Partial 3D Reconstruction of Objects for Early Reactive Grasping

Niklas Bergström
Partial 3D Reconstruction of Objects for Early Reactive Grasping

Niklas Bergström and Danica Kragic
Centre for Autonomous Systems
Computational Vision and Active Perception Lab
School of Computer Science and Communication
KTH, Stockholm, Sweden
{bergst,danik}@kth.se
July 30, 2009

Contents
1 Introduction 2
2 Related work 3
3 System SetUp and Grasp Modeling 4
4 Generating Grasping Hypotheses 5
  4.1 Dynamic Time Warping 5
  4.1.1 Adaptive Weighting 7
  4.2 Contour Extraction 7
  4.3 Matching 8
  4.3.1 Contour Level Matching 8
  4.3.2 Plane Level Matching 10
5 Evaluation 11
6 Conclusion 13
Abstract

Grasping unknown objects in natural environments remains an open problem. An important question is how we devise simple, fast and robust methods that rely on minimal representations of objects. In this report, we present a stereo system that generates grasping hypotheses based on curve matching between two views. The contribution of the work is the use of dynamic time warping approach for enforcing the similarity between the curves in stereo images. This serves as the base for partial reconstruction of objects in 3D. We further show how the system can be used to generate elementary grasping actions or early reactive grasps.

1 Introduction

Robots performing tasks in natural environments need the capability to deal with manipulation and grasping of both known and unknown objects. It has been recognized recently that grounding in the embodiment of a robot and continuous interaction with the environment are required to facilitate learning of objects and object categories, [1]. The idea is that robots will not be able to form useful categories or object representations by only being a passive observer of its environment, but by interacting with it.

Our previous work in this context considered both grasping of known and unknown objects, [2–4]. In this report, we continue the work on grasping of unknown objects. We note that the purpose of this work is not to develop yet another grasping strategy for a specific setting, but rather to provide low-level grasping reflexes that can be used to generate successful grasps on various objects.

We present a method for generating a representation based on planar surfaces in 3D space, and focus on objects of uniform color or simple texture using a stereo vision system. The result is a partial 3D reconstruction of the object. This report draws inspiration from our previous work on tracking, where Dynamic Time Warping (DTW) was used to match hand contours in stereo images [5]. The contribution of the work presented here is the extension of the method to handle non closed contours and remove the prerequisite that each contour corresponds to exactly one contour in the other image. Also, we present examples of how the method performs for different types of objects, and how the resulting representation can be used when generating grasps. In summary, we concentrate on partial 3D reconstruction of objects with little or no texture.

With the work presented here we want to push the minimum representation to the edge: How little information is really needed to be able to pick up an object? For our approach we suggest that it is not necessary to perform full reconstruction in order to generate successful grasps. We are merely interested in finding one or more parts of the object that are graspable. Since we assume objects with no or little texture, we follow the hypothesis that most prominent edges belong to the outer contours of the object. We first find corresponding edges in both images, and then point correspondences along the paired contours. We finally generate 3D points from the correspondences, which are then used to generate plane hypotheses suitable for grasping.

This report is organized as follows: related work is reviewed in Section 2. In Section 3, different aspects of the grasping system are presented and the overall methodology outlined in Section 4. The experimental evaluation is given in Section 5 and the report is finally concluded in Section 6.
2 Related work

There has been a significant amount of work presented in the area of robotic grasping during the last two decades. Some of the work deals with analytical methods where the shape of the objects being grasped is known a-priori. Such work has focused primarily on computing grasp stability based on force and form-closure properties or contact-level grasps synthesis based on finding a fixed number of contact locations.

Examples that have been demonstrated in realistic scenarios, such as [6], commonly use models of a-priori known objects. Work on grasping of unknown objects deal with different types of representations. In general, the representation is affected by the complexity of the scene and type of the sensory data. An often forgotten problem is that the representation should also allow for the grasping module to be integrated with a grasp planning step, that is we need a representation that allows for different approach directions towards an object. Our work is a step in this direction.

Different approaches vary from appearance based methods where grasping points are detected locally in 2D images [7] and approaches that rely on 3D data. In the latter case, 3D information is generated with vision or with laser range scanners. One common approach is to fit simple shapes to the 3D data, like boxes [8] or superquadrics [9, 10]. The grasping problem is then reduced to grasp a primitive, which can be learned offline. When vision is used, highly textured objects are needed in order to be able to retrieve a dense point cloud. In typical household scenarios, however, objects of interest such as plates, cups or books are often uni-colored or even transparent, which makes vision based 3D reconstruction hard. One of the solutions to this problem is the use of structured light, [11]. While they might be effective, these approaches are commonly not applicable in a regular camera setup. There have also been approaches where shape retrieval is based on several views and contour extraction using the visual hull [12, 13]. This approach however requires that either the robot moves or that the object rotates relative to the camera to retrieve the necessary views, which is time consuming or impossible to achieve for some configurations.

Grasp generation techniques based on primitive shapes are common, [8, 10, 14] and there are also examples where machine learning techniques are used for extraction of grasping points in 2D images, [7]. There are also examples of automatic grasp synthesis and planning, [14], [15], [16], [17]. This work concentrates on automatic generation of stable grasps given assumptions about the shape of the object and robot hand kinematics. One related problem is that these methods often assume that a grasp can be applied from the side or from above which is not always resulting in the most stable grasp. With our method, we are able to generate approach vectors for grasping the object from any angle.

Related to the work presented here, we also need to review the work on contour matching, which has been an active research field for a long time in the area of computer vision. It has been applied to problems like recognition, tracking and stereo matching. While some work is
done on matching in feature space, like [18], where the features are straight lines, most work perform the matching directly on the contours [19], or on parameterization or sampling of the contours [20], [21]. The latter techniques, whether matching closed contours [22], [20], [23], or open contours [21], [24], view one contour as a deformation of the other one. The amount of deformation that is required gives a dissimilarity measure between the curves. The key concept is the usage of a matching curve \( f(k) = (i(k), j(k)) \) that maps point \( i \) on the first contour to point \( j \) on the second contour. The dissimilarity is then a function \( d(f) \) that gives a value for the dissimilarity between point \( i \) and \( j \). The DTW algorithm used in this work is based on this concept.

There are different approaches on how to compute this dissimilarity. One way is to consider contours as strings from an alphabet. Work based on the Edit Distance (ED) [25], or variants [26], define the dissimilarity based on the number of edit operations required to transform one contour to the other. Other work look at a combination of the difference in curvature and how much the matching curve is stretched [27], [21], [20]. Algorithms based on these measures often use dynamic programming, which is also the case for DTW.

In our approach, we rely on stereo and thus the contour in one image is a perspective projection of the contour in the other image\(^1\). Since we need the epipolar geometry for segmentation, we use this as a base for the dissimilarity measure as well. While [21] use intrinsic properties of the contours in their dissimilarity measure, they still rely on an estimated epipolar geometry when performing matching. In our work we acknowledge the importance of measures like curvature, but use them only to resolve disambiguates in the first phase of the matching. In this way we get good performance when initially selecting matching contour points, while being able to densely match segments with low curvature.

In our work, we also need to solve the correspondence problem: both in terms of the whole contours and their parts. Most of the existing work describes only the matching of two selected contours. In [21] they address the issue of contour correspondence, but when performing matching of two contours, they do not address how to handle segments from the matching that remain unmatched. As described in Section 4.3, this is crucial in our approach and we describe a solution to the problem.

3 System SetUp and Grasp Modeling

In our approach, the process of grasping assumes three different steps: i) identification, ii) feature extraction, and iii) grasping, see Fig. 1. The first step involves identifying and segmenting the objects, while the last step deals with generating grasps from the object representation generated in the previous step. In our previous work, we have dealt with both of these, [4]. Our current work focuses on the second step. In particular, in [28] we have presented how individual objects are segmented from the background which is an important part of the approach shown in Fig. 1. The system uses a combination of peripheral and foveal cameras for this purpose.

Planar regions and coplanar relationships afford different types of grasps. In our work we consider five elementary grasping actions (EGA), as shown in Fig. 2. EGA1 is a “pinch” grasp on a thin edge like structure with approach direction along the surface normal of the plane spanned by the primitives. EGA2 is an “inverted” grasp using the inside of two edges with approach along the surface normal. EGA3 is a “pinch” grasp on a single edge with approach direction perpendicular to the surface normal. EGA4 is similar to EGA2 but its approach direction is perpendicular to the surface normal. Also it tries to go in “below” one of the primitives. EGA5 is wide grasp making contact on two separate edges with approach direction along the surface.

---

\(^1\)This is not entirely correct, as the cameras will be looking at two different contours for circular objects. For our purposes though, this approximation is valid.
The EGAs are parametrized by the final pose (position and orientation) and gripper configuration. For the simple parallel jaw gripper, an EGA will thus be defined by seven parameters: $\text{EGA}(x, y, z, \gamma, \beta, \alpha, \delta)$ where $p = [x, y, z]$ is the position of the gripper “center” according to Fig. 3; $\gamma, \beta, \alpha$ are the roll, pitch and yaw angles of the vector $\mathbf{n}$; and $\delta$ is the gripper configuration, see Fig. 3. Note that the gripper “center” is placed in the “middle” of the gripper. The main motivation for choosing these grasps is that they represent the simplest possible two fingered grasps humans commonly use.

\section{Generating Grasping Hypotheses}

Full 3D reconstruction of objects with little or no texture from stereo vision is a difficult problem. It is though questionable how often a full reconstruction of an object is needed for grasping, something noted by Speth et al. \cite{12}. The general idea is that many objects in a household scenario, including cups, plates, trays have planar surfaces. The idea is then to generate hypotheses of these surfaces and use them for initial grasping. After the robot has picked up an object, it can move it in front of its eyes in a controlled manner to retrieve a more detailed representation of the object, \cite{4}.

In our approach, contours of an object are reconstructed from a pair of stereo images. The contours that belong to the same surface are grouped by generating hypotheses of planes and grouping points that belong to the same hypothesis.

\subsection{Dynamic Time Warping}

Dynamic Time Warping (DTW) is commonly used for aligning two sequences, for example sound signals for speech recognition. In the computer vision field it has both been used for tracking of deformable contours \cite{29} and contour based shape retrieval \cite{22}. In this work we use the
algorithm for the matching problem of vision based 3-D reconstruction. We build upon our earlier work presented in [5].

The algorithm includes the following steps.

1. Select two contours and compute the distance matrix $D$, i.e. a matrix holding the distance $d(i, j)$ between each pair of points $(p_i, p_j)$ on the contours.

2. Select one pair of points that seems to correspond to each other as a starting point.

3. Compute the accumulated distance matrix $D_{acc}$, i.e. a matrix holding the distance between each pair of points and the starting point. This is calculated with dynamic programming, see Eq. 1.

4. Search from the end pair in the accumulated distance matrix, the path with pairs of minimum accumulated distances until the start pair is reached. This path corresponds to the optimal match.

The accumulated distance matrix is calculated according to the following. From each matrix entry, corresponding to a pair in the sequence, it is possible to either take one step forward in both sequences, referred to here as matching, or take one step in either of the sequences while standing still in the other, referred to here as alignment. Thus each pair, except pairs containing the first position of each sequence, can be reached in three ways, and the cost of reaching a pair can be expressed as:

$$D_{acc}(i, j) = D(i, j) + \min\left\{ \begin{array}{l} P \cdot D_{acc}(i - 1, j) \\ P \cdot D_{acc}(i, j - 1) \\ D_{acc}(i, j) \end{array} \right.$$  \hspace{1cm} (1)

Here, $D_{acc}$ is the entry in the accumulated distance matrix, $D$ is the entry in the distance matrix, and $P$ is a cost for making alignments, see Section 4.1.1.

If the sequences are a perfect match, only matching steps will be taken. This is however rarely the case. For example in the 3D case, the cameras will look at one edge from different perspectives, resulting in for example one edge shorter than the other, or with less curvature. In these cases alignment steps are taken to correct the dissimilarities.

Which distance measure to use depends on the application. In the case of 3D reconstruction from stereo image pairs, epipolar geometry has been used before [5]. We have also adopted a distance measure denoted $d(i, j) = f(p_i, p_j)$ between points $p_i$ and $p_j$. The distance is calculated using the definition of a epipolar line $e$ as follows

$$e = Fp^h_j$$

$$f = \frac{e \cdot p^h_i}{e_x^2 + e_y^2}$$  \hspace{1cm} (2)

Here $p^r$ and $p^h$ are the homogeneous coordinates of corresponding image points. There is however a problem with this measure. When a contour is parallel or close to parallel to the epipolar line, the distance is roughly equally small for a large range of contour points, and the 3D contour often gets the characteristic look of Fig. 4. For straight edges this can be handled by tilting the robot’s head, but with circular edges the problem cannot be circumvented like this. We propose a solution to this as presented in next section.
4.1.1 Adaptive Weighting

In our previous work we have used a cost for alignment transitions. This is done in order to favor matching transitions whenever nothing unusual happens such as in case of occlusions. We adopt this cost as well, but instead of letting it be constant, we adapt it to be higher for problematic pairs in order to guarantee matching for segments where the distance measure is not reliable. If however using a constant high cost, there will never be necessary alignments, and the method will fail. The cost is inversely proportional to the difference in orientation between the epipolar line for the current point in the right image, and the contour in the left image as follows:

\[
P = \begin{cases} 
1.5 + \frac{1}{\alpha (e' - p')} & e' - p' > 0.1 \\
1.5 + \frac{1}{\alpha 0.1 + \beta} & \text{otherwise}
\end{cases}
\]

The reconstruction of the contour is unstable when the difference between the epipolar line orientation, \( e' \), and the orientation of the contour in the current point, \( p' \), is less than 0.3 radians, why we want a steep increase in the weight in these cases. Based on our previous work, we keep the cost equal to 1.5. Running a number of experiments in different settings, we found \( \alpha = 10 \) and \( \beta = -0.6 \) to give good performance. In addition, we only allow alignments each third step under the condition that \( (e' - p') < 0.3 \).

When using adaptive weighting the resulting contour is much smoother and it also deviates less in the direction of the normal of the plane, see Fig. 4. This, in its turn, generates better plane hypotheses in the next step.

![Figure 4: The right images show the top view and a side view of a circular contour without adaptive weighting. The left images show the same contour warped with adaptive weighting. These contours are taken from the upper rim of the cup in Fig. 7](image)

4.2 Contour Extraction

For acquisition of edges, the Canny edge detector is used. Since DTW works on sequences, in this case of contours, an exhaustive search for connected edgels is performed. One set of contours, \( C = \{c_k\} \), is formed for each edge image:

\[
c_k = \{p_i, p_j | \mathcal{N}(p_i, p_j)\}
\]

where \( \mathcal{N}(p_i, p_j) \) denotes the neighbors if

\[
x_{\text{diff}}(p_i, p_j) \leq 1 \land y_{\text{diff}}(p_i, p_j) \leq 1
\]

Inherit to the Canny, it is possible for edges to have branches. For each contour we therefore first remove shorter stubs, which are shorter than three pixels. After that, for all the contours, if

\[
\mathcal{N}(p_i, p_j) \land \mathcal{N}(p_i, p_k) \land \mathcal{N}(p_i, p_l), \ i \neq j \neq k \neq l
\]
then we split on \( p_i \). Finally, the orientation with respect to the contour’s direction and curvature are calculated for each contour point as

\[
\text{orientation}(p_i) = \arctan\left(\frac{y_{\text{diff}}(p_{i-2}, p_{i+2})}{x_{\text{diff}}(p_{i-2}, p_{i+2})}\right) \quad (4)
\]

\[
\text{curvature}(p_i) = \max_i\left(\frac{v_1^Tv_2}{\|v_1\|\|v_2\|}\right) \quad (5)
\]

where

\[
v_1 = p_{i-j} - p_i, \quad v_2 = p_{i+j} - p_i, \quad j = 1..N \quad \text{and} \quad N \leq \# \text{ contour points}/4
\]

A rough filtering based on the curvature summed over each contour is also performed. The filtered contours are unlikely to correspond to an edge, but rather some texture on the object. The procedure is repeated for both left and right images.

### 4.3 Matching

The matching step is a new part of the work presented here compared to our previous work. In the original work, we have solved the correspondence problem by matching the centroids of tracked objects. In the work presented here, the correspondence problem is not trivial since Canny very often breaks the edges differently in the corresponding images, see Fig. 5. This is affected by lighting conditions and different Canny parameters. However, the contour segments of interest often consist of a large number of edgels, and the number of contour segments are few compared to the number of edgels. This is why even an exhaustive search for corresponding contour segments does not impact performance significantly. As previously stated, long contour segments are assumed to belong to the object, motivating the reason why they are sorted according to length.

Figure 5: Two examples how edges break differently in the left and right images under two different Canny parameter settings.

Matching is made on two different levels: Contour level, where DTW is used to find the most likely pairs of contours and to generate a 3D contour, and surface level, where the contours are grouped into ones belonging to the same surface.

#### 4.3.1 Contour Level Matching

The contour level matching solves i) the correspondence problem: which contour corresponds to which, as well as ii) the mapping problem: which point in one contour corresponds to which point in the other contour. Longer contours are in our setting more likely to belong to an object.
This is why the contours are sorted based on length, and the algorithm sorts contours in that order. The algorithm is similar to the general DTW algorithm presented in Section 4.1, but with some modifications. In the original implementation, the correspondence problem was solved to start with and it was assumed that the two contours could be matched. Furthermore, since the original application was hand tracking, the first frame was used as an initial estimate. In that case, even if the wrong starting point was selected, there was still a part of the contour that was correctly matched, and a good starting point pair was possible to be chosen in the next frame.

In our current application, for one contour in the right image we search for possible corresponding contours in the left image. The starting point selection proceeds as follows:

- Select one point \( p_i \) on the right contour \( c^r \).
- For each point \( p_j \in c^l \) estimate \( d(i,j) \) using Eq. 2
  - iff \( d(i,j) < 3 \), add \( p_j \) to \( P \)
- Select from \( P \), \( p_j \) that are local minima
- Among the local minima, select \( p_j \) with most similar curvature and orientation to \( p_i \).

In our earlier work, [5] it was known that the starting point and the end point are neighbors, since the contour was closed. It was therefore easy to backtrack from the end point to the starting point of the accumulated distance. In the current implementation, we are commonly faced with situations like in Fig. 6 left, where the upper part of the right contour matches the lower part of the left contour. Therefore we adjust step four of the algorithm in Section 4 according to the following when matching two contours \( c^l \) and \( c^r \):

- Begin at \( D_{\text{acc}}(n,m) \), where \( n \) = \#contour points in \( c^l \) and \( m \) = \#contour points in \( c^r \).
- Search the first matching step (\( D_{\text{acc}}(n-1,j) \) or \( D_{\text{acc}}(i,m-1) \)).
- Create correspondences for the rest of the points until \( D_{\text{acc}}(i,0) \) or \( D_{\text{acc}}(0,j) \) is reached.

The contour parts in the beginning and the end are considered as new contours, and are inserted in the list of unmatched contours where they will be subjects for new possible matchings, (Fig. 6 right). The contour level matching continues until all contours are either grouped or discarded if no corresponding contour was found.

Figure 6: The left image pair shows the matched part of the contours in black, and the unmatched parts in gray. The right image pair shows the unmatched parts from the left, matched with new contours in red.
4.3.2 Plane Level Matching

As mentioned in the beginning, we are interested in finding planar surfaces, and thus in finding contours that are lying in the same plane. The search is performed both within one contour and between the contours. By contour here we refer to an entire contour segment as produced by the DTW algorithm.

Two kinds of contours exist. Either the contour itself makes up at least one plane, as in the case of two consecutive edges of a box, or not. In the first case, different hypotheses are created about what planes that possibly belong to the contour. In the second case other contours must be considered as well in order to be able to create hypotheses.

The contours are searched to generate different hypotheses about planes. We define the set of plane hypotheses according to the following:

$$\Pi_h = \{ \pi_i \}, \pi_i = (\bar{n}_i, \mu_i)$$ (6)

where $\mu_i$ is the mean of the contour points belonging to plane $i$, and $\bar{n}_i$ its normal.

RANSAC [30] is used when searching for a plane hypothesis, since incorrect matching can produce outliers. When a hypothesis is created three contour points, $p_1, p_2, p_3$ are selected. We define our hypotheses as

$$v_1 = p_2 - p_1, \quad v_2 = p_3 - p_1$$ (7)

$$\bar{n}_h = \bar{v}_1 \times \bar{v}_2, \mu_h = mean(p_1, p_2, p_3)$$ (8)

$$\pi_h = (\bar{n}_h, \mu_h)$$ (9)

For a plane to be reliable, the contour(s) must vary with a certain degree in two directions, why the angle $\gamma$ between the vectors $\bar{v}_{p_1,p_2}$ and $\bar{v}_{p_1,p_3}$ must not be too small. We found $40^\circ$ to be a good threshold. When verifying the hypothesis, the distance between the plane and the contour points lying in the vicinity of the selected points is measured. We define $P = \{ p_i \}$ to be the set of points lying within 1 cm. 1 cm roughly corresponds to two times the minimum distance between two contour points at the scene depth our robot is working on. If $P$ contains 70% of the measured points, the plane is accepted.

When a good estimate of the plane is found, we use the supporting contour points to find the plane that minimizes the orthogonal distance from the points to the plane. This can be done using singular value decomposition.

$$M = \begin{bmatrix} \vdots & (p_i - mean(P))^T \\ \vdots \end{bmatrix}$$ (10)

$$M = USV^T$$ (11)

The eigenvector of $V$ corresponding to the minimum eigenvalue in $S$ is the normal vector $\pi_{svd}$. We set

$$\pi_h = (\pi_{svd}, mean(P))$$ (12)

and add $\pi_h$ to $\Pi_h$.

When a plane is found, the rest of the contour points are considered whether they belong to the plane or not. As long as the distance in 3D is within 1 cm, the point is considered to belong to the plane. The search for hypotheses continues until all contours have been either assigned to one or more planes or rejected as not belonging to any plane. Finally each two hypotheses that have near parallel normals and means that are close each others planes are merged. We also make sure that the normals are in the direction from the mean of all contour points.
5 Evaluation

We have tested our method for different objects in order to show that the method works for objects of different shape and complexity. We also show the generated grasp hypotheses on the uppermost surface. We will restrict the grasp type to EGA5 from Section 3. Fig. 7 shows a sequence of images for three of our test objects. We only show the image as seen by the left camera of the robot head. For each of the objects we show the original image, images overlaid with the filtered contour image and the contours that have been matched in the DTW-algorithm. Finally Fig. 8 shows several plane hypotheses and the uppermost surface together with the suggested grasp. In all cases the \((x, y, z)\)-part of the EGA-parameterization will be set to \(\pi_{\text{svd}} + R \times \text{mean}(P)\) (Eq. 12), where \(R\) depends on the embodiment of the gripper. \((\gamma, \beta, \alpha)\) will be adjusted to align the gripper with \(\pi_{\text{svd}}\).

The first object is a textured box of cocoa with a simple geometrical structure. Most of the curves resulting from the texture are filtered already in the contour extraction phase. The second object is a uni-colored cup that blends well with the background. This represents a problem for the edge detection step using Canny which does not find many edges. The third object is a hole puncher with a more complicated geometry. This object is furthermore tilted in order to show how the method benefits from this approach when grasping from any angle is attempted.

Figure 7: From above: The original images; After extraction and filtering of contours; Grouped contours after the DTW-algorithm

In the case of the cocoa box, some structure on the surface of the box remain after contour filtering. Those contour segments are however not matched successfully in the DTW-process. Remaining are contours that belong to the outer edges of the object. The hypothesis that has largest support is naturally the one corresponding to the largest surface of the box. For the grasping reflex we use the blue, uppermost hypothesis. This is slightly different from the actual surface, since contour points on the lower edge of the lid are also assigned to this hypothesis, and the normal is slightly tilted. The error is however small, and a successful grasp would still be possible. The gripper is rotated around the normal of the plane to make the gripper close in
the direction of least variance among the points in the plane.

The contour segments of the cup are all matched, but some are slightly shortened because of the differences in the left and right image. All segments are matched, but note the lower right corner at the three almost parallel lines. Here the DTW-algorithm has successfully matched contours that do not actually correspond to each other, why in Fig. 8 two of the segments are pointing in different direction. In this case though, it does not result in a large error in 3D space. The uppermost plane has by far the largest support, and we can generate a grasping reflex by grasping two opposing contour points towards the center, as demonstrated in Fig. 2. One drawback is that in this case the reconstructed contour is not circular, and thus does not correspond very well to the actual contour. However, this can be solved by adjusting ($\gamma, \beta, \alpha$) of the EGA, so that contour points that have the largest difference between their orientation and the epipolar line are grasped, see Sec. 4.1.1.

Figure 8: Upper) Three plane hypotheses. Corresponding contour points and their normal vectors. Note that in all cases planes share contour points, why points belonging to the most supported plane (red) are covered by other points. Lower) The uppermost plane with a parallel jaw gripper using the normal as an approach vector. The normals of the cocoa box and the cup are roughly parallel to the normal of the table, but for the hole puncher the normal is indicated with blue.

Figure 9: Upper left) The scene with the overlaid segmentation mask. Upper right) Contours extracted and filtered. Lower left) Contours grouped by the DTW-algorithm. Lower right) Three plane hypotheses.
The puncher has many more edge segments, which can be seen in the contour image. Here the DTW-algorithm is only partially successful in finding correspondences, but the ones that are found are correct. The resulting contours constitute the ones that are important for generating a grasping reflex. All three plane hypotheses that are generated here could be used for grasping, but depending on the hand, the green hypothesis might be too large to grasp. In this case, the benefit of not restricting the system to top and side grasps is clear. Although it might be possible to grasp the red hypothesis from the top, this is not the case for the blue or green one. If the robot cannot reach the red hypothesis due to its embodiment or other objects being in the way, we would not be able to pick up the object.

Finally, we show how the system works in a cluttered scene. Fig. 9 shows a scene where the robot has fixated on the middle object, in this case the cocoa box, and segmented it. The edges that are within the segment is used in the algorithm. Naturally these edges are weaker than the ones generated with a background that gives big contrasts. However, the algorithm still finds hypotheses for the three major planes of the box.

6 Conclusion

Grasping unknown objects in natural environments is a challenging problem. For robots that need to explore new objects, we need to envelop grasping methods that are simple, fast, robust and rely on minimal representations of objects. In this report, we have presented a stereo based approach that generates grasping hypotheses based on curve matching between two views. The work builds upon our previous application in hand tracking. The developed methodology is based on the use of dynamic time warping which is here used to enforce the similarity between the curves in corresponding images. This serves at the base for partial reconstruction of objects in 3D which is then used to generate elementary grasping actions or early reactive grasps.

One of the difficult parts of the method that both relates to accuracy and performance, is finding good starting points. The distance measure combined with difference in orientation and curvature that we use is satisfactory but it sometimes generates good starting points on wrong contours. This results in decreased performance when there are many contours to match since then we rely on the DTW-algorithm to reject the matching, and often the DTW-algorithm successfully matches parts of the contour. The idea for the future is therefore to integrate the local appearance in the detection of the starting points.

Acknowledgments

This project has been supported by the EU IST-FP7-IP GRASP (2008-2012) and Swedish Foundation for Strategic Research through project CORS.

References


