2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

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

David Schiebener, Aleš Ude and Tamim Asfour

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

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

I. INTRODUCTION AND RELATED WORK

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

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

D. Schiebener and T. Asfour are with the Institute for Anthropomatics and Robotics, High Performance Humanoid Technologies Lab (H^{2}T), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany. A. Ude is with the Humanoid and Cognitive Robotics Lab, Jožef Stefan Institute, Ljubljana, Slovenia.
schiebener@kit.edu, asfour@kit.edu, ales.ude@ijs.si

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

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

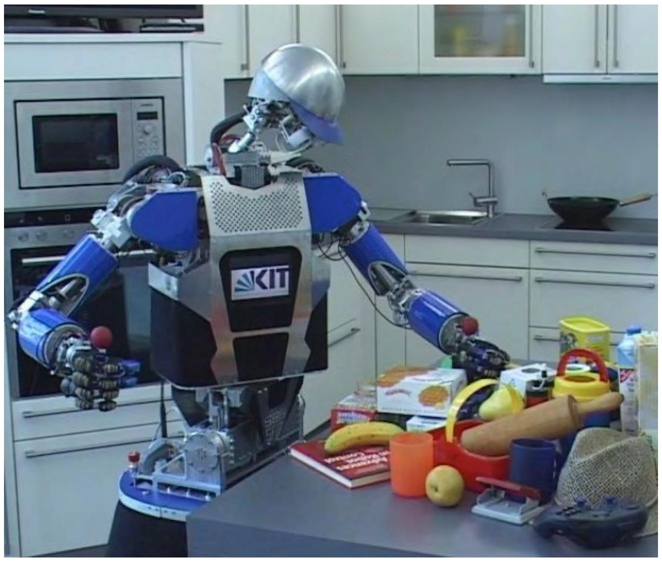
Fig. 1: Interactive object segmentation performed by the humanoid robot ARMAR-III [2]. By pushing unknown objects, they can be segmented from the environment based on the induced motion.

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

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

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

The following section will give an overview of our approach, which will be explained in detail in sections III and IV. In section V we present the results of our experiments on the humanoid robot ARMAR-III, and section VI concludes the paper.

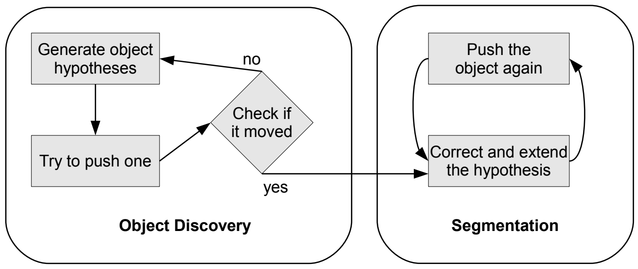

Fig. 2: System overview: Our approach can be divided into two main phases. First, the robot generates object hypotheses and tries to verify one of them by moving it. If an object has been discovered, the segmentation is improved and different views can be learned in the course of several further pushes.

II. PHYSICAL INTERACTION FOR SEGMENTATION

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

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

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

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

III. INITIAL OBJECT DISCOVERY

A. Generation of object hypotheses

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

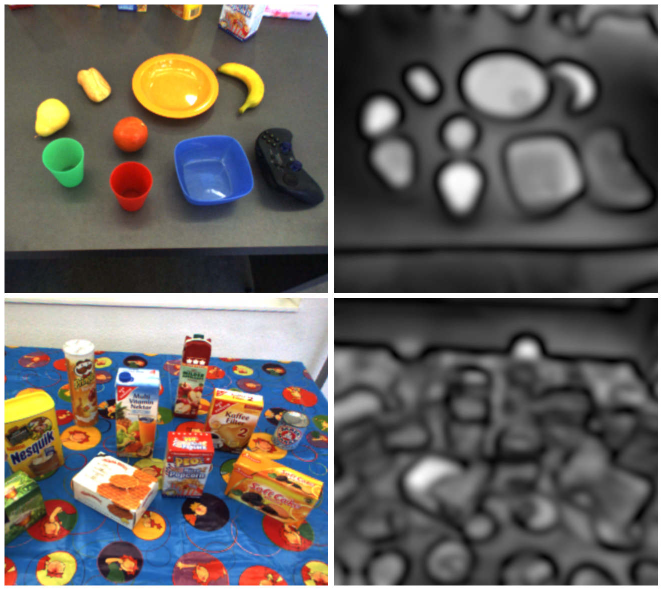
Fig. 3: A relatively simple and a confusing scene with their respective saliency images. As can be seen, the algorithm for saliency computation is not of much use in scenes where objects and background are equally rich in colors and contrasts.

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

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

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

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

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

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

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

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

B. *Pushing for Verification*

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

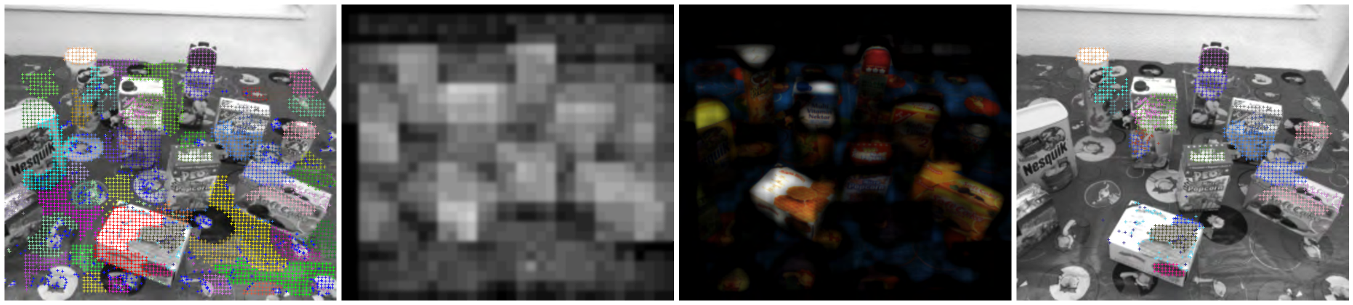
Fig. 4: Suppression of object hypotheses that do not lie in a region that is higher than its direct surroundings. The first image shows a complex scene that leads to the creation of very many initial object hypotheses. The second image displays the map quantifying the relative local height of the image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

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

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

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

C. Detection and Analysis of Change in the Scene

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

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

A first result we get immediately from this difference is an answer to the question if anything happened at all. If nothing changed in the scene, the robot was evidently unable to move the potential object or did not hit anything at all. In this case, the robot tries pushing another object candidate. If a change in the scene is detected, all initial object hypotheses are analyzed on whether they lie in image regions that changed. Each object hypothesis is represented by a set of RGBD points, and if more than half of them lie in a region that changed due to the push, the hypothesis will be analyzed for having moved; otherwise it is discarded. In addition to the initial hypotheses, we create new ones from the points that changed. This is done by determining 2-5 clusters[1] amongst these points using x-means, a variant of k-means that automatically chooses the number of clusters [21]. These new hypotheses frequently match the actual object better than the initial ones, although usually not perfectly either.

IV. OBJECT SEGMENTATION

A. Estimation of the Object Motion

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

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

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

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

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

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

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

Thus, we execute ICP several times with different initial estimates of the new object pose, and keep the resulting transformation that yields the best match. As the object may have been moved over a large distance, finding it again requires an appropriate choice of the initial poses for ICP. To this end, we detect image regions that resemble the hypothesis in terms of color histogram similarity and initialize the alignment there. If the object surface contains stable local visual features, those can be used to get an initial estimation of the motion, too. The necessary number of different initial positions can be reduced by taking into account the direction of the push, which must not be done in a too restrictive manner as the caused object motion is rather unpredictable.

The best transformation returned by the differently initialized registration attempts is refined by another execution of ICP on a reduced point set where all those points are left out that still have a large distance to their nearest neighbor. The resulting final transformation is used to decide whether the estimated object motion is accepted, and if this is the case, to determine the object segmentation.

B. Verification, Correction and Extension of the Segmentation

After the motion of an object hypothesis has been estimated, the robot needs to decide whether the determined match and transformation are plausible. A hypothesis is only accepted, i.e. considered to correspond to an actual object, if it meets the following three criteria: First, the estimated motion has to be large enough to be sure that it is not due to noise or a slight mismatching[3]. Secondly, the match must be good, i.e. the average distance of the hypothesis points to their respective nearest neighbors in cartesian and normalized RGB space must be below a threshold. Thirdly, the relocalized hypothesis must lie mostly in image regions that have changed. This removes mismatches where by pure chance a good alignment to some part of the scene could be found, e.g. a part of the table surface that was matched to another part of the table after the object has been moved onto it.

The remaining hypotheses do most likely belong to an actual object that has been moved by the robot. But of course we must assume that they do not cover the object completely, and that they also contain points that do not belong to the object. We remove the latter ones by checking each point of the hypothesis: After applying the estimated object motion, a point must match its nearest neighbor in the scene point cloud well with respect to cartesian and color distance. It also has to lie in a region that changed due to the push. If both of these criteria are met, the point is considered to be verified, otherwise it is removed from the hypothesis.

After removing the false points, we try to extend the hypothesis to cover the whole object. To this end, we add all those points to it as candidates that lie close to the verified points and within the image region that changed. By pushing the object again and repeating the steps described before, these new candidate points can be verified or discarded, and new candidates can be added. Depending on the object size and the quality of the initial hypothesis, it usually takes two or three pushes until the whole object is contained in the hypothesis and thus segmented completely.

Usually, more than one object hypothesis is verified by the first push and the subsequent analysis. This happens

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

[3]Given the precision of our stereo calibration and a distance of 50-80 cm between camera and object, a threshold of 3 cm turned out to be definitely safe.

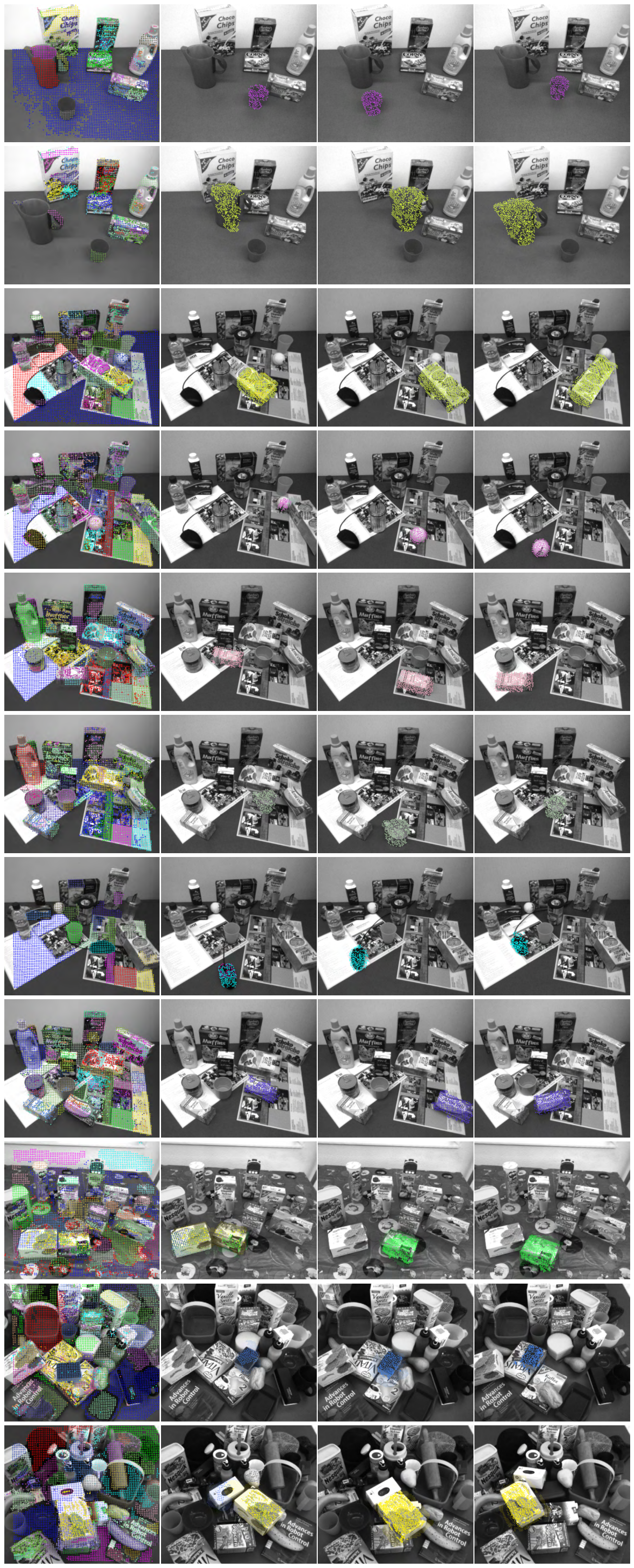

Fig. 5: Examples of object segmentations in different scenes. The first image in each row shows the initial object hypotheses, the second to fourth images show the verified hypothesis after one, two and three pushes.

in particular when several actual objects are moved. We choose the hypothesis containing the maximal number of confirmed points for the second push. After that, we discard the hypotheses that did not move again, and from the remaining ones we keep only the one with the maximal number of confirmed points and continue examining it as long as desired. If the robot did indeed move several objects, all of them can be segmented, but for the sake of simplicity we only observed one in our experiments. As long as the objects undergo different 3D transformation, they can easily be separated based on their different motion. It may happen though that two objects move exactly alike, in which case they are subsumed in one hypothesis. Most likely they are separated when pushed several times from different directions. Heuristics for systematical pushing to this end have been proposed in [5] and [7]. When two objects contained in one hypothesis are separated, the hypothesis will follow the object that is matched better after the motion, which is usually the bigger one.

Pushing an object several times will reveal different sides of it, thus the creation of a multi-view object descriptor is possible, although some sides will probably never be observed. In section V-D we demonstrate that the obtained segmentations are well suited to train an object descriptor that allows for reliable recognition.

V. EXPERIMENTAL EVALUATION

A. *System Setup*

We have implemented and tested our approach on the humanoid robot ARMAR-III [2]. The video accompanying this paper shows an interactive object segmentation executed by it. The robot has an active stereo camera system in its head, and its arms have seven degrees of freedom each and are equipped with force-torque sensors in the wrists. The cameras provide color images with a resolution of 640×480 pixels. About 85% of the stereo images overlap, and after calculating the dense depth map we use only every second pixel in x and y direction for the point cloud, thus we obtain around 65000 RGBD points that we work with.

The computational effort is dominated by the relocalization of the object hypotheses using the ICP algorithm, in which the computational complexity is proportional to $n\ log(m)$, with n being the number of points of the object hypothesis and m the overall number of points in the scene. On a 3 year old standard PC with a quadcore processor, the computations after each push took between 2 and 5 seconds, depending on the size and number of moved objects.

An important aspect in comparison to some related work is that in our case the robot itself executes the object pushing, and we do not use an artificial setup where the camera always has an undisturbed view of the object. This is the reason why we do not try to track the object during the push, as the robot's hand frequently occludes large parts of it.

B. *General Observations*

Our approach aims at making it possible to segment rigid objects independently of their appearance or shape, thus we tested it with a large variety of items. They can roughly be classified by their visual appearance as being strongly textured, sparsely or partially textured, multicolored but (almost) non-textured, unicolored, reflective (e.g. polished metallic objects or mirrors), or transparent. As far as we know, the related work in this field (including ours) has so far either depended on local features, i.e. texturedness, on unicoloredness, or on a certain shape.

It turned out that our segmentation approach works very well for all kinds of objects except the very reflective and the transparent ones. This is due to the fact that they appear to change their color when moved, and also tend to cause problems when trying to obtain depth information. All other objects were segmented successfully by our approach; for the transparent and reflective ones a special treatment might be necessary. Although we were able to tune the parameters of the background subtraction and the matching so that the segmentation worked for most of them, it does not function reliably and the chosen parameters depend strongly on the lighting conditions, thus we do not claim that our approach can handle this kind of objects.

In contrast, the shape of the examined objects did not seem to make an observable difference. While distinctive shape features are necessary for algorithms that match point clouds solely based on 3D data, the fact that we use color helps to overcome ambiguities that might arise otherwise. The combined use of shape and color information usually allows a good alignment of the object hypothesis with the object after it has been pushed. An exception here are symmetric unicolored objects, but in that case it actually does not matter if the orientation around the axis of symmetry is met correctly as long as the match is good. The only case in which problems occurred was when a flat, unicolored object was placed on a table of the same color.

C. Assessing the Segmentation Quality

We examined the performance of our interactive segmentation approach by testing it with 30 objects of different shape, size and visual appearance type (as defined above), which have been segmented twice each. To measure the quality of the obtained segmentations, two metrics are determined: First, the object should be segmented as completely as possible, i.e. in an optimal case the point cloud forming the object hypothesis should fully cover the object. The second metric is the size of the falsely segmented area, i.e. the part of the scene that is segmented but does not belong to the object. This happens when the object hypothesis includes points that belong to the background or other objects.

Figure 6 shows these two values depending on the number of pushes executed. As can be seen, after the first push the object is usually not covered completely, but already to a large part. After two to three pushes, the hypothesis contains almost the complete object, with the exception of small patches that newly appeared due to object rotation or that were discarded from the hypothesis due to a change in their appearance (e.g. reflections or bad depth estimation). After four or more pushes, the coverage does not improve further, but different parts of the object may become visible, thus more information can still be gained.

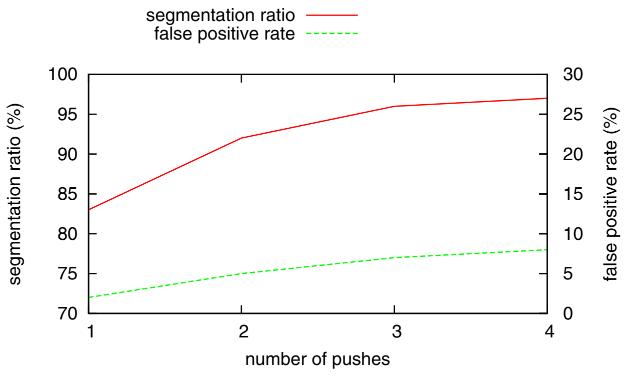

Fig. 6: The average segmentation quality depending on the number of pushes that were executed. The red line shows the segmentation ratio, i.e. the percentage of the object that is included in the segmentation. The dashed green line depicts the false positive rate, i.e. the fraction of the segmentation that does not belong to the actual object.

The ratio of falsely segmented image regions compared to the whole object is always quite small. It seems to grow a bit from the first to the second push, but not any further afterwards. Such false positives occur when the shadow cast by the object leads to neighboring image regions being considered to have changed, and some of them look alike before and after the push, which frequently happens on unicolored table surfaces. In this case, the part of the table on which the object casts a shadow appears to belong to the object itself. We are not sure whether there is a theoretically sound solution for this specific ambiguity; it is probably necessary to grasp and lift the object to dissolve it.

D. Learning of an Object Descriptor

To demonstrate that the obtained segmentations are sufficiently complete and correct, we use them to train a simple object recognition system. The available information we can use are the image region that contains the object hypothesis, i.e. the segmentation, as well as the 3D and color information contained in the hypothesis point cloud itself. After each push, the object hypothesis and thus the segmentation are different, therefore we could generate several descriptors for each object from different perspectives. For the sake of simplicity, we just use the segmentation obtained after the second push for each object, which usually yields a good coverage, and generate only one descriptor.

To detect the learned objects in new images, we train a color histogram based descriptor using the image region that is occupied by the object hypothesis. The descriptor uses Receptive Field Cooccurrence Histogram (RFCH) features [23], [24] which are based on histograms of the colors and their first and second derivatives in the segmented image area.

These features allow to find image regions that have the same color distribution as the learned object. We then try to

TABLE I: Object recognition rates.

similar point of view	different point of view	partly occluded	false positive rate
98.5 %	70.6 %	67.2 %	3.8 %

match the learned RGBD point cloud in those areas using Iterative Closest Point (ICP) as in the motion estimation step of our segmentation approach. The localization is accepted if the resulting average point distance in Cartesian and color space is below an equivalent of 1 cm (with the maximal possible color distance being equivalent to 10 cm in Cartesian space).

Table I displays the recognition results for our set of autonomously learned objects. They are placed in potentially confusing scenes comparable to those shown in figure 5. When the object is seen from approximately the same point of view as during learning, the recognition rate is almost 100%. If the object has a significantly different orientation with relation to the camera, or if it is partly occluded by other objects, the recognition rate drops to around 70%. This can be improved by using object descriptors generated from different views, as we did in [11]. The false positive rate is about 4%, which is entirely due to two small unicolored objects in our test set that are sometimes fitted into blobs of similar color. These solid recognition results demonstrate the usefulness and quality of the segmentations obtained by the robot following our approach.

VI. CONCLUSIONS

We have presented a new approach for interactive object segmentation exploiting the manipulation capabilities of a humanoid robot. The proposed method enables it to discover and segment unknown rigid objects in an unknown, complex scene by pushing them and analyzing the motion of color-annotated 3D points obtained from the robot's stereo vision system. We have demonstrated that the provided segmentation results are of excellent quality and allow to train a well performing object recognition system. As already shown in [14], it is also possible to subsequently grasp the discovered objects for further examination or manipulation.

In contrast to our previous work in this direction, the approach proposed here works with almost any kind of rigid object except those which are transparent, highly reflective or impossible for the robot to move. We therefore believe that it is a small but important step for increasing the adaptability and autonomy of humanoid robots that will frequently have to deal with new, unknown objects in realistic scenarios.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement 270273 (Xperience).

REFERENCES

[1] G. Metta and P. Fitzpatrick, "Grounding vision through experimental manipulation," *Philosophical Transactions of the Royal Society: Mathematical, Physical and Engineering Sciences*, vol. 361, no. 1811, 2003.

[2] T. Asfour, K. Regenstein, P. Azad, J. Schröder, A. Bierbaum, N. Vahrenkamp, and R. Dillmann, "ARMAR-III: An integrated humanoid platform for sensory-motor control," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2006.

[3] D. Katz and O. Brock, "Manipulating articulated objects with interactive perception," in *IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, CA, 2008.

[4] W. H. Li and L. Kleeman, "Segmentation and modeling of visually symmetric objects by robot actions," *International Journal of Robotics Research*, vol. 30, no. 9, 2011.

[5] L. Chang, J. Smith, and D. Fox, "Interactive singulation of objects from a pile," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 3875–3882.

[6] M. Gupta and G. Sukhatme, "Using manipulation primitives for brick sorting in clutter," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 3883–3889.

[7] T. Hermans, J. Rehg, and A. Bobick, "Guided pushing for object singulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2012.

[8] E. S. Kuzmič and A. Ude, "Object segmentation and learning through feature grouping and manipulation," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2010.

[9] D. Schiebener, A. Ude, J. Morimoto, T. Asfour, and R. Dillmann, "Segmentation and learning of unknown objects through physical interaction," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Bled, Slovenia, 2011.

[10] A. Ude, D. Schiebener, N. Sugimoto, and J. Morimoto, "Integrating surface-based hypotheses and manipulation for autonomous segmentation and learning of object representations," in *IEEE International Conference on Robotics and Automation (ICRA)*, St. Pauls, Minnesota, 2012.

[11] D. Schiebener, J. Morimoto, T. Asfour, and A. Ude, "Integrating visual perception and manipulation for autonomous learning of object representations," *Adaptive Behavior*, vol. 21, no. 5, pp. 328–345, 2013.

[12] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. Int. Conf. Computer Vision*, Corfu, Greece, 1999.

[13] P. Forssen, "Maximally stable colour regions for recognition and matching," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[14] D. Schiebener, J. Schill, and T. Asfour, "Discovery, segmentation and reactive grasping of unknown objects," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2012.

[15] K. Hausman, F. Balint-Benczedi, D. Pangercic, Z. Marton, R. Ueda, K. Okada, and M. Beetz, "Towards tracking-based interactive segmentation of textureless objects," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[16] S. Frintrop, E. Rome, and H. Christensen, "Computational visual attention systems and their cognitive foundations: A survey," in *ACM Transactions on Applied Perception 7*, 2010.

[17] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998.

[18] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk, "Frequency-tuned salient region detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 1597–1604.

[19] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, Feb. 2008.

[20] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Efficient inverse kinematics computation based on reachability analysis," *International Journal of Humanoid Robotics*, vol. 9, no. 4, 2012.

[21] D. Pelleg and A. Moore, "X-means: Extending k-means with efficient estimation of the number of clusters," in *Proc. 17th Int. Conf. Machine Learning*, San Francisco, CA, 2000.

[22] P. Besl and N. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[23] K. Welke, "Memory-based active visual search for humanoid robots," Ph.D. dissertation, Karlsruhe Institute of Technology (KIT), Computer Science Faculty, Institute for Anthropomatics (IFA), 2011.

[24] S. Ekvall and D. Kragic, "Receptive field cooccurrence histograms for object detection," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2005, pp. 84–89.