(e)), a relaxation process is needed in order to fix erroneous bonds that can take place during the spin states transfer (see 2.2). We found that 20 additional Metropolis updating iterations are sufficient to obtain satisfactory segmentation results (see Fig. 2(f)).

2.5. Stereo segmentation using disparity-based label warping

Segmentation of every right frame is obtained in a similar way. Here, label initialization is obtained by translating the labels from the segmentation of the left image using the disparities from phase-based stereo. The stereo algorithm yields a sparse disparity map D providing information not for all pixels. For this reason, we make the assumption that pixels from frame $\{t_L\}$ having no correspondence in frame$\{t_R\}$ have zero displacement, i.e. $D(x^t) = 0$.

We suppose that the left frame $\{t_L\}$ is segmented and S^t_L is its final label configuration (see Fig. 3(d)). Labels are transferred from frame $\{t_L\}$ to frame $\{t_R\}$ according to

$$S^t_R(x^t_R, y^t_R) = S^t_L(x'^t_L, y'^t_L) \quad , \tag{14}$$

where

$$x'^t_L = x'^t_R - D(x^t_L, y^t_L) \tag{15}$$

$$y'^t_L = y'^t_R \quad . \tag{16}$$

Pixels that did not obtain a label initialization this way are given a randomly chosen label between 1 and q (see Fig. 3(e)). Once frame $\{t_R\}$ is initialized, again a relaxation process is needed in order to fix erroneous bonds (see 2.2). We found that about $50 - 100$ additional Metropolis updating iterations are sufficient to obtain satisfactory 3D segmentation results (see Fig. 3(f)).

2.6. Disparity from stereo-segment correspondences

In weakly-textured scenes, segment correspondences can be used to derive the disparity map of the scene in situations

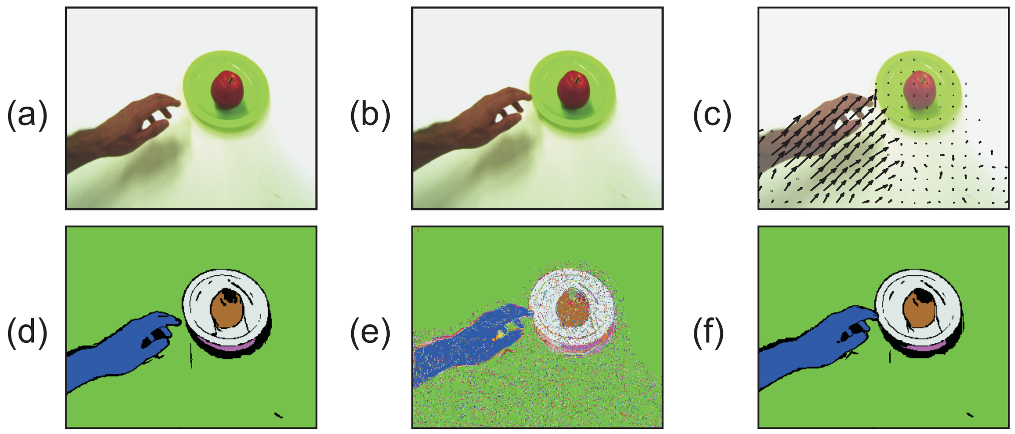


Figure 2. Segmentation of two adjacent frames in a sequence. (a) Original frame $\{t\}$. (b) Original frame $\{t+1\}$. (c) Estimated optical flow field from phase-based method (subsampled 16 times and scaled 5 times). (d) Extracted segments S_t for frame $\{t\}$. (e) Initialization of frame $\{t+1\}$ after the spin transfer. (f) Extracted segments S_{t+1} for frame $\{t+1\}$.

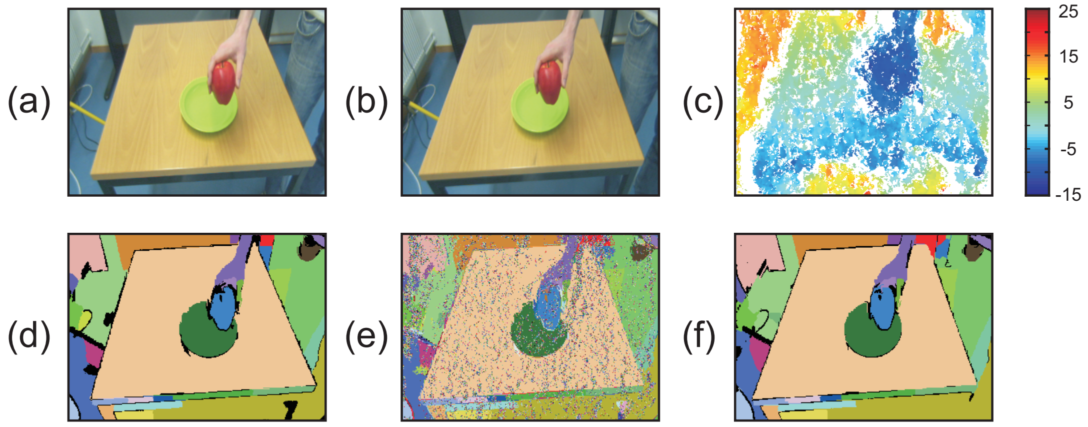


Figure 3. Segmentation of a stereo pair. (a) Original left frame $\{t_L\}$. (b) Original right frame $\{t_R\}$. (c) Estimated disparity map from phase-based stereo. (d) Extracted segments S_L^t for frame $\{t_L\}$. (e) Initialization of frame $\{t_R\}$ after the spin transfer. (f) Extracted segments S_R^t for frame $\{t_R\}$.

where texture-based methods are bound to fail [7]. Following the method proposed by [7], stereo segments are computed first using our framework, then, reliable disparity estimates from segment boundaries and weak inner-segment texture are extracted together with an occlusion map, which is derived from the approximate depth ordering of the stereo segments. Finally the information is fused and a segment-constrained interpolation algorithm is employed to obtain a dense disparity map (see Fig. 4(c) and Fig. 5(c)). The disparity map can then be used to establish 3D relations between segments in the graphical representation (see 2.7). We further defined a confidence value for each segment by summing the input confidence values (before interpolation - see [7]) for the respective segment and dividing it by the total size of the segment. Large segments for which little inner-segment disparities or edge disparities could be found will receive a lower confidence than smaller segments for which, e.g., the disparities are known along the whole segment boundary. Using this approach, we found that for the large white table of the sequences shown in Fig. 4(a) and Fig. 5(a), disparities could be interpolated only with very low confidence. This is because most of the boundaries of the table are cut off by the frame boundaries and because the table has no texture. As a consequence, stereo cannot provide sufficient depth information here.

2.7. 3D Semantic Scene Graphs

Once 3D segments are extracted, we represent the scene by undirected and unlabeled semantic graphs. The graph nodes are the segment labels and plotted at the center of each segment. The nodes are then connected by an edge if the segments are neighbors and their depth differences are less than a predefined threshold value (see Fig. 4(d) and Fig. 5(d)). We also define a 3D field of view boundary and ignore segments whose depth value does not exceed this boundary. With this method we create a focus of attention and ignore objects outside the 3D boundary.

In the temporal domain, 3D scene graphs represent spatial relations between nodes. Unless spatial relations change, the scene graphs remain topologically the same. The only changes in the graph structures are the node positions or the edge lengths depending on the object trajectory and speed. Consequently, any change in the spatial relation between nodes corresponds to a change in the main structure of the scene graphs. Therefore, those changes in the graphs can be employed to define action primitives. Considering this fact, we apply an exact graph-matching method in order to extract the main graphs by computing the eigenvalues and eigenvectors of the adjacency matrices of the 3D graphs [21]. A change in the eigenvalues or eigenvectors then corresponds to a structural change of the graph. As experimental data we use two real stereo-image sequences (see Section 3). The total frame number of the first image sequence "Making a sandwich" is 396, however, after extracting the main graphs, only 12 frames are left, each defining a single action primitive. Due to page limitations Fig. 4(d) shows only 7 of the main graphs. In the second image sequence we have interleaved chained actions, i.e., "Cutting a salami", "Making a sandwich", and "Putting on a plate", making a total of 2977 frames. In Fig. 5(d), 6 sample main graphs are shown, each representing an action primitive.

2.8. Experimental environment

As hardware platforms for our 3D segment tracking framework we use one NVIDIA card GeForce GTX 295 (with 896 MB device memory) consisting of two GPUs each of which has 30 multiprocessors and 240 processor cores in total and CPU 2.2GHz AMD Phenom Quad 9550 (using a single core) with 2 GB RAM.

We use the first GPU of the card to calculate optical flow and disparity. The second GPU is mainly used for frame-wise image segmentation. Using two GPUs allows us to run some parts of the framework physically in parallel and achieve better processing time as compared to having one GPU only.

3. Experimental Results

The developed framework is applied to two real stereo-image sequences: "Making a sandwich" (see Fig. 4(a)) and a sequence consisting of interleaved chained actions, i.e., "Cutting a salami", "Making a sandwich", and "Putting on a plate" (see Fig. 5(a)). In the first example (see Fig. 4(a)) two arms are appearing in the scene, putting the salami and cheese slices on a piece of bread, and then leaving the scene. The second sequence (see Fig. 5(a)) consists of two arms that are first taking a bread from a toaster, putting a piece of a cheese on it, and then cutting off a slice of salami with a knife. After putting the salami on top of the cheese, the sandwich is being placed on a plate and the arms are leaving the scene. Respective image segments of some sample frames from the left sequences are given in Fig. 4(b) and 5(b). Dense disparity maps obtained for extracted stereo segments are given in Fig. 4(c) and 5(c). The low-confidence-value area of the table segment is depicted with a black color in the dense disparity maps (see Section 2.6). Fig. 4(d) and 5(d) illustrate 3D semantic scene graphs of the selected frames. The graphs show that all relevant object parts in the scene can be represented by unique labels and tracked during the whole image sequence. Moreover, each graph defines a topological change in the scene. For instance, in frame number 112 of the first image sequence (see Fig. 4(d)) we observe that the arm represented by graph node number 113 has an edge only with the table (graph node number 3). In 2D, the arm would have an edge with the cheese since their respective segments are 2D neighbors, but this edge is ignored because the depth difference between the arm and the cheese is too large. In the next main graph (frame number 167) an edge connects the arm (node number 113) with the salami (node number 247) because the segments are touching in 3D.

For mono-image sequences with a frame size of 256×320 pixels a processing time of 23 Hz was achieved that demonstrates the applicability of the framework to mono-video processing tasks in real-time or close to real-time. However, for sequences containing weakly-textured regions more additional Metropolis updating iterations are needed in order to achieve a final stable configuration (see 2.4). For the stereo video segmentation with the same frame size the maximum processing time that can be achieved for the moment is 10 Hz. However this is not the case for all stereo-image sequences.

4. Discussion

In this work we presented a novel highly parallel framework for deriving a condensed 3D semantic representation of visual scenes based on a near real-time segment tracking procedure implemented in parallel on GPUs. We achieved (i) stereo image-sequence segmentation and segment track-

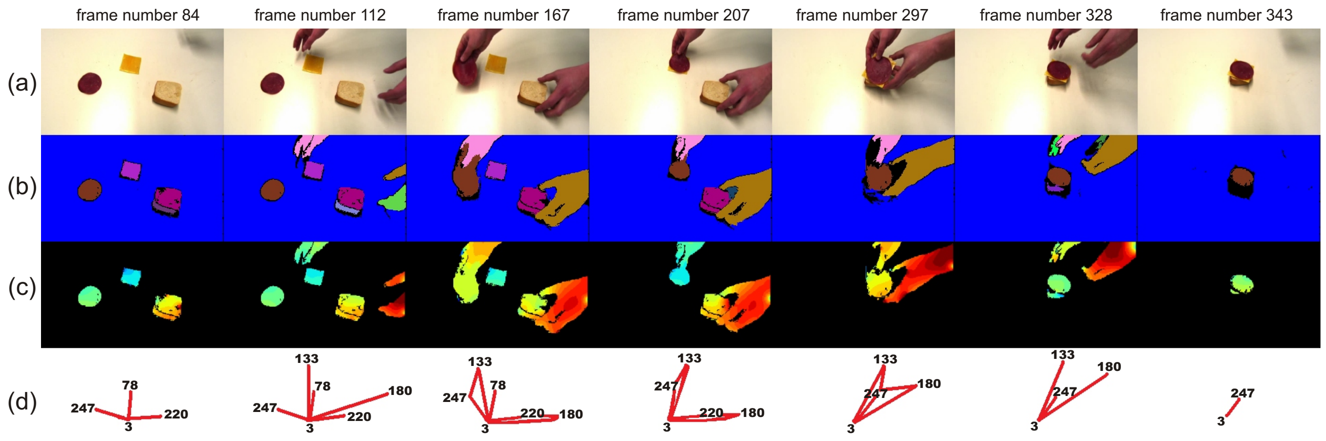


Figure 4. Results for the sample action "Making a sandwich". (a) Original frames from the left image sequence. (b) Extracted segments for frames of the left sequence. (c) The dense disparity maps obtained for extracted stereo segments. The disparity values are color-coded from blue (small) to red (large). Areas of low confidence are colored black, i.e., the uniform and untextured area of the table, for which only poor disparity results could be obtained. (d) Final 3D semantic scene graphs, representing action primitives.

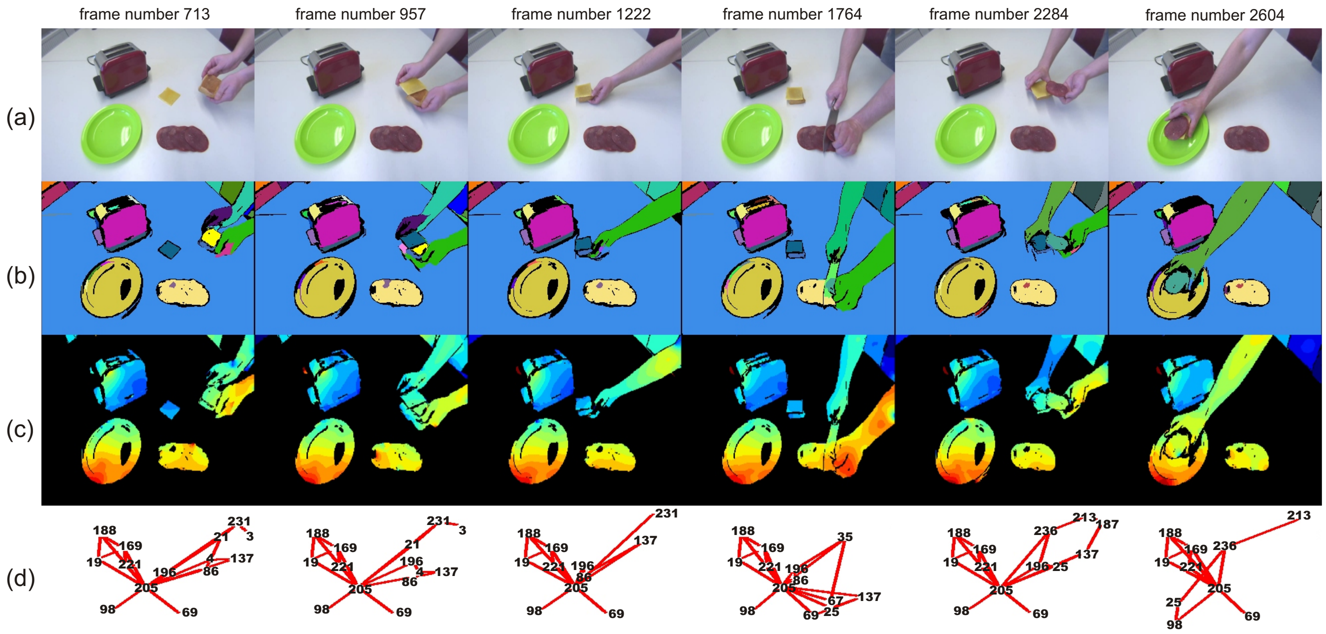


Figure 5. Results for the interleaved sample actions, i.e. "Cutting a salami", "Making a sandwich", and "Putting on a plate". (a) Original frames from the left image sequence. (b) Extracted segments for frames of the left sequence. (c) The dense disparity maps obtained for extracted stereo segments. The disparity values are color-coded from blue (small) to red (large). Areas of low confidence are colored black, i.e., the uniform and untextured area of the table, for which only poor disparity results could be obtained. (d) Final 3D semantic scene graphs, representing action primitives.

ing in near real time using a highly parallel architecture involving two GPUs, and, in a subsequent step, (ii) to build 3D semantic graphs using a recent stereo method for defining the 3D segment relations and exact graph matching for the extraction main graphs (action primitives). Algorithmic parts for (ii) are implemented on CPU and are not yet running in real-time, but potential parallel solutions are currently investigated. The main contribution of this work is the provision of a novel framework for representing image sequences in an efficient way, which may serve as vision-front end for cognitive robots in the future. Most importantly, the segment tracking is entirely data driven (model-free) and thus does not require prior data models to be provided, making the method more robust and more widely applicable.

To our knowledge, the quite extensive framework presented in this paper is the first of its kind. Although video segmentation has been proposed before, the methods used therein are often model based, or not running real time [5, 20, 23, 8, 16, 28, 26, 14, 6]. Furthermore, our segment tracking procedure is developed for a specific goal, i.e. the construction of 3D semantic graphs, which has not been attempted in this way before.

The proposed framework has been applied to several real stereo-image sequences. Obtained results demonstrate stability of the segment tracking procedure both for mono-image sequences and for stereo-image sequences. For the segmentation of mono-image sequences with a frame size of 256×320 pixels we obtained processing time sufficient for real-time or close to real-time applications. Stereo-image sequence segmentation is not real time yet, but the developed framework is sufficiently fast for being applicable to robot-real-time 3D applications.

The algorithm has some weak points. At the moment, relaxation times increase if two previously disjoint regions ought to be joined. The processing time of stereo-video segmentation also depends on texture information and the size of occluded areas. For sequences with weakly textured areas and relatively large occlusions a longer relaxation process is required to label those areas. Currently, we are developing an improved parallel Metropolis procedure to cope with these challenges.

Furthermore, problems arise for scenarios with relatively high complexity. By high complexity we mean actions that entail splitting, e.g., cutting actions. In this case some parts become new independent segments, so new labels have to be assigned to them. Clearly, these problems cannot be resolved on the pixel level. A high level procedure is needed for the detection of object splittings. We are currently investigating this problem in more detail.

In the future, we aim to apply the developed framework to tasks in cognitive robotics. Since each main graph represents an action primitive and also the respective object relations, we are planning to use this information to recognize and classify the actions and categorize objects based on their role during the action.

References

[1] Authors. Real-time image segmentation on a GPU. In *Facing the Multicore-Challenge, Submitted*. 4

[2] G. T. Barkema and T. MacFarland. Parallel simulation of the Ising model. *Physical Review E*, 50(2):1623–1628, 1994. 4

[3] M. Blatt, S. Wiseman, and E. Domany. Superparamagnetic clustering of data. *Physical Review Letters*, 76(18):3251–3254, 1996. 3

[4] P. Carnevali, L. Coletti, and S. Patarnello. Image processing by simulated annealing. *IBM Journal of Research and Development*, 29(6):569–579, 1985. 3

[5] J. G. Choi, S. W. Lee, and S. D. Kim. Spatio-temporal video segmentation using a joint similarity measure. *IEEE Trans. Circuits Syst. Video Technol.*, 7:279–286, 1997. 2, 8

[6] B. Dellen, E. E. Aksoy, and F. Wörgötter. Segment tracking via a spatiotemporal linking process including feedback stabilization in an n-d lattice model. *Sensors*, 9(11):9355–9379, 2009. 2, 8

[7] B. Dellen and F. Wörgötter. Disparity from stereo-segment silhouettes of weakly-textured images. In *British Machine Vision Conference*, London, UK, 2009. 2, 5

[8] Y. Deng and B. S. Manjunath. Unsupervised segmentation of color-texture regions in images and video. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23:800–810, 2001. 2, 8

[9] C. Eckes and J. C. Vorbrüggen. Combining data-driven and model-based cues for segmentation of video sequences. *World Congress on Neural Networks, San Diego*, 1996. 3

[10] D. Geman and S. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(6):721–741, 1984. 3

[11] D. Geman, S. Geman, C. Graffigne, and P. Dong. Boundary detection by constrained optimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(7):609–628, 1990. 3

[12] 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. 1

[13] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. of Chem. Phys.*, 21(11):1087–1091, 1953. 3

[14] V. Mezaris, I. Kompatsiaris, and M. G. Strintzis. Video object segmentation using bayes-based temporal tracking and trajectory-based region merging. *IEEE Trans. Circuits Syst. Video Technol.*, 14(6):782–795, 2004. 2, 8

[15] R. Opara and F. Wörgötter. A fast and robust cluster update algorithm for image segmentation in split-lattice models without annealing - visual latencies revisited. *Neural Computation*, 10(6):1547–1566, 1998. 3

[16] L. Patras, E. A. Hendriks, and R. L. Lagendijk. Video segmentation by map labeling of watershed segments. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23:326–332, 2001. 2, 8

[17] K. Pauwels and M. Van Hulle. Realtime phase-based optical flow on the gpu. In *IEEE Conference on Computer Vision and Pattern Recognition, Workshop on Computer Vision on the GPU*, Anchorage, Alaska, 2008. 2, 4

[18] R. B. Potts. Some generalized order-disorder transformations. *Proc. Cambridge Philos. Soc.*, 48:106–109, 1952. 3

[19] S. Sabatini, G. Gastaldi, F. Solari, J. Diaz, E. Ros, K. Pauwels, M. Van Hulle, N. Pugeault, and N. Krüger. Compact and accurate early vision processing in the harmonic

space. In *International Conference on Computer Vision Theory and Applications*, pages 213–220, Barcelona, 2007. 2, 4

[20] P. Salembier and F. Marques. Region-based representations of image and video: Segmentation tools for multimedia services. *IEEE Trans. Circuits Syst. Video Technol.*, 9:1147–1169, 1999. 2, 8

[21] M. F. Sumsi. *Theory and Algorithms on the Median Graph. Application to Graph-based Classification and Clustering*. PhD thesis, Universitat Autònoma de Barcelona, 2008. 6

[22] R. Swendsen and S. Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 76(18):86–88, 1987. 3

[23] E. Tuncel and L. Onural. Utilization of the recursive shortest spanning tree algorithm for video-object segmentation by 2-d affine motion modeling. *IEEE Trans. Circuits Syst. Video Technol.*, 10:776–781, 2000. 2, 8

[24] C. von Ferber and F. Wörgötter. Cluster update algorithm and recognition. *Physical Review E*, 62(2):1461–1464, 2000. 3

[25] J. C. Vorbrüggen. Zwei modelle zur datengetriebenen segmentierung visueller daten. *Frankfurt am Main, Harri Deutsch, Thun*, 1976. 3

[26] D. Wang. Unsupervised video segmentation based on waterheds and temporal tracking. *IEEE Trans. Circuits Syst. Video Technol.*, 8:539–546, 1998. 2, 8

[27] U. Wolff. Collective Monte Carlo updating for spin systems. *Physical Review Letters*, 62(4):361–364, 1989. 3

[28] Y. Yokoyama, Y. Miyamoto, and M. Ohta. Very low bit rate video coding using arbitrarily shaped region-based motion compensation. *IEEE Trans. Circuits Syst. Video Technol.*, 5:500–507, 1995. 2, 8

Categorizing Object-Action Relations from Semantic Scene Graphs

Eren Erdal Aksoy, Alexey Abramov, Florentin Wörgötter, and Babette Dellen

Abstract— **In this work we introduce a novel approach for detecting spatiotemporal object-action relations, leading to both, action recognition and object categorization. Semantic scene graphs are extracted from image sequences and used to find the characteristic main graphs of the action sequence via an exact graph-matching technique, thus providing an event table of the action scene, which allows extracting object-action relations. The method is applied to several artificial and real action scenes containing limited context. The central novelty of this approach is that it is model free and needs *a priori* representation neither for objects nor actions. Essentially actions are recognized without requiring prior object knowledge and objects are categorized solely based on their exhibited role within an action sequence. Thus, this approach is grounded in the affordance principle, which has recently attracted much attention in robotics and provides a way forward for trial and error learning of object-action relations through repeated experimentation. It may therefore be useful for recognition and categorization tasks for example in imitation learning in developmental and cognitive robotics.**

I. INTRODUCTION

One central goal for humanoid robotics is to imitate, understand, and learn from human behavior. Part of this problem is to relate a manipulation to its manipulated object. The difficulty lies here in the fact that individual manipulations even when "doing the same thing" can take vastly different forms just due to changes in posture, action sequence, and/or differences in the general (visual) context surrounding the core manipulation. Nonetheless humans have no problem in classifying manipulation types, such as "moving an object", "closing a book" or "making a sandwich", and to link objects with actions. The goal of this paper is to devise a method which can, at least to some degree, do the same and thereby classify manipulation types.

Recently, these questions have been approached in an abstract way by the concept of object-action complexes (OACs) [1], [2], claiming that objects and actions are inseparably intertwined. This is linked to the way humans perceive the world by relating objects with actions. The OAC concept proposes such a human-like description by which an object is identified considering both its (visual) properties *and* the actions that have been performed with it. The OAC concept attaches the performed actions to the objects as attributes. This approach is therefore related to the affordance principle [3], which especially in the recent years has had increasing influence in robotics [4]. Take, for example, a cup, which is an entity for filling and drinking. However, not only this single specific cup but any other cylindrical, hollow object could be used for the same actions. Thus, objects, which are supporting common actions, can be considered similar. Filling creates the object-type "container"! Consider now an inverted cup, which cannot be filled. Now the former container has become a pedestal on which you could put something. While physically the same thing, a "pedestal" is a different object type altogether. In cognitive vision, many new approaches for object recognition from 3D models have been introduced [5], [6], [7]. However, these model-based approaches cannot identify object-action relations. In this work, we introduce a novel approach for detecting spatiotemporal object-action relations using semantic scene graphs, leading to both action recognition and object categorization. Using this method, objects are connected to recognized actions considering their roles within a scenario.

The approach relies on a front-end algorithm which allows for the continuous tracking of scene segments using superparamagnetic clustering, which proven convergence properties [8], [9], [10]. The presented core algorithm, used for recognition and classification, then relies on the sequence of neighborhood relations between those segments, which for a given action will always be "essentially" the same. Hence different from feature based (or model based) approaches our system operates on object-part relations without presupposing assumptions about the structure of object and action. Thus, it is model free. This leads to a high degree of invariance against position, orientation, etc. but we need to make sure that segment tracking is stable, which is currently achieved by several means described elsewhere [11]. Furthermore, at this stage we show examples from a 2D (projected) domain. True 3D tracking is currently being implemented for difficult action sequences like "making a complete breakfast".

Therefore, we would like to emphasize that the core contribution of this work is the novel categorization method. The computer vision front end is a required prerequisite, but other tracking methods could be used here as well and improvements are possible.

The structure of the paper is as follows. In Section II we discuss related works. In Section III, we introduce the action classification and the object categorization algorithm. In Section IV experimental results with real images are presented. Finally, in Section V, the results are discussed and directions for future research are given.

The work has received support from the BMBF funded BCCN Göttingen and the EU Project PACO-PLUS under Contract No. 027657.

Eren Erdal Aksoy, Alexey Abramov, and Florentin Wörgötter are with Bernstein Center for Computational Neuroscience, University of Göttingen, Friedrich-Hund Platz 1, 37077 Göttingen, Germany eaksoye,abramov,worgott@bccn-goettingen.de

B. Dellen is with Bernstein Center for Computational Neuroscience, Max-Planck Institute for Dynamics and Self-Organization, Bunsenstr. 10, 37073 Göttingen, Germany bkdellen@bccn-goettingen.de

II. RELATED WORK

Action recognition and object categorization have received increasing interest in the Artificial Intelligence (AI) and cognitive-vision community during the last decade. The problem of action recognition has been addressed in previous works, but only rarely in conjunction with object categorization. Modayil *et al.* (2008) presented a framework focusing on the recognition of activities in daily living [12]. In order to detect the activities, the test subject (e.g. human) was equipped with Radio Frequency Identification (RFID) reader and tags. The types of actions and used objects were recorded by the RFID reader to learn a model that recognizes the activities performed during observations, and an Interleaved Hidden Markov Model (HMM) was used to increase the accuracy of the learned model. Similar to this study, Liao *et al.* (2005) provided an approach to perform location-based activity recognition by using Relational Markov Networks [13]. This work also covered high-level activities, e.g. working, shopping, dining out, during long periods of time. The system used data from a wearable GPS location sensor and considered time, place of action, and sequence of action, which were extracted from the GPS sensor. Although those kinds of sensor-based multitasking-activity-recognition approaches provide promising results, they do not cover object-categorization issues and have handicaps like limited coverage area. Hongeng (2004) introduced a Markov network to encode the entire event space for scenes with limited context but without considering object classification [14]. Sridhar *et al.* (2008) showed that objects can also be categorized by considering their common roles in actions, resulting however in large and complex activity graphs, which have to be analyzed separately [15]. Our framework provides a novel approach that represents scenes by semantic graphs which hold spatiotemporal object-action relations. By analyzing semantic scene graphs, we not only recognize actions but also categorize objects based on their action roles.

III. METHODS

A. Overview of the Algorithm

In the current study, we analyze movies of scenes containing limited context. Fig. 1 shows the block diagram of the algorithm. As a first processing step, image segments are extracted and tracked throughout the image sequence, allowing the assignment of temporally-stable labels to the respective image parts [9], [11]. The scene is then described by semantic graphs, in which the nodes and edges represent segments and their neighborhood relations, respectively. For segmentation and graph examples of real images, see Fig. 2. Graphs can change by continuous distortions (lengthening or shortening of edges) or, more importantly, by discontinuous changes (nodes or edges can appear or disappear). Such a discontinuous change represents a natural breaking point: All graphs before are topologically identical and so are those after the breaking point. Hence, we can apply an exact graph-matching method after a breaking point and extract the next following topological main graph. The sequence of these main graphs thus represents all structural changes in the scene. The temporal order by which those main graphs follow each other defines an "event table". An event signifies that something has happened in the scene which caused a true topological change in the graph. This method allows classifying object-action relations by calculating the similarity between event tables from different scenes. Furthermore, nodes playing the same role in an classified action sequence can be identified and then be used to categorize objects by returning to the signal level via image segments.

B. Segmentation and Tracking

We use an image-segmentation method in which segments are obtained trough a 3D linking process [9], [11], [10]. First, a spin variable σ_i is assigned to each pixel i of the stereo image. To incorporate constraints in form of local correspondence information, we distinguish between neighbors within a single frame (2D bonds) and neighbors across frames (3D bonds). We create a 2D bond $(i,k)_{2D}$ between two pixels within the same frame with coordinates (x_i, y_i, z_i) and (x_k, y_k, z_k) if $|(x_i - x_k)| \leq 1$, $|(y_i - y_k)| \leq 1$, and $z_i = z_k$. Across frames, we create a 3D bond $(i,j)_{3D}$ between two spins i and j if $|(x_i + d^x_{ij} - x_j)| \leq 0.5$, $|(y_i + d^y_{ij} - y_j)| \leq 0.5$, $z_i \neq z_j$, and $a_{ij} = 1$. The values d^x_{ij} and d^y_{ij} are the shifts of the pixels between frames z_i and z_j along the axis x and axis y, obtained from an initial optic flow map. The parameters a_{ij} are the respective amplitudes (or confidences). However, since the images in the examples given in this paper are changing only little from frame to frame, we will assume that the flow is zero everywhere. Hence the values d^x_{ij} and d^y_{ij} are zero, and $a_{ij} = 1$ everywhere.

The spin model is now implemented such that neighboring spins with similar color have the tendency to align. We use a q-state Potts model [16] with the Hamiltonian

$$H = -\sum_{\langle ik\rangle_{2D}} J_{ik}\delta_{\sigma_i,\sigma_k} - \sum_{\langle ij\rangle_{3D}} J_{ij}\delta_{\sigma_i,\sigma_j} \quad , \tag{1}$$

with $J_{ij} = 1 - \triangle/\bar{\triangle}$ and $\triangle_{ij} = |g_i - g_j|$, where g_i and g_j are the gray (color) values of the pixels i and j, respectively. The mean distance $\bar{\triangle}$ is obtained by averaging over all bonds.

Here, $\langle ik\rangle_{2D}$ and $\langle ij\rangle_{3D}$ denote that i,k and i,j are connected by bonds $(i,k)_{2D}$ and $(i,j)_{3D}$, respectively. The Kronecker δ function is defined as $\delta_{a,b} = 1$ if $a = b$ and zero otherwise. The segmentation problem is then solved by finding clusters of correlated spins in the low temperature equilibrium states of the Hamiltonian H. The total number M of segments is then determined by counting the computed segments. It is usually different from the total number q of spin states, which is a parameter of the algorithm (here $q = 10$).

We solve this task by implementing a clustering algorithm. In a first step, "satisfied" bonds, i.e. bonds connecting spins of identical spins $\sigma_i = \sigma_j$, are identified. Then, in a second step, the satisfied bonds are "frozen" with a some probability P_{ij}. Pixels connected by frozen bonds define a cluster, which are updated by assigning to all spins inside the same clusters

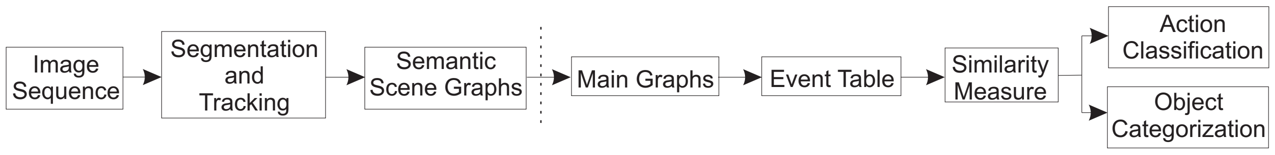


Fig. 1. Block diagram of the algorithm.

the same new value [17]. In the method of superparamagnetic clustering proposed by [18] this is done independently for each cluster. In this paper, we will employ the method of energy-based cluster updating (ECU), where new values are assigned in consideration of the energy gain calculated for a neighborhood of the regarded cluster [8], [19]. The algorithm is controlled by a single "temperature" parameter, and has been shown to deliver robust results over a large temperature range. After a 100 iterations, clusters are used to define segments.

In this paper, we segment always two consecutive frames of the image sequence at the same time, i.e. frame i and $i+1$, then, we segment the next pair, i.e. $i+1$ and $i+2$, where the last image of the first pair is identical with the first image of the second pair. Then, consecutive pairs are connected by identifying the identical segments in the overlapping images. This strategy allows handling long motion image sequences [11].

C. Semantic Scene Graphs

Once the image sequence has been segmented and segments have been tracked, we represent the scene by undirected and unweighted labeled graphs. The graph nodes are the segment labels and plotted at the center of each segment. The nodes are then connected by an edge if segments touch each other.

Fig. 2 shows original frames with respective segments and semantic scene graphs from three different real action types: *Moving Object*, *Opening Book*, and *Making Sandwich*. In the *Moving Object* action a hand is putting an orange on a plate while moving the plate together with the orange (see Fig. 2(a-c)). The *Opening Book* action represents a scenario in which a hand is opening a book (see Fig. 2(d-f)). In the *Making Sandwich* action two hands are putting pieces of bread, salami, and cheese on top of each other (see Fig. 2(g-i)).

D. Main Graphs and Event Tables

In the following we will first use simpler scenes to describe the remaining parts of the algorithm (to the right of the dashed line in Fig. 1). Fig. 3(a-b) depicts original frames with respective segments of an artificial *Moving Object* action (sample action 1) in which a black round object is moving from a yellow vessel into a red vessel.

In the temporal domain, scene graphs represent spatial relations between nodes. Unless spatial relations change, the scene graphs remain topologically the same. The only changes in the graph structures are the node positions or the

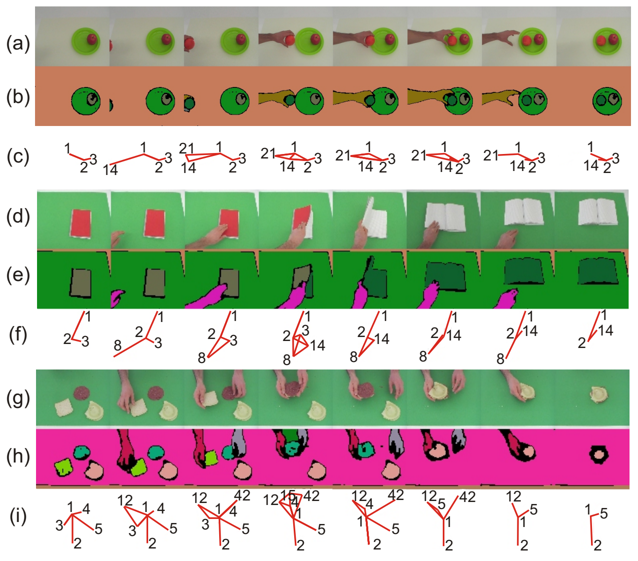
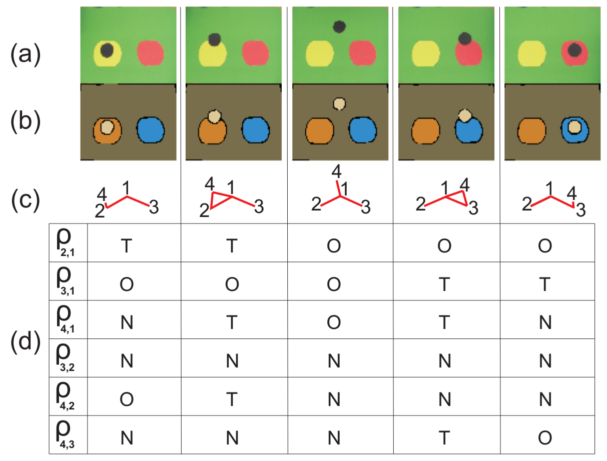


Fig. 2. Three different real action types. (a) Original images from the *Moving Object* action. (b) Respective image segments. (c) Semantic scene graphs. (d) Original images from the *Opening Book* action. (e) Respective image segments. (f) Semantic scene graphs. (g) Original images from the *Making Sandwich* action. (h) Respective image segments. (i) Semantic scene graphs.

edge lengths depending on the object trajectory and speed. Consequently, any change in the spatial relation between nodes corresponds to a change in the main structure of the scene graphs. Therefore, those changes in the graphs can be employed to define action primitives. Considering this fact, we apply an exact graph-matching method in order to extract the main graphs by computing the eigenvalues and eigenvectors of the adjacency matrices of the graphs [20]. A change in the eigenvalues or eigenvectors then corresponds to a structural change of the graph. The whole image sequence of the sample *Moving Object* action has 92 frames, however, after extracting the main graphs, only 5 frames are left, each defining a single action primitive (see Fig. 3(c)).

Following the extraction of the main graphs, we analyze the spatial relations between each pair of nodes in the main graphs. We denote the spatial relations by $\rho_{i,j}$ in which i and j are the nodes of interest. Note that the spatial relations are symmetric, i.e. $\rho_{i,j} = \rho_{j,i}$.

Possible spatial relations of each node pair are *absence* (A), *no connection* (N), *overlapping* (O), and *touching* (T). We define those relations by calculating the number of edges of both currently considered nodes i and j in each main graph. As an example, all possible spatial relations between



$\rho_{2,1}$	T	T	O	O	O
$\rho_{3,1}$	O	O	O	T	T
$\rho_{4,1}$	N	T	O	T	N
$\rho_{3,2}$	N	N	N	N	N
$\rho_{4,2}$	O	T	N	N	N
$\rho_{4,3}$	N	N	N	T	O

Fig. 3. Simple example of the *Moving Object* action (sample action 1). (a) Original images. (b) Respective image segments. (c) Semantic scene graphs. (d) Event table.

the black object and yellow vessel are illustrated in Fig. 4. Since those objects are represented by graph nodes 4 and 2, we write the relation as $\rho_{4,2}$. The relation *absence* means that one of the considered nodes is not observed in the scene, i.e. the black object node 4 does not exist in the graph (see Fig. 4(a)). In the case of *no connection*, the considered nodes have no edge between them (see Fig. 4(b)). In the *overlapping* relation one of the considered nodes is completely surrounded by the other node. Therefore, the surrounded node has only one edge (see Fig. 4(c)). The *touching* relation represents the situation in which segments touch each other and both considered nodes have more than one edge (see Fig. 4(d)). More complex spatial relations between nodes are currently not considered but could be included in the future.

The total number of spatial relations is defined as

$$\rho_{\text{total}} = \sum_{i=1}^{n-1}(n-i) \quad , \tag{2}$$

where n is the total number of objects. For the sample *Moving Object* action mentioned above we have $n = 4$ (yellow and red vessels, a black moving object, and a green background) and therefore $\rho_{\text{total}} = 6$. Those relations are $\rho_{2,1}$, $\rho_{3,1}$, $\rho_{4,1}$, $\rho_{3,2}$, $\rho_{4,2}$, and $\rho_{4,3}$.

All existing spatial node relations in the main graphs are saved in the form of a table where the rows represent spatial relations between each pair of nodes. Since any change in the spatial relations represents an event that defines an action, we refer to this table as an *event table* (ξ). Fig. 3(d) shows the *event table* of the action above. However, the fourth row of the *event table* does not hold any change in the sense of a changing spatial relation since the yellow and red vessels never move. For this reason, we ignore the fourth row. For the sake of simplicity, we substitute numbers -1, 0, 1, and 2 for possible spatial relations A, N, O, and T. The final *event table* of sample action 1 is given in Table 1.

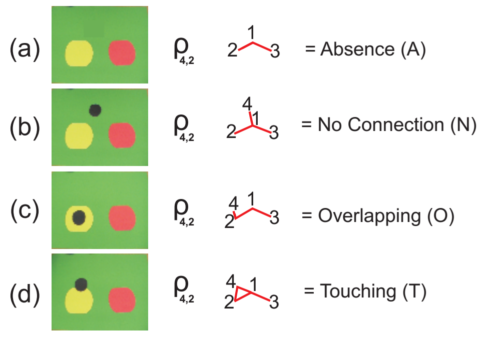


Fig. 4. Possible spatial relations between black object and yellow vessel which are represented by graph nodes 4 and 2, respectively. (a) The relation *absence*. (b) The relation *no connection*. (c) The *overlapping* relation. (d) The *touching* relation.

E. Similarity Measure

So far we showed how to represent a long image sequence by an event table the dimensions of which are related to the spatial node relations in the main graphs. Next we will discuss how to calculate the similarity of two actions. To this end we created one more sample for the *Moving Object* action. Fig. 5 depicts the main graphs of sample action 2 in which a red rectangular object is moving from a blue vessel into a yellow vessel following a different trajectory with different speed as compared to the first sample. Moreover, the scene contains two more objects which are either stationary (red round object) or moving randomly (black round object). Following the same procedure, the *event table* for the second sample is calculated and given in Table 2. Note that even though the second sample contains more objects, the dimension of the event tables is accidentally the same. This makes explanations simpler, but, as we will see later,

$\rho_{2,1}$	2	2	1	1	1
$\rho_{3,1}$	1	1	1	2	2
$\rho_{4,1}$	0	2	1	2	0
$\rho_{4,2}$	1	2	0	0	0
$\rho_{4,3}$	0	0	0	2	1

TABLE I
EVENT TABLE (ξ_1) OF THE FIRST SAMPLE ACTION. SPATIAL RELATIONS BETWEEN THE NODES OF SAMPLE ACTION 1.

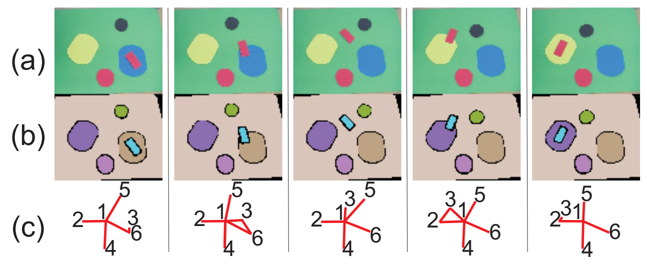


Fig. 5. Different version of the simple *Moving Object* action (sample action 2). (a) Original images. (b) Respective image segments. (c) Semantic scene graphs.

$\rho_{2,1}$	1	1	1	2	2
$\rho_{3,1}$	0	2	1	2	0
$\rho_{6,1}$	2	2	1	1	1
$\rho_{3,2}$	0	0	0	2	1
$\rho_{3,6}$	1	2	0	0	0

TABLE II
EVENT TABLE (ξ_2) OF THE SECOND SAMPLE ACTION. SPATIAL RELATIONS BETWEEN THE NODES OF SAMPLE ACTION 2.

the dimensions of the event tables are not important and can even be different between two cases.

Similarity measurement of actions is based on the comparison of the event tables. Basically, each row of the first *event table* (ξ_1) is compared with each row of the second *event table* (ξ_2) in order to find the highest similarity. (For event tables with different dimensions, sub-matrices need to be used.) Considering this simple rule we start determining the similarity with the first rows of ξ_1 and ξ_2, giving [2 2 1 1 1] and [1 1 1 2 2], respectively. Those lines are written one below the other. Next, the amount of equal digits (equal relations!) are counted and divided by total number of digits. Since only one digit (third digit) out of five digits is the same, the similarity of those two rows is 20%. Once all rows have been compared with each other, the determined similarity values are saved in the form of a table where rows and columns give the similarity relations between ξ_1 and ξ_2. The resulting table is called *similarity table* (ζ) and shown in Table 3.

The final similarity measure is determined by calculating the arithmetic mean value of the highest values in each row of ζ. Consequently, our two sample actions have 100% similarity.

In case of having event tables with different dimensions, we apply a window-based search algorithm to the bigger table in order to find out a region that has the highest similarity with the smaller table. In this case, the number of total search is defined as

$$s_{\text{total}} = (|r_1 - r_2| + 1)(|c_1 - c_2| + 1) \quad , \tag{3}$$

where r_1, r_2, c_1, and c_2 are the row and column numbers of the first and second event tables. The final similarity measurement is the highest similarity observed during this total search. If the dimensions are inconsistent in size to decide which one is smaller (such as $r_1 < r_2$ and $c_1 > c_2$ or $r_1 > r_2$ and $c_1 < c_2$), the event table with less columns is extended by adding the last column until it has the same number of columns as the bigger table. This sort of operation does not affect the action content since we do not change spatial node relations in the temporal domain.

ξ_1 \ ξ_2	$\rho_{2,1}$	$\rho_{3,1}$	$\rho_{6,1}$	$\rho_{3,2}$	$\rho_{3,6}$
$\rho_{2,1}$	20%	40%	**100%**	20%	20%
$\rho_{3,1}$	**100%**	40%	20%	20%	0%
$\rho_{4,1}$	40%	**100%**	40%	40%	40%
$\rho_{4,2}$	20%	40%	20%	20%	**100%**
$\rho_{4,3}$	20%	40%	20%	**100%**	20%

TABLE III
SIMILARITY TABLE (ζ). SIMILARITY VALUES BETWEEN ξ_1 AND ξ_2.

As a result we can now measure how similar the two actions are and we find 100%. Thus, these actions are of the same type ("type-similar").

F. Object Categorization

The *similarity table* also implicitly encodes the similarity of the nodes between the two different examples. Intriguingly, this can be used to extract *nodes with the same action roles* in type-similar actions. For this we first list all relations ρ of both actions with highest individual similarity. For instance, the relation between nodes 2 and 1 ($\rho_{2,1}$) in the first row has a 100% similarity with the relation between nodes 6 and 1 ($\rho_{6,1}$) in the third column. Doing this for all relations, we find the following maximal similarities in ζ:

$$\begin{array}{ccc} \rho_{2,1} & \Leftarrow 100\% \Rightarrow & \rho_{6,1} \\ \rho_{3,1} & \Leftarrow 100\% \Rightarrow & \rho_{2,1} \\ \rho_{4,1} & \Leftarrow 100\% \Rightarrow & \rho_{3,1} \\ \rho_{4,2} & \Leftarrow 100\% \Rightarrow & \rho_{3,6} \\ \rho_{4,3} & \Leftarrow 100\% \Rightarrow & \rho_{3,2} \end{array} \quad .$$

Those similarity values represent the correspondences between manipulated nodes in ξ_1 and ξ_2. In order to determine these correspondences, we analyze which node number in ξ_1 is repeating in conjunction with which node number in ξ_2. We start with node number 1 in ξ_1, and obtain

$$\left.\begin{array}{ccc} \rho_{2,1} & \Leftarrow 100\% \Rightarrow & \rho_{6,1} \\ \rho_{3,1} & \Leftarrow 100\% \Rightarrow & \rho_{2,1} \\ \rho_{4,1} & \Leftarrow 100\% \Rightarrow & \rho_{3,1} \end{array}\right. \Rightarrow \quad 1 \approx 1 \qquad .$$

While 1 is repeating three times in ξ_1, the same node number 1 in ξ_2 is also repeating three times. However, node numbers 2, 3, and 6 in ξ_2 occur only once. Therefore, we conclude that graph nodes 1 in both ξ_1 and ξ_2 had the same roles. In fact, both graph nodes represent the green background which plays same role in both actions.

We continue the spatial node relation analysis with node number 2 in ξ_1, and obtain

$$\left.\begin{array}{ccc} \rho_{2,1} & \Leftarrow 100\% \Rightarrow & \rho_{6,1} \\ \rho_{4,2} & \Leftarrow 100\% \Rightarrow & \rho_{3,6} \end{array}\right. \Rightarrow \quad 2 \approx 6 \qquad .$$

Node number 2 in ξ_1 is repeating twice with node number 6 in ξ_2. Those graph nodes represent the yellow and blue vessels within which the moving objects are initially located and from which they then move away.

For the case of node number 3 in ξ_1 we obtain

$$\left.\begin{array}{ccc} \rho_{3,1} & \Leftarrow 100\% \Rightarrow & \rho_{2,1} \\ \rho_{4,3} & \Leftarrow 100\% \Rightarrow & \rho_{3,2} \end{array}\right. \Rightarrow \quad 3 \approx 2 \qquad .$$

Node number 3 in ξ_1 corresponds to node number 2 in ξ_2 because both of them are repeating twice. Those graph nodes define the destination vessels for the moving objects.

The last node number 4 in ξ_1 is obtained as

$$\begin{array}{lcl} \rho_{4,1} & \Leftarrow 100\% \Rightarrow & \rho_{3,1} \\ \rho_{4,2} & \Leftarrow 100\% \Rightarrow & \rho_{3,6} \\ \rho_{4,3} & \Leftarrow 100\% \Rightarrow & \rho_{3,2} \end{array} \Rightarrow \quad 4 \approx 3 \quad .$$

As node number 4 in ξ_1, node number 3 in ξ_2 is also repeating three times. In fact, both graph nodes represent the moving objects which are the round black object in ξ_1 and the rectangular red object in ξ_2.

In the case of having a *similarity table* which has the same highest value more than once in a column, e.g. having two times 100% similarity values in the same column, the object categorization section leads to ambiguous results, i.e. one object corresponds to two different objects. Since this sort of correspondence is not allowed in the framework, the final similarity value is calculated again by taking the second highest values into account. This way we can get rid of any kind of mismatching in the object categorization process.

IV. Results with Real Images

We applied our framework to three different real action types: *Moving Object*, *Opening Book*, and *Making Sandwich* (see Fig. 2). For each of these actions, we recorded four movies with different trajectories, speeds, hand positions, and object shapes. All those twelve movies were recorded by a stable camera that was focused on the hands and the manipulated objects.

In Fig. 6, some sample frames for the other versions of all three action types are shown. In this version of the *Moving Object* action a hand is appearing in the scene, taking an apple from a plate, and leaving the scene (see Fig. 6(a)). The other version of the *Opening Book* action type represents a scenario in which a hand is closing a book (see Fig. 6(b)). In another version of the *Making Sandwich* action two hands are putting pieces of bread and cheese on top of each other in different order (see Fig. 6(c)).

Sample images for each action type are shown in Fig. 7(a-c) to give an impression of the level of complexity, i.e. amount of texture, reflections, and shadows, of the images used here.

Event tables of each real test data are compared with each other. The resulting similarity values are given in Fig. 8. Each test data has at least 69% similarity with the other versions of its type-similar action (see close to diagonal). In general the similarity between type-similar actions is for all

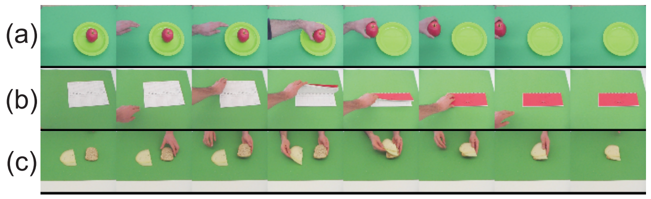


Fig. 6. Different versions of the real action types. (a) *Moving Object*. (b) *Opening Book*. (c) *Making Sandwich*.

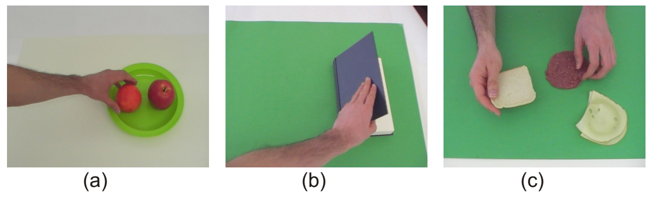
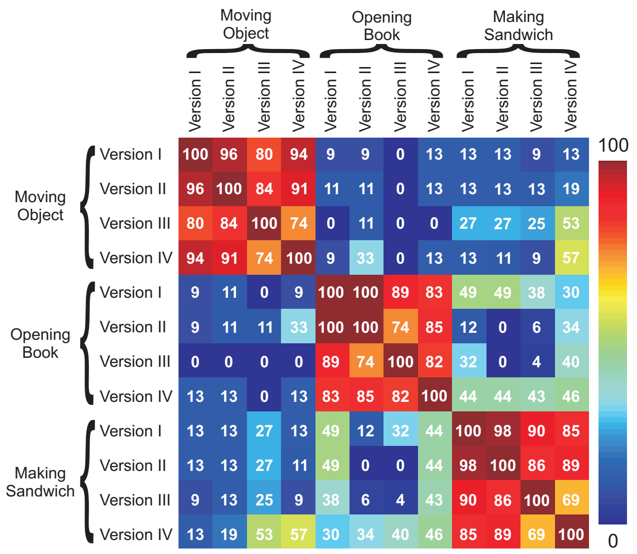


Fig. 7. Sample images taken from each real action type. (a) *Moving Object*. (b) *Opening Book*. (c) *Making Sandwich*.



		Moving Object				Opening Book				Making Sandwich			
		Version I	Version II	Version III	Version IV	Version I	Version II	Version III	Version IV	Version I	Version II	Version III	Version IV
Moving Object	Version I	100	96	80	94	9	9	0	13	13	13	9	13
	Version II	96	100	84	91	11	11	0	13	13	13	13	19
	Version III	80	84	100	74	0	11	0	0	27	27	25	53
	Version IV	94	91	74	100	9	33	0	13	13	11	9	57
Opening Book	Version I	9	11	0	9	100	100	89	83	49	49	38	30
	Version II	9	11	11	33	100	100	74	85	12	0	6	34
	Version III	0	0	0	0	89	74	100	82	32	0	4	40
	Version IV	13	13	0	13	83	85	82	100	44	44	43	46
Making Sandwich	Version I	13	13	27	13	49	12	32	44	100	98	90	85
	Version II	13	13	27	11	49	0	0	44	98	100	86	89
	Version III	9	13	25	9	38	6	4	43	90	86	100	69
	Version IV	13	19	53	57	30	34	40	46	85	89	69	100

Fig. 8. Action-classification results. Similarity values between event tables of the real test data set.

scenes much bigger than the similarity between non-type-similar actions, except in one case. For the fourth version of *Making Sandwich* and the fourth version of *Moving Object* we receive a large similarity of 57%. This may happen in some cases when action primitives are quite similar and, in addition, noise in the data leads to a few spurious nodes and false tracking.

Moreover, the results showed that the manipulated objects in each action type can be categorized according to their roles in the actions. Fig. 9 illustrates the categorization results of objects that performed same roles in different versions of actions. As an example, the apple and orange are in the same group since they are being manipulated in the *Moving Object* action.

Notwithstanding some remaining problems, the results shown here clearly demonstrate that it is possible to classify objects and actions in scenes with limited context without prior (model) knowledge.

V. Discussion and Future Work

We presented a novel algorithm that represents a promising approach for recognizing actions *without* requiring prior object knowledge, and for categorizing objects solely based on their exhibited role within an action sequence. Our framework is mainly based on the analysis of object relations in the spatiotemporal domain. We are aware of the fact that

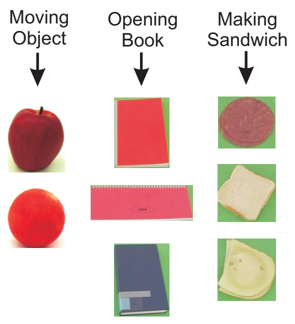


Fig. 9. Object categorization results. In each action type the manipulated objects can be detected based on their action roles.

"segment permanence" (i.e., reliable tracking) needs to be assured without which our method would fail. Clearly on the computer vision side improvements can be made to better assure this. This, however, is not the point of this paper. As far as we see it this is one of the first papers in which the categorization of object-action relations has become possible in a model free way. This procedure can thus be entirely based on the experimentation of the robot (here simulated by a human). Hence we arrive at a very high sub-symbolic representational level in a fully grounded way. From there on the grounded development and the learning of symbols (for example verbal utterances) which describe actions should be easier than before and this has been deemed as one of the major challenges in cognitive robotics. Furthermore, it should also be possible to "backwards unwrap" the learned event tables (the OACs) and this way generate an action. Obviously complex inverse kinematic (and dynamic) problems need to addressed to arrive at an actual movement sequence. However, the event graphs specify the fundamental "breaking points" whenever certain object relations change. Therefore movement segments between two such breaking points could be seen as motor primitives. The execution of such a primitive may then be optimized by whatever means but one always must assure that its starting point (prior) and its endpoint (posterior) corresponds to two subsequent entries in the event table.

The proposed algorithm has been applied to three different real action sequences of scenes containing limited context. Each action type had four different versions which differed in trajectories, speeds, hand positions, and object shapes. The experimental results showed that the agent can categorize all three action types by measuring the amount of similarity between action sequences and also categorize the participating manipulated objects according to their roles in the actions.

Several extensions of this algorithmic framework will be pursued in the future (for example 3D-tracking to eliminate false touching-relations introduced by the 2D-perspective projection). By using depth information either from stereo or a real-time depth camera [21] we will assign $2\frac{1}{2}$ attributes to the segments, which will allow to describe abstract invariant $2\frac{1}{2}$ segment relations, e.g. the relative angle between approximative approximated segment-surface normals.

Furthermore, determining similarity values between each action makes the whole system computational expensive, especially if the database contains a lot of training data. In order to avoid this problem, we plan to construct a template main-graph model for each kind of action. Template graph models can be constructed by considering the main graphs of a scenario which accurately represents the respective action type. Actions will then be classified by calculating the similarity values with those template models instead of with one another. In addition to this, we intend to let the agent learn the template main-graph models from a training data set.

In the future, we plan to apply our framework to more complex and even longer scenes containing high-level context and currently we are working on the disentangling of a complete "making a breakfast" sequence. Because the dimensions of the event tables are expected to increase, we are decomposing these action sequences into action subsequences and analyze each of them separately. We are also planning to compress the event tables without losing temporal relations. This way we are decreasing the computation time and can calculate the actual similarity between long actions containing similar sub-actions in different order. A parallel implementation of the framework on GPUs for real-time robotics applications is currently being implemented.

Thus, this study is one of the first to show that it is indeed possible to treat objects and actions as conjoint entities as suggested by the abstract idea of object-action complexes (OACs, [1], [2]). This is the first description of our new approach and the discussion above shows that it seems to have high potential. In general this contribution shows that this complex concept is algorithmically treatable and therefore we believe that the OAC indeed provides a promising approach for treating problems involving cause-effect relations in cognitive robotics.

VI. Acknowledgment

We thank Tomas Kulvicius for valuable discussion.

References

[1] C. Geib, K. Mourao, R. Petrick, N. Pugeault, M. Steedman, N. Krüger, and F. Wörgötter, "Object action complexes as an interface for planning and robot control," in *IEEE RAS Int. Conf. Humanoid Robots (Genova)*, 2006.

[2] F. Wörgötter, A. Agostini, N. Krüger, N. Shylo, and B. Porr, "Cognitive agents - a procedural perspective relying on predictability of object-action complexes (oacs)," *Robotics and Autonomous Systems*, vol. 57, no. 4, pp. 420–432, 2009.

[3] J. Gibson, "The ecological approach to visual perception." Boston: Houghton Mifflin, 1979.

[4] S. Hart and R. Grupen, "Intrinsically motivated affordance learning," in *Workshop on Approaches to Sensorimotor Learning on Humanoids, IEEE Conference on Robotics and Automation (ICRA)*, 2009.

[5] 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," *International Journal of Computer Vision*, vol. 66, no. 3, pp. 231–259, 2004.

[6] C. H. Andez, C. Hern, E. Esteban, and F. Schmitt, "Silhouette and stereo fusion for 3d object modeling," *Computer Vision and Image Understanding*, vol. 96, pp. 367–392, 2004.

[7] M. Tomono, “3d object modeling and segmentation based on edge-point matching with local descriptors,” in *ISVC ’08: Proceedings of the 4th International Symposium on Advances in Visual Computing*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 55–64.

[8] R. Opara and F. Wörgötter, “A fast and robust cluster update algorithm for image segmentation in spin-lattice models without annealing - visual latencies revisited,” *Neural Computation*, vol. 10, pp. 1547–1566, 1998.

[9] N. Shylo, F. Wörgötter, and B. Dellen, “Ascertaining relevant changes in visual data by interfacing ai reasoning and low-level visual information via temporally stable image segments,” in *Proceedings of the International Conference on Cognitive Systems (Cogsys 2008)*, 2009.

[10] B. Dellen and F. Wörgötter, “Disparity from stereo-segment silhouettes of weakly textured images,” in *Proceedings of the British Machine Vision Conference*, 2009.

[11] B. Dellen, E. E. Aksoy, and F. Wörgötter, “Segment tracking via a spatiotemporal linking process in an n-d lattice model (submitted),” 2009.

[12] J. Modayil, T. Bai, and H. Kautz, “Improving the recognition of interleaved activities,” in *Proceedings of the 10th international conference on Ubiquitous computing*, 2008, pp. 40–43.

[13] L. Liao, D. Fox, and H. Kautz, “Location-based activity recognition using relational markov networks,” in *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, 2005, pp. 773–778.

[14] S. Hongeng, “Unsupervised learning of multi-object event classes,” in *Proc. 15th British Machine Vision Conference*, 2004, pp. 487–496.

[15] M. Sridhar, G. A. Cohn, and D. Hogg, “Learning functional object-categories from a relational spatio-temporal representation,” in *Proc. 18th European Conference on Artificial Intelligence*, 2008.

[16] R. B. Potts, “Some generalized order-disorder transformations,” *Proc. Cambridge Philos. Soc.*, vol. 48, pp. 106–109, 1952.

[17] R. H. Swendsen and S. Wang, “Nonuniversal critical dynammics in monte carlo simulations,” *Physical Review Letters*, vol. 76, no. 18, pp. 86–88, 1987.

[18] M. Blatt, S. Wiseman, and E. Domany, “Superparametric clustering of data,” *Physical Review Letters*, vol. 76, no. 18, pp. 3251–3254, 1996.

[19] C. von Ferber and F. Wörgötter, “Cluster update algorithm and recognition,” *Physical Review E*, vol. 62, pp. 1461–1664, 2000.

[20] M. F. Sumsi, “Theory and algorithms on the median graph. application to graph-based classification and clustering,” Ph.D. dissertation, Universitat Autonoma de Barcelona, 2008.

[21] B. Dellen, G. Alenya, S. Foix, and C. Torras, “3d object reconstruction from swissranger sensor data using a spring-mass model,” in *Proceedings of the International Conference on Computer Vision Theory and Applications (Visapp 2009)*, vol. 2, 2009, pp. 368–372.

3D Edge and Contour Relation Histograms for Object Structure Encoding and Identification

Emre Başeski[*,a], Nicolas Pugeault[b], Justus H. Piater[c], Norbert Krüger[a]

[a]*The Mærsk Mc-Kinney Møller Institute, University of Southern Denmark Odense, Denmark*
[b]*Centre for Vision, Speech and Signal Processing, University of Surrey Guildford, United Kingdom*
[c]*Department of Electrical Engineering and Computer Science, University of Liège Liège, Belgium*

Abstract

This paper investigates the use of second-order relations between local and semi-global 3D edge features for object identification in visual scenes. Relations between 3D features have the inherent property of being pose-invariant, and therefore allow for direct comparison without knowledge of object pose. We define histograms of relations such as coplanarity, distance, and color difference between local 3D edge descriptors and semi-global groups thereof, and use them to encode the object structure. Histogram intersection is then used to identify a set of low-textured objects that are poorly described by classical feature descriptors. Moreover, the relation histograms are shown to describe structural properties of the objects, and the histogram similarities between objects reflect structural similarities between them.

Key words: 3D features, relation histograms, object identification

1. Introduction

After their introduction by Swain and Ballard (1991), histogram-based approaches have been used in computer vision for various tasks (see, e.g., Pass and Zabih (1996); Schreiber (2008); Zhang et al. (2006)) since they are easy to compute, the computational cost of comparing histograms is small, and additional visual cues can easily be added to the system as another dimension in the histogram. Histograms have also been used in the context of object encoding, retrieval and recognition (see, e.g., Evans et al. (1993); Huet and Hancock (1998); Saykol et al. (2005); Schiele and Crowley (1996)).

In this work, objects are described as multidimensional histograms of a set of relations between 3D edge features, and between groups of features. The relations considered cover both appearance and geometric aspects: they are coplanarity, normal distance and mean color. Dissimilarity between two objects can then be viewed as the distance in this relational space, and thus reflects structural similarities between distinct objects. This distance is estimated as the intersection between the two histograms encoding the respective objects.

One common approach to the problem of defining shapes in terms of descriptors is to use robust local features such as SIFT (Lowe, 2004). These features rely heavily on texture rather than object shape (see, e.g., Mikolajczyk and Schmid (2005)), whereas the importance of the latter for visual recognition has been shown in many context (see, e.g., Biederman (1987); Ullman (1996)). As a possible approach to encode the shape of objects, geometrical and appearance-based visual feature relations have been successfully used. Vosselman (1992) discusses relational matching techniques based on graphs. In Wang et al. (1997), 2D spatial relations are used as features for object recognition. Similarly, Henricsson (1995) uses geometric relations such as proximity, cocurvilinearity and symmetry between contours to describe objects by combining these relations. In more recent work, Savarese et al. (2006) use correlograms of spatial correlations between visual words for object recognition.

Relation histograms in 2D have been applied to various tasks like image retrieval and recognition. Evans et al. (1993) employ the distribution of pairwise geometric relationships between local features as histograms to represent objects. In Huet and Hancock (1998), two-dimensional histograms of relative angle and relative position of pairwise attributes for directed segments are used in the context of image retrieval in 2D. The histogram-based method proposed by Saykol et al. (2005) utilizes distance, angle and color histograms in 2D for object-based querying. Note that all the histogram-based methods mentioned above that make use of relational space use 2D approaches.

The main novelty of this work is the use of relations between three-dimensional visual features, reconstructed by, e.g., stereopsis. Although 2D relations are easier to extract and are subject to less uncertainty, some important relations such as, e.g., coplanarity or 3D distance are essentially three-dimensional.

[*]Corresponding author
Email addresses: emre@mmmi.sdu.dk (Emre Başeski), n.pugeault@surrey.ac.uk (Nicolas Pugeault), Justus.Piater@ULg.ac.be (Justus H. Piater), norbert@mmmi.sdu.dk (Norbert Krüger)

Moreover, we analyze the approach both for local and semi-global[1] visual features with similar relations. Also, because the histograms are built from both geometric (e.g., coplanarity and distance) and appearance-based (mean color) relations, they express both types of information about objects.

The approach is evaluated in the context of object identification using a database of 10 objects featuring low amounts of texture, viewed from 5 different poses each[2]. The performance of the proposed system is evaluated through an experiment where we asked how well relation histograms encode structural differences and similarities between objects, and how reliably the same object is identified when viewed in a different pose. We also evaluated the relative advantages and drawbacks of histograms built from relations between either local edges or semi-global visual features.

In general, various aspects of visual information (e.g., color, edge structures, texture, 2D, 3D, local or global features or feature relations) can be used as input for high-level tasks such as object identification. It is likely that top performance can only be reached by merging these different aspects with additional, contextual knowledge. However, the focus of this article is to specifically demonstrate the potential of 3D edge and contour relations.

In the following section we describe the local and semi-global features used in this work. After presenting a set of relations between local features (Section 2.1) and between contours (Section 2.2), we discuss in Section 3 how relation histograms are created and used. The experiment that was conducted to evaluate the performance for both local features and contours is presented in Section 4. We conclude with a discussion of the results in Section 4.2 and 5.

2. Visual Representation

In this work, we make use of the visual representation presented by Krüger et al. (2004) and Pugeault (2008). A calibrated stereo camera setup is used to create sparse 2D and 3D features, namely multi-modal[3] *primitives*, along image contours. 2D primitives represent a small image patch in terms of position, orientation, phase and color, and are denoted as

$$\pi = (\mathbf{x}, \theta, \phi, (\mathbf{c_l}, \mathbf{c_m}, \mathbf{c_r}))\,. \tag{1}$$

These features are then matched across two stereo views, and pairs of corresponding 2D features allow the reconstruction of a 3D equivalent, encoded by the vector

$$\Pi = (\mathbf{X}, \Theta, \Phi, (\mathbf{C_l}, \mathbf{C_m}, \mathbf{C_r}))\,. \tag{2}$$

An example is shown in Figure 1 which illustrates what kind of information exists on different levels of the representation. A stereo image pair (Figure 1 (a)) is used to create filter responses (Figure 1 (b)) which then give rise to the multi-modal 2D primitives (Figure 1 (c)).

These 2D features are grouped together by using the perceptual organization scheme presented by Pugeault et al. (2006) to create 2D contours (Figure 1 (d)). Local 3D features (Figure 1 (e)) are reconstructed from the matching 2D features and are then grouped together to create 3D contours. Note that 2D primitives cover an image patch and 3D primitives cover a volume in space. The primitives divide these patches into two halves, denoted left and right relatively to their orientation. Therefore, for every 2D and 3D primitive, color is defined as left, middle and right color.

In the following, *primitives* will denote local 3D primitives, and *contours* will denote semi-global 3D contours, unless specified otherwise.

2.1. Relations between Local Features

The sparse and symbolic nature of the primitives allows for an easy definition of relevant spatial relations in 2D and 3D. In this section, relations between local 3D primitives are presented.

Normal Distance: Because the primitives locally encode contours, the Euclidean distance would not accurately capture the distance between two features. Instead, we define the normal distance as the distance between the line described by the position $\mathbf{X}$ and orientation Θ of one feature and the position of the other feature. The normal distance between two primitives $\mathbf{\Pi}_i$ and $\mathbf{\Pi}_j$ is taken to be infinite if $\mathbf{\Pi}_j$ is outside the cylindrical volume surrounding $\mathbf{\Pi}_i$ (see Figure 2 (a)), and is otherwise defined as the distance between $\mathbf{\Pi}_j$ and the line through the location of $\mathbf{\Pi}_i$ parallel to the orientation vector Θ of $\mathbf{\Pi}_i$.

Cocolority: Two primitives are defined to be cocolor if their parts that face each other have similar color. We define the cocolority (see Figure 2 (b)) of two 2D primitives π_i and π_j as:

$$coc(\pi_i, \pi_j) = \mathbf{d}_c(\mathbf{c}_i, \mathbf{c}_j), \tag{3}$$

where $\mathbf{c}_i$ and $\mathbf{c}_j$ are the CIE **Lab** representation of the colors of the parts of the primitives π_i and π_j that face each other, and $\mathbf{d_c}(\mathbf{c}_i, \mathbf{c}_j)$ is the CIE 1994 color difference (see Hunt (1998)) between $\mathbf{c}_i$ and $\mathbf{c}_j$. Cocolority of two 3D primitives is computed using their 2D projections.

The cocolority relation between pairs of features for two objects with different colors may give similar results although the actual color of the features are different; for example, color difference in rgb space between gray (0.5,0.5,0.5) and black (0,0,0) is same as the color difference between gray and white (1,1,1). Therefore, instead of using cocolority value as a relation, we use $(\mathbf{c}_i + \mathbf{c}_j)/2$ as the mean color of two primitives and use cocolority to estimate how reliable the mean color is.

Coplanarity: The coplanarity relation (see Figure 2 (c)) between two primitives $\mathbf{\Pi}_i$ and $\mathbf{\Pi}_j$ is defined as:

$$cop\,(\mathbf{\Pi}_i, \mathbf{\Pi}_j) = \frac{\Theta_j \times \mathbf{V}_{ij}}{|\Theta_j \times \mathbf{V}_{ij}|} \bullet \frac{\Theta_i \times \mathbf{V}_{ij}}{|\Theta_i \times \mathbf{V}_{ij}|}, \tag{4}$$

where $\mathbf{V}_{ij}$ is the vector connecting the two primitives' positions, $\times$ is the vector (cross) product, and $\bullet$ is the inner product.

[1]A composition of local features

[2]This dataset is available at *http://www.mip.sdu.dk/covig/histogram.html*, and contains the required calibration parameters and stereo image pairs for every object.

[3]That is, including different visual modalities such as stereo and color.

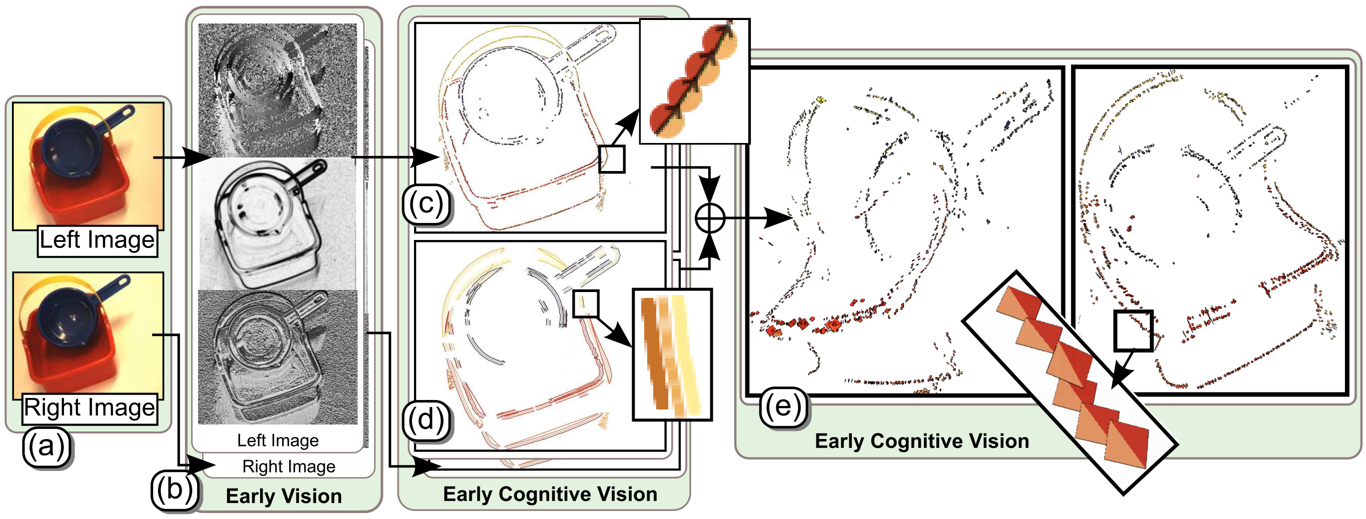


Figure 1: An overview of the visual representation. (a) Stereo image pair, (b) Filter responses, (c) 2D primitives, (d) 2D contours, (e) 3D primitives from two different viewpoints.

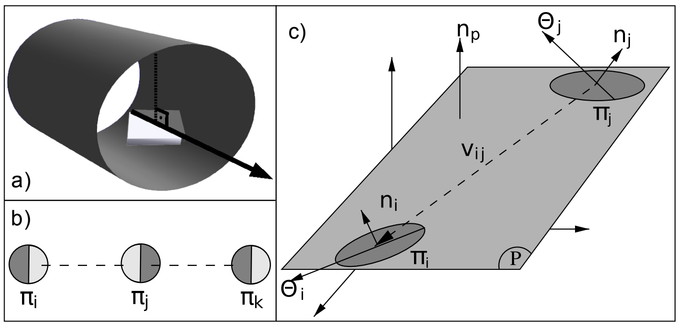


Figure 2: Illustration of relations between primitives. (a) Normal distance. (b) Cocolority. (c) Coplanarity.

2.2. *3D Contours and Their Relations*

Collinear and similar primitives are linked together to form structures denoted as multi-modal *contours* that contain not only geometrical but also appearance information. The multi-modality of contours (e.g., left, middle and right color) originates from the local primitives from which the contours are computed. The geometrical and visual smoothness of 3D contours give rise to relations between them. Since contours represent larger portions of scenes than primitives, contours and their relations can give a more global insight about the scene. The perceptual relations between 3D contours that are used in this work can be summarized as follows:

Normal Distance: The normal distance between two 3D contours is defined by the distance between one contour's centroid and the 3D line defined by the other's centroid and first principal component direction[4]. Therefore, the distance between the i^{th} and j^{th} contours in the scene is defined as

$$\frac{|\mathbf{w}_i - (\mathbf{w}_i \cdot \mathbf{u}_i)\mathbf{u}_i| + |\mathbf{w}_j - (\mathbf{w}_j \cdot \mathbf{u}_j)\mathbf{u}_j|}{2}, \tag{5}$$

where $\mathbf{w}_i = (\mathbf{k}_j - \mathbf{k}_i)$, $\mathbf{w}_j = (\mathbf{k}_i - \mathbf{k}_j)$, $\mathbf{k}_i$ is the centroid, and $\mathbf{u}_i$ is the first principal component of the i^{th} contour. Note that both terms of the numerator in Equation 5 are the point-to-line distance formula. An example is shown in Figure 3 (c).

Cocolority: The mean color of a contour is defined in CIE **Lab** color space as the average of **L**, **a** and **b** components of the color of the primitives that are part of the contour. Since the primitives have left, right and middle colors, every contour has mean left, right and middle colors as well. Cocolority of two contours is defined as the color difference (see Equation 3) between the mean colors on the contours' sides that are facing each other. For example, contours satisfying the cocolority relation for a selected contour are presented in Figure 3 (d). In a fashion similar to the primitives' cocolority definition, a mean color is calculated for the contours' sides that are facing each other. Cocolority is then used to estimate the reliability of the mean color.

Coplanarity: Coplanarity between two contours is inversely proportional to their distance to a common plane. This calculation employs a plane fitting algorithm based on *renormalization* (Kanazawa and Kanatani, 1995) that takes the uncertainty of the data into account. The coplanarity between the i^{th} and j^{th} contours in the scene is defined as

$$\frac{1}{N}\sum_{k=0}^{N}(\mathbf{p}_i^k)^T(\Lambda_i^k)^{-1}(\mathbf{p}_i^k) + \frac{1}{M}\sum_{l=0}^{M}(\mathbf{p}_j^l)^T(\Lambda_j^l)^{-1}(\mathbf{p}_j^l), \tag{6}$$

where N and M are the numbers of primitives in i^{th} and j^{th} contours respectively, $\mathbf{p}_i^k = (\mathbf{x}_i^k - \hat{\mathbf{x}}_i^k)$, Λ_i^k is the uncertainty matrix of the k^{th} feature (Pugeault et al., 2008) in the i^{th} contour, $\mathbf{x}_i^k$ is its position vector, and $\hat{\mathbf{x}}_i^k$ is the point closest to $\mathbf{x}_i^k$ on the plane that has been fit to feature locations of the i^{th} and j^{th} contours by renormalization. An example is shown in Figure 3 (b).

[4]The eigen-vector of the highest eigen-value in PCA

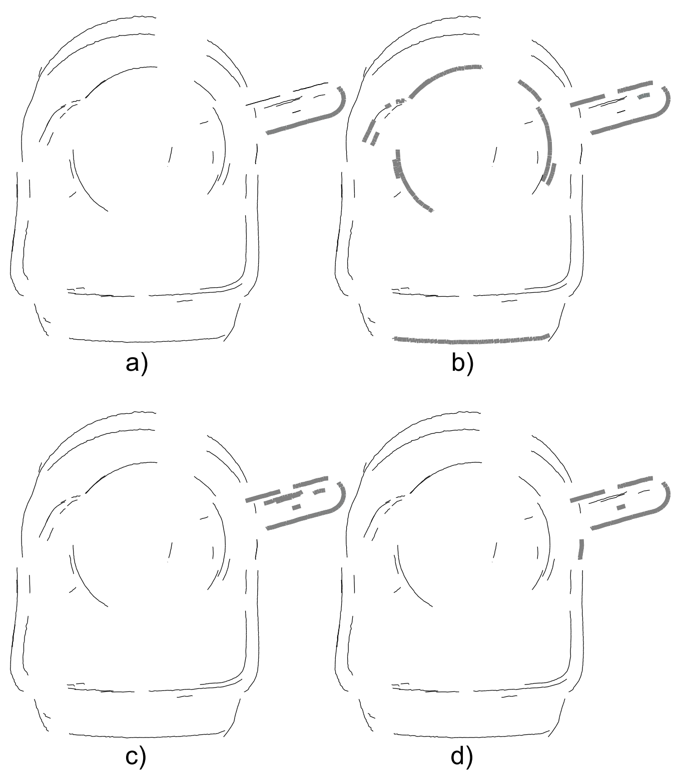


Figure 3: Illustration of 3D relations. (a) 3D contours of the object in Figure 1(a). A selected contour is marked in grey. (b) Contours coplanar to the selected contour with maximum Mahalanobis distance of 2 are shown in grey. (c) Contours at a maximum distance of 30 mm to the selected contour are shown in grey. (d) Contours that have a maximum color difference of 10 to the selected contour are shown in grey.

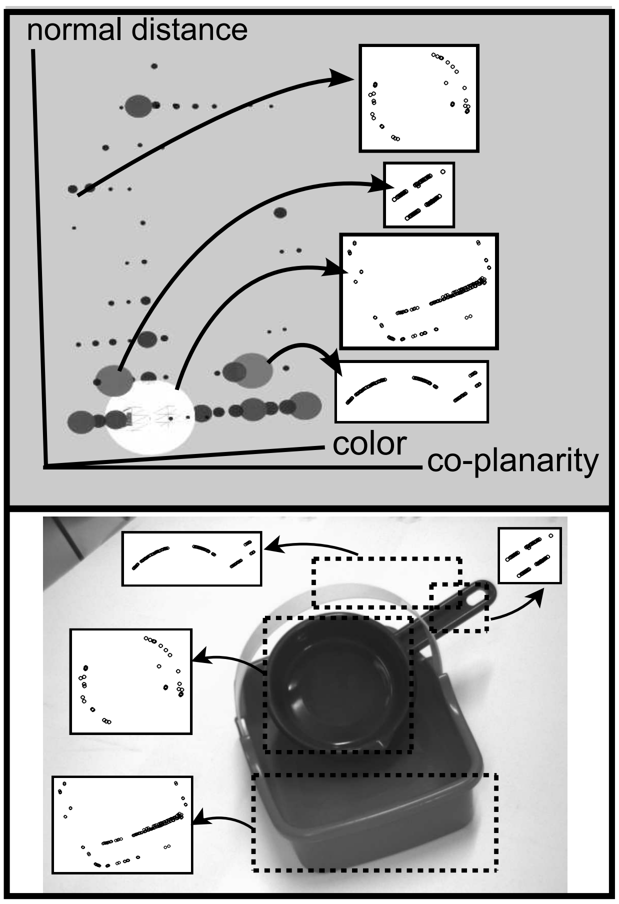


Figure 4: A sample 3-dimensional histogram (top) of primitive relations and the primitives that created specific bins of the histogram (bottom). The blobs represent the locations of histogram bins. Their brightness and size are proportional to the value stored in the bins (a high count is represented as a big and bright blob).

3. Object Structure Encoding via Relation Histograms

Since the geometrical structure of an object is independent of the pose in 3D, we make use of relations between 3D visual features that were discussed in Section 2. Objects are encoded as relation histograms where the axes are coplanarity, normal distance, and mean (L,a,b) components (see Section 2.1). Since the color is encoded with three components, the relation histograms are 5-dimensional. Due to the fact that the relations are defined for both local and semi-global features, we can create both local and semi-global relation histograms for the same object. Note that, departing from the conventional relation histogram approaches, we use not only geometrical (coplanarity and normal distance) but also appearance-based (components of the mean color) relations. The idea behind the selection of these axes is to split objects into parts that are both geometrically and visually similar. Therefore, instead of calculating the histograms from every feature pair, only cocolor pairs are used. An example histogram created from primitive relations is shown in Figure 4. Since we cannot display 5-dimensional histograms, in this example gray-scale mean color between primitive pairs is displayed instead of CIE **Lab** mean color.

The similarity between two histograms is calculated using a histogram intersection technique introduced by Swain and Ballard (1991). The intersection between normalized histograms H_1 and H_2 can be formulated as

$$\mathcal{D}(H_1, H_2) = \sum_{i=1}^{n} \min(H_1(i), H_2(i)), \tag{7}$$

where n is the total number of bins, and $H_j(i)$ is the value stored in the i^{th} bin of the j^{th} histogram. For normalized histograms, a perfect match results in 1 and a total mismatch results in 0.

4. Identification by Relation Histograms

Once the object structure is encoded as a relation histogram, the similarity of two objects can be measured by histogram intersection as discussed in Section 3. In this section, an identification methodology based on 5-dimensional relation histograms is discussed for the objects in Figure 5.

The main idea is to encode objects as histograms of coplanarity, normal distance and mean CIE **Lab** color of the facing features, and to use histogram intersection to measure the similarity between histograms. We applied the method on a variety of objects using relation histograms of both local and semi-global features. Since histogram techniques rely on statistical

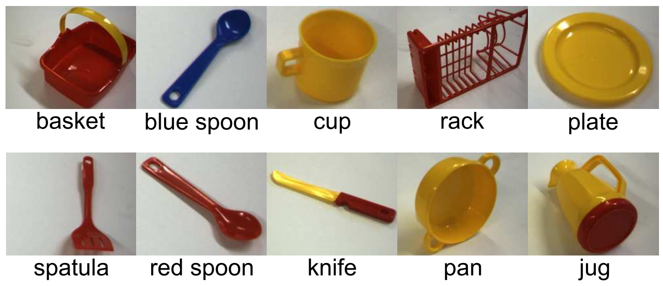


Figure 5: Objects used in the experiments.

variation in input data, it is important to have a large number of relations defining the object. Therefore, we restricted the size of semi-global features to a maximum of 10 local features to have fewer semi-global than local features while preserving a sufficiently large number for statistical stability.

The evaluation is based on a set of 10 objects, viewed in 5 poses each. The objects are toy kitchen utensils, which were specifically chosen for having little to no texture, making them difficult to identify using texture-based methods. An experiment was conducted based on different poses of the objects where 5 different poses of each object in Figure 5 were used to create two sets of 50 relation histograms, one for primitive and one for contour relations. Each histogram was then compared to every other histogram in the set to find the similarity between every pose of every object. We conclude the section with a brief discussion of the experimental results.

4.1. Comparison of Different Poses with Each Other

The experiment was conducted to analyze the discriminative power of relation histogram intersection for primitives and contours. Five different poses of each object in Figure 5 were used to create a total of 50 relation histograms, for both primitive and contour relations (All different poses are shown in Figure 7.). Every two histograms were compared to find the similarity score between every pose of every object. The results for both primitives and contours are presented in Figure 6. High values close to the diagonal in Figure 6 indicate good matches whereas other high values indicate either bad matches or structural similarity between objects. For example, all poses of the 'basket' object are found to be similar to all poses of the 'rack' object, for both primitives and contours. On the other hand, the relation histograms of the 'knife' object responded weakly to every object in the experiment.

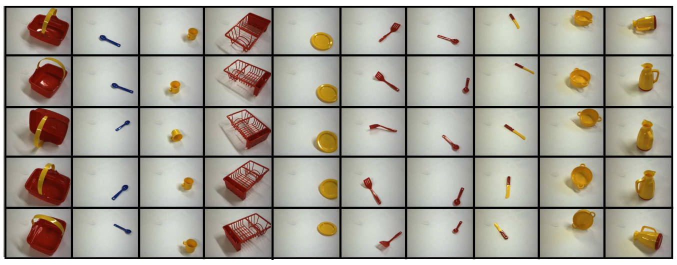


Figure 7: All different poses of the objects used in the experiment.

For every pose used in the experiment, similarity results were sorted in descending order to find the best matches. Since 5 poses were used, for every object and for any pose, ideally the first 4 matches (excluding itself) are the 4 other poses of the same object. Therefore, for any object in the experiment the maximum number of correct matches (i.e., matches between two poses of the same object) over all poses is $5 \times 4 = 20$. Figure 8 presents the number of correct matches ranked in the best 4 and best 6. When contour relations are used, the mean (μ) and standard deviation (σ) of the number of correct matches across objects in the best 4 is 14.1 and 3.96 respectively. The number of correct matches increases to $\mu = 15.8$ and $\sigma = 3.55$ for the best 6 matches. When primitive relations are used, we obtain $\mu = 15$ and $\sigma = 4.4$ for the best 4, and $\mu = 16.1$ and $\sigma = 4.15$ for the best 6. Therefore, although contour-based histograms produce higher similarity values as shown in Figure 6, primitive-based histogram intersection performs slightly better in terms of object discrimination (Figure 8).

Up to this point, we evaluated the discriminative power of relation histograms in terms of how good they encode object structure for different poses of the same object. Another interesting observation is the evaluation of the best match for every object, apart from itself. When contour-based histograms are used, for 50 poses used in the experiment, 46 of them had a best match with a pose of the same object which leads to 92% success for the best match. This performance have been found as 88% for the primitive-based histograms where 44 out of 50 poses had a best match with a pose of the same object.

To further analyze the discriminative power of relation histograms and how well they encode object structure, we reduced the 50x50 similarity matrices to a dimension of 50×2 using multidimensional scaling (Borg and Groenen, 2005). The results are plotted in Figure 9. There, both primitive- and contour-based relation histograms encode appearance-based (e.g., small distance between 'spatula' and 'red spoon' clusters) as well as structural (e.g., small distance between 'pan' and 'plate' clusters, 'red spoon' and 'blue spoon' clusters) information successfully. Also, the high degree of similarity between objects in Figure 6 (e.g., 'basket' and 'rack') appears in Figure 9 as well as the low degree of similarity (e.g., very large spread of the 'knife' cluster due to its simple structure which makes it similar to every other object).

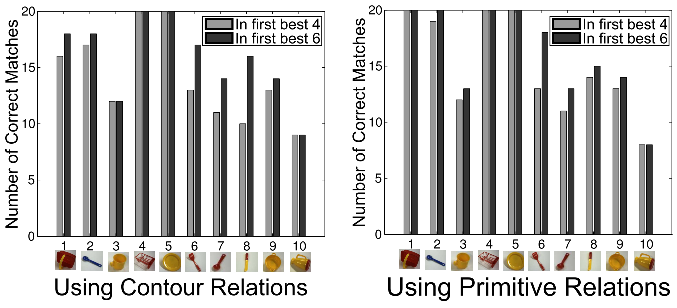


Figure 8: Number of correct matches in best 4 and 6 matches for the matrices in Figure 6. Each label in the graphs covers 5 poses.

Note that for n visual features we need to calculate $n(n-1)/2$

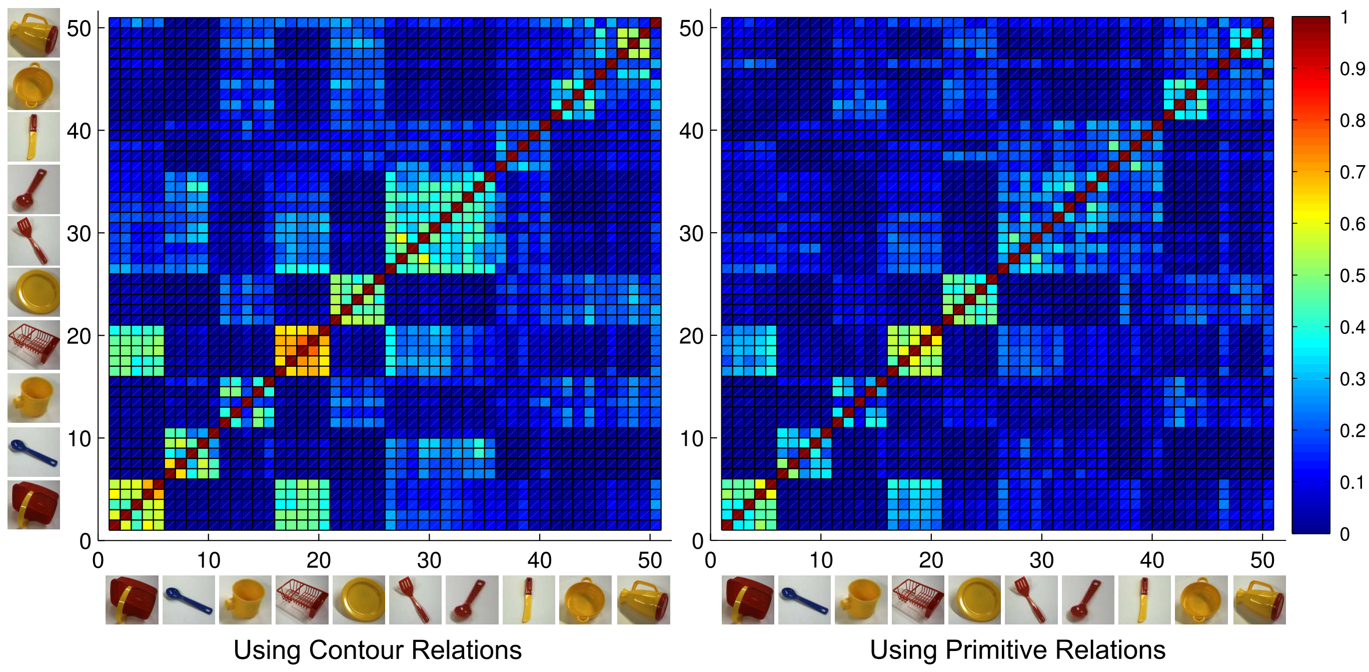


Figure 6: Similarity scores between 5 different poses of the 10 objects in Figure 5. Each small object figure on the axes illustrates only one pose out of 5. All the different poses can be seen in Figure 7.

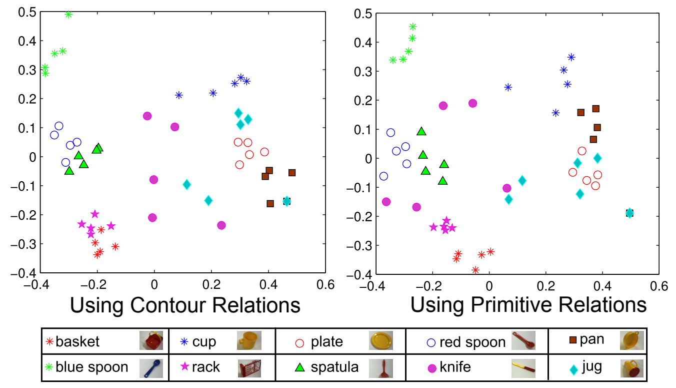


Figure 9: Similarity maps between different poses for primitives and contours.

relations. Therefore, compared to contour-based histograms, the high number of primitives in primitive-based histograms causes significant computational overhead (e.g., for a sample object, the creation of a contour-based histogram took 2 seconds while the primitive-based histogram took 120 seconds). This overhead could be reduced by using a subset of the primitives, but without knowledge of objects or perceptual organization, such subset creation may lead to loss of important information.

4.2. Discussion

In the previous experiment, we have tested the quality of object structure encoding and identification by using relation histograms. The collection of objects studied was chosen to contain little texture so that the distinction between objects needed to be mediated by structural considerations rather than feature matching. The previous experiment has outlined how relation-histograms encode structural properties of objects and allow for object identification. In the experiment, multidimensional scaling revealed the fact that the different poses of most objects were well clustered in relation space (e.g., 'pan', 'plate'), whereas some others were not (e.g., the 'knife'). Moreover, the clusters describing objects sharing structural (e.g., 'blue spoon' and 'red spoon') or appearance properties (e.g., 'pan' and 'plate') generate nearby clusters in relation space. When looking at histogram similarities, we see that higher similarities are achieved between different poses of the same objects. The poor results for the 'knife' are explained by the large spread of the different poses in relation space. The 'red spoon' and 'red spatula' are quite similar in terms of both structure and appearance, leading to a difficult distinction between the two.

Note that, the semi-global relations are extensions of relations between local features and performance comparison between histograms that are based on local and semi-global relations is possible. Overall, both contour and primitive-based histograms performed similar in terms of object structure encoding and identification. We demonstrate that although local features benefit from the statistical stability due to their numerosity compared to semi-global features, the latter benefit from more robust relations and less amount of computational cost.

5. Conclusions

We discussed the use of 3D relation histograms to encode structural and appearance properties of objects. This was assessed using a collection of low-textured objects, and was shown to be adequate for the purpose of object identification under unknown pose. The use of contour (semi-global) rather than primitive (local) relation histograms lead to a considerable reduction of the relational space and therefore of the computational cost.

In summary, we showed that 3D relations can make a significant contribution to object identification due to their inherent invariance under viewpoint transformations. Of course in an ideal system, such 3D relations only represent one aspect which needs to be merged with different cues.

Acknowledgment

This work was conducted within the EU Cognitive Systems project PACO-PLUS (IST-FP6-IP-027657) funded by the European Commission.

References

Biederman, I., 1987. Recognition-by-components: A theory of human image understanding. Psychological Review 94 (2), 115–147.

Borg, I., Groenen, P. J. F., September 2005. Modern Multidimensional Scaling: Theory and Applications (Springer Series in Statistics), 2nd Edition. Springer, Berlin.

Evans, A., Thacker, N., Mayhew, J., 1993. The use of geometric histograms for model-based object recognition.

Henricsson, O., 1995. Inferring homogeneous regions from rich image attributes. In: Automatic Extraction of Man-Made Objects from Aerial and Space Images. Birkhuser Verlag, pp. 13–22.

Huet, B., Hancock, E. R., 1998. Relational histograms for shape indexing. In: ICCV. pp. 563–569.

Hunt, R., 1998. Measuring Colour. 3rd edition. Fountain Press, Kingston-upon-Thames.

Kanazawa, Y., Kanatani, K., 1995. Reliability of Fitting a Plane to Range Data. IEICE transactions on information and systems 78 (12), 1630–1635.

Krüger, N., Lappe, M., Wörgötter, F., 2004. Biologically Motivated Multimodal Processing of Visual Primitives. The Interdisciplinary Journal of Artificial Intelligence and the Simulation of Behaviour 1 (5), 417–428.

Lowe, D. G., 2004. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 2 (60), 91–110.

Mikolajczyk, K., Schmid, C., 2005. A performance evaluation of local descriptors. IEEE Trans. Pattern Anal. Mach. Intell. 27 (10), 1615–1630.

Pass, G., Zabih, R., 1996. Histogram refinement for content-based image retrieval. In: IEEE Workshop on Applications of Computer Vision. pp. 96–102.

Pugeault, N., 2008. Early Cognitive Vision: Feedback Mechanisms for the Disambiguation of Early Visual Representation. Verlag Dr. Muller.

Pugeault, N., Kalkan, S., Başeski, E., Wörgötter, F., Krüger, N., 2008. Reconstruction uncertainty and 3d relations. In: Proceedings of Int. Conf. on Computer Vision Theory and Applications (VISAPP'08).

Pugeault, N., Wörgötter, F., Krüger, N., 2006. Multi-modal Scene Reconstruction Using Perceptual Grouping Constraints. In: Proc. IEEE Workshop on Perceptual Organization in Computer Vision (in conjunction with CVPR'06).

Savarese, S., Winn, J., Criminisi, A., 2006. Discriminative object class models of appearance and shape by correlatons. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on. Vol. 2. pp. 2033–2040.

Saykol, E., Güdükbay, U., Ulusoy, O., 2005. A histogram-based approach for object-based query-by-shape-and-color in image and video databases. Image Vision Comput. 23 (13), 1170–1180.

Schiele, B., Crowley, J. L., 1996. Object recognition using multidimensional receptive field histograms. In: ECCV '96: Proceedings of the 4th European Conference on Computer Vision-Volume I. Springer-Verlag, pp. 610–619.

Schreiber, D., 2008. Generalizing the lucas-kanade algorithm for histogram-based tracking. Pattern Recogn. Lett. 29 (7), 852–861.

Swain, M. J., Ballard, D. H., 1991. Color indexing. Int. J. Comput. Vision 7 (1), 11–32.

Ullman, S., 1996. High-level Vision. MIT Press.

Vosselman, G., 1992. Relational Matching. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Wang, X., Keller, J., Gader, P., Sep 1997. Using spatial relationships as features in object recognition. Fuzzy Information Processing Society, 1997. NAFIPS '97., 1997 Annual Meeting of the North American, 160–165.

Zhang, H., Gao, W., Chen, X., Zhao, D., April 2006. Object detection using spatial histogram features. Image and Vision Computing 24 (4), 327–341.

Grasp Affordances from Multi-Fingered Tactile Exploration using Dynamic Potential Fields

Alexander Bierbaum, Matthias Rambow,
Tamim Asfour and Rüdiger Dillmann
Institute for Anthropomatics
University of Karlsruhe (TH)
Karlsruhe, Germany
{bierbaum,rambow,asfour,dillmann}@ira.uka.de

Abstract— **In this paper, we address the problem of tactile exploration and subsequent extraction of grasp hypotheses for unknown objects with a multi-fingered anthropomorphic robot hand. We present extensions on our tactile exploration strategy for unknown objects based on a dynamic potential field approach resulting in selective exploration in regions of interest. In the subsequent feature extraction, faces found in the object model are considered to generate grasp affordances. Candidate grasps are validated in a four stage filtering pipeline to eliminate impossible grasps. To evaluate our approach, experiments were carried out in a detailed physics simulation using models of the five-finger hand and the test objects.**

I. Introduction

Robotic grasping using multi-fingered hand constitutes a complex task and introduces challenging problems. For well known scenes a grasping or other manipulation process may be pre-programmed when using todays robots. On the other hand, adaptation of a grasping algorithm to formerly unknown or only partially known scenes remains a difficult task, to which different approaches have been investigated.

A classical approach consists in grasp analysis and planning, based on a geometric scene model. In force based grasp planning the forces and moments at selected grasping points are analyzed and matched against a grasp quality criterion considering e.g. force closure. This approach is usually independent of the hand kinematics. In contrast mere geometry based algorithms are tailored to specific gripper designs, especially in the context of multi-fingered hands. Comprehensive overviews on grasp planning are given in [1], [2]. Using grasp planning for previously unknown objects consequently introduces the difficulty of model building from sensor data which is delivered by robot perception. As alternatives to the mere planning approach online control algorithms driven by tactile information have been developed, which make use of *a priori* assumptions on the object to grasp, and control the grasping process by displacing robot fingers. Different control goals have been formulated for grasping convex objects in [3], [4] and later [5], where contact displacements are calculated in order to minimize a grasp quality cost function. The function values are computed using estimation of local surface parameters from haptic feedback, thus resulting in an online control scheme. A further extension capable of dealing with concavities on an object's surface was presented in [6]. Online grasping approaches using a discrete set of hand postures or motions have also been presented [7], [8].

Beside vision based methods tactile exploration may be used for 3D reconstruction of an unknown object, as tactile sensing solves some severe limitations of computer vision, such as sensitivity to illumination and limited perspective. A reconstructed 3D object model may be used for grasp planning and execution as shown e.g. in [9].

Single finger tactile exploration strategies for recognizing polyhedral objects have been presented and evaluated in simulation, see [10] and [11]. In [12] a method for reconstructing shape and motion of an unknown convex object using three sensing fingers is presented. In this approach friction properties must be known in advance and the surface is required to be smooth, i.e. it must have no corners or edges. Further, multiple simultaneous sensor contacts points are required resulting in additional geometric constraints for the setup.

In general, previous approaches in robot tactile exploration for surface reconstruction did not cover the problem of controlling multi-finger robot hands during the exploration process. Also, real world constraints such as manipulator limits or robustness over measurement errors have not been considered. In [13] we have presented first results on the application of a dynamic potential field control technique for guiding a multi-finger robot hand across the surface of an unknown object and simultaneously building a 3D model from contact data.

In this paper we extend our approach in tactile exploration to serve the purpose of extracting grasp affordances for a previously unknown object. Therefore, we have added modifications to our exploration strategy which lead to a homogenous exploration process and prevent sparsely explored regions in the acquired 3D model. We have added a grasp planning system based on a comprehensive geometric reasoning approach as initially reported in [14]. We chose a geometric reasoning approach here as object modelling from tactile exploration currently does not deliver the details required for force analysis and contact modelling, as it is performed in force-based grasp planners, e.g. [15]. As we believe that robustness and applicability of tactile exploration

and robotic grasping algorithms depend significantly upon the deployed hardware configuration, we have evaluated our approach in the framework of a physical simulator, reflecting non-neglectable physical effects such as manipulator kinematics, joint constraints or contact friction. As in related approaches we initially limit our scope to the exploration of static scenes, which means the objects are fixated during exploration and may not move during interaction, although we wish later to develop means of pose estimation and tracking for objects in dynamic scenes.
This paper is organized as follows. In the next section a short introduction to the potential field technique is given and the relevant details of the robot model are described. In Sec. IV-A we present the tactile exploration process and in Sec. IV-B grasp planning and execution. We give details on our simulation scenario and exploration results in Sec. V. Finally, our conclusions and outlook on our future work may be found in Sec. VI.

II. Potential field control

Artificial potential fields have originally been introduced for the purpose of on-line collision avoidance in the context of robot path planning [16]. In the original approach, real-time efficiency was emphasized over obtaining a complete planner. The basic idea is that the robot behaves like a particle influenced in motion by a force field. The field is generated by artificial potentials Φ_i, where obstacles are represented as repulsive potentials $\Phi_r(x) > 0$ and goal regions are represented as attractive potentials $\Phi_a(x) < 0$. The superposition property allows to combine potentials in an additive manner,

$$\Phi(x) = \sum_i \Phi_{r,i}(x) + \sum_j \Phi_{a,j}(x) \quad .$$

The force vector field or potential field F, which influences a *Robot Control Point* (RCP) at position x is defined as

$$F = -\nabla\Phi(x) \quad .$$

A major drawback of potential fields is the existence of local minima outside the goal configurations in which the imaginary force exerted on an RCP is zero. By applying harmonic potential functions it is possible to construct potential fields without spurious local minima for point-like robots. This is not the case with robots that can not be approximated by a point, e.g. a manipulator arm. These are likely to exhibit structural local minima which need to be treated by dedicated escaping strategies [17].

III. Robot hand kinematics, control and sensors

For exploration and grasping we consider a setup comprising a 6-DoF manipulator arm with a five finger robot hand attached to its *Tool Center Point* (TCP). The manipulator arm was modelled according to the Mitsubishi RM-501 five axis small-scale industrial manipulator, which is currently used as a research platform for dexterous haptic exploration in our lab. The model was augmented with a sixth DoF before the TCP to provide a larger configuration space. In

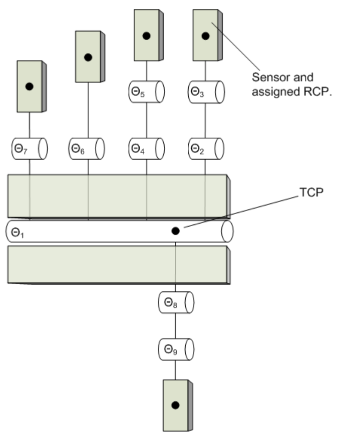


Fig. 1. Kinematics of the robot hand with joint axes, contact sensor locations (grey shaded) with assigned RCPs (black dots) and the TCP.

our exploration control scheme we apply controller outputs to a set of five RCPs, located at the fingertips of the robot hand and to the TCP of the manipulator. The kinematic model of the robot hand is shown in Fig. 1. The hand model provides nine degrees of freedom and is modelled according to the FRH-4 anthropomorphic robot hand presented in [18].
During haptic exploration we are interested in controlling the velocity vectors of the RCP's, which is a different task compared to trajectory control. In trajectory control the end-effector is commanded to follow a desired trajectory with the motion control goal of asymptotic tracking. Yet, the given exploration task does not induce specific trajectories due to the uncertainty in the environment. In our approach we compute the velocity vector applied to an RCP directly from the dynamic potential field, which guides the exploration process. In order to evaluate our concept in a physics simulation environment it was not required to develop a solution to the multipoint end effector inverse kinematic problem. Instead we chose to take advantage of the physical model of the robot system and directly specify velocity vectors to the RCPs by using a virtual actuator which is commonly available in physics simulation frameworks. The joint angles are then determined by solving the constrained rigid body system and a stable and consistent configuration of the robot hand is maintained. In general, this approach is known as *Virtual Model Control* (VMC), which is described in detail in [19]. In our case we specify joint constraints and joint friction for the robot model for achieving an appropriate force distribution over the joint serial paths, while we do not model a compliant behavior. The physics simulation is solved by using the *Inventor Physics Modeling API* (IPSA) which was introduced in [20].
We also make use of the dynamic potential field concept during initialization and grasp execution by placing attractive sources at desired target locations.
For haptic exploration and contact sensing during grasping, tactile sensors are required which we have modelled in our physics simulation. Of course the simulation environment itself may be regarded as omniscient and therefore it is

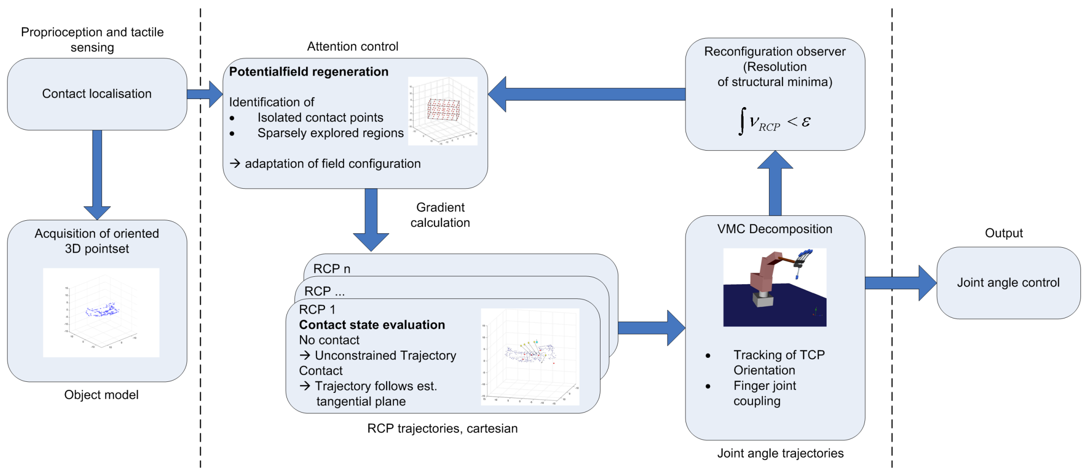

Proprioception and tactile sensing
Contact localisation
Acquisition of oriented 3D pointset
Object model
Attention control
Potentialfield regeneration
Identification of
• Isolated contact points
• Sparsely explored regions
→ adaptation of field configuration
Gradient calculation
RCP n
RCP ...
RCP 1
Contact state evaluation
No contact
→ Unconstrained Trajectory
Contact
→ Trajectory follows est. tangential plane
RCP trajectories, cartesian
Reconfiguration observer (Resolution of structural minima)
$\int v_{RCP} < \varepsilon$
VMC Decomposition
• Tracking of TCP Orientation
• Finger joint coupling
Joint angle trajectories
Output
Joint angle control

Fig. 2. Overview tactile exploration module.

possible to query all contact locations and force vectors during the interaction of modelled physical bodies. We have restricted contact sensing to dedicated sensor areas which cover the fingertips and the palm of the robot hand, see also Fig. 1. Further, we did not consider the contact force vector but only the contact location on the sensor area to provide a more realistic sensor model. This complies with current tactile sensor technology which in general can not provide both types of information. It is also possible to model more specific sensor characteristics such as a certain resolution in contact location or contact force tresholding, which we did not yet consider in our experiments.

IV. Exploration and Grasping System

The goal of our work is a system enabling a robot with a multi-fingered hand to explore an unknown object using tactile sensing and subsequentially find suitable grasps. Therefore, our system comprises a module for tactile exploration as depicted in Fig. 2. In the following we will describe the exploration and grasp planning process and transition between both modes of operation. Tactile exploration is executed in closed-loop and online in simulation. In contrast, the extraction of grasp affordances is an offline planning process executed subsequently to exploration. Please note that major details of the dextrous tactile exploration process have been reported in [13]. Therefore we will summarize the basic concept and point out the improvements to the original algorithm.

A. *Dexterous tactile exploration*

As a prerequisite the system requires a rough initial estimate about the objects position, orientation and dimension. In simulation we introduce this information to the system, while this information will be provided by a stereo camera system in the real application. From this information an initial potential field containing only attractive sources is constructed. The trajectories for the RCPs are continuously calculated from the field gradient, while contact point locations and normals are sensed and stored as oriented 3D point set. The normal vectors are estimated by averaging the finger sensor orientations within a spherical neighborhood around a contact point. The RCP trajectories are constrained depending on the contact state of the sensor associated with each RCP, which aims to produce tangential motion during contact.

The potential field is updated from the tactile sensor information as follows. If a contact is detected, a repelling source is inserted at the corresponding location in the potential field. Otherwise, if no contact is found in the circumference of an attractive source, this source becomes deleted from the field. The robot system is likely to reach structural minima during potential field motion. We therefore introduced a reconfiguration observer which detects when the TCP velocity and the mean velocity of all RCPs fall below predefined minimum velocity values. This situation leads to a so called *small reconfiguration* which is performed by temporarily inverting the attractive sources to repulsive sources. This forces the robot into a new configuration from which previously unexplored goal regions may be explored. As this method is not guaranteed to be free of limit cycles we further perform a *large reconfiguration* if subsequent small reconfigurations remain ineffective, i.e. the robot does not escape the structural minimum. During a large configuration the robot is moved to its initial configuration.

Our approach to extract grasp affordances relies on identifying suitable opposite and parallel faces for grasping. Therefore, we needed to improve the tactile exploration process as described above to explore the object surface in a dense scheme and prevent sparsely explored regions. The faces become extracted after applying a triangulation algorithm [21] upon the acquired 3D point set. Triangulation naturally generates large polygons in regions with a low contact point count. We use this property to introduce new attractive sources and guide the exploration process to fill the contact information gaps. Within fixed time step intervals we execute a full triangulation of the point cloud and rank the calculated faces by their size of area. We then add an attractive source at the centers of the ten largest faces. This leads to preferred exploration of sparsely explored regions, i.e. regions that need further exploration, and conseqently to a more reliable estimate for the objects surface.

We apply a similar scheme to isolated contact points, i.e. contacts that have no further contact points in their immediate neighborhood. We surround these by eight cubically arranged attractive charges. This leads to the effect that once an isolated contact is added, the according RCP now explores its neighborhood instead of being repelled to a more distant unexplored region.

B. Grasping Phase

As an exemplary application for our exploration procedure we have implemented a method for identifying grasp affordances from the oriented point set.

We did not choose a traditional force-based grasp planning algorithm as this would require to calculate a triangulated geometric object model from the 3D point set. The point set delivered by tactile exploration is inherently sparse and irregular and we found that most triangulation algorithms would fail to produce results in a usable way. Instead we found that extraction of local features from the point set is more robust than triangulation. We therefore chose a subset of a geometric reasoning approach as proposed in [14] in order to compute grasp affordances based on the acquired object information.

1) Extraction of grasping features: A grasp affordance contains a pair of object features from which the grasping points are determined in subsequent steps. In general, planar faces, edges and vertices of a polygonal object representation may be used as object features. We only consider planar faces in our implementation, as estimation and extraction of planar faces from the given 3D point set is much more reliable than that of edges or vertices. Therefore, we investigate the oriented 3D point set for neighboured contact points with similar normal vectors. Using a region growing method the contact points in adequate dense regions are assigned to faces. The original method is designed for parallel robot grippers therefore the grasp affordances found are consequently of a parallel type with opposing planar faces for grasping. We apply a mapping scheme as described below in Sec. IV-B.3 to compute the five finger tip target locations for the robot hand within each face.

2) Geometric feature filters: Initially every possible face pairing is considered as a potential grasp affordance. In a sequential geometric filtering process all grasps unlikely to be executed successfully with the given robot hand are eliminated from the set of all pairings. The filter parameters are chosen for the FRH-4 hand. We use a four stage filtering pipeline in our approach. The results of the filter stages are summed up to a score for each grasp affordance. Each filter is designed to return a value of 0 when disqualifying a pairing and value $1 \leq o \leq 1.1$ for accepting a pairing. As only grasp affordances with filter score ≥ 4 are considered valid this automatically implies that valid grasps have to pass all filter stages successfully.

- *Parallelism:* This filter tests the two faces for parallelism. Let $\vec{n}_1$ and $\vec{n}_2$ be the normal vectors of the two faces f_1 and f_2, ϕ the angle between $\vec{n}_1$ and $\vec{n}_2$ and ϕ_{max} the maximum angle for acceptance. The output o of the filter is:
$$o = \begin{cases} 0, & \text{if } \phi > \phi_{max} \\ 1 + \frac{(\phi_{max} - \phi)}{\phi_{max}} \cdot 0.1, & \text{otherwise.} \end{cases}$$
- *Minimum Face Size:* This filter tests the two faces for adequate size of area. Let a_1 and a_2 be the areas of the faces f_1 and f_2. The minimum area for acceptance is a_{min} , k_a is a normalization factor. Then the output o of this filter is:
$$o = \begin{cases} 0, \text{if } (a_1 < a_{min}) \vee (a_2 < a_{min}) \\ 1 + \min(\min(\frac{a_1}{k_a}, \frac{a_2}{k_a}), 0.1), & \text{otherwise.} \end{cases}$$
- *Mutual Visibility:* With this filter the two faces are projected into the grasping plane gp, which is the plane with the mean normal vector $\vec{n}_{gp}$ situated in the middle of the two faces f_1 and f_2. So let $f_{1\downarrow gp}$ and $f_{2\downarrow gp}$ be the projections of f_1 and f_2 onto gp. Then, a_{int} is the intersection area of $f_{1\downarrow gp}$ and $f_{2\downarrow gp}$. The minimum intersection area for acceptance is a_{min} , k_{mv} is a normalization factor. The filter's output is:
$$o = \begin{cases} 0, & \text{if } a_{int} < a_{min} \\ 1 + \min(\frac{a_{int}}{k_{mv}}, 0.1), & \text{otherwise.} \end{cases}$$
- *Face Distance:* The last filter incorporates the characteristics of the used manipulator tool, i.e. the robot hand. The filter checks if the robot hands spreading capability matches the distance of the faces. Let d be the distance between the centers of the faces f_1 and f_2, d_{min} and d_{max} are the minimum respectively maximum admitted distance values. Then the filters output is
$$o = \begin{cases} 0, & \text{For } d \notin [d_{min}, d_{max}] \\ 1, & \text{otherwise.} \end{cases}$$

3) Grasp execution: The grasp affordance with the highest score is used as the candidate for grasp execution. In a first step we compute the grasping position $\vec{p}_{tcp,a}$ of the TCP and the grasping approach direction as depicted in Fig. 3.

Initially we estimate the centers $\vec{c_1}$, $\vec{c_2}$ of the two faces f_1, f_2 as the centers of gravity of all contact points assigned to each face. From this we determine the center point $\vec{gp} = \frac{\vec{c_1} + \vec{c_2}}{2}$ on the line connecting the centers of the two faces. Then we analyse the first principle component $\vec{pc}$ of the acquired 3D point cloud and calculate the grasping position as
$$\vec{p}_{tcp,a} = \vec{gp} + (\vec{n}_{gp} \times \vec{pc}) \cdot d,$$
where d is a distance which considers the fingers length of the robot hand. The cross product $(\vec{n}_{gp} \times \vec{pc})$ becomes the approach direction. We only consider grasping the object from top. Therefore, in the case the coordinate $\vec{p}_{tcp,a}$ is below the object to grasp, we mirror its location across the center

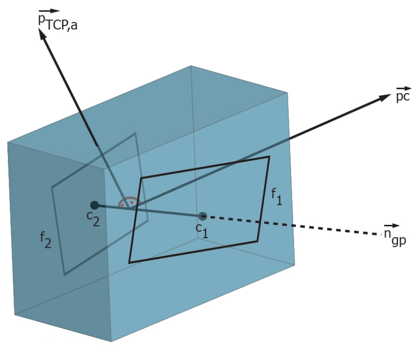


Fig. 3. Calculation of the grasp center point and approach direction.

between c_1 and c_2 and along the approach direction to a location above the object. Clearly, we make an assumption about the object's extension here. From the face pair of the grasp affordance finger tip target locations need to be computed. This is achieved by the following mapping scheme.

The target for the thumb $\vec{p}_{thumb,a}$ is set to be the center of the smaller of the two faces. We choose target locations $\vec{p}_{index,a}, \vec{p}_{middle,a}, \vec{p}_{ring,a}$ and $\vec{p}_{pinkie,a}$ for the opponent fingers around the center and in the plane of the larger of the two faces. The arrangement is chosen, so that it is perpendicular to the approach direction in the plane of target face. If the target location of ring finger or pinkie is not situated within the face area the fingers will not be used for grasping. This way the number fingers involved during grasping is automatically adapted.

Motion execution starts with the hand in an initial pose, as it is always reached after a large reconfiguration. From here we apply the potential field control to the RCPs and the TCP. Unlike during the exploration phase, the TCP and the RCPs share the set of repulsive potential sources while having individual attractive potential sources as mentioned above. Repulsive sources located in the target planes become deleted.

As long as the TCP is distant from its target $\vec{p}_{tcp,a}$ the potential field velocity control is only applied to the TCP while the finger joints remain open via direct joint control. When the TCP is close to its target we additionally apply the potential field control to the RCPs. If an RCP is not in use because the finger is not involved in grasping, the associated finger joints are still kept open. Further, the palm normal $\vec{n}$ is aligned towards $\vec{gp}$ by controlling forces acting on the hand's pitch and roll DoFs.

If the RCPs in use have approached the finger target locations, the fingers are closed and the corresponding sensors are checked for contact. Once all assigned RCP sensors are in contact with the object, potential field control is turned off and the finger joints are closed directly. The virtual fixture of the object then becomes disabled in the simulation and the robot arm moves back to its initial position with the object grasped and lifted.

V. Simulation Results

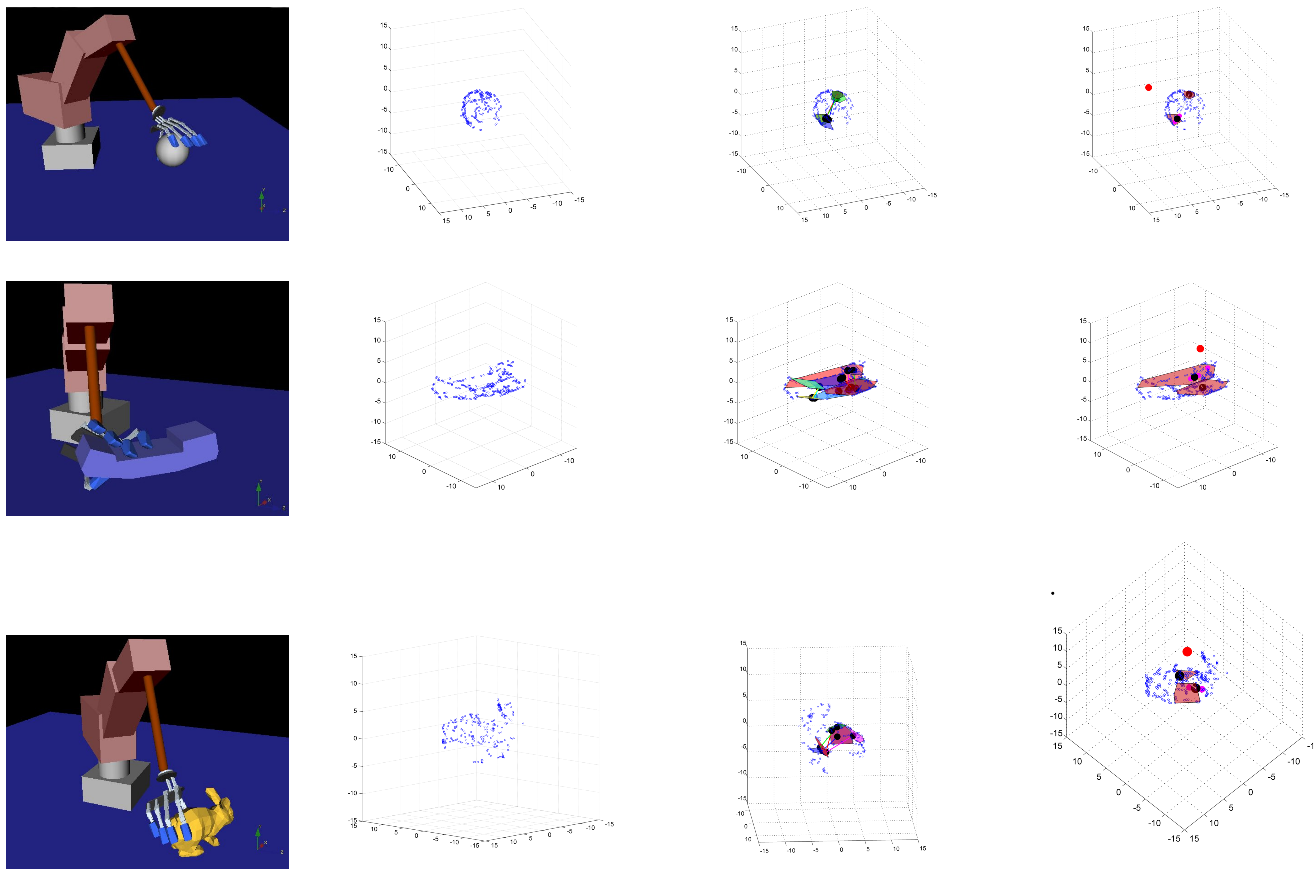
We evaluated our exploration and grasping system in several virtual scenes using our physics simulator with standard earth gravity $g_N = 9.81$ applied. For contacts Coulomb friction with a friction coeefficient $\mu = 0.5$ is considered. The virtual scenes were set up with different rigid objects of suitable size for grasping by the hand: a sphere, a telephone receiver and a rabbit. The objects are placed approximately in the center of the robots workspace. All objects are fixated floating above the simulators virtual ground to avoid interference, as we currently do not differ between contact between the object of interest and any other obstacle in the workspace. As described in Sec. IV-A the cubical bounding box of the object is computed from position and space occupancy estimates and used to initialize the exploration potential field. Grasp affordances are extracted after a fixed number of 2000 control time steps, whereby each control time step comprises ten simulation time steps with a temporal resolution of $T = 0.04s$.

Fig. 4 shows typical results. Here figures in column (c) show the 6 best candidate faces for grasping. The color indicates score ranking in following order: red, green, blue, magenta, cyan, yellow. Black dots indicate the center of a face, which is calculated as mean value of all points in the face. Colored lines connect corresponding centers of corresponding faces. In colum (d) the grasp affordance with the highest score is shown. Purple dots indicate grasping points for index, middle, ring and pinkie finger. Ring and pinkie grasping points are only plotted if they are used. The red dot marks the location of the attractive potential source for the TCP at start of the approaching phase. Naturally, the algorithm performs worse with objects exposing curved regions as the algorithm searches for planar faces. Therefore, only one grasp affordance was found for the sphere in the given exploration interval. The exploration of the rabbit shows similar results. Still, successful grasps can be performed with the grasp affordances identified.

In contrast, several affordances could be identified with the model of the telephone receiver consisting of large polygons. In general, the number of found grasp affordances increases with exploration time. The video accompanying this paper shows examples of tactile exploration and grasp execution for the rabbit.

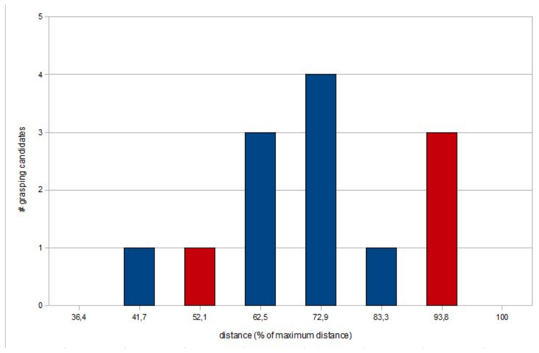
Beside experiments with different objects we also investigated performance of the system with objects placed at different positions and orientations in the workspace. For the experiments a grasp is considered successful if the manipulator can grasp and lift the object in simulation. We believe this is still a good approximation for reality as the simulator only calculates with rigid body dynamics and assumes point contacts. In reality such a robot system would be equipped with deformable rubber finger tips which will provide a significant larger contact area leading to higher tangential forces. Therefore we assume that a real robot system could execute the simulated successful grasps.

In a first experiment we placed the sphere, which is naturally

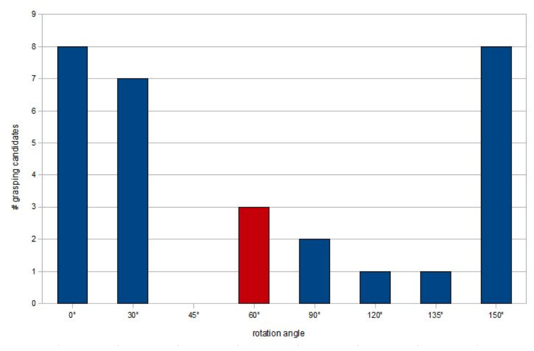


(a) (b) (c) (d)

Fig. 4. Typical simulation results from top to bottom: Sphere, telephone receiver, rabbit. Column (a) shows a virtual scene snapshot during exploration,(b) final point cloud, (c) grasp affordances, (d) best grasp and grasping points.



(a) Sphere at different distances



(b) Telephone receiver at different orientations

Fig. 5. Number of identified grasp affordances. Blue: successful grasp execution, red: failed grasp execution with best candidate.

invariant to rotations, at different distances ranging from minimum to maximum reaching distance for the manipulator arm in the workspace. Fig. 5(a) shows the number of found grasp affordances after $N = 2000$ exploration steps. After generation the grasp affordance with the highest score is executed as desribed in Sec. IV-B.3. In the figure a red bar indicates a failed grasp execution, a blue bar indicates a successful grasp execution, both with the best candidate grasp applied. The failed grasps may be deduced to the error between the estimated grasping plane and the local tangential plane of the sphere in combination with an inappropriate situation of the sphere within the robots workspace. This could be improved by increasing the exploration time in order to collect more contact data points.

In a second experiment we investigated the scheme with the robot model for sensitivity towards different orientations of an elongated object as the telephone receiver. Therefore, the receiver is placed in the scene with different orientations around the Y-axis (direction of gravity). The initial configuration can be seen in the mid image of Fig. 4(a). The receiver was situated in the workspace center area of the manipulator arm. The results are depicted in Fig. 5(b) and indicate that the receiver provides less features to extract grasp affordances from with its longer axis pointing toward the manipulator. The reasons for the failed grasp agree with those from experiment 1. Note that the receiver is not a symmetric object, therefore the number of grasping candidates is also not symmetric over rotation.

VI. Conclusions

In this paper we have presented a control scheme for tactile exploration and subsequent extraction and execution of grasp affordances for previously unknown objects using an anthropomorphic multi-fingered robot hand. Our approach is based on dynamic potential fields for motion guidance of the fingers. We have shown that grasp affordances may be generated from geometric features extracted from the contact point set resulting from tactile exploration. The complete control scheme was evaluated in a detailed physics simulation of the robot system with test objects of different shape and presented the results of the grasp planner based on the exploration data. Finally, we tested the best grasp candidate by executing the grasp within the physics simulation. In further experiments we have reported results for different object locations and orientations in the manipulator workspace.

For the future we are working on an extension of the presented set of geometric filters in order to further improve the success rate upon grasp execution with our robot hand. Further we will consider the incorporation of the palm during grasp execution, which would enable power grasps.

Concluding, we are confident that the dynamic potential field based approach presented may be used for real world tactile exploration and grasping with an anthropomorphic robot hand, as it appears robust enough to autonomously control interaction of the robot hand with a previously unknown object using tactile information. We assume that the proposed scheme is transferable to different manipulator and robot hand kinematics by adapting filter parameters, number of RCPs and RCP locations. We further plan to investigate possibilities of combination with exploration methods based on sensors of different modalities than haptics, e.g. vision based object exploration. The developed control scheme based on VMC and dynamic potential fields is currently subject to implementation on our real world robot system equipped with five-finger hands [22].

Acknowledgement

The work described in this paper was conducted within the EU Cognitive Systems projects PACO-PLUS (FP6-027657) and GRASP (FP7-215821) funded by the European Commission.

References

[1] J. Pertin-Troccaz, "Grasping: A state of the art," in *The Robotics Review*, O. Khatib, J. J. Craig, and T. Lozano-Perez, Eds. The MIT Press, 1989, vol. 1.

[2] A. Bicchi and V. Kumar, "Robotic grasping and contact: a review," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 1, 24-28 April 2000, pp. 348–353vol.1.

[3] M. Teichmann and B. Mishra, "Reactive algorithms for 2 and 3 finger grasping," in *IEEE/RSJ International Workshop on Intelligent Robots and Systems, Grenoble, France*, 1994.

[4] J. Coelho and R. Grupen, "A control basis for learning multifingered grasps," *Journal of Robotic Systems*, vol. 14, no. 7, pp. 545–557, 1997.

[5] J. Platt, R., A. Fagg, and R. Grupen, "Nullspace composition of control laws for grasping," in *Intelligent Robots and System, 2002. IEEE/RSJ International Conference on*, vol. 2, 30 Sept.-5 Oct. 2002, pp. 1717–1723vol.2.

[6] D. Wang, B. T. Watson, and A. H. Fagg, "A switching control approach to haptic exploration for quality grasps," in *Robotic Science and Systems*, 2007.

[7] R. Platt, "Learning grasp strategies composed of contact relative motions," in *IEEE-RAS International Conference on Humanoid Robots, Pittsburgh, PA*, Dec 2007.

[8] J. Steffen, R. Haschke, and H. Ritter, "Experience-based and tactile-driven dynamic grasp control," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, Oct 2007, pp. 2938–2943.

[9] B. Wang, L. Jiang, J. LI, and H. Cai, "Grasping unknown objects based on 3d model reconstruction," in *Proc. IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2005, pp. 461–466.

[10] K. Roberts, "Robot active touch exploration: constraints and strategies," in *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, 13-18 May 1990, pp. 980–985 vol.2.

[11] S. Caselli, C. Magnanini, F. Zanichelli, and E. Caraffi, "Efficient exploration and recognition of convex objects based on haptic perception," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, 22-28 April 1996, pp. 3508 – 3513 vol.4.

[12] M. Moll and M. A. Erdmann, *Reconstructing the Shape and Motion of Unknown Objects with Active Tactile Sensors*, ser. Springer Tracts in Advanced Robotics. Springer Verlag Berlin/Heidelberg, 2003, ch. 17, pp. 293–310.

[13] A. Bierbaum, M. Rambow, T. Asfour, and R. Dillmann, "A potential field approach to dexterous tactile exploration," in *International Conference on Humanoid Robots 2008, Daejeon, Korea*, 2008.

[14] J. Pertin-Troccaz, "Geometric reasoning for grasping: a computational point of view," in *CAD Based Programming for Sensory Robots*, ser. NATO ASI Series, B. Ravani, Ed. Springer Verlag, 1988, vol. 50, iSBN 3-540-50415-X.

[15] A. T. Miller and P. K. Allen, "Graspit!: A versatile simulator for grasp analysis," in *Proceedings ASME International Mechanical Engineering Congress & Exposition, Orlando*, Nov. 2000, pp. 1251–1258.

[16] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[17] J.-O. Kim and P. Khosla, "Real-time obstacle avoidance using harmonic potential functions," in *Proc. IEEE International Conference on Robotics and Automation*, 1991, pp. 790–796 vol.1.

[18] I. Gaiser, S. Schulz, A. Kargov, H. Klosek, A. Bierbaum, C. Pylatiuk, R. Oberle, T. Werner, T. Asfour, G. Bretthauer, and R. Dillmann, "A new anthropomorphic robotic hand," in *IEEE-RAS International Conference on Humanoid Robots*, 2008.

[19] J. Pratt, A. Torres, P. Dilworth, and G. Pratt, "Virtual actuator control," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems '96, IROS 96*, vol. 3, 1996, pp. 1219–1226 vol.3.

[20] A. Bierbaum, T. Asfour, and R. Dillmann, "Ipsa - inventor physics modeling api for dynamics simulation in manipulation," in *IROS - Workshop on Robot Simulation, 22. Sept. 2008, Nice, France*, 2008.

[21] N. Amenta, S. Choi, and R. Kolluri, "The power crust," in *Sixth ACM Symposium on Solid Modeling and Applications*, 2001, pp. 249–260.

[22] T. Asfour, K. Regenstein, P. Azad, J. Schroder, A. Bierbaum, N. Vahrenkamp, and R. Dillmann, "Armar-III: An integrated humanoid platform for sensory-motor control," in *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, Dec. 2006, pp. 169–175.

Robotics Group
The Maersk Mc-Kinney Moller Institute
University of Southern Denmark

Technical Report no. 2010 – 1

Initial Investigation of Local and Global Grasp Transferability Utilizing Visual Feature Relations

Leon Bodenhagen, Emre Başeski, Dirk Kraft, Justus Piater, Norbert Krüger

January 28, 2010

Title	Initial Investigation of Local and Global Grasp Transferability Utilizing Visual Feature Relations
	Copyright © 2010 Leon Bodenhagen, Emre Başeski, Dirk Kraft, Justus Piater, Norbert Krüger. All rights reserved.
Author(s)	Leon Bodenhagen, Emre Başeski, Dirk Kraft, Justus Piater, Norbert Krüger
Publication History	

1 Introduction

The aim of this article is to investigate how successful grasps can be synthesized based on previous grasping experiences and what kind of feature relations are decisive for determining if a grasp will be successful or not. The general goal is to form a basis which allows us to replace a random exploration procedure used for learning object specific grasp densities [1] with an empirically grounded selection process, which is likely to lead to a significant speed up of the grasp density learning process. We subdivide this problem by first looking at which features trigger grasps that are likely to be successful. We then explore the effect of forming a new grasp based on transforming a previously successful grasp on a local and global scale.

To generate grasping hypotheses in vision based robotics, a general approach is to use visual feature constellations that are likely to trigger successful grasps (see, e.g., [2, 3]). In this work, we used the approach introduced in [4] as a basis and we evaluated the possibility of generating grasping hypotheses from previously tried grasps and geometrical properties of visual features they are associated to. Similar to our approach, grasp prototypes are used to search for good grasps in [5]. Also in [6], a demonstrated grasp is used to grasp similar objects.

1.1 Problem Statement

In this paper we want to address two questions that are closely related:

Q1 What visual feature relations predict if a grasp is likely to be successful?

Q2 How can an existing successful grasp be transformed into a new successful grasp by using a space of visual features and the relations defined on it.

The two problems are closely related since knowledge about feature-relations and associated grasp success likelihoods (**Q1**) can be used in an active exploration process, which uses previously gained knowledge about concrete grasping experiences.

In mathematical terms the two tasks can be formulated as probability density estimation problems:

$$\begin{array}{ll} \mathbf{Q1} & P\left(\text{Grasp}(F_1, F_2) \text{ is successful} \,|\, \text{Relation}(F_1, F_2) \in I^n\right) \\ \mathbf{Q2} & P\left((T(F_1, F_2) * A_1) \text{ is successful} \,|\, \text{Relation}(F_1, F_2) \in I^n \wedge A_1 \text{ is successful}\right) \end{array}$$

where F_1 and F_2 represent two visual features, Grasp$(*, *)$ is a function that computes a grasping hypothesis based on two features, Relation$(*, *)$ is an n-dimensional function that computes a set of relations between two features, I^n is an n-dimensional interval. For **Q2** this set of prerequisites needs to be extended by A_1 which represents a grasp connected to F_1 ($A_1 = \text{Grasp}(F_1)$) and $T(F_1, F_2) : \mathbb{SE}(3) \mapsto \mathbb{SE}(3)$ is a transformation that transforms one 3D pose into another, where the concrete transformation is dependent on F_1 and F_2.

To be more precise, we are interested in the impact of the transformation T on the grasp success. This transformation can be seen from a local point of view where we are interested how small pose changes effect the success likelihood or from a more global perspective where we transform a grasp according to the transformation between two features (denoted F_1, F_2 up till now).

Note that since not much can be said about the densities inherent in **Q1** and **Q2** a priori, non-parametrized methods (i.e. kernel density estimation (KDE)) are a suitable method for approximation them.

The complexity of problem **Q2** arises from the fact that there is a six-dimensional space (a pose transformation in $\mathbb{SE}(3)$) that defines how to associate a grasp to a contour. In addition there is also an up to six–dimensional space of pose transformations between the two visual features contours. The actual dimensionality depends on the concrete nature of the features but we can use $\mathbb{SE}(3)$ here as an upper bound. Note that F_1 does not need to be modeled since it is assumed to be placed in the origin in some canonical way. Estimating a twelve dimensional density is hard and requires a large amount of samples. Therefore, we need to reduce the problem leading to a number of sub-problems. But we first express **Q1** more precisely.

The questions **Q1** and **Q2** can be described in more detail as follows. **Q2** will be subdivided into **Q2.1** and **Q2.2** to reduce it to feasible subproblems.

Q1 **Computing the predictivity of specific relations for successful grasps:** This problem is addressable in a relatively straightforward way. Given two contours F_1 and F_2, we need to define a function that defines a concrete grasp $\mathrm{Grasp}(F_1, F_2)$ based on the two contours. For the work here, we will limit the used relations to coplanarity and co-colority since these have been used previously in [4] to trigger grasps, and it is expected that coplanar and co-color contours lead to grasping hypotheses which are likely to be successful.

$$P(\mathrm{Grasp}(F_1, F_2) \text{ is successful} \,|\, \mathrm{coplanarity}(F_1, F_2) \in I_p \wedge \text{co-colority}(F_1, F_2) \in I_c),$$

hence we need to approximate a function depending on coplanarity and co-colority.

Q2.1 **Estimating the influence of local position changes:** As a first step we investigate the space that defines how the grasp gets associated to a contour by varying an existing grasp in a small neighborhood.

$$P((T * A_1) \text{ is successful} \,|\, A_1 \text{ is successful})$$

Note that no relations are taken into account in this step and the pose is changed in a rather limited local area.

Q2.2 **Computing the global impact of grasping experiences depending on concrete relations:** In this case, we simplify **Q2** by only looking at two meaningful relations. Hence we look at

$$P((T * T'(F_1, F_2) * A_1) \text{ is successful} \,|\, \mathrm{Relation}(F_1, F_2) \in I \wedge A_1 \text{ is successful}).$$

In contrast to **Q2.1**, we need to define an explicit prediction how a grasp at a contour F_1 (A_1) can be transferred in some canonical way by knowing the transformation between the contours F_1 and F_2, which is expressed as $T'(F_1, F_2)$ here. Note that another transformation T is used in this context to represent an additional pose offset as in **Q2.1**.

1.2 Organization

The rest of the article is organized as follows. In Section 2, the visual representation and the relational space that we use to trigger grasping actions are briefly introduced. The discussion about how tried grasps for specific objects were obtained and adapted for our use (Section 3) is followed by a description of how grasping hypotheses are calculated based on visual features and how they are transformed (Section 5). After presenting the experimental results in Section 6 we conclude with a brief discussion in Section 7.

2 Visual Representation

In this work we make use of a visual representation based on local descriptors called *primitives* [7]. They are extracted sparsely along image edges and form a feature vector that contains visual modalities such as position, orientation, phase and color ($\pi = (\mathbf{x}, \theta, \phi, (\mathbf{c_l}, \mathbf{c_m}, \mathbf{c_r}))$ where color of a patch is defined by left, right and middle color. 2D-primitives are matched across two stereo views and pairs of corresponding primitives afford the reconstruction of a 3-dimensional equivalent called 3D-primitive which is encoded by the vector $\Pi = (\mathbf{X}, \Theta, \Phi, (\mathbf{C_l}, \mathbf{C_m}, \mathbf{C_r}))$. An overview of the visual representation and how information is represented in each level is shown in Figure 1.

The 3D primitives are linked according to geometrical and visual good continuation, as described in [8], to create more global entities that we call *contours*. These contours are described analytically as NURBS (Non-uniform Rational B-Splines) [9] approximations of the curves. In the rest of this work, we use C_i to denote a 3D contour and $C_i(t)$ to denote the NURBS approximation of C_i.

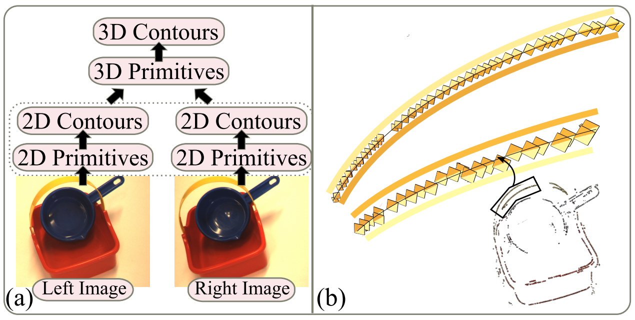


Figure 1: An overview of the visual representation. **(a)** The hierarchy of the visual representation and how information is represented in each level. **(b)** A closer view at 3D primitives and 3D contours.

Similar to the 3D primitives they are based on, contours also have modalities like position, color and orientation. Since primitives have left and right colors, a contour has mean left and right colors as well (see Figure 1(b)). To further characterize a 3D contour, we fit a line by taking the position uncertainties of the 3D primitives into account (see [10] for details of how to use position uncertainties for this process). We define the position of a contour (x_i) as the projection of its centroid on to this line, and its orientation (C_i^v) as the orientation of the fitted line. Note that, these attributes make it possible to define semantic relations such as co-planarity and distance between contours.

Within the context of this work, we use five inter-contour relations, namely, coplanarity, co-colority, normal distance, angle and collinearity. Coplanarity between two contours C_i and C_j ($R^P(C_i, C_j)$) measures the mean angle between the fitted lines of the contours and their common plane. The distance between the centroid of one contour to the fitted line of the other gives a parallel distance, which we define as the normal distance ($R^D(C_i, C_j)$). We also use the fitted lines of the contours to define the angle between them ($R^A(C_i, C_j)$) and to measure how collinear ($R^L(C_i, C_j)$) they are.

Note that all these relations are defined in 3D and they are geometrical relations. On the other hand, co-colority is an appearance based relation and it measures the color difference between the sides of two contours that are facing each other. The formal definitions of the geometrical relations are:

$$
\begin{aligned}
R^P(C_i, C_j) &= \frac{\pi - \alpha_i - \alpha_j}{2}, \quad \alpha_i = acos(\frac{n \cdot C_i^v}{|n||C_i^v|}), && (1)\\
R^D(C_i, C_j) &= \frac{|\mathbf{w}_i - (\mathbf{w}_i \cdot \mathbf{u}_i)\mathbf{u}_i| + |\mathbf{w}_j - (\mathbf{w}_j \cdot \mathbf{u}_j)\mathbf{u}_j|}{2}, && (2)\\
&\quad \mathbf{w}_i = x_j - x_i, \quad \mathbf{u}_i = C_i^v, \\
R^A(C_i, C_j) &= acos\left(\frac{C_i^v \cdot C_j^v}{|C_i^v||C_j^v|}\right), && (3)\\
R^L(C_i, C_j) &= 1 - sin\left(\frac{\beta_i + \beta_j}{2}\right), && (4)\\
&\quad \beta_i = acos(\gamma \cdot C_i^v), \quad \gamma = (x_j - x_i).
\end{aligned}
$$

3 Grasping Data

Grasps are in this work considered to be grasps with a two-finger gripper and are defined by a 6D pose consisting of a 3D location $w \in \mathbb{R}^3$ and a rotation quaternion $q \in \mathbb{H}$.

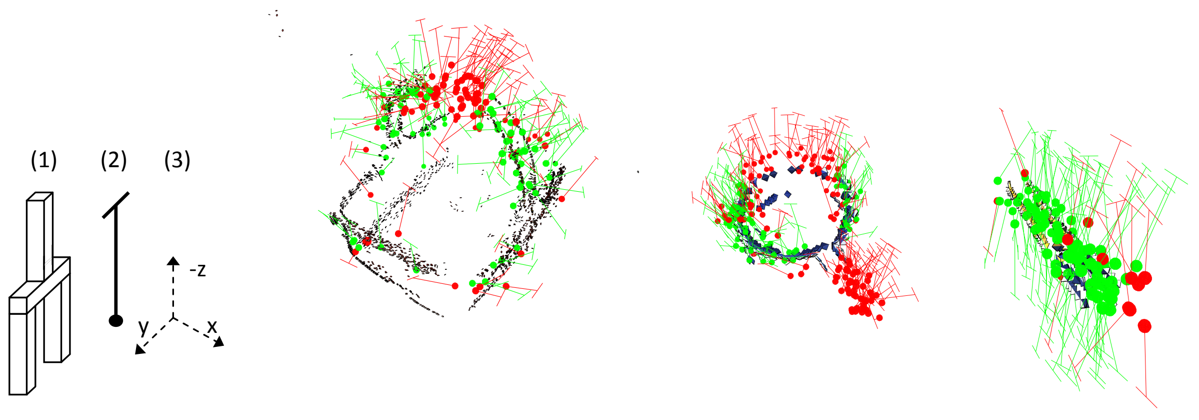
$$a = \{w, q\} \text{ for } w \in \mathbb{R}^3, q \in \mathbb{H} \qquad (5)$$

In this work, we use accumulated visual representations of objects [11] and associate tried grasps to this objects to gain knowledge on how these objects can be grasped successfully. For three objects, an accumulated model as well as a set of evaluated and successful grasps are presented in Figure 2(b–d). The pose of each grasp is given in the reference frame of the individual object model. However, the grasps are not directly linked to specific visual features of the object model as they are created using the representation of a unique scene. The object model has been used to estimate the pose of the object in the current scene and subsequently to transform the evaluated actions to the reference frame of the model (see [1] for further details how the grasps have been recorded). Therefore we determine which contour of the model is closest to a given grasp in terms of Euclidean distance and create hereby a set of pairs, each consisting of a contour and a grasp. Grasps that are not close to any contour of the object model are discarded.

Note that these object models along with recorded grasps are suitable for our investigations as they both offer dense visual representations and provide a large number of tried grasps at various locations at the object.

4 Kernel Density Estimation

To analyze the experimental results, we have used multi-dimensional histograms. Conventional histogram methods have inherent problems since they heavily depend on the bin size and location. For example, if data is mostly accumulated close to bin borders, small deviation in measurements will cause significant change in the shape of the histogram. To avoid this problem, we choose to use histograms where data can effect more than one bin. The procedure is illustrated in Figure 3.



(a) Basket (b) Basket (c) Pan (d) Knife

Figure 2: Recorded grasps for three different objects. **(a)** Figure showing the relationship between (1) a two finger grasp and (2) the grasp representation used in the rest of this figure and (3) axes labels. **(b–d)** Green grasps have been associated to contours, red ones have been discarded as no contour was sufficiently close.

When an occurrence of 8 is inserted into the conventional histogram in Figure 3(a), it increments the second bin by 1. If we want each data instance to effect two bins, we can distribute the effect to the neighboring bins depending on the value of the data instance (see Figure 3(b)). The resultant histogram is shown in Figure 3(c). Note that, the histogram in Figure 3(c) is a smoother histogram than the histogram in Figure 3(a) since it distributes the effect of a vote into more than one bin.

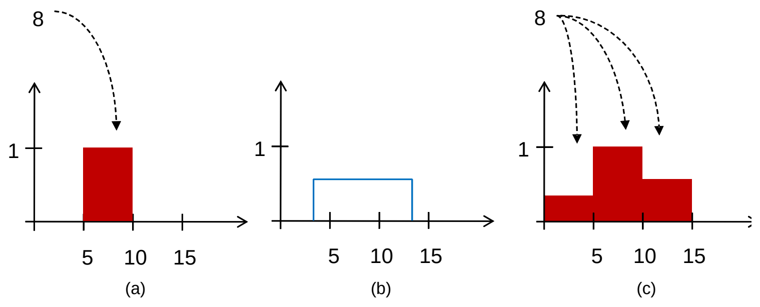


Figure 3: Smoothing a histogram by distributing the effect of a vote into more than one bin. **(a)** A conventional histogram example. **(b)** The coverage of a data instance if we let a data instance have a width of two bins. **(c)** The resultant histogram where a vote has a width of two bins.

5 Feature-Relation Grasp Statistics

To answer **Q1**, we need to define grasping actions, based on visual features. In Section 5.1, we discuss a feature-based grasp generation method, which is based on visual 3D contours. In the context of **Q1**, we present a method to predict grasps based on a tried action and two 3D contours in Section 5.2, and in Section 5.3 we discuss a method to evaluate the quality of a predicted grasp based on previously tried grasping data.

5.1 Feature-Based Grasp Generation

As mentioned in Section 3, grasps for a two-finger gripper (see Figure 4(a)) are defined through a 3D location (w) and a rotation quaternion (q). A grasp at a certain position $C_i(t_0)$ on a contour C_i is defined by using s second contour C_j:

$$\begin{aligned} w &= C_i(t_0) \\ r_x &= C_i'(t_0) \\ r_z &= n \\ r_y &= r_z \times r_x \end{aligned} \tag{6}$$

where $C_i'(t_0)$ is the derivative of $C_i'(t_0)$ at t_0 and n is the normal on the common plane that is formed by C_i and C_j. r_x, r_y and r_z form together a coordinate system. q is the rotation quaternion that rotates from this coordinate system into base coordinates.

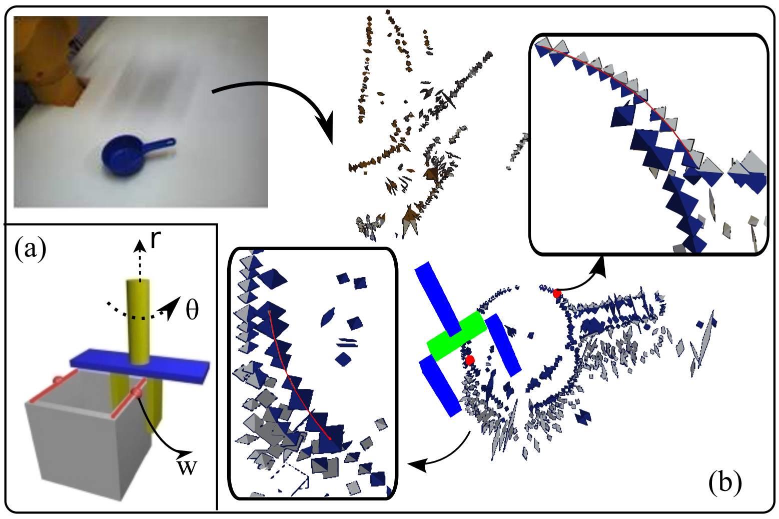


Figure 4: Grasping with a two-fingered gripper. **(a)** A grasp (action) is defined by a location and a orientation. **(b)** An example action.

5.2 Grasp Transformation

A grasp $\hat{a}_j$ is predicted by transforming a tried grasp a_i from its associated contour C_i to C_j. The new position is obtained directly from the contour C_j, additionally the displacement v_{disp} of a_i with respect to C_i is applied to the predicted action as well. The orientation of $\hat{a}_j$ is obtained directly from a_i. This leads to the following definition of $\hat{a}_j = \{w_j, q_j\}$ given $a_i = \{w_i, q_i\}$:

$$w_j = x_j + q_f \, v_{disp} \, q_f^* \tag{7}$$

$$q_j = q_f \, q_i \tag{8}$$

where q_f is the quaternion defining the rotation which aligns C_i and C_j.

Calculating the orientation from a_i may not be the best choice under all conditions, as the relation between the contours is not necessarily sufficient to define this orientation.

As we do not have any preference where on the contour the predicted grasp should be placed, we sample the contour (illustrated in Figure 5(a)).

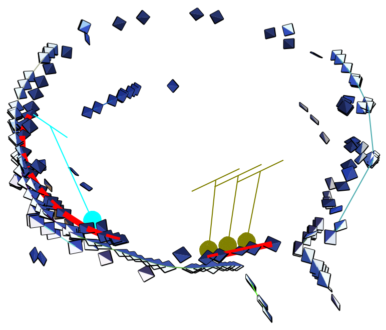


Figure 5: Illustration of the transfer of a grasp. The teal grasp to the left has been transferred to another contour which has been sampled. The resulting grasps are colored brown and the involved contours are highlighted in red.

In a secondary step, the position of a predicted grasp is varied both in the main direction of the contour and in the plane defined by the contours orientation and position. This is illustrated in Figure 9.

Finally the predicted grasps are varied in terms of position. The variation of the position is encoded with respect to the main direction of the associated contour as well as the plane perpendicular to the main direction.

Within the context of this work, the grasp transformation shown in this section is only used for **Q2.2**. In **Q2.1** a discretized local position displacement along the XYZ coordinates (see Figure 2(a)) is used. For **Q2.2** we apply such a position displacement as well, after the grasp transformation detailed in this section has been used.

5.3 Evaluation of Predicted Grasps

In order to evaluate the predictions, a quality measure is defined based on the Euclidean distance to the closest tried grasp and the angle of the rotation needed to align the prediction with a tried grasp. Thus when a predicted grasp matches a tried grasp, it is assumed that it would be successful. Each grasp is evaluated by comparing it with all tried grasps and using the one resulting in the highest estimate:

$$\begin{aligned} QL(\hat{a}) &= \max\left(1 - \operatorname{dist}(\hat{a}, a)\right) \forall a \in S && (9) \\ \operatorname{dist}(a_i, a_j) &= \min\left(1, (1-c)\,\frac{\min\left(\|w_i - w_j\|, 2 \cdot d\right)}{d} + c\frac{|\operatorname{angle}(q_i^* \cdot q_j)|}{\pi}\right) && (10) \end{aligned}$$

where $QL()$ is a function that measures the quality of a predicted grasp, S the set of all tried grasps, d is a distance threshold that encodes the maximum expected distance and c is a weighting constant whose value is chosen to 0.5 here.

The symmetry of the gripper is taken into account but omitted here. Further, the Euclidean distance is allowed to dominate the overall distance measure. This ensures that only nearby tried grasps have an impact. As it is likely that there always exist a tried grasp with a similar orientation as the predicted one (e.g. the one tried grasp used to create the prediction) the expressiveness of the evaluation would otherwise decrease.

6 Results

6.1 Impact of relations on assigned grasps.

In order to address the problem defined in **Q1** the grasping mechanism outlined in Section 5.1 is used to create grasping hypotheses for all combinations of contours. Subsequently, the quality of each grasping hypothesis for being successful is estimated as outlined in Section 5.3.

The likelihood of a grasping hypothesis for being successful and the value of specific relations between the contours which have been used to trigger a grasp are used as basis for the histograms shown in Figure 6.1. To improve the visualization, for each bin the number of grasps that have a higher likelihood for being successful than some threshold is computed and this amount is indicated by the color of the bin.

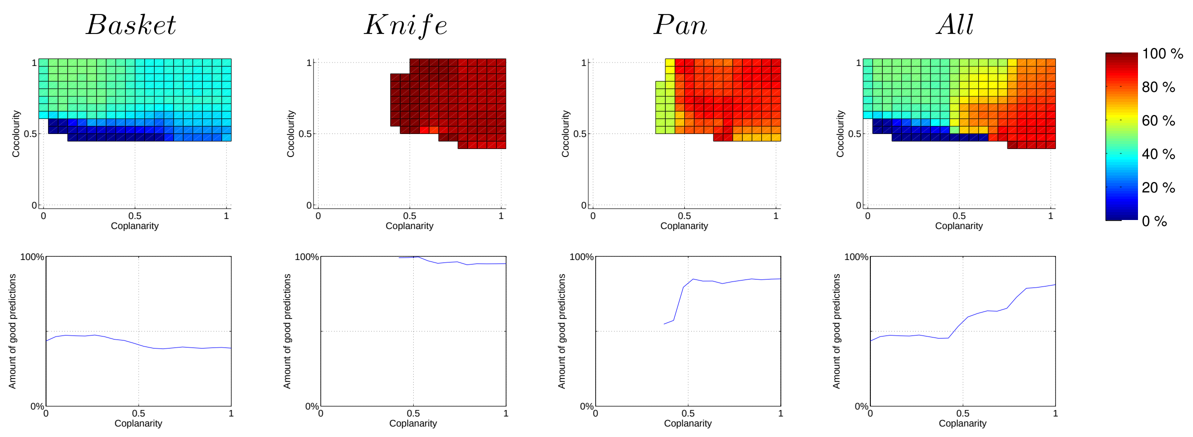


Figure 6: Results for **Q1**. Top row shows the amount of good predictions with respect to coplanarity and co-colority, bottom only for coplanarity. The fourth column is the sum over all objects, giving each object the same impact.

As mentioned earlier, the relations investigated were co-colority and coplanarity. The results indicate that the co-colority-relation is problematic when accumulated object representations are used (this becomes explicit for the the pan). Even though it is unicolored (blue), the co-colority values range from 0.5 to 1. We assume that this problem is caused by the fact that the color of the object and the color of the background are not distinguished properly. Furthermore both pan and knife are close to planar objects. Therefore, only one object, the basket, is providing pairs of non-coplanar contours. In this respect, it is difficult to judge the importance of the coplanarity relation, even though results indicate that coplanar contours are more likely to lead to successful grasps.

6.2 Local variation of tried grasps

In **Q2.1** we are investigating how much a tried grasp can be transformed locally. For this purpose the position of a tried grasps is varied with respect to the XYZ-axes (illustrated in Figure 7). Currently only translations have been considered. In the future, also variations of the orientation will be addressed. For each variation of the grasp, the likelihood for being successful is estimated. Subsequently the four-tuple consisting of the displacements and the success-likelihood are used as basis for a histogram. In order to reduce the dimensionality, the histogram is split into five histograms according to the displacement in the Z-direction (see Figure 8). The displacement ranges from -40 to 40 mm in the Z-axis, and from 0 to 40 mm in the X- and Y-axes due to symmetry in the two finger gripper.

Similar to the previous histograms, the color in each bin indicates how many percent of the grasps in this bin have a higher likelihood for being successful than a threshold.

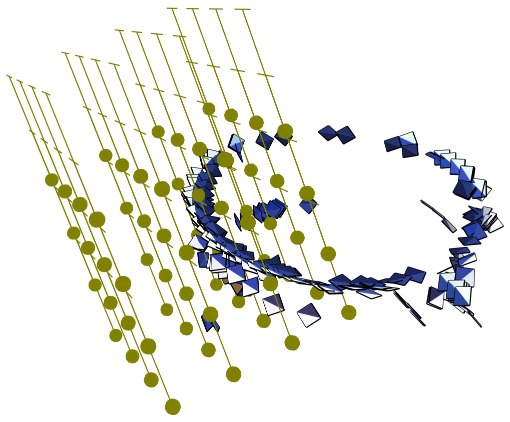


Figure 7: Illustrating variation of the positions in **Q2.1**. To increase visibility, the step-size in the position variation has been increased here.

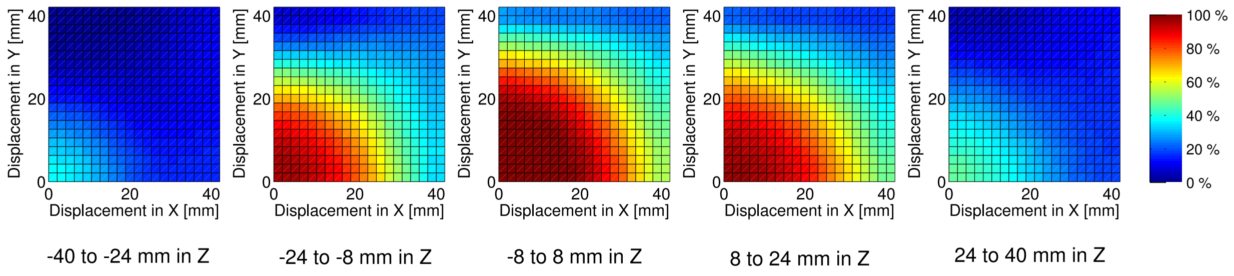


Figure 8: Results for **Q2.1**. The Displacement in the Z-axis varies from -40 to +40mm (left to right).

The results shown in Figure 8 for all three objects combined show that, grasps are more robust against displacements along their X-axis than their Y- and Z-axis. Yet, more significant results have been expected as the Z-axis is expected to be aligned (to some degree) with the contours. One possible reason for this is that a grasp still can be successful even though it is rotated slightly around it's approach vector, the Z-axis. Also, if grasps are rotated slightly, their axes are no longer aligned with the contour and therefore we cannot expect that grasps which have been displaced along their X-axis still have a high likelihood for being successful. When the orientation is varied as well, it will become explicit how much the orientation can be varied. The results that have been achieved so far indicate that the possible variance of the position resp. orientation should not be investigated independently.

6.3 Global transformation of grasping experience

We address problem **Q2.2** by transforming a grasp from one contour to another as outlined in Section 5.2. However, we cannot assume that the transformation we apply is suitable in every situation. Therefore the grasps are varied relative to the contour. Currently only the position is varied. In the future, also variation of the orientations will be addressed. The position is varied with respect to the main direction of the contour and the plane perpendicular to the main direction. This is illustrated in Figure 9.

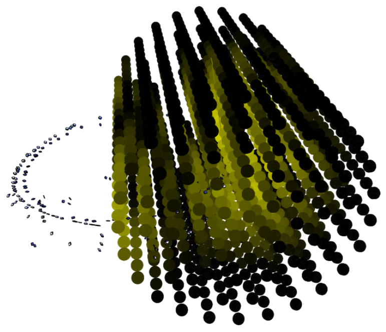


Figure 9: Illustration of the displacement of a grasp along the main direction of the contour and in the plane perperndicular to the main direction.

Each contour/action pair can be combined with every other contour to predict a new grasp. We then estimate the success likelihood for each of these new grasps. For each of these combinations, we use the relations between the contours, the predicted grasp success and the used position variation to form a common histogram. In order to decrease the dimensionality, the histogram is divided into several smaller ones as shown in Figure 10. The color of a bin again shows how many of the predictions falling into this bin have a likelihood for being successful higher than some threshold.

The two relations which have been investigated in this context are parallelism and normal distance. The latter is limited to the range from 0 to 250 mm which covers the dimensions of the largest object used. The results clearly indicate that grasps can not be easily transformed successfully when the contours have a high normal distance and that parallel contours indicate that grasps can be transferred. Still the results also indicate that more data is needed in order to cover the space spanned by the relations.

7 Discussion

We have shown that a tried grasp can be used to trigger another grasp by transforming it both locally and globally. When the transformation is done in a small neighborhood (local transformation), the transformation along the axis of the tool which is aligned with the main orientation of a contour is more successful than transformations in the orthogonal directions. We also showed that, global transformation of a tried grasp is more likely to be successful under certain inter-contour relations.

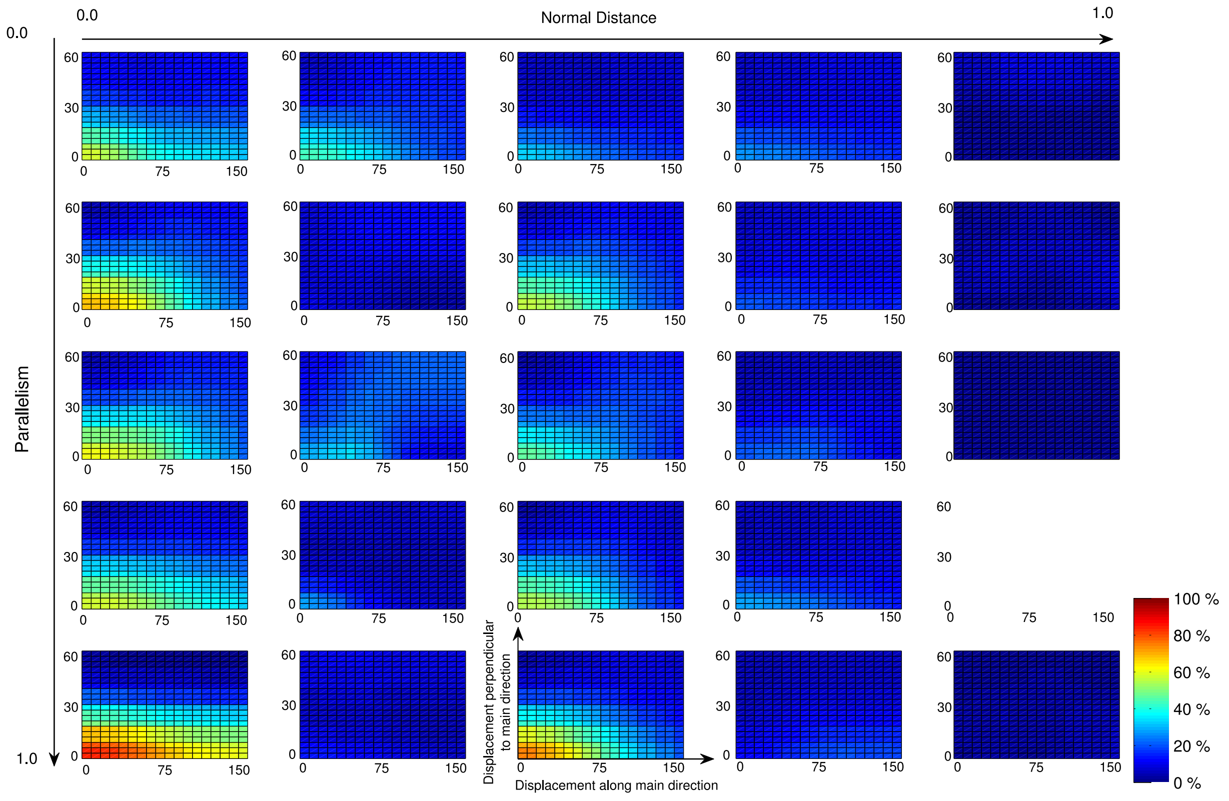


Figure 10: Results for transferred grasps that have been displaced along the main direction of the associated contour and in the plane perpendicular to the main direction. The results are shown with respect to parallelism and normal distance.

Acknowledgments

This work was supported by the EU Cognitive Systems project PACO-PLUS (IST-FP6-IP-027657).

References

[1] Renaud Detry, Dirk Kraft, Anders Glent Buch, Norbert Krüger, and Justus Piater. Refining grasp affordance models by experience. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2010. (accepted).

[2] A. Saxena, J. Driemeyer, and A. Y. Ng. Robotic grasping of novel objects using vision. *International Journal of Robotics Research (IJRR)*, 2008.

[3] Ishay Kamon, Tamar Flash, and Shimon Edelman. Learning to grasp using visual information. In *In Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 2470–2476, 1994.

[4] Mila Popovic, Dirk Kraft, Leon Bodenhagen, Emre Baseski, Nicolas Pugeault, Danica Kragic, and Norbert Krüger. A strategy for grasping unknown objects based on co-planarity and colour information. *RAS (accepted)*, 2010.

[5] N.S. Pollard. Synthesizing grasps from generalized prototypes. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 3, pages 2124–2130 vol.3, Apr 1996.

[6] K. Hsiao and T. Lozano-Perez. Imitation learning of whole-body grasps. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5657–5662, Oct. 2006.

[7] N. Krüger, M. Lappe, and F. Wörgötter. Biologically Motivated Multi-modal Processing of Visual Primitives. *The Interdisciplinary Journal of Artificial Intelligence and the Simulation of Behaviour*, 1(5):417–428, 2004.

[8] N. Pugeault, F. Wörgötter, and N. Krüger. Multi-modal Scene Reconstruction Using Perceptual Grouping Constraints. In *Proc. IEEE Workshop on Perceptual Organization in Computer Vision (in conjunction with CVPR'06)*, 2006.

[9] Les Piegl and Wayne Tiller. *The NURBS Book.* Springer-Verlag, London, UK, 1995.

[10] Donald R. Murray. *Patchlets: a method of interpreting correlation stereo three-dimensional data.* PhD thesis, 2004.

[11] N. Pugeault, F. Wörgötter, and N. Krüger. Accumulated Visual Representation for Cognitive Vision. *In Proceedings of the British Machine Vision Conference (BMVC)*, 2008.

Manuscript Number:

Title: Grounded concept and context learning: a connectionist approach

Article Type: Regular Article (12,000 words)

Keywords: knowledge representation; concept; context; connectionist modeling

Corresponding Author: Dr Saskia van Dantzig, Ph.D.

Corresponding Author's Institution: University of Leiden

First Author: Saskia van Dantzig, PhD

Order of Authors: Saskia van Dantzig, PhD; Antonino Raffone, PhD; Bernhard Hommel, professor

Abstract: Conceptual knowledge is acquired through recurrent experiences, by extracting statistical regularities at different levels of granularity. At a fine level, patterns of feature co-occurrence are categorized into concepts. At a coarser level, patterns of concept co-occurrence are categorized into contexts. We present and test CONCAT, a connectionist model that simultaneously learns to categorize concepts and contexts. The model contains two hierarchically organized CALM modules (Murre, Phaf, & Wolters; 1992). The first module, the Concept Module, categorizes concepts based on co-occurrences between features. These concepts are used as input for the second module, the Context Module, which categorizes contexts based on concept co-occurrences. Feedback connections from the Context Module to the Concept Module send activation from the active context to those concepts that frequently occur within this context. We demonstrate that context feedback contributes to the successful categorization of input patterns, especially when these patterns are degraded or ambiguous.

Grounded Concept and Context Learning: A Connectionist Approach

Saskia van Dantzig[1], Antonino Raffone[1,2], Bernhard Hommel[1]

[1]Leiden University (The Netherlands)

[2]Sapienza University of Rome (Italy)

Word count: 8754

Address correspondence to:

Saskia van Dantzig

Leiden University

Department of Psychology

Wassenaarseweg 52, Room 2B11

P.O. Box 9555

2300 RB Leiden, The Netherlands

phone: +31 (0)71-527 6667

Abstract

Conceptual knowledge is acquired through recurrent experiences, by extracting statistical regularities at different levels of granularity. At a fine level, patterns of feature co-occurrence are categorized into concepts. At a coarser level, patterns of concept co-occurrence are categorized into contexts. We present and test CONCAT, a connectionist model that simultaneously learns to categorize concepts and contexts. The model contains two hierarchically organized CALM modules (Murre, Phaf, & Wolters; 1992). The first module, the *Concept Module*, categorizes concepts based on co-occurrences between features. These concepts are used as input for the second module, the *Context Module*, which categorizes contexts based on concept co-occurrences. Feedback connections from the Context Module to the Concept Module send activation from the active context to those concepts that frequently occur within this context. We demonstrate that context feedback contributes to the successful categorization of input patterns, especially when these patterns are degraded or ambiguous.

Word count: 148

Grounded Concept and Context Learning: A Connectionist Approach

People are able to make sense of the world because they possess conceptual knowledge. Concepts are mental representations, referring to classes of objects or events in the world, which are organized in larger knowledge structures such as scripts and schemata. According to Murphy, "concepts are a kind of mental glue, in that they tie our past experiences to our present interactions with the world" (Murphy, 2004, p. 1). This quote refers to two important aspects of conceptual knowledge. First, we use concepts during our interactions with the environment. They enable us to categorize objects, to infer those properties of objects that cannot be perceived directly, and to react to objects in an appropriate way. For example, we might categorize a small round red object as a yew berry, infer that this is a poisonous type of berry, and therefore refrain from eating it. Concepts enable us to interpret and organize the continuous stream of perceptual information that enters our senses during our interactions with the world. Without concepts, the world would be experienced as "one great blooming, buzzing confusion" (James, 1890/1981, p. 464). The second aspect addressed by the quote is that conceptual knowledge is gained through experience. During repeated interactions with a particular class of objects (e.g., balls), the conceptual system extracts the statistical regularities of these experiences and stores them in memory to form a mental concept representing that class of objects (e.g., the concept BALL[1]).

Most theories of concept learning agree that concepts are formed by extracting the regularities that emerge from recurrent experiences (e.g., O'Connor, Cree & McRae, 2009; Rogers & McClelland, 2004; Smith, Shoben & Rips, 1974). During interactions with different exemplars of a particular category, the conceptual system detects which features are invariant across exemplars and are therefore relevant for the category. In order to create a mental concept representing the category, it stores those features in memory, while discarding the category-irrelevant (i.e. variant) features and information about the different contexts in which exemplars are encountered. These models thus assume that concepts are primarily based on bottom-up feature information, and they generally ignore the role of context information in conceptual processing (Yeh & Barsalou, 2006).

However, across many domains of cognition, it has been shown that the context profoundly influences conceptual processing and performance (for an overview, see Yeh & Barsalou, 2006). The term *context* has been defined in many different ways, but here we follow the definition of Yeh and Barsalou: a context is a region of perceived space, extending over some temporal duration. Studies of visual processing have shown that visual recognition of objects is influenced by context information (for an overview see Bar, 2004). Objects are recognized more accurately when they are seen in a typical context than when they are seen in isolation or in an atypical context. For example, Davenport and Potter (2004) showed participants photographs containing an object (e.g., a camel, truck, or football player) in front of a background setting (e.g., a beach, mountain, desert, or library). Objects and backgrounds were recognized better when they were consistent (e.g., a camel in front of a desert scene) than when they were inconsistent (e.g., a camel in front of a snowy mountain). Objects are also recognized better when they are presented together with a contextually related object than with an unrelated object. In a study by Davenport (2007), participants viewed and named two objects that were presented in front of a consistent or inconsistent background. The foreground objects were named more accurately when they were contextually related (e.g., an eskimo and an igloo) than when they were contextually unrelated (e.g., an eskimo and a camel). In addition, objects were named more accurately when they appeared in a consistent setting than in an inconsistent setting, similar to the findings of Davenport and Potter (2004). These effects are not limited to the visual domain. A recent study by Özcan and van Egmond

(2009) has shown that sounds of household products (e.g., an electric toothbrush) are recognized faster and more accurately when congruent visual context information (e.g., a picture of a bathroom) is presented than when incongruent information (e.g., a picture of a kitchen) is presented.

Similar effects occur in language processing. Words are recognized more accurately when they are presented in a meaningful sentence than when they are presented in an anomalous sentence (Schwanenflugel & Shoben, 1985). Furthermore, meaningful sentences can be read and understood even at very high presentation rates of 12 words per second (e.g., Potter, Kroll, Yachzel, Carpenter & Sherman, 1986). In contrast, when a sequence of random words is presented at that rate, participants can typically recognize and remember only 2 or 3 words from the sequence (Potter, 1993). Another study shows that semantically related words prime each other more strongly when people have read a sentence describing a congruent context relative to an incongruent context (Zeelenberg, Pecher, Shiffrin & Raaijmakers, 2003). For example, the word “sun” primes the word “beach” more strongly if participants have first read the sentence “he had a nice tan after a warm day on the beach” than if they have read the sentence “children like to play with scoops and buckets on the beach”, while the reverse is true for the prime “sand”.

Context also influences memory performance. In a famous study by Godden and Baddeley (1975), participants learned lists of words either underwater or on land, and subsequently recalled the words in the same context or in the alternative context. Memory was better if the context at recall was the same as the context at encoding. Other studies have shown that objects are remembered better when they appear in a coherent scene than when they appear in a scrambled scene (Mandler & Parker, 1976; Mandler & Stein, 1974). Furthermore, context does not only influence true memories, but false memories as well. For example, in a study by Miller and Gazzaniga (1998), participants viewed images depicting typical scenes (e.g., a beach). On a later memory task, they made more false recognition errors to items that were contextually congruent with the viewed scenes (e.g., a beach ball) than to scene-incongruent items (e.g., a snow scooter). Context effects have also been found in many conceptual tasks. For example, Barsalou (1982) demonstrated that mental concepts are flexible and that their exact content depends on situational factors such as current goals and task demands. In a property-verification task, participants were faster to verify contextually relevant properties than contextually irrelevant properties. Finally, context information can contribute to the categorization of new objects. A recent study by Chaigneau, Barsalou and Zamani (2009) showed that new objects were categorized more accurately when situational information was provided than when the objects were presented in isolation. In addition, inferences about the objects became more accurate as more situational information was presented during categorization.

Context information thus appears to be very relevant for conceptual processing and for mechanisms that rely on conceptual representations. To account for this influence, we propose that the conceptual system does not only form mental representations of concepts, but of contexts as well. These context representations can be learned in the same way as concept representations, by extracting the statistical regularities that emerge from repeated experiences. Regularities across experiences can be detected at various temporal and spatial scales. At a fine scale, patterns of co-occurrence between sensory-motor features (e.g., <red>, <round>, <soft>, <bouncing>) emerge, because features belonging to a particular object tend to be grouped in time and space. These patterns of feature co-occurrence can be used to form concept representations. At a coarser scale, patterns of co-occurrence between objects emerge. Objects are not distributed randomly in the world, but tend to co-occur in time and space with other objects, resulting in coherent scenes or contexts. For example, a toaster is typically found in a kitchen, and tends to co-occur with other objects that are often found in kitchens, such as ovens, stoves, sinks, pans and fridges. These patterns of co-occurrence between objects can be used to form mental representations of contexts. Thus, a context is defined as a coherent collection of objects that co-occur within a certain window of time and space. For example, a room is defined as a kitchen by the objects that are typically found in kitchens (e.g., a stove, sink, toaster, fridge, sideboard, etc.). Contexts can be defined at different grain sizes, and they can be hierarchically organized, in the sense that a higher-order context (e.g., a house) may be composed of multiple lower-order contexts (e.g., different rooms). As Yeh and Barsalou (2005) noted, "a situation can range from a large region of space over an extended period of time down to a small region of space for a brief moment. As a result, a given stimulus typically exists in a hierarchically organized set of situations across many levels of grain size simultaneously. There is not just one situation for a stimulus; typically there are many" (pp. 356-357). The idea that a context can be described as a pattern of co-occurring objects is not new. For example, Bar (2004) proposed the term 'context frame' as a representation of a prototypical scene, containing information about the objects that are most likely to appear in this scene. Likewise, Tversky, Zacks and Martin (2008) argued that scenes (contexts) are characterized by the activities that take place in them and the objects that occur in them. These objects are not distributed randomly across a scene, but their spatial arrangement is loosely constrained by the scene (see also Bar, 2004). For example, in a dining room, chairs are typically grouped around a table, a lamp is hanging above the table and food is placed on the table, rather than on the chairs. Although informative, this information is not used in our model. For the current purposes, a context is defined only as a pattern of co-occurrence between objects, disregarding information about their typical spatial arrangements.

In this paper, we distinguish only two levels of representation, concepts and contexts. Concepts are interpreted as object representations, and contexts are interpreted as collections of objects. Following the definition of concepts given above, however, both types of representations are actually concepts, in the sense that both refer to a category of objects or events in the world. Furthermore, other levels of representation could also be distinguished. For example, concepts can also represent features (e.g., RED), object parts (e.g., LEG), actions (e.g., THROW) and events (e.g., WEDDING). These concepts differ in the level of granularity, and each concept can encompass other concepts. For the sake of parsimony, here we distinguish only two levels and give them separate names, although we acknowledge that reality is much more complicated than this. Importantly, the model we suggest can easily be applied to object configurations, actions, and other events and be extended to capture hierarchical relations between these entities.

If contexts are represented as distributions of concepts, in a symmetrical way, concepts can be represented as distributions over contexts. Thus, concepts are based on two sources of information: feature-based data, and context-based or *distributional* data[2]. With respect to the former, we are basing our reasoning on the Theory of Event Coding (TEC: Hommel, Müsseler, Aschersleben & Prinz, 2001), which assumes that stimulus and action events are cognitively represented in terms of their distal features. Accordingly, we take feature-based information to refer to all the features (also called properties or attributes) of a particular category that are acquired through interactions with different exemplars of that category. This includes perceptual information from various sensory modalities, referring to the category’s typical shape, color, smell, taste, and feel, but also affective information and information about the actions afforded by (exemplars of) the given category. The similarity between two concepts is partially defined by the degree to which their features overlap. In addition to this experiential information, a concept is based on distributional information, which refers to the different contexts in which a concept is encountered (see also Barsalou & Wiemer-Hastings, 2005). Concepts may occur in a range of different contexts. The similarity between two concepts is therefore also defined by the degree to which they share the same contexts.

Summarizing, we claim that concepts relate to *both* various features and various contexts. Because of the distributed nature of such representations, features and contexts may be shared by multiple concepts. Any pair of concepts has a certain degree of feature overlap and context overlap, which defines how similar these two concepts are. In addition, concepts are not related to features and contexts in an all-or-none way, but in a more graded fashion. The strength of association with a particular concept varies across features. While some features are concept-defining features that are strongly related to the concept (e.g., <round> for BALL), others are more variable and are only weakly related to the concept (e.g., <red> for BALL). Reversely, features also vary in the degree to which they are distinctive for a particular category. Whereas some features (e.g., <can whinny>) are associated with only one category (HORSE) and are thus distinctive, others occur in many concepts (e.g., <made of metal>) and are therefore less distinctive. These differences in feature distributions may contribute to the organization of conceptual knowledge (McRae, de Sa & Seidenberg, 1997). In a parallel manner, concepts are connected to contexts in varying degrees (e.g., Bar & Aminoff, 2003). Some concepts are strongly related to a certain context (e.g., a *BEDROOM* typically contains a BED), whereas others are less strongly related to a certain context (e.g., a *BEDROOM* sometimes contains a TELEVISION). Conversely, some concepts are constrained to only a few contexts (e.g., a MICROWAVE is frequently found in a *KITCHEN* or an *APPLIANCE STORE*, but not in many other contexts) whereas other concepts (e.g., a BOOK) are distributed over many contexts.

Because concepts and contexts are connected to each other, they can influence each other bidirectionally. Activating a particular concept will lead to activation of the contexts that are associated with this concept. In turn, activating a context will prime concepts that are likely to occur within this context. As a result, those concepts can be recognized more easily. This explains why visual object recognition is better when an object is presented in a relevant context than in isolation or in an irrelevant context (Davenport & Potter, 2004). It also explains why the activation of a concept facilitates the recognition of contextually related concepts, as has been demonstrated in numerous semantic priming tasks (e.g., Meyer & Schvaneveldt, 1972; Neely, 1991).

In this article we propose that the conceptual system forms not only conceptual representations but also contextual representations. These representations are learned in a similar way, by extracting the statistical regularities over repeated experiences. Regularities can be detected at different temporal and spatial scales. At a small scale, features that co-occur in time and space are bound into concept representations. At a coarser scale, concepts that co-occur in time and space are bound into context representations. Conceptual and contextual representations are connected to each other, such that a particular concept will activate the contexts in which it frequently appears and reversely, a particular context will activate the concepts that are likely to occur within this context. Thus, concept representations are based on two sources of information; *experiential data*, bottom-up information about the (sensory, motor, affective, etc.) features that are relevant for interaction with a particular category, and *distributional data*, top-down information about the different contexts in which this category is likely to occur. These functional principles are here implemented in a connectionist model, named CONCAT, which simultaneously learns to categorize concepts and contexts.

The Model

Model Overview, Architecture and Functional Logic

At the core of this model are two Categorization and Learning Modules (CALM) (Murre et al., 1992; see also Murre, 1992). A CALM module is able to perform unsupervised categorization of distributed input patterns, in a localist fashion (see also Page, 2000). That is, a particular input pattern (or set of similar input patterns) is associated with a unique node in the module. CALM modules display two modes of learning, depending on the novelty of the input pattern (Graf & Mandler, 1984; Wolters, & Phaf, 1990). When a novel input pattern is presented, *elaboration learning* takes place. An increased learning rate combined with the distribution of nonspecific, random activations in the module results in strong competition between the uncommitted nodes (those nodes that do not yet represent a pattern). As a result, the input pattern is associated with a node that is not yet committed by any other pattern. When a familiar input pattern is presented, *activation learning* takes place. A low learning rate and weak competition between nodes result in the activation and strengthening of the existing representation. Multiple CALM modules can be combined to construct a modular neural network that is able to perform hierarchical categorization. CALM networks thus contain some psychologically and biologically plausible properties, such as modularity, unsupervised learning and novelty-dependent learning (Murre et al., 1992).

In CONCAT, depicted in Fig. 1, two CALM modules are implemented to perform concept-context based hierarchical categorization of input patterns. The first CALM module, the *Concept Module*, forms concept representations based on feature co-occurrences. These representations are used as input for the second CALM module, the *Context Module*, which categorizes contexts based on concept co-occurrences. The CONCAT model incorporates two activation streams; a bottom-up or feedforward stream and a top-down or feedback stream. The feedforward stream originates at the input layer, called *Feature Detection Layer*. In this layer, a perceived object is represented as a distributed pattern of activation across nodes representing different features from various sensory modalities, as well as affective and affordance features. We assume that some pre-processing of the raw sensory signal has already taken place, such that the Feature Detection Layer represents features in a subsymbolic but distal, rather abstract way (Hommel, Müsseler, Aschersleben & Prinz, 2001). The patterns of activation in the input layer are categorized by the Concept Module, by mapping different input patterns onto different nodes in the module.

Following categorization, information is propagated from the Concept Module, through several layers of internodes (see Fig. 2), to a next layer, called *Memory Buffer*. The Memory Buffer is similar to the Conceptual Short Term Memory (Potter, 1993), retaining the most recently activated concepts. As a result, a pattern of activation arises in the Memory Buffer, representing the co-occurrence of concepts across a stretch of time. This pattern is used as input for the next level, the Context Module. Importantly, stimuli do not appear in completely random order, but in coherent clusters of co-occurring patterns, which can be regarded as different contexts. As a result of this clustering, the activation patterns in the Memory Buffer show some regularities, which are picked up and categorized by the Context Module. In addition to this feedforward stream of information, feedback connections provide direct top-down input from the Context Module to the Concept Module. These feedback connections are modified through a form of novelty-dependent competitive Hebbian learning (for details, see below). Due to the feedback connections, concepts are effectively represented not only by bottom-up feature information, but also by top-down context information.

- FIGURE 1 ABOUT HERE -

- FIGURE 2 ABOUT HERE -

In CONCAT, concepts are thus represented as distributed patterns of activation across nodes representing features *and* as distributed patterns of activation across nodes representing contexts. Whenever a context node is activated, it propagates activation to those concepts that frequently occur within this context. As will be demonstrated by several simulations (see below), this top-down context information facilitates the categorization of concepts, especially when a given input pattern is ambiguous or degraded. In the case of an ambiguous input pattern, multiple concept nodes may potentially get activated, thus resulting in a competition between different concept nodes. Top-down activation from the active context increases the activation of the concept node representing the pattern that is most likely given the current context. As a result, this ‘expected’ concept will have a higher chance of winning the competition. In other words, a form of context priming takes place. A similar process takes place when the input is degraded. In this case, the bottom-up input may be too weak to activate the correct concept node to a sufficient degree to let it win the competition. The activation of the concept node can be enhanced by a top-down signal from the current context. In other words, contextual feedback can make concept nodes more sensitive to patterns belonging to the currently active context.

As will be demonstrated by the simulations, contextual feedback strongly improves concept categorization performance. This characteristic of the model explains why objects are more accurately recognized when they are presented within a relevant context than in isolation or in an irrelevant context, and reversely, why scenes are recognized more accurately when a relevant object is presented in the foreground (Davenport & Potter, 2004). It also explains how objects can facilitate the recognition of other objects that belong to the same context (Davenport, 2007).

Processes

As shown by the simulations reported below, the CONCAT model learns to categorize concepts distributed over a number of different contexts. To that end, input patterns were not presented randomly to the model, but they were grouped within clusters of contextually related patterns. As a metaphor, think of someone walking through a house and opening doors to different rooms in that house. When looking into a specific room, attention shifts from one object to the other, in order to recognize and categorize those objects as, for example, a faucet, a toaster, an oven, and a fridge. Once a sufficient number of objects has been recognized, the room (i.e., context) can be reliably categorized as a kitchen. Had one seen a faucet, shower, towels and a bath instead, the room would have been categorized as a bathroom. Thus, the collection of objects determines what kind of room (i.e. context) one is looking at. In order to simulate this procedure, several contexts are created. A context is defined as a collection of various input patterns. Because input patterns can occur in multiple different contexts (just like some types of objects are found in different rooms), contexts are partially overlapping.

In order to learn the different contexts, the model experiences several *contextual episodes* (c.f., opening the various doors in a house). During a contextual episode (see Fig. 3), a context is selected at random and all patterns belonging to the selected context are presented in random order. A pattern is presented on the Feature Detection Layer, and categorized by the Concept Module based on the bottom-up feature information. Activation is forwarded, through several layers of internodes, to the Memory Buffer. Note that activation spreads through the network at every iteration. However, due to the wiring of the internodes (explained in more detail below) activation does not reach the Memory Buffer until the Concept Module has reached convergence (i.e., the pattern has been categorized). In other words, the Memory Buffer only represents concepts that have been fully categorized by the

Concept Module. After a fixed number of iterations, the presentation of the input pattern is terminated. Before presenting a new input pattern from the same context, there is a delay during which the model does not receive any input pattern. This delay is implemented in order to let the activations of the Concept Module decay to zero. Since the Memory Buffer has a much slower decay rate than the Concept Module, the active concept is retained in the Memory Buffer, throughout the delay that resets the Concept Module. After the delay, the next pattern of the current context is presented and categorized in a similar way as the first pattern. Again, activation is forwarded to the Memory Buffer, where the previously activated node(s) is (are) still active. As a result, a pattern of activation arises in the Memory Buffer, reflecting the most recently categorized concepts. Once the amount of activation in the Memory Buffer reaches a certain level, the pattern of activation is categorized by the Context Module.

- FIGURE 3 ABOUT HERE -

Contexts are thus represented as constellations of concurrently active concepts, based on coincidence in time and place. The active context node sends feedback down to the Concept Module through modifiable feedback connections. The strength of these connections is adapted through Hebbian-like learning, such that connections get stronger when the sending context node and the receiving concept node are concurrently active, and they get weaker when one node is active while the other one is inactive. Once all patterns of a context have been presented, the contextual episode is finished with a long delay, in order to let all activations in the network decay to zero. Of course, in real life, episodes are not separated by a delay. Instead, information about the world enters our senses continuously. According to the Event Segmentation Theory (Zacks, 2007), people parse this continuous input into discrete events[3]. Simulations with a computational model of event segmentation have demonstrated that event boundaries can be reliably detected in a continuous stream of input (Reynolds, Zacks, & Braver, 2007). In CONCAT, this aspect was bypassed by presenting discrete episodes, separated by a delay. However, the work of Zacks and colleagues (Zacks, 2005; Reynolds, Zacks and Braver; 2007) shows that it is not unreasonable to assume that continuous input is parsed into discrete events, and that event boundaries can be used to reset the activations in the model.

Node Activation Rule

The input e_i to each network node *i* may be either excitatory (positive) or inhibitory (negative). The activation of each of a generic CONCAT node *i* at the iteration (t+1), $a_i(t+1)$, is described by the following rule:

$$a_i(t+1) = (1-k)\,a_i(t) + \frac{e_i}{1+e_i}\left[1-(1-k)a_i(t)\right] \qquad e_i \geq 0 \qquad \text{(1a)}$$

$$= (1-k)\,a_i(t) + \frac{e_i}{1+e_i}(1-k)\,a_i(t) \qquad e_i < 0 \qquad \text{(1b)}$$

where $$e_i = \sum_j w_{ij}\,a_j(t)$$

k is an activation decay term, w_{ij} denotes the connection weight from node *j* to node *i*. This activation rule ‘squashes’ the input to a number between 0 and 1, and ensures that the increase in activation due to input excitation, or the decrease in activation due to input inhibition, diminishes as the activation approaches the maximum or the minimum, respectively (asymptotic approach).

Internal Structure of Concept and Context Modules

The Concept and Context CALM Modules in CONCAT consist of different types of nodes that are connected to each other in the module by fixed weights (see Fig. 4 and Table 1 of the Appendix for details). ‘Representational-nodes’ (furthermore called R-Nodes) are the only nodes that receive external input. In order to create competition between R-nodes, each R-node activates a matched ‘Veto-node’ (V-node), which inhibits to all other nodes in the module. In particular, V-nodes are characterized by mutual inhibition; they strongly inhibit unmatched R-nodes, but they inhibit their matched R-Node only weakly. The modules also contain an ‘Arousal-node’ (A-node), which is sensitive to the novelty of the input pattern. In a sense, activation of the A-node can be regarded as a representation of the arousal response, which may result in an orientation reaction (e.g., Sokolov, 1963). Given the specific wiring pattern of the connections in a CALM module (see Fig. 4), the activation of the A-node is a positive function of the amount of competition in a module. Thus, the A-node activation measures the novelty of a given input pattern. Finally, an ‘External-node’ (E-node), which might correspond to an arousal center external to the module (e.g. a subcortical center in the brain), receives input from the A-node of the module, and sends random activation pulses to all R-nodes in the same module. Such random pulses are uniformly distributed over the range $[0, a_E(t)]$, where $a_E(t)$ stands for the activation of the E-node at the iteration *t*. The module structure and the weight values were drawn from Murre et al. (1992) (see Tables 1 through 3 of the Appendix for details on the values that were used).

- FIGURE 4 ABOUT HERE -

Learning rule

Full connectivity is assumed from node layers projecting onto the Concept Module (from the Feature Detection Layer and the Context Module) and the Context Module (from the Memory Buffer). The weights of such connections (‘inter-weights’) are modifiable in the range [0,K]. The learning rule used in CALM networks (Murre et al., 1992) is a variation of Grossberg’s (1976) learning rule, which is related to the Hebb-rule (Hebb, 1949), and is expressed by the following formula:

$$\Delta w_{ij} = \mu_t \, a_i \left[(K - w_{ij}(t))a_j - Lw_{ij}(t) \sum_{f \neq j} w_{if}(t) a_f \right] \tag{2}$$

where *i*, *j* and *f* are *R*-nodes, *L* and *K* are positive constants. The first term within the brackets represents weight increases, whereas the second term determines the background-dependent weight decrease. The symbol μ_t denotes the variable learning rate at iteration *t*, which is computed as follows:

$$\mu_t = d + w_{\mu E} \, a_E \tag{3}$$

where d is a constant and a_E is the activation of the E-node. To bind inter-weights in the range [0,*K*], the following condition is included:

$$w_{ij}(t+1) = \max\{\min[w_{ij}(t) + \Delta w_{ij}(t+1), K], 0\} \tag{4}$$

The learning parameters used in the simulations are specified in Tables 2 and 3 of the Appendix.

Connection Chain from the Concept Module to the Context Module

As shown in Fig. 2, a sequence of intermediate nodes (internodes) is placed between the Concept Module and the Memory Buffer, which in turn sends output to the Context Module. These internodes ensure that nodes in the Memory Buffer do not get activated until convergence has been reached in the Concept Module. In other words, the Memory Buffer only retains fully activated concepts. The R-nodes of the Concept Module send excitatory input to a matched ‘Excitatory-internode’ (E-internode) and to a matched ‘Inhibitory-internode’ (I-internode). In turn, the matched E-internode sends forward excitatory input to a matched ‘Feeding-internode’ (F-internode), whereas the matched Inhibitory-internode sends (forward) inhibition to all other unmatched E-internodes. Each F-internode, matched backward to a Concept R-node, sends activation to a matched node in the Memory Buffer. The excitatory input to the R-nodes of the Context Module includes input from individual Memory Buffer nodes, multiplied by modifiable inter-weights, and from a ‘Convergence node’ (C-node), the excitatory input of which is described by a sigmoidal function of the sum of the activations of all Memory Buffer nodes, as follows:

$$input_c(x) = \frac{1}{1+\exp(-\beta(x-\theta))} \qquad (5)$$

As the Context Module R-nodes receive a fixed amount of tonic inhibition, $Inht$, the input from the C-node unselectively enables activation of the R-nodes in the Context Module, which becomes selective with the differential modification of the inter-weights from Memory Buffer nodes to Context Nodes. This enabling mechanism plays a functional role to make activation of the Context Module dependent on the number of active nodes in the Memory Buffer.

Simulations

Simulation I: Basic performance of the CONCAT model

The first goal of Simulation I was to investigate whether CONCAT could learn to categorize concepts and contexts simultaneously. The neural network was presented with 16 different input patterns, which were clustered across four contexts. Half of the concepts were presented in one context only, the other half of the concepts appeared in two different contexts. As a result, each context contained six different concepts, and contexts were partially overlapping. The second aim of Simulation I was to investigate how the model's performance was influenced by the degree of overlap between the input patterns. The model's input was represented as a distributed binary pattern over 12 feature detection nodes. Two different sets of input patterns were created. In the *Low-overlap* condition, sparse patterns were used in which only two out of twelve nodes were 'on'. In the *High-overlap* condition, dense patterns were used in which, on average, six out of twelve nodes were 'on'. We expected that categorization performance would be better in the Low-overlap condition than in the High-overlap condition.

The simulation procedure consisted of 140 episodes. The first 80 episodes of the simulation were regarded as training episodes. The last 60 episodes were regarded as test episodes during which the performance of the model was assessed. It is important to note that learning was not disabled during the test phase. During each episode, a context was selected semi-randomly; we ensured that all contexts were presented equally often, both during the training stage and during the test stage. Once a context was selected, each of the patterns contained in that context were presented in a random order. Each pattern was presented for a fixed number of iterations.

Activation was propagated through the model as described above. The specific parameters of the simulations are given in Table 6 of the Appendix. During the test stage, the model's performance was assessed in the following way. After each pattern presentation, if the model had reached convergence, the winning concept node was registered. Similarly, at the end of each context presentation, the winning context node was recorded. After completion of the simulation, this information was aggregated into two different tables. One table displayed the number of times that each pattern converged on the different concept nodes (see Table 1 for an example). A similar table was created for the contexts, displaying the number of times that each context converged on the different context nodes. The node that most often represented a particular input pattern was labeled as the ‘winner’ for that pattern. In a symmetrical fashion, the pattern that was most often represented by a particular node was labeled as the winning pattern for that node. This information was used to compute two measures of performance. The *Convergence Index* was used as a measure how well a particular input pattern was categorized by a single node. It was defined as the number of times that the particular pattern was mapped onto its winning node, divided by the total number of presentations of that pattern. This resulted in a value between 0 and 1, where 0 meant that the input pattern never reached convergence onto any node, and 1 meant optimal convergence onto a single node. Orthogonally, the *Separation Index* was used as a measure of how well a particular node came to represent a single pattern. The Separation Index was computed as the number of times that a particular node was converged on by its winning pattern, divided by the total number of times that it reached convergence. This resulted in a value between 0 and 1, where 0 meant that the node did not represent any pattern, and 1 meant that the node only represented its winning pattern. In addition to these measures, convergence times were logged for each pattern presentation. Convergence time was defined as the number of iterations needed by the Concept Module to reach convergence.

The simulation procedure was replicated ten times for each of the two different input sets. This was done in order to get an indication of the variance in performance as a result of different random settings used in each of the replications. Convergence in the Concept

Module was influenced by the amount of overlap between the input patterns. In the Low-overlap condition, the Concept Module needed on average 11.3 iterations to reach convergence. In the High Overlap condition, an average of 16.6 iterations was needed to reach convergence. Table 2 shows the Convergence Index and Separation Index of the Concept Module and the Context Module. In both conditions of pattern overlap, the model performs quite well, resulting in high Convergence and Separation Indexes (all above 0.80). Although performance is better in the Low-overlap condition, the model still performs reasonably well in the High-overlap condition.

These results show that the model can learn to categorize concepts and contexts simultaneously. This is an important and novel finding. For example, previous simulations with a hierarchical CALM network have shown that it is possible to perform hierarchical categorization (Murre, Phaf, & Wolters, 1992; Murre, 1992). In those simulations, however, the model first learned the lower-level categories separately, before learning the higher-level categories. In contrast, for CONCAT it is not necessary to learn the concepts separately before learning them within their contexts. In other words, the current simulation demonstrates that it is possible to learn categories at both levels at the same time.

Simulation II: Influence of Context Feedback on Categorization of Ambiguous Patterns

The aim of Simulation II was to investigate the role of context-related feedback connections in categorizing ambiguous input patterns. In this simulation, the model learned to categorize two highly overlapping input patterns, among other, less overlapping, patterns. The highly overlapping patterns differed by only two nodes (out of a total of 12 nodes). During the training stage, the two overlapping patterns were presented in two different contexts. During the test stage, instead of the original patterns, an ambiguous pattern was presented which equally deviated from each of the two patterns (see Table 3 for an example). This input was presented to two different versions of the model. In the *Feedback-present* simulation condition, feedback input was allowed to propagate from the Context Module to the Concept Module. This feedback input could facilitate categorization of ambiguous input patterns in the following way: since the ambiguous pattern is equally similar to both original patterns, it will activate the two concept nodes representing those patterns to a similar degree. The concept node that is most likely within the given context will receive some extra top-down activation from the active context node, which increases the probability that this node will win the competition. As a result, the ambiguous pattern will be categorized as either one of the original patterns, depending on the context in which it is presented. In the *Feedback-absent* simulation condition, no feedback was propagated from the Context Module down to the Concept Module. Thus, context information could not be used to bias categorization towards the most likely concept.

In this simulation, each context consisted of four concepts. During the training stage, a contextual episode consisted of the random presentation of each of the patterns belonging to that concept in random order, as was the case in Simulation I. During the test stage, patterns were not presented completely randomly. Instead, the ambiguous pattern was always the last pattern to be presented within a contextual episode. This was done to ensure that the context representation was already active at the time of presentation of the ambiguous pattern, such that the categorization of that pattern could be biased by top-down information flowing from the activated context. The test phase was repeated with the original input patterns instead of the degraded patterns, in order to verify that the original patterns were categorized correctly. The model's performance in the 'ambiguous' test stage was compared to its performance in the 'original' test stage, which resulted in a measure of categorization accuracy. As in Simulation I, the whole procedure was replicated ten times with both versions of the model. The results of Simulation II are presented in Fig. 7, which displays the categorization of the ambiguous pattern. As becomes clear from the figure, context feedback influenced the categorization of the ambiguous pattern in a consistent manner. In the Feedback-present condition, categorization was strongly biased by the context. More specifically, the ambiguous pattern was most likely to be categorized as the concept that was congruent with the context in which it appeared. Thus, if the ambiguous pattern was presented in Context A, it was most likely to be categorized as Concept A. On the other hand, if the pattern was presented in Context B, it was most likely to be categorized as Concept B. No such bias existed in the Feedback-absent condition. In that case, the pattern was equally likely to be categorized as Concept A as Concept B, regardless of the context in which it was presented. Note that this simulation does not show that context feedback necessarily results in higher accuracy than when no feedback is present. For example, if a contextually incongruent pattern is presented, it may erroneously be categorized as the concept that is more likely in the current context. However, given that concepts most often occur in a congruent context, the context bias will usually lead to better performance.

Simulation 3; Influence of Context Feedback on Categorization of Degraded Patterns

The role of the feedback connections was further investigated in Simulation III. During the test phase of this simulation, the model received a number of degraded input patterns, and categorization performance on these patterns was assessed. One of the patterns from each context was degraded by reducing all units with a value of 1 and increasing all units with a value of 0 by a fixed value, corresponding to the severity of degradation. Ranging from mild to severe degradation, patterns were degraded by a value of 0.2, 0.25, 0.3, 0.35 and 0.4, respectively. Similar to the procedure of Simulation II, the input patterns were presented to two versions of the model. In the *Feedback-present* condition, feedback was propagated from the Context Module down to the Concept Module. In the *Feedback-absent* condition, no such feedback was propagated through the network. Since the bottom-up input from the Feature Detection Layer to the Concept Module is degraded, it may not activate the correct concept node to a sufficient degree to let it win the competition. Top-down activation from the active context node may facilitate the categorization, by enhancing the concept nodes that are most likely given the current context. Therefore, we expected that performance in the Feedback-present condition would be better than in the Feedback-absent condition.

The training phase was similar to that of Simulation II, except that there were no pairs of highly overlapping input patterns. As in Simulation II, each context included four patterns, such that patterns were non-overlapping. During the training stage, a contextual episode consisted of the random presentation of each of the patterns belonging to that context. During the test stage, however, the patterns were not presented randomly. Instead, the degraded pattern was always presented last within a contextual episode. This was done to ensure that the Context Module had already categorized the context at the time of presentation of the degraded pattern, such that the categorization of that pattern could be biased by top-down information flowing from the active context node.

The results of Simulation III are displayed in Fig. 6 and Fig. 7. Fig. 6 shows the categorization accuracy of the Feedback-present and the Feedback-absent versions under the different conditions of degradation. As predicted, categorization performance is better in the Feedback-present than in the Feedback-absent condition under all conditions of degradation. Fig. 7 shows the average number of iterations needed to reach convergence. Convergence times were faster in the Feedback-present version than in the Feedback-absent version. This finding corroborates the results of Simulation II, by demonstrating how context information can constrain concept categorization. This is especially useful when the bottom-up input is ambiguous or degraded and therefore provides insufficient information to enable successful categorization of the pattern.

Discussion

In this paper, we argue that the conceptual system forms mental representations of concepts *and* contexts. We assume that concepts and contexts can both be learned through experience, by extracting the statistical regularities that emerge during repeated experiences. Features that frequently co-occur in time and space are likely to belong to the same object. This information is used to form conceptual representations. At a coarser temporal and spatial scale, frequently co-occurring objects are likely to belong to the same context. This information can be used to form contextual representations. Thus, while concepts can be defined as patterns of co-occurring feature patterns, contexts can be defined as higher-order patterns of co-occurring concepts. We furthermore propose that concept representations and context representations are dynamically connected to each other. As a result, when a concept is activated, activation will spread towards the contexts in which it is likely to occur, and reversely, when a context is activated, activation will spread towards the concepts that are likely to occur in this context. As a result, cognitive processing can be tailored to the current situation.

To test these ideas, they were implemented in a connectionist model, named CONCAT. The model contains two hierarchically organized modules that simultaneously learn to categorize concepts and contexts by extracting regular patterns from experience at different spatial and temporal scales. Input to the model is represented as a distributed pattern of activation across nodes representing the (sensory-motor) features of a particular category. This input is categorized by the Concept Module, which forms concept representations based on patterns of co-occurrence across features. The concept representations formed by the Concept Module are used as input for the higher level Context Module, which forms context representations based on patterns of co-occurrence across concepts. In addition to this bottom-up stream of activation, top-down activation is propagated from the Context Module to the Concept Module trough feedback connections, which bind a context to its constituent concepts. As a result, conceptual knowledge in CONCAT is based on two sources of information: *feature based data*, bottom-up information about the (sensory, motor, affective, etc.) features that are relevant for interaction with a particular category, and *distributional data,* top-down information about the different contexts in which this category is likely to occur. This sets our model apart from most other theories of conceptual processing. Whereas they agree that mental concepts are learned by distilling invariant information from repeated experiences with exemplars of a particular category, they typically assume that context information is discarded during this process of distillation (Barsalou, 2005). However, the results of our simulations show that context information can play an important role in guiding and constraining conceptual processing. Our findings are corroborated by many studies across different domains of cognitive science (perception, language comprehension, memory, concept representation), demonstrating a variety of context effects.

In CONCAT, concepts and context representations are connected bidirectionally. Activating a particular concept will lead to activation of the contexts in which this concept is likely to occur, and reversely, activating a context will lead to activation of the concepts that are associated to it (see also Bar, 2004). As we have shown in Simulations II and III, these bidirectional connections are beneficial for conceptual processing. Top-down context information can be used to bias encoding of the input. This is especially useful when the bottom-up feature input is degraded or ambiguous. Indeed, Yeh and Barsalou argued; “Because specific entities and events tend to occur in some situations more than others, capitalizing on these correlations constrains and thereby facilitates processing. Rather than having to search through everything in memory across all situations, the cognitive system focuses on the knowledge and skills relevant in the current situation. Knowing the current situation constrains the entities and events likely to occur. Conversely, knowing the current entities and events constrains the situation likely to be unfolding” (Yeh & Barsalou, 2006, p. 350).

The idea that the conceptual system is composed of multiple, hierarchically organized, categorizing modules bears a similarity to Damasio’s theory of Convergence Divergence Zones (Damasio, 1989; Meyer & Damasio, 2009). According to this theory, Convergence Divergence Zones (CDZs) function as association areas that register the temporal co-occurrences between patterns of activation in lower level areas. During interaction with an object, patterns of activation arise across different sensorimotor areas, coding for various aspects of the interaction. The temporally coincident activity at separate sites is bound by a CDZ. In order to recall a particular event, the CDZ (partially) reconstructs the original pattern of activation in the sensorimotor sites by sending activation through divergent back projections. Damasio argues that multiple convergence zones can be organized in a hierarchical manner, each representing input at different levels of abstractness: “There are convergence zones of different orders; for example, those that bind features into entities, and those that bind entities into events or sets of events, but all register combinations of components in terms of coincidence or sequence, in space and time.” (Damasio, 1989, p. 26). Although CONCAT currently encompasses only two levels, we assume that it is possible to use the same logic to build a network composed of multiple layers that categorize patterns occurring at various spatial and temporal scales. In addition, we assume that it is possible to implement feedback connections between those layers, similar to the divergent pathways proposed by the CDZ theory.

Another parallel can be drawn to a recent theory of word meaning, proposed by Andrews, Vinson and Vigliocco (2009). They argue that word meaning is based on two sources of information. The first source is information about the sensory-motor properties of the word’s referent, as has been proposed in many theories of word meaning (e.g., Collins & Quillian, 1969; Rips, Shoben, & Smith, 1973; McRae et al., 1997). The second source is information about the lexical contexts in which a word occurs, as proposed by theories such as Latent Semantic Analysis (Landauer & Dumais, 1997) and Hyperspace Analogue to Language (Lund & Burgess, 1996). The model proposed by Andrews, Vigliocco and Vinson (2009) integrates both sources of information (for similar work, see Steyvers, in press). The data generated by the combined model corresponded better to human data of semantic representations than data generated by the separate sources. Thus, this study showed that experiential and distributional information can be combined to yield rich and realistic representations, similar to the findings of the present paper.

Andrews, Vigliocco and Vinson (2009) define distributional data as information about the distribution of words across linguistic contexts, and they call this type of information intra-linguistic. We would like to argue that this information might not be purely linguistic, but that it could indirectly reflect the actual distribution of the words’ referents across real-world situations. After all, language refers to objects and situations in the world. Words might co-occur in language because their referents co-occur in the real world. For example, distributional theories of word meaning consider the words *‘cat’* and *‘dog’* as similar because they often occur in similar sentences. However, the reason that they occur in similar sentences might be because these sentences typically describe real-world situations in which cats and dogs occur. In the real world, cats and dogs are often found in similar contexts (e.g., in and around the house). As a result, the sentences describing these contexts will be similar. Indeed, a recent study by Louwerse and Zwaan (2009) showed that the spatial distance between pairs of US cities can be derived from their distributions across texts. In the words of Louwerse and Zwaan (2009), “cities that are located together are debated together.” To sum up, while we agree with Andrews, Vigliocco and Vinson (2009) that distributional data form an important source of information about semantic or conceptual representations, we do not agree that this information is completely linguistic. Instead, the distribution of a word across linguistic contexts might correspond strongly to the distribution of the word's referent across real-world contexts (for a similar argument, see Zwaan & Madden, 2005).

Summarizing, we have stressed the importance of context information in conceptual processing. In addition, we have demonstrated how context information can be extracted from experience and categorized into context representations. These representations, through their connections with associated conceptual representations, can bias and constrain conceptual categorization. This is especially useful when the bottom-up input is degraded or ambiguous. By acknowledging the context, the conceptual system taps into a rich and powerful source of information. The context can prepare the conceptual system for the objects and events that are most likely given the current situation. As a result, these objects and events can be recognized and categorized more accurately. Another important consequence of such mechanism is that it allows abstract concepts, which have no perceptual features themselves, to be grounded in contextual experiences. Indeed, Barsalou and Wiemer-Hastings (2005) demonstrated that abstract concepts are based on situational information. They argue that it is impossible to represent an abstract concept like TRUTH without providing information about the situation in which this concept is set. Furthermore, although concrete concepts are typically processed better and faster than abstract concepts, these benefits disappear when abstract concepts are presented within a relevant situation (e.g., Schwanenflugel, Harnishfeger, & Stowe, 1988; Schwanenflugel, & Stowe, 1989). Thus, context information appears to be highly relevant for the representation of abstract concepts.

We live in a world full of regularities. Not only do we repeatedly encounter similar things, but we also experience similar situations and find ourselves in similar environments over and over again. Our conceptual system is specialized in extracting these regularities from experience, in order to form mental representations of objects, events and situations. This results in a rich hierarchically organized knowledge structure that may represent concepts and contexts, as well as the connections between them. Capitalizing on these connections facilitates processing in many ways. It enables one to select the representations that are relevant given the current context, to know which objects to expect in a given context, where to look for those things, and how to interact with them. In other words, it enables one to interact with the environment in a sensible and efficient manner.

References

Andrews, M., Vigliocco, G. & Vinson, D. (2009). Integrating experiential and distributional data to learn semantic representations. *Psychological Review, 116,* 463-498.

Bar, M. (2004). Visual objects in context. *Nature Reviews: Neuroscience, 5*, 617-629.

Bar, M. and Aminoff, E. (2003). Cortical analysis of visual context. *Neuron, 38,* 347-358.

Barsalou, L.W. (2005). Situated conceptualization. In H. Cohen & C. Lefebvre (Eds.), Handbook of categorization in cognitive science (pp. 619-650). St. Louis: Elsevier.

Barsalou, L.W. (1982). Context-independent and context-dependent information in concepts. *Memory & Cognition, 10,* 82-93.

Barsalou, L.W., & Wiemer-Hastings, K. (2005). Situating abstract concepts. In D. Pecher and R. Zwaan (Eds.), *Grounding cognition: The role of perception and action in memory, language, and thought* (pp. 129-163). New York: Cambridge University Press.

Chaigneau, S.E., Barsalou, L.W., & Zamani, M. (2009). Situational information contributes to object categorization and inference. *Acta Psychologica, 130,* 81-94.

Collins, A., & Quillian, M. (1969). Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior, 12,* 240-247.

Damasio, A.R. (1989) Time-locked multiregional retroactivation: a systems-level proposal for the neural substrates of recall and recognition. *Cognition, 33,* 25–62.

Davenport, J. L. (2007). Consistency effects between objects in scene processing. *Memory & Cognition, 35,* 393-401.

Davenport, J. L., & Potter, M. C. (2004). Scene consistency in object and background perception. *Psychological Science, 15,* 559-564.

Godden, D. R., & Baddeley, A.D. (1975). Context-dependent memory in two natural environments: on land and underwater. *British Journal of Psychology, 66,* 325-331.

Graf, P., & Mandler, G. (1984). Activation makes words more accessible, but not necessarily more retrievable. *Journal of Verbal Learning and Verbal Behavior, 23,* 553-568.

Hommel, B., Müsseler, J., Aschersleben, G., & Prinz, W. (2001). The theory of event coding (TEC): A framework for perception and action planning. *Behavioral and Brain Sciences, 24,* 849-878.

James, W. (1890). *The Principles of Psychology,* Cambridge, MA: Harvard University Press, 1981. Original work published 1890.

Landauer, T., & Dumais, S. (1997). A solution to Plato's problem: The Latent Semantic Analyis theory of acquistion, induction and representation of knowledge. *Psychological Review, 104,* 211-240.

Louwerse, M.M., & Zwaan, R.A. (2009). Language encodes geographical information. *Cognitive Science, 33,* 51-73.

Lund, K., & Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instrumentation, and Computers, 28,* 203-208.

Mandler, J. M., & Parker, R. E. (1976). Memory for descriptive and spatial information in complex pictures. *Journal of Experimental Psychology: Human Learning and Memory, 2,* 38–48.

Mandler, J. M., & Stein, N. (1974). Recall and recognition of pictures by children as a function of organization and distractor similarity. *Journal of Experimental Psychology, 102,* 657–669.

McRae, K., de Sa, V. R., & Seidenberg, M. S. (1997). On the nature and scope of featural representations of word meaning. *Journal of Experimental Psychology: General, 126,* 99-130.

Meyer, K., & Damasio, A. (2009). Convergence and divergence in a neural architecture for recognition and memory. *Trends in Cognitive Science, 32,* 376-382.

Meyer, D.E., & Schvaneveldt, R.W. (1971). Facilitation in recognizing pairs of words: evidence for a dependence between retrieval operations. *Journal of Experimental Psychology, 2*, 227-234.

Miller, M. B., & Gazzaniga, M. S. (1998). Creating false memories for visual scenes. *Neuropsychologia, 36,* 513–520.

Murphy, G.L. (2004). *The big book of concepts.* The MIT Press, Cambridge, Massachusetts.

Murre, J.M.J., Phaf, H., & Wolters, G. (1992). CALM: Categorizing and learning module. *Neural Networks, 5,* 55-82.

Neely, J. H. (1991). Semantic priming effects in visual word recognition: A selective review of current findings and theories. In D. Besner, & G. W. Humphreys (Eds.), *Basic processes in reading* (pp. 264-336). Hillsdale, NJ: Lawrence Erlbaum Associates.

O'Connor, C.M., Cree, G.S., & McRae, K. (2009). Conceptual hierarchies in a flat attractor network: dynamics of learning and computations. *Cognitive Science, 33,* 665-708.

Özcan, E. & van Egmond, R.(2009). The effect of visual context on the identification of ambiguous environmental sounds. *Acta Psychologica, 131,* 110-119.

Page, M.P.A. (2000) Connectionist modelling in psychology: a localist manifesto. *Behavioral and Brain Sciences, 23*, 443-467.

Potter, M.C. (1993). Very short-term conceptual memory. *Memory & Cognition, 21,* 156-161.

Potter, M.C., Kroll, J.F., Yachzel, B., Carpenter, E., & Sherman, J. (1986). Pictures in sentences: Understanding without words. *Journal of Experimental Psychology: General, 115*, 281-294.

Reynolds, J. R., Zacks, J. M., & Braver, T. S. (2007). A computational model of event segmentation from perceptual prediction. *Cognitive Science, 31,* 613-643.

Rips, L. J., Shoben, E. J., & Smith, E. E. (1973). Semantic distance and the verification of semantic relations. *Journal of Verbal Learning and Verbal Behavior, 12,* 1-20.

Rogers, T. T., & McClelland, J. L. (2004). *Semantic cognition: A parallel distributed processing approach.* Cambridge, MA: MIT Press.

Schwanenflugel, P. J., Harnishfeger, K. K., & Stowe, R. W. (1988). Context availability and lexical decisions for abstract and concrete words. *Journal of Memory and Language, 27,* 499-520.

Schwanenflugel, P. J., & Stowe, R. W. (1989). Context availability and the processing of abstract and concrete words in sentences. *Reading Research Quarterly, 24,* 114-126.

Schwanenflugel, P. J., & Shoben, E. J. (1985). The influence of sentence constraint on the scope of facilitation for upcoming words. *Journal of Memory and Language, 24,* 232-252.

Smith, E. E., Shoben, E. J., & Rips, L. J. (1974). Structure and process in semantic memory: A featural model for semantic decisions. *Psychological Review, 81,* 214–241.

Sokolov, E. N. (1963). *Perception and the conditioned reflex.* New York: Macmillan.

Steyvers, M. (in press). Combining feature norms and text data with topic models. *Acta Psychologica*.

Tversky, B., Zacks, J. M., & Martin, B. (2008). The structure of experience. In T. F. Shipley & J. M. Zacks (Eds.), *Understanding events: From perception to action*. (pp. 436-464).

Yeh, W., & Barsalou, L.W. (2006). The situated nature of concepts. *American Journal of Psychology, 119,* 349-384.

Wolters, G., & Phaf, R.H. (1990). Implicit and explicit memory: implications for the symbol-manipulation versus connectionism controversy. *Psychological Research, 52,* 137-144.

Zwaan, R.A., & Madden, C.J. (2005). Embodied sentence comprehension. In Pecher, D. & Zwaan, R.A. (Eds.) *Grounding cognition: The role of perception and action in memory, language, and thinking*. (pp 224-245). Cambridge University Press, Cambridge, UK.

Zeelenberg, R., Pecher, D., Shiffrin, R.M., & Raaijmakers, J.G.W. (2003). Semantic context effects and priming in word association. *Psychonomic Bulletin & Review, 10*, 653-660.

Footnotes

Throughout this paper, angle brackets will be used to describe features (e.g., <round>), uppercase font will be used to describe concepts (e.g., BALL), and uppercase italic font will be used to describe contexts (e.g., *KITCHEN*).

[2] The term *distributional* data is borrowed from Andrews, Vigliocco and Vinson (2009). In their paper, however, the term had a slightly different meaning. The authors referred to experiential (feature-based) and distributional information as two contributors of word meaning. In their definition, distributional information refers to the linguistic contexts (sentences, paragraphs) in which a word occurs. In the current paper, distributional information refers to the actual contexts in which a concept occurs.

[3] The Event Segmentation Theory (Zacks, 2007) supposes that the perceptual system not only perceives current input, but also predicts what will be perceived next. This prediction is based on the current input as well as on more stable information about the ongoing event, which is represented as an *event model*. Event boundaries are usually characterized by a relatively high *prediction error*, which occurs when the system's prediction is substantially different from the actual input. Whenever an event boundary is perceived, the event model is reset in order to represent the new event.

Table 1

Example of Output Table.

		Concept Node															
Input pattern	Nr of presentations	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	15									15							
1	15				14				1								
2	15						15										
3	15											15					
4	15							15									
5	15								15								
6	15		2								13						
7	15		15														
8	15			14						1							
9	15															15	
1	15	15															
11	15														15		
12	15													15			
13	15																15
14	15			1		13							2				
15	15					3							12				

Table 2

Results of Simulation I. Convergence Index and Separation Index of the Concept Module and Context Module under Conditions of Low and High Pattern Overlap.

Input	Module	Convergence Index	Separation Index
Low overlap	Concept Module	0,98	0,96
	Context Module	0,94	0,95
High overlap	Concept Module	0,88	0,85
	Context Module	0,83	0,82

Table 3

Example of Overlapping Patterns and Ambiguous Test Pattern Presented in Simulation II.

Concept	Distributed Feature Pattern												Context
1	1	0	1	1	1	0	0	1	0	1	1	0	A
2	1	0	1	0	1	0	0	1	0	1	1	1	B
Test	1	0	1	0.5	1	0	0	1	0	1	1	0.5	A/B

Figure Captions

Figure 1. CONCAT Architecture

Figure 2. Wiring of internodes

Figure 3. Snapshots of activation propagation in the model. The figure only shows the Feature Detection Layer, Concept Module, Memory Buffer and Context Module. Four different input patterns belonging to the same context are presented. In Panel *a.* the first input pattern is presented, categorized by the Concept Module and stored in the Memory Buffer. In Panel *b.*, the second input pattern is categorized and stored in the Memory Buffer. In Panel *c.,* the third input pattern is categorized and stored in the Memory Buffer. The total activation of the Memory Buffer now exceeds a certain value, resulting in activation of the Context Module, which categorizes the pattern in the Memory Buffer. In Panel *d.,* the activated context node sends top-down (voltage-dependent) input to all associated concept nodes. This facilitates the categorization of a new pattern belonging to the same context.

Figure 4. CALM Module

Figure 5. Results of Simulation II. Accuracy of categorization of ambiguous input patterns by model with feedback connections and without feedback connections.

Figure 6. Results of Simulation III. Accuracy of categorization of input patterns under different conditions of degradation by model with feedback connections and without feedback connections.

Figure 7. Results of Simulation III. Convergence times under different conditions of degradation by model with feedback connections and without feedback connections.

Figure 1

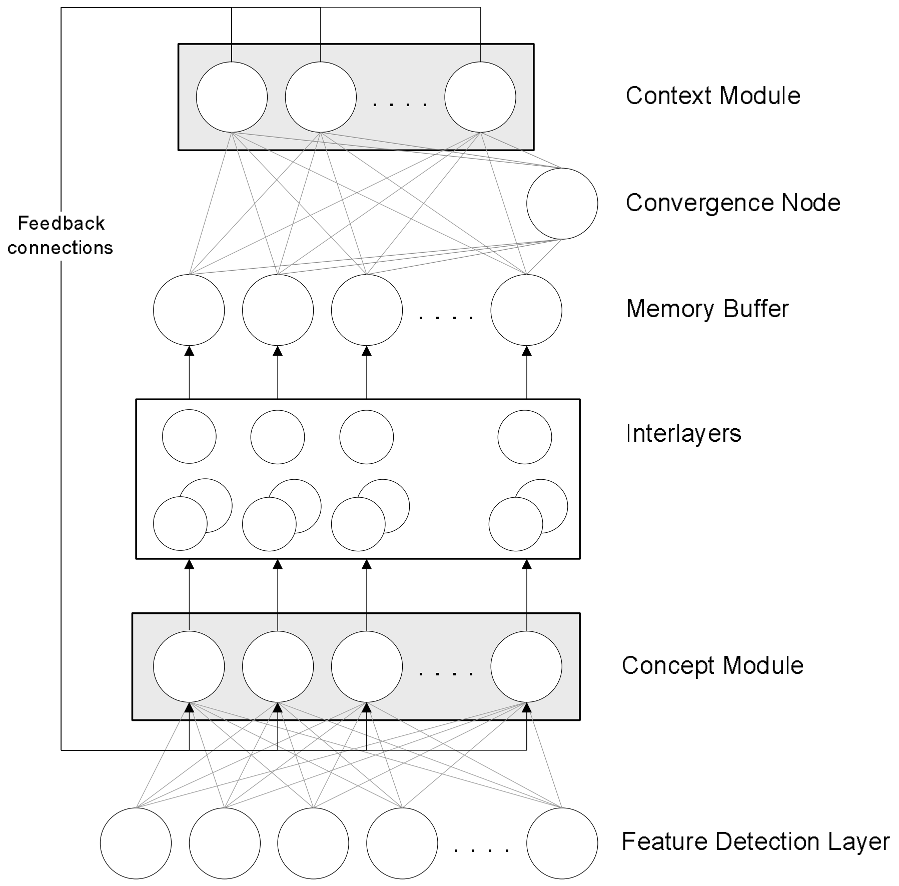

Context Module
Convergence Node
Feedback connections
Memory Buffer
Interlayers
Concept Module
Feature Detection Layer

Figure 2

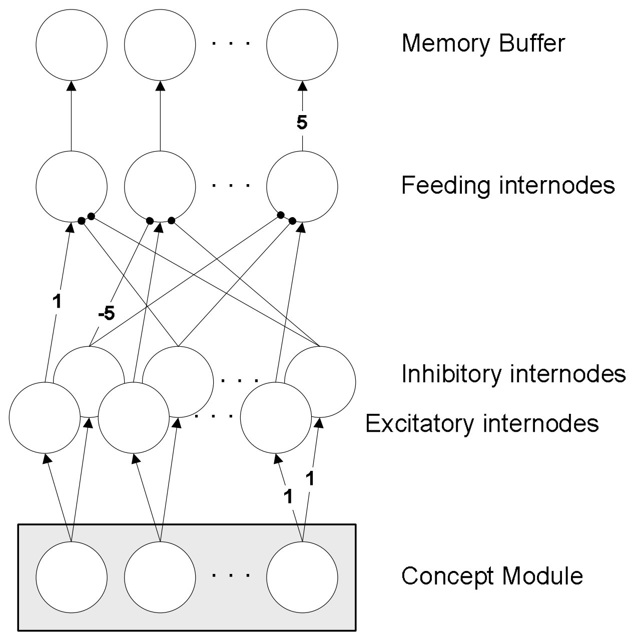

Memory Buffer
Feeding internodes
Inhibitory internodes
Excitatory internodes
Concept Module
5
1
-5
1
1

Figure 3

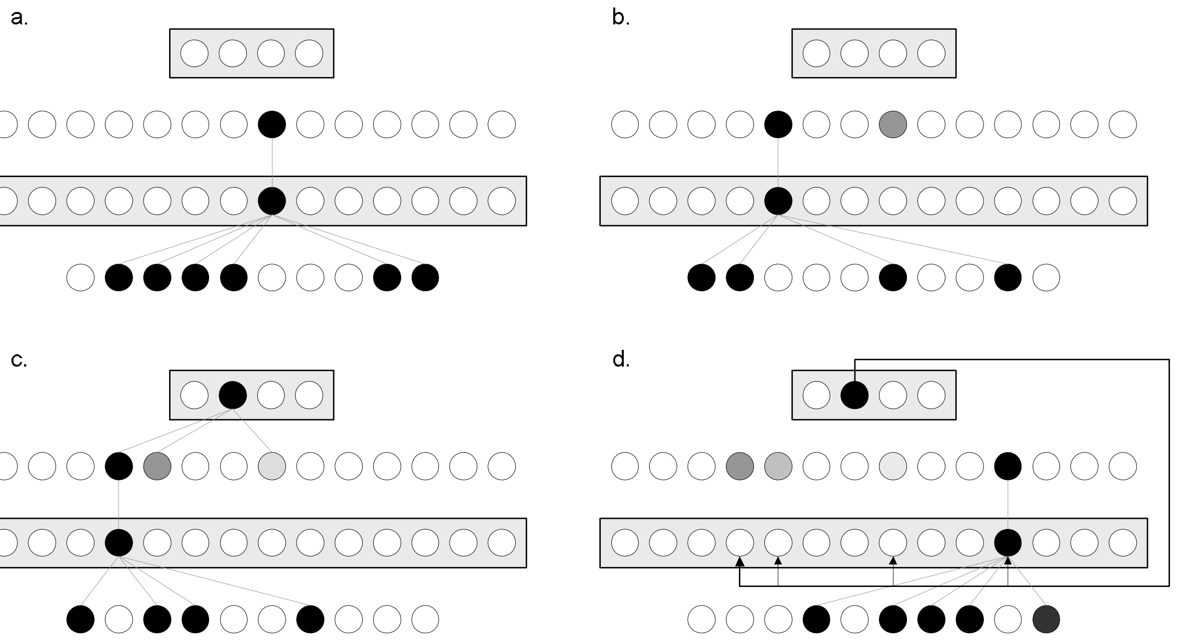


Figure 4

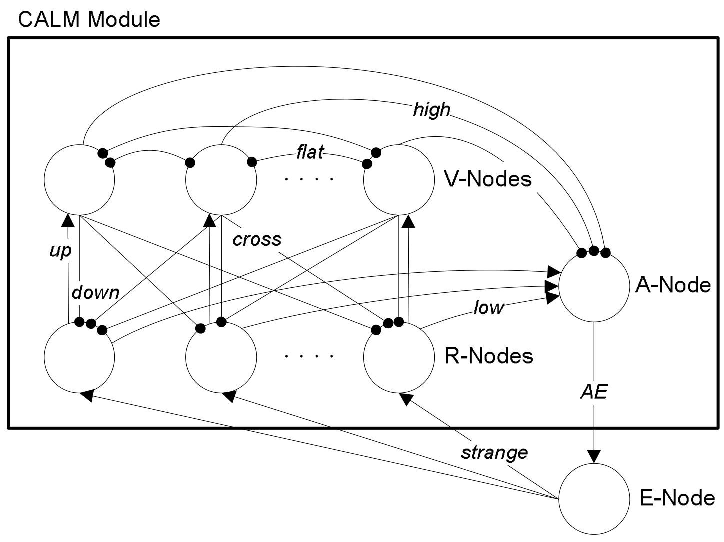

CALM Module
high
flat
V-Nodes
up
cross
down
A-Node
low
R-Nodes
AE
strange
E-Node

Figure 5

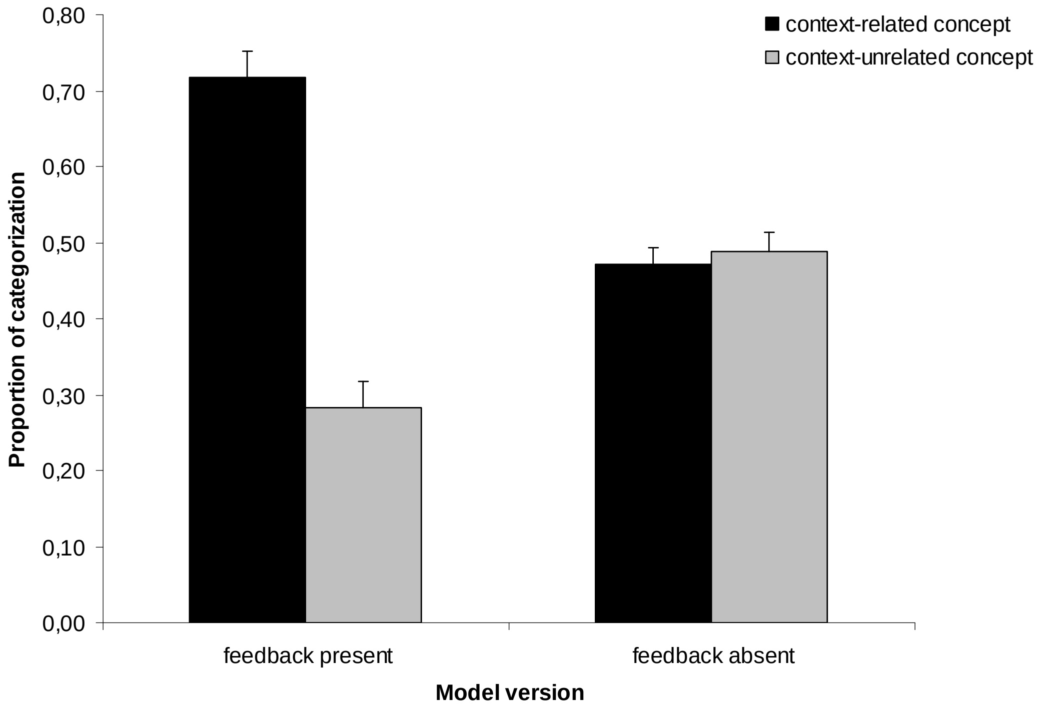

0,80
0,70
0,60
0,50
0,40
0,30
0,20
0,10
0,00
Proportion of categorization
context-related concept
context-unrelated concept
feedback present
feedback absent
Model version

Figure 6

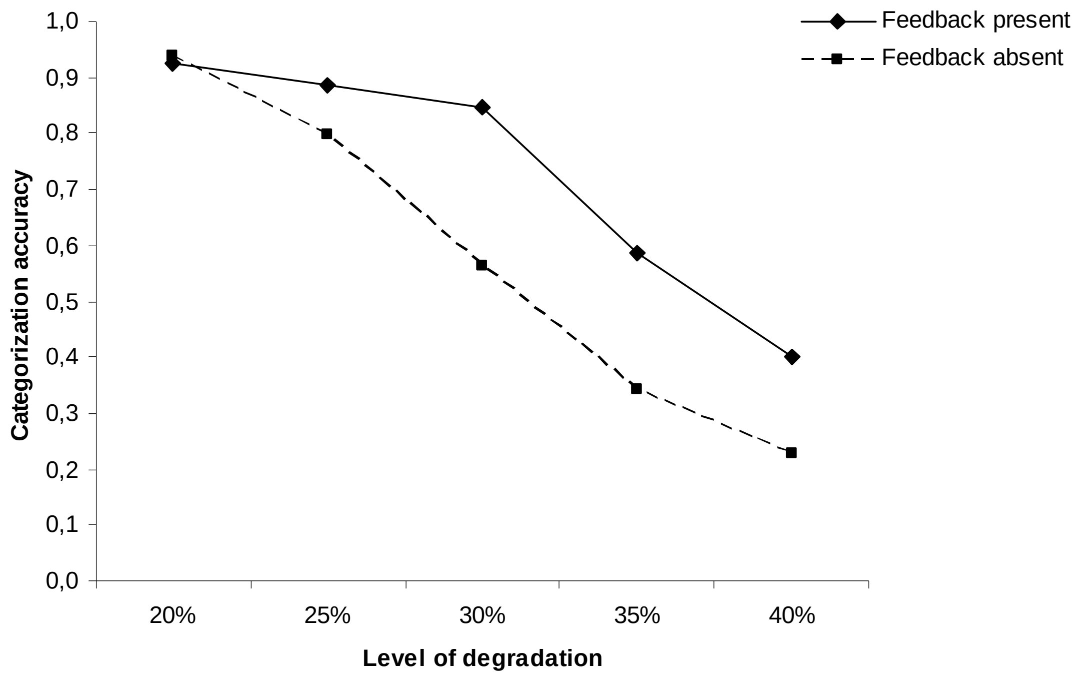
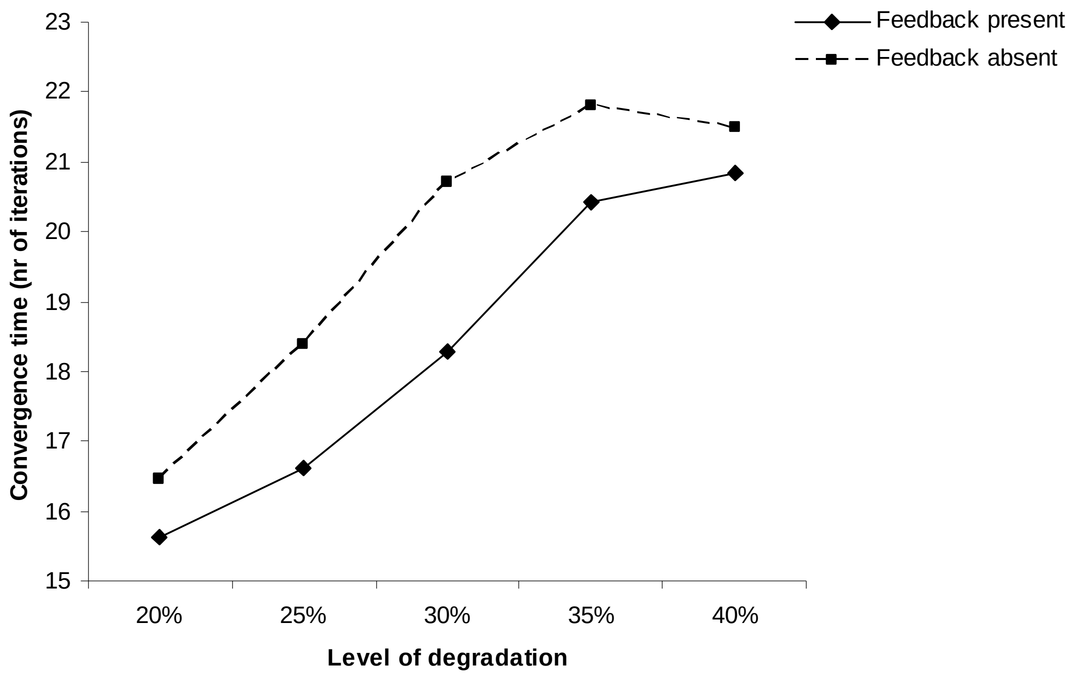

Feedback present
Feedback absent
Categorization accuracy
1,0
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0,0
20%
25%
30%
35%
40%
Level of degradation

Figure 7


23
22
21
20
19
18
17
16
15
Convergence time (nr of iterations)
20%
25%
30%
35%
40%
Level of degradation
Feedback present
Feedback absent

APPENDIX

Table 1

Weight Values of Connections within CALM Modules, as specified by Murre, Phaf & Wolters (1992).

Weights	**Description**	**Value**
Up	Connects R-node to its matched V-node	0.5
Down	Connects V-node to its matched R-node	-1.2
Cross	Connects V-node to all non-matched R-nodes	-10.0
Flat	Interconnects all V-nodes (no self-connections)	-1.0
Low	Connects R-node to A-node	0.4
High	Connects V-node to A-Node	-0.6
AE	Connects A-node to E-node	1.0
Strange	Connects E-node to R-nodes	0.5
Inter	Initial value of intermodular adaptive weights	0.6

Table 2

Equation Parameters of Concept Module

Globals	**Description**	**Value**
k	Decay of activation in an iteration	0.05
L	Learning competition between bottom-up connections	1.0
K	Maximum value of bottom-up weights	1.0
d	Base rate of learning	0.001
$W_{\mu E}$	Virtual weight between E-node and learning rate	0.015
C_h	High convergence threshold	0.001
C_l	Low convergence threshold	0.0001

Table 3

Equation Parameters of Context Module

Globals	**Description**	**Value**
k	Decay of activation in an iteration	0.025
L	Learning competition between connections	1.0
K	Maximum value of learning weights	1.0
d	Base rate of learning	0.005
$W_{\mu E}$	Virtual weight between E-node and learning rate	0.025
C_h	High convergence threshold	0.001
C_l	Low convergence threshold	0.0001

Table 4

Weight Values of Connections between Different Layers of the Model

Weights	**Description**	**Value**
R-I	Connects R-node in Concept Module to matched Inhibitory internode	1
R-E	Connects R-node in Concept Module to matched Excitatory internode	1
I-F	Connects Inhibitory internode to all non-matched Feeding nodes	-5
E-F	Connects Excitatory internode to matched Feeding node	1
F-M	Connects Feeding node to matched Memory Buffer node	5
M-Cv	Connects Memory Buffer nodes to Convergence node	1
Cv-C	Connects Convergence node to R-nodes in Context Module	2
M-C	Initial value of connections between Memory Buffer nodes and R-nodes in Context Module	0.6
FB	Initial value of Feedback connections (from R-nodes in Context Module to R-nodes in Concept Module)	0.6

Table 5

Equation Parameters of Other Elements of the Model

Parameter	**Description**	**Value**
k_{inter}	Decay of activation of internodes	0.1
k_{mem}	Decay of activation of nodes in Memory Buffer	0.001
β	Gain of input function of Convergence Node	-20
θ	Threshold of input function of Convergence Node (dependent on nr of concepts per context)	1.5 - 3
Inht	Tonic inhibition to R-nodes in Context Module	-1.05
L_{fb}	Learning competition of feedback connections	1.5
K_{fb}	Maximum value of feedback weights	1.5
d_{fb}	Base rate of learning of feedback connections	0.005

Table 6

Simulation Parameters

Description	**Value**
Number of nodes in Feature Detection Layer	12
Number of nodes in Concept Module	16
Number of nodes in Context Module	4
Iterations per concept presentation	60
Number of context episodes during training	80
Number of context episodes during testing	20
Number of iterations during short reset interval	120
Number of iterations during long reset interval	3000

Continuous Surface-point Distributions for 3D Object Pose Estimation and Recognition

Renaud Detry
University of Liège
Belgium
Renaud.Detry@ULg.ac.be

Justus Piater
University of Liège
Belgium
Justus.Piater@ULg.ac.be

Abstract

We present a 3D, probabilistic object-surface model, along with mechanisms for constructing a model from unregistered 2.5D views and segmenting model instances in cluttered scenes. The object representation is a probabilistic expression of object parts through smooth surface-point distributions obtained through kernel density estimation on 3D point clouds. A multi-part, viewpoint-invariant model is learned incrementally from a set of roughly segmented, unregistered views, by sequentially registering and fusing the views with the incremental model. Registration is conducted by nonparametric inference of maximum-likelihood model parameters, using Metropolis–Hastings MCMC with simulated annealing. This mechanism is robust to clutter, and avoids direct model-to-scene correspondences. The learning of viewpoint-invariant models and the applicability of our method to pose estimation, object detection, and object recognition is demonstrated on 3D-scan data, providing qualitative, quantitative and comparative evaluations.

1. Introduction

Our object model constitutes a 3D, generative, part-based, probabilistic expression of object surface primitives and their spatial configuration.

The definition of object models in 3D point clouds has been achieved through a variety of approaches [4]. The idea is generally to break down the object surface into a number of primitives; an object is then described by describing each primitive, and possibly also their spatial configuration. Primitives correspond e.g. to complex parametric shapes such as superquadrics [2, 17], local surface descriptor [7, 8, 9, 12, 14], or local edge descriptors [5]. In this work, primitives correspond directly to 3D points, with each point further parametrized by a local surface orientation computed from the distribution of the k nearest neighbors. In the following, we simply refer to these as *points*.

Depending on the application, recording the geometric configuration of surface primitives may or may not be necessary. Methods aiming at object recognition (without segmentation/pose estimation) [7, 8, 12] can afford to completely ignore the spatial configuration of primitives, or encode it implicitly. They can also afford to match a model by matching each view contained in the model separately, therefore also avoiding model view registration. Object recognition finally motivates *discriminative* models. Conversely, pose estimation and segmentation require the modeling of the global shape of the object through the encoding of relative primitive configurations [9, 14, 5]. This generally leads to a *generative* model. Although it may not be their primary aim, generative models often provide recognition, too [14]. When building a generative 3D model from multiple views, it becomes necessary to derive an exhaustive registration of individual 2.5D views. Our method learns a initial model from a single view of an object; the model can then be used to detect and estimate the pose of the object in novel scenes, provided that the view is sufficiently similar. If a new view provides more information, the model can be extended, in principle until the entire surface is completely modeled. Thus, model learning and model exploitation are seamlessly integrated.

Detection and alignment of generative models are typically achieved through the matching of model descriptors to scene descriptors, possibly followed by the derivation of a model pose from geometrically-constrained optimization [14]. We follow a different approach and encode object structure through a continuous probability density function representing the distribution of object-surface points. This allows us to achieve detection through probabilistic inference, which in turn avoids explicit model-to-scene correspondences. Our model is inferred by a Markov-chain Monte-Carlo (MCMC) algorithm which yields the maximum-likelihood pose of a model in an arbitrary scene.

Within a point-cloud reconstruction, the quantity of information conveyed by a point from a large and uniform surface is arguably smaller than the information conveyed by a point on a smaller, distinctive surface. In other words, the contribution of a surface segment to the identity of an object is generally not proportional to the number of points supporting the segment in a point-cloud reconstruction. Many 3D modeling techniques acknowledge this observation, e.g. through the detection of salient points [11], or the use of surface primitives of varying size [14]. We proceed by splitting object points into groups that represent object *parts* of different spatial size, and give each part the same weight in the detection process.

The learning of viewpoint-invariant models is demonstrated on 3D-scan data from Biegelbauer and Vincze [2], and on the popular CAD dataset of Hetzel *et al*. [8]. The applicability of our method to pose estimation in cluttered point clouds is demonstrated on the data of Biegelbauer and Vincze, and object recognition rates are presented for the dataset of Hetzel *et al*.

This work is largely inspired by the work of Detry *et al*. [5]. The main differences, and key contributions of this paper are: (a) a maximum-likelihood Monte Carlo inference algorithm, (b) a learning method which autonomously registers independent views, and autonomously identifies parts from spatial structure, and (c) experiments on range data with both pose estimation and object recognition results.

2. Object Model

As mentioned above, we consider point-cloud reconstructions in which each point is composed of a position and a local surface normal. The surface normal is computed at each point of the cloud from the covariance matrix C of its k nearest neighbors [13]. Let us denote by e_1, e_2, e_3 the eigenvalues of C, with $e_1 \geq e_2 \geq e_3$; depending on whether $e_1 - e_2$ is smaller or greater than $e_2 - e_3$, the local direction is set to the eigenvector associated to e_3 or e_1 respectively, allowing for stable orientations on both surface and line configurations. Denoting by $S^2_{\mathbf{z}+}$ the $\mathbf{z}$-positive hemisphere of the 2-sphere (the space of unoriented direction vectors), computing local orientations yields a point-cloud $O = \left\{\left(\lambda^\ell, \theta^\ell\right)\right\}_{\ell\in[1,n]}$ where $\lambda^\ell \in \mathbb{R}^3$ and $\theta^\ell \in S^2_{\mathbf{z}+}$.

Our pose estimation method relies on the modeling of 3D surface with *surface-point distributions* through kernel density estimation. Kernel density estimation (KDE) is a technique that allows one to model a continuous density function from a set of *observations* drawn from it, by assigning a local kernel function to each observation [16]. The value of the continuous function at an arbitrary point x is defined as the sum of the evaluation of all kernels at x. Random variates from the density are drawn by selecting an observation at random and drawing from the associated kernel. KDE allows us to define a continuous *surface-point distribution* from the point-cloud reconstruction of an object. The surface observations we are dealing with are points which belong to $\mathbb{R}^3 \times S^2_{\mathbf{z}+}$; denoting the separation of point components and kernel parameters into position and orientation by $x = (\lambda, \theta)$, $\mu = (\mu_t, \mu_r)$, $\sigma = (\sigma_t, \sigma_r)$, an applicable kernel for our surface-point space is defined with

$$\mathbf{K}_{\mu,\sigma}(x) = \mathbf{N}_{\mu_t,\sigma_t}(\lambda)\left[\mathbf{\Theta}_{\mu_r,\sigma_r}(\theta) + \mathbf{\Theta}_{\mu_r,\sigma_r}(-\theta)\right] \quad (1)$$

where μ is the kernel mean point, σ is the kernel bandwidth, $\mathbf{N}(\cdot)$ is a trivariate isotropic Gaussian kernel, and $\mathbf{\Theta}(\cdot)$ is the von-Mises Fisher distribution [6] on the 2-sphere in $\mathbb{R}^3$. The pair of opposite $\mathbf{\Theta}$ kernels in $\mathbf{K}$ ensures that $\mathbf{K}_{(\lambda,\theta),\sigma}(x) = \mathbf{K}_{(\lambda,-\theta),\sigma}(x)$ [18]. The bandwidths σ_t and σ_r are computed using a k-nearest neighbor technique [16] on point positions.

Given a point-cloud reconstruction $\{x^\ell\}_{\ell\in[1,n]}$ of an object, a surface-point distribution $\psi(x) = \sum_{\ell=1}^{n} \mathbf{K}_{x^\ell,\sigma}(x)$ can readily be used as a "3D template" that provides an *object pose likelihood* when convolved with the surface-point distribution of an arbitrary scene. This observation probably constitutes the most intuitive illustration of our method.

We model an object composed of p parts with a set of surface-point distributions $\psi_i(X_i)$, where $X_i \in \mathbb{R}^3 \times S^2_{\mathbf{z}+}$ is a random variable denoting the distributions of points belonging to part i. All parts are defined in a common reference frame, so that $\sum_{i=1}^{p} \psi_i(x)$ yields a reconstruction of the whole object. Let us denote by $SE(3) = \mathbb{R}^3 \times SO(3)$ the group of 3D poses, and by $\phi(x)$ the surface-point distribution of an arbitrary scene in which the object appears. We model the pose of an object with a random variable $W \in SE(3)$; the pose distribution in the scene is given by

$$p(W) \propto \prod_{i=1}^{p} \underbrace{\int \psi_i\left(t_W^{-1}(X_i)\right)\phi(X_i)\mathrm{d}X_i}_{m_i(W)}, \quad (2)$$

where $t_W^{-1}(X_i)$ denotes the rigid transformation of X_i by the opposite of W, such that $t_w \circ t_w^{-1}$ yields an identity transformation for all $w \in SE(3)$. Each integral $m_i(w)$ corresponds to the evaluation at w of the convolution of part i with the scene.

The maximum-likelihood (ML) pose of an object is given by $\arg\max_w p(w)$, which is analytically intractable; we thus naturally turned to Monte Carlo methods. Evaluating the integrals $m_i(w)$ is done by sampling n times $x_i^\ell \sim \psi_i(X_i)$, evaluating $\phi(t_w(x_i^\ell))$, and taking the average over n, which produces an approximation of $m_i(w)$ up to a multiplicative constant. Simulating $p(W)$ directly is not possible, although simulating one integral $w \sim m_i(W)$ can be achieved by generating $x_i \sim \phi(X_i)$; w is then obtained as $w \sim \psi_i(t_W^{-1}(x_i))$.

The ML pose of an object in a scene is computed via simulated annealing on a Markov chain whose invariant distribution at iteration j is proportional to $p^{1/T_j}(W)$

[10, 1], where T_j is a decreasing cooling schedule such that $\lim_{j\to\infty} T_j = 0$. The chain is defined with a mixture of two local- and global-proposal Metropolis–Hastings transition kernels. Our choosing of the standard Metropolis–Hastings algorithm is motivated by the complexity of $\mathbb{R}^3 \times S^2_{\mathbf{z}+}$, which renders the calculation of complex quantities such as local derivatives difficult. Also, $p(W)$ is likely to present a large number of narrow modes. A mixture of global and local proposals will compromise between distributed exploration of the pose space and fine tuning of promising regions. The independence-chain component of our transition kernel requires a global proposal function which we can simulate, and which should ideally resemble $p(W)$. In this paper, the global proposal corresponds to $s(W) = \sum_i m_i(W)$, which can be simulated by selecting $i \sim U_{[1,...,p]}$, and sampling from $m_i(W)$. The local proposal for the random-walk component of the transition kernel is given by the $SE(3)$ kernel

$$\mathbf{K}^{\star}_{\mu,\sigma}(x) = \mathbf{N}_{\mu_t,\sigma_t}(\lambda)\left[\mathbf{\Theta}^{\star}_{\mu_r,\sigma_r}(\theta) + \mathbf{\Theta}^{\star}_{\mu_r,\sigma_r}(-\theta)\right] \quad (3)$$

where $\mathbf{\Theta}^{\star}$ is the von-Mises Fisher distribution on the 3-sphere, and rotations θ and μ_r are expressed as quaternions [18]. The location bandwidth σ_t of this kernel is set to a fraction of the size of the object, which in turn is computed as the standard deviation of input object points to their center of gravity. Its orientation bandwidth is set to a constant allowing for 5° of deviation. The complete algorithm is listed as Algorithm 1. For our purposes, the mixture weight ν is typically set to 0.75; T_0 and T_N are set to 0.5 and 0.05 respectively; N is of the order of 5000. Simulated annealing does not guarantee convergence to the global maximum of $p(W)$. We thus run several chains in parallel and eventually select the best estimate.

The model presented above is intrinsically sensible to relative spatial resolution within the input point cloud: the convolution of parts with scene evidence (2) will be proportional to the global value scale of $\psi(X_i)$ in the region covered by part–surface-point densities. Unfortunately, the spatial resolution of 3D-scans is generally not uniform. For example, objects closer to the sensor will generate more return than further ones. Hence, the maximum of the pose likelihood (2) may not correspond to the pose that best explains *surface shape*. In the experiments presented below, we largely mitigate this effect by evaluating scene densities $\phi(x_i)$ as the *maximum* of underlying kernel evaluations at x_i. We note that model distributions $\phi_i(X_i)$ are not concerned by this issue since they are integrated over multiple views.

Finally, we note that the independence properties of the random variables involved in the expression of the object pose distribution (2) can be highlighted in a Markov random tree of unit height of which W is the root and X_i are the leaves. The network compatibility potential linking W to X_i directly corresponds to $\psi_i(t_W^{-1}(X_i))$; observation potentials are given by $\phi(X_i)$. Each integral $m_i(W)$ corresponds to the message sent from X_i to W in a belief-propagation inference of the marginal distribution $p(W)$.

Algorithm 1 Simulated annealing

```
Initialize $w_0$ arbitrarily
Initialize $\sigma_t$ and $\sigma_r$ as explained in the text
For $j = 0$ to $N$ :
  $T_j = \max\left\{T_0\left(\frac{T_N}{T_0}\right)^{j/N}, T_N\right\}$
  Sample $u \sim U_{[0,1]}$
  If $u < \nu$ :
    Sample $w_* \sim s(W)$
    Sample $v \sim U_{[0,1]}$
    If $v < \min\left\{1, \left(\frac{p(w_*)}{p(w_j)}\right)^{1/T_j}\frac{s(w_j)}{s(w_*)}\right\}$ : $w_{j+1} = x_*$
    Else : $w_{j+1} = w_j$
  Else :
    Sample $w_* \sim \mathbf{K}^{\star}_{w_j,(\sigma_t,\sigma_r)}(W)$
    Sample $v \sim U_{[0,1]}$
    If $v < \min\left\{1, \left(\frac{p(w_*)}{p(w_j)}\right)^{1/T_j}\right\}$ : $w_{j+1} = x_*$
    Else : $w_{j+1} = w_j$
```

3. Learning

The generation of a model from a single point cloud reconstruction of an object is described in Section 3.1. Section 3.2 explains how a model is learned from multiple views.

3.1. Modeling A Point-cloud Reconstruction

Learning a model from a point-cloud reconstruction amounts to identifying the number and shape of object parts. Object parts are computed by clustering object points in $\mathbb{R}^3$; they are identified through the mixture of k trivariate Gaussians that best explains object point positions. K mixtures of $p = 1, ..., K$ Gaussians are fit using the Expectation-Maximization (EM) algorithm, and the most appropriate mixture is selected in a way inspired by the Bayesian information criterion [15]: the selected mixture is the one that minimizes

$$-2\log L + Cp\log n, \quad (4)$$

where L is the maximized value of the data likelihood, n is the number of points, and C is a numerical constant which allows us to strongly penalize large mixtures, hence keeping the number of parts reasonably low. The object model M is eventually composed with p surface-point distributions

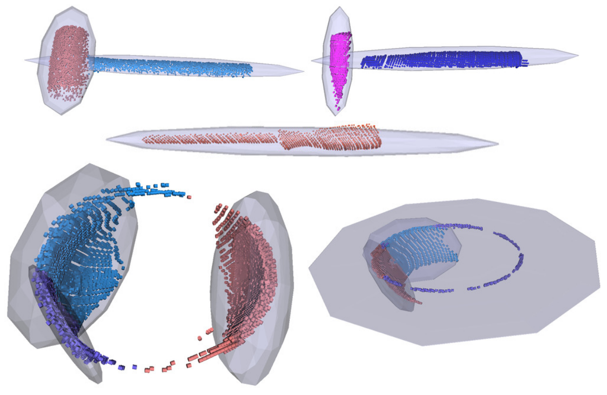


Figure 1: Objects and their parts. Color indicates to which part a point belongs. Ellipsoids illustrate the mixture that identifies object parts.

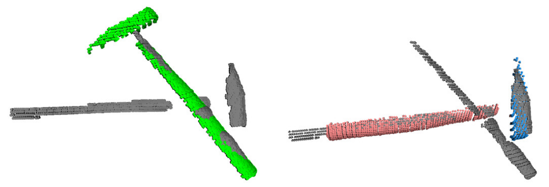


Figure 2: Example pose estimates. Grey dots correspond to scene points; color dots show object points aligned to the scene through pose estimation. The single-part model fails to produce a correct estimate (left), whereas a two-part model succeeds (right).

$\psi_i(X_i)$ built through KDE on the points that belong to cluster i.

Clustering is responsible for the identification of characteristic object parts (Figure 1), drawing attention to smaller areas that would otherwise be overwhelmed by larger surfaces. In Figure 2, the left image shows a pose estimate computed from a single-part model of a hammer. Inference finds a best match of the handle of the hammer on a screwdriver, and ignores the unmatched head of the hammer. In the right image, the two-part model of Figure 1 draws inference towards a correct estimate. The part identification method described above is similar to the procedure of Bouchard and Triggs [3], except that their work eventually expresses parts as cluster centers. Here, clustering is exclusively used to *identify* parts. Parts are represented by fine-grained surface-point densities (Section 2), which hold significantly more information than a single Gaussian.

3.2. Learning From Multiple Views

The construction of a model that expresses the full 3D geometry of an object requires pairwise registration of multiple views. Naturally, only pairs of sufficiently overlapping views can be registered. Finding overlapping views through an exhaustive registration of all pairs is unfortunately rather inefficient. Therefore, a meta-process should ideally detect strongly correlated views, which are good candidates for registration [14]. In this section, we present a somewhat simpler method, which iteratively integrates views into a model, expecting each additional view to overlap with at least one of the previous views. Let us assume that each view contains n points, and let us denote by M_ℓ a model made up of ℓ views, and denote by O_ℓ the set of points used to construct M_ℓ. The first model M_1 is built, following the procedure of the previous section, from the points produced by the first available view v_1. Let us then assume that we have a model M_ℓ constructed from ℓ views, and the set O_ℓ from which it was built. Adding $v_{\ell+1}$ to the model works as follows. The pose of M_ℓ is estimated in $v_{\ell+1}$ (Section 2), which allows us to transform the points of $v_{\ell+1}$ into the object reference frame, yielding an object-registered point set T. A set of points $O_{\ell+1}$ that spans $\ell+1$ views is then formed by sampling $n/(\ell+1)$ points from T and $n\ell/(\ell+1)$ points from O_ℓ. $M_{\ell+1}$ is constructed by applying the procedure of Section 3.1 to $O_{\ell+1}$.

4. Evaluation

In the following experiments, models typically contain 1 to 4 parts. Scene surface-point distributions are computed from 5000 scan points. The total number of surface-point observations within object parts is limited to 500. The number of parallel chains in MCMC inference is typically set to 16. Our `C++` implementation estimates the pose of a model in a scene in about 5–10s on an 8-core desktop computer, and its memory footprint is always below 50MB. The cost of detecting multiple objects is linear in the number of objects. Source code for learning and inference is available online (link to be provided in the final copy).

4.1. Cluttered-scene Pose Estimation

The suitability of our model for pose estimation in cluttered scenes is demonstrated on 3D-scan data from Biegelbauer and Vincze [2], which are available online (link provided in the camera-ready version of the paper). We learned a model of a mallet, a hammer, a screwdriver, and two bowls, using between 1 and 4 segmented range views of each object. The objects and their parts are illustrated in Figure 1. The pose of these objects was estimated in 4 range scenes. Because points from the ground plane represent approximately 85% of each scene, we removed these prior to detecting the objects, by isolating them through RANSAC plane fitting. Although this step is not necessary, it significantly lowers inference time. As illustrated in Figure 3, all 11 pose estimates were correct. We followed the scenario of Biegelbauer and Vincze and reproduced the experiment several times using different software random seeds, and every

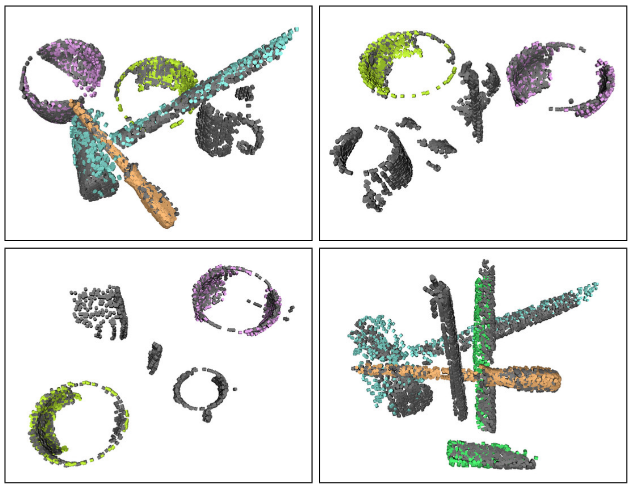


Figure 3: Cluttered scenes with pose estimates. Grey dots correspond to scene points. The rest of the dots illustrate pose estimtates: they correspond to object points (Figure 1) aligned to the poses of the objects in the scenes.

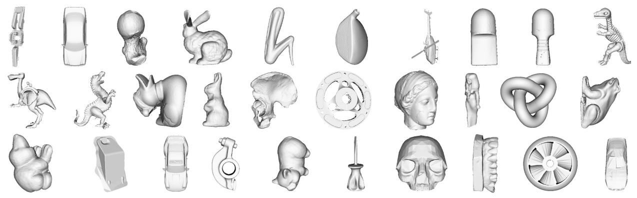


Figure 4: Object library from Hetzel *et al*. [8]. Illustration kindly provided by Li and Guskov [12].

run lead to the same correct estimates. When using models made of a single part, instead of the multi-part models of Figure 1, only 7 out of 11 poses were correct, for reasons identical to these explained in Section 3.1. Despite its simplicity, the part-learning process is instrumental in discriminating between objects of similar shapes.

We note that the scenes of Figure 3 contain more objects than those shown in Figure 1. These objects are not part of the experiment simply because we do not have segmented views of them. A segmentent view of the deeper bowl (pruple in Figure 3) was also missing. Its model was built from data extracted from the top-left scene of Figure 3. It seemed relevent to include that object because of its similarity with the second (yellow) bowl.

4.2. Object Detection and Recognition

The primary purpose of our model is to provide a ML object pose. Yet, the value of the pose-likelihood expression (2) at its peak may be used as a matching score, hence yielding object detection and recognition.

Object detection was evaluated on the online-available CAD dataset of Hetzel *et al*. [8]. This dataset contains 258

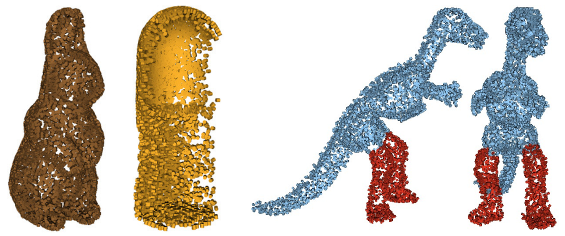


Figure 5: Object points obtained from the registration of sequences of 66 views. Color indicates learned parts; the bunny and deodorant bottle are made up of a single part, whereas the dinosaur yields a two-part model (legs and body).

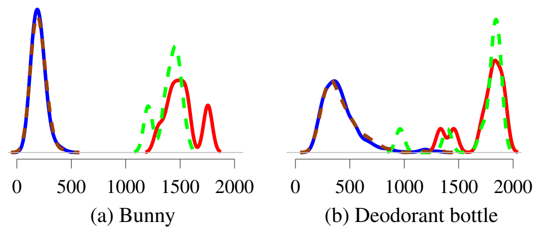


Figure 6: Distribution of detection scores for the bunny and deodorant bottle (see Figure 5). The dashed green line shows the distribution of the scores resulting from the instantiation of the object model in all training images of that object. The dashed brown line (almost overlapping with the blue line) corresponds to the instantiation of the model in the rest of the training set. The red line corresponds to testing images of the object; the blue line corresponds to testing images of the other objects. The scores provided by the bunny model are clearly separable. The deodorant bottle is less robustly detected, largely because of a second, similar bottle in the object set.

simulated range images for each of its 30 objects (Figure 4). It is divided into a training and a testing set, containing respectively 66 and 192 views of each object. We learned a view-invariant model of each object using its 66 training views, providing them to the multi-view learning algorithm of Section 3.2 in the order in which they appear on the dataset website. Even though this order is not always ideal, it allowed for the construction of a good model of most objects. Figure 5 shows three examples. The bunny and the dinosaur are correctly reconstructed. The deodorant bottle is missing a side; this is explained by the symmetry of the object, which causes all model views to be registered to the same side.

Object detection determines whether a given model is present in a given view. In order to produce object detection decisions, all 30 models were instantiated in 300 views of the training set – 10 views from each object, providing 9000 training scores. Figure 6 (dashed curves) shows the distribution of the resulting detection scores for two objects. We ran the same process on 300 images from the testing set (solid curves of Figure 6), yielding 9000 testing scores. For each object o, we trained a binary naive Bayes classifier on the 300 training scores produced by o, providing means of distinguishing o from all other objects. Confronting the testing scores to the 30 detection classifiers yielded a 98% detection rate, i.e. out of the 9000 binary classifications, there are 298 true positives, 8580 true negatives, 2 false negatives and 120 false positives.

By contrast to object detection, object recognition determines, given one view, which object this view is most likely to show. For this purpose, we trained on the 9000 training scores a single SVM classifier to determine, given the matching scores of the 30 objects on one view v, which object is in v. This experiment yielded a 99% recognition rate, i.e. 297 true positives and 3 false positives. This result is directly comparable and competitive with recent discriminative approaches on the same dataset which yield 98% [12] and 93% [8]. It is also comparable with the 95% presented in Section 8.1 of the article from Mian *et al.* [14], although the object library used by Mian *et al.* is a superset the one we are using.

We emphasize that the classifiers above are only applied to find appropriate score separating thresholds or planes. The underlying inference mechanism is not discriminative, and goes further than object recognition by providing an $SE(3)$ object pose. We have not quantified the accuracy of pose estimation. However, we note that the biggest difficulty in inference is to find the largest mode of the pose density. Once the Markov chain has found it, moving to its maximum is an easier task. The informative value of quantitative accuracy tests is thus limited. A more intersting study would be that of convergence rate as a function of the amount of scene clutter and occlusion, which we plan to analyze next.

5. Conclusion

We presented the definition, inference and construction of a 3D object model. The model consists of a set of parts represented with smooth surface-point densities. Object pose likelihood is defined through the convolution of parts with scene evidence. The ML pose is computed through simulated annealing on a Markov chain whose invariant distribution is proportional to an increasing power of the pose likelihood, yielding an effective balance between exploration and convergence. The learning procedure probabilistically registers and fuses partly overlapping object views and identifies object parts through expectation-maximization. The suitability of our model for pose estimation was demonstrated on cluttered range scenes, using a set of objects of similar shapes; object recognition results competitive with recent generative and discriminative methods were obtained on a publicly available dataset.

Acknowledgments

The authors warmly thank Dr. Georg Biegelbauer and Pr. Markus Vincze for providing us with 3D-scan data. This work was supported by the Belgian National Fund for Scientific Research (FNRS) and the EU Cognitive Systems project PACO-PLUS (IST-FP6-IP-027657).

References

[1] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1):5–43, January 2003. 3

[2] G. Biegelbauer and M. Vincze. Efficient 3D object detection by fitting superquadrics to range image data for robot's object manipulation. In *IEEE International Conference on Robotics and Automation*, 2007. 1, 2, 4

[3] G. Bouchard and B. Triggs. Hierarchical part-based visual object categorization. In *Computer Vision and Pattern Recognition*, volume 1, pages 710–715, 2005. 4

[4] R. J. Campbell and P. J. Flynn. A survey of free-form object representation and recognition techniques. *Comput. Vis. Image Underst.*, 81(2):166–210, 2001. 1

[5] R. Detry, N. Pugeault, and J. Piater. A probabilistic framework for 3D visual object representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2009. 1, 2

[6] R. Fisher. Dispersion on a sphere. In *Proc. Roy. Soc. London Ser. A.*, 1953. 2

[7] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *European Conference on Computer Vision*, 2004. 1

[8] G. Hetzel, B. Leibe, P. Levi, and B. Schiele. 3D object recognition from range images using local feature histograms. In *Computer Vision and Pattern Recognition*, pages 394–399, 2001. 1, 2, 5, 6

[9] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21:433–449, 1999. 1

[10] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. 3

[11] X. Li and I. Guskov. Multi-scale features for approximate alignment of point-based surfaces. In *Eurographics symposium on Geometry processing*, 2005. 2

[12] X. Li and I. Guskov. 3D object recognition from range images using pyramid matching. In *International Conference on Computer Vision*, pages 1–6, 2007. 1, 5, 6

[13] P. Liang and J. S. Todhunter. Representation and recognition of surface shapes in range images: A differential geometry approach. *Computer Vision, Graphics, and Image Processing*, 52(1):78–109, 1990. 2

[14] A. S. Mian, M. Bennamoun, and R. A. Owens. Three-dimensional model-based object recognition and segmentation in cluttered scenes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1584–1601, 2006. 1, 2, 4, 6

[15] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978. 3

[16] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, 1986. 2

[17] F. Solina and R. Bajcsy. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(2):131–147, 1990. 1

[18] E. B. Sudderth. *Graphical models for visual object recognition and tracking*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2006. 2, 3

Anticipative Control of Voluntary Action: Towards a Computational Model

Pascal Haazebroek[1] and Bernhard Hommel[2,*]

[1] Leiden University
Cognitive Psychology Unit & Leiden Institute for Brain and Cognition
Leiden, The Netherlands
[2] Leiden University
Department of Psychology
Cognitive Psychology Unit
Wassenaarseweg 52
2333 AK Leiden, The Netherlands
hommel@fsw.leidenuniv.nl

Abstract. Human action is goal-directed and must thus be guided by anticipations of wanted action effects. How anticipatory action control is possible and how it can emerge from experience is the topic of the ideomotor approach to human action. The approach holds that movements are automatically integrated with representations of their sensory effects, so that reactivating the representation of a wanted effect by "thinking of it" leads to a reactivation of the associated movement. We present a broader theoretical framework of human perception and action control—the Theory of Event Coding (TEC)—that is based on the ideomotor principle, and discuss our recent attempts to implement TEC by means of a computational model (HiTEC) to provide an effective control architecture for artificial systems and cognitive robots.

Human behavior is commonly proactive rather than reactive. That is, people do not await particular stimulus events to trigger certain responses but, rather, carry out planned actions to reach particular goals. Planning an action ahead and carrying it out in a goal-directed fashion requires prediction and anticipation: in order to select an action that is suited to reach a particular goal presupposes knowledge about relationships between actions and effects, that is, about which goals can be realized by what action. Under some circumstances this knowledge might be generated ad hoc. For instance, should your behavior ever make a flight attendant to drop you by parachute in a desert, your previously acquired knowledge may be insufficient to select among reasonable action alternatives, so you need to make ad hoc predictions to find out where to turn to. But fortunately, most of the situations we encounter are much more familiar and, thus, much easier to deal with. We often have a rough idea about what actions may be suitable under a given goal and in a particular context, simply because we have experience: we have had and reached the same or similar goals and acted in the same or similar situations before.

How experience with one's own actions generates knowledge that guides the efficient selection of actions, and how humans carry out voluntary actions in

[*] Corresponding author.

general, was the central issue in ideomotor approaches to human action control. Authors like Lotze (1852), Harless (1861), and James (1890) were interested in the general question of how the mere thought of a particular action goal can eventually lead to the execution of movements that reach that goal in the absence of any conscious access to the responsible motor processes (*executive ignorance*). Key to the theoretical conclusion they came up with was the insight that actions are means to generate perceptions (of wanted outcomes) and that these perceptions can be anticipated. If there would be an associative mechanism that integrates motor processes (m) with representations of the sensory effects they produce (e), and if the emerging association between movements and effect representations would be bidirectional (m←→e), reactivating the representation of the effect by voluntarily "thinking of it" may suffice to reactivate the associated motor processes (e→m). In other words, integrating movements and their sensory consequences provides a knowledge base that allows for selecting actions according to their anticipated outcomes—for anticipative action control that is.

After a flowering period in the second half of the 19th century ideomotor approaches were effectively eliminated from the scientific stage (Prinz, 1987; Stock & Stock, 2004). A major reason for that was the interest of ideomotor theoreticians in conscious experience and the relationship between conscious goal representations and unconscious motor behavior, a topic that did not meet scientific criteria in the eyes of the behaviorist movement gaining power in the beginning of the 20th century (cf., Thorndike, 1913). Starting with an early resurrectional attempt by Greenwald (1970), ideomotor ideas have recently regained scientific credibility and explanatory power however. In their Theory of Event Coding (TEC), Hommel, Müsseler, Aschersleben, and Prinz (2001) have even suggested that the ideomotor principle may represent a firm base on which a comprehensive theory of human perception and anticipatory action control can be built. In the following, we will elaborate on what such a theory may look like. In particular, we will briefly discuss the basic principles and basic assumptions of TEC and then go on to describe our recent attempts to implement these principles and assumptions by means of a computational model of human perception and action control—a model we coined HiTEC (Haazebroek & Hommel, submitted).

1 TEC

The core idea underlying TEC (Hommel et al., 2001) is that perception and action are in some sense the same thing and must therefore be cognitively represented in the same way—the notion of *common coding* (Prinz, 1990). According to the ideomotor principle, action consists in intentionally producing wanted effects, that is, in the execution of motor processes for the sake of creating particular sensory events. In contrast to action, perception is commonly conceived of as the passive registration of sensory input. However, Hommel et al. (2001) argue that this conception is incorrect and misleading, as sensory input is commonly actively produced (Dewey, 1896; Gibson, 1979). For instance, even though visual perception needs light hitting the retina, we actively move our eyes, head, and body to make sure that our retina is hit by the light that is reflecting the most interesting and informative events. That is, we actively search for the information we are interested in and move our receptive surfaces to optimize the intake of that information. This is even more obvious for the tactile sense, as almost nothing would be perceived by touch without systematically moving the sensor surface across the objects of interest. Hence, we perceive by executing motor processes for the sake of creating particular sensory events. Obviously, this is exactly the way we just defined action, which implies that action and perception are one process.

The second central assumption of TEC is that cognitive representations are *composites* of feature codes (Hommel, 2004). Our brain does not represent events through individual codes or neurons but by widely distributed feature networks. For instance, the visual cortex consists of numerous representational maps coding for various visual features, such as color, orientation, shape, or motion (DeYoe & Van Essen, 1988) and similar feature maps have been reported for other modalities. Likewise, action plans are composites of neural networks coding for various action features, such as the direction, force, or distance of manual actions (Hommel & Elsner, 2009). One implication of the assumption that cognitive event representations are composites is that binding operations are necessary to integrate the codes referring to the same event, and another is that different events can be related to, compared with, or confused with each other based on the features they do or do not share. For instance, TEC implies that stimuli and responses can be similar to each other, in the sense that the binding representing the stimulus and the binding representing the response can include the same features, such as location or speed, and can thus prime each other (which for instance explains effects of stimulus-response compatibility) or interact in other ways.

The third main assumption of TEC is that the cognitive representations that underlie perception and action planning code for *distal* but not *proximal* aspects of the represented events (Prinz, 1992). In a nutshell, this means that perceived and produced events are coded in terms of the features of the external event *as external event* (i.e., as objectively or inter-subjectively definable) but not with respect to the specifics of the internal processing, such as retinal or cortical coding characteristics, or particular muscle parameters. This terminology goes back to Heider (1926, 1930), who discussed the problem that our conscious experience refers to objective features of visual objects (the distal attributes), even though the intermediate processing steps of the physical image on the retina and the physiological response to it (the proximal attributes) are not fully determined by the distal attributes. Brunswik (1944) extended this logic to action and pointed out that goal representations refer to distal aspects of the goal event and, thus, do not fully determine the proximal means to achieve it.

To summarize, TEC assumes that perceived events are represented by activating and integrating feature codes—codes that represent the distal features of the event. Given that perceptions are actively produced, these bindings are likely to also include action features, that is, codes that represent the features of the action used to produce that perception. In turn, action plans are integrated bindings of codes representing the distal features of the action. As actions are carried out to create sensory events, action plans also comprise of feature codes referring to these events. In other words, both perceived and produced events are represented by sensorimotor bindings or "event files" (Hommel, 2004). However, not all features of a perceived or a produced event are relevant in a particular context. To account for that, TEC assumes that feature codes are "intentionally weighted" according to the goal or task at hand. For instance, if you are searching for a particular color, or if what matters for your actions is the location of your fingertip, color and location codes would be weighted higher, respectively, and thus affect perception and action planning more strongly. TEC was very helpful in interpreting and integrating available findings in a coherent manner, as well as in stimulating numerous experiments and studies on various topics and perception-action phenomena. However, as Hommel et al. (2001) pointed out, TEC only provides a general framework and the theoretical concepts needed to get a better understanding of higher level perception, action, and their relationship. Deeper insight and theoretical advancement calls for more detail and additional assumptions. To meet this challenge we began developing HiTEC, a computational implementation of TEC's basic principles and assumptions. In the following, we provide a brief overview of the main strategies guiding our implementation, but refer to Haazebroek and Hommel (submitted) for a broader treatment.

2 HITEC

HiTEC (Haazebroek & Hommel, submitted) is an attempt to translate the theoretical framework of TEC (Hommel et al, 2001) into a runnable computational model. Our ambition is to develop a broad, cognitive architecture that can account for a variety of empirical effects related to stimulus-response translation and that can serve as a starting point for a novel control architecture for cognitive robots in the PACO-PLUS project (www.paco-plus.org).

From a modeling perspective TEC provides a number of constraints; some of them enforce structural elements while others impose the existence of certain processes. First, we describe the general structure of HiTEC. Next, we elaborate on the processes operating on this structure, following the two-stage model (Elsner and Hommel, 2001) for the acquisition of voluntary action control. Finally, we discuss how the mechanisms of HiTEC might operate in a real life scenario and show that anticipation plays a crucial role in quickly generating and controlling appropriate responses.

3 HITEC's Structure and Representations

HiTEC is architected as a connectionist network model that uses the basic building blocks of parallel distributed processing (PDP; e.g., McClelland, 1992; Rumelhart, Hinton, & McClelland, 1986). In a PDP network model processing occurs through the interactions of a large number of interconnected elements called units or nodes. Nodes may be organized into higher structures, called modules, each containing a number of nodes. Modules may be part of a larger processing pathway. Pathways may interact in the sense that they can share common modules.

Each node has an activation value indicating local activity. Processing occurs by propagating activity through the network; that is, by propagating activation from one node to the other, via weighted connections. When a connection between two nodes is positively weighted, the connection is excitatory and the nodes will increase each other's activation. When the connection is negatively weighted, it is inhibitory and the nodes will reduce each other's activation. Processing starts when one or more nodes receive some sort of external input. Gradually, node activations will rise and propagate through the network while interactions between nodes control the flow of processing. Some nodes are designated output nodes. When activations of these nodes reach a certain threshold (or when the time allowed for processing has passed), the network is said to produce the corresponding output(s).

In HiTEC, the elementary units are codes. As illustrated in Figure 1, codes are organized into three main systems: the sensory system, the motor system and the common coding system. Each system will now be discussed in more detail.

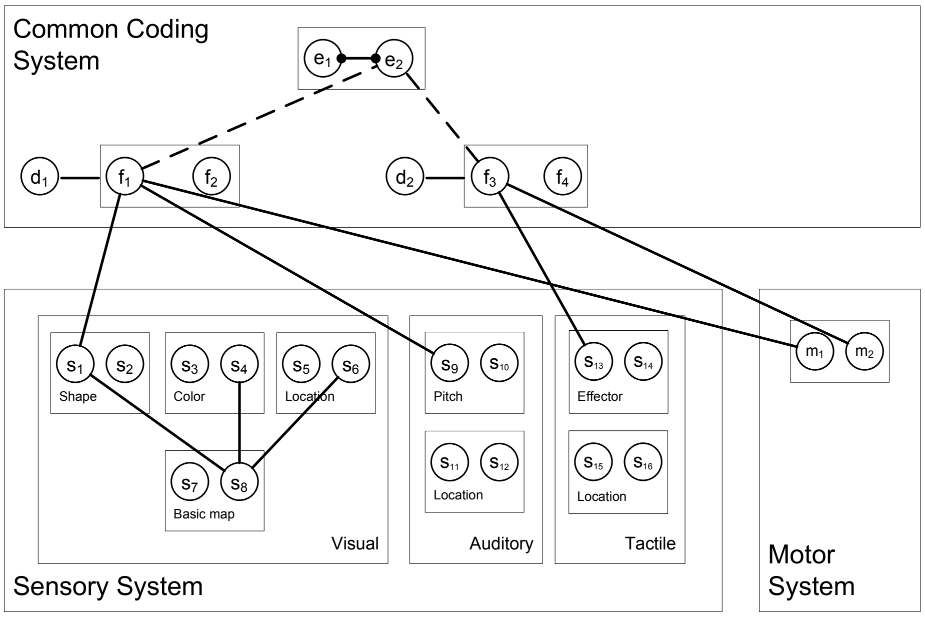


Fig. 1. General architecture of HiTEC

3.1 Sensory System

As already mentioned, the primate brain encodes perceived objects in a distributed fashion: different features are processed and represented across different cortical maps (e.g., Cowey, 1985; DeYoe & Van Essen, 1988). In HiTEC, different modalities (e.g., visual, auditory) and different dimensions within each modality (e.g., visual color and shape, auditory location and pitch) are processed and represented in different sensory maps. Each sensory map is a module containing a number of sensory codes that are responsive to specific sensory features (e.g., a specific color or a specific pitch). Note that Figure 1, shows only two sensory codes per map for clarity.

In the visual brain, there are two major parallel pathways (Milner & Goodale, 1995) that follow a common preliminary basic feature analysis step. The ventral pathway is seen as crucial for object recognition and consists of a hierarchy of sensory maps coding for increasingly complex features (from short line segments in the lower maps to complex shapes in higher maps) and increasingly large receptive field (from a small part of the retina in the lower maps to anywhere on the retina in higher maps). The second pathway, the dorsal pathway, is seen as crucial for action guidance as it loses color and shape information but retains information about contrast, location of objects, and other action-related features.

In HiTEC, a common visual sensory map codes for basic visual parts of perceptual events. This common basic map projects to both the ventral and the dorsal pathways. The ventral pathway consists of sensory maps coding for combinations (such as more specific shapes) or abstractions (e.g., object color). The dorsal pathway is currently simply a sensory map coding for visual location—to be extended for processing other action-related features in a later version of HiTEC.

Distributed processing allows a system to dramatically increase its representational capacity as it no longer requires each combination of features to have its own dedicated representational structure but can rather encode a specific combination on demand in terms of activating a collection of constituting feature structures. On the downside, in typical scenarios, this inevitably results in binding problems (Treisman, 1996). For instance, when multiple objects are perceived and they are both represented in terms of activating the structures coding for their constituting features, how to tell which feature belongs to which object? This clearly calls for an integration mechanism that can tell them apart.

Recent studies in the visual modality have shown that this problem can, partly, be solved by employing local interactions between feed-forward and feed-back processes in the ventral and dorsal pathways (Van der Velde & De Kamps, 2001). It is true that higher ventral sensory maps do not contain information on location and that higher dorsal sensory maps do not contain information on object shape or color, but these pathways can interact using the common basic visual feature map as a visual blackboard (Van der Velde, De Kamps, & Van der Voort van der Kleij, 2004). For instance: when a specific color is activated in a higher sensory map, it can feed back activation to lower sensory maps, thereby modulating the activity of these sensory codes in a way that those codes that code for simple parts of this color are enhanced. This can modulate the processing in the dorsal pathway as well resulting in enhanced activation of those codes in the location map that code for the location(s) of objects of the specified color.

This principle also works the other way round: activating a specific location code in the location map can modulate the sensory codes in the lower sensory maps that code for simple parts at this location. This can modulate the processing in the ventral pathway, resulting in enhanced activation of the more complex or abstract features of the object at the specified location. In HiTEC, this is the way the visual sensory system can be made to enhance the processing of objects with specific features or on a specific location. For now, we assume the following sensory maps in the HiTEC architecture: visual basic features map, visual color map, visual shape map, visual location map, auditory pitch map, auditory location map, tactile effector (i.e., hands or feet) map and tactile location map.

3.2 Motor System

The motor system contains motor codes, referring to proximal aspects of movements. Motor codes can also be organized in maps, following empirical evidence that suggests distributed representations at different cortical locations in the motor domain (e.g., Andersen, 1988; Colby 1998). For example, cortical maps can be related to effector (e.g., eye, hand, arm, foot) or movement type (e.g., grasping, pointing). It makes sense to assume that there is some sort of hierarchical structure as well in motor coding. However, in the present version of HiTEC, we consider only one basic motor map with a set of motor codes. As our modeling efforts in HiTEC evolve, its motor system may be extended further.

It is clear that motor codes, even when structured in multiple maps, can only specify a rough outline of the motor action to be performed as some parameters depend strongly on the environment. For instance, when grasping an object, the actual object location is not represented by a motor code (this would lead to an explosion of the number of necessary motor codes, even for a very limited set of actions). So it makes sense to interpret a motor program as a blueprint of a motor action that needs to be filled in with this specific, on line, information, much like the schemas put forward by Schmidt (1975) and Glover (2004). In our discussion of HiTEC processes we will discuss this issue in more detail.

3.3 Common Coding System

According to TEC both perceived events and action generated events are coded in one common representational domain (Hommel et al, 2001). In HiTEC, this domain is the common coding system that contains common feature codes. Feature codes refer to distal features of objects, people and events in the environment. Example features are distance, size and location, but on a distal, descriptive level, as opposed to the proximal features as coded by the sensory codes and motor codes.

Feature codes may be associated to both sensory codes and motor codes and are therefore truly sensorimotor. They can combine information from different modalities and are in principle unlimited in number. Feature codes are not given but they evolve and change. In HiTEC simulations, however, we usually assume a set of feature codes to be present initially, to bootstrap the process of extracting sensorimotor regularities in interactions with the environment.

Feature codes are contained in feature dimensions. As feature dimensions may be enhanced as a whole, for each dimension an additional dimension code is added that is associated with each feature code within this dimension. Activating this code will spread activation towards all feature codes within this dimension, making them more sensitive to stimulation originating from sensory codes.

3.4 Associations

In HiTEC, codes can become associated, both for short term and for long term. Short term associations between feature codes reflect that these codes 'belong together in the current task or context’ and their binding is actively maintained in working memory. In Figure 1, these temporary bindings are depicted as dashed lines. Long term associations can be interpreted as learned connections reflecting prior experience. For now, we do not differentiate between episodic and semantic memory—even though later versions are planned to distinguish between a “literal” episodic memory that stores event files (see below) and a semantic memory that stores rules abstracted from episodic memory (O'Reilly & Norman, 2002). At present, both types of experience are modeled as long term associations between (any kind of) codes and are depicted as solid lines in Figure 1.

3.5 Event file

Another central concept in the theory of event coding is the event file (Hommel, 2004). In HiTEC, the event file is modeled as a structure that temporarily associates to feature codes that 'belong together in the current context' in working memory. The event file serves both the perception of a stimulus as well as the planning of an action. Event files can compete with other event files.

4 HITEC's Processes

How do associations between codes come to be? What mechanisms result of their interactions? And how do these mechanisms give rise to anticipation based, voluntary action control? Elsner and Hommel (2001) proposed a two-stage model for the acquisition of voluntary action control. At the first stage, the cognitive system observes and learns regularities in motor actions and their effects. At the second stage, the system uses the acquired knowledge of these regularities to select and control its actions. For both stages, we now discuss in detail how processes take place in the HiTEC architecture. Next, we discuss some additional process related aspects of the architecture.

Stage 1: Acquiring Action-Effect Associations

The framework of event coding assumes that feature codes are grounded representations as they are derived by abstracting regularities in activations of sensory codes. However, the associations between feature codes and motor codes actually signify a slightly different relation: feature codes encode the (distal) perceptual effect of the action that is executed by activating the motor codes. Following the ideomotor principle, the cognitive system has no innate knowledge of the actual motor action following the activation of a certain motor code. Rather, motor codes need to become associated with their perceptual action effects so that by anticipating these effects, activation can propagate via these associations to those motor codes that actually execute the corresponding movement.

Infants typically start off with a behavioral repertoire based on stimulus-response (SR) reflexes (Piaget, 1952). As the infant exhibits these stimulus-response reflexes, as well as random behaviors (e.g., motor babbling), its cognitive system learns the accompanying response-perceptual effect (RE) regularities that will serve as some sort of database of 'what action achieves what environmental effect'. Following Hommel (1996), we assume that any perceivable action effect is automatically coded and integrated into an action concept, which is, in the HiTEC architecture, an event file consisting of feature codes. Although all effects of an action become integrated automatically, intentional processes do affect the relative weighting of integrated action effects—TEC's intentional-weighting principle.

Taken together, action – effect acquisition is modeled in HiTEC as follows: motor codes m_i are activated, either because of some already existing associations or simply because of network noise. This leads to a change in the environment (e.g., the left hand suddenly touches a cup) which is picked up by sensory codes s_i. Activation propagates from sensory codes towards feature codes f_i. And eventually, these feature codes are integrated into an event file e_i which acts as an action concept. Subsequently, the cognitive system learns associations between the feature codes f_i belonging to this action concept and the motor code m_i that just led to the executed motor action. Crucially, task context can influence the learning of action effects. Not by selecting which effects are associated but by weighting the different effect features. Nonetheless, this is an interactive process that does not exclude unintended but utterly salient action effects to become involved in strong associations as well.

Stage 2: Using Action Effect Associations

Once associations between motor codes and feature codes exist, they can be used to select and plan voluntary actions. Thus, by anticipating desired action effects, feature codes become active. Now, by integrating the feature codes into an action concept, the system can treat the features as constituting a desired state and propagate their activation towards associated motor codes. Crucially, anticipating certain features needs integration to tell them apart from the features that code for the currently observed environment. Once integrated, the system has ‘a lock’ on these features and can use these features to select the right motor action.

Initially, multiple motor codes m_i may become active as they typically fan out associations to multiple feature codes f_i. However, some motor codes will have more associated features that are also part of the active action concept and some of the m_i - f_i associations may be stronger than others. Taken together, the network will – in PDP fashion – converge towards one strongly activated motor code m_i which will lead to the selection of that motor action.

In addition to the mere selection of a motor action, feature codes also form the actual action plan that specifies (in distal terms) how the action should be executed: namely, in such a way the intended action effects are realized. By using anticipated action effects to choose an action, the action actually is selected because the cognitive system intended this, not because of a reflex to some external stimulus. Thus, in HiTEC, using anticipation is the key to voluntary action.

4.1 Task Context

Task context can modulate both action-effect learning and the usage of these links. This can help focus processing to action alternatives that ‘make sense’ in the current context. In real life this is necessary as the action alternatives are often rather unconstrained. Task context comes in different forms. One is the overall environment, the scene context in which the interaction takes place. The cognitive system may just have seen other objects in the room, or the room itself, and feature codes that code for aspects of this context may still have some activation. This can, in principle, influence action selection. As episodic and semantic memory links exist as well, this influence may also be less salient: the presence of a certain object might recall memories of previous encounters or similar contexts that influence action selection in the current task.

A task can also be very specific, as given by a tutor or instructor in terms of a verbal description. In HiTEC, it is assumed that feature codes can be activated by means of verbal labels. Thus, when a verbal task is given, this could directly activate feature codes. The cognitive system integrates these codes into an event file that is actively maintained in working memory. For example, when approached with several options to respond differently to, different event files e_i are created for the different options. Due to the mutual inhibitory links between event files, they will compete with each other. Because of the efficiency the cognitive system can now display, one could state that a cognitive reflex has been prepared (Hommel, 2000) that anticipates certain stimuli features. The moment these features are actually perceived, the reflex 'fires' and - by propagating activation to event codes and subsequently to other feature codes - quickly anticipates the correct action effects, which results in the selection and execution of the correct motor action.

4.2 Online vs. Offline Processing

In HiTEC, action selection and action planning are interwoven, but on a distal feature level. This leaves out the necessity of coding every minute detail of the action, but restricts action planning to a ballpark idea of the movement. Still, a lot has to be filled in by on line information. Currently, this falls outside the scope of HiTEC, but one could imagine that by activating distal features, the proximal sensory codes can be top down moderated to 'focus their attention' towards specific aspects of the environment (e.g., visual object location), see Hommel (in press). In addition, actions need still not to be completely specified in advance, as they are monitored and adjusted while they are performed—which in humans seems to be the major purpose of dorsal pathways (Milner & Goodale, 1995)

4.3 Action Monitoring

The anticipated action effects are a trigger for action selection, but also form an expectation of the perceptual outcome of the action. Differences between this expectation and reality lead to adjusting the action on a lower sensorimotor level than is currently modeled in HiTEC. What matters now, is that the feature codes are interacting with the sensory codes, making sure that the generated perception is within the set parameters, as determined by the expected action outcome. If this is not (well enough) the case, the action should be adjusted.

However, when a discrepancy of this expectation drastically exceeds 'adjustment thresholds', it may actually trigger action effect learning (stage 1). Apparently, the action-effect associations were unable to deliver an apt expectation of the actual outcome. Thus, anticipating the desired outcome falsely led to the execution of this action. This may trigger the system to modify these associations, so that the motor codes become associated with the correct action effect features.

Crucially, having anticipations serve as expectations, the system is not forced into two distinct operating modes (learning vs. testing). With anticipation as retrieval cue for action selection and as expectation of the action outcome, the system has the means to self-regulate its learning by making use of the discrepancy between actual effects and these anticipations.

5 Model Implementation

The HiTEC model is implemented using neural network simulation software that facilitates the specification and simulation of interactive networks. In interactive networks, connections are bidirectional and the processing of any single input occurs dynamically during a number of cycles. Each cycle, the network is gradually updated by changing the activation of each node as a result of its interactions with other nodes.

5.1 Code Dynamics

HiTEC aims at a biologically realistic implementation of network dynamics. In the human brain, local interactions between neurons are largely random, but when looking at groups of neurons (i.e., neuron populations) their average activation can be described using mean field approximation equations (Wilson and Cowan, 1972). In HiTEC, a single code is considered to be represented by a neuron population. Its dynamics can therefore be described using differential equations such as:

$$\frac{dA}{dt} = -A + \sum_{k} w_k F(A_k) + N$$

This equation states that the change in activation *A* of a code is a result of a decay term and the weighted sum of the outputs of those nodes *k* that it connects to. Also, each node receives additional random noise input *N*. Node output is computed using an activation function *F(A)* that translates node activation into its output as governed by the following logistic function:

$$F(A) = \frac{1}{1 + \exp(-A)}$$

The simulator uses numerical integration to determine the change of activation for each node in each cycle.

5.2 Codes

Currently, in our simulations we hard code all sensory codes including their receptive field specification (e.g., whether a code is responsive to a red or a blue color). Also, feature codes are assumed to exist, as well as all connections between sensory codes and feature codes reflecting prior experience with sensory regularities. In the future it may become an interesting endeavor to learn the grounding of feature codes in terms of proximal sensory codes, possibly by means of self organizing map methods that can be moderated by HiTEC processes (e.g., failing to predict an action outcome may signal relevant novelty and moderate the creation or update of a feature code). Also, for now, we assume a limited set of motor programs that are simply represented by fixed motor codes. Thus, in simulations we currently focus on the interactions between perception and action and how task context influence these interactions, rather than on the grounding of codes per se.

5.3 Action-Effect Learning

Learning action effects is reflected by creating long term connections between feature codes and motor codes. This is currently done by simple associative, Hebbian learning, as described by the following equation:

$$\frac{dw_{ij}}{dt} = \gamma A_j (A_i - w_{ij})$$

Thus, the change of the connection strength is determined by the activation of the nodes *i* and *j* that are connected. This way, feature codes that were activated more strongly will become more strongly connected to the motor code that caused the perceptual effect. Surely, this type of learning is known to be limited but serves our current purposes.

5.4 Short Term Associations and Event File Competition

Crucial in HiTEC is the short term memory component. A task instruction is represented using short term connections between feature codes and event files. In the current set up, an event file is simply a node that is created on demand, as a result of the task instruction, and temporarily connects to those feature codes that were activated by the task instruction (i.e., via verbal labels). An event file has an enhanced baseline activation, reflecting its task relevance. Moreover, event files compete with each other by means of lateral inhibition (i.e., they are interconnected with negative connections) resulting in a winner-take-all mechanism: as activation gradually propagates from feature codes to event files (and back), their activation changes as well. Due to the lateral inhibition, only one event file will stand as the 'winner', while weakening the other event files. This results in selective activation at the feature code level and subsequently in action selection at the motor code level.

5.5 Related Work

We must note that we do‸advertise the associative learning method used in HiTEC as a competitive alternative to highly specialized machine learning techniques that are traditionally used in classification tasks (e.g., Hiddden Markov Models, Support Vector Machines et cetera) or reward based learning tasks (e.g., Reinforcement learning, Q-learning et cetera). However, we do focus on the context of learning: the interplay between (the coding of) task context and action effect anticipation and perception triggers and mediates learning. In particular, we stress that the cognitive system employs anticipation as reflection of both its learned knowledge so far and its interpretation of the current context. Anticipation can subsequently mediate learning by influencing which features engage in learning (and even further: what features to look for in the sensory input) and how strongly these features may be associated to motor codes, thereby constraining whatever (machine) learning technique used to actually create or change the associations.

Moreover, failing to correctly anticipate an action effect may be a major trigger to update the learned knowledge. In the future we may add this as a reinforcement learning component that drives on biologically plausible reward mechanisms (e.g., dopamine moderated learning).

Finally, we stress that although simulations may be set up in terms of instruction, train and test phases, the HiTEC model itself does not artificially 'switch' between two modes of operation: learning occurs on line as a result of perceiving action effects.

6 Examplary Scenario: Responding to Traffic Lights

In order to clarify the co-operation of the different processes and mechanisms in HiTEC on a functional level, the following example real life scenario is presented: learning to respond to traffic lights. In this example, s_i denotes sensory codes, f_i denotes feature codes and m_i denotes motor codes in the HiTEC architecture. Figure 2 shows a scenario-specific version of the HiTEC architecture.

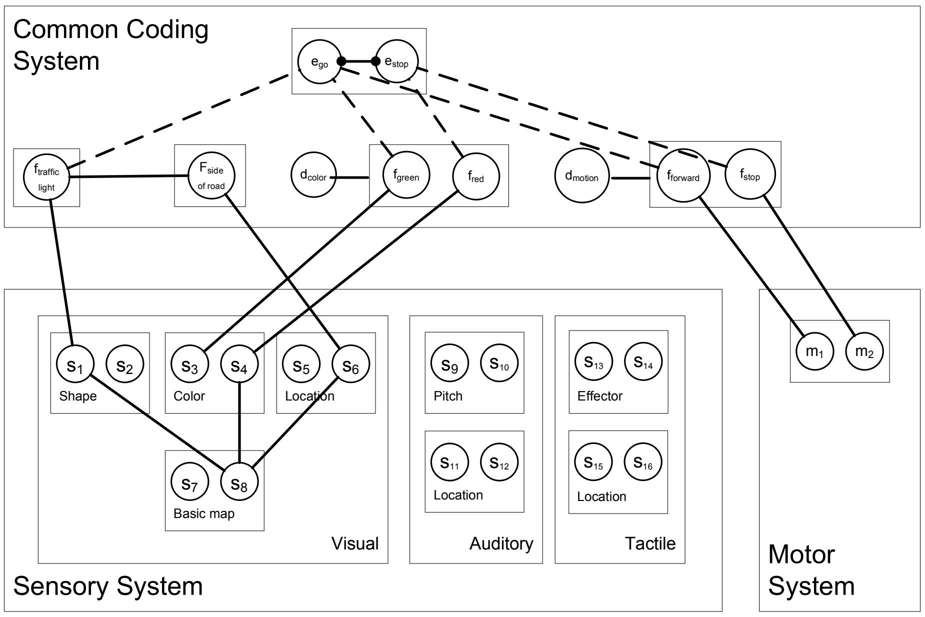


Fig. 2. Learning to respond to traffic lights in HiTEC

6.1 Action Effect Acquisition

Let's say you are a student driver who has never paid attention to the front seat before and this is your first driving lesson. You climb behind the steering wheel and place your feet above the pedals. Now, the instructor starts the car for you and you get the chance of playing around with the pedals. After a while, you get the hang of it: it seems that pressing the right pedal results in a forward movement of the car, and pressing the left one puts the car on hold.

From a HiTEC perspective, you just have tried some motor codes and learned that m_1 (pressing the gas pedal) results in a forward motion, coded by $f_{forward}$ and m_2 in standing still, coded by f_{stop}. In other words: you acquired these particular action-effect associations. Note that we assume that you have been able to walk before, so it is fair to say that $f_{forward}$ and f_{stop} are already present as feature codes in your common coding system.

6.2 Using Action Effect Associations

Now, in your next lesson you actually need to take cross roads. The instructor tells you to pay attention to these colored lights next to the road. When the red light is on, you should stop, and when the green light is on, you can go forward.

In HiTEC, this verbal instruction is modeled as creating two event files that hold short term associations in working memory: $e_{stop\ for\ red\ light}$ for the 'stop' condition, and $e_{go\ at\ green\ light}$ for the 'forward' condition. The event file $e_{stop\ for\ red\ light}$ contains bindings of feature codes f_{red}, $f_{traffic\ light}$, f_{stop} and the event file $e_{go\ at\ green\ light}$ relates to the feature codes f_{green}, $f_{traffic\ light}$, $f_{forward}$.

These event files are activated and their activation spreads to their associated feature codes which will become increasingly receptive for interaction with related sensory codes. In addition to the specific features, the feature dimensions these features are contained in (d_{color}, d_{motion}) are weighted as well. The anticipation of traffic lights also serves as a retrieval cue for prior experience with looking at traffic lights. As traffic lights typically stand at the side of the road, one could expect associations between $f_{traffic\ light}$ and $f_{side\ of\ road}$ to exist in episodic or semantic memory. Consequently, anticipating a traffic light activates $f_{traffic\ light}$ and propagates activation automatically towards $f_{side\ of\ road}$, which makes the system more sensitive to objects located on the side of the road.

Ok, there it goes... you start to drive around, take some turns, and there it is… your very first cross road with traffic lights!

Now, from a HiTEC perspective, the following takes place: the visual scene consists of a plethora of objects, like road signs, other cars, houses and scenery, and of a cross road with traffic lights at the side. The sensory system encodes the registration of these objects by activating the codes in the sensory maps. This leads to the classical binding problem: multiple shapes are registered, multiple colors and multiple locations. However, we now have a top down 'special interest' for traffic lights. As mentioned above, this has resulted in increased sensitivity of the $f_{traffic\ light}$ feature code, that now receives some external stimulation from related sensory codes. Also, from prior experience we look more closely at $f_{side\ of\ road}$ locations in the sensory location maps.

The interaction between this top down sensitivity and the bottom up external stimulation results in an interactive process where the sensory system uses feedback signals to the lower level visual maps where local interactions result in higher activation of those sensory codes that code for properties of the traffic light, including its color. In the visual map for object color, the traffic light color will be more enhanced than colors relating other objects. On the feature code level, the color dimension already was enhanced because of the anticipation of features in the d_{color} dimension, resulting in fast detection of f_{red} or f_{green}.

Meanwhile, the event files $e_{stop\ for\ red\ light}$ and $e_{stop\ for\ red\ light}$ are still in competition. When the sensory system collects the evidence, activation propagates towards feature codes and event codes, quickly converging into a state that where either $f_{forward}$ or f_{stop}

is activated more strongly than the other. This activation is propagated towards the motor codes m_1 or m_2 via associations learned in your first drivers lesson. This results in the selection and execution of the correct motor action.

It is clear that by preparing the cognitive system for perceiving a traffic light color and producing a stop-or-go action allows the system to effectively attend its resources to the crucial sensory input and already pre-anticipate the possible action outcome. This way, upon perceiving the actual traffic light color, the system can quickly respond with the correct motor action.

Luckily, for your safety and that of all your fellow drivers on the road, practicing this task long enough will also result in long term memory bindings between f_{red}, $f_{traffic\ light}$ and f_{stop} that will also be retrieved during action selection and bias you towards pressing the brake pedal, even when no instructor is sitting next to you.

7 Conclusions

We have introduced HiTEC's three main modules: the sensory system, the motor system, and the emergent common coding system. These systems interact with each other. In the common coding system anticipations are formed that have a variety of uses in the architecture, allowing the system to be more flexible and adaptive. In action selection, anticipation acts as a rich retrieval cue for associated motor programs. At the same time, forming this anticipation reflects the specification of an action plan that can be used during action execution.

One of the drawbacks of creating anticipations is that it might not be worth the costs (Butz & Pezzulo, 2008). However, from a real life scenario perspective, the number of possible action alternatives is enormous. Creating anticipations at a distal level seems as a necessity to constrain the system in its actions to select from. Doing this, as we propose in HiTEC, not only aids action selection but also delivers the rudimentary action plan at the same time.

Another concern often mentioned is the inaccuracy of predictions. Following the framework of event coding, events – including action plans – are coded in distal terms that abstract away from the proximal sensory values. Only inaccuracies on the distal level could disturb the use of anticipations in action selection and planning. The feature codes on this distal level are based on sensorimotor regularities that are stable over time. Thus minor inaccuracies in sensors should be relatively easily overcome.

Actions are usually selected and planned in a task context. When forced with different behavioral alternatives to choose from, multiple anticipations of features are created and compete with each other. When features are actually perceived, anticipatory activation quickly propagates to the correct action effects, which results in the selection and execution of the correct motor action.

In action monitoring, anticipation serves as the representation of expected and desired action effects that helps adjusting the movement during action execution. In action evaluation, this expectation acts as a set of criteria for success of the action. If the actual action effect can no longer – on a lower sensorimotor level - be adjusted to fulfill the expected action effect, the existing action-effect associations are considered insufficient and learning is triggered. During action-effect learning, anticipation also may weight the different action effect features in the automatic integration into action concepts, influencing the action-effect association weights.

In conclusion, anticipation plays a crucial role in virtually all aspects of action control within the HiTEC architecture. Just as it does in real life.

Acknowledgments

Support for this research by the European Commission (PACO-PLUS, IST-FP6-IP-027657) is gratefully acknowledged.

References

1. Andersen, R.A.: The neurobiological basis of spatial cognition: Role of the parietal lobe. In: Stiles-Davis, J., Krtichivsky, M., Belugi, U. (eds.) Spatial cognition: Brain bases and development. Erlbaum, Mahwah (1988)
2. Brunswik, E.: Distal focussing of perception: Size constancy in a representative sample of situations. Psychological Monographs 56(1) (1944)
3. Butz, M.V., Pezzulo, G.: Benefits of Anticipations in Cognitive Agents. In: Pezzulo, G., Butz, M.V., Castelfranchi, C., Falcone, R. (eds.) The Challenge of Anticipation. LNCS, vol. 5225, pp. 45–62. Springer, Heidelberg (2008)
4. Colby, C.L.: Action-oriented spatial reference frames in the cortex. Neuron 20, 15–20 (1998)
5. Cowey, A.: Aspects of cortical organization related to selective attention and selective impairments of visual perception: A tutorial review. In: Poster, M.I., Marin, O.S.M. (eds.) Attention and performance XI, pp. 41–62. Erlbaum, Hillsdale (1985)
6. Dewey, J.: The reflex arc concept in psychology. Psychological Review 3, 357–370 (1896)
7. DeYoe, E.A., Van Essen, D.C.: Concurrent processing streams in monkey visual cortex. Trends in Neuroscience 11, 219–226 (1988)
8. Elsner, B., Hommel, B.: Effect anticipation and action control. Journal of Experimental Psychology: Human Perception and Performance 27 (2001)
9. Gibson, J.J.: The ecological approach to visual perception. Houghton Mifflin, Boston (1979)
10. Glover, S.: Separate visual representations in the planning and control of action. Behavioral and Brain Sciences 27, 3–24 (2004)
11. Greenwald, A.: Sensory feedback mechanisms in performance control: With special reference to the ideomotor mechanism. Psychological Review 77, 73–99 (1970)
12. Harless, E.: Der Apparat des Willens. Zeitschrift fuer Philosophie und philosophische Kritik 38, 50–73 (1861)
13. Haazebroek, P., Hommel, B.: HiTEC: A computational model of the interaction between perception and action (submitted)
14. Heider, F.: Thing and medium. Psychological Issues, Monograph 3 (original work published 1959) (1926/1959)
15. Heider, F.: The function of the perceptual system. Psychological Issues, 1959, Monograph, 371–394 (1930/1959) (original work published 1930)
16. Hommel, B.: The cognitive representation of action: Automatic integration of perceived action effects. Psychological Research 59, 176–186 (1996)
17. Hommel, B.: The prepared reflex: Automaticity and control in stimulus-response translation. In: Monsell, S., Driver, J. (eds.) Control of cognitive processes: Attention and performance XVIII, pp. 247–273. MIT Press, Cambridge (2000)

18. Hommel, B.: Event files: Feature binding in and across perception and action. Trends in Cognitive Sciences 8, 494–500 (2004)
19. Hommel, B.: Grounding attention in action control: The intentional control of selection. In: Bruya, B.J. (ed.) Effortless attention: A new perspective in the cognitive science of attention and action. MIT Press, Cambridge
20. Hommel, B., Elsner, B.: Acquisition, representation, and control of action. In: Morsella, E., Bargh, J.A., Gollwitzer, P.M. (eds.) Oxford handbook of human action, pp. 371–398. Oxford University Press, New York (2009)
21. Hommel, B., Müsseler, J., Aschersleben, G., Prinz, W.: The Theory of Event Coding (TEC): A framework for perception and action planning. Behavioral and Brain Sciences 24, 849–937 (2001)
22. James, W.: The principles of psychology. Dover Publications, New York (1890)
23. Lotze, R.H.: Medicinische Psychologie oder die Physiologie der Seele. Weidmann'sche Buchhandlung, Leipzig (1852)
24. McClelland, J.L.: Toward a theory of information processing in graded, random, and interactive networks. In: Meyer, D.E., Kornblum, S. (eds.) Attention and performance XIV: Synergies in experimental psychology, artificial intelligence and cognitive neuroscience – A Silver Jubilee Volume. MIT Press, Cambridge (1992)
25. Milner, A.D., Goodale, M.A.: The visual brain in action. Oxford University Press, Oxford (1995)
26. O'Reilly, R.C., Norman, K.A.: Hippocampal and neocortical contributions to memory: Advances in the complementary learning systems framework. Trends in Cognitive Sciences 6, 505–510 (2002)
27. Piaget, J.: The origins of intelligence in childhood. International Universities Press (1952)
28. Prinz, W.: Ideo-motor action. In: Heuer, H., Sanders, A.F. (eds.) Perspectives on perception and action. Erlbaum, Hillsdale (1987)
29. Prinz, W.: A common coding approach to perception and action. In: Neumann, O., Prinz, W. (eds.) Relationships between perception and action, pp. 167–201. Springer, Berlin (1990)
30. Prinz, W.: Why don't we perceive our brain states? European Journal of Cognitive Psychology 4, 1–20 (1992)
31. Rumelhart, D.E., Hinton, G.E., McClelland, J.L.: A general framework for parallel distributed processing. In: Rumelhart, D.E., McClelland, J.L., The PDP Research Group (eds.) Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1, pp. 45–76. MIT Press, Cambridge (1986)
32. Schmidt, R.A.: A schema theory of discrete motor skill learning. Psychological Review 82, 225–260 (1975)
33. Stock, A., Stock, C.: A short history of ideo-motor action. Psychological Research 68, 176–188 (2004)
34. Thorndike, E.L.: Ideo-motor action. Psychological Review 20, 91–106 (1913)
35. Treisman, A.: The binding problem. Current Opinion in Neurobiology 6, 171–178 (1996)
36. Van der Velde, F., De Kamps, M.: From knowing what to knowing where: Modeling object-based attention with feedback disinhibition of activation. Journal of Cognitive Neuroscience 13(4), 479–491 (2001)
37. Van der Velde, F., De Kamps, M., Van der Voort van der Kleij, G.: Clam: Closed-loop attention model for visual search. Neurocomputing 58-60, 607–612 (2004)
38. Wilson, H., Cowan, J.: Excitatory and inhibitory interactions in localized populations of model neurons. Biophysics Journal 12, 1–24 (1972)

Contents

Grasping by Parts: Robot Grasp Generation from 3D Box Primitives

Kai Huebner Danica Kragic

Computer Vision and Active Perception Lab, KTH, Stockholm, Sweden, {khubner,danik}@kth.se

Abstract

In this paper, we continue our previous work on shape approximation by box primitives for the goal of simple and efficient grasping, and extend it with a more thorough investigation of methods and robot experiments. The contributions of work presented here are twofold: in terms of shape approximation, we provide an updated algorithm for a 3D box primitive representation to identify object parts from 3D point clouds. We motivate and evaluate this choice particularly toward the task of grasping. As a contribution in the field of grasping, we additionally provide a grasp hypothesis generation framework that utilizes the chosen box presentation in a flexible manner.

keywords: grasping, 3D shape approximation, object part representation

1 INTRODUCTION

Robot grasping capabilities are essential for perceiving, interpreting and acting in arbitrary and dynamic environments. While classical computer vision and visual interpretation of scenes focus on the robot's internal representation of the world rather passively, robot grasping capabilities are needed to actively execute tasks, modify scenarios and thereby reach versatile goals.

In robotic object grasping there has been a lot of effort during the past few decades (e.g. see Siciliano and Khatib [1] for a survey). However, the existing artificial systems performing grasping and manipulation of objects are still far away from closely emulating the human perception-action system. Current robotic systems dealing with object grasping and manipulation rarely take into account task dependency, planning or exception handling, especially when the whole eye-hand coordination problem is considered. On the processing side, it has been widely recognized that high-level task-related grasp planning is difficult due to the large search space resulting from all possible hand configurations, grasp types, and object properties that occur in realistic settings. Innovative work in this field included kinematic constraints of the hand in order to prune the search space, e.g. [2, 3]. The most common way to approach the problem has been the model-based approach. Different grasp-related components such as objects, surfaces, contacts, forces, etc., are modeled according to very specific physical laws assuming a good knowledge of the environment. Thus, the research has mainly focused on (i) *grasp analysis*, i.e. the study of the physical properties of a given grasp [4, 5], and (ii) *grasp synthesis*, the computation of grasps that meet certain pre-defined properties [3, 6, 7, 8, 9, 10]. Unfortunately, these approaches have failed to deliver practical implementations for different platforms independent of the hardware.

Early work on contact-level grasp synthesis focused mainly on finding a fixed number of contact locations without regarding hand geometry [6]. Considering specifically object manipulation tasks, the work on automatic grasp synthesis and planning is of significant relevance [3, 7, 10, 11]. In these approaches, finger contact locations, forces and grasp wrench spaces can be simulated. Different criterions can be defined to rate grasp configurations, e.g. force closure, dexterity, equilibrium, stability and dynamic behavior [10]. However, the dependency on a-priori known or dense and detailed object models is apparent. Miller et al. [3] therefore proposed grasp planning on simple shape primitives like spheres, cylinders and cones, clearly demanding a pre-classification of object shape. Dependent on the primitive shape, one can test several grasp configurations for their static stability. This work was continued by Goldfeder *et al.* [12], using superquadrics as more sophisticated shape primitives. Ekvall and Kragic [13] showed how a robot system can learn grasping by human demonstration using a grasp experience database. The human grasp is recognized with the help of a magnetic tracking system and mapped to the kinematics of the robot hand using a predefined lookup-table. Other than in [3], the system can distinguish between ten different human grasps, adapted from Cutkosky's grasp taxonomy [14]. The grasp controller takes into account not only the object pose and the kind of grasp, but also the approach strategy of the human demonstrator.

Thus, an important question is: how can robots be equipped with capabilities of gathering and interpreting the necessary information for novel tasks through interaction with the environment, based on minimal prior knowledge? In order to answer this question and overcome its difficulties, machine vision has been proposed as a solution to obtain the information about object shapes, or contact information to explore the object. Another trend has focused on machine learning approaches to determine the relevant features indicating a successful grasp [7, 15]. Finally, there have been efforts to use human demonstrations for learning grasp tasks. Problematically, these approaches also commonly consider grasps as a fixed number of contact locations without any regard of hand geometry and hand kinematics [4]. An alternative paradigm, often motivated by studies on human grasping, is the so-called *knowledge-based* approach. It tries to simplify the visual grasp planning problem by reasoning on a more symbolic level. In this paradigm, object shapes are often described using shape primitives, like constellations of cubes or ellipsoids. Grasp prototypes are defined in terms of purposeful hand pre-shapes, e.g. power-grasp or pinch-grasp, and planning and selection of grasps is made according to programmed decision rules. Taking into account both hand kinematics and a-priori knowledge about the feasible grasps has been acknowledged as a more flexible and natural approach towards automatic grasp planning [3]. It is obvious that knowledge about the object shape and task is important for grasp planning [16]. This interplay of 3D shape and robot grasping is the main research problem considered in this paper.

Previous Work, Paper Structure and Contributions. In regard to shape approximation, we presented a bounding box decomposition approach for arbitrary object shape approximation and robot grasping in [17]. The initial technique based on Minimum Volume Bounding Boxes from 3D point clouds proposed in this work will be revisited in Section 2. We have further improved the decomposition algorithm to be more robust under influence of noise and clutter. The new technique will be presented in Section 2.3. The content of that section holds the box decomposition itself, leading from an arbitrary point cloud to a constellation of 3D boxes. In Section 3, we will summarize and discuss how we continue with such a box constellation for the purpose of grasping. The basic ideas were introduced in [18]: in the work presented here, we decribe an improved algorithm and extend it with additional detail and experiments. In Section 4, we present an experiment demonstrating the framework capabilities, and then conclude our work in Section 5. We first start with an outline of our system, sorting earlier work and linking it to the primary focus of this paper.

The contributions of work presented here are twofold: in terms of shape approximation, we provide an algorithm for a 3D box primitive representation to identify object parts from 3D point clouds. We motivate and evaluate this choice particularly toward the task of grasping. For this purpose, and as a contribution in the field of grasping, we additionally provide a grasp hypothesis generation framework that utilizes the chosen box presentation in a highly flexible manner.

Outline of the System. We have observed that modeling 3D data by shape primitives is a valuable step for object representation. Sets of such primitives can be used to describe instances of the same object classes, e.g. cups or tables. However, it is not our aim to focus on such high-level classifications or identification of objects, but specifically on grasping. The very basic features we work with are 3D point clouds. From a stereo vision system like ours (described in Section 4.1), these are typically dependent on image resolution, disparity processing method, or an object's segmentation, between 20.000 and 200.000 points per scene. Processing an enormous number of data points takes time, both in approaches that use raw points for grasp hypotheses and in those that try to approximate them as good as possible by shape primitives. Thus, the question remains how rudimentary a model of an object can be in order to be handled successfully and efficiently. While comparable work uses pairs of primitive feature points, e.g. [19], or a-priori known models for each object [20], we are interested in looking into which primitive shape representations might be sufficient for the task of grasping arbitrary, previously unseen objects.

We believe that a mid-level solution is a promising trade-off between good approximation and efficiency for this purpose. Complex shapes are difficult to process, while simple ones will give bad approximations, resulting in unsuccessful grasps. However, we can keep in mind the capabilities of accessible methods to handle imminent approximation inaccuracies for grasping: e.g., haptic feedback, visual servoing and advanced grasp controllers for online correction of grasps. Unknown objects are difficult to parameterize but need real-time application for robot grasping. A computation in terms of minutes for a superquadric approximation is therefore not feasible. We depict an outline in terms of modules, input / output data structures and algorithms that will be handled in this paper in Fig. 1.

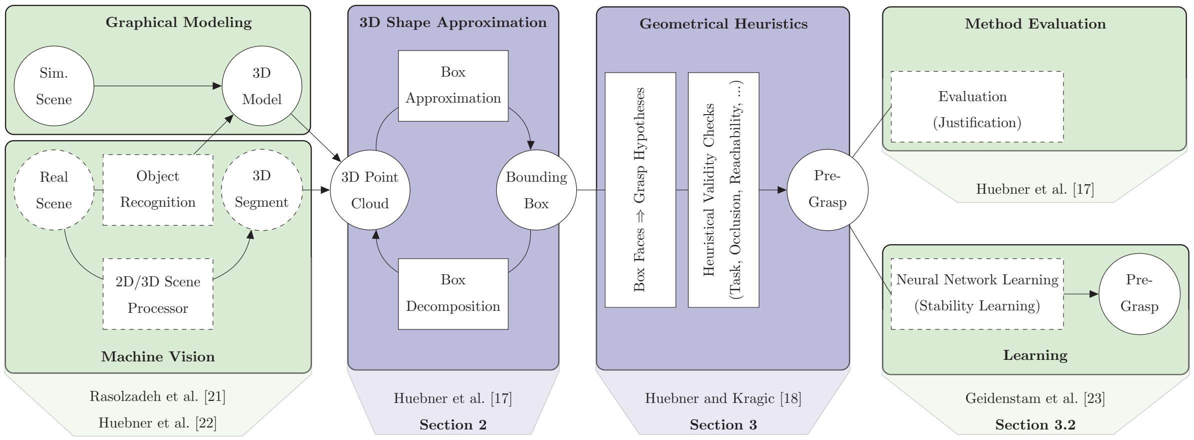


Figure 1: Our work touches several fields of research. In this paper, we will not focus on areas depicted in light green, but on the core idea of bridging basic object representations (3D Point Clouds) and grasping (Pre-Grasp) by 3D Shape Approximation and Geometrical Heuristics. For specific material on the side-areas we refer to the publications. Here, we will consider this material very superficially only.

2 FITTING BOXES TO ARBITRARY POINT CLOUDS

2.1 Minimum Volume Bounding Box Algorithm

We base our algorithm on the minimum volume bounding box computation proposed by Barequet and Har-Peled [24]. Given a set of n 3D points, the implementation of the algorithm computes a Minimum Volume Bounding Box (MVBB) in $O(n \log n + n/\varepsilon^3)$ time, where ε is a factor of approximation. The algorithm is quite efficient and parameterizable by sample and grid optimizations, performing the computation from an arbitrary point cloud to one tight-fitting, oriented MVBB enclosing the data points.

Our aim is now to iteratively split the box and the data points, respectively, in such a way that the new point sets yield a better box approximation of the shape. Iterative splitting of a root box corresponds to the build-up of a hierarchy of boxes. Gottschalk et al. [25] present the OBBTree (Oriented Bounding Box Tree) for this purpose. The goal is to efficiently detect collisions between polygonal objects by the OBBTree representation. The realization of the splitting step is quite straightforward: each box is cut at the gravity center point of the vertices, perpendicular to the longest axis. This is done iteratively, until a box cannot be divided any further.

2.2 Fit-and-Split Adaptation

In our case, the above commonly used strategy is suboptimal. We want to conveniently approximate a shape with as few boxes as possible, thus a splitting into as many small boxes as possible is against our overall aim, if we refrain from merging them again. Additionally, though the MVBB algorithm is efficient, a fitting step after each splitting consumes valuable computation time. On the other hand, splitting at the central point is then not optimal. A heuristic to find a 'good' split is needed. In our case, we define a ‘good’ split is by consulting the relation of the box volume before and after performing the split. A split of the parent box is the better, the less volume the two child MVBBs include. Intuitively, this is clear, as shape approximation is better with highly tight-fitting boxes. In [17], we tested planes parallel to the box surfaces for the best splitting plane. Each MVBB has six sides, whereof opposing pairs are parallel and symmetric. In-between each of these pairs, we can shift a cutting plane. We defined the best split as the one that minimizes the summed volume of the two partitions.

For review, the two core algorithms have been sketched in Fig. 2, Alg. 4.1 and 4.2. Though this is a very approximative method, it is quite fast, as rectangle volume and bound computation are easy to process. However, several problems arise from this split estimation and motivate further improvement:

Cutting Non-Convex Shapes. In the referred earlier work, we stated that cutting non-convex structures is not that simple, especially when distorted, sparse and insecure data is provided. The issue is how to distinguish between a non-convexity and incompleteness of the data. An add-on for the solution of this problem would therefore be more complex and time-consuming.

Cutting Shape Extremities. The minimum volume box fitting approach naturally prefers fitting extremities of the shape into the corners of the boxes, as this keeps the box smaller. The bunny’s ear is again an example for this, since it is almost diagonally suited into one of the box corners. However, especially such extremities can rarely be nicely cut by a face-parallel cut as proposed.

Sensitivity to Noise. A third reason is the result of the box decomposition’s robustness evaluation that we presented in [23]. Briefly summarizing, the evaluation shows that face-parallel splitting is very sensitive to each kind of inaccuracy that can emerge from a real 3D scene and sensors: noise, outliers, shape incompleteness due to viewpoint or viewpoint change, etc.

2.3 Advanced Best Split Computation

These issues prompted us to revisit the split computation. Finally, we developed an algorithm based on convex hulls that solves all the three issues of the simple best splitting and additionally presents much more confident splitting results. For efficiently computing convex hulls on a set of 2D points p, like our projections, we use a Monotone Chain Algorithm [26]. Starting from the convex hull $CH(p)$ of the whole projection, we select those segments S_i of the hull which exceed a given threshold in length. We thereby assume that those either span a ‘valley’ of the outer contour of the data, or they represent a very straight edge. On these segments, we interpolate a number n of sample points $S_{i,j}, j < n$. Between each pair of points $(S_{i,j}, S_{k,l})$ with $i \neq k$, we simulate a cut that splits the point set p into two subsets p_1 and p_2. The two segment points that minimize

$$\theta' = \frac{area(CH(p_1)) + area(CH(p_2))}{area(CH(p))} \tag{1}$$

Algorithm 2.1: BoxApproximate($points^{3d}$)

```
$P \leftarrow findBoundingBox(points^{3d})$
$\{\overline{A}, \overline{B}, \overline{C}\} \leftarrow nonOppositeFaces(P)$
$(p_1^{3d}, p_2^{3d}) \leftarrow split($FindBestSplit$(\{\overline{A}, \overline{B}, \overline{C}\}, points^{3d}))$
$(C_1, C_2) \leftarrow (findBoundingBox(p_1^{3d}), findBoundingBox(p_2^{3d})$
if $(percentualVolume(C_1 + C_2, P) < t)$                                  $\leftarrow$ see (2)
  then BoxApproximate($p_1^{3d}$) and BoxApproximate($p_2^{3d}$)
  else return $(P)$
```

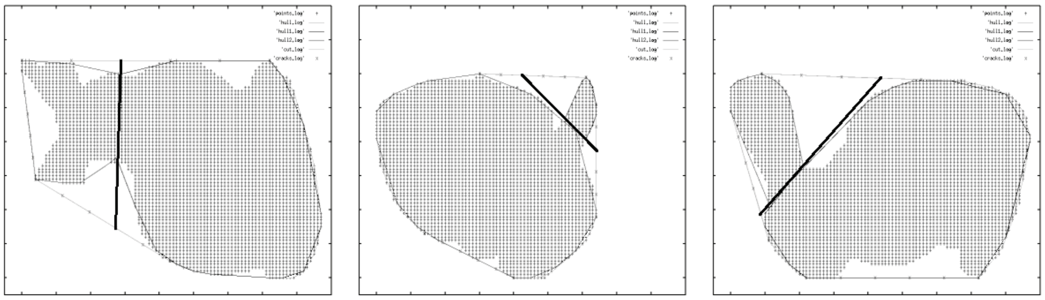
Algorithm 2.2: FindBestSplit_Bound($\{\overline{A}, \overline{B}, \overline{C}\}, points^{3d}$)

```
for $\overline{F} \leftarrow \overline{A}$ to $\overline{C}$
  do
    $p^{2d} \leftarrow project(points^{3d}, \overline{F})$
    for $i \leftarrow 1$ to $\overline{f^x}_{max}$
      do
        $(p_1^{2d}, p_2^{2d}) \leftarrow verticalSplit(p^{2D}, i)$
        $\theta = \frac{boundArea(p_1^{2d}) + boundArea(p_2^{2d})}{area(\overline{F})}$
        if $(\theta < \theta^*)$ then $(\theta^* \leftarrow \theta)$ and $(bestSplit \leftarrow (\overline{F}, \overline{f^x}, i))$
    for $i \leftarrow 1$ to $\overline{f^y}_{max}$
      do
        $(p_1^{2d}, p_2^{2d}) \leftarrow horizontalSplit(p^{2D}, i)$
        $\theta = \frac{boundArea(p_1^{2d}) + boundArea(p_2^{2d})}{area(\overline{F})}$
        if $(\theta < \theta^*)$ then $(\theta^* \leftarrow \theta)$ and $(bestSplit \leftarrow (\overline{F}, \overline{f^y}, i))$
return $(bestSplit)$
```

Algorithm 2.3: FindBestSplit_ConvexHull($\{\overline{A}, \overline{B}, \overline{C}\}, points^{3d}$)

```
for $\overline{F} \leftarrow \overline{A}$ to $\overline{C}$
  do
    $points^{2D} \leftarrow project(points^{3d}, \overline{F})$
    $S \leftarrow longSegments(CH(points^{2D}), lengthThreshold)$
    for each $(S_{(i,j)}, S_{(k,l)})$
      with $(i, k = segmentIndex; i \neq k), (j, l = pointIndex; j, l < n)$
      do
        $(p1, p2) \leftarrow split(p^{2D}, S_{(i,j)}, S_{(k,l)})$
        $\theta' = \frac{area(CH(p1)) + area(CH(p2))}{area(CH(points^{2D})}$                  $\leftarrow$ see (1)
        if $(\theta' < \theta^*)$ then $(\theta^* \leftarrow \theta')$ and $(bestSplit \leftarrow (S_{(i,j)}, S_{(k,l)}))$
return $(bestSplit)$
```

Figure 2: Pseudocode: a point set and its bounding box, respectively, are recursively split (Alg. 2.1). A good split is estimated through analysis of 2D splits of the projected points onto each of the box faces, either using edge-parallel cuts (Alg. 2.2) or convex hull computations (Alg. 2.3).



(a) $\theta'_{\overline{A}} = 0.69$ (b) $\theta'_{\overline{B}} = 0.71$ (c) $\theta'_{\overline{C}} = 0.72$

Figure 3: The three best projection splits using advanced convex hull algorithm. The result is much more confident than with the algorithm from [17],while the additional computational effort is still acceptable.

define our best split, where A is the area function for a convex hull. Practically, we use $n = 6$ (see also Fig. 3). Increasing n might produce more precise cuts, but for the price of additional convex hull computation cost. Pseudocode of this algorithm is sketched in Fig. 2, Alg. 4.3.

Performance of both the old and the new technique can be compared taking a look at Fig. 4. The new technique is more robust to the change of the gain threshold t which will be discussed below. The duck model (Fig. 4f) is not even affected in the cases tested, but stays with the same constellation of three boxes. Another visible effect is that the decompositions seem more intuitive, e.g. the cuts of the handle of the cup (Fig. 4e), or the ears of the bunny (Fig. 4h).

2.4 Fit-and-Split Hierarchy Building

According to the best split θ^*, which would be θ_1 or θ_2 and $\theta'_{\overline{A}}$ respectively in our examples above, the original point cloud can be divided into two subsets of the data points. These can be used as inputs to the MVBB algorithm again and will produce two child MVBBs of the root MVBB. In this way, the complete technique of fit-and-split can iteratively be performed. It is important to note that by MVBB computation, the MVBBs are not axis-aligned.

Additionally, the previous step of 2D cutting is just equal to computing an approximative gain value, for the purpose of efficiency. As an iteration breaking criterion, we subsequently test the real MVBB volume gain Θ^* of the resulting best split measure θ^*. Therefore, we compute the gain in volume defining

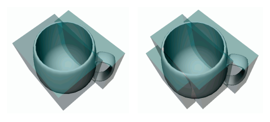
$$\Theta^* = \frac{volume(C_1) + volume(C_2) + V(A^{\setminus P})}{volume(P) + volume(A^{\setminus P})}, \tag{2}$$

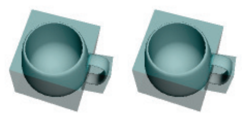
where A is the overall set of boxes in the current hierarchy, P is the current (parent) box, C_1, C_2 are the two child boxes that might be produced by the split, and *volume* being a volume function.

If the gain of free volume is too low, a split should not be executed. For this purpose, we include a threshold value t. The precision of the whole approximation can be parameterized by simply preventing a split if Θ^* exceeds t. Our commonly used thresholds between $t = 0.90$ and $t = 0.95$ are based on thorough experimental evaluations. We also remove boxes with very low number of points.

Simple Best Split

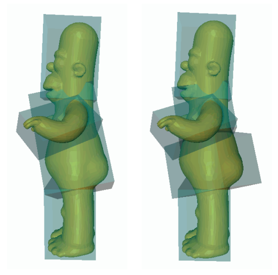
Advanced Best Split

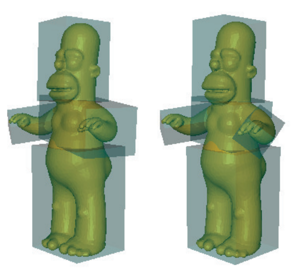

(a) Mug (2,3 boxes)

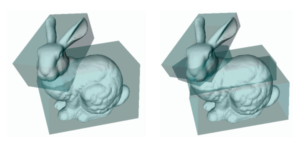

(e) Mug (2,2 boxes)

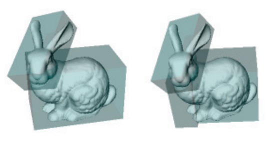
(b) Duck (3,5 boxes)

(f) Duck (3,3 boxes)


(c) Homer (4,5 boxes)


(g) Homer (4,5 boxes)


(d) Bunny (2,4 boxes)


(h) Bunny (2,3 boxes)

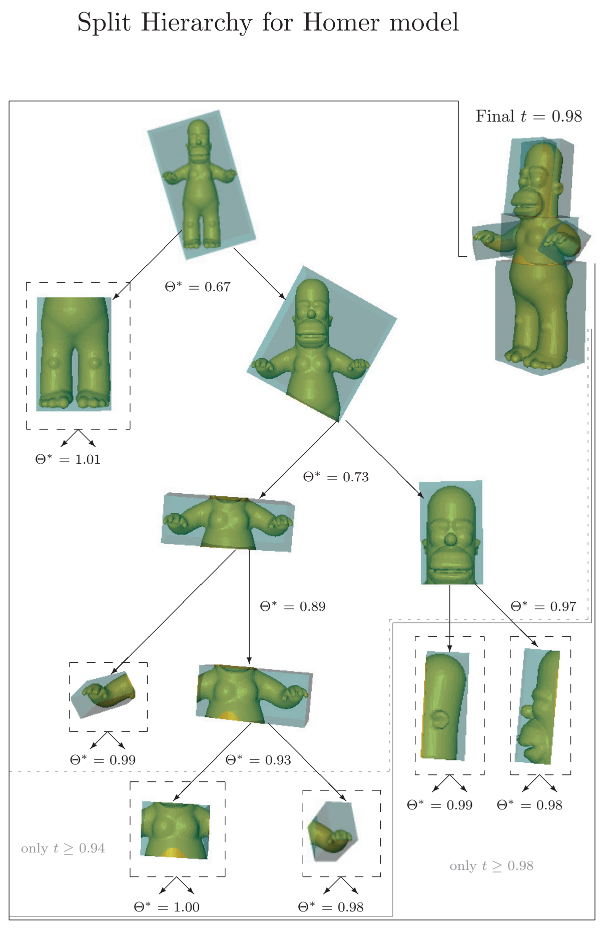


Figure 4: Left: Examples of box decomposition using increasing gain thresholds t=0.90 and t=0.94, where numbers in parentheses correspond to numbers of boxes. (a)-(d) show the results with the simple cutting proposed in [17],while (e)-(h) shows the advanced cutting proposed in Section 2.3.
Right: Visualization of a decomposition hierarchy. The example shows a model decomposition, using a gain threshold $t = 0.98$. If the best volume split value Θ^* below this threshold, a valid cut is detected. Otherwise, the box is a leaf box (*dashed*), a part of the final constellation. One can also trace the results with lower thresholds ($t = 0.9$ dotted gray; $t = 0.94$ solid gray).

Note that by t, both the depth of the hierarchy, the number of leaf boxes, and thereby the detail of approximation is parametrized. Where a split is done is not dependent on t. That is why it is easy to evaluate good values of t, e.g. from a rough root box approximation in the beginning that already can be used for transport, size attribution or grasping, to a higher degree of decomposition into parts.

An example of a decomposition hierarchy can be seen in Fig. 4 using a gain threshold $t = 0.98$. Each time, a best volume split value Θ^* is below t, a valid cut is detected and it is continued separately with two new point clouds. Otherwise, the treated box is a leaf box and thereby part of the final constellation. Besides the result for $t = 0.98$, which is the whole binary tree presented, sub-graphs represent the state of lower thresholds, e.g. those presented for $t = 0.90, 0.94$ in Fig. 4g.

3 GRASP HEURISTICS GIVEN BOX REPRESENTATIONS

The common way to evaluate grasping strategies is extensive evaluation which is practically possible only in simulation [3, 12]. Miller *et al.* have simulated pre-models and shape primitives using their public grasp simulation environment GraspIt! [3]. We also base most of our following experiments on model-based grasping in the GraspIt! simulator. In our experiments, each face will be used for grasp hypotheses, directed along the face's normal vector, and oriented to an edge of the face.

We introduce the observation that generation of pre-grasp hypotheses from box-based face representations reduces the number of hypotheses drastically [17]. In that paper, a random pre-grasp generation included 22104 hypotheses. Though box decomposition effectively produced only very few valid hypotheses (usually <50), they still feature good grasp quality.

3.1 Introducing Higher-Level Dependencies

This result shows that box shapes give efficient clues for planning grasps on arbitrary objects or object parts. For most of the robot tasks we envision, it should be sufficient to find one of the stable grasps, not necessarily the most stable one. Additionally, the part-describing box concept enables grasp semantics to be integrated in the representation, e.g. 'approach the biggest part to stably move the object' or 'approach the smallest part to show a most unoccluded object to a viewer.' The description of an object by shape-based part representation, which is claimed to be a criterion of what grasp is the 'best' in terms of task-dependent grasping, is thereby made available. To briefly refer to the box decomposition approach, a compact box set

$$\mathcal{B} = \{B_1, \ldots, B_n\} \tag{3}$$

encloses a set of 3D points and thereby offers a primitive shape approximation. For each box B_i in the set, we focus on its six rectangular faces

$$\mathcal{F}_i = \mathcal{F}(B_i) = \{F_{(i,1)}, \ldots, F_{(i,6)}\}. \tag{4}$$

Each face spawns up to four grasp approach hypotheses by using the face normal as approach vector and the four edges as orientation vectors, using a pre-defined grasp. Thus, we can define the overall set of hypotheses emerging from the box representation as

$$\mathcal{H} = \{H(F) | F \in \mathcal{F}(\mathcal{B})\}, \quad \text{with } \; H(F_{(i,j)}) = \{F^{0}_{(i,j)}, F^{90}_{(i,j)}, F^{180}_{(i,j)}, F^{270}_{(i,j)}\} \tag{5}$$

In this section, we will extend this framework by introducing several grasp selection criteria, whereof each is based on a different dependency. In each of them, the matter of 'good' is connected to very different dependencies, e.g. a task dependency might vote for a particular box, or a view-point dependency might vote for particular faces only. Technically, all the dependencies will filter out certain $F^{k}_{(i,j)}$ from $\mathcal{H}$ to aim for an even smaller set of valid grasp hypotheses.

3.1.1 Task Dependencies – restricting $\mathcal{B}$

Given a box set (Eq. 3), one can easily compute criteria like the overall mass center (assuming uniformly distributed mass density), volume and dimension of a box, or the relations between boxes. For example, we can define the *outermost* or *innermost* (according to maximum or minimum L2 distance to the estimated center of mass), the *largest* or *smallest*, the *top* or the *bottom*, etc., or even rank the boxes according to these criteria. Given a task, we can easily map an action like *pick-up*, *push*, *show*, *rotate*, etc., to a selected box. For example, in order to *pick-up* something to place it somewhere else, it would intuitively be a good choice to grasp the *largest* box. When showing the same object to a viewer, it would be better to grasp the *outermost* box instead.

Similarly, different tasks can be mapped to grasp configurations. In earlier experiments, we used two static grasp pre-shapes [17]. One can extend this idea towards the selection of different grasp pre-shapes [14], or even the selection of controllers for different tasks. In fact, Prats et al. [27] also use box representations for task-oriented grasping with hand pre-shapes and task frames. However, they assume geometrical knowledge about each object (using a database of 3D models) and structural and mechanical knowledge about a task, e.g. ‘turning’ a door handle.

3.1.2 Box Face Visibility – restricting $\mathcal{F}$

Each box provides six rectangular faces (Eq. 4). We here consider that incomplete data is produced by a single view of an object, since the back of the object is not visible to the sensor. In general, we describe three types of faces: those that are free and visible, those that are blocked by other objects or parts, and those that are not visible for the sensor. The latter is the type we remove in this stage. We motivate this decision not only from the point of box faces produced from erroneous data, but also from the intuitive observation that humans tend to use grasping movements that involve minimum energy effort [28].

3.1.3 Box Face Occlusion and Blocking – restricting $\mathcal{F}$ and $\mathcal{H}$

While the visibility criterion is a check for orientation of faces towards a camera’s or an end-effector’s viewpoint, occlusions and blockings between faces in the box set are also considered. As an example, grasping the head box of the Homer model (revisit Fig. 4) from the bottom is not profitable, since this face is ‘occluded’ by the body box. The corresponding face is then removed from $\mathcal{F}$. We also classify other grasps on the head as being unprofitable. Imagine a grasp towards the head box B_1 from one of the sides. The fingers will not contact the approached face $F_{(1,a)}$, but two of its neighbors, $F_{(1,b)}$ and $F_{(1,c)}$, depending on the hand orientation $k \in \{0, 90, 180, 270\}$. We then define a grasp hypothesis $F^k_{(1,a)}$ as ‘blocked’ in this grasp orientation, if $F_{(1,b)}$ or $F_{(1,c)}$ is occluded, and remove it from $\mathcal{H}$.

This technique has proven to be very effective in reducing the number of hypotheses. Technically, the detection of opposing faces is more complex than the visibility check, since each face of a box has to be compared to each other valid face in $\mathcal{F}$. Therefore, it forms the end of the heuristical selection sequence. However, even if two faces face each other, this is usually not a sufficient condition to mark the face as occluded, since a finger could fit in-between. The handling of such situations would demand additional computational effort. For this reason, and since a more extensive restriction reduces the number of hypotheses drastically, we remove all occluded and blocked hypotheses from our selection.

3.1.4 Reachability and Graspability – restricting $\mathcal{H}$

If there is information available about the embodiment and the kinematics of the robot platform, i.e. its arm and gripper, it is possible to use graspability and reachability criteria to further reduce the number of hypotheses. In terms of graspability, our approach already compares the gripper aperture with the width of the approached face. In terms of reachability, an inverse kinematics solver can be applied for dropping hypotheses that are not reachable with one of the available hands. An integration of an IK solver with our framework is presented in [22].

3.2 Projection Grids and Selection – finding H^*

We can now use the presented box decomposition algorithm to perform a box approximation of the point cloud and reduced hypotheses according the previously presented heuristics. These were aiming at reducing the number of grasp hypotheses according to the task, and 3D orientation or 3D shape of the object. Also the size, i.e. the dimensions of a face, was considered. However, there is usually a set of remaining hypotheses $\mathcal{H}'$ after the restriction steps, from which we would like to select one final 'best' grasp H^*. Our current approach to this issue is selection through estimation of grasp qualities from 2.5D shape projections.

Considering a box and the points that it envelopes, each face produces a projection of the points onto the face plane. In fact, these projections were already computed for best cut detection (see Section 2.3). Discretization was made by dividing each face into equally sized cells, thus projections were represented as dynamically sized binary grids. Additionally, opposing faces shared the same projection grid. These grids kept binary information and were dynamically sized. To adapt this representation and enrich it, we now compute linear information, i.e. minimum distance information to the face plane, in a normalized, fixed-sized grid. Thus, all six projections have to be stored instead of three, since opposing faces do not share the same projection anymore. In return, this representation both allows analyzing the 2.5D depth map of each face and fulfills the input space conditions of a classical neural network.

By providing two grasp quality measures which will be introduced in detail in Section 4.2, GraspIt! [29] is used as a teacher for a supervised network, estimating the stability of a grasp from a given face F and its 2.5D projection grid *proj(F)*, respectively. Since due to normalization in width, height and depth, information about the dimension of F is lost, the box dimensions *dim(F)* are added in terms of three additional neural network inputs.

3.2.1 Final Grasp Decision

Finally, we have to decide *where* and *how* to grasp after initially having reduced the hypotheses to a smaller set. The *'where'* component equals a decision on grasping one of the faces with one orientation. To do this, we apply the neural network approach presented above. The face projections of the remaining hypotheses are fed into the net that has been previously off-line trained with artificial examples. In our experiments, these are mostly complete models which have been processed by the algorithm and their projections grasped in the grasp simulator. By providing the two quality measures, GraspIt! was automatically used as a teacher for the supervised network, estimating the stability of a grasp on a given 2.5D projection grid. After sorting out hypotheses that do not result in good force-closure response (third network output) larger than 0.5, we decide for the one hypothesis with maximum *vol* grasp quality (second network output). According to the definition of grasp hypotheses in (5), this is

$$\mathcal{H}^* = \underset{F^{\phi}_{(i,j)} \in \mathcal{H}' \,\wedge\, \mathcal{N}_{fc}(F^{\phi}_{(i,j)}) \,\geq\, 0.5}{\arg\max} \mathcal{N}_{vol}(F^{\phi}_{(i,j)}), \tag{6}$$

where $\mathcal{N}_x$ is the corresponding output of our neural net $\mathcal{N}$ and $\mathcal{H}'$ the set of hypotheses after the heuristical selection processes.

As briefly described above, the *'how'* component is currently a direct mapping between a manually given task description (e.g. pick, show) to a grasp pre-shape (e.g. power, pinch). An extension which approaches the kinematic properties of the applied gripper and connects them to the projection, in order to estimate good finger contact positions by a set of quality measures, has been proposed in [23].

At this point, we completed a method of selecting a 'best pre-grasp hypothesis' $\mathcal{H}^*$ from a 3D point cloud, using box decomposition. In terms of 'best', this not only considers 'good' stability, which can to some extent be learnt and supported by a neural network, but also the proposed heuristical dependencies, i.e. being 'good' in relation to the task at hand, the gripper embodiment, or the pose of the object.

4 IMPLEMENTATION

In this section, we present an experiment showing the capabilities of the presented techniques. Earlier described experiments have been performed in simulation, but here we test the box decomposition on real 3D stereo data.

4.1 Experimental Acquisition of 3D Data from a Stereo Setup

Our experimental 3D data will be produced from disparity using the four-camera Armar-III stereo head shown in Fig. 5a. More information about the whole Armar-III robot, a humanoid platform at the University of Karlsruhe, can be found in [30] and on `www.paco-plus.org`. The whole system consists of two foveal cameras for recognition and pose estimation, and two wide field cameras for attention. We proposed a grasping strategy for known objects, comprising an off-line, box-based grasp generation technique on 3D shape representations on the complete platform in [22]. our interest here will be the practical processing of the proposed heuristical selection mechanism, including the considered decisions on task, view-point, shape and size properties on unknown objects.

In our current system and scenario, where there is only one table for reasons of simplicity, detecting the table plane is done once by Hough Transformation in 3D. Given a table plane, the 3D scene can further be purged by removing points lying on or below this plane. See Fig. 5 for an example. More details on our work on 3D segmentation can be found in Rasolzadeh et al. [21].

To demonstrate the output of our real system, the 3D data processing from stereo images, image differencing, disparity processing, table plane reduction and 2.5D segmentation is visualized in Fig. 5. Both detected 'objects' are clearly influenced by incompleteness, observable by some holes and by the backsides which are not visible. Additionally, and due to disparity processing, there is noise in disparities, as also some false assumptions, e.g. the uniformly colored top of the mug has been interpolated to a flat surface. Most effects of these uncertainties become clear in Fig. 5d, where the 3D model of both objects are shown from a different viewpoint.

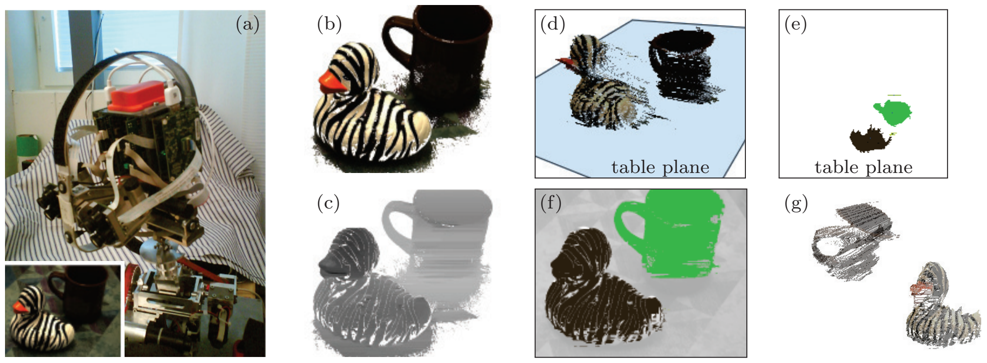


Figure 5: (a) A duplicate of the Armar-III stereo head, used in our lab, including a clipped region of an acquired rectified image. (b) Result of image differencing related to an image without the objects. This mask is applied in (c) to results from the disparity processor. Note that apart from white being the mask region, intensity corresponds to distance to the viewpoint. (d) Reconstructed scene purged by table plane assumption. (e) 2.5D segmentation on table plane projection. The table is not detected, just the (infinite) table plane visualized. (f) Reprojection of the segmented 3D point sets (g) to the image.

4.2 Experimental Simulation of Different Grasps Types

We can apply two types of approach techniques: a (backup) power-grasp and a pinch-grasp.

The *Power Grasp (Backup)* will put each pre-grasp position to a constant distance from the face's center aligned to its normal. We let the hand approach the object along the normal until an arbitrary contact is detected. Afterwards, the hand retreats a small distance (the backup) to call the autograsping function (see below). The *backup* is mainly used due to technical constraints of the simulator, and based on contact instead of shape representation. We will call this type of grasp a *power grasp* to be in line with common grasp taxonomies.

The *Pinch Grasp* will force a grasp towards the center of mass by approximating the distance to the box center the current face belongs to. The *pinch* is therefore based on the shape representation of the object. In contrast to the power grasp, this technique is assumed better for small part grasping, as our power grasp will usually retreat due to contact with another object first (e.g. a table under a pen). From the approximated distance, the *Autograsp* is called. The Autograsp is a built-in grasping technique of GraspIt! which closes all fingers of a gripper simultaneously. It is important to note that we apply just one initial posture for each hand and do not consider different grasp pre-shapes at this point.

When all fingers are in contact, GraspIt! provides computation of two different grasp quality measures, namely eps_{L1} and vol_{L1}. Both measures rely on the convex hull of the union of force cones derivable from the contact points [31]. While vol_{L1} describes the volume as a measure of grasp quality, eps_{L1} corresponds the the radius of the maximum sphere that can be placed inside this volume.

4.3 Experimental Box Decomposition and Grasping

Starting from the two segmented object point clouds in Fig. 5g, we trigger the framework modules (decomposition, hypotheses generation and pre-grasp generation) with the parameters shown in Fig. 6.

Decomposition results for the treated examples are presented in Fig. 7, each (a) and (b). As one can see, both examples are decomposed in a very similar way, resulting in three leaf boxes each. Though not ideally, handle of the cup and head of the duck are separated from the other parts. Decomposition time strongly depends on the complexity of the model, but is linear in relation to splits. In our cases at hand, each split step takes around 4 seconds.

Hypotheses results for the treated examples are presented in Fig. 7, each (c) to (f). (c) shows the occluded and blocked components opposed to the valid ones. Note that also a lot of backsides are invalid, since the viewpoint in the scene is almost equal to the one in the sketches. (d) provides a view on the face projections of all faces (also invalid). On those, the network is tested at a later stage. While in (e), all valid pre-grasp hypotheses are depicted, (f) only shows those that relate to the chosen task-dependency.

The Final Pre-Grasp is determined by applying the trained network on the hypothesis set in (f). The pre-grasp that results in best grasp stability estimate is chosen and performed (in GraspIt!). (g) shows the approach position with fully opened hand, while (h) shows the state after approach and grasp.

4.4 Discussion

The presented results point to a couple of issues to discuss. As one can see, the decomposition of objects into parts is only partially convincing. This is caused by general features of using vision and dense stereo disparity, using a dynamic programming approach from [32, 33] for point cloud generation.

In particular, artifacts appear in the point cloud for uniformly colored regions, since unmatched image points can only be interpolated. This issue makes the cup (Fig. 7b) appear to be closed at the top surface, as also the front shape is quite erroneous. As an additional issue, the only data observable is the one seen from only one view, causing the point cloud to be highly incomplete. Due to effect that

I. Box Decomposition

- ○ Main parameters MVBB calculation [24]
  - • 200 sample points and Grid(B) parameter 3.
- ○ Gain threshold t (see Section 2.3)
  - • 0.90.

III. Pre-Grasp Execution

- ○ Enabled Embodiment-Dependency (see Section 3.1.4)
  - • Graspability test with ARMAR hand (aperture of 120mm)
  - • No reachability check by Arm kinematics
- ○ Enabled Grasp Quality Learning (see Section 3.2)
  - • With ARMAR hand model,
  - • trained on artificial object models (Homer, Mug, Duck)

II. Grasp Hypotheses Generation

- ○ Enabled Task-Dependency (see Section 3.1.1)
  - • $task : pick \rightarrow box : largest, grasp : power$
- ○ Enabled View-Dependency (see Section 3.1.2)
  - • Respective to camera viewpoint
- ○ Enabled Constellation-Dependency (see Section 3.1.3)
  - • Occlusion and Blocking

Figure 6: Algorithm parameters for the grasping experiment in Fig. 7.

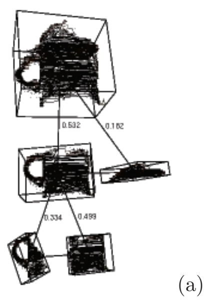
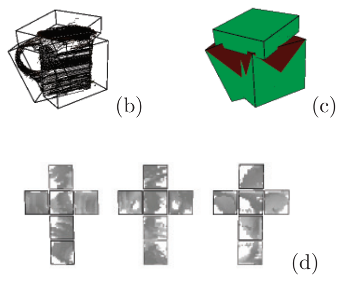
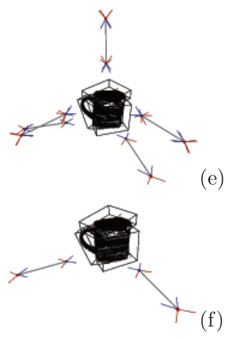
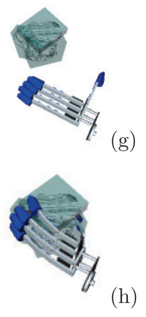
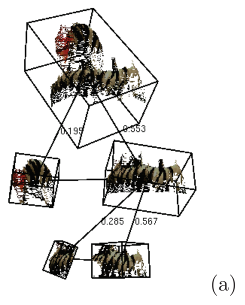
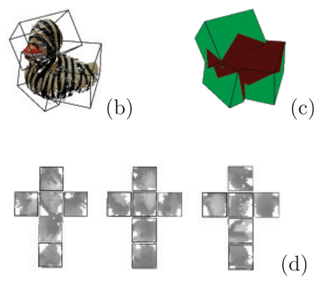
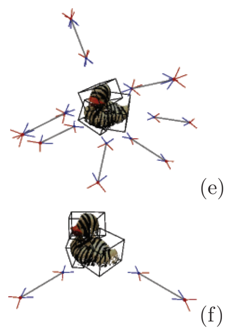
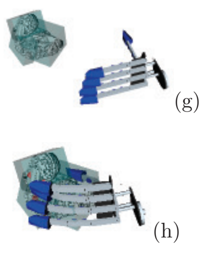
	I. Box Decomposition	**II. Grasp Hypotheses Generation**		**III. Pre-Grasp**
Point Set #1	 - 45.773 Points - 3 Leaf Boxes - 7.33 sec decomp. time	 - 18 overall faces = 72 overall hypotheses - 6(14) valid faces after view (blocking) - 16 hypotheses after blocking / aperture	 - 4 approach hypotheses after box selection (*largest*)	 - Final best grasp
Point Set #2	 - 46.820 Points - 3 Leaf Boxes - 7.81 sec decomp. time	 - 18 overall faces = 72 overall hypotheses - 7(14) valid faces after view (blocking) - 20 hypotheses after blocking / aperture	 - 4 approach hypotheses after box selection (*largest*)	 - Final best grasp

Figure 7: Process for the two point clouds produced in Fig. 5d. For description, see text in Section 4.3.

more a 3D surface than a 3D model is considered by this, also the decomposition fits boxes to those surfaces mainly and thereby looses valuable information about 3D shape. As an example for this, the cup's 'cylindrical' part (Fig. 7b) is finally represented by two orthogonal surfaces which are separated by the decomposition. In comparison to experiments which applied complete models from a database of known objects [22], performance is therefore not as good in terms of shape approximation.

The various steps of hypotheses reduction were intentionally set relatively strictly in the experiment. The view-point heuristic removes all hypotheses that are not visible from the view-point, relating to the issues above. However, in case of complete models, applying this heuristic would be rather unliked: grasps from the backside are valid and interesting, especially when front-hypotheses are not well-rated. The task-related box selection is clearly influenced by the unfavorable box decomposition. Note, however, though in Fig. 7f only hypotheses connected to one box are visualized, all of the hypotheses in Fig. 7e are considered through ranking both related to size of box and grasp quality estimate. This will prevent an empty result if there is no good hypothesis for, e.g. the largest, box by switching to the second-largest, and so on. It can be noticed that the hypotheses set is very restricted. However, the flexibility of the framework allows to disable or enable heuristics to control the size of the final hypotheses set.

For the final pre-grasp generation from a set of hypotheses, the trained network estimates grasp quality from the samples shown in Fig. 7d. The best ranked are performed in Fig. 7g and 7h with a right-hand gripper. As one can see, the grasps are awkward having in mind the embodiment of the right hand on the right arm of the humanoid platform. This problem can be solved by integrating an inverse kinematics solver which is then able to rate hypotheses by their reachability, as also if the left or the right hand can be used [22]. Another issue in this context is that one separate neural network has to be trained for each hand and each type of grasp pre-shape type (e.g. power-grasp, pinch-grasp). It has not been analyzed yet if, and up to which degree, grasp quality measures are generalizable over such options. In the experiment, only one network was trained for the right hand gripper model and the power grasp.

Despite these issues, the framework presented in this paper is one of few that approach 3D shape approximation from dense stereo data instead of 3D range data or 3D meshes for the purpose of grasping. The source of data for our algorithm is arbitrary, as long as it represents 3D point clouds. Nevertheless, the high complexity and manifold difficulties of a vision-based approach have been pointed out. However, we believe that the proposed framework is flexible enough to be extended toward such issues.

5 CONCLUSIONS

We presented the continuation of box approximation work used for robot grasping. We specified the core algorithm and specific extensions of connecting box shape approximation and grasp hypotheses generation in earlier work [17, 18, 22, 23]. In our approach, we combined several motivations known from the shape approximation and grasping literature. In short, we prune the search space of possible approximations and grasp hypotheses by rating and decomposing very basic shapes, which intuitively corresponds to the 'grasping-by-parts' strategy. In this paper, we focused in greater detail on all the parts of an entire framework taking advantage of the very simple shape representation of boxes. Starting from boxes and their faces that the core algorithm produces, we extended the idea of 'grasping on boxes' towards an applicable grasping strategy. This strategy includes various heuristical selection criteria based on efficient geometrical calculations, as also learning from off-line simulation. Basic task-dependencies have been included in this process. We see the strength of our approach in its simplicity and its modularity.

The simplicity is obvious by using boxes and faces in 3D space. Geometric calculations are easier to do in contrast to more sophisticated shape primitives like superquadrics. As presented, boxes and faces can additionally take advantage of linear shape projections. The modularity is established by mostly independent criteria and heuristics that complement each other and flexibly leave space for extensions.

There are many possibilities to extend, adapt and optimize the current framework. Considerations have to be made for the neural net structure, e.g. if it is better to extend the learning to grasp qualities dependent on the chosen grasp pre-shape, i.e. setting three quality outputs for *each* available grasp pre-shape. Additionally, the simulation part for learning is currently done using static simulation. Thus, contact will stay static between gripper and object, while in dynamics, and reality, the object pose will change dependent on the force applied to it. As discussed, we are aware that our approach is a *pre-grip component* based on very robust shape information. The grip component, as an additional module, would contribute in terms of fine correction based on haptic feedback [34]. We see haptic feedback and exploration also as a solution to approach the problem of incomplete models acquired from stereo vision. Merging the 3D data from stereo with 3D data from haptic contact points along the backsides of objects may therefore be an issue of future work.

As another issue, the projection of an object onto the box faces ignores to some extent the real 3D shape of the object, disregarding correct surface normals of the object in the grasp planning. Thus, there is a possibility that planned grasps are infeasible, which addresses the limitation of the proposed planning. In [23], we tried to approach this issue using explicit gripper kinematics in order to analyze finger position estimations on the projections, extending work of Morales et al. [35].

As future work, one could also imagine higher-level part classification to infer suitable grasp pre-shapes from a wider variety of primitives. Given all three projections of a box or the enclosed point cloud itself, one could try to classify the represented shape, which is ought to correspond to an object part. This relates to work on view-based object (part) representation. Classification of shape is a beneficial, but also complex task, as additionally, the box constellation may be very different as influenced by noise, perspective view and uncertainties. For the purpose of grasping on faces, this is not a very severe problem, while in part and object classification, it probably will be. Evaluations of these high-level ideas are not a topic of our short-term goal. However, we are planning to evaluate a model-based part-matching technique like in [36], matching 3D data of shape primitives (e.g. cylinders, spheres, cones) to the point subsets generated from the box decomposition.

Another high-level issue is task dependency. There are different task types on which a grasp may depend. Just to pick up a cup and place it somewhere else might yield a different grasping action as picking up the cup to show it or hand it over. These grasp semantics can be mapped to boxes in the set, e.g. ‘grasp the largest box for a good force grasp to securely move the object’, ‘grasp the smallest box for a good pinch grasp to show a most unoccluded object to a viewer/camera’ or ‘grasp a very outlying box so that another human / robot hand can overtake the object easily’. The latter semantics are quite valuable for applications that are based upon interacting with objects *before* the exploration and recognition stage.

ACKNOWLEDGMENTS

This work was supported by EU through PACO-PLUS, IST-FP6-IP-027657.

REFERENCES

[1] B. Siciliano and O. Khatib, editors. *Springer Handbook of Robotics*. Springer, 2008.

[2] C. Borst, M. Fischer, S. Haidacher, H. Liu, and G. Hirzinger. DLR Hand II: Experiments and Experiences with an Antropomorphic Hand. In *IEEE International Conference on Robotics and Automation*, pages 702–707, 2003.

[3] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen. Automatic Grasp Planning Using Shape Primitives. In *IEEE International Conference on Robotics and Automation*, pages 1824–1829, 2003.

[4] A. Bicchi and V. Kumar. Robotic Grasping and Contact: A Review. In *IEEE International Conference on Robotics and Automation*, pages 348–353, 2000.

[5] X. Zhu, H. Ding, and J. Wang. Grasp Analysis and Synthesis based on a New Quantitative Measure. *IEEE Transactions on Robotics and Automation*, 19(6):942–953, 2003. ISSN 1042-296X.

[6] Y. H. Liu, M. Lam, and D. Ding. A Complete and Efficient Algorithm for Searching 3-D Form-Closure Grasps in Discrete Domain. *IEEE Transactions on Robotics*, 20(5):805–816, 2004.

[7] A. Morales, E. Chinellato, A. H. Fagg, and A. P. del Pobil. Using Experience for Assessing Grasp Reliability. *International Journal of Humanoid Robotics*, 1(4):671–691, 2004.

[8] N. S. Pollard. *Parallel Methods for Synthesizing Whole-Hand Grasps from Generalized Prototypes*. PhD thesis, Dept. of Electrical Engineering and Computer Science, MIT, 1994.

[9] N. S. Pollard. Closure and Quality Equivalence for Efficient Synthesis of Grasps from Examples. *International Journal of Robotic Research*, 23(6):595–613, 2004.

[10] K.B. Shimoga. Robot Grasp Synthesis Algorithms: A Survey. *International Journal of Robotic Research*, 15(3):230–266, 1996.

[11] A. M. Okamura, N. Smaby, and M. R. Cutkosky. An Overview of Dexterous Manipulation. In *IEEE International Conference on Robotics and Automation*, pages 255–262, 2000.

[12] C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelossof. Grasp Planning Via Decomposition Trees. In *IEEE International Conference on Robotics and Automation*, pages 4679–4684, 2007.

[13] S. Ekvall and D. Kragic. Learning and Evaluation of the Approach Vector for Automatic Grasp Generation and Planning. In *IEEE Int. Conf. on Robotics and Automation*, pages 4715–4720, 2007.

[14] M. Cutkosky. On Grasp Choice, Grasp Models and the Design of Hands for Manufacturing Tasks. *IEEE Transactions on Robotics and Automation*, 5:269–279, 1989.

[15] A. Saxena, J. Driemeyer, and A. Y. Ng. Robotic Grasping of Novel Objects using Vision. *Journal of Robotics Research*, 27(2):157–173, 2008.

[16] C. Borst, M. Fischer, and G. Hirzinger. Grasp Planning: How to Choose a Suitable Task Wrench Space. In *IEEE International Conference on Robotics and Automation*, pages 319–325, 2004.

[17] K. Huebner, S. Ruthotto, and D. Kragic. Minimum Volume Bounding Box Decomposition for Shape Approximation in Robot Grasping. In *IEEE International Conference on Robotics and Automation*, pages 1628–1633, 2008.

[18] K. Huebner and D. Kragic. Selection of Robot Pre-Grasps using Box-Based Shape Approximation. In *IEEE International Conference on Intelligent Robots and Systems*, pages 1765–1770, 2008.

[19] D. Aarno, J. Sommerfeld, D. Kragic, N. Pugeault, S. Kalkan, F. Wörgötter, D. Kraft, and N. Krüger. Early Reactive Grasping with Second Order 3D Feature Relations. In *ICRA Workshop: From Features to Actions*, pages 319–325, 2007.

[20] J. Tegin, S. Ekvall, D. Kragic, B. Iliev, and J. Wikander. Experience based Learning and Control of Robotic Grasping. In *Workshop: Towards Cognitive Humanoid Robots, IEEE-RAS Int. Conference on Humanoid Robots*, 2006.

[21] B. Rasolzadeh, M. Björkman, K. Huebner, and D. Kragic. An Active Vision System for Detecting, Fixating and Manipulating Objects in Real World. *International Journal of Robotics Research*, 2009.

[22] K. Huebner, K. Welke, M. Przybylski, N. Vahrenkamp, T. Asfour, D. Kragic, and R. Dillmann. Grasping Known Objects with Humanoid Robots: A Box-Based Approach. In *14th International Conference on Advanced Robotics*, 2009.

[23] S. Geidenstam, K. Huebner, D. Banksell, and D. Kragic. Learning of 2D Grasping Strategies from Box-Based 3D Object Approximations. In *2009 Robotics: Science and Systems Conference*, 2009.

[24] G. Barequet and S. Har-Peled. Efficiently Approximating the Minimum-Volume Bounding Box of a Point Set in Three Dimensions. *Journal of Algorithms*, 38:91–109, 2001.

[25] S. Gottschalk, M. C. Lin, and D. Manocha. OBBTree: A Hierarchical Structure for Rapid Interference Detection. *Computer Graphics*, 30(Annual Conference Series):171–180, 1996.

[26] A. M. Andrew. Another Efficient Algorithm for Convex Hulls in Two Dimensions. *Information Processing Letters*, 9:216–219, 1979.

[27] M. Prats, P. J. Sanz, and A. P. Del Pobil. Task-Oriented Grasping using Hand Preshapes and Task Frames. In *IEEE International Conference on Robotics and Automation*, pages 1794–1799, 2007.

[28] R. M. Alexander. A Minimum Energy Cost Hypothesis for Human Arm Trajectories. *Biological Cybernetics*, 76(2):97–105, 1997.

[29] A. T. Miller and P. K. Allen. Graspit! A Versatile Simulator for Robotic Grasping. *Robotics and Automation*, 11(4):110–122, 2004.

[30] 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 *6th IEEE-RAS International Conference on Humanoid Robots*, pages 169–175, 2006.

[31] A. T. Miller and P. K. Allen. Examples of 3D Grasp Quality Computations. In *IEEE International Conference on Robotics and Automation*, pages 1240–1246, 1999.

[32] D. Scharstein and R. Szeliski. StereoMatcher C++ implementation, `http://vision.middlebury.edu/stereo/`, 2007.

[33] D. Scharstein and R. Szeliski. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, 47(1/2/3):7–42, November 2002. Microsoft Research Technical Report MSR-TR-2001-81.

[34] J. Tegin, S. Ekvall, D. Kragic, B. Iliev, and J. Wikander. Demonstration-based Learning and Control for Automatic Grasping. *Intelligent Service Robotics*, 2:23–30, 2009.

[35] A. Morales, E. Chinellato, A. H. Fagg, and A. P. del Pobil. Experimental Prediction of the Performance of Grasp Tasks from Visual Features. In *IEEE/RSJ International Conference on Robots and Systems*, pages 3423–3428, 2003.

[36] R. Detry, N. Pugeault, and J. Piater. Probabilistic Pose Recovery Using Learned Hierarchical Object Models. In *6th International Conference on Vision Systems, International Cognitive Vision Workshop*, 2008.