2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Estimation of Spatio-Temporal Object Properties for Manipulation Tasks from Observation of Humans

Susanne Petsch and Darius Burschka

***Abstract*— We propose a system for vision-based estimation of manipulation-relevant properties of objects in natural scenes based on observation of human actions. The system consists of an a-priori *(Atlas)* knowledge about known generic objects in the scene and classifies the scene into mission relevant objects and background geometry that is important only for collision avoidance. We present the object-centric structure of our system consisting of an *Atlas* representation and a *Working Memory* storing the current knowledge about the scene, the manipulated objects and actions applied to them in the local environment.**

We present experimental results how the system maintains the information in the database and we show the quality of the results that can be obtained with our system.

I. Motivation

Cognitive systems need to be capable of identifying the mission relevance and of learning the model description of objects by themselves during a joint action with a human operator. Most generally, a model of context specifies the entities to observe, the properties to measure and the relations to detect according to [17]. Dey [6] proposed an operational model for context aware perception. In this model, a situation is defined as a configuration of entities and relations relative to a task. The task serves to determine which entities and relations are of interest and should be observed. We transfer these findings into our environment representation, which allows to decouple complex object recognition loops from the low level 3D reconstruction.

Sensation and perception are key components of cognitive systems. Cognition can be defined as "generation of knowledge on the basis of perception, reasoning, learning and prior-models". Perception is the main source of information for reasoning and learning capabilities. Scene classification is an important task in cognitive systems. It helps in sensor-based 3D model generation to discriminate between objects interesting for missions (*foreground*) and *background* objects relevant merely for localization and obstacle avoidance.

A cognitive system is one that is capable of interacting with humans and other systems in an environment and that is capable to respond to an unexpected event that we will refer to as a *surprise* in the following text. Our system uses the *surprise* to control the learning about the scene and to trigger its own actions as responses to the external stimuli in the environment. We aim to develop a knowledge representation that allows to define manipulation actions based on the current action context and the estimated object properties from observation of human actions and previous interactions with the given object. The distinction between *foreground* and *background* elements allows the system to deal with a possible high complexity of the scene. It focuses the processing only on the structures relevant for manipulation. Our system observes a human operator who identifies the mission relevant objects through a direct interaction with them (manipulation). This way, our system does not need to identify and to learn about all objects in the scene but only about the objects that were used by the human. These objects define the *foreground* layer of our representation while the geometrical model of the entire scene remains as a global three-dimensional structure in a *background* layer. Only a contact of a human hand with an object followed by a change of its position renders the action as something that the system should know about (Fig. 1).

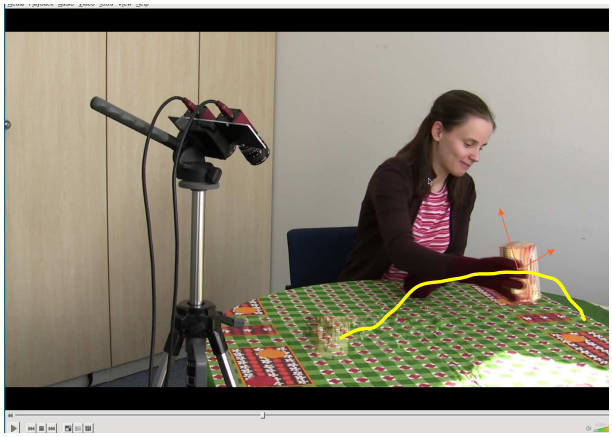

Fig. 1: System observes human actions and completes the internal knowledge representation for objects relevant for a manipulation task.

The target selection task is a challenging part of the system and can be implemented as a manual or automatic process. Examples in 2D image space are described in [14], [13] in more detail. Interesting targets like single standing objects in the scene need to be separated from the supporting planes of the table and floor that are merely relevant for collision avoidance.

Single standing objects are categorized as *foreground* iff a human operator interacted with them or iff they are known to be mission-relevant from previous actions. They need to

This work was supported by the European Communitys Seventh Framework Programme FP7/2007-2013 under grant agreement 215821 (GRASP project)

Susanne Petsch and Darius Burschka are with the Machine Vision and Perception Group, Department of Informatics, Technische Universität München. 85748 Garching, Germany
{petsch|burschka}@in.tum.de

be separated from the environment structure (*background*) first. In an additional step, the remaining *foreground* objects are classified according to their shape, appearance, and their observed allowed motion relative to the scene. The *background* structures are used in a subsequent classification process to estimate the scene context for the current mission.

We consider in general visual and haptic perception as the stimuli generating the input for our cognitive processing. This multi-modal sensor input allows to extract the initial information about *foreground objects* in the scene, to classify them, and to match them to already known representations in the *Atlas (long-term memory)* (Fig. 2). In this paper we focus on visual observation as a first step to acquire an initial guess about the object properties from its appearance.

The paper is structured as follows: in the next section we present details of our approach. We present the way how the a-priori and working knowledge about the actual environment is represented and how the processing of the robot is implemented. In Section III, we present our experimental results showing the different steps of the processing chain. We conclude in Section IV with an evaluation of the current system and present our future work in this area.

A. Related Work

Modayil and Kuipers [10], [11] developed a method where a learning agent can autonomously learn about object models, by detecting, tracking, and characterizing clusters of foreground pixels in the sensory stream. Their agent is a mobile robot that receives a stream of sensory information from a laser range-finder. Grauman and Darrel [7] learned feature masks for object categories by embedding sets of unordered image features into a space where they cluster according to their partial-match correspondences. Weber et al. [16] focused on learning object models that are represented as flexible constellations of rigid parts. Savarese and Fei-Fei [12] proposed a model to represent and learn generic 3D object categories by linking together diagnostic parts of the objects from different viewing points. All these methods learn models for particular objects or object categories from a database of static images under different viewpoints and different backgrounds. Our approach works in 3D space providing a more robust segmentation and registration performance.

In the field of object tracking, Comaniciu et al. [5] proposed a kernel-based tracking algorithm where an object is represented by an ellipsoidal region in the image and the mean-shift tracker maximizes the appearance similarity iteratively. Isard and Blake [8] presented a particle filter based tracking algorithm where object shape is represented by B-splines. Yilmaz et al. [19] proposed a contour-based tracking method using the color and texture models in a band around the objects boundary. Tran and Davis presented a robust object tracking method using regional affine invariant features [15]. In our approach, we use our previously presented 6DoF system VGPS that tracks the structure in monocular images and provides in real-time all six motion parameters.

II. Approach

The robot needs to know about the geometric and physical properties of the object to perform a successful manipulation. Hypotheses about the possible grasp points for the robotic manipulator need to be generated based on the shape and the physical properties like mass and friction of an object. The properties that we currently consider as important for a successful manipulation are: mass, center of gravity, shape to find appropriate surfaces for a successful grasp with a given manipulator, and allowed actions that can be applied to an object. Not all of these properties are observable with a camera and, therefore, we use an additional information database *Atlas* in our system (Fig. 2) to represent the "experience" (*a-priori information*) of the system.

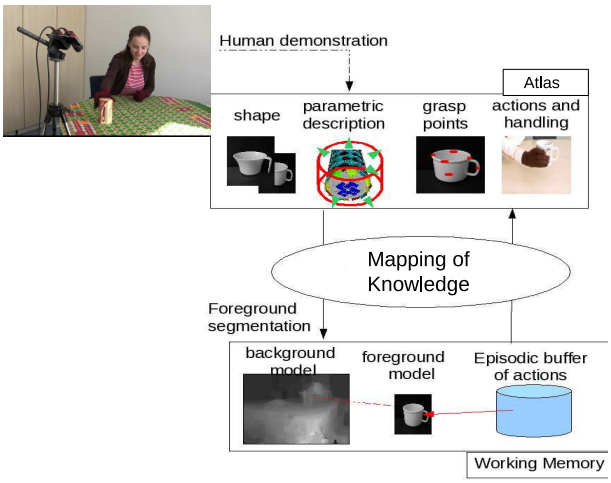

Fig. 2: The system moves the knowledge from a-priori database (Atlas) and instantiates it in the Working Memory representing the actual setup of the manipulation task.

We use for the knowledge representation in the Atlas an analogy to the cognitive capabilities of the human brain and its different strategies, how to store and process the information in the most efficient way. The brain does not store memories in one unified structure. Instead, different types of memory are stored in different regions of the brain. *Long-term memory* in the brain is memory that can last as little as a few days or as long as decades. It differs structurally and functionally from *working memory* or short-term memory, which stores items for only a short time. Working memory (also referred to as short-term memory, depending on the specific theory) is a theoretical construct within cognitive psychology that refers to the structures and processes used for temporarily storing and manipulating information. There are numerous theories as to both the theoretical structure of working memory as well as to the specific parts of the brain responsible for working memory. Baddeley and Hitch (1974) introduced and made popular the multi-component model of working memory [1].

We follow the structure suggested by Baddeley with the long-term memory and the short-term memory maintained

by the central executive (Mapping of the Knowledge in Fig. 2). Our system consists of two databases storing a-priori knowledge about the world (*the Atlas*) corresponding to the long-term memory and a *Working Memory* representing the current visual an spatial representation of the world (visuospatial sketchpad). In this layer, the episodic buffer is implemented as a system storing the typical actions applied to a mission relevant object.

The two layers (Fig. 2) have the following representation:

- **Atlas Representation (Experience of the System)** - this information represents a-priori knowledge given to the system from an expert or representations of the environment collected in previous operations in the same or similar environment. An important difference of the proposed system to many other systems suggested before is that it is supposed to interact with its environment in a cognitive way. This means that the system does not operate based on a set of pre-defined rules but it tries to learn from its own actions and actions of other agents in the environment (human or other robots). The information stored in the Atlas represents a generic knowledge about a class of object.
- **Working Memory**- Working memory is a theoretical construct within cognitive psychology that refers to the structures and processes used for temporarily storing and manipulating information. In our system, the *experience* needs to be grounded to a given environment. We expect to operate in highly complex environments, where the system must not try to analyze all elements of the scene as it is often the case in other current manipulation systems but it needs to focus its *attention* on mission relevant objects whose properties need to be explored for a successful interaction with the world.

An important novelty in the presented system is that the objects are represented not only with their spatial and physical properties (shape, mass, friction) but include also temporal handling information which is essential for the system to handle the object with the same constraints regarding its orientation relative to the gravity vector and accelerations in the translational and rotational motions as presented by the human. The following processing chain allows us to extract this information from the visual system of the robot.

Our system (depicted in Fig. 3) contains the entire processing chain for the visual interpretation of a human action. It starts with the detection of candidates for mission-relevant objects in the world using in our first implementation a simple Supporting Plane Removal algorithm presented already in [2], [4]. In the next step, we use our Vision Interaction Cues (VICs) approach [18] to speed up the processing of human actions. In analogy to VICs, each object defines its own actions and defines a monitoring space around itself. In our system, the human triggers any new knowledge acquisition by presenting new actions to the system. Each cluster segmented in the initial segmentation step defines an *interaction space* where gestures are actually analyzed. It is only necessary to do it if the hand is in the vicinity of a given

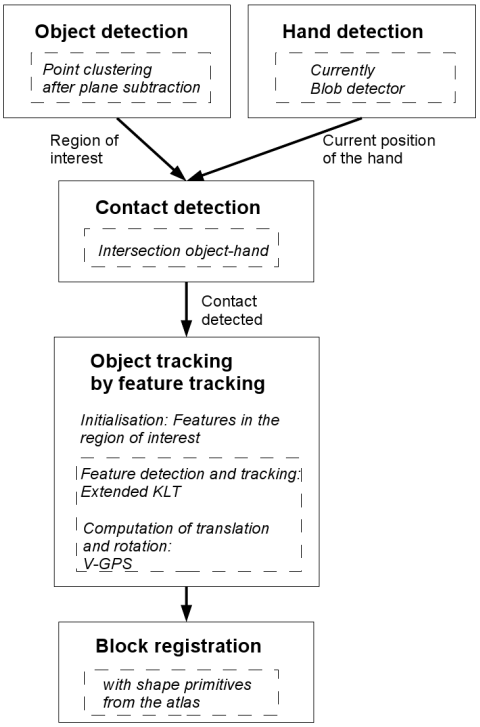

Fig. 3: Overview of the approach. The boxes represent the single modules of the system. Each box contains a dashed box, where the implementations can be found, which are used for the realization of the modules here.

object. Once a grasp gesture at a given cluster is detected the system starts tracking the 6DoF pose of the object to understand the action performed by the user. It stores the corresponding trajectory for later analysis until the object is released. In a final step, a registration step is performed to match the given cluster to the known geometries in the *Atlas* using the shape representations stored in there. The system has the choice to use direct shape registration for known objects or parametric shape analysis to categorize the shape to a specific generic class representation in the *Atlas*.

A. Knowledge Representation

We can tell from Fig. 2 that the *Atlas* contains several distinctive object representations that provide information which is important for the recognition of an object (geometric shape for direct 3D shape registration, and parametric shape description for generalized object class representation) and additional information which is important to initialize parameters which are not observable by the system. These additional parameters are mass, center of gravity and friction leading to specific grasp point representation, and actions that are known to be associated with a given object (e.g., motion constraints on cups or glasses that my contain water).

This a-priori information (*experience*) from the *Atlas* needs to be mapped on the current environment representation surrounding the robot, which is stored in the *Working Memory*. The *Working Memory* contains the geometric shape description as well which is now complete in opposite to the

current sensor reconstruction that usually provides only a partial view due to occlusions in the scene. The registration step to the *Atlas* information allows a completion here. Additionally, now the system is able to store also the texture information representing the appearance of an actual instance of an object in the scene. Now we know not only that there is e.g., a cup, but we also know that this is a cup with a specific texture or logo on it. We move the initial hypothesis about the grasping points and actions from the *Atlas* to the *Working Memory*. Finally, we get also hypotheses about the mass range, center of gravity position, friction and stiffness of the object as an initial guess for the first interaction of the robot with the object. This information is provided as a container for other processing steps and not considered in this paper.

B. Action Representation

An important novelty in our object description is the representation of the temporal changes to the object. We decided to use an object-centric representation of actions. We consider the robot and the human as agents that can imply changes to the state of an object. We are interested in this context only in three phases of the change depicted in Fig. 4

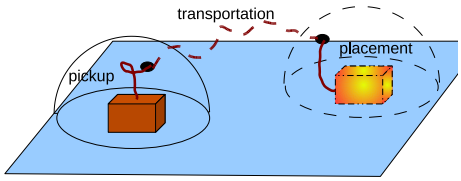

Fig. 4: Three phases defining an action: the type of pickup, the way the transportation is done, and the placement of an object.

The pickup and the placement is mostly concerned with the grasp type performed by the human operator and not part of this paper. This is an information, which is important for an emulation of the grasp by the robot and requires a hand gesture recognition which is out of the scope of this paper. In this paper, we are interested in the analysis of the transportation phase of the action. It is important for us, how free the motion of the object can be (which enforces constraints of coupling between the joints of the robot to ensure a specific orientation relative to, e.g., the gravitational vector) is and where the object is usually placed in the scene. We found it not necessary to save any actual trajectories presented by the human since our focus is on a detailed description of object properties here and the repeatability of the trajectory is relatively low in most cases. For an object it is not important which way it took through the environment but only how it was handled (speeds, orientations) and where it was picked up and placed. This is the information that we need to extract from the vision system.

C. Scene Clustering

In order to detect relevant objects, a plane-subtraction is applied first, which is described in [2], [4]. The approach uses the fact that there is a homography between the (u, v, D) coordinates of the disparity image ($[u, v]$-image coordinates and disparity D) and the corresponding Cartesian coordinates from the 3D scene. According to [2], the planar surface P_r can be represented as

$$P_r : a_r x + b_r y + c_r z = d_r. \tag{1}$$

It is shown in [4], that the equivalent disparity plane is given by

$$D(u,v) = \begin{pmatrix}\rho_1\\ \rho_2\\ \rho_3\end{pmatrix} \cdot \begin{pmatrix}u\\ v\\ 1\end{pmatrix} = n_r^* \cdot \begin{pmatrix}u\\ v\\ 1\end{pmatrix} \tag{2}$$

with the disparity $D(u, v)$ at image coordinates (u, v), $\rho_1 = \frac{a_r}{k}$, $\rho_2 = \frac{b_r}{k}$, $\rho_3 = \frac{c_r}{k}$, $k = \frac{d_r}{B}$ and baseline B.

The next step is the search for the planar candidates. These candidates depend on the gradient of the disparity-map: A high gradient or low gradients with different directions refer to the border of a planar plane, whereas pixel with low gradients of the same direction form a plane. The biggest area with low gradients of the same direction is assumed to be a part of the plane. This area is used for the estimation of the normal vector n_r^* of the plane. The vector n_r^* is estimated according to (6) in [4]:

$$\begin{pmatrix}\sum u_i \cdot D_i\\ \sum v_i \cdot D_i\\ \sum D_i\end{pmatrix} = \begin{pmatrix}\sum u_i^2 & \sum u_i v_i & \sum u_i\\ \sum u_i v_i & \sum v_i^2 & \sum v_i\\ \sum u_i & \sum v_i^2 & \sum 1\end{pmatrix} \cdot n_r^* \tag{3}$$

The direction vector n_r^* enables a comparison between the observed disparity of a pixel and the expected disparity of the plane at the position of the pixel according to the direction vector n_r^* of the plane. If the difference between both is higher than a certain threshold, the pixel is assumed to not belong to the plane. All pixel, which belong to the plane, are deleted in the disparity-map. Consequently the objects, which are placed on the plane, remain in the disparity-map. An example of the plane subtraction is given in Fig. 5.

Fig. 5: Results of plane subtraction. *Left column:* Original color image. *Middle column:* Disparity image of the color image on its left. *Right column:* Remaining object in the disparity image after the plane subtraction.

For the further processing the outer bounding box of the object as the biggest connected component is taken as region of interest. Any other representation could be applied as well.

D. Parsing of Human Action

The manipulation of the objects is going to be parsed as follows. The first step is the detection of the contact of the object and the hand, which will take the object. The position of the object is detected as described before. Since

the position of the object is not changing until its contact with the hand, the computation of the position of the object has not to be computed again. The position of the hand can be determined in different ways, a blob-detector is used here. Therefore the color-image is split in HSV-planes and appropriate thresholds are applied. Just the pixel with the color of the hand remain in the image. If the hand touches the region of interest, a contact is detected. Otherwise the procedure for the contact detection is repeated until a contact is detected.

After the detection of the contact between the hand and the region of interest (the object), the tracking of the features of the object is initialized. Features are selected in the region of the object. If an outer bounding box is used as region of interest, there will be features, which are not on the object and cannot be used for the tracking of the object. The positions of these features do not contain disparity after the plane-subtraction and the features can be deleted. The valid features on the object are used for the tracking of the manipulated object. The contact between the hand and the object is assumed to be lost, when the object and its features are not moving (first trigger) any more and a separation from the object was detected (second trigger). Therefore the tracking of the objects features is finished when all features stop moving. The extended KLT [9] is used here for feature detection and tracking. An example of the contact detection and the tracked features is given in Fig. 6.

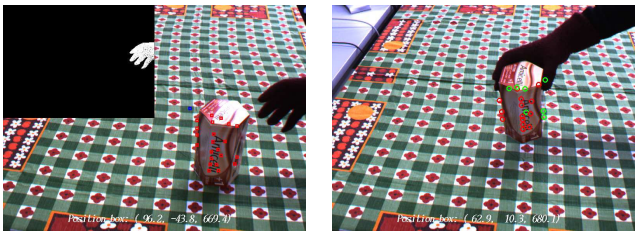

Fig. 6: Contact detection and object tracking. *Left:* The tracking is initialized after the contact detection between the hand and the region of interest. The small red boxes are the valid features, whereas the blue ones are the deleted features, which are not on the box. The top left corner of the image shows the position of the hand, when the contact occurs. *Right:* Example of features during the tracking. The tracked features are shown in red, the assumed position of the lost features are drawn in green.

The recorded trace of the tracked features of the manipulated object enables the computation of its rotation and translation. V-GPS is used for the computation of the rotation and translation [3]. The computed angles and the translation during the movement determine the possible movements of the objects. Additionally the trace of lost features can be reconstructed.

III. Results

In this section the results of the experiments are presented. The used sequences (seq.) have different motion properties, shown in Table I. The movement was either a straight line or an arbitrary motion, the object was either tilted or not. All movements were tested with two different boxes. The scene was recorded with a Firewire Marlin FO46C camera. The following settings were used: image size = 780x582 pixel (width x height). OpenCV, XVision, extended KLT [9] and V-GPS [3] were used in the algorithm, which was running on a Linux system.

TABLE I: Properties of the used sequences

Seq.:	Box 1	Box 2	Movement	Rotation	# Images
1	x		line		705
2	x		line	x	740
3	x		arbitrary		1055
4	x		arbitrary	x	1035
5		x	line		725
6		x	line	x	870
7		x	arbitrary		860
8		x	arbitrary	x	1600

A. Clustering of Object Candidates on a Table

The first part of the experiment is the plane subtraction, as described in II-C, in order to get the position of the object as the region of interest. A sliding-average window was used to get a fill holes in the disparity-map, although that results in smoother transitions between an object and the plane. Just one sequence (seq. 6) required the modification of two additional parameters (a larger kernel for the expected size of the ROI and a higher number of neighbors considered for the comparison of the direction of the gradient) because of a too smooth transition between the object and the table, which included the box as a candidate region for background subtraction.

Fig. 7 shows the result of seq. 6. The result of seq. 3 has already been presented in Fig. 5. The ROI is successfully detected for all sequences, the size of the all ROI is given in table II. The boxes used for the experiments have different sizes (box 2 is larger than box 1), therefore, the algorithm computes correctly a larger ROI for box 2.

Fig. 7: Results of plane subtraction (Seq. 6). *Left column:* Original color image. *Middle column:* Disparity image of the color image on its left. *Right column:* Remaining object in the disparity image after the plane subtraction.

B. Tracking of Human-Induced Motions on the Objects

After the detection of the ROI, the manipulated object is tracked as described in II-D. The tracking is initialized when a contact between the hand and the object is detected. The feature tracking is implemented with extended KLT tracker [9]. The rotation and translation are computed

with V-GPS approach [3]. The rotation and translation is also used for the reinitialization of lost features, since the assumed position of the lost feature can be computed from the estimated rotation and translation of the object. The initialization of the tracker and features during the tracking have already been shown in Fig. 6. An example for the used feature set and the object trajectory is shown in Fig. 8. The trajectory is computed from the position of the tracked features using V-GPS. All sequences show that the trace of a tracked arbitrary feature in the sequence is similar to its projected 3D trajectory.

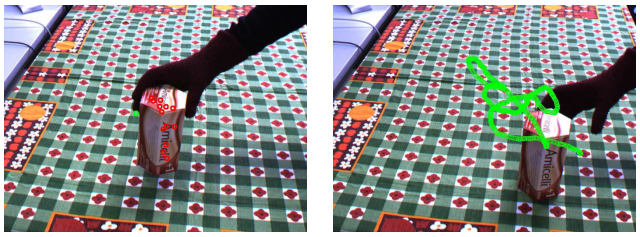

Fig. 8: Tracking of the object (6DoF from monocular) (Seq. 3). *Left:* Example of feature set used for tracking. The tracked features are shown in red, the predicted position of the lost features is drawn in green. *Right:* the object trajectory is shown in green.

Table II contains also the number of features at the beginning and at the end of the sequence. Additionally, the number of features, which were tracked during the whole sequence without a reinitialization, can be seen. The number of features which are tracked during the whole sequence without any reinitialization decreases with the length of the sequence (seq. 3,4,8) and the influence of rotation (seq. 2,4,6,8). The table shows also that the reinitialization of lost features is successful, especially seq. 8.

TABLE II: ROI and number of tracked features

Seq.:	Size ROI (pixel)	# features:	start	end	whole seq.
1	27.945		20	15	7
2	25.488		22	13	4
3	27.800		14	11	3
4	20.088		11	8	2
5	39.026		10	9	8
6	48.830		16	9	5
7	52.398		41	37	30
8	52.832		34	21	7

C. Analysis of the Trajectories

The calculated information about the rotation and translation of the manipulated object during tracking enables the computation of several properties of the manipulation. As already described in III-B, the computation of the object trajectory is possible. Furthermore, the rotation of the object can be computed at each step as well as the speed of the object along the trajectory. Fig. 9 shows the rotation, translation and speed of the object in seq. 7. The orientation of the vertical axis of the object is drawn every 50 steps. We assume, that at the beginning of the trajectory the object orientation is aligned with the calculated normal vector of the table since we do not use any model information for the object. Moreover, Fig. 9 depicts the shape of the speed curve during manipulation. The speed at each position is the average of the past 50 steps in Cartesian coordinates in the 3D Scene. The speed increases during the task.

Fig. 9: Rotation, translation and speed of the object (Seq. 7). *Left:* The development of the rotated and translated normal vector of the table is shown in green. *Right:* The development of the speed along the trajectory is shown in different colors: green = no movement, yellow = slow movement, red = movement, blue = fast movement.

Fig. 10 shows the rotation, the translation and the speed of the manipulation in other trials. The rotation of the object is visible for seq. 4 and 8 while seq. 5 does not contain any rotation of the object. It is a translation along a straight line. Besides the shown rotations and translations of some sequences, the translation and (if applied) rotation of the objects have been drawn for all sequences. Fig. 10 contains also the shape of the speed during the manipulation of the object. The manipulation of the object in seq. 5 along a straight line shows clearly the increasing speed after the pickup, the (in average) constant speed during the transportation and the decreasing speed before the placement.

The results of the angle analysis between the original position of the object and its rotated position during the manipulation are in Table III. The magnitude of the average angle along the trajectory indicates if the object needs to be kept in a vertical orientation or can be tilted during manipulation. As it can be seen, Seq. 2, 6 and 8, which contain rotations, have a clearly higher average angle than seq. 1, 5 and 7 without rotations. In seq. 3 and 4 the system switched to wrong features during tracking resulting in a bias in angle estimates. As table II shows, the number of tracked features in seq. 3 and 4 is really low, therefore, it is obvious that the computation of the rotation and translation, which is based on these features and their number, is challenging for the complex movements in seq. 3 and 4. The fact that there is a small average angle for seq. 1, 5 and 7, which do actually not contain rotations, is also caused by the human operator, since it is hardly possible to move an object without any rotation at all. The results for the maximum angles between the original position and the rotated position (table III) show a similar result: The maximum angle of seq. 2, 6 and 8, which contain rotations, is much higher than for seq. 1, 5 and 7 without rotations. The remaining angle between the original

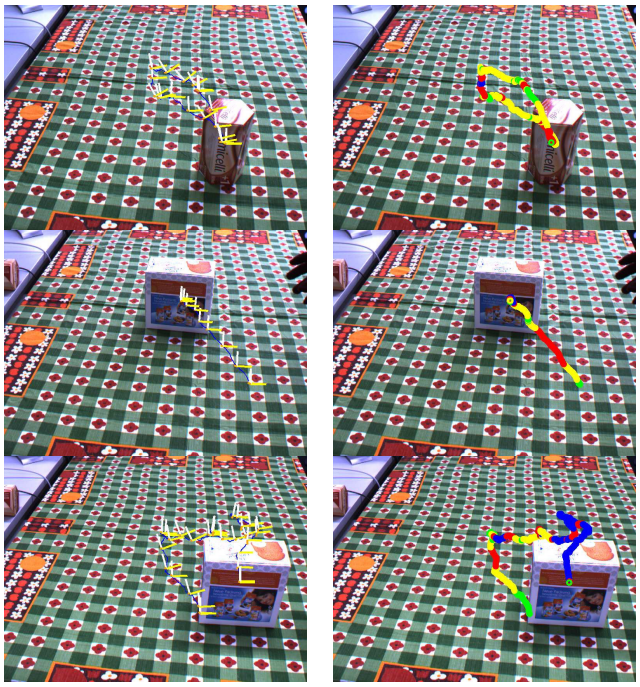

Fig. 10: Rotation and translation of the object in different sequences (Seq. 4, 5, 8). *Left column:* The drawn coordinate system shows the computed rotation and translation of the tracked object. The object trajectory is drawn in yellow. *Right column:* The development of the speed of the object trajectory is shown in different colors: green = no movement, yellow = slow movement, red = movement, blue = fast movement.

position and the final position should be close to zero, since the object is placed on the table again. The results in table III show, that there is a relatively high remaining angle in seq. 2 and 6. These sequences have a small number of constantly tracked features, similar to seq. 3 and 4. The remaining angle of seq. 7 reaches with 0.36 nearly zero. This sequence has the highest number of constantly tracked features among all sequences, therefore it can be concluded that the number of constantly tracked features influences the performance.

TABLE III: Analyzed angles of the sequences

Seq.:	Average	Maximum	Remaining angle (end)
1	4.28	11.11	5.54
2	10.36	31.99	19.69
3	10.08	20.72	4.57
4	6.60	14.23	1.83
5	4.40	10.87	4.86
6	11.30	21.50	20.07
7	5.47	11.28	0.36
8	10.33	25.99	4.08

IV. Conclusions and Future Work

The initial representation developed in the current system is the our testbed how to represent knowledge in a manipulation system and how to define action representations that are necessary for a successful surprise detection. The detection accuracy is already sufficient and will be improved through usage of a bifocal setup in the near future, where the object is observed with a long focal length camera that will allow an even better spatial resolution.

Our next goal is to focus more on the representation of actions in the local environment and to include them in the predictions of the system. We started already work on registration of generic shape descriptions that will allow a classification of objects to a global category. This will allow to provide a-priori suggestion about the manipulation capabilities of an object which may still be unknown to the system.

References

[1] A.D. Baddeley and G.J. Hitch. Working Memory. *New York: Academic Press, vol. 8*, 1974.

[2] D. Burschka and G. Hager. Scene Classification from Dense Disparity Maps in Indoor Environments. In *Proc. ICPR*, 2002.

[3] D. Burschka and G. Hager. V-GPS – Image-Based Control for 3D Guidance Systems. In *Proc. of IROS*, pages 1789–1795, October 2003.

[4] D. Burschka and G.D. Hager. Vision-Based 3D Scene Analysis for Driver Assistance. *ICRA*, 2005.

[5] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based Object Tracking. *Transactions on Pattern Analysis and Machine Intelligence, vol. 25*, pages 564–577, 2003.

[6] A.K. Dey. Understanding and Using Context. In *Personal and Ubiquitous Computing*, volume 5(1), pages 4–7, 2001.

[7] K. Grauman and T. Darell. Unsupervised learning of categories from sets of partially matching image features. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June, 2006.

[8] M. Isard and A. Blake. CONDENSATION - Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision, vol. 29, no. 1*, pages 5–28, 1998.

[9] E. Mair, K. Strobl, M. Suppa, and D. Burschka. Efficient Camera-Based Pose Estimation for Real-Time Applications. *IROS*, 2009, to appear.

[10] J. Modayil and B. Kuipers. Bootstrap learning for object discovery. *IROS*, 2004, vol. 1.

[11] J. Modayil and B. Kuipers. Autonomous Shape Model Learning for Object Localization and Recognition. *IEEE Int. Conf. on Robotics and Automation*, 2006.

[12] S. Savarese and F. Li. 3D Generic Object Categorization, Localization and Pose Estimation. *IEEE International Conf. in Computer Vision (ICCV)*, 2007.

[13] C. Schmid, R. Mohr, and C. Bauckhage. Comparing and evaluating interest points. *Proc. of International Conference on Computer Vision*, 1998.

[14] S. Simhon and G. Dudek. Selecting targets for local reference frames. *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 2840–2845, 1998.

[15] S. Tran and L. Davis. Robust Object Trackinng with Regional Affine Invariant Features. *IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[16] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. *Proc. ECCV, vol. 1*, pages 18–32, 2000.

[17] T. Winograd. Architecture of Context. In *Human Computer Interaction*, volume 16, pages 401–419.

[18] Guangqi Ye, Jason Corso, Darius Burschka, and Gregory D. Hager. VICs: A Modular Vision-Based HCI Framework. In *Proceedings of 3rd International Conference on Computer Vision Systems*, pages 257–267, 2003.

[19] A. Yilmaz, X. Li, and M. Shah. Contour-based Object Tracking with Occlusion Handling in Video Acquired using Mobile Camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 11*, pages 1531–1536, 2004.