Vision-based grasp planning of 3D objects by extending 2D contour based algorithms

Johannes Speth¹, Antonio Morales², and Pedro J. Sanz²

Abstract— This paper addresses the problem of grasp planning of unmodelled 3D objects with the aide of vision. The approach proposes the extension of fast 2D contour-based grasp planning algorithms, and the use of images of the target object from different points of view to recover critical 3D information like the size, location and the pose. A complete method is described, and experimental results are presented showing its feasibility.

I. INTRODUCTION

The problem of robotic grasping is a difficult one, since it needs external information very difficult to obtain with current sensors, as, for example, the object's model, the friction coefficient or the object's mass. For that reason, a simplified version of the problem was first addressed, namely 2D grasping. This technique concerns the stable grasping of an object, but taking only a top view of it. It is the approach adopted by almost all works that make use of vision for grasping, because, in this case, the vision processing simply consists in extracting the object's contour and possible grasp regions. Nice results have been obtained in this line [1,2,3], but, unfortunately, they can only cope with planar objects, seen from a restricted set of views.

Most works on 3D grasping have assumed to have a model of the objects. Then the problem consists on finding a set of contacts that meet force-closure condition grasp stability with a reasonable computational effort. In [4] the object surface is sampled into a set of points that are used for computing 4-fingered force-closure grasps in 3D. In [5], the authors implement a randomized grasp generation algorithm for quickly finding force-closure grasps in 3D, taking as input the object's 3D model, while in [6] a more structured approach is proposed using a description of the object based on basic shape primitives. The main drawback of these approaches is that the object's 3D model must be known. This is a strong assumption in service tasks scenarios, where robots must perform manipulation of very different unknown objects.

To overcome this problem, sensor based approaches retrieving 3D information are currently used for grasp planning. Some of them use laser range sensor for obtaining the 3D structure of a given object with grasping purposes [7]. Vision has been also widely used to obtain object shapes. In [8] the authors use stereo vision to obtain a cloud of 3D points that are later grouped into polygonal meshes, and compared with stored object models whose grasp points are already known.

But perhaps the most active line is shape from silhouette, because of its robustness. Any physical object under any condition has a boundary which generates a silhouette in the image that provides much information about the object's geometry [9]. Several authors on vision communities have opted for the volume intersection approach [10], which consists in back-projecting the silhouette for several viewpoints and intersecting all the volumes (visual hulls). Some [11] have published some efficient algorithms, while others [12] make use of differential geometry for simultaneously estimating object's geometry and camera motion.

In the grasping field, the general approach is to use 3D reconstruction algorithms for computing object's geometry, and grasp planning algorithms for computing contacts [7]. But the question of what kind of 3D information does grasp algorithms need in order to successfully compute a stable grasp has been underestimated. In our opinion, a complete 3D model is not necessary neither suitable for doing 3D grasping, and recent works from neuroscience support this thesis [13,14]. In order to have service robots working on unstructured and dynamic scenarios, we need simple and fast visual processing, as the 2D case, and effective grasping capabilities, as the 3D case.

We propose in this paper a minimalist approach that consists in retrieving partial 3D structure from simple 2D silhouette image processing for enabling a successful grasping algorithm. To the best of our knowledge, only one work [15] followed a roughly similar approach to ours. In [15], the authors proposed an active vision approach for visually-guided grasping of unknown, smooth objects with a parallel jaw gripper. Instead of reconstructing the object, they searched for an antipodal grasp in 3D. Thus, by means of simple 2D image processing, they were able to guide the gripper in 3D towards the object. However, they only tested the algorithm with a very specific scenario (i.e. a pile of potatoes).

II. SYSTEM FUNCTIONAL DESCRIPTION

The practical goal of the system is to make a robot manipulator able to grasp an unknown object lying on flat surface. No assumption is made about the shape of object.

¹J Speth is with Technische Universität München, München, Germany (jospeth@gmail.com).
²A. Morales and P. J. Sanz are with the Robotics Intelligence Lab., Universitat Jaume I, E-12071, Castellón, Spain {morales,sanzp}@uji.es.
The robotic system consists of a 7 d.o.f. Mitsubishi PA10 robot arm, endowed with a camera-in-hand and a three-fingered Barrett Hand. The camera is a compact model which small size allowing to be assembled between the spreadable fingers of the Barrett Hand.

Figure 1 describes the complete functional flowchart of the system. The goal is firstly analyze the object by taking images of the object from predefined directions. The analysis of these images will provide the necessary information to carry out the next stages: namely, grasp synthesis and analysis (1), hand approaching (2), and grasp execution (3) that finally lifts up the object. These latter three modules are described elsewhere [19]. In this paper we focus on the image processing and object model building module.

III. IMAGES ACQUISITION AND ANALYSIS

The basic idea consists of taking images of the target object from different positions; analyzing these images using 2D contour-based algorithms; and providing the necessary 3D information to the grasp planning and execution modules.

Figure 2 depicts the location with respect the object of the camera from where the images are taken. The object is represented by the blue box. The camera, attached to the arm’s wrist, starts facing down in some place over the object. It is placed over the center of the object, without changing the orientation, where image top1 is obtained. Then it is moved to locations front1, and front2. An image is taken in each of these locations.

A. 3D object model acquisition

We consider 3D model of an object that is not properly a model of the shape of the object, but a compilation of 2D contour based features obtained from different views and some critical 3D information. To understand this model is firstly necessary to describe how the images of the object are obtained and processed.

The complete procedure is illustrated in figure 3. The different boxes in this flowchart also describe the image processing algorithm applied:

Step 1: Initialization. A visual servoing loop is used to follow the centroid of the object surface and placing it in the center of the image. Thus, a position directly above the object is reached so that only the contour as projected straight down on the workspace will be seen. This approach avoid the parallax problem (see Fig. 4), where the contours of the object as seen from these two points have hardly anything in common. Image top1 is obtained at the end of this step.

Step 2: Object contour analysis. In parallel with previous step each time an image is recorded, the object contour
Step 4: have a full understanding of the method.

are still some important details that must be mentioned to

computed from it. These features are used in the further
course of the algorithm. Only computation of the moments
up to order two is needed, and from them, the centroid and
orientation of the main axis. Low level details about how
those features can be calculated can be found in [16]. The
relevant features used in the current work are the following:

- **Contour and centroid.** Contour is necessary for the
grasp planning phase, and the centroid (center of the
object blob) is the most relevant reference point.

- **The axes of inertia** $(I_{min}, I_{max})$. Since humans also
tend to analyze and grasp objects from the directions
of its main axes, as described in [13], it seems a
promising way to be developed in the robotics side.
In our case, the axes of inertia of the top contour
point in the directions from where the object should
be analyzed next.

- **The bounding box** is used as an approximation for
the dimensions of the object. Since we do not want a
complete reconstruction of the object, a very
simplified model of the object had been assumed. In
this case a 3D bounding box around the object.

- **The curvature** of the contour at every point is
calculated and will be used in the grasp planning
phase, explained later in section IV.

- **The circularity** $(C)$ is used to get more predictable
directions for the axes of inertia for circular contours
(e.g. projections of spherical and cylindrical objects).
It is expressed as,

$$ C = \frac{A}{\text{maxdist}^2 \pi} $$

Whereas $A$ is the area of the region and $\text{maxdist}$ is the
distance of the farthest contour pixel to the centroid.

C. The parallax problem

The main goal of initialization module (step 1), is to
overcome the parallax problem. As can be appreciated in
figure 4, the contours of the object as seen from these two
points have hardly anything in common. Our way to prevent
this problem has been to move straight over the object, so
that the sides of the object do not falsify the contour. This
means the optical axis of the camera shall be inside the cone
projected by the contour of the object. In other words the
center of the image shall lie inside the projected contour of
the object. Thus, a 2D visual servoing loop with the contour
centroid as visual reference has been implemented to move
the camera straight over the object.

D. Determination of feature reference points

Feature points are needed to be able to perform a partial
3D reconstruction in order to know the distance from the
object and also the real sizes of the object. These feature
points must be invariant to the scaling of the contour while
moving along the optical axis, so that corresponding points
can easily be found for the reconstruction. For the top view
the intersections of the axis of inertia, $I_{min}$ and $I_{max}$, with the
bounding box can be used to calculate the real length and
width of the bounding box by reconstruction. Also the
distance of these reconstructed points to the camera can be
used to calculate an approximation of the real height of the
object if the level of the workspace is known.

is obtained with a simple binarization algorithm. The
extracted contour is then analyzed to compute some
geometric features: axes of inertia; bounding box along
these axes; length and width of bounding box in pixels;
intersection points of axes of inertia with bounding box;
curvature of contour, and circularity. This information
together with the pose of the camera is saved as part of
the internal representation of the object.

**Step 3:** **Move along optical axis.** After top1 positions the
camera moves along its optical axis. The intersection
points of the axes of inertia with the 2D bounding box
are reconstructed. From this the length, width and
height of the 3D bounding box as seen from above are
calculated.

**Step 4:** **Front pose calculation.** After saving information
of image top2, the front pose is calculated from
direction of minimum of inertia axis, $I_{min}$, the size and
the centroid of the object. Then step 3 is repeated for
front1 and front2 positions. For these values, the center
of mass of the bounding box is calculated. The
combination of these pairs of views allows computing
the 3D location of the bounding box.

**Step 5:** **Object model compilation.** All the information
contributes to build a partial model of the object. Each
taken image provides a contour and a bounding box.
Two images from a view point allow computing the
location of the bounding box. In general every new
view point will provide a lot of extra information about
the object. In this case only the width and height of the
object can be refined. Actually, from the taken images,
a reasonable estimation of the center of mass of the
object is computed.

These are the main steps of the algorithm; however, there
are still some important details that must be mentioned to
have a full understanding of the method.

B. **Contour based 2D features.**

Each time and image of an object is saved its contour is
obtained (step 2) and some moment-based features are
computed from it. These features are used in the further
course of the algorithm. Only computation of the moments
up to order two is needed, and from them, the centroid and

Fig. 4. A bottle is shown to exemplify the parallax problem.
In case of circular contours, which circularity is over the threshold 0.85, the orientation of \( I_{\text{min}} \) is just set to a predefined value (e.g. 90 degrees), which results in unvarying axes of inertia between different images also for a circular object.

E. Object height estimation.

Once the object has been centered in the image the camera is move forward along its optical axis. This avoids again the parallax problem, and thereby the contour resulting in the image will only change in size, but not in shape. Thus it is much more robust to reconstruct corresponding points for non-symmetric objects. Detailed information about 3D reconstruction can be found elsewhere [17].

In our case, the ray for an image point \( p \) for a camera with the pose \( C \) and the calibration matrix \( C_{\text{cal}} \) can be described in base coordinates by equation.

\[
\overrightarrow{\text{ray}}_j = \hat{t} + mCC_{\text{cal}}^{-1}p
\]  

with \( C = (R \ t) \) being the \( 3 \times 4 \) matrix containing the relative pose (rotation \( R \) and translation \( t \)) of the center of projection with respect to the reference coordinate system. The factor \( m \) can be varied to scale the length of the ray. With this equation the rays describing the same point on two different images can be described. Their intersection will be the locations of the real 3D point (see figure 5). To avoid the skew problem derived form the inaccuracies the point with the shortest distance to both lines of sight is computed.

If the location of the surface where the object is lying is known, this process results in the computation of an estimation of the height of the object. This estimation can be accurate if the object contour from the top images is the contour of the top side of the object. This might not be the case often. This estimation can be refined if additional views of the object are taken, as figure 6 describes.

F. Computation of the front view pose

In addition to the top view, a second view from a side of the object would provide helpful extra information. It could be used in a refinement of the height of the object (fig 6.) providing additional information for calculating the real height of an object. The side contour is also used to compute lateral grasps.

In order to compute the pose of the front view the direction of \( I_{\text{min}} \) in the top view is used to move along. This is done so because of the very high probability of being also the axis of mirror symmetry of the object. Initially a side position with a horizontal optical axis as in figure 6 was tried. Unfortunately, many inconveniences were encountered which made the movement of the hand/camera to a perfectly horizontal front position, impossible for most cases. The main reasons were: a) for small objects the hand would hit into the work ground in order to get the camera in the right position; b) tall objects (large height) could not fit into the field of vision; and c) the joints of the arm could not be able to perform this extreme movement.

To overcome this problem an alternative side position for the front view of the object was adopted (Fig. 2). In the pose the camera is at same height of the initial top pose. And only needs to move horizontally. Now, the challenge was to compute the displacement from the top view position and the pitch angle \( \theta \) so the object remains within the field of view. This angle is expressed,

\[
\theta = \tan \left( \frac{dx}{dz} \right)
\]  

with \( dx \) being the distance of the camera along the direction of \( I_{\text{min}} \) and \( dz \) being the difference between the \( z \)-components of the top-front camera pose and the calculated center of mass of the bounding box (fig.2). As shown before this center of mass can lie too low if the height of the object cannot be seen correctly from above. In this case the object might be out of the center of the image.

In the case of the front view the feature reference points are chosen to be the most extreme points of the front contour with respect to their \( x \)- and \( y \)-components.

IV. GRASP PLANNING AND EXECUTION

The main focus of the current paper is the vision based process for partial 3D reconstruction. However, as described in section II and figure 1, this is only a part of the whole grasping system.
A. Grasp planning

In few words the grasp planning algorithm takes as input the contours of the target object and produce a hand configuration to grasp this object reliably. It consists of the next steps:

1. **Extraction of grasp regions from the contour.** The contour of the object is analyzed to determine low curvature segments (grasp regions) that ensure a stable contact with the fingertips.

2. **Filtering of compatible regions.** Pairs of triplets of grasp regions are compared in order to find groups that allow a two or three-finger grasps.

3. **Refinement to hand configuration.** The pairs and triplets of compatible regions are instantiated in particular two and tree-finger hand configurations that contact on each of the grasp regions. The particular cinematic constraints of the Barrett hand are taken into account during this process. These three first steps are described with full details in [18].

4. **Selection of the grasp.** The three previous steps provide as a result a list of feasible hand configurations. To decide among them a set of quality metrics are calculated and combined [19]. The best ranked hand configuration is selected to be executed.

B. Approaching and grasp execution

The 3D pose of the object has been established after the visual analysis. The approaching phase consists of moving the robot hand to the starting position of the grasp. Since the bounding box of the object has been established, the hand configuration has been computed with respect the contour and its bounding box, and because the geometry of the hand is known the determination of the hand pose to starting the grasp is trivial. It is noticeable the objects can be approached from top or from frontal direction depending on the source origin of the contour that is used for the computations. In any case, the algorithm can run with no regard of the point of view.

Once the starting grasping pose has been established the hand is moved to that location and the grasp started. Two control laws have been implemented to improve the grasping performance of the hand: (a) Simultaneous closing of the fingers and (b) compensation of z-movement for precision grasp. The control (a) is necessary because the fingers might exert unwanted forces on the object if the closing extent of the fingers differs much. In velocity control this can happen easily, since a desired velocity results in slightly differing velocities for different fingers. In order to overcome this, a control law is used to make sure the fingers close simultaneously. For finger $F1$ this means:

$$v_{F1adv} = v_{F1des} + \kappa \left( \frac{\theta_{M2} + \theta_{M3} - \theta_{M1}}{2} \right)$$

where $v_{F1adv}$ is the velocity value of finger 1 as it is given to the control software. $v_{F1des}$ is the desired velocity of finger 1, which is regulated by the difference between its motor position ($\theta_{M1}$) and the average of the other fingers $\theta_{M2}$ and $\theta_{M3}$. This difference is multiplied by a factor $\kappa$ to change the effect of the regulation. This regulation is performed for each finger. If the object is centered in the palm of the hand this makes sure the fingers intersect with the object at the same time and unwanted forces and torques are minimized.

The control (b), takes into account that the algorithm also makes a difference between grasps from top and grasps from front. Top grasps are performed as precision grasps whereas front grasps are performed as power grasps, because this approach makes easier the gripper adaptation for objects processed as bounding boxes like in this case. The difference is that power grasps try to enclose the object in the hand and also use the palm to support the grasp, whereas precision grasps only make contact with the finger tips. For
power grasps the palm must be moved as close as possible to object and the fingers are just closed, thus reaching as far around the object as possible. In the case of precision grasps the hand kinematics are used to make sure that the finger tips stay at a fixed height while closing the hand. This is done by moving the arm in velocity control to compensate the movement of the fingers in z-direction of the tool coordinate system, enabling the fingers always close to the top of the object and so over the center of mass, guaranteeing stability.

V. IMPLEMENTATION AND RESULTS

In order to test the algorithm many experiments were made with a variety of different objects. Fig. 7 shows a sample of relevant examples. In the two first cases the objects are grasped from the top approaching direction. In both cases pure three finger precision grasp are used.

In the other three cases the objects are grasped from frontal directions, since the images used are from the frontal view. In these cases two finger grasps, in which two physical fingers are used together conforming a virtual finger [18], are executed.

It is noticeable that in the current implementation, the decision between grasping from the top or from the front pose is taken by the user, but there is enough information to be executed for a task planner in automatic way if necessary.

VI. DISCUSSION AND CONCLUSIONS

In this paper a new vision based approach for grasping previously unknown 3D objects has been implemented and tested on a real robotic system. The goal was to look at the object from only two singular directions, analyze the contour of the object and combine the information in order to create a partially 3D model of the object. Then this information is used to calculate possible grasps and guiding the hand to the right position before closing the fingers during grasping. The main steps of the followed strategy have been detailed in this paper, obtaining as output all feasible grasps according to the applied hand model (i.e. Barrett Hand) and stability criteria. All the information necessary has been obtained from the moment based geometrical features described. These features allow locating the object, and determining its rough dimensions. Then simple and faster 2D contour analysis is applied for the grasp planning. The experimental results demonstrate the reliability and feasibility of this system.

However some improvements would become more robust the system, like by using other sensorial capabilities on the execution phase that would compensate the inaccuracies of both, grasp planning and visual perception.

ACKNOWLEDGMENTS

This paper describes research carried out at the Robotic Intelligence Laboratory of Universitat Jaume-I. The work has been funded by Generalitat Valenciana, under project GV-2007-109 and Fundació Bancaixa, under project P1-1A2006-11.

REFERENCES