Original Paper

Adaptive Behavior

Adaptive Behavior
21(5) 328–345

DOI: 10.1177/1059712313484502
adb.sagepub.com
SAGE

Integrating visual perception and manipulation for autonomous learning of object representations

David Schiebener[1,2], Jun Morimoto[3], Tamim Asfour[2] and Aleš Ude[1,3]

Abstract

Humans can effortlessly perceive an object they encounter for the first time in a possibly cluttered scene and memorize its appearance for later recognition. Such performance is still difficult to achieve with artificial vision systems because it is not clear how to define the concept of objectness in its full generality. In this paper we propose a paradigm that integrates the robot's manipulation and sensing capabilities to detect a new, previously unknown object and learn its visual appearance. By making use of the robot's manipulation capabilities and force sensing, we introduce additional information that can be utilized to reliably separate unknown objects from the background. Once an object has been identified, the robot can continuously manipulate it to accumulate more information about it and learn its complete visual appearance. We demonstrate the feasibility of the proposed approach by applying it to the problem of autonomous learning of visual representations for viewpoint-independent object recognition on a humanoid robot.

Keywords

Humanoid robotics, developmental robotics, object perception, active vision

1 Introduction

The human ability to discern an object from its background is not innate but rather acquired during the development of a child (Fitzpatrick, Needham, Natale, & Metta, 2008). From birth on, children are constantly exposed to events caused by the effects of their own actions. The information thus gained can be utilized to evolve the agent's perceptual judgements, including the way in which objects are perceived. Studies about human object perception revealed that object perception and recognition continuously improve with age (Nishimura, Scherf, & Behrmann, 2009). For example, three-dimensional shapes can be recognized already by 3- to 4-month-old infants (Kraebel, West, & Gerhardstein, 2007). On the other hand, the ability to generalize the acquired knowledge to different viewpoints in order to recognize objects is acquired much later in adolescence (Jüttner, Müller, & Rentschler, 2006). Such insights can help us develop an artificial system for object learning and recognition. In this paper we propose a paradigm that operationalizes the idea of active exploration for learning of visual representations for object recognition.

Many successful systems for object recognition have been developed in recent years, but a comprehensive theory of human object perception remains elusive (Peissig & Tarr, 2007). Segmentation of unknown objects from the background, which is required by many object recognition systems before classification, is easily resolved by humans, but this task is still difficult to implement on artificial vision systems, mainly because it is hard to define what exactly constitutes an object. The meaning of the word "object" is very broad and dependent on semantics and context (Feldman, 2003). While many different principles can be found, e.g. closure, connectedness (Palmer & Rock, 1994), bilateral symmetry (Li & Kleeman, 2011), co-planarity and co-linearity of contours (Kraft et al., 2008), etc., counterexamples can be found for each of them. Hence, such basic principles can only be used to generate hypotheses about the existence of objects in

[1]Jožef Stefan Institute (JSI), Ljubljana, Slovenia
[2]Karlsruhe Institute of Technology (KIT), Karlsruhe, German
[3]ATR Computational Neuroscience Laboratories, Kyoto, Japan

Corresponding author:
Aleš Ude, Jožef Stefan Institute (JSI), Ljubljana, Slovenia; ATR Computational Neuroscience Laboratories, Kyoto, Japan.
Email: ales.ude@ijs.si

two-dimensional images, but additional processes are needed to confirm or reject such hypotheses.

The ecological approach to perception (Gibson, 1979) emphasizes the role of movement. If object manipulability is taken into account, it is much easier to define the concept of object than when only visual characteristics are used (Feldman, 2003). It has been shown that 3- to 5-month-old infants cannot use the Gestalt principle of perceptual organization to distinguish overlapping objects (Spelke, 1990). In contrast, they perceive two objects as separate units if one object moves relative to the other, even when they touch throughout the motion (Spelke, 1990). Based on the concept of object manipulability, we can define objects as physical entities that are manipulable and whose features move in unison when an agent manipulates them. This definition encompasses many physical objects, including for example objects that afford pushing actions. In this paper we show how to exploit this definition in order to learn appearance-based object models that can be used for object recognition. We take the view that visual learning should not be studied in isolation, but should rather involve an agent that actively exploits its manipulation and sensing capabilities.

An early proposal to exploit robot manipulations such as pushing to acquire visual models for recognition was reported by Metta and Fitzpatrick (2003) and Fitzpatrick and Metta (2003). This research focused on the early development of a humanoid robot's visuomotor system and the authors did not attempt to integrate the idea with state of the art approaches to object recognition. Ude, Omrcen, and Cheng (2008), Welke, Issac, Schiebener, Asfour, and Dillmann (2010), Krainin, Henry, Ren, and Fox (2011) and Browatzki, Tikhano, Metta, Bültho, and Wallraven (2012) studied the acquisition of visual models of objects held in the robot's hand. In these works, the authors sidestepped the problem of grasping unknown objects by placing them into the robot's hand. They showed that by having a more accurate control over the object than in the case of pushing, object snapshots from different viewpoints could be systematically acquired. Kraft et al. (2008) proposed an approach to generate grasp hypotheses for unknown objects using a mid-level feature representation. They used motion coherency to accumulate features in a three-dimensional model. It is also possible to move the robot around the object to acquire additional object views (Foissotte, Stasse, Wieber, Escande, & Kheddar, 2010), but in this case the object does not move differently from the rest of the scene, thus motion cannot be used as a cue for segmentation.

Similar to our work, Li and Kleeman (2011) used pushing to disambiguate objects from the background. The detection of bilateral symmetry formed the basis to generate the initial object hypotheses. Motion caused by pushing was used to segment objects from the background also by Kenney, Buckley, and Brock (2009). The developed system relied on background models, which is a problem for active vision systems that are usually mounted on humanoid robots. We proposed to generate hypotheses about the existence of objects in the scene using feature ensembles that form planar surfaces. This work was published in a preliminary form by Stergaršek Kuzmič and Ude (2010) and Schiebener, Ude, Morimoto, Asfour, and Dillmann (2011). Here we discuss our work in the context of developmental robotics and propose a complete framework for autonomous object learning and recognition, which enables the robot to acquire multiview object representations.

A number of systems based on RGB-D and other range cameras, where active manipulation was used to discern objects from the background, have been developed in the past. The main aim of these systems were different manipulation tasks rather than visual learning. Tsikos and Bajcsy (1991) developed arguably the first robotic system that used pushing to support segmentation of flat objects. More recently, Chang, Smith, and Fox (2012) and Gupta and Sukhatme (2012) dealt with the problem of singulation of unknown objects from a pile. They used feature proximity in three-dimensional point clouds to generate the initial object hypotheses, which were then used to generate pushing actions with the goal of separating an object to make grasping easier.

The main contribution of this paper is a new, integrated approach to object segmentation, learning and recognition based on an active perception paradigm. By exploiting its manipulation capabilities, a robot can generate additional information that enables it to autonomously segment, learn and recognize previously unknown objects. We integrated active robot manipulation with state-of-the-art techniques for visual processing, which resulted in a powerful object learning and recognition system with significant improvements over current systems that use only passive vision or systems with active cameras but no manipulation capabilities.

1.1 System overview

Although many papers on image segmentation contain statements such as "segmentation is an important preprocessing step for object recognition", the practical usefulness of low-level segmentation algorithms for the purpose of object recognition has been questionable up to now (Roth & Ommer, 2006). This is due to many ambiguities in natural images that can lead to different segmentation results. In this paper we propose an interactive approach that resolves such ambiguities and makes low-level segmentation viable both for learning of visual models and for object recognition. Since the most successful current object recognition approaches are based on statistical methods, we developed a system

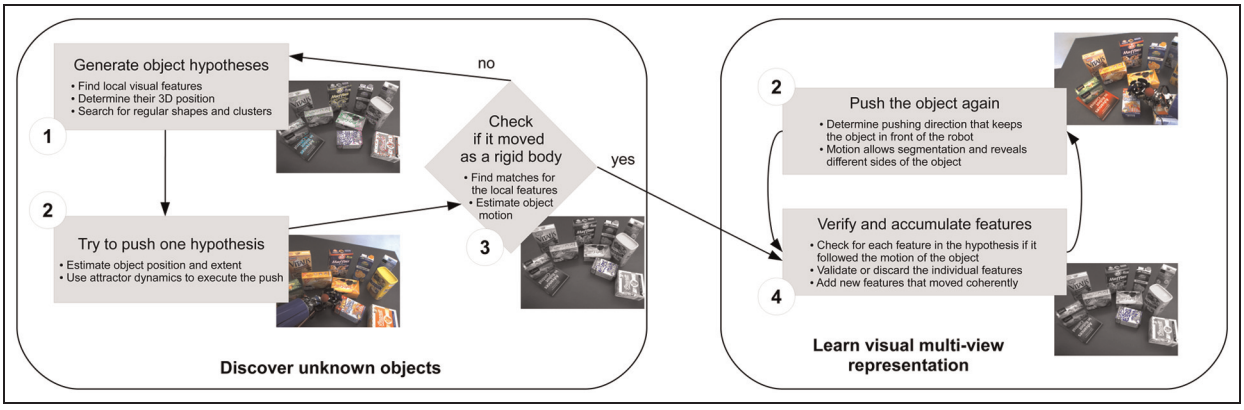

Figure 1. A diagram showing the general structure of our approach to discovering new objects and learning their appearance.

that integrates object manipulation with statistical techniques such as random sampling consensus (RANSAC) (Fischler & Bolles, 1981), bag-of-features (Csurka, Dance, Fan, Willamowski, & Bray, 2004) and support vector machines (SVMs) (Crammer & Singer, 2001). This way autonomous object learning and recognition can be realized in a robust way.

Our approach to object segmentation and learning consists of two main phases, as depicted in Figure 1. In the first phase the aim is to discover an unknown object in a possibly cluttered environment. Initial hypotheses are generated by exploiting simple principles including surface regularity and feature proximity. We use scale-invariant feature transform (SIFT) (Lowe, 1999) and color maximally stable extremal regions (MSER) (Forssen, 2007) as visual features for the estimation of visible surfaces. However, surface regularity and feature proximity are not sufficient to reliably segregate objects in cluttered scenes. Our robot therefore verifies the initial hypotheses by attempting to push the hypothetical object. If the hypothetical object features move as a rigid body after being pushed, then a physical object has been detected with high probability. RANSAC provides the basis both for surface detection and motion verification.

The verification is based on the assumed motion model, which is in our system three-dimensional rigid body motion. In most cases pushing results in a planar object motion, but this fact is not used in the developed system because there are situations in which this assumption is not true, e.g. when the object tumbles after being pushed. By considering full three-dimensional rigid-body motion, we also enable further extensions of the system, e.g. in-hand object manipulation as proposed by Ude et al. (2008), Kraft et al. (2008) and Krainin et al. (2011).

In the second phase, the robot repeatedly pushes the object to first add additional features to the object model that were not part of the initial hypothesis, e.g. because they belong to a different boundary surface

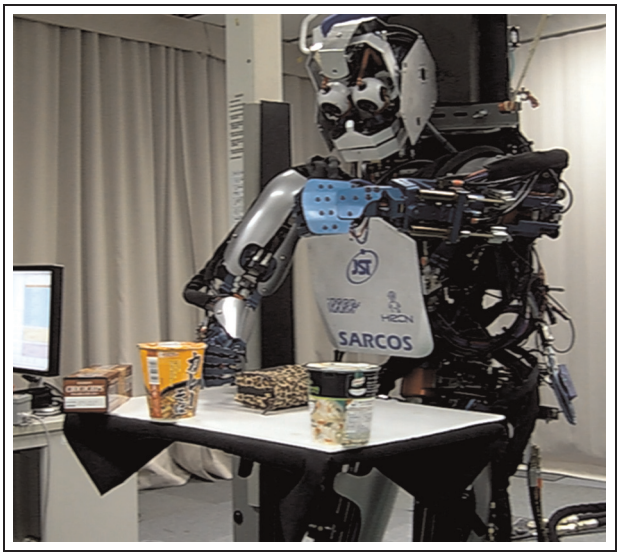

Figure 2. The humanoid robot CB-i pushing an object during the autonomous learning of its appearance.

than the hypothesis, and second to acquire additional snapshots from different viewpoints. The robot plans suitable pushing movements to induce the rotational object motion and interchangeably uses both of its arms to keep the object in front of itself, which is essential to acquire snapshots from different viewpoints. Force sensing is used to prevent uncontrolled collisions with other objects in the scene. In this way the robot autonomously acquires multiple snapshots of the object that can be used to learn a bag-of-features-type model (Csurka et al., 2004) of the object's visual appearance.

When the task is to recognize an object, the first phase, i.e. object discovery, is exactly the same as when learning its model. After the initial object hypothesis has been verified by pushing, additional features that moved in unison with the initial hypothesis, but were not part of it, can be added to the verified object

Figure 3. The humanoid robot ARMAR-III interactively discovers objects that can typically be found in a kitchen environment, segments them and learns their visual appearance.

features. The accumulated features can then be used as input to the learned classifier for object recognition.

In our experiments we used two humanoid robots, CB-i (Cheng et al., 2007) and ARMAR-III (Asfour et al., 2006). They are shown in Figures 2 and 3, respectively. Both robots are equipped with an active stereo vision system and can perform standard oculomotor behaviors including saccadic movements and smooth pursuit (Shibata, Vijayakumar, Conradt, & Schaal, 2001). The eye and head movements are necessary to keep the pushed object within the robot's field of view. This is important to enlarge the area that can be used for learning. We applied active calibration procedures (Ude & Oztop, 2009) to account for the changing robot configuration and thus enable three-dimensional vision. In this respect the developed system is more flexible than previous vision systems for interactive object learning, which used systems with fixed eyes, thus severely limiting the available field of view.

2 Generating object hypotheses

As explained in the introduction, it has not yet been possible to find a fully general definition of object unity (or objectness). We therefore designed various heuristics to generate initial hypotheses about the possible locations of objects in the scene (phase 1 in Figure 1). Any ambiguities can be resolved later in the verification phase. The basic idea of our approach to generating object hypotheses is to search for regular shapes in the acquired stereo images. Most common household objects consist, at least partially, of regular geometric shapes such as planes, spheres and cylinders. Thus, the detection of such a shape is a strong indication about the existence of an object. To be able to push it, the robot just needs a hint about its pose. The position of one of the bordering surfaces is sufficient for this purpose.

From stereo images we first extract Harris interest points (Harris & Stephens, 1988), which are points where the brightness of the image changes significantly in all directions. We use a calibrated stereo camera system to determine correspondences for these points and calculate their three-dimensional positions. The relative uniqueness of Harris interest points and the fact that we only need to search along the epipolar line to find correspondences lead to an almost error-free stereo matching. As a consequence, the obtained three-dimensional points are reliable and precise.

The disadvantage of Harris interest points is that they appear only sparsely on surfaces that are not highly textured. To obtain a complementary feature type for less textured regions of the image, we use maximally stable extremal color regions (Forssen, 2007), which are an extension of the concept of MSER (Matas, Chum, Urba, & Pajdla, 2004) to colors. A color MSER is characterized by having a single color that is very different from the colors of the surrounding area. Stereo matching can be implemented using these regions, providing three-dimensional points in less-textured areas of the image. The estimated positions of color MSER features are usually less precise than those obtained using Harris interest points,[1] but still useful for object detection.

The process described above results in a set of three-dimensional points in all parts of the image, regardless whether they contain textured or single-colored patches. These points are used to search for underlying geometric structures. In Appendix A we describe how some surfaces typical for household environments, e.g. planes, spheres and cylinders, can be detected using a RANSAC algorithm (Fischler & Bolles, 1981). Figure 4 shows some typical scenes and the detected surfaces.

2.1 Hypothesis selection

The algorithms of Appendix A find regular shapes within a set of three-dimensional points acquired by stereo vision. Among these shapes, we select the one that contains the largest number of feature points. This shape is added to the set of initial object hypotheses if the number of its feature points is large enough, and the points belonging to it are removed from the overall set. All three algorithms are then again applied to the remaining point set, the best found hypothesis is saved, its points removed and so on. This process is repeated until no more regular shapes with a sufficient number of points are found. The leftover points are clustered by spatial proximity using the x-means algorithm (Pelleg & Moore, 2000). This way we can detect objects with irregular shapes as long as they contain a sufficient number of feature points. Point clusters are used as object

Figure 4. Initial object hypothesis generation. In the top row, the left image shows all three-dimensional points resulting from the stereo matching of Harris interest points and color MSERs. The right image depicts the object hypotheses that were generated from these points. The extracted features are shown with differently colored crosses to indicate which of the hypotheses they belong to. The bottom row shows the initial hypotheses that were generated for two different settings. Objects are well separated in the left image scene, hence it is relatively easy to create object hypotheses. If the scene is as cluttered as in the right image, purely vision-based approaches reach their limits and hypotheses become more likely to include features from different objects.

hypotheses if they contain enough feature points and the point density within their area is high.

Among the generated hypotheses we select the one containing the maximal number of points because hypotheses that contain more points are easier to re-localize after the push. For this purpose, we first check whether the object hypothesis can be reached by one of the robot's arms. If not, the hypothesis with the second largest number of points is selected, its reachability checked and so on.

Even if the object hypothesis is reachable, we still cannot guarantee that the robot will actually succeed in pushing the object. For example, it can happen that the object is not accessible due to other objects in its neighborhood. This problem can only be solved by a motion planner, but the application of a motion planner requires the system to compute a complete three-dimensional reconstruction of the scene, which is difficult in presence of unknown objects. We therefore do not attempt to solve the accessibility problem in its full generality. Instead we ensure that the robot properly reacts in case of collisions while moving the arm towards the initial pushing configuration, which is done by force sensing (see Appendix B). If pushing is not successful, the robot simply computes a new pushing hypothesis and tries to push the object from a different side or it starts exploring another object hypothesis.

3 Hypothesis verification by manipulation

As discussed in the introduction, as long as there is no exact definition of objectness, we can only use heuristics to find sets of feature points that could belong to a physical object. Such object hypotheses must be validated to confirm whether or not the detected features really originate from a physical object (phase 2 in Figure 1). In case of confirmation, the robot starts acquiring a more complete description of the object's visual appearance. In our system, the robot verifies the existence of an object by physically interacting with the hypothetical object features. By applying actions that cause the features to move, we can check whether the detected features moved according to the assumed motion model, which is in our system the rigid body motion. Pushing is the simplest way to move an object, especially if there is no information about its exact size and shape, which are usually necessary for reliable grasp planning.

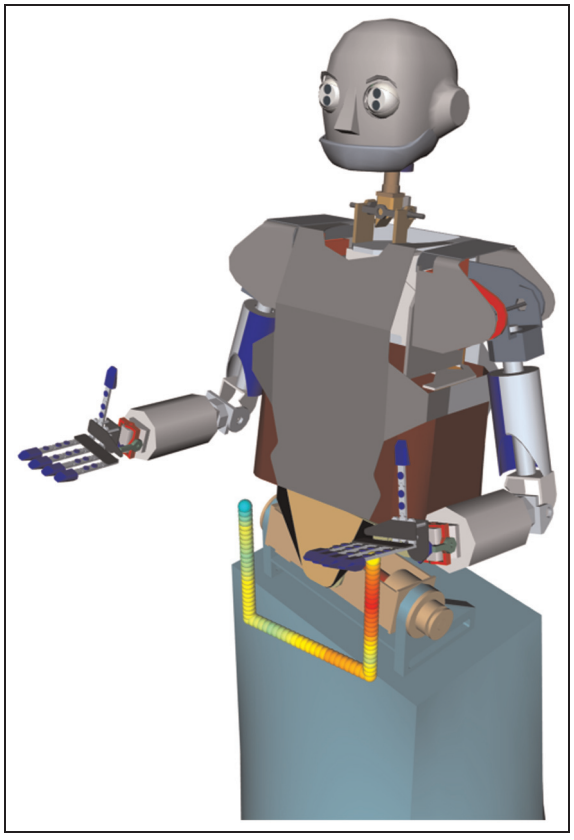

Figure 5. Reachability values on a pushing path for the left hand of ARMAR-III. Dark red dots indicate low, bright yellow dots high reachability values.

The first decision to be made is which of the two robot hands should be used to generate probing pushes. To this end, we analyze the reachability of both hands as described by Vahrenkamp, Berenson, Asfour, Kuffner, and Dillmann (2009). The reachability is encoded as a discretization of the Cartesian space around the robot that gives information about the probability that an inverse kinematics solution exists for a given six-dimensional pose. This model allows the robot to determine whether a position can be reached by one of its hands and how flexible the selected hand is in this area. The robot should use the hand that is more versatile during the whole course of the pushing motion, therefore we integrate the reachability along the pushing trajectory from the position above the starting point to the actual starting point, from there to the end point, and finally to the position above the end point. Figure 5 shows an example of such a path. The hand that accumulates higher reachability over the complete trajectory is selected for pushing.

3.1 Generation of pushing movements

The motion to be induced on the object must be significant enough to allow for verification of the assumed motion model. On the other hand, the resulting object motion should not be so large to completely change the object's appearance, consequently making feature matching very difficult. In addition, the object needs to stay within the robot's field of view and inside the area that the robot can reach with at least one of its arms. The latter aspect becomes increasingly more important when the object is pushed several times to learn its complete visual representation.

To generate a pushing movement, we estimate the object position to be the mean of the points contained in the hypothesis. Although this is not the true position of the object center, it is a sufficient approximation for the generation of the pushing movement. To direct the push, we choose a central point, which is positioned right in front of the robot at the same height as the hypothetical object centre. Such a point is well within the field of view of the robot and is also easy to reach with both arms. The starting point of the pushing movement is taken to be on the side of the hypothesis that is opposed to the central point. The main difficulty is to choose the right offset from the object center, as the size of the object is unknown. The size is estimated by taking the maximal distance of hypothetical object points from the center of the hypothesis. To this value, we add the size of the robot hand and a safety margin (see Figure 6). The end point of the pushing movement is determined by calculating a point at a fixed distance from the starting point towards the central point. A detailed description of how to generate the pushing behavior for the manipulation of object hypotheses (including force control to avoid collisions) is provided in Appendix B. The resulting behavior is shown in Figure 7.

Since the real size of the object is not known, the length of the actual object motion can not be guaranteed. The object does not move at all if we largely overestimate its size. This issue can be mitigated by limiting the estimation of the object size and by using a larger safety margin. On the other hand, starting far away from the hypothesis increases the probability of colliding with other objects.

3.2 Hypothesis validation

After the probing push, the robot needs to check whether the hypothetical object points moved as a physical object (phase 3 in Figure 1). Such feature point motion strongly indicates that the hypothesis indeed constitutes an object. The verification could be limited only to the hypothesis that the robot attempted to push, but even if the hypothesis was wrong, the robot might still by chance move a real physical object. Therefore it is reasonable to check all available hypotheses for indications if their feature points moved as a physical object. In the current version of the system, we assume that the robot interacts with rigid objects.

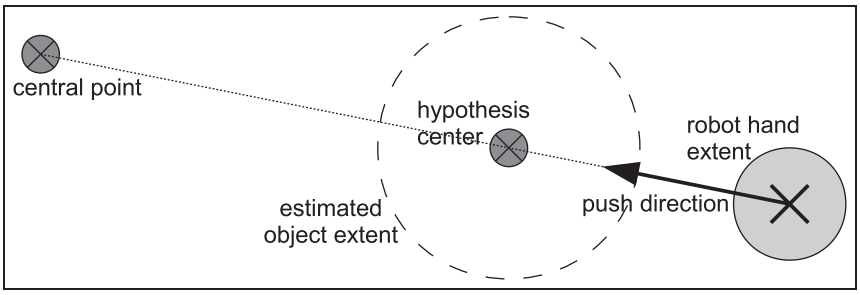

Figure 6. The pushing motion is determined based on the estimated object position and size. The push is always directed towards a fixed central point in front of the robot so that the object stays within its reach and field of view.

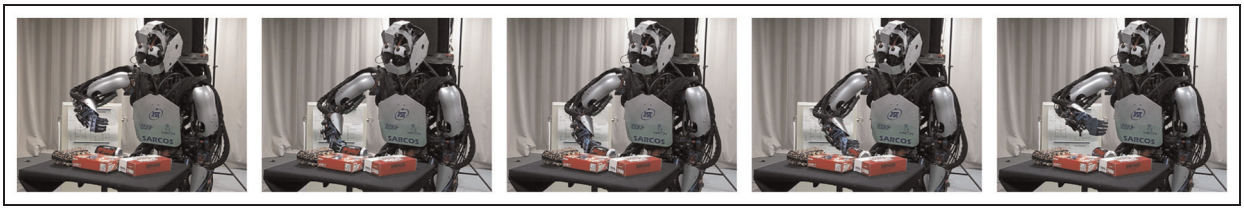

Figure 7. A successful probing push. The robot starts at the position above the object, moves to the starting position for pushing, applies the probing pushing movement and withdraws to the position above the object. All movements are generated using linear, autonomous dynamic systems. After the push, the robot removes the arm from the field of view to allow for unobstructed acquisition of the object image.

More complex models such as articulated (Katz & Brock, 2008) or deformable motion models are possible, but they would lead to more ambiguities in the verification process and longer computational times.

A necessary prerequisite for the analysis of feature point motion is that we can match features extracted from images before and after the push. As mentioned before, we use Harris interest points and color MSER features. For matching we need distinctive descriptors that are as invariant as possible to the rigid-body transformation. In the case of Harris interest points, we use the SIFT descriptor (Lowe, 1999), which consists of a normalized histogram of gradient orientations around the respective point. SIFT descriptors were shown to be robust against viewpoint changes that are not too large. In the case of color MSERs, we use color, saturation and the lengths of the two principal axes of the corresponding region for matching.

After matching, the robot estimates the motion of each hypothesis, i.e. the three-dimensional translation and rotation that maps the features extracted from images before the push onto the features after the push. Since the initial hypothesis may contain features that do not belong to the actual physical object, and furthermore some features may be mismatched between the images before and after the push, there may be many outliers with respect to the assumed motion model. Thus, it is important to apply a robust method for estimating the transformation. Again, RANSAC fits the bill because it is robust against feature mismatches and can filter out the features that do not move according to the assumed motion model. Three pairs of corresponding features before and after the push are sufficient to determine the rigid-body transformation (Horn, 1987), so the implementation of RANSAC is straightforward, just as in Algorithm 2 of Appendix A.

Each rotation can be represented by its axis and angle. We use the weighted sum of the rotation angle and length of translation to evaluate the rigid motion of the hypothesis. If this measure exceeds a threshold (we use 3 cm for translation), we consider the object to have moved and we can proceed with the verification. The features are validated or discarded depending on whether they moved according to the estimated motion model. The validated features originate with high probability from three-dimensional points on the physical object and can thus be added to the training data. Note, however, that it is well possible that some of the visible features are not detected, therefore the robot should continue exploring the object to acquire additional features as well as snapshots from different viewpoints.

After the verification step, we perform a saccadic movement towards the center of the verified feature points. This way we ensure that the object stays in the center of the robot's field of view. Eye and neck degrees of freedom can be used for this purpose. We add two translational, virtual links to the head kinematics (see Figure 8), which enables us to use standard inverse kinematics methods to compute the appropriate eye

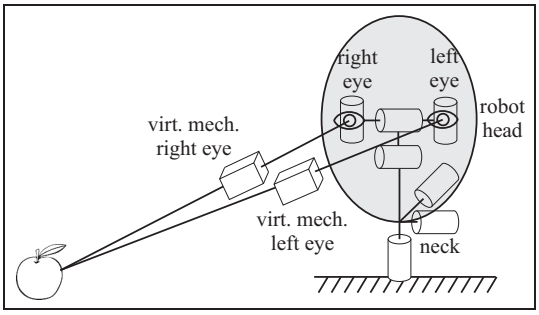

Figure 8. ARMAR-III head kinematics with two translational, virtual degrees of freedom.

and neck movement. With these two additional degrees of freedom, the problem of turning the head towards the three-dimensional point **p** can be formulated as solving the equation

$$\mathbf{p} = \mathbf{f}(\mathbf{y}, l_1, l_2) \quad (1)$$

where **y** are the robot head degrees of freedom, l_1 and l_2 are the two translational virtual links, and **f** is the head's forward kinematics supplemented by the two virtual links. Redundancy is resolved by imposing a secondary task, which is to keep the head as close as possible to the ideal configuration, i.e. head in upright position with eyes directed straight towards the front of the robot.

4 Learning a model for recognition

After successful verification the robot can be quite certain that it has indeed discovered a physical object. The next step is to accumulate object features from different viewpoints.

4.1 Feature accumulation

To acquire a complete object appearance model, the robot repeatedly pushes the object over a constant

Algorithm 1 Learning of a visual multiview object representation.

Generate initial object hypotheses
Select one hypothesis and try to push it
for all hypotheses h **do**
  Estimate the motion of h
  if motion is larger than a threshold **then**
    Confirm h to be an object
  end if
end for
if any hypothesis was confirmed **then**
  Select the confirmed hypothesis containing the maximal number of features and continue with it
else
  Restart from the beginning
end if
while we want to learn more about the object **do**
  Push the object
  Determine matches for all object features
  Estimate object motion
  for all object features f_{ob} **do**
    if f_{ob} moved concurrently **then**
      Mark f_{ob} as confirmed
    else
      Discard f_{ob}
    end if
  end for
  Determine matches for all other features
  for all non-object features f_{no} **do**
  if f_{no} moved concurrently **then**
    Add f_{no} to object as candidate
  else if f_{no} lies within the hypothesis area **then**
    Add f_{no} to object as candidate
    end if
  end for
  Create object descriptor using the confirmed features
end while

distance. The result of this behavior is that the object oscillates around the central point (see Figure 6). Feature motion coinciding with the assumed motion model is a very strong cue for deciding whether a feature is a part of the object that the robot has discovered. After the successful estimation of the rigid-body transformation, the robot looks for other features that underwent the same transformation as the discovered object (phase 4 in Figure 1). If this is the case for a feature that did not belong to the initial hypothesis, we mark it as a candidate feature. Features that lie within the convex hull of the validated points are also marked as candidates. The robot now computes and executes another pushing movement based on the verified feature points. After the push, a new rigid-body transformation is estimated and the candidate features are tested whether or not they moved coherently with the hypothesis. If they did, they are considered validated, otherwise they are discarded. Of course, this only makes sense if the object has moved.

As before, other features that moved concurrently or lie within the convex hull of the verified feature points are added as new candidates. This process of adding candidate features and validating them with the next push can be repeated arbitrarily often, thereby accumulating all features that belong to the object, including those that newly come into view due to the rotational motion of the object. On the other hand, features may get out of view as the object rotates. Validated features that do not follow the motion of the object at a later time may do so either because they were in fact not part of the object, or because they were mismatched when their matching feature after the push was determined. If a feature fails to move together with the hypothesis once, it is not considered when the motion is estimated after the next push, and if it fails twice, it is removed from the hypothesis. Data acquired after a number of successive pushes are shown in Figure 9.

Although SIFT descriptors are fairly tolerant to viewpoint changes, there is still a risk of not correctly matching the features when the rotation in depth is too large. To be more robust to such changes, we accumulate several SIFT descriptors for each feature point (by adding SIFT descriptors calculated from successive snapshots of the object). To reduce the computational effort, we limit the number of SIFT descriptors at each feature point by means of k-means clustering once their number exceeds a specified maximum. Thus, once a feature point has been seen from a few different angles, we have associated with it a set of descriptors, which allows the robot to reliably find the matching feature after a push.

4.2 Model generation

After each iteration of pushing, validation and accumulation, the hypothesis contains four different kinds of features:

- new candidates features;
- validated features that are visible;
- validated features that are not visible; and
- validated features that did not move concurrently.

We use only the visible, validated feature descriptors to generate training data associated with the current viewpoint. As the object is pushed repeatedly, training data from different viewpoints are acquired and a viewpoint-invariant classifier can be learned. Feature vectors generated for training of the classifier consist of two main components. The first is the bag-of-features model (Csurka et al., 2004), which is essentially a histogram of SIFT descriptors. To create such a histogram, the space of SIFT descriptors is divided into a finite number of bins (1000 bins were used in our experiments), which are defined by clustering all descriptors from a large number of training views. Bag-of-features has been shown to be a distinctive and robust classification method and is particularly popular in the field of object recognition. Another important advantage of this method is that it does not require that features are accurately tracked across views.

As a complement to the bag-of-features model, which is based on SIFT descriptors and therefore uses only grayscale pixel values, we also create a color histogram for the area spanned by the feature points. We use a saturation-weighted hue histogram, which means that the hue of each pixel is weighted with its saturation when it is added to the histogram. This avoids problems that may arise if large parts of the object have a very faint color, which might lead to significantly changing histograms when the illumination changes.

4.3 Object recognition

The combination of these two models leads to a very powerful object recognition system, as will be shown in the experimental section. For the actual classification we use techniques such as k-nearest-neighbors (kNN) and SVMs.

To apply recognition techniques based on global descriptors, the object to be recognized needs to be discerned from the background. Traditionally, segmentation is realized through feature clustering, regular windowing or randomized windowing (Ramisa, Vasudevan, Scaramuzza, Mántaras, & Siegwart, 2008), but these are sensitive and time-consuming processes. However, in our system the segmentation problem at recognition time is no different from the one the robot faces when learning a new object, therefore it can deal with it in the same way. We first generate object hypotheses and apply pushing actions to segregate all feature points belonging to the hypothetical object. The extracted, segmented features are then used as input to the trained classifier.

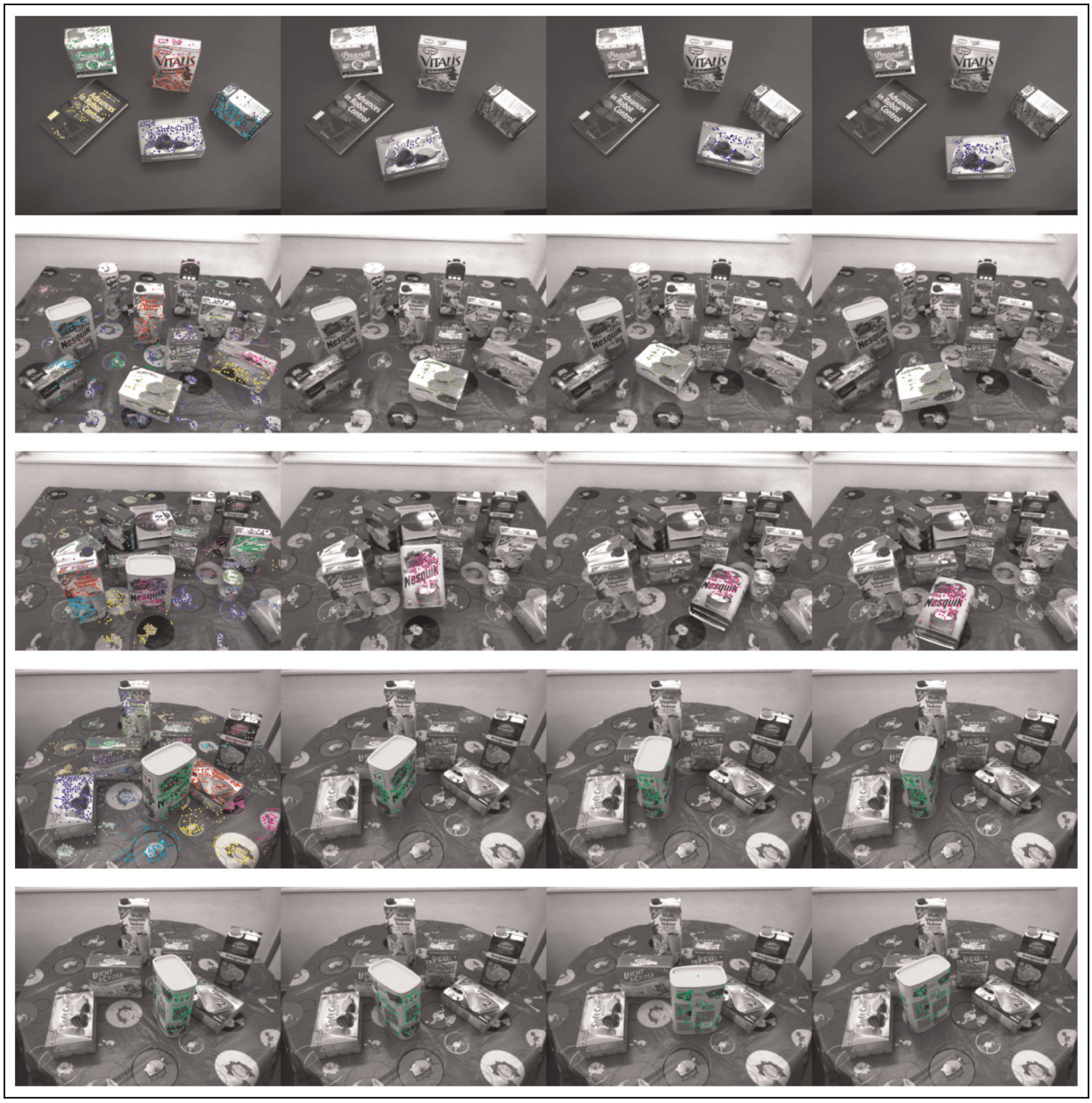

Figure 9. The first three rows show object discovery and segmentation in different settings: in each row, the first image depicts the initial object hypotheses that were generated. As can be seen, their quality decreases in more complex scenes. One hypothesis is then selected for verification and pushed. The second to fourth image in each row show the validated features belonging to that hypothesis after each push. Note that the object is always pushed approximately towards or across a central point in front of the robot, thus staying in the field of view and within the reach of the arms. The last two rows show the segmentation of an object throughout multiple pushes, during which its different sides become visible. This allows the accumulation of object descriptors from multiple viewpoints. Owing to the rather uncontrolled object motion, it may take many pushes until all sides of the object have been revealed. These pictures are an excerpt from a series of 20 pushes.

5 Experimental evaluation

To evaluate different aspects of our system, we conducted extensive experiments on two humanoid robots, CB-i (Cheng et al., 2007) and ARMAR-III (Asfour et al., 2006). Since the proposed paradigm was designed to handle difficult scenes, we tested it in complex scenarios with 5–10 objects placed on a table in front of the robot. As expected, the purely vision-based initial hypotheses could not achieve reliable segmentation in cluttered scenes. Table 1 shows the percentages of hypotheses that coincide with a real object, those which contain only a part of the object and those that are wrong because they extend over more than one object. The first two results allow for a successful push, which leads to a complete and correct segmentation. If the

Table 1. Quality of the initial object hypotheses.

Good	Part of an object	Wrong
50%	39%	11 %

Table 2. Object recognition rate for the initial hypotheses and after *n* pushes.

Initial hypotheses	One push	Two pushes	Three pushes
77%	86%	96%	98%

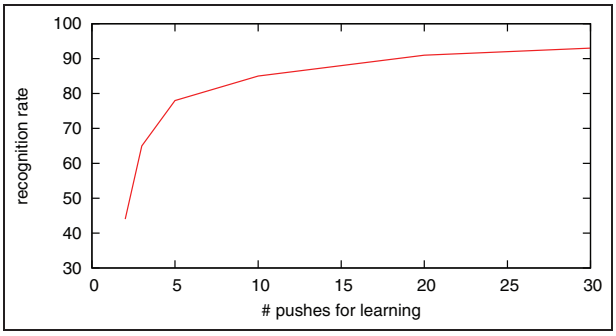

Figure 10. The performance of multiview object recognition with respect to the number of training images acquired from different viewpoints.

robot tries to push an incorrect hypothesis, there is still a chance that it moves one or more of the physical objects and as a consequence is able to segment them correctly. This is the main reason why we test all hypotheses and not only the selected one in the verification phase. We consider this robustness to visual misinterpretations a great advantage of our approach. However, attempting to push an incorrect hypothesis increases the chance of unexpected collisions with other objects.

The most important question that has to be answered is, of course, whether the developed system enables a humanoid robot to autonomously learn classifiers that result in reliable recognition. We let our robots learn the visual appearance of 35 different typical household objects such as food packages, bottles, books, etc. For each object, we accumulated 20 descriptors obtained from different viewpoints. An example sequence used for appearance learning is shown in Figure 9. For recognition, we used support vector machines and 3-nearest-neighbors classification with χ^2 histogram distance. Both classifiers produced similar recognition rates. To segregate the object features from other features and thus obtain input data for object recognition, we used the proposed object hypothesis generation followed by pushing for verification.

Table 2 shows the recognition rate for the initial hypotheses and after 1–3 pushes. The success rate for the initial hypotheses is relatively low, which is due to hypotheses that only cover a small part of the visible object surface or two objects at once and thus result in a partially incorrect segmentation (see also Table 1). After the first push, false and unstable features are removed, which significantly improves the recognition rate. After two pushes, the hypothesis includes almost all visible features of the object, therefore the reliability of the recognition becomes excellent. After three pushes, the percentage of classification errors is further reduced, except when a significant part of the object becomes invisible. This shows that our system can learn appearance-based object models that can be used for reliable recognition.

Another important aspect of the proposed paradigm is that a robot can learn a viewpoint-independent representation of the manipulated object. After each push, the object is observed from a different perspective and the corresponding visual descriptor is created (see also Figure 9). Thus, recognition from multiple viewpoints becomes possible. Since pushing causes a rather uncontrolled motion, we cannot guarantee that the manipulated object has been observed from all sides after training. However, there is a high probability that after a large number of pushes, the viewpoints that are possible without lifting the object have been sampled with sufficient density. In our current implementation, the robot cannot autonomously grasp and place the object at completely different orientations, which is necessary to obtain snapshots of the object from all possible viewpoints. However, instead of the robot a human user can place the object at different orientations, which enables the robot to continue the exploration process and acquire views from fundamentally different viewpoints. In this way, a more complete model of object appearance can be acquired.

In the next experiment we tested how many pushes are necessary to achieve a good multiview coverage of different objects. To this end, we performed a training process with 30 pushes for 5 of our test objects. We took test images of each of them from 8 different view directions, where the object was turned on the table in steps of 45°. We then tried to recognize the object using classifiers learned from subsets of the acquired descriptors. As expected, when only the first few training images were used, the object recognition succeeded only from some of the 8 viewpoints, but with an increasing number of pushes and consequently more training images from different viewpoints, the recognition rate improved (see Figure 10). When looking at the individual objects, it seemed that a certain saturation was usually reached between 15 and 25 pushes.

Both SIFT features at Harris interest points and color MSERs are invariant to motion within the image plane, but they are sensitive to large changes in scale

Table 3. Object recovery rate after scale change.

Scale ratio	1.2 ×	1.3 ×	1.4 ×	1.5 ×	1.6 ×
Recovery rate	100%	100%	91%	54%	4%

Table 4. Object recovery rate after rotation.

Rotation angle	20°	30°	40°	50°	60°
Recovery rate	100%	100%	83%	56%	11%

and especially to rotations in depth. In (Moreels & Perona, 2007), state-of-the-art feature descriptors were analyzed with respect to their robustness against three-dimensional rigid-body motions. As all our hypotheses contain several features, the risk that none of them are rediscovered after the push is naturally smaller than for a single feature. Still, many features may be lost if the change is too large. This risk is particularly high when the object has not yet been seen from different sides, i.e. during the first few pushes. Therefore, we analyzed the sensitivity to viewpoint changes for objects that were pushed only once and thus only contain features belonging to its front side.

Table 3 shows the recovery rate after scale changes. As can be seen, scale changes up to a factor of 1.3 are unproblematic. This corresponds to a motion of 15 cm at an object–camera distance of 50 cm, which is the lower bound for the distance typically seen when a humanoid robot manipulates an object. In our system, the threshold to verify the object motion was usually set to 3 cm, which leaves a generous tolerance margin for the relatively uncontrolled motion, especially because it is unlikely that the pushing motion would occur exactly along the direction that causes the maximal scale change.

Rotations in depth are a bigger problem, as e.g. a push against the edge of an object may cause significant rotation. As described by Moreels and Perona (2007), local descriptors including SIFT tend to fail for orientation changes higher than 25°. Table 4 shows the recovery rate for object hypotheses depending on the angle of rotation caused by the push. For viewpoint changes of up to 30°, the objects could be relocalized reliably, although part of the features belonging to the hypothesis may be lost after a large rotation. Above 40°, there is a high risk of losing the hypothesis completely. In practice, this happens only very rarely and only during the first few pushes. After the manipulated object has been seen from a variety of viewpoints and for each feature we have acquired several different SIFT descriptors, the re-localization becomes more stable. If the robot really looses the track of the current object hypothesis, it restarts the learning process from scratch, starting with the newly generated object hypotheses.

The performance of our system on both robots is shown in two videos, BimanualPushing.mov and PushingForce.mov, which are available as supplementary material to this paper.

6 Conclusions

We presented a new approach that combines object manipulation with visual processing techniques originating in robust statistics. The developed system can autonomously learn visual models of unknown objects, which can later be used for recognition. The proposed approach is robust because, first, it enables the robot to reliably segment unknown objects from the background, second, it does not require long-term feature tracking (features need to be tracked only between successive training views, but any later disappearance or reappearance does not significantly affect the quality of learning) and, third, it enables the acquisition of state-of-the-art statistical models of object appearance. By making use of force sensing, the robot can react to unexpected collisions with other objects during exploration, thus expanding the variety of scenes that can be dealt with by the proposed system. The approach used for segmentation at the time of exploration is also used when the task is to recognize an object. This way segmentation becomes beneficial as a preprocessing step for object recognition. By semi-randomly exploring the object, i.e. by applying pushing movements from different sides of the object, a viewpoint-independent appearance model of the manipulated object can be learnt. Our experiments confirm that the proposed paradigm results in a robust object learning and recognition system, even in cluttered scenes with many objects and textured background.

We use pushing to induce motion on hypothetical objects because it is much easier to push than to grasp an unknown object. However, by grasping the object the robot can control its motion more accurately and acquire training views more systematically. One possible direction for future research is therefore to combine both types of manipulation to learn more comprehensive object models for recognition.

Studies in infant development showed that motion influences the perception of object unity (Spelke, 1990; Johnson & Aslin, 1996; Smith, Johnson, & Spelke, 2003). While there is some controversy about when other cues such as Gestalt principles start affecting the perception of objectness, it is clear from this research that motion starts having an effect from early on. Similarly, even young infants can perceive three-dimensional shapes (Kraebel et al., 2007). Kellman (1993) proposed a two-stage process of how object unity is formed: first, the primitive process which takes into account motion and, second, the rich process which considers also the edge orientation besides motion. Initially, with infants younger than 6 months old, only the

primitive process is operational. Analogous to these findings, the developed technical system segregates objects from the background based on three-dimensional shape and motion cues. The acquired information could be used to train other visual processes that are useful for the perception of object unity.

Passive vision systems, which attempt to learn visual representations that can be used for segmentation from scratch, normally require many thousands of annotated images. While some recent approaches attempt to incrementally learn new representations from little additional data (Fei-Fei, Fergus, & Perona, 2006), they still rely on the existence of prior information, which must come from somewhere. It is therefore necessary to equip an autonomous robot with the capability to learn new visual representations without requiring many thousands of new images and/or a large amount of prior information. The proposed paradigm can be applied to such learning problems.

Note

1. This is because MSER features are region based whereas Harris interest points are calculated directly from the local image structure.

Funding

This work was supported by the EU Seventh Framework Programme (grant agreement number 270273), Xperience, SRBPS, MEXT, the Ministry of Internal Affairs and Communications (contract "Novel and innovative R&D making use of brain structures"), MEXT KAKENHI (grant number 23120004) and by the Strategic International Cooperative Program of JST. A. Ude would also like to thank NICT for its support within the JAPAN TRUST International Research Cooperation Program.

References

Asfour, T., Regenstein, K., Azad, P., Schröder, J., Bierbaum, A., Vahrenkamp, N., & Dillmann, R. (2006). ARMAR-III: An integrated humanoid platform for sensory-motor control. In *2006 6th IEEE-RAS Int. Conf. Humanoid Robots* (p. 169–175). Genoa, Italy.

Beder, C., & Förstner, W. (2006). Direct solutions for computing cylinders from minimal sets of 3D points. In *European Conf. Computer Vision* (ECCV) (p. 135–146).

Browatzki, B., Tikhanoff, V., Metta, G., Bülthoff, H. H., & Wallraven, C. (2012). Active object recognition on a humanoid robot. In *IEEE Int. Conf. Robotics and Automation (ICRA)* (p. 2021–2028). Saint Paul, Minnesota.

Chang, L., Smith, J. R., & Fox, D. (2012). Interactive singulation of objects from a pile. In *IEEE Int. Conf. Robotics and Automation (ICRA)* (p. 3875–3882). Saint Paul, Minnesota.

Chaperon, T., & Goulette, F. (2001). Extracting cylinders in full 3D data using a random sampling method and the Gaussian image. In *Proc. Vision Modeling and Visualization Conference*.

Cheng, G., Hyon, S.-H., Morimoto, J., Ude, A., Hale, J. G., Colvin, G., Scroggin, W., & Jacobsen, S. C. (2007). CB: A humanoid research platform for exploring neuroscience. *Advanced Robotics*, *21*(10), 1097–1114.

Crammer, K., & Singer, Y. (2001). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, *2*, 265–292.

Csurka, G., Dance, C., Fan, L. X., Willamowski, J., & Bray, C. (2004). Visual categorization with bags of keypoints. In Proc. ECCV *Int. Workshop Statistical Learning in Computer Vision*. Prague, Czech Republic.

Fei-Fei, L., Fergus, R., & Perona, P. (2006). One-shot learning of object categories. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, *28*(4), 594–611.

Feldman, J. (2003). What is a visual object? *Trends in Cognitive Sciences*, *7*(6), 252–256.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381–395.

Fitzpatrick, P., & Metta, G. (2003). Grounding vision through experimental manipulation. *Philosophical Transactions of the Royal Society – Series A, 361*(1811), 2165–2185.

Fitzpatrick, P., Needham, A., Natale, L., & Metta, G. (2008). Shared challenges in object perception for robots and infants. *Infant and Child Development*, *17*, 7–24.

Foissotte, T., Stasse, O., Wieber, P.-B., Escande, A., & Kheddar, A. (2010). Autonomous 3d object modeling by a humanoid using an optimization-driven next-best-view formulation. *International Journal of Humanoid Robotics*, *7*(3), 407–428.

Forssen, P. (2007). *Maximally stable colour regions for recognition and matching*. Minneapolis, Minnesota.

Gibson, J. (1979). *The ecological approach to visual perception*. Boston, MA: Houghton Mifflin.

Gupta, M., & Sukhatme, G. S. (2012). Using manipulation primitives for brick sorting in clutter. In *IEEE Int. Conf. Robotics and Automation (ICRA)* (p. 3883–3889). Saint Paul, Minnesota.

Harris, C., & Stephens, M. (1988). *A combined corner and edge detector*.

Horn, B. K. P. (1987). Closed-form solution of absolute orientation using unit quaternions. *Journal Optical Society America* A, *4*(4), 629–642.

Johnson, S. P., & Aslin, R. N. (1996). Perception of object unity in young infants: The roles of motion, depth, and orientation. *Cognitive Development*, *11*, 161–180.

Jüttner, M., Müller, A., & Rentschler, I. (2006). A developmental dissociation of view-dependent and view-invariant object recognition in adolescence. *Behavioural Brain Research*, *175*(2), 420–424.

Katz, D., & Brock, O. (2008). Manipulating articulated objects with interactive perception. *In IEEE Int. Conf. Robotics and Automation (ICRA)* (p. 272–277). Pasadena, CA.

Kellman, P. J. (1993). Kinematic foundations of infant visual perception. In C. E. Granrud (Ed.), *Visual perception and cognition in infancy* (p. 121–173). Hillsdale, NJ: Erlbaum.

Kenney, J., Buckley, T., & Brock, O. (2009). Interactive segmentation for manipulation in unstructured environments. In *IEEE Int. Conf. Robotics and Automation (ICRA)* (1377–1382). Kobe, Japan.

Kraebel, K. S., West, R. N., & Gerhardstein, P. (2007). The influence of training views on infant's long-term memory for simple 3D shapes. *Developmental Psychobiology*, *49*(4), 406–420.

Kraft, D., Pugeault, N., Baseski, E., Popovic, M., Kragic, D., Kalkan, S., Wörgötter, F., & Krüger, N. (2008). Birth of the object: Detection of objectness and extraction of object shape through object-action complexes. *International Journal of Humanoid Robotics*, *5*(2), 247–265.

Krainin, M., Henry, P., Ren, X., & Fox, D. (2011). Manipulator and object tracking for in-hand 3D object modeling. *The International Journal of Robotics Research*, *30*(11), 1311–1327.

Li, W. H., & Kleeman, L. (2011). Segmentation and modeling of visually symmetric objects by robot actions. *The International Journal of Robotics Research*, *30*(9), 1124–1142.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Int. Conf. Computer Vision (ICCV)* (1150–1157). Corfu, Greece.

Matas, J., Chum, O., Urba, M., & Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. Image and Vision Computing, *22*(10), 761–767.

Metta, G., & Fitzpatrick, P. (2003). Better vision through manipulation. *Adaptive Behavior*, *11*(2), 109–128.

Moreels, P., & Perona, P. (2007). Evaluation of features detectors and descriptors based on 3D objects. *International Journal of Computer Vision*, *73*, 263–284.

Nishimura, M., Scherf, S., & Behrmann, M. (2009). Development of object recognition in humans. *F1000 Biology Reports*, *1*(56).

Palmer, P., & Rock, I. (1994). Rethinking perceptual organization: the role of uniform connectedness. *Psychonomic Bulletin and Review*, *1*, 29–55.

Peissig, J. J., & Tarr, M. J. (2007). Visual object recognition: Do we know more now than we did 20 years ago? *Annual Review of Psychology*, *58*, 75–96.

Pelleg, D., & Moore, A. (2000). X-means: Extending k-means with efficient estimation of the number of clusters. In Proc. 17th *Int. Conf. Machine Learning* (p. 727–734). San Francisco, CA.

Ramisa, A., Vasudevan, S., Scaramuzza, D., Mántaras, R. L. de, & Siegwart, R. (2008). A tale of two object recognition methods for mobile robots. In *Int. Conf. Computer Vision Systems (ICCV)* (p. 353–362). Santorini, Greece.

Roth, V., & Ommer, B. (2006). Exploiting low-level image segmentation for object recognition. *In Pattern Recognition* (p. 11–20). Berlin, Heidelberg: Springer-Verlag.

Schaal, S., Peters, J., Nakanishi, J., & Ijspeert, A. (2005). Learning movement primitives. In *Robotics Research: The Eleventh International Symposium* (p. 561–572). Berlin, Heidelberg: Springer-Verlag.

Schiebener, D., Ude, A., Morimoto, J., Asfour, T., & Dillmann, R. (2011). Segmentation and learning of unknown objects through physical interaction. In *2011 11th IEEE-RAS Int. Conf.* Humanoid Robots (p. 500–506). Bled, Slovenia.

Shibata, T., Vijayakumar, S., Conradt, J., & Schaal, S. (2001). Biomimetic oculomotor control. *Adaptive Behavior*, *9*(3–4), 189–207.

Smith, W. C., Johnson, S. P., & Spelke, E. S. (2003). Motion and edge sensitivity in perception of object unity. *Cognitive Psychology*, *46*, 31–46.

Spelke, E. S. (1990). Principles of object perception. *Cognitive Science*, *14*, 29–56.

Stergaršek Kuzmič, E., & Ude, A. (2010). Object segmentation and learning through feature grouping and manipulation. In *2010 10th IEEE-RAS Int. Conf. Humanoid Robots* (p. 371–378). Nashville, Tennessee.

Tsikos, C. J., & Bajcsy, R. K. (1991). Segmentation via manipulation. *IEEE Transactions on Robotics and Automation*, *7*(3), 306–319.

Ude, A., Omrčen, D., & Cheng, G. (2008). Making object learning and recognition an active process. *International Journal of Humanoid Robotics*, *5*(2), 267–286.

Ude, A., & Oztop, E. (2009). Active 3-D vision on a humanoid head. In *International Conference on Advanced Robotics (ICAR)*. Munich, Germany.

Vahrenkamp, N., Berenson, D., Asfour, T., Kuffner, J., & Dillmann, R. (2009). Humanoid motion planning for dual-arm manipulation and re-grasping tasks. In *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)* (p. 2464–2470). St. Louis, MO.

Welke, K., Issac, J., Schiebener, D., Asfour, T., & Dillmann, R. (2010). Autonomous acquisition of visual multi-view object representations for object recognition on a humanoid robot. In *IEEE Int. Conf. Robotics and Automation (ICRA)* (p. 2012–2019). Anchorage, Alaska.

About the Authors

David Schiebener received his diploma degree in computer science from the Karlsruhe Institute of Technology (KIT) in 2011. Currently, he is a PhD student at the High Performance Humanoid Technologies Lab (H^2T). In 2011, he was a visiting researcher at Jožef Stefan Institute, Ljubljana, Slovenia. His research activities include computer vision with the focus on active vision, autonomous object learning, and visuo-haptic exploration.

Jun Morimoto received his PhD in information science from the Nara Institute of Science and Technology, Nara, Japan, in 2001. From 2001 to 2002, he was a postdoctoral fellow with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA. Since 2002, he has been with the Advanced Telecommunications Research Institute International, Kyoto, Japan, where he was a researcher with the Computational Brain Project, International Cooperative Research Project, Japan Science and Technology Agency, from 2004 to 2009, and is currently the head of the department of brain robot interface, ATR Computational Neuroscience Laboratories.

Tamim Asfour is a full professor at the Institute for Anthropomatics, Karlsruhe Institute of Technology (KIT). He is chair of Humanoid Robotics Systems and Head of the High Performance Humanoid Technologies Lab (H2T). He received his diploma degree in Electrical Engineering and PhD in computer science from the University of Karlsruhe (TH) in 1994 and 2003, respectively. His current research interest is high performance humanoid robotics. Specifically, he is working on the engineering of versatile 24/7 humanoids able to predict, act and interact in the real world. He has been active in the field of humanoid robotics for the last 14 years with focus on engineering complete humanoid robot systems including humanoid mechatronics and mechanoinformatics, dexterous grasping and manipulation, action learning from human observation, goal-directed imitation learning, active vision and active touch, whole-body motion planning, system integration, robot software and hardware control architecture. He is developer and leader of the development team of the ARMAR humanoid robot family.

Aleš Ude received a diploma degree in applied mathematics from the University of Ljubljana, Slovenia, in 1990 and a PhD from the Faculty of Informatics, University of Karlsruhe, Germany, in 1995. He is currently the head of Humanoid and Cognitive Robotics Lab, Department of Automatics, Biocybernetics, and Robotics, Joef Stefan Institute, Ljubljana. He is also with the ATR Computational Neuroscience Laboratories, Kyoto, Japan. His research interests include autonomous robot learning, imitation learning, humanoid robot vision, perception of human activity, humanoid cognition, and humanoid robotics in general. Dr. Ude is a recipient of the Science and Technology Agency fellowship for postdoctoral studies with the Exploratory Research for Advanced Technology (ERATO) Kawato Dynamic Brain Project, Japan.

Appendix A

The main difficulty we encounter when searching for geometric structures within a point cloud is that only a small subset of all points constitutes a surface that really exists in a three-dimensional world. So for each surface we detect, most of the points from the overall set are actually outliers with respect to it. An algorithm that is well suited to deal with such situations is the RANSAC (Fischler & Bolles, 1981). The main idea of RANSAC is to randomly select samples of points that contain the minimal number of points that are needed to calculate a parametric description of the sought surface. The parameters of the surface are then calculated and the number of points in the scene that lie on the surface is estimated. This is repeated many times, and the parameters that define the surface with the largest number of points on it are returned.

A.1 Planes

First we describe how to find a plane that contains (within a small tolerance margin) the maximum number of points from the overall set. A plane is uniquely determined by three non-colinear points $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$. With $\mathbf{v}_1 = \mathbf{p}_2 - \mathbf{p}_1$ and $\mathbf{v}_2 = \mathbf{p}_3 - \mathbf{p}_1$, the surface normal is $\mathbf{n} = \mathbf{v}_1 \times \mathbf{v}_2$. The plane is then defined by the equation $\mathbf{n}^T\mathbf{x} + d = 0$ with $d = -\mathbf{n}^T\mathbf{p}_1$. With this in mind, we can search for planes using the basic RANSAC technique shown in Algorithm 2. If this algorithm is executed with a large number of iterations n, it will, with high probability, return the plane containing the maximal possible subset of three-dimensional points from the overall set.

There are two problems with this approach. First, some of the points that belong to other parts of the scene may by chance lie on the plane defined by the physical surface. To get rid of such outliers, we remove all points from the hypothesis that lie far away from its center, i.e. more than twice the standard deviation. Second, if two or more objects with planar surface parts are arranged in such a way that their surfaces lie on one plane, they will be merged into one hypothesis. We therefore apply clustering to the resulting plane, which mostly resolves this problem. We chose the x-means algorithm (Pelleg & Moore, 2000) for feature clustering, which is an extension of the standard k-means algorithm that also estimates the optimal number of clusters.

Even with these extensions, if two objects are placed directly next to each other, the clustering may fail to separate them. Experimental results on that problem can be found in the evaluation section. On the other hand, it may happen that one large object is separated into two hypotheses. Both situations can be resolved by applying pushing actions to the hypothetical surfaces.

A.2 Spheres

The search for spheres is completely analogous to the search for planes. A sphere is uniquely defined by four non-coplanar points. The sphere's center c and radius r

Algorithm 2 Surface detection by RANSAC. Here $m = 3$ in the case of planes and $m = 4$ in the case of spheres; n denotes the number of random samples that are drawn.

```
for i = 1..n do
  Randomly select m different points from the set
  Calculate the parameters of the surface that is defined by these m points
  for all points p in the set do
    if p within tolerance margin around surface then
      Increase counter for inliers
    end if
    end for
  if number of inliers > max then
    Save actual surface parameters as best ones
  end if
end for
Return the parameters of the surface with the maximal number of inliers
```

can be calculated by solving the determinant equation $|\mathbf{M}| = 0$, with $\mathbf{M}$ defined as

$$\mathbf{M} = \begin{pmatrix} \mathbf{x}^T\mathbf{x} & \mathbf{x}^T & 1 \\ \mathbf{p}_1^T\mathbf{p}_1 & \mathbf{p}_1^T & 1 \\ \mathbf{p}_2^T\mathbf{p}_2 & \mathbf{p}_2^T & 1 \\ \mathbf{p}_3^T\mathbf{p}_3 & \mathbf{p}_3^T & 1 \\ \mathbf{p}_4^T\mathbf{p}_4 & \mathbf{p}_4^T & 1 \end{pmatrix} \tag{2}$$

With $\mathbf{M}_{ij}$ denoting the submatrix of $\mathbf{M}$ formed by leaving out row i and column j, the solution is given by

$$\mathbf{c} = \frac{1}{2}\begin{pmatrix} \frac{|\mathbf{M}_{12}|}{|\mathbf{M}_{11}|} \\ -\frac{|\mathbf{M}_{13}|}{|\mathbf{M}_{11}|} \\ \frac{|\mathbf{M}_{14}|}{|\mathbf{M}_{11}|} \end{pmatrix}, \quad r = \mathbf{c}^T\mathbf{c} - \frac{|\mathbf{M}_{15}|}{|\mathbf{M}_{11}|} \tag{3}$$

If $|\mathbf{M}_{11}| = 0$, the four points are coplanar and there is no solution. To find a sphere, we again apply RANSAC. Since more points are needed to calculate sphere parameters than in the case of planes (four instead of three), the random sampling iteration needs to be repeated more often (parameter n of Algorithm 2) to have the same probability of finding the optimal sphere.

A.3 Cylinders

The parameters of a cylinder cannot be computed so easily from a few points on its surface, therefore detecting them is significantly more difficult than detecting planes or spheres. But since many household objects such as bottles or cans have a cylindrical shape, or can be approximated very well by cylinders, we believe that it is worth the effort to be able to find this specific shape.

Beder and Förstner (2006) presented methods that allow the calculation of cylinder parameters given 5–9 points on its surface, but only the methods that require 7 or 9 points return a unique solution. In all cases, equation systems must be solved that require a relatively high computational effort. Moreover, the probability of randomly selecting 7 points belonging to a cylinder is rather small if the overall set consists mainly of outliers, which we have to expect in our case. Therefore, many iterations of standard RANSAC would be necessary.

For this reason, instead of using the direct RANSAC approach, we follow the idea presented in (Chaperon & Goulette, 2001). Here, two nested RANSAC loops are executed, but both of them are less computationally intensive. In the outer loop, promising candidates for the cylinder axis are discovered. In the inner loop, the points are projected onto a plane that is orthogonal to the candidate cylinder axis, which reduces the problem of finding the rest of the cylinder parameters to the search for the best two-dimensional circle within the projected points.

The intermediate step of finding cylinder axis candidates requires the calculation of the Gaussian image of a three-dimensional point set. The Gaussian image consists of surface normals at each of the three-dimensional points. With surface normals having unit length, this is equivalent to a set of points on the unit sphere. We calculate the Gaussian image by approximating the surface normal at every three-dimensional point using its nearest neighbors.

The surface normals of a cylinder form a great circle on the unit sphere (see Figure 11), therefore we search for great circles in the Gaussian image that contain the largest number of estimated normals. The axis of the cylinder is perpendicular to the corresponding great circle. The search for great circles is simplified by the fact that it is equivalent to the intersection of the unit sphere with a plane through its center. Such a plane is uniquely defined by only two normals, i.e. two points on the unit sphere. This allows efficient use of RANSAC for planes to find great circles with maximal support.

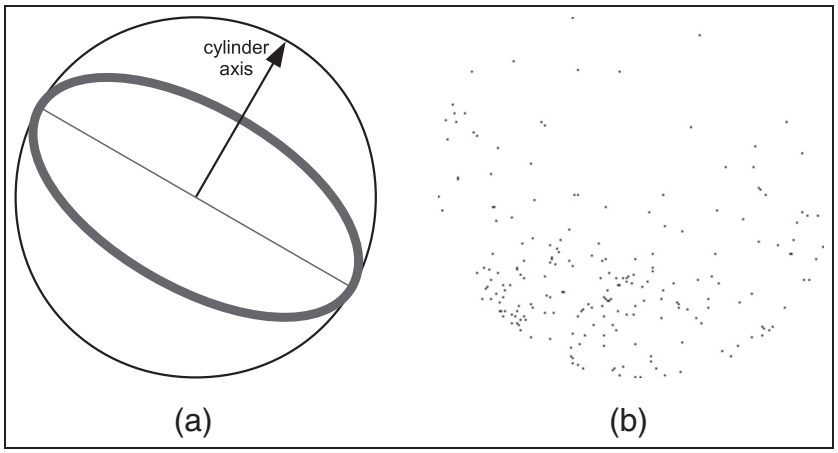

Figure 11. (a) Depiction of how the Gaussian image of a perfect cylinder would look. (b) The Gaussian image resulting from a set of three-dimensional points in a scene containing a cylindrical object on an empty table. Of course only the front side of the object is visible for the cameras, therefore only half of the great circle is contained in the Gaussian image.

Given a cylinder axis, we need to find the radius and offset such that the maximal number of points lie on the cylinder surface. This can be simplified by projecting all points to a plane perpendicular to the axis and searching for the circle that contains the maximal number of the projected points. The search can be further sped up by considering only those points that have contributed to the great circle in the Gaussian image which defined the cylinder axis.

A two-dimensional circle is uniquely defined by three non-collinear two-dimensional points (x_i, y_i) with $i \in \{1, 2, 3\}$. The coordinates of the center of the circle (x_c, y_c) are then given by

$$x_c = \frac{(y_3 - y_2)(x_1^2 + y_1^2) + (y_1 - y_3)(x_2^2 + y_2^2) + (y_2 - y_1)(x_3^2 + y_3^2)}{2\delta}$$
$$y_c = \frac{(x_3 - x_2)(x_1^2 + y_1^2) + (x_1 - x_3)(x_2^2 + y_2^2) + (x_2 - x_1)(x_3^2 + y_3^2)}{2\delta} \quad (4)$$

where

$$\delta = x_1(y_3 - y_2) + x_2(y_1 - y_3) + x_3(y_2 - y_1).$$

The cylinder radius is evidently $r = \sqrt{(x_1 - x_c)^2 + (y_1 - y_c)^2}$, and the offset is an arbitrary point on the line that results from the backprojection of the circle center onto three-dimensional space.

Appendix B

The robot pushing behavior is generated as a sequence of point-to-point movements, which are specified using third-order attractor dynamics (Schaal, Peters, Nakanishi, & Ijspeert, 2005). We applied the following linear system

$$\tau \dot{r} = \alpha_g(g - r) \quad (5)$$

$$\tau \dot{z} = \alpha_z(\beta_z(r - y) - z) \quad (6)$$

$$\tau \dot{y} = z \quad (7)$$

Here y is one of the degrees of freedom that define the complete robot configuration $\mathbf{y}$, and z and r are auxiliary variables. It is easy to show that the above system is critically damped and that it has a unique attractor point at $y = g$, $z = 0, r = g$ for $\alpha_z = 4\beta_z > 0$, $\alpha_g > 0$, $\tau > 0$. System (5)–(7) is suitable for the generation of probing pushes because it is guaranteed to converge to the desired end point, here denoted by g, in a smooth manner regardless of the starting position and perturbations that might arise due to unexpected collisions. In addition, the speed of movement can be modulated with parameter τ and even if the end point g is changed on the fly, the movement remains smooth up to the second order.

To ensure that the object is not occluded by the robot's arm, we move the arm to the home position outside of the robot's view after the push. Thus, images from before and after the push are acquired with the arm removed from the robot's field of view. The complete probing behavior is accomplished by executing a sequence of five dynamic systems (5) – (7), which result in the following movements:

- Relocate the hand from its home position to the position above the starting point for the pushing movement (leftmost image in Figure 7).
- Move the hand towards the starting position for pushing (second image left in Figure 7).
- Move the hand from the starting to the end position, thus generating the probing push (from second to fourth image in Figure 7).
- Move the hand to a position above the end position for pushing (rightmost image in Figure 7).
- Move the arm to the home position away from the robot's field of view.

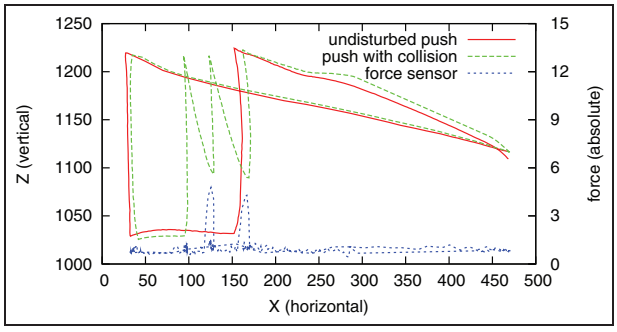

Figure 12. Trajectories of the right hand during two pushes. The trajectory represented by the continuous line was recorded during an undisturbed push, while the dashed one was generated when the hand hit an obstacle twice before it could be lowered besides the object. The forces arising due to collisions with the obstacle are shown by the dotted line.

By moving the hand first to the position above the starting point for the pushing motion, we mostly avoid unexpected collisions with other objects in the scene. It is therefore not necessary to use sophisticated approaches for collision-free path planning, which would be difficult to accomplish in scenes with many unknown objects.

In cluttered scenes, the hand may still collide with other objects, especially during the phase when it is lowered to the starting position for pushing. Such collisions are detected by the force–torque sensor in the wrist of the robot. If such a collision occurs, the hand is raised again and lowered a bit closer to the estimated object position. This is repeated until it could be lowered without collision. Figure 12 shows example trajectories of the hand during pushes with and without collisions. If the hand collides with an obstacle several times, the correcting movements eventually bring it very close to the object. When the hand is positioned with high probability above the object, the robot moves it down until contact with the object and executes a sliding movement instead of a push. This reactive strategy enables the robot to move the objects even in very difficult situations.