Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Robotic Systems Lab
Prof. Dr. Marco Hutter

Semester Thesis

Contour Based Grasp Synthesis of Unknown Objects with a Humanoid Hand

Sammy Christen Martin Wermelinger
Dario Bellicoso Marco Hutter

Autumn Term 2017

Supervised by:
Martin Wermelinger
Dario Bellicoso

Author:
Sammy Christen

Abstract

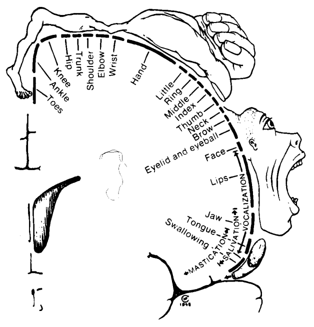
In robotics, grasping objects with unknown properties is becoming increasingly important, as we move from labratory setups to real world environments. In this work, we present an approach to grasp unknown objects with multi-fingered grippers. We model object contours with Elliptic Fourier Descriptors and calculate an optimal gripper pose in order to grasp objects by considering the curvature value of the contour at the grasping points and the ratio between initial position of fingertips and grasping points. Our goal is to obtain grasping points at large surfaces and concave areas of the object and have the fingers reach the grasping points simultaneously. The approach works with RGB-D images from single view and a 2D approximation of the gripper. The framework we implement is versatile and can be adapted to different gripper configurations. Experiments with an underactuated, anthropomorphic hand that can execute a power grasp are carried out and show that the pipeline is able to find grasps that work on simple objects, i.e., objects with contours that have few edges and favorable grasping properties, but faces issues with more complex objects, i.e., objects with contours that have several edges and unfavorable grasping properties.

Contents

Chapter 1

Introduction

Grasping seems to be an effortless task for humans. Once we have selected an object we want to grasp, our brain unconsciously estimates the optimal grasp type, the angle of approach and when we have to execute the grasp. Though it may seem simple to humans, the brain is highly active during this process. Interestingly, the areas dedicated to motor processing in the cortex of the brain for the hand and fingers are much larger than the dedicated areas for other body parts, as illustrated in Figure 1.1. This hints at the complexity that comes with a seemingly trivial task such as grasping.



(a) Homunculus from Penfield and Rasmussen [1]

(b) Model figure of motor homunculus [2]

Figure 1.1: Homunculus that illustrates the relative size of different body parts according to their dedicated areas in the brain.

Accordingly, in robotics the problem of grasping is very difficult. It is considered one of the most complex problems to solve and has become a hot topic of research. In the industry, most tasks are very specific, e.g., pick and place tasks for objects on a conveyor belt. End-effectors such as suction cups or parallel-jaw grippers are often used to accomplish such tasks. Even though such devices are very simple and efficient for a specific task, they are not very versatile and may not generalize to other tasks. End-effectors that allow more flexibility and adaptability are needed. Through improvement in actuator technology and miniaturization, such devices have emerged in the form of anthropomorphic, i.e., human-like, end-effectors. Another distinction is made between underactuated and fully actuated devices. Underactuated devices have less actuators than Degrees of Freedom (DoF)[3]. Fully actuated devices, on the other hand, have an actuator for every joint and hence every joint can be controlled individually. By making use of the natural dynamics of the system, underactuated devices are more power efficient and make movements look smoother compared to fully actuated devices. In this project, we want to explore the possibilites of synthesizing grasps with an underactuated, anthropomorphic hand.

The goal of grasp synthesis is to find an optimal pose, i.e., a position and orientation of the gripper in space, such that a stable grasp on a given object can be executed successfully. As shown by Borst et. al [4], it is often not required to find an optimal grasp in order to meet certain criteria. A grasp with average quality is usually already sufficient. Hence, there may be many possible poses that lead to a successful grasp. Generally, a grasping strategy should meet criteria such as stability, task compatibility and adaptability to novel objects [5]. A grasp is considered stable if a small disturbance on the finger force or object position brings the system back to its original configuration through a restoring wrench [6]. Existing methods to tackle grasp synthesis can be categorized in analytical and empirical approaches. Analytical approaches consider geometric, kinematic and dynamic properties of the object and gripper to compute a certain quality measure and optimize the pose for it. The main issue with analytical approaches is computational complexity. Empirical approaches, on the other hand, try to observe and mimic human strategies for grasping with a set of heuristics [3]. In such approaches, the strategy is to either learn grasps by observation of a human performing a grasp or find a grasp by object observation. The latter strategy tries to learn how to associate object properties with predefined grasps. Computation time is not an issue with empirical approaches, but problems arise with selecting an optimal grasp for new objects. A deeper survey into these approaches is provided by Sahbani et. al [7].

The grasp synthesis problem proves to be difficult in real world applications, as different factors such as noisy sensor data, missing shape information or position inaccuracies of the end-effector can generate uncertainties. The goal of this project is to find a strategy to cope with these uncertainties and thus create a robust grasp synthesis framework. A pipeline that takes images of unknown objects from a RGB -D camera and finds an optimal end-effector pose as output should be implemented. The term “unknown object” is used in the sense that there is no prior information about the object. Additionally, the pipeline should be easily extendable and generic, such that it works with different hand configurations. In the end, the pipeline will be tested on a real robot, using an underactuated, anthropomorphic hand that is limited to an one DoF power grasp (see Section 4.1.2). The test scenario is a simplified workspace with only one object on a planar surface, which is also the most studied scenario in literature [8]. In such a scenario, object segmentation becomes easier, since there are no occlusions with other objects to be dealt with. The goal for the robot is to reach the optimal pose, execute a grasp and pick up the object.

The remaining chapters of this report are structured as follows: In Chapter 2, related literature is discussed. The methods used in the different stages of the grasp synthesis pipeline are described in Chapter 3. In Chapter 4, the conducted experiments are explained and the results are illustrated. In Chapter 5, we analyze and discuss the results. Finally, a conclusion and an outlook are given in Chapter 6.

Chapter 2

Related Work

Several different strategies for grasping unknown objects have been explored in literature. Generally, RGB-D cameras are most commonly used as sensor input for grasp synthesis, while some approaches additionally use contact-reactive feedback from tactile sensors to improve grasps.
Richtsfeld and Vincze [9] consider grasps only from the top surface of objects. Grasp synthesis is done using depth images. The objects are first segmented and then grasping contact points close to the center of mass of the object are selected. This approach works for different grippers, but requires an accurate 3D model of the selected gripper. Hsiao et. al [10] also use depth images for the synthesis of grasping with a parallel jaw gripper. An optimal grasp is selected from a set of possible grasps based on an estimation of the object's overall shape and local features. To execute a more robust grasp and react to disturbances, tactile sensors are attached to the fingertips. They sense contacts that indicate whether a grasp will push the object away and thus adapt the grasp to increase the likelyhood of being successful. Schiebener et. al [11] also make use of tactile sensors to prevent undesired contacts during grasp execution. For their approach, they calculate an estimate of the object's principle axis and align the gripper, a pneumatically actuated anthropomorphic hand, with it. For object segmentation, they use camera images to recognize potential objects and then push them with their end-effector to verify that the segemented parts really constitute of objects. A limitation is that they require the object surfaces to be textured, i.e., they cannot segment unicolored objects or objects with weak texture surfaces. Cordella et. al [12] suggest a method for optimal grasping of cylindrical objects, e.g., rails and handles, with anthropomorphic grippers. The algorithm minimizes the distances of the hand joints from the object surface. The drawback of this approach is that it is limited to cylindrical objects. Krug [3] calculates Independent Contact Regions on the object's surface. They help qualify different grasps, e.g., they indicate the robustness of a grasp to position inaccuracies. In this work, they use observations from human grasping to impose constraints on the corresponding optimization problem and thus reduce the grasp solution space. The approach works for different types of grippers, but requires an object database, which may not be feasible for online operation in unknown environments. The method proposed by Gallardo and Kyrki [13] use a single stereo image pair to generate a partial three-dimensional point cloud describing the objects. They assign shape primitives, e.g. cylinders or boxes, to the objects, assuming that this information is sufficient for grasping. Then, grasps that have been previously assigned to different shape primitives are executed using a parallel jaw gripper.

A fundamental basis of our work is provided by Calli et. al [14]. They present a novel approach to grasp unknown objects using active vision. They use Elliptic Fourier Descriptors (EFD) to model object contours in 2D and define grasping points as concave local extrema of curvature. In order to find the optimal grasp, grasping point sets are formed and evaluated using the sum of curvature of the grasping points and a force closure test to guarantee stability. In a last step, visual servoing helps moving the gripper to a position where the curvature of the optimal grasping points are maximized, i.e., an optimal viewpoint. The visual servoing works under the assumption that the grasping points are still reachable for the gripper after the change of viewpoint. The suggested method is well suited for grippers with two or three fingers, but obtaining and reaching a set of optimal grasping points as local extrema of curvature with five fingers becomes much more difficult. Also, our conditions are different due to kinematic restrictions of the gripper, i.e., the gripper only has one actuated DoF and is thus limited to a power grasp. This leads to the following consequences for our approach:

1. Synthesizing for local extrema of curvature as grasping points does not work, because the anthropomorphic end-effector will likely not be able to reach all of those points. Instead of looking for local extrema, we optimize for poses that lead to grasping points with high curvature value.

2. Visual servoing is not possible, since it can not be guaranteed that the grasping points will still be reachable after a change of viewpoint. Therefore, we limit our approach to single view and do not use viewpoint optimization.

There are several advantages of this approach over the above mentioned methods: It does not require a 3D model of the object or gripper. Furthermore, there are no restrictions on object properties such as shape or surface texture. Additionally, it does not require any prerequisites, such as predefined grasps or shape primitives. Lastly, it uses the assumption that concavities and flat, large surfaces are optimal grasping regions, which seems to be where humans would intuitively grasp an object. Therefore, we use the general idea of finding grasping points at concave curvature of the modeled object contour from single view and try to adapt this to our conditions.

Chapter 3

Methodology

In this chapter, the different steps of the grasp synthesis pipeline are explained in detail. We impose the restrictions that grasp synthesis from single view is sufficient and that the image plane is always kept parallel to the ground. The whole pipeline is illustrated in Figure 3.1. In a first step, RGB and depth images are combined to calculate an object contour as explained in Section 3.1. We then describe this contour with Elliptic Fourier Descriptors as shown in Section 3.2. In Section 3.3, we show our finger model and explain the computation of the optimal end-effector pose to reach the desired grasping points on the contour in Section 3.4.

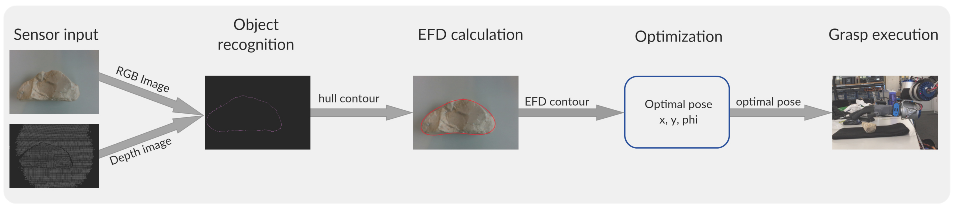


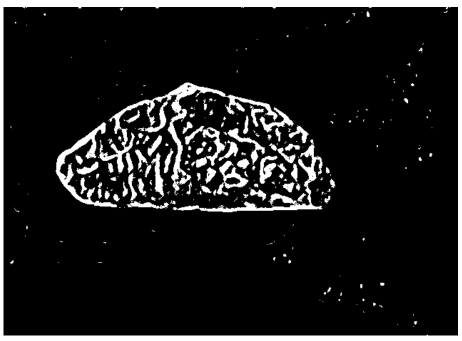
Figure 3.1: Grasp synthesis pipeline from image output to grasp execution

3.1 Object Contour Recognition

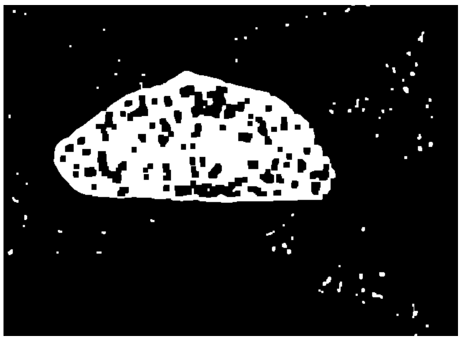
The goal of the object contour recognition is to get a 2D estimation of the object shape. We therefore extract a contour from RGB images and combine it with information from the depth camera to get an approximation of the object contour in meters. The extraction of the contour from the RGB image consists of the following steps explained below:

1. Conversion to grayscale
2. Median blurring
3. Adaptive thresholding
4. Closing operation (dilation and erosion)
5. Finding all closed contours in image
6. Finding largest object contour

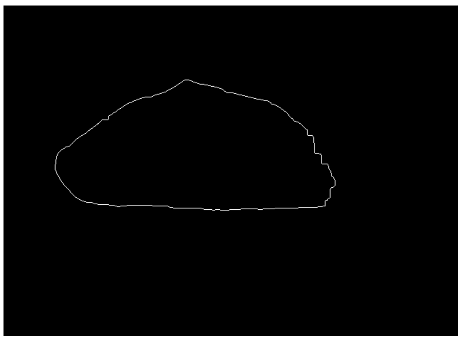
(a) Example object



(b) Binary image after thresholding



(c) Binary image after closing operation



(d) Contour extracted from RGB image

Figure 3.2: Different steps of the object recognition process

In a first step, the RGB-image is converted to grayscale for further processing. Since we only use the RGB-image for edge detection, a grayscale image is assumed to be sufficient. Converting to grayscale also reduces computational complexity. In a next step, median blurring is applied to smoothen the grayscale image. Thus, some edges, such as shadows or edges within the object, will get weaker or even blur out. Adaptive thresholding converts the smoothened image into a binary image. It is an extended form of thresholding, which decides for each pixel whether it is below or above a certain threshold, also taking into account neighboring pixels. Our goal is to find an outer contour of an object without any holes or gaps. This can be achieved with the closing operation, which is a two step process to strengthen edge connections. The first step consists of a convolution of the binary image with a structuring element that leads to expansion of shapes and closing of small holes, which is called dilation. The second step, erosion, uses a structuring element that expands the background and makes shapes smaller. Thus, the object contour is first strengthened and expanded through dilation and then reduced through erosion. Finally, we extract all closed contours in the image. A contour is defined by adjacent pixels that have the same intensity value. For further analysis, the largest contour found in the optical frame is assumed to be the object contour. The visualization of different steps of the contour extraction are shown in Figure 3.2. For the example object shown in 3.2(a), the described steps will lead to a contour as seen in Figure 3.2(d). The next step is the conversion of the contour from pixel values to meters in the optical frame. This is achieved using the intrinsics of the camera and the information of the depth image at the object location, leading to the following formulas:

$$x_{[m]} = \frac{1}{f_x}(q_x - pp_x)z_{\max} \tag{3.1}$$

$$y_{[m]} = \frac{1}{f_y}(q_y - pp_y)z_{\max} \tag{3.2}$$

The point q is the pixel point that has to be converted. The information retrieved from the depth image at point q is the depth $z_{\max}$. The values f_x and f_y indicate the focal length of the image plane as a multiple of pixel width/height. The values pp_x and pp_y are the coordinates of the principal point of the image as pixel offsets from the left/top edge. Due to noisy or missing sensor data from the depth sensor at some locations on the contour, the maximum depth found on the contour is used for the conversion of all points.

3.1.1 Depth Value

For the grasp execution (see Section 4.1.4), we need to find a value z that indicates how far the end-effector should move in the direction perpendicular to the camera's image plane. In the current implementation, we assign the minimum depth value that is found within or on the contour to z and add a small offset, since we do not want the end-effector to collide with the object.

3.2 Elliptic Fourier Descriptors

To model the object contours, we use Elliptic Fourier Descriptors (EFD) [15]. EFD can be used to mathematically describe a closed contour in 2D with a Fourier series. They are defined by the following parametrization:

$$x_{\text{contour}}(t) = A_0 + \sum_{n=1}^{k} a_n \cos(\frac{2n\pi t}{T}) + b_n \sin(\frac{2n\pi t}{T}) \tag{3.3}$$

$$y_{\text{contour}}(t) = C_0 + \sum_{n=1}^{k} c_n \cos(\frac{2n\pi t}{T}) + d_n \sin(\frac{2n\pi t}{T}) \tag{3.4}$$

where k indicates the number of harmonics, A_0 and C_0 are the DC components, i.e., the geometric locus of the contour, and a_n, b_n, c_n and d_n the Fourier coefficients. For any given t, a point on the contour is defined by x_{contour} and y_{contour}. Using low harmonics, we obtain concave curvature at flat surfaces and concave areas of the object. These are the areas that we assume to be best for grasping. Using higher harmonics, the model precision increases, but it will also lead to higher computational cost. Furthermore, flat surfaces may not have a high curvature anymore, which is undesired in our case, since we assume that both flat surfaces and concave areas are optimal grasping regions [14]. The curvature value can be extracted from the second derivative of the EFD parametrization:

$$C_{\text{curvature}}(t) = |N(t)| = \left| \begin{bmatrix} N_x \\ N_y \end{bmatrix} \right| \tag{3.5}$$

where

$$N_x(t) = \ddot{x}_{\text{contour}}(t) = \sum_{n=1}^{k} -a_n(\frac{2n\pi}{T})^2 \cos(\frac{2n\pi t}{T}) - b_n(\frac{2n\pi}{T})^2 \sin(\frac{2n\pi t}{T}) \tag{3.6}$$

$$N_y(t) = \ddot{y}_{\text{contour}}(t) = \sum_{n=1}^{k} -c_n(\frac{2n\pi}{T})^2 \cos(\frac{2n\pi t}{T}) - d_n(\frac{2n\pi}{T})^2 \sin(\frac{2n\pi t}{T}) \tag{3.7}$$

Curvature is an indication of how convex or concave a certain point is. It should be noted that a high curvature value can imply both a convex or concave extrema. Therefore, a further check is necessary in the optimization (see Section 3.4) to find out whether a point is convex or concave. We do this by making use of the cross product of two adjacent points. Assuming the contour is read in a clockwise direction, a point is concave if:

$$N(t) \times N(t + \varDelta t) > 0 \tag{3.8}$$

In case the contour is read in a counter-clockwise direction, the sign in Equation 3.8 changes.

3.2.1 Calculation of EFD Parameters

The Fourier coefficients for a closed contour array can be calculated using the following formulas:

$$a_n = \frac{T}{2n^2\pi^2} \sum_{i=1}^{K} \frac{\varDelta x_i}{\varDelta t_i} \left[\cos(\frac{2n\pi t_i}{T}) - \cos(\frac{2n\pi t_{i-1}}{T}) \right] \tag{3.9}$$

$$b_n = \frac{T}{2n^2\pi^2} \sum_{i=1}^{K} \frac{\varDelta x_i}{\varDelta t_i} \left[\sin(\frac{2n\pi t_i}{T}) - \sin(\frac{2n\pi t_{i-1}}{T}) \right] \tag{3.10}$$

$$c_n = \frac{T}{2n^2\pi^2} \sum_{i=1}^{K} \frac{\varDelta y_i}{\varDelta t_i} \left[\cos(\frac{2n\pi t_i}{T}) - \cos(\frac{2n\pi t_{i-1}}{T}) \right] \tag{3.11}$$

$$d_n = \frac{T}{2n^2\pi^2} \sum_{i=1}^{K} \frac{\varDelta y_i}{\varDelta t_i} \left[\sin(\frac{2n\pi t_i}{T}) - \sin(\frac{2n\pi t_{i-1}}{T}) \right] \tag{3.12}$$

$$t_i = \sum_{j=1}^{i} \varDelta t_j \tag{3.13}$$

where $\varDelta t_i$ are the euclidean distances between two adjacent points in the contour array and t_i is the summed up distance of all edges from the inital point in the array to i. The value K is the total number of points in the contour array, the values $\varDelta x_i$ and $\varDelta y_i$ indicate the difference in distance between two adjacent points on the contour in x and y direction. Furthermore, the DC components are calculated as follows:

$$A_0 = \frac{1}{T} \sum_{i=1}^{K} \left[\frac{\varDelta x_i}{2\varDelta t_i}(t_i^2 - t_{i-1}^2) + \mu_i(t_i - t_{i-1}) \right] \tag{3.14}$$

$$C_0 = \frac{1}{T} \sum_{i=1}^{K} \left[\frac{\varDelta y_i}{2\varDelta t_i}(t_i^2 - t_{i-1}^2) + \delta_i(t_i - t_{i-1}) \right] \tag{3.15}$$

where

$$\mu_i = \sum_{j=1}^{i-1} \varDelta x_j - \frac{\varDelta x_i}{\varDelta t_i} \sum_{j=1}^{i-1} \varDelta t_i \text{ and } \mu_1 = 0 \tag{3.16}$$

$$\delta_i = \sum_{j=1}^{i-1} \varDelta y_j - \frac{\varDelta y_i}{\varDelta t_i} \sum_{j=1}^{i-1} \varDelta t_i \text{ and } \delta_1 = 0 \tag{3.17}$$

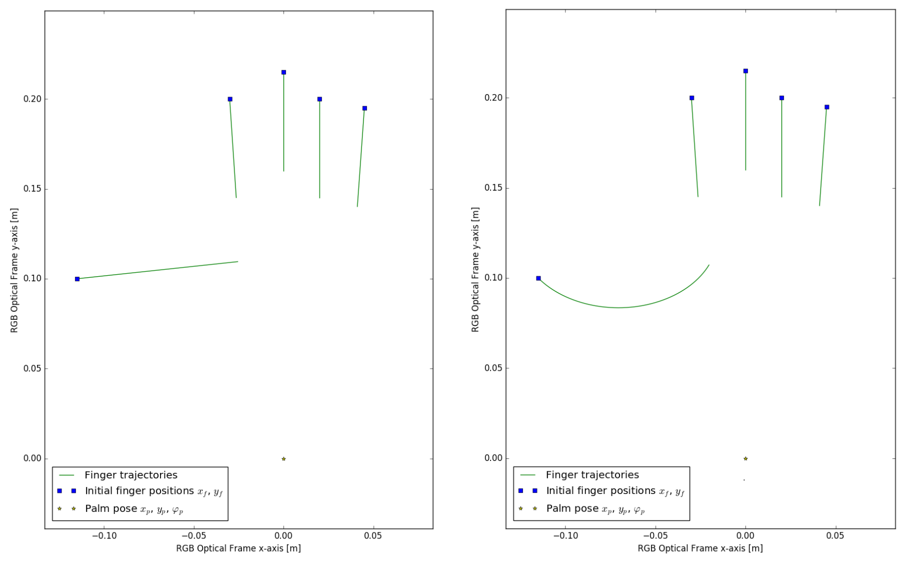
3.3 Finger Model

In this section, we describe how we approximate the finger closing motion in 2D. Our basic approximation of a finger trajectory in the image plane is a ray. We assume this approximation is accurate enough to get a good estimation of the points on the object surface where the fingers will touch the object. Mathematically, a parameterized way to write this is as follows:

$$\begin{pmatrix} x_{\text{ray}}(\lambda) \\ y_{\text{ray}}(\lambda) \end{pmatrix} = \begin{pmatrix} x_p \\ y_p \end{pmatrix} + R_{\varphi_p} \begin{pmatrix} x_f \\ y_f \end{pmatrix} + \lambda R_{\varphi_p} \begin{pmatrix} m_x \\ m_y \end{pmatrix} \tag{3.18}$$

$$R_{\varphi_p} = \begin{pmatrix} \cos(\varphi_p) & \sin(\varphi_p) \\ -\sin(\varphi_p) & \cos(\varphi_p) \end{pmatrix} \tag{3.19}$$

where x_p, y_p and φ_p define the pose, i.e., the position and orientation of the end-effector. The values x_f and y_f indicate the finger position relative to the gripper pose if the gripper is fully open and m_x and m_y indicate the closing direction of the fingers relative to the pose. The rotation matrix R_{φ_p} orients the fingers according to the value φ_p. The value λ implies the location of a point on the finger trajectory. E.g., in the case of an anthropomorphic hand, the trajectory would be a ray from the initial position of the finger tip when the hand is fully open into the direction of the knuckle towards which the finger is closing. An illustration of such a model, based on measurements from the gripper used in the experiments, is given in Figure 3.3(a).



(a) Five finger hand model with a linear thumb trajectory
(b) Five finger hand model with a curved thumb trajectory

Figure 3.3: Hand model with five fingers and different types of thumb trajectories

Through observation, we found that the thumb of our gripper has a slightly curved trajectory. Thus, in some cases, a linear approximation of a finger trajectory might not be sufficient. We therefore consider curved trajectories as well, as shown in Figure 3.3(b). The basic idea is to replace the values m_x and m_y in Equation 3.18

by an ellipse and then consider only an arc of the ellipse for the trajectory. We can write this as follows:

$$\begin{pmatrix} x_{\text{ray}}(\lambda) \\ y_{\text{ray}}(\lambda) \end{pmatrix} = \begin{pmatrix} x_p \\ y_p \end{pmatrix} + R_{\varphi_p} \begin{pmatrix} x_f \\ y_f \end{pmatrix} + R_{\varphi_p} \begin{pmatrix} A_{\cos}\cos(\lambda + \varphi_{\text{off}}) \\ A_{\sin}\sin(\lambda + \varphi_{\text{off}}) \end{pmatrix} - \begin{pmatrix} x_{\text{off}} \\ y_{\text{off}} \end{pmatrix} \tag{3.20}$$

$$\begin{pmatrix} x_{\text{off}} \\ y_{\text{off}} \end{pmatrix} = R_{\varphi_p} \begin{pmatrix} A_{\cos}\cos(\varphi_{\text{off}}) \\ A_{\sin}\sin(\varphi_{\text{off}}) \end{pmatrix} \tag{3.21}$$

where $A_{\cos}$ and $A_{\sin}$ indicate the radii of the ellipse or in other words the amplitudes of the elliptic arc. The value φ_{off} is an angular offset on the arc and the values x_{off} and y_{off} imply the offset of the finger tip from the center of the ellipse in x and y direction. The value λ indicates a position on the arc relative to the initial position of the finger tip and is periodic between $\lambda = [0, 2\pi)$.

3.4 End-effector Pose Optimization

In this section, we show how to find the optimal pose, given the EFD contour as input as described in Section 3.2. The general idea is to find a pose where the fingers will reach grasping points at flat surfaces or concavities of the object. Due to the nature of the problem, where grasping points that are needed for the cost function can only be calculated after we have defined a pose, we created a nested optimization with an inner and outer optimization. In the inner optimization, we calculate grasping points for a given pose. In the outer optimization, we sample over different poses and find the pose that will lead to an optimal grasp.

3.4.1 Inner Optimization

The inner optimization solves the problem of finding grasping points between the parameterized description of the contour provided by Equations 3.3 and 3.4 and the parameterized rays of the fingers given in Equation 3.18. Our goal is to minimize the distance between the points on the EFD contour and their corresponding points on the finger trajectories. The problem is defined by:

$$\begin{aligned} \underset{\xi}{\text{argmin}} \quad & h \\ \text{subject to} \quad & l(\xi) \leq 0 \end{aligned} \tag{3.22}$$

where h is the cost function and $l(\xi)$ are the constraints. ξ indicates the intersection points between the finger trajectories and the contour and is defined as follows:

$$\xi = \begin{pmatrix} t_1 \\ ... \\ t_{\text{nF}} \\ \lambda_1 \\ ... \\ \lambda_{\text{nF}} \end{pmatrix} \tag{3.23}$$

where nF indicates the number of fingers, the t values imply where the grasping points are on the EFD contour and the λ values are the location of the grasping points on the finger trajectories. We now discuss the cost function and constraints in detail.

Cost Function

The cost function is the summed up, squared difference of distance between the points on the EFD contour and their corresponding points on the finger trajectories for all fingers. It is defined as follows:

$$h = \frac{1}{2}\sum_{i=1}^{\text{nF}} d_i^2 \tag{3.24}$$

where

$$d_i = \left\| \begin{bmatrix} d_{x,i} \\ d_{y,i} \end{bmatrix} \right\| \tag{3.25}$$

$$d_{x,i} = x_{\text{contour}}(t_i) - x_{\text{ray}}(\lambda_i) \tag{3.26}$$

$$d_{y,i} = y_{\text{contour}}(t_i) - y_{\text{ray}}(\lambda_i) \tag{3.27}$$

where the values $d_{x,i}$ and $d_{y,i}$ indicate the distance between a point pair in x and y direction. The equations for $x_{\text{ray}}(\lambda_i)$ and $y_{\text{ray}}(\lambda_i)$ depend on ρ, which is why the problem is not solvable in a single numerical optimization and we thus require a nested optimization.

Constraints

We find two kinds of inequality constraints for Equation **??**. The first type is due to kinematical limitations of the finger joints, i.e., the finger trajectories are limited in distance:

$$\lambda_i \geq 0 \qquad for\ i = 1, \ldots, \text{nF} \tag{3.28}$$

$$F_i - \lambda_i \geq 0 \qquad for\ i = 1, \ldots, \text{nF} \tag{3.29}$$

where F_i is the distance from the initial position of each finger to a fully closed position. Equation 3.28 accounts for the fingers only closing in one direction and Equation 3.29 for the maximum distance of the finger trajectories. The second type of constraint we impose on the points t_i on the EFD contour. Since the functions given in Equations 3.3 and 3.4 are periodic between $t = [0, T)$, we add the following constraints:

$$t_i \geq 0 \qquad for\ i = 1, \ldots, \text{nF} \tag{3.30}$$

$$T - t_i > 0 \qquad for\ i = 1, \ldots, \text{nF} \tag{3.31}$$

3.4.2 Outer Optimization

For a single pose ρ, we are able to calculate grasping points, given by ξ, in the inner optimization (see Section 3.4.1). We can now solve the outer optimization. We find the optimal pose by sampling over different poses and solving the inner optimization for each sample pose. The outer optimization is defined as follows:

$$\begin{aligned} \min_{\rho} \quad & f(\operatorname*{argmin}_{\xi}\ h) \\ \text{subject to} \quad & g(\rho) \leq 0 \end{aligned} \tag{3.32}$$

where f is the cost function and $g(\rho)$ are the constraints. ρ indicates the pose and is defined as follows:

$$\rho = \begin{pmatrix} x_p \\ y_p \\ \varphi_p \end{pmatrix} \tag{3.33}$$

where x_p and y_p imply the location and φ_p the orientation of the end-effector in the image plane of the optical frame. We now discuss the cost function and constraints of the outer optimization in detail.

Cost Function

The cost function we use is a weighted average between a cost of curvature and a cost of ratio:

$$f = \sum_{i=1}^{\text{nF}} \alpha_i C_{\text{curvature}}(t_i) + \beta_i C_{\text{ratio}}(t_i) \tag{3.34}$$

where the values α_i and β_i allow for adjustment of the gains for each finger individually. E.g., if it is more important for the thumb to reach a concave point than it is for the other fingers, a higher α gain can be put on the thumb. The term $C_{\text{curvature}}(t_i)$ is defined by the second derivative of the EFD contour, as shown in Equation 3.5, and is a measurement of how concave or convex a grasping point is.

The cost of ratio influences the ratio between a finger's initial distance to the object and the average initial distance of the other fingers to the object. The cost C_{ratio} for a finger i is defined as follows:

$$C_{\text{ratio}}(t_i, \lambda_{\text{virt},i}) = \text{dist}(t_i, \lambda_{\text{virt},i}) - \frac{1}{\text{nF} - 1} \sum_{j=1}^{\text{nF}} \text{dist}(t_j, \lambda_{\text{virt},j}) \tag{3.35}$$

where

$$\text{dist}(t, \lambda_{\text{virt}}) = |p_{\text{virt}} - p_{\text{grasp}}| = \left| \begin{bmatrix} x_{\text{ray}}(\lambda_{\text{virt}}) - x_{\text{contour}}(t) \\ y_{\text{ray}}(\lambda_{\text{virt}}) - y_{\text{contour}}(t) \end{bmatrix} \right|, \tag{3.36}$$

$$p_{\text{virt}} = \begin{pmatrix} x_{\text{ray}}(\lambda_{\text{virt}}) \\ y_{\text{ray}}(\lambda_{\text{virt}}) \end{pmatrix}, \tag{3.37}$$

and

$$p_{\text{grasp}} = \begin{pmatrix} x_{\text{contour}}(t) \\ y_{\text{contour}}(t) \end{pmatrix}. \tag{3.38}$$

It may occur that not all fingers close at the same velocities, which we need to consider in the cost of ratio. Thus, the values p_{virt} are virtual initial points of the fingers which may be further away or closer to the object depending on the velocity of the closing finger. The virtual initial points can be calculated with Equation 3.18 and the values λ_{virt}, which can be determined through measurement. The grasping points on the contour can be calculated with Equations 3.3 and 3.4. This allows control of the time at which fingers ideally touch the object. E.g., suppose we want all fingers to reach the object at the same time, while we observe that one finger is closing slower than all others. We measure the closing velocity for each finger and hence determine the virtual initial points. The finger that is closing slower has a virtual initial point further away to the closed finger position than the initial point of the fingertip. This way, the cost of ratio will try to move this finger closer to the object than the other fingers.

Constraints

The constraints we impose on the outer optimization are the following:

$$|x_p - A_0| - D_{\text{max},x} \leq 0 \tag{3.39}$$

$$|y_p - C_0| - D_{\text{max},y} \leq 0 \tag{3.40}$$

where A_0 and C_0 are the DC components of the EFD contour and $D_{\text{max},x}$ and $D_{\text{max},y}$ are the values for the maximum distance away from the DC components we want the initial pose to be positioned at. Hence, by limiting the possible pose locations in x and y direction, we can decrease the search space. Further, we also define the following constraints on the initial position of the fingers:

$$\text{dist}(t_i, 0) - D_{\text{min}} \geq 0 \qquad for\ i = 1, \ldots, \text{nF} \tag{3.41}$$

where the distance $\text{dist}(t_i, 0)$, with λ_{virt} values equal to zero, which indicates the actual initial location of the fingertips, can be calculated with Equation 3.36. The constraints force the initial points of the fingertips to be at a minimum distance D_{min} (in the direction of the closing finger) away from the contour in order to achieve better grasps.

Chapter 4

Experiments and Results

In this chapter, we first illustrate the setup, i.e., the used hardware, software and the workspace in Section 4.1 and then show the results of different stages of the grasp synthesis pipeline and the grasping experiments on the real robot in Section 4.2.

4.1 Setup

4.1.1 Software Implementation

The grasp synthesis pipeline was implemented with the Robot Operating System (ROS) [16] and written in C++ [17]. Visualization plots were implemented with Python [18]. For the implementation of the outer optimization (see Section 3.4.2), we created a sampling grid around the recognized contour, where each sampling point indicates a pose. We vary x and y for position and φ for different orientations of the end-effector. The grid resolution parameters are shown in Table 4.1. An example of a sampling grid is illustrated in Figure 4.1. The extent of the grid in x and y direction is restricted through the constraints of the outer optimization. The parameters used in all experiments for the constraints of the outer optimization are given in Table 4.2.

$\boldsymbol{x}_{\textbf{resolution}}$	0.05
$\boldsymbol{y}_{\textbf{resoltuion}}$	0.05
$\boldsymbol{\varphi}_{\textbf{resoltuion}}$	2°

Table 4.1: Parameters for sampling grid resolution

$\boldsymbol{D}_{\textbf{max},\boldsymbol{x}}$	0.15
$\boldsymbol{D}_{\textbf{max},\boldsymbol{y}}$	0.15
$\boldsymbol{D}_{\textbf{min}}$	0.015

Table 4.2: Parameters for constraints of the outer optimization

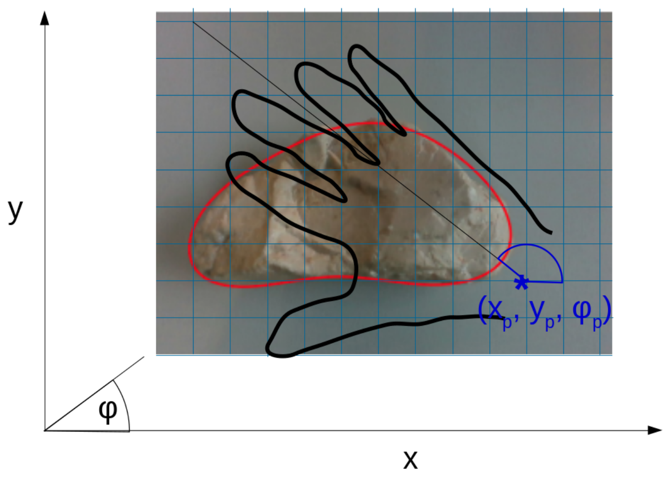
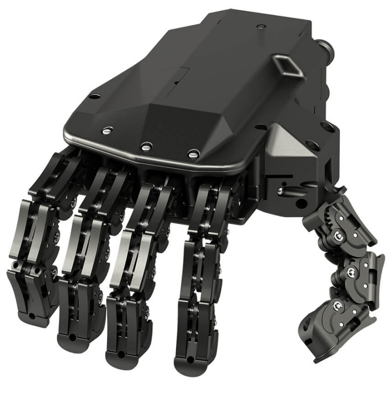


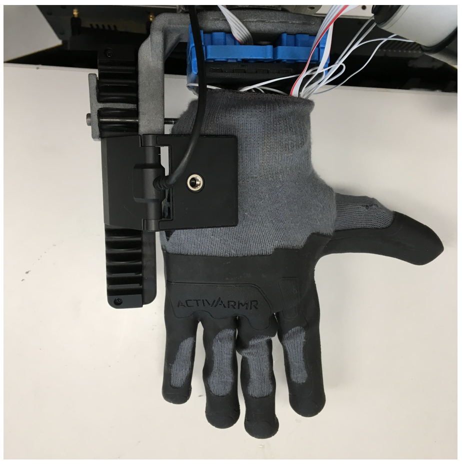
Figure 4.1: Example of a sampling grid with a sample pose of a five fingered hand

4.1.2 Hardware

An overview of the whole setup can be seen in Figure 4.3. The robot consists of an end-effector that is attached to a six DoF Anypulator robotic arm manipulator [19]. As shown in Figure 4.2(a), the end-effector used is an underactuated, anthropomorphic five-finger hand (qbHand[1]). It has one actuated DoF and is thus limited to a power grasp, i.e. a grasp where the fingers enclose an object. To get a better grip while grasping, a glove is put on the hand. Additionally, a RGB-D camera (Intel®RealSense™ SR300[2]) was mounted on the wrist as can be seen in Figure 4.2(b). The camera has a RGB camera and a depth sensor, thus captures colored images as well as depth images. The range of the depth sensor is from 0.2m to 1.5m.



(a)



(b)

Figure 4.2: (a) Detailed view of qbHand (b) Detailed view of underactuated, anthropomorphic hand with glove and wrist mounted RGB-D camera

[1]http://www.qbrobotics.com/products/qbhand/
[2]https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/realsense-sr300-datasheet1-0.pdf

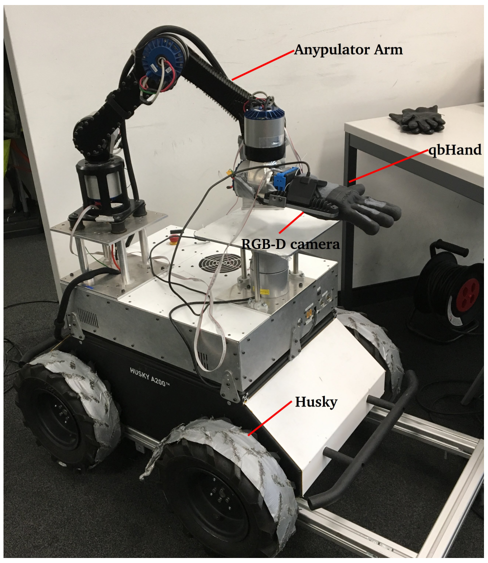


Figure 4.3: Overview of the robot with the Husky base, the Anypulator arm and the qbHand with the wrist mounted RGB-D camera

4.1.3 Workspace

For the general setup of the grasping experiments, we created a simplified workspace. The following assumptions are made:

- There is only one object in the workspace
- The object is of reasonable size for the hand to grasp
- The objects are unknown, i.e., the grasp synthesis pipeline has no prior information about the object properties.
- The object is completely visible in the optical frame
- The object is within the range of the depth sensor (0.2m to 1.5m)
- The object is placed on a table with a texture that does not interfere with the object recognition algorithms
- The hand is able to reach the desired pose in the workspace

We selected five objects with different properties such as shape, weight, texture as seen in Figure 4.4. For each try, an object was placed on a table in a random orientation and the grasp synthesis pipeline was started. A grasp was declared successful if the hand was able to grasp the object and pick it up without slippage. If the hand could not grasp it or let the object slip, the attempt was unsuccessful. In case of the hand not being able to reach the pose due to a singularity configuration, the attempt was repeated.

4.1.4 Grasp Execution

In this section, we explain how the pose calculated in the optimization is executed. Particularly, the optimal pose we calculate is in 2D and lies within the image plane, but the pose of the end-effector has six DoF in 3D. Accordingly, we want the gripper to stay parallel to the image plane. Once the end-effector is aligned with

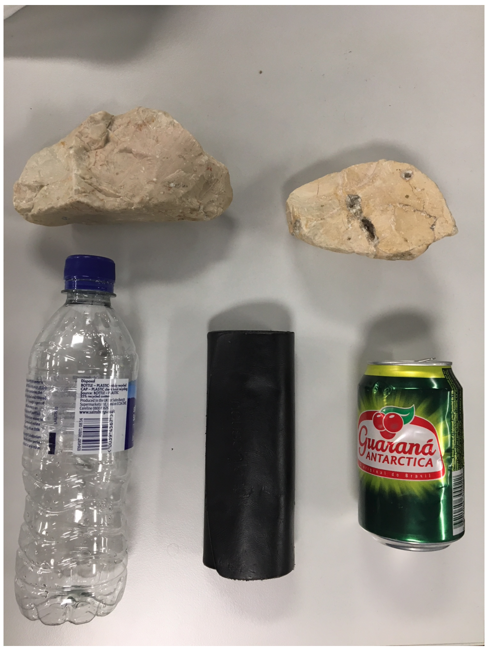


Figure 4.4: Objects used in the experiments from top left to bottom right: stone 1, stone 2, water bottle, glasses case, can

the camera's image plane, we neglect 2 rotational DoF. Thus, the only additional information we need for the grasp execution is the depth value z in the direction of the surface normal of the camera's image plane. The value z indicates how close we move our optimal pose to the object before executing the grasp and is extracted as explained in Section 3.1.1. In the current implementation, the end-effector autonomously aligns with the image plane of the camera's optical frame and moves to the calculated optimal pose within this plane. The control of the depth value z, however, makes the end-effector stop at a short distance from the object within the direction perpendicular to the image plane to avoid undesired collision with the object. The last part of the execution, i.e., lowering the hand to be slightly above or on the surface and executing the grasp, has to be carried out manually.

4.2 Results

In this section, we present the results of different stages in the pipeline. First, we take a closer look at the influence of different number of harmonics on the EFD contours in Subsection 4.2.1 and then illustrate the results of EFD contour calculations on the test objects in Subsection 4.2.2. In Subsection 4.2.3, we demonstrate how the grasp synthesis pipeline works for different hand configurations through our generic interface. Finally, we present the results of our grasping experiments with the real robot in Subsection 4.2.4.

4.2.1 EFD Harmonics Analysis

The choice of number of harmonics for the EFD contour is of utter importance for our project. As explained in Section 3.2, we need a model in which flat, large surfaces and concave areas become concave. Our goal is to find a number of harmonics that on one hand does not oversimplify the object contour and on the other hand is not too complex such that flat surfaces would not be concave anymore. In Figures 4.5(a)-(d), EFD contours with different number of harmonics for a simple stone are shown. As can be observed, only using one or two harmonics oversimplifies the object too much, as there are no concave areas at all. Using four harmonics, the largest surface has a concave curvature, which vanishes in the case of eight harmonics. Thus, for this object, the desired number of harmonics for our purposes would be four.

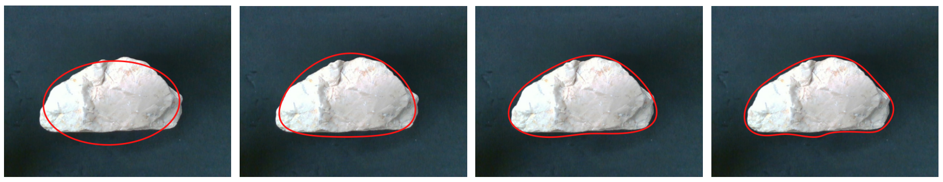

(a) 1 harmonic (b) 2 harmonics (c) 4 harmonics (d) 8 harmonics

Figure 4.5: EFD contours with different number of harmonics for a simple stone

EFD contours for a more complex object, i.e., an object that has more edges and a more complex symmetry, is shown in Figures 4.6(a)-(d). We see that some concavities become convex with four harmonics. Using eight harmonics, these areas remain concave and hence higher harmonics would lead to a more useful model of the object contour. Thus, we notice that the optimal choice of the harmonics may not be the same for different objects.

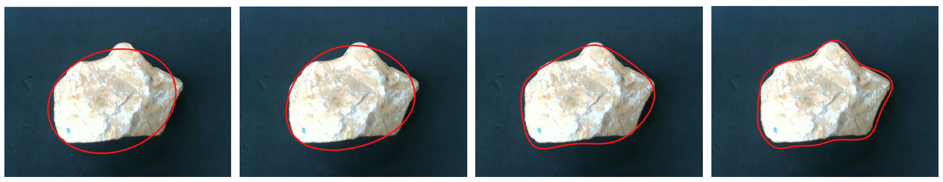
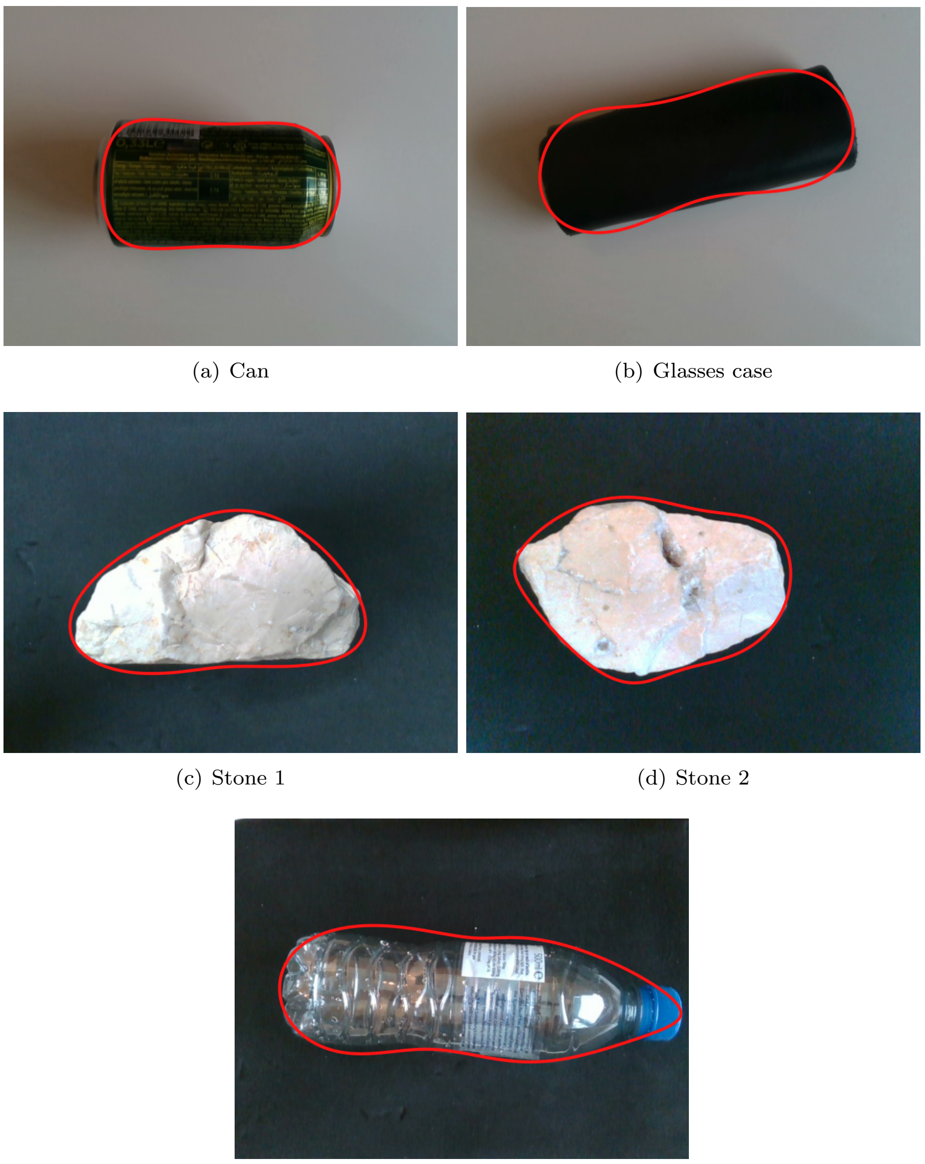

(a) 1 harmonic (b) 2 harmonics (c) 4 harmonics (d) 8 harmonics

Figure 4.6: EFD contours with different number of harmonics for a complex stone

4.2.2 EFD Results

In this section, we show the resulting EFD contours using four harmonics for the test objects in Figures 4.7(a)-(e). As can be easily observed, larger surfaces tend to have a concave curvature, while smaller surfaces and edges become convex. For stone 2, one smaller concavity is missed at the top left of the object, but the two largest surfaces still show a concave curvature. Since it is not possible to select the number of harmonics used in the EFD calculations adaptively per object in the current implementation, we had to find the balance between oversimplification and being too exact. We determined empirically that using four harmonics works well as a compromise. For further visualization of different harmonics for the test objects, refer to Figures A.1-A.4 in the appendix.


(a) Can (b) Glasses case

(c) Stone 1 (d) Stone 2

(e) Water bottle

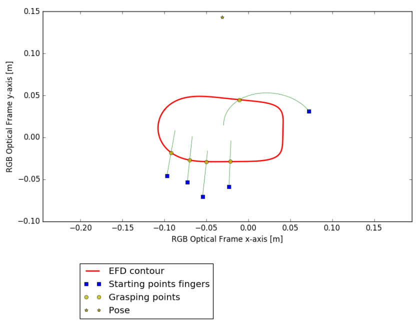
Figure 4.7: EFD contours with 4 harmonics for the test objects

4.2.3 Synthesis Results for Different Hand Configurations

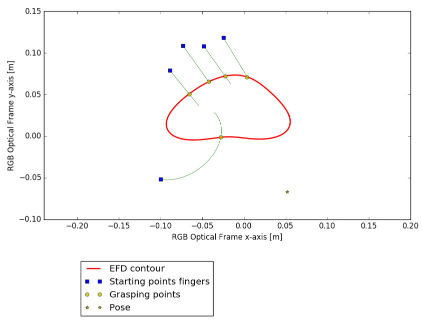
In this section, we show the results of the end-effector pose optimization on two test objects for three different hand configurations. First, we illustrate a configuration with a high gain on one finger for the cost of curvature. Secondly, we show that the synthesis pipeline also works for a three finger gripper with equally weighted cost of ratio and cost of curvature. Lastly, we present the configuration that was used for the grasping experiments in Section 4.2.4.

Five Fingers with Curved Thumb Trajectory

In this case, we configured the end-effector to have five fingers with a thumb that has a curved trajectory (see Equation 3.20) and assuming all fingers close at the same velocities. The gains were set such that the cost of curvature for the thumb was the most important factor, while having relatively smaller gains on the cost of ratio and almost completely neglecting the gain on the other four fingers. The parameters for the configuration can be found in Table A.1 in the appendix. As shown in Figure 4.8, the thumb has grasping points at the larger surfaces of both objects as desired. In Figure 4.8(a), we observe the ratio not having a significant influence on the result for this object, as the thumb reaches the object later than the other fingers. In Figure 4.8(b), two of the fingers have grasping points close to a convex local extrema, which is due to the low gains on these fingers.



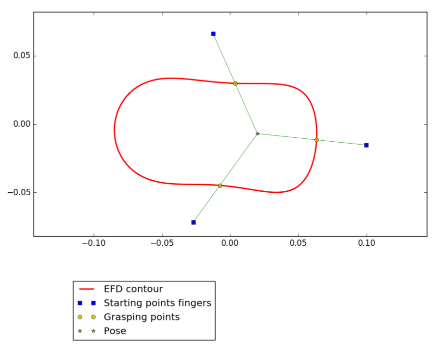
(a) Optimal pose for the can



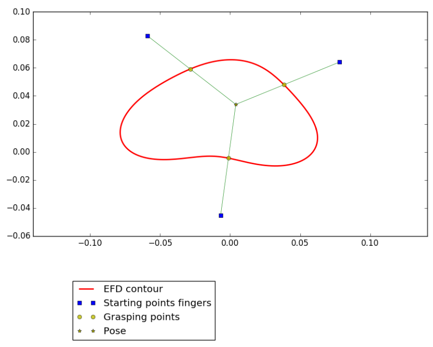
(b) Optimal pose for stone 1

Figure 4.8: Output of grasp synthesis for two test objects with a five finger hand and a curved thumb trajectory

Three Finger Gripper



(a) Optimal pose for the can



(b) Optimal pose for stone 1

Figure 4.9: Output of grasp synthesis for two test objects with a three finger gripper

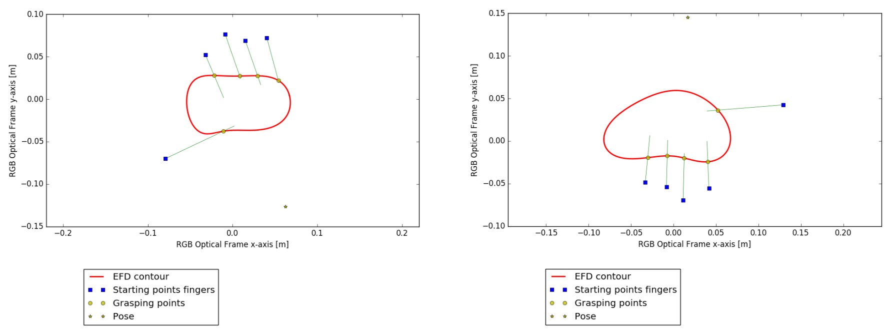
To further demonstrate that the pipeline also works with other configurations, we show the results for a gripper with three fingers in Figure 4.9. For this configuration, the gains for cost of ratio and cost of curvature were set to have a roughly equal influence on the cost function and the closing velocities of the fingers are assumed to be equal. As can be seen in both figures, the ratio between initial position of the finger tips and grasping points is similar for all fingers. Furthermore, the grasping points are on the two large surfaces with concave curvature of the object in Figure 4.9(a) and on the three larger surfaces of the object in Figure 4.9(b), which is what we desired. For more information about the parameters for this configuration, refer to Table A.2 in the appendix.

Five Fingers with Linear Thumb Trajectory

In this case, we configured the end-effector to have five fingers with a thumb that has a linear trajectory. We found this to be the most accurate 2-D model for our underactuated, anthropomorphic hand. Thus, for the experiments in Subsection 4.2.4, we used this configuration. The details of the parameters are given in Table 4.3. The α gains and β gains are not normalized in the current implementation. By trying different parameter sets through trial and error for the kinematic power grasp of our underactuated hand, we came to the conclusion that grasps are most successful when:

- all fingers have equal gains α for curvature
- all fingers have equal gains β for the ratio
- the gains for curvature α have a slightly larger influence on the cost function than the gains for the ratio β.
- the thumb is virtually shortened for the cost of ratio, as it starts its closing motion before the other fingers and has the fastest closing velocity.
- the ring and little fingers are also virtually shortened, because they start closing before the index and middle finger do, but after the thumb.

In the ideal case, given the stated observations, all fingers would grasp larger surfaces or concavities instead of edges. Moreover, the thumb is ideally located farther away to the object contour than the other four fingers. The ring and little finger should have a slightly longer distance to the object surface than the index and middle finger. In Figure 4.10(a), the effect of the cost of ratio can be observed, as the thumb is located farther away from the contour than the other four fingers. Also, the ring and little finger show larger distances from initial to grasping points than the index finger. Furthermore, all fingers but the little finger reach grasping points in areas with concave curvature. In Figure 4.10(b), the thumb is again located farther away than the other fingers as expected. The grasping point of the thumb is on a smaller surface with convex curvature, while the other four fingers touch the object on the largest surface with concave curvature. The ring and little finger may touch the object first during grasp execution, as they are at about the same distance to the object as the index finger.



(a) Optimal pose for can (b) Optimal pose for stone 1

Figure 4.10: Output of grasp synthesis for two test objects with a five finger hand and a linear thumb trajectory

	Thumb	**Index**	**Middle**	**Ring**	**Little**
$\boldsymbol{x_f}$	-0.115	-0.03	0.0	0.02	0.045
$\boldsymbol{y_f}$	0.1	0.2	0.215	0.2	0.195
$\boldsymbol{m_x}$	-0.994	-0.067	0.0	0.0	0.071
$\boldsymbol{m_y}$	-0.106	0.998	1.0	1.0	0.997
$\boldsymbol{F}$	0.09	0.055	0.055	0.055	0.055
$\boldsymbol{\lambda_{\mathrm{virt}}}$	-0.03	0.0	0.0	-0.01	-0.01
$\boldsymbol{\alpha}$	25.0	25.0	25.0	25.0	25.0
$\boldsymbol{\beta}$	10000.0	10000.0	10000.0	10000.0	10000.0

Table 4.3: Parameters for a five finger hand with a linear thumb trajectory

4.2.4 Results Grasping Experiments

In this section, we present the results of the grasping experiments, carried out in the simplified workspace shown in Subsection 4.1.3. The parameters used for the model of the hand are given in Table 4.3. In Table 4.4, the results of the grasping experiments are illustrated.

Test ID	**Can**	**Case**	**Bottle**	**Stone 1**	**Stone 2**
1	success	fail	success	fail	success
2	success	fail	success	success	fail
3	fail	success	success	fail	success
4	success	success	success	fail	fail
5	success	fail	fail	fail	success
6	success	fail	success	fail	fail
7	success	success	success	fail	success
8	fail	fail	fail	fail	success
9	success	fail	fail	fail	success
10	succes	success	success	success	fail
success rate	**80%**	**40%**	**70%**	**20%**	**60%**

Table 4.4: Results of the grasping experiments

For the can and the water bottle, the success rates are at 80% and 70%. Both these objects have properties such as weight, texture and shape that provide favorable grasping conditions. Reasons of failure were due to position offsets and the object being moved before all fingers reached the surface. The glasses case could be grasped successfully four out of ten times, where reasons of failure were mainly due to all grasping points laying on the same surface. Stone 1 and stone 2 show success rates of 20% and 60%. Stone 1 is the most complex test object, as it is the heaviest and largest object. Even though the shape of stone 2 is a little more complex, it is significantly lighter and smaller than stone 1. Failures when grasping stone 1 were mainly caused by the fingers not being able to exert enough force on the object's surface, even when the synthesized grasps looked good. A further discussion of these results is given in Chapter 5.

Chapter 5

Discussion

In Section 4.2.4, we saw that it is possible to reach success rates of up to 80% for easily graspable objects. On the other hand, we observed that for a more complex object, the success rate is as low as 20%. In the experiments, the reasons of failure were the following:

1. A position offset when trying to reach the optimal pose
2. All grasping points laying on the same surface
3. The fingers not being able to exert enough force on the object
4. The object being moved by fingers that reach the object surface first
5. Collision with the object surface when trying to reach the grasping points

The first failure can occur due to estimation errors in the object recognition part of the framework. The depth sensor of the RGB-D camera has an accuracy range of +-5%, which can cause the object contour to become larger or smaller than it actually is. A calibration of the camera extrinsics may lead to minor improvements of estimation errors. Also, there are small angular offsets between the RGB-D camera and the wrist that are very difficult to measure. This can lead to errors in the estimation of the object's position. The second failure occurs when there is a large surface, e.g., the long sides of the glasses case, such that possibly all grasping points will touch the object on the same side. It may be prevented by adding a force closure test. For the third type of failure, poses that seemed to produce successful grasps failed. The fingers would reach the grasping points but slip out of the hand when trying to lift them up. This type of failure occurred on the heavier objects, i.e., stone 1 and stone 2. We think this is due to the fingers not being able to exert enough force or create enough friction to reach a stable grasp on the object. The object being moved by fingers that reach the object first is an error we try to prevent with the cost of ratio. Such an error shows that either the gains for the cost of ratio in the optimization are set too low or the estimation of the closing velocities of the fingers are not accurate enough. The latter may certainly be the case, since we estimated the velocities by visual observation. We also noticed that the fingers not only close at different velocities, but also start their motion at different times, which complicates the issue further. The thumb starts closing first, with the ring and index finger following shortly after. After a small delay, the index and middle finger also start closing, but at a slightly higher velocity than the ring and little finger, until they catch up with each other at about two thirds of their closing trajectories. From this point on, these four fingers close roughly synchronous. We therefore notice that it is very difficult to find an accurate model for the cost of ratio. Thus, there may be the requirement for a more sophisticated or improved method to consider this correctly. Another way to avoid this problem would be to add tactile sensors to the finger tips. Once a conact with the object is detected, the grasp execution could be aborted if not all other fingers reach the surface within a short time interval. This solves the problem of false grasps being executed, but the challenge of finding a better model for the cost of ratio still remains. Collision when trying to reach grasping points may occur because of our approximation of the finger trajectories as rays. The thickness of the fingers are not considered in the current implementation. Thus, it may happen that a ray comes fairly close to the object surface before reaching the desired grasping point, which will ultimately lead to a collision in the experiments. This can be fixed by including a collision detection mechanism either in the grasp synthesis or in the grasp execution with tactile sensors. The latter could be implemented by estimating the time the fingers need to reach the object surface from a fully opened position. If the time difference between the detected contact and the estimation exceeds a certain value, the grasp execution could be aborted.

Chapter 6

Conclusion

The goal of this project was to find an optimal gripper pose in order to grasp objects with unknown properties. We implemented a pipeline that takes RGB-D images as input and calculates an optimal gripper pose as output. Under the assumption that grasp synthesis in 2D is sufficient, we use Elliptic Fourier Descriptors to model the contour of an object. Then, we find an optimal pose through maximizing the curvature value of grasping points as well as minimizing the difference of ratios of the distance between the object surface and the closing fingers. We tested the algorithm on a real robot, using an underactuated, anthropomorphic hand, that can execute a power grasp with one actuated DoF, as an end-effector. Results show success rates as high as 80% easily graspable objects and as low as 20% for more complex objects in a simplified workspace. This project can be considered a successful step into creating a grasp synthesis pipeline that can be generically used for various types of end-effectors.

6.1 Outlook

There are several things that can be done in order to improve and extend the pipeline. The following steps could be taken:

1. The control of the end-effector in the z-direction (perpendicular to the image plane) should be improved, such that the grasp can be executed fully autonomous.

2. A force closure test can be added to guarantee stability of the optimal grasp. Another way to do this for an anthropomorphic hand would be to calculate an estimate of the principle axis of the object and check whether the thumb's grasping point lays on the other side of the axis than the grasping points of the other fingers.

3. The number of harmonics of the Elliptic Fourier Descriptors could be adaptive to the object complexity, i.e., the more edges on the object contour, the higher the order of the EFD should be. This would lead to a more realistic modeling of the objects and prevent oversimplification, e.g., where concave regions of complex objects might be modeled as convex due to a low order harmonic of the EFD.

4. The efficiency of the optimization can be improved by limiting the search space and thus lowering the computational load. This could be done by imposing constraints gathered from observation of human grasping such as suggest by Krug [3].

On the hardware side, tactile sensors could be added to the fingertips. This could help improve grasps during execution and prevent unwanted contacts. Furthermore, parameters used for the optimization were determined empirically through trial and error. A more thorough analysis for different parameters could be carried out to find an optimal parameter set for a given end-effector. Lastly, an idea to extend this approach to not only having a single view for grasp synthesis would be to systematically or randomly check different viewpoints and find the pose that has the lowest overall cost.

Bibliography

[1] R. F. Becker, "The cerebral cortex of man. by wilder penfield and theodore rasmussen." *American Journal of Physical Anthropology*, vol. 11, no. 3, pp. 441–444, 1953.

[2] N. H. M. London. (2014) Motor homunculus.

[3] R. Krug, "Optimization-based robot grasp synthesis and motion control," Ph.D. dissertation, Örebro University, School of Science and Technology, 2014.

[4] C. Borst, M. Fischer, and G. Hirzinger, "Grasping the dice by dicing the grasp," vol. 4, pp. 3692 – 3697 vol.3, 11 2003.

[5] S. El-Khoury and A. Sahbani, "A new strategy combining empirical and analytical approaches for grasping unknown 3d objects," *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 497 – 507, 2010.

[6] W. S. Howard and V. Kumar, "On the stability of grasped objects," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 6, pp. 904–917, Dec 1996.

[7] A. Sahbani, S. El-Khoury, and P. Bidaud, "An overview of 3d object grasp synthesis algorithms," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 326 – 336, 2012.

[8] B. Calli, "Fundamentals of grasp synthesis," Ph.D. dissertation, TU Delft, 2015.

[9] M. Richtsfeld and M. Vincze, "Grasping of unknown objects from a table top," 10 2008.

[10] K. Hsiao, S. Chitta, M. Ciocarlie, and E. G. Jones, "Contact-reactive grasping of objects with partial shape information," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2010, pp. 1228–1235.

[11] D. Schiebener, J. Schill, and T. Asfour, "Discovery, segmentation and reactive grasping of unknown objects," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, Nov 2012, pp. 71–77.

[12] F. Cordella, L. Zollo, A. Salerno, E. Guglielmelli, and B. Siciliano, "Experimental validation of a reach-and grasp optimization algorithm inspired to human arm-hand control," in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Aug 2011, pp. 8150–8153.

[13] L. F. Gallardo and V. Kyrki, "Detection of parametrized 3-d primitives from stereo for robotic grasping," in *2011 15th International Conference on Advanced Robotics (ICAR)*, June 2011, pp. 55–60.

[14] B. Calli, "Grasping of unknown objects via curvature maximization using active vision," Ph.D. dissertation, TU Delft, 2015.

[15] F. P. Kuhl and C. R. Giardina, "Elliptic fourier features of a closed contour," *Computer Graphics and Image Processing*, vol. 18, no. 3, pp. 236 – 258, 1982.

[16] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[17] B. Stroustrup, *The C++ Programming Language*, 3rd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2000.

[18] G. Rossum, "Python reference manual," Amsterdam, The Netherlands, The Netherlands, Tech. Rep., 1995.

[19] K. Bodie, C. D. Bellicoso, and M. Hutter, "Anypulator: Design and control of a safe robotic arm," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 1119–1125.

Appendix A

A.1 Glossary

A.1.1 Symbols

α	gains for fingers on curvature
β	gains for fingers on ratio cost
λ	point on finger trajectory
q	point in the depth image
nF	number of fingers
ρ	pose of end-effector
t	point on EFD contour
ξ	grasping points
z	depth value

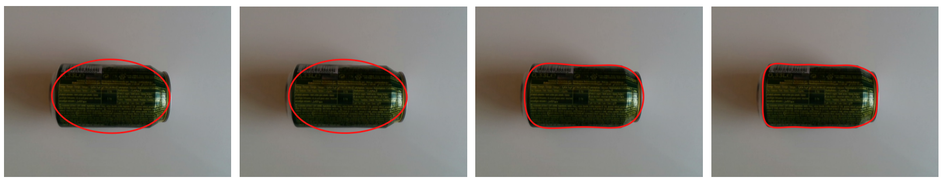
A.1.2 Indices

f	finger
p	pose
virt	virtual
x	x axis
y	y axis

A.1.3 Acronyms and Abbreviations

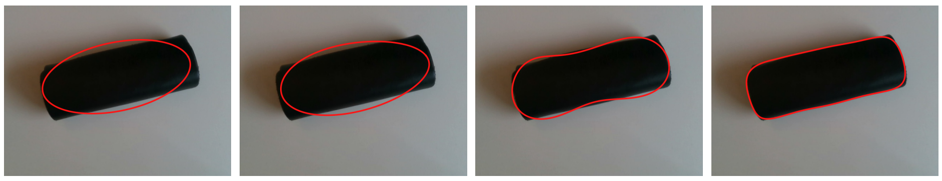
DoF	Degree of Freedom
EFD	Elliptic Fourier Descriptors
ETH	Eidgenössische Technische Hochschule
RGB-D	Red, Green, Blue and Depth

A.2 EFD Results for Test Objects



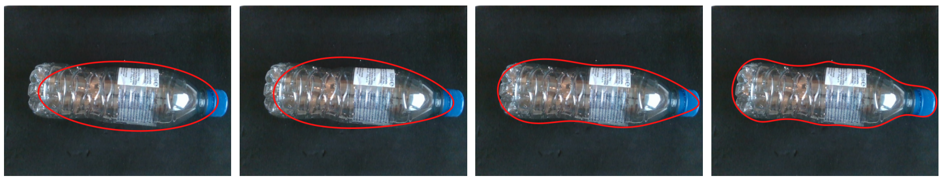
(a) 1 harmonic (b) 2 harmonics (c) 4 harmonics (d) 8 harmonics

Figure A.1: EFD contours with different number of harmonics for a can



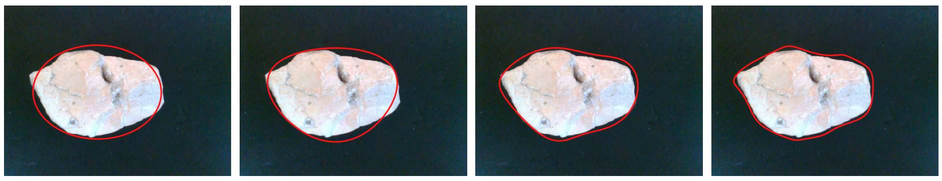
(a) 1 harmonic (b) 2 harmonics (c) 4 harmonics (d) 8 harmonics

Figure A.2: EFD contours with different number of harmonics for a glasses case



(a) 1 harmonic (b) 2 harmonics (c) 4 harmonics (d) 8 harmonics

Figure A.3: EFD contours with different number of harmonics for a water bottle



(a) 1 harmonic (b) 2 harmonics (c) 4 harmonics (d) 8 harmonics

Figure A.4: EFD contours with different number of harmonics for stone 2

A.3 Parameters for Different Hand Configurations

	Thumb	**Index**	**Middle**	**Ring**	**Little**
$\boldsymbol{x_f}$	-0.115	-0.03	0.0	0.02	0.045
$\boldsymbol{y_f}$	0.1	0.2	0.215	0.2	0.195
$\boldsymbol{m_x}$	-	-0.067	0.0	0.0	0.071
$\boldsymbol{m_y}$	-	0.998	1.0	1.0	0.997
$\boldsymbol{F}$	0.09	0.055	0.055	0.055	0.055
$\boldsymbol{A_{\mathbf{cos}}}$	0.055	-	-	-	-
$\boldsymbol{A_{\mathbf{sin}}}$	0.04	-	-	-	-
$\boldsymbol{\varphi_{\mathbf{off}}}$	3.77	-	-	-	-
$\boldsymbol{\lambda_{\mathbf{virt}}}$	0.0	0.0	0.0	0.0	0.0
$\boldsymbol{\alpha}$	100.0	1.0	1.0	1.0	1.0
$\boldsymbol{\beta}$	10000.0	10000.0	10000.0	10000.0	10000.0

Table A.1: Parameters for a five finger hand with a curved thumb trajectory

	Finger 1	**Finger 2**	**Finger 3**
$\boldsymbol{x_f}$	0.0	0.069	-0.069
$\boldsymbol{y_f}$	0.08	-0.04	-0.04
$\boldsymbol{m_x}$	0.0	0.866	-0.866
$\boldsymbol{m_y}$	1.0	-0.5	-0.5
$\boldsymbol{F}$	0.08	0.08	0.08
$\boldsymbol{\lambda_{\mathbf{virt}}}$	0.0	0.0	0.0
$\boldsymbol{\alpha}$	25.0	25.0	25.0
$\boldsymbol{\beta}$	50000.0	50000.0	50000.0

Table A.2: Parameters for a three finger gripper