

# Grasp Optimization from Learning-based Initial Guess

Scientific thesis for the procurement of the degree B.Sc.  
from the Department of Electrical and Computer Engineering at the  
Technical University of Munich.

**Supervised by** Univ.-Prof. Dr.-Ing. Sandra Hirche  
M.Sc. Jan Brüdigam  
Chair of Information-Oriented Control

**Submitted by** Kaustabh Paul  
Donaustrasse 10a  
91052 Erlangen  
01733727280

**Submitted on** Munich, 21.04.2023





## **Abstract**

This thesis introduces an optimization-based approach to improving a robotic grasp configuration resulting from deep reinforcement learning. The grasp optimization of multifingered hands can be divided into two subcategories:

- a) Improving contact positions through methods such as force and moment residual control.
- b) Grasping-force optimization by minimizing an objective function subject to friction and balance constraints of external forces.

The optimized grasps are inspected by observing grasp quality metrics. The thesis is comprised of three parts, a literature review, theoretical methods for grasp optimization and its practical implementation in the simulation environment MuJoCo. Furthermore, we will analyze the simulation results which prove the feasibility of our approach.



# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Related Work . . . . .	6
<b>2</b>	<b>Background</b>	<b>9</b>
2.1	Contact Friction Models . . . . .	9
2.2	Linearized Friction Cone . . . . .	10
2.3	Grasping Fundamentals . . . . .	12
2.3.1	Grasp Matrix . . . . .	12
2.3.2	Force Closure . . . . .	15
2.3.3	Equilibrium Grasps . . . . .	16
2.4	Grasp Quality Metrics . . . . .	16
2.4.1	Grasp Position Quality Metrics . . . . .	16
<b>3</b>	<b>Contact Position Optimization</b>	<b>19</b>
3.1	Objective Functions . . . . .	19
3.2	Force Residual Control . . . . .	20
3.2.1	Surface Parametrization . . . . .	20
3.2.2	Force Residual Control Algorithm . . . . .	21
3.3	Moment Residual Control . . . . .	23
3.4	Switching Grasp Control . . . . .	25
3.5	Implementation . . . . .	25
<b>4</b>	<b>Force Optimization</b>	<b>29</b>
4.1	Lagrangian Method . . . . .	29
4.1.1	Constraints . . . . .	29
4.1.2	Objective . . . . .	32
4.1.3	Solution . . . . .	33
4.2	Implementation . . . . .	34
4.2.1	Projection on Surface Frame . . . . .	34
4.2.2	Correlation Cone Edges and Optimal Force . . . . .	36
<b>5</b>	<b>Evaluation</b>	<b>37</b>
5.1	Position Optimization Results . . . . .	37
5.2	Force Optimization Results . . . . .	39

5.3 Combined Optimization Results . . . . .	40
<b>6 Discussion</b>	<b>43</b>
6.1 Contact Position Optimization . . . . .	43
6.1.1 Residual Control Algorithm Generalizations . . . . .	43
6.1.2 Further Issues . . . . .	46
<b>7 Conclusion</b>	<b>47</b>
<b>A</b>	<b>49</b>
A.1 Notation . . . . .	49
<b>List of Figures</b>	<b>51</b>
<b>Bibliography</b>	<b>55</b>

# Chapter 1

## Introduction

The ability of robots to dexterously grasp objects is one of the major challenges in robotics imitating humans' manipulation capabilities. One of the foremost problems in grasping is deciding where to make contact with the object and how much force to apply in order to ensure properties such as the ability to resist external disturbances. For potential future application purposes, this process also needs to be computationally efficient.

The goal of grasp position optimization is to find contact points such that the grasp can remain robust while requiring lower optimal grasp forces. When it comes to optimizing the grasping force one has to minimize the applied contact wrench while ensuring that the grasp doesn't slip by remaining in the friction cone and resisting external wrenches on the object such as gravity. This approach requires knowledge of the friction coefficients and the object's geometry.

In a recent work on robotic grasping based on reinforcement learning, a framework for grasping objects with various grippers on a 7 degrees of freedom Panda robot was proposed [SBC<sup>+</sup>22]. Despite having a relatively high success rate, those grasps need to be evaluated and tested for applicability in real life since they have only been tested in the simulation environment MuJoCo so far. Aspects such as grasp stability, robustness and grasp force optimization are required before the framework can have a reasonable practical implementation. This thesis will propose optimization strategies for the grasps considered, and evaluate their performance using various quality metrics.

After covering the necessary mathematical grasping foundations in Chapter 2, this thesis will begin with grasp position optimization in Chapter 3 where we will first propose general algorithms that ameliorate the contact positions with regard to two different objective functions. The optimization will subsequently be evaluated by analyzing the improvements made compared to the initial grasps. In Chapter 4, we will present a Lagrangian optimization model that incorporates various constraints. The resulting optimization problem will be evaluated through simulation results in



Chapter 5. Additionally, in Chapter 6, we will explore several advances for the position optimization algorithms, which aim to make them applicable for grasping more complex object geometries. Finally, we will conclude the thesis by assessing the proposed optimization methods and suggesting further improvements.

### Problem Statement

Given an initial guess for contact points and grasp force from the reinforcement learning framework, we want to find optimal contact points and force given corresponding metrics, such that the overall grasp is more stable and resistant to disturbances.

## 1.1 Related Work

Grasp planning algorithms are the foundation for further optimization algorithms and can take different approaches. The possible methods are shown in the table in Figure 1.1. The initial algorithm that provides us with the sample grasps [SBC<sup>+</sup>22]

Grasping algorithms						
<b>Model Based</b> The geometry or shape of the object to be grasped is given a-priori. Planning happens without integrating sensor information.			<b>Model-less</b> Shape is not known a-priori and the robot uses sensors to acquire information about the object to be grasped.			
<b>Geometric only</b>		<b>Data Driven</b>	<b>Vision based</b>		<b>Touch based</b>	
<b>Consider Object Only</b>	<b>Consider Object and Environment</b>		<b>Construct model</b>	<b>Operate in feature space</b>	<b>Construct model</b>	<b>Operate in feature space</b>

Figure 1.1: Different variants of grasping algorithms according to [CLFVW16].

uses reinforcement learning, a model-based data-driven approach. Therefore, we will continue using model-based methods for the optimization as well. Methods for grasp optimization usually take one aspect of grasping into account, such as force optimization [XWF04, HTL99] or finding the ideal contact points [LHBR12, PFG10]. Those approaches oftentimes assume ideal initial contact positions and knowledge about object geometry. Furthermore, methods that attempt to solve both issues at once tend to be more complex as they do not use a uniform surface parametrization [ZW03]. Hence, our objective is to combine two independent optimization techniques for contact force and optimization that use a uniform surface parametrization and remain efficient.

Force optimization problems are oftentimes solved by transforming friction constraints into matrices, converting those into linear matrix inequalities (LMI) and optimizing them for an objective function with different numerical methods such as semidefinite programming or gradient flow methods [BHM96, BFM97, HSSR05,

LL04]. Another technique finds the optimal grasping force by trying to fulfill the requirements for closure properties by ensuring that the forces remain in their corresponding friction cones [JC10, DMT18]. These methods cannot be associated with an explicit type of grasping algorithm since it usually depends on how it they are implemented.

Contact point optimization methods often require knowledge about the gripper kinematics and object surface geometry, which cannot always be provided [LHBR12]. As a result, many procedures ignore gripper kinematics and do not require an object model but rather use touch-based methods with sensors to get more object surface information [LHBR12, PFG10]. But those methods can easily be changed from sensor-based to model-based, given sufficient knowledge about object geometry.

Contact position optimization deals with finding ideal locations of contact points such that their grasp configuration satisfies desired metrics. It is usually conducted by trying to iteratively maximize or minimize an objective function. However, each one tries to fulfill a different goal. For instance, the authors of [LHBR12] try to maximize the grasp's stability and manipulability by defining nonlinear objective functions that take gripper kinematics into account. These objectives are used to avoid singularities. However these goals do not always focus on improving grasp quality. In contrast, objective functions such as the ones presented in [PFG10, ZW03] attempt to find contact configurations for force closure grasps, which is our desired goal. Since the latter involves both force and position optimization using different mathematical methods, we have decided to focus solely on position optimization for simplicity and clarity. Therefore, our position optimization algorithm is based on the ideas presented in [PFG10].



# Chapter 2

## Background

### 2.1 Contact Friction Models

There are various types of contact models between a finger and an object surface. In this thesis, we will focus on the three most common static friction models, which describe the wrench components that can be transmitted through a static contact. The following models are summarized from [RB19, Li94, LMSB14]. The normal forces are denoted as  $f_n$ , the tangential forces as  $f_t$  and  $f_o$  and the torque in normal direction as  $\tau$ .

#### 1. Frictionless point contact (FPC)

There is no friction between the finger and object in a frictionless point contact, which is why forces can only be applied along the surface normal of the body. Hence, no torques or tangential forces can be applied. This contact is also often referred to as a hard finger contact (HF). The only requirement for such contacts is that the normal force stays positive, which can be described with following inequality:

$$f_n \geq 0. \quad (2.1)$$

#### 2. Point contact with friction (PCWF)

A point contact with friction is used to model smaller contact patches as no significant friction moments exist. In this case, we can use the Coulomb friction model, which allows applied tangential forces to be proportional to the normal force multiplied by a friction coefficient  $\mu$ , which is dependent on the surface material. As long as the tangential force satisfies the following inequality the contact cannot slip on the surface:

$$f_t \leq \mu f_n. \quad (2.2)$$

The applied force on the contact, such as the blue arrow in Figure 2.1, must lie

within a friction cone, which is centered around the surface normal with half-angle  $\beta = \arctan(\mu)$ .

The circular friction cone  $FC_i$  for the  $i$ -th contact is described by:

$$FC_i = \{f \in \mathbb{R}^3 | \sqrt{f_{t_i}^2 + f_{o_i}^2} \leq \mu f_{n_i}, f_{n_i} \geq 0\}. \quad (2.3)$$

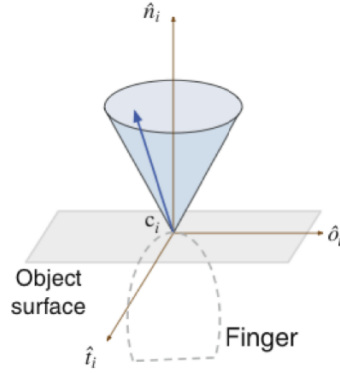


Figure 2.1: Visualization of nonlinear friction cone [LMSB14],

### 3. Soft finger contact (SFC)

The soft contact model is more realistic as it allows torques to be applied along the contact normal in addition to forces in the friction cone. The torque is limited by a torsional friction coefficient  $\gamma$ :

$$FC_i = \{f \in \mathbb{R}^3, \tau \in \mathbb{R} | \sqrt{f_{t_i}^2 + f_{o_i}^2} \leq \mu f_{n_i}, f_{n_i} \geq 0, |\tau_{n_i}| \leq \gamma f_{n_i}\}. \quad (2.4)$$

## 2.2 Linearized Friction Cone

A friction cone can also be approximated with a pyramid, which has a polyhedral base and multiple surface vectors. For a contact point with surface normal in  $z$ -direction, the  $j$ -th friction cone edge vector can be written as:

$$f_{j,c_i} = \begin{bmatrix} \mu \cos\left(\frac{2\pi(j-1)}{n}\right) \\ \mu \sin\left(\frac{2\pi(j-1)}{n}\right) \\ 1 \end{bmatrix} \in \mathbb{R}^3, \quad j = 1 \dots n. \quad (2.5)$$

An exemplary linear friction cone is shown in Figure 2.2.

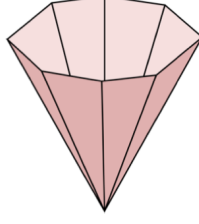


Figure 2.2: Visualization of linearized friction cone [Boh19].

We assume that the number of edge vectors for a linearized friction cone is the variable  $s = 2n$  with  $n \in \mathbb{N}$ . Measurements have shown that setting  $s = 2^n$  delivers better results for force optimization. An arbitrary PCWF contact force that satisfies the friction constraints must lie in its corresponding friction cone. Therefore, it can be denoted as a linear combination of the  $s$  edges of the approximated pyramid:

$$f_{c_i} \in FC_i \Rightarrow \exists \alpha_{k,c_i} \geq 0, \forall k = 1 \dots s : f_{c_i} = \sum_{k=1}^s \alpha_{k,c_i} f_{k,c_i}, \quad (2.6)$$

where  $\alpha_{k,c_i}$  are the entries of the parameter vector  $\alpha_{c_i}$ , which can also be written as:

$$\alpha_{c_i} = \begin{bmatrix} \alpha_{1,c_i} \\ \vdots \\ \alpha_{s,c_i} \end{bmatrix} \in \mathbb{R}^s. \quad (2.7)$$

In order to depict equation (2.6) as a matrix equation we need to define a matrix  $F_{c_i}$ , which contains all  $s$  FC edge vectors in its columns:

$$F_{c_i} = [f_{1,c_i} \dots f_{s,c_i}] \in \mathbb{R}^{3 \times s}. \quad (2.8)$$

Now the linearized friction cone constraint can be described as:

$$\begin{aligned} f_{c_i} &= F_{c_i} \alpha_{c_i}, \\ \alpha_{c_i} &\geq 0. \end{aligned} \quad (2.9)$$

It states that  $f_{c_i}$  only lies in the FC if there exists an  $\alpha_{c_i}$  that satisfies the equality. The equation (2.9) can also be considered as a linearized friction cone constraint for PCWF contacts.

The FC constraints for SFC require that the force components remain in the polyhedral cone and the torque component fulfill the additional inequality constraint mentioned in (2.4).

The set of combined friction cones for all  $n_c$  contacts in a grasp is the Cartesian product of each corresponding friction cone:

$$FC = FC_1 \times \dots \times FC_{n_c}. \quad (2.10)$$

## 2.3 Grasping Fundamentals

In order to present a mathematical model for robotic grasping, some fundamental notations need to be introduced: In the following, a generalized force (wrench) and a generalized velocity (twist) will be defined [LMSB14]. The wrench at contact point  $i$  is a vector

$$\mathcal{F}_i = \begin{bmatrix} f_i \\ \tau_i \end{bmatrix} = \begin{bmatrix} f_i \\ (c_i - p) \times f_i \end{bmatrix} \in \mathbb{R}^6, \quad (2.11)$$

with force  $f_i$  and torque  $\tau$ . The vectors  $c$  and  $p$  are the position of the contact point and the position of the object's center of mass. From now on their distance will be referred to as  $c_i - p = p_{oc_i}$ .

The twist is denoted as a vector

$$\mathcal{V}_i = \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} \in \mathbb{R}^6, \quad (2.12)$$

with velocity  $v$  and angular velocity  $\omega$ . The two most important matrices for dealing with grasp analysis are the grasp matrix  $G$  and the hand Jacobian  $J$ . The former is responsible for mapping twists and wrenches between the contact and object frames, while the latter maps joint velocities to twists in the contact frame. Since this thesis does not deal with gripper kinematics we will only be using the grasp matrix.

### 2.3.1 Grasp Matrix

The grasp matrix, also referred to as grasp map, has various definitions across the literature [RB19, Li94, LMSB14, SK16]. In general, it maps the contact forces onto the object wrench. We assume that the basis vector of the contact frame in  $+z$  direction equals the contact surface normal.

#### Partial grasp matrix

In order to denote the partial grasp matrix, we first define the hat operator, which is used to define a cross product as a matrix vector product:

$$N \times H = \begin{bmatrix} N_x \\ N_y \\ N_z \end{bmatrix} \times \begin{bmatrix} H_x \\ H_y \\ H_z \end{bmatrix} \Leftrightarrow \hat{N} \cdot H = \begin{bmatrix} 0 & -N_z & N_y \\ N_z & 0 & -N_x \\ -N_y & N_x & 0 \end{bmatrix} \cdot \begin{bmatrix} H_x \\ H_y \\ H_z \end{bmatrix}. \quad (2.13)$$

The partial grasp matrix, also known as the wrench transformation matrix, for contact point  $i$  is defined as:

$$\tilde{G}_i = \begin{bmatrix} {}^O R_{c_i} & 0 \\ \hat{p}_{O,c_i} {}^O R_{c_i} & {}^O R_{c_i} \end{bmatrix} \in \mathbb{R}^{6 \times 6}, \quad (2.14)$$

where  $p_{O,c_i}$  and  ${}^O R_{c_i}$  define the translation and rotation between the object and contact frame. The origin of the object frame, denoted as  $O$ , is typically the center

of mass, whereas the origin of the contact frame  $c_i$  is the contact point itself. The complete grasp matrix is the combination of the partial grasp matrices for each of the  $n_c$  contact points [LMSB14]:

$$\tilde{G} = \begin{bmatrix} \tilde{G}_1 & \dots & \tilde{G}_{n_c} \end{bmatrix} \in \mathbb{R}^{6 \times 6n_c}. \quad (2.15)$$

### Derivation

The grasp matrix is responsible for transforming the contact forces from the contact to the object frame. The partial grasp matrix, in particular, transforms the wrench  $\mathcal{F}_{c_i}$  from the contact frame (denoted as "con") to the object's center of mass frame (denoted as "obj" for now). Additionally, the force  $f_c \in \mathbb{R}^3$  in the following equations refers to the force at a contact  $c$ . When looking at one contact we get the equation:

$${}^{\text{obj}}\mathcal{F}_c = \begin{bmatrix} {}^{\text{obj}}f_c \\ {}^{\text{obj}}\tau_c \end{bmatrix} = \tilde{G} \cdot \begin{bmatrix} {}^{\text{con}}f_c \\ {}^{\text{con}}\tau_c \end{bmatrix} = \tilde{G} \cdot {}^{\text{con}}\mathcal{F}_c. \quad (2.16)$$

The force on the object is solely dependent on the contact force. This can be computed by projecting the contact force onto the basis vectors of the object frame using a rotation matrix  ${}^{\text{obj}}R_{\text{con}}$ :

$${}^{\text{obj}}f_c = \begin{bmatrix} {}^{\text{obj}}e_x & {}^{\text{obj}}e_y & {}^{\text{obj}}e_z \end{bmatrix} \cdot {}^{\text{con}}f_c = {}^{\text{obj}}R_{\text{con}} \cdot {}^{\text{con}}f_c. \quad (2.17)$$

The torque in the object frame depends on the transformed contact torque and contact force. It can be expressed using the standard formula for torque,  $\tau = r \times f$ , which involves the force  $f$  and distance vector  $r$ . In this case we denote  $r$  as  $p_{\text{con,obj}}$  which refers to the distance between contact point and object's center of mass. The torque in obj frame can be written as the sum of transformed contact torque and cross product of  $r$  with the transformed contact forces:

$$\begin{aligned} {}^{\text{obj}}\tau_c &= \begin{bmatrix} {}^{\text{obj}}e_x & {}^{\text{obj}}e_y & {}^{\text{obj}}e_z \end{bmatrix} \cdot {}^{\text{con}}\tau_c + p_{\text{con,obj}} \times {}^{\text{obj}}R_{\text{con}} \cdot {}^{\text{con}}f_c \\ &= {}^{\text{obj}}R_{\text{con}} \cdot {}^{\text{con}}\tau_c + p_{\text{con,obj}} \times {}^{\text{obj}}R_{\text{con}} \cdot {}^{\text{con}}f_c. \end{aligned} \quad (2.18)$$

### Selection matrix

The selection matrix  $B_i \in \mathbb{R}^{6 \times l_i}$  represents the wrench basis or constraint matrix for contact  $i$ . It acts as a filter to transmit specific wrench components across the contact, and its dimensions are determined by the contact model, with  $l_i$  being the number of transmissible wrench components. The applied wrench on contact  $i$  can be obtained by multiplying the corresponding selection matrix with the contact force, which also depends on the contact model [Li94].

$$\mathcal{F}_{c_i} = B_i f_{c_i}, \quad B_i \in \mathbb{R}^{6 \times l_i}, \quad f_{c_i} \in \mathbb{R}^{l_i} \quad (2.19)$$



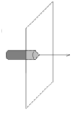
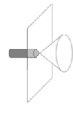
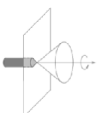
Contact type	Picture	Wrench basis	FC
Frictionless point contact		$\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$f_z \geq 0$
Point contact with friction		$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\sqrt{f_x^2 + f_y^2} \leq \mu f_z$ $f_z \geq 0$
Soft-finger		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\sqrt{f_x^2 + f_y^2} \leq \mu f_z$ $f_z \geq 0$ $ \tau_z  \leq \gamma f_z$

Figure 2.3: Selection matrices for the common contact types, based on [SK16].

The selection matrix for all  $n_c$  contacts can be written as:

$$B = \text{Blockdiag}(B_1, \dots, B_{n_c}) = \begin{bmatrix} B_1 & & \\ & \ddots & \\ & & B_{n_c} \end{bmatrix} \in \mathbb{R}^{6n_c \times l}, \quad (2.20)$$

where  $l$  is the total number of wrench and twist components transmitted, defined as:

$$l = \sum_{i=1}^{n_c} l_i. \quad (2.21)$$

We can now denote the grasp matrix as the product between the complete grasp matrix and the selection matrices:

$$G = \tilde{G}B = [\tilde{G}_1 B_1 \dots \tilde{G}_{n_c} B_{n_c}], \quad G \in \mathbb{R}^{6 \times l}. \quad (2.22)$$

### Grasping equations

The grasp matrix maps the exerted force on the contacts from the corresponding contact frames to the object wrench in the object frame. The set of wrenches the contacts can apply to the object can be written as:

$$\mathcal{F}_O = Gf_c, \quad f_c \in FC, \quad (2.23)$$

with  $f_c$  being the overall contact force on the object:

$$f_c = \begin{bmatrix} f_{c_1} \\ \vdots \\ f_{c_{n_c}} \end{bmatrix}, f_c \in \mathbb{R}^l. \quad (2.24)$$

The transposed grasp matrix  $G^T$  maps the object twist onto the  $n_c$  twists in the contact frames:

$$V_{c,\text{obj}} = G^T \mathcal{V}_O. \quad (2.25)$$

The overall contact twist  $V_{c,\text{obj}}$  is defined as:

$$V_{c,\text{obj}} = \begin{bmatrix} \mathcal{V}_{c_1,\text{obj}} \\ \vdots \\ \mathcal{V}_{c_{n_c},\text{obj}} \end{bmatrix}, V_{c,\text{obj}} \in \mathbb{R}^{6n_c}. \quad (2.26)$$

### 2.3.2 Force Closure

A force closure grasp grants the ability to withstand external object wrenches by applying appropriate contact wrenches with the fingers. Such grasps are necessary when lifting an object, as the hand needs to exert forces against gravity, and sometimes wrenches in other directions must also be resisted. During this process, it is important to ensure that the finger forces remain within the friction cone.

#### Definition

A grasp is in force closure if there exist contact forces  $f_c \in FC$  for any external wrench  $\mathcal{F}_{\text{ext}} \in \mathbb{R}^6$  on the object such that following conditions are fulfilled as mentioned in [LL04, SK16]:

$$1) \text{ rank}(G) = 6, \quad (2.27)$$

$$2) \exists f_c : f_c \in N(G) \text{ and } f_c \in \text{int}(FC), \quad (2.28)$$

with  $N(G)$  standing for nullspace and  $\text{int}(FC)$  referring to the interior points of the friction cone. The second criteria states that there must be a force inside the friction cone that also remains in the nullspace of  $G$ .

Note that the following equivalence is valid:

$$\forall \mathcal{F}_{\text{ext}} \in \mathbb{R}^6 \exists f_c \in \mathbb{R}^l : Gf_c = -\mathcal{F}_{\text{ext}} \Leftrightarrow \text{rank}(G) = 6. \quad (2.29)$$

#### Geometric metrics for force closure

There are also simple geometric metrics to check for force closure in two- or three-fingered grasps for specific contact models.

Two SFC contacts are in force closure when the vector connecting the points lies

inside of both friction cones [ZWH05]. Note that for two-fingered grasps, SFCs are required for force closure as there would be no way to resist torques about the axis connecting both points.

Grasps with three PCWF contacts are in force closure if the friction cones at each contact point intersect the plane generated by the three points in a planar cone [LP17].

### 2.3.3 Equilibrium Grasps

#### Equilibrium

A grasp is in equilibrium when the sum of all applied wrenches of the fingers in a common reference frame sum up to zero:

$$\sum_{i=1}^{n_c} \mathcal{F}_i = 0. \quad (2.30)$$

#### Unit Frictionless Equilibrium (UFE)

The grasp is in unit frictionless equilibrium when it is in equilibrium and the contacts apply forces in the object's surface normal direction, hence making the contact wrenches proportional to the wrench:

$$\tilde{\mathcal{F}}_i = \begin{bmatrix} n_i \\ r_i \times n_i \end{bmatrix} [\text{PFG10}]. \quad (2.31)$$

#### Nonmarginal Equilibrium

A grasp is in nonmarginal equilibrium when it is in equilibrium and their contact forces are strictly within their corresponding friction cones. If the contacts of a grasp can apply tangential and torsional frictional forces, then nonmarginal equilibrium is a sufficient condition for force closure. Additionally, contact forces in unit frictionless equilibrium grasps remain inside their friction cones, making them suitable for force closure in soft finger contact models [PFG10].

## 2.4 Grasp Quality Metrics

The following section will introduce quality metrics that are necessary to evaluate contact point location and gripper configuration. These metrics are cited from [RS15].

### 2.4.1 Grasp Position Quality Metrics

#### Metrics based on properties of grasp matrix $G$

The grasp matrix  $G \in \mathbb{R}^{6 \times l}$  has 6 singular values. When one of the singular values goes to zero the grasp is in a singular configuration. This means that the grasp is

incapable of resisting external wrenches in at least one direction.

**Minimum singular value  $Q_{MSV}$**

This quality metric is able to indicate how close the current grasp configuration is from falling into a singular one:

$$Q_{MSV} = \sigma_{\min}(G). \quad (2.32)$$

A large  $Q_{MSV}$  results in higher contributions from contact forces to the object wrench, which leads to a better grasp. However, this metric does not consider all singular values and is not invariant to changes in its reference frame.

**Volume of ellipsoid in the wrench space  $Q_{VEW}$**

This metric considers all singular values equally and is invariant to changes in the basis frame. However, it does not directly show the difference between the individual contributions of the fingers:

$$Q_{VEW} = \sigma_1 \sigma_2 \dots \sigma_6. \quad (2.33)$$

$Q_{VEW}$  needs to be maximized to find an optimal grasp.

**Grasp isotropy index  $Q_{GII}$**

The grasp isotropy index checks the contributions of the contact forces to the grasp. In an optimal grasp all fingers have a uniform contribution and the index reaches 1:

$$Q_{GII} = \frac{\sigma_{\min}(G)}{\sigma_{\max}(G)}. \quad (2.34)$$

Since these metrics have different value ranges we can introduce relative quality metrics. In this case, we divide the current metric value by optimal value that is possible for that configuration.

**Metrics based on grasp geometry**

**Distance between centroid of contact polygon and object's com**

When this metric is minimized external wrenches have a smaller influence on the grasp.



## Chapter 3

# Contact Position Optimization

During contact point optimization we try to ascend different objective functions such that our final grasp fulfills some desired metrics. In this chapter, we will first present objective functions which are needed to reach UFE. Finally, algorithms for a two-finger gripper based on [PFG10] will be proposed that are able to reach force closure granting grasp configuration on a cube.

### 3.1 Objective Functions

The goal of the following objective functions is to displace the contacts into a force closure grasp by trying to reach unit frictionless equilibrium. Reaching this goal accomplishes force closure as it is a type of nonmarginal equilibrium which is mentioned in Section 2.3.3. Unit frictionless equilibrium is achieved by attaining unit frictionless moment and force equilibrium which we will be referring to as UFME and UFFE.

The objective function for the UFFE is an error function called unit frictionless force residual which is defined as:

$$\varepsilon_f = 0.5 \mathbf{f}^T \mathbf{f}, \quad (3.1)$$

where  $\mathbf{f}$  is the sum of all unit contact forces

$$\mathbf{f} = \sum_{i=1}^{n_c} \mathbf{n}_i. \quad (3.2)$$

The error function  $\varepsilon_\tau$ , also known as unit frictionless moment residual, acts as the objective for UFME and can be written as:

$$\varepsilon_\tau = 0.5 \boldsymbol{\tau}^T \boldsymbol{\tau}, \quad (3.3)$$

where torque  $\tau$  is defined as the sum of all contact torques

$$\boldsymbol{\tau} = \sum_{i=1}^{n_c} \mathbf{r}_i \times \mathbf{n}_i, \quad (3.4)$$

with  $r_i$  being the 3D Cartesian position of the  $i$ -th contact in the contact centroid reference frame.

## 3.2 Force Residual Control

During force residual control we will be descending the unit friction less force residual. However, before we can do so we need to parametrize each contact  $r_i(u, v)$  by orthogonal parameter curves  $u$  and  $v$  such that the gradients along these parameters form a right-hand orthonormal coordinate frame with the surface normal at each contact point, as required by [PFG10]. A right-hand coordinate frame of the form  $(\nabla_u r_i, \nabla_v r_i, n_i)$  must satisfy following conditions:

$$\begin{aligned}\nabla_u r_i \times \nabla_v r_i &= n_i, \\ \nabla_v r_i \times n_i &= \nabla_u r_i, \\ n_i \times \nabla_u r_i &= \nabla_v r_i,\end{aligned}\tag{3.5}$$

$$\|\nabla_u r_i\|_2 = \|\nabla_v r_i\|_2 = \|n_i\|_2 = 1.\tag{3.6}$$

### 3.2.1 Surface Parametrization

To find an appropriate parametrization of a contact point that fulfills the previously mentioned conditions, we will be using a form of sphere coordinates for unit spheres which are usually parameterized by the angles  $\varphi$  and  $\theta$ . One initial candidate for this parametrization would be classical spherical coordinates :

$$r(\varphi, \theta) = \begin{bmatrix} \sin(\theta) \cos(\varphi) \\ \sin(\theta) \sin(\varphi) \\ \cos(\theta) \end{bmatrix}, \quad \varphi \in [0, 2\pi], \quad \theta \in [0, \pi],\tag{3.7}$$

which have the normalized gradients:

$$\begin{aligned}\frac{1}{\|\nabla_\varphi r\|_2} \nabla_\varphi r &= \frac{1}{\sin(\theta)} \cdot \begin{bmatrix} -\sin(\theta) \sin(\varphi) \\ \sin(\theta) \cos(\varphi) \\ 0 \end{bmatrix} = \begin{bmatrix} -\sin(\varphi) \\ \cos(\varphi) \\ 0 \end{bmatrix}, \\ \frac{1}{\|\nabla_\theta r\|_2} \nabla_\theta r &= \nabla_\theta r = \begin{bmatrix} \cos(\theta) \cos(\varphi) \\ \cos(\theta) \sin(\varphi) \\ -\sin(\theta) \end{bmatrix}.\end{aligned}\tag{3.8}$$

This parametrization fulfills the conditions from (3.5) only for  $u = \theta$  and  $v = \varphi$ . However, when applying moment residual control on a cube for certain configurations, the parameter selection required in (3.20) cannot be fulfilled using this selection. To address this issue, an alternate sphere coordinate description is used instead, which satisfies the conditions for  $u = \varphi$  and  $v = \theta$ :

$$r(\varphi, \theta) = \begin{bmatrix} -\sin(\theta) \cos(\varphi) \\ \sin(\theta) \sin(\varphi) \\ \cos(\theta) \end{bmatrix}, \quad \varphi \in [0, 2\pi], \quad \theta \in [0, \pi], \quad (3.9)$$

$$\begin{aligned} \frac{1}{\|\nabla_{\varphi} r\|_2} \nabla_{\varphi} r &= \begin{bmatrix} \sin(\varphi) \\ \cos(\varphi) \\ 0 \end{bmatrix}, \\ \nabla_{\theta} r &= \begin{bmatrix} -\cos(\theta) \cos(\varphi) \\ \cos(\theta) \sin(\varphi) \\ -\sin(\theta) \end{bmatrix}. \end{aligned} \quad (3.10)$$

For a given point  $r$  on the unit sphere its parameters can be calculated as:

$$\begin{aligned} \varphi &= \text{mod}(\arctan 2(r_y, -r_x), 2\pi), \\ \theta &= \arccos(r_z). \end{aligned} \quad (3.11)$$

At first glance this parametrization appears to work fine, however, for cases such as  $n = [0 \ 0 \ \pm 1]^T$  there are no definite solutions for the parameters since  $\theta$  needs to be either 0 or  $\pi$  and  $\varphi$  can be arbitrarily selected. This problem will be covered in more detail in the algorithms for force and moment residual control.

### 3.2.2 Force Residual Control Algorithm

#### Definitions

The surface parameters of the  $i$ -th contact can be summarized in the variable  $\tilde{\mathbf{u}}_i$ :

$$\tilde{\mathbf{u}}_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix}. \quad (3.12)$$

Now we can define the generalized vector for all surface parameters:

$$\mathbf{u} = \begin{bmatrix} \tilde{\mathbf{u}}_1 \\ \vdots \\ \tilde{\mathbf{u}}_{n_c} \end{bmatrix}. \quad (3.13)$$

#### Derivation

In order to apply gradient descent on the unit frictionless force residual, we first need to calculate its gradient with respect to  $\mathbf{u}$ . This can be calculated using the product rule and expressed as

$$\nabla_{\mathbf{u}} \varepsilon_f = J_f^T \mathbf{f}, \quad (3.14)$$

with the unit frictionless force residual Jacobian being defined as

$$J_f = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial \tilde{\mathbf{u}}_1} & \cdots & \frac{\partial \mathbf{f}}{\partial \tilde{\mathbf{u}}_{n_c}} \end{bmatrix} \in \mathbb{R}^{3 \times 2n_c}. \quad (3.15)$$



We can write the  $i$ -th partial derivative as follows:

$$\frac{\partial \mathbf{f}}{\partial \tilde{\mathbf{u}}_i} = \frac{\partial n_i}{\partial \tilde{\mathbf{u}}_i} = [\nabla_u r \quad \nabla_v r] K_i, \quad (3.16)$$

where  $K_i$  is a  $2 \times 2$  surface curvature matrix. Now the unit frictionless force residual can be denoted as

$$J_f = [\nabla_u r \quad \nabla_v r_1 \quad \dots \quad \nabla_u r_{n_c} \quad \nabla_v r_{n_c}] K = \hat{J}_f K. \quad (3.17)$$

In this case  $\hat{J}_f$  is a matrix of surface tangents at all contact points and  $K$  is the combination of all contact curvature matrices in a block diagonal matrix.

$$K = \begin{bmatrix} K_1 & & \\ & \ddots & \\ & & K_{n_c} \end{bmatrix} \in \mathbb{R}^{2n_c \times 2n_c}. \quad (3.18)$$

For a unit sphere, all contact surface curvature matrices are the identity matrix, including  $K$ , which leads to  $J_f = \hat{J}_f$ . For further information regarding curvature matrices, also known as Weingarten maps, see [Pre10].

Now we can denote the gradient descent algorithm used for force residual control, which follows the negative gradient of the unit frictionless force residual with step size  $\gamma_f$  while assuming unit curvature:

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \gamma_f \nabla \varepsilon_f(\mathbf{u}_k) = \mathbf{u}_k - \gamma_f \hat{J}_f^T \mathbf{f}(\mathbf{u}_k). \quad (3.19)$$

The convergence of the force residual controller is shown in [PFG10].

#### Algorithm for two-finger gripper

In this section, we will present an algorithm for force residual control with a parallel jaw on a cube.

According to the authors of [PFG10] the parameter  $v$  will be selected such that the surface tangents in its direction are equal:

$$\nabla_v r_1 = \nabla_v r_2. \quad (3.20)$$

Now we can denote the gradient as :

$$\nabla_{\mathbf{u}} \varepsilon = \begin{bmatrix} \nabla_u r_1^T \\ \nabla_v r_1^T \\ \nabla_u r_2^T \\ \nabla_v r_2^T \end{bmatrix} \cdot \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} = \begin{bmatrix} \nabla_u r_1^T \cdot n_2 \\ 0 \\ \nabla_u r_2^T \cdot n_1 \\ 0 \end{bmatrix}. \quad (3.21)$$

In addition, [PFG10] has proven that orthonormal frames as mentioned in (3.5) and (3.6), which satisfy the condition (3.22), have the following relationship that can be used to simplify the previous expression by using:

$$\alpha = \nabla_u r_1^T \cdot n_2 = -\nabla_u r_2^T \cdot n_1. \quad (3.22)$$

The origin of the unit sphere coordinates for each contact  $r_i$  is selected such that the contact position equals the contact normal in the new frame in this algorithm. The Algorithm 1 can be summarized as follows:

Firstly, we need to check if the surface normal is in the z-direction, as this would make the parameter  $\varphi$  ambiguous. If this is the case, the ambiguous parameter must be selected in a way that fulfills (3.20). If it is not possible to do so, then the gradients must be calculated, and the parameters  $u$  and  $v$  should be selected accordingly. Once we have the values of  $u$  and  $v$ , we perform an iteration of gradient descent, obtain new points on the unit sphere, and project them onto the closest surface of the object. These steps are repeated until the unit frictionless force residual is below a certain tolerance, which we refer to as "*TOL*", or the maximum number of iterations is reached.

### 3.3 Moment Residual Control

#### Derivation

For moment residual control we first need to calculate the gradient of the unit frictionless moment residual with respect to  $u_i$ :

$$\nabla_{\mathbf{u}} \varepsilon_\tau = J_\tau^T \boldsymbol{\tau}, \quad (3.23)$$

with the unit frictionless moment residual Jacobian defined as:

$$J_\tau = \begin{bmatrix} \frac{\partial \boldsymbol{\tau}}{\partial \tilde{\mathbf{u}}_1} & \cdots & \frac{\partial \boldsymbol{\tau}}{\partial \tilde{\mathbf{u}}_{n_c}} \end{bmatrix} \in \mathbb{R}^{3 \times 2n_c}. \quad (3.24)$$

We can write the  $i$ -th partial derivative as

$$\frac{\partial \boldsymbol{\tau}}{\partial \tilde{\mathbf{u}}_i} = \nabla_{\tilde{\mathbf{u}}_i} r_i \times n_i + r_i \times \frac{\partial n_i}{\partial \mathbf{u}_i}, \quad (3.25)$$

where the second term is set to zero since zero surface curvature is assumed to simplify the calculations. The simplified partial derivative can therefore be denoted as:

$$\frac{\partial \boldsymbol{\tau}}{\partial \mathbf{u}_i} = \nabla_{\mathbf{u}_i} r_i \times n_i = \begin{bmatrix} \nabla_u r_i \times n_i & \nabla_v r_i \times n_i \end{bmatrix} = \begin{bmatrix} -\nabla_v r_i & \nabla_u r_i \end{bmatrix}. \quad (3.26)$$

Due to surface curvature approximations, the approximated Jacobian can be written as:

$$\hat{J}_\tau = \begin{bmatrix} -\nabla_v r_1 & \nabla_u r_1 & \cdots & -\nabla_v r_{n_c} & \nabla_u r_{n_c} \end{bmatrix}. \quad (3.27)$$

With the definitions provided earlier, we can now represent a gradient descent iteration as follows:

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \gamma_\tau \nabla \varepsilon_\tau(\mathbf{u}_k) = \mathbf{u}_k - \gamma_\tau \hat{J}_\tau^T \boldsymbol{\tau}(\mathbf{u}_k). \quad (3.28)$$

The convergence of the moment residual controller is proven in [PFG10].

**Algorithm for two finger gripper**

This section presents an algorithm for force residual control using a parallel jaw gripper on a cube.

For the torque calculations, the positions of the contact points  $r_1$  and  $r_2$  are defined in a frame with their centroid  $c$  as the origin. By doing so we get the simplification

$$\zeta_{r_1} = -\zeta_{r_2}, \quad (3.29)$$

which we can also use to describe the unit frictionless moment residual gradient:

$$\begin{aligned} \nabla_u \varepsilon_\tau &= \begin{bmatrix} -\nabla_v r_1^T \\ \nabla_u r_1^T \\ -\nabla_v r_2^T \\ \nabla_u r_2^T \end{bmatrix} \cdot [\zeta_{r_1} \times n_1 + \zeta_{r_2} \times n_2] = \begin{bmatrix} -\nabla_v r_1^T \\ \nabla_u r_1^T \\ -\nabla_v r_2^T \\ \nabla_u r_2^T \end{bmatrix} \cdot [\zeta_{r_1} \times n_1 - \zeta_{r_1} \times n_2] \\ &= \begin{bmatrix} \zeta_{r_1}^T (\nabla_u r_1 - \nabla_u r_2) \\ \zeta_{r_1}^T (\nabla_v r_1 - n_2 \times \nabla_u r_1) \\ \zeta_{r_1}^T (\nabla_u r_1 - \nabla_u r_2) \\ -\zeta_{r_1}^T (\nabla_v r_1 - n_1 \times \nabla_u r_2) \end{bmatrix}. \end{aligned} \quad (3.30)$$

For further simplifications see [PFG10].

Similar to the force residual control we will be selecting the parameters  $u$  and  $v$  such that (3.22) is fulfilled. Additionally, origins of the unit sphere are identical to the ones for force residual control. Before we can begin the algorithm, we need to define the rotation matrix around an axis  $k$  as:

$$\text{Rot}(\theta, k) = \begin{bmatrix} \cos(\theta) + k_x^2 a & k_x k_y a - k_z \sin(\theta) & k_x k_z a + k_y \sin(\theta) \\ k_x k_y a + k_z \sin(\theta) & \cos(\theta) + k_y^2 a & k_y k_z a - k_x \sin(\theta) \\ k_x k_z a - k_y \sin(\theta) & k_y k_z a + k_x \sin(\theta) & \cos(\theta) + k_z^2 a \end{bmatrix}, \quad (3.31)$$

with  $a = 1 - \cos \theta$ .

The moment residual control Algorithm 2 will now be summarized:

We begin the algorithm for moment residual control in a similar fashion as the force residual control algorithm. Firstly, we check if any surface normals are in the  $z$ -direction. If so, we rotate them by 270 degrees around the  $y$ -axis or the surface normal of the other contact point, depending on whether it's one or both surface normals. The rotation angle of 270 degrees was chosen as it has shown to work well when applying moment residual control on a cube. Once an iteration of gradient descent is performed, the solutions are rotated back to their original frame.

Similar to the force residual control algorithm, we aim to fulfill the requirement for  $v$  in (3.22) by checking which gradients are equal and projecting newly calculated points onto the closest object surface. These steps are repeated until the unit frictionless moment residual is below a certain tolerance or the maximum number of iterations is reached.

### 3.4 Switching Grasp Control

After describing both the force and moment residual controllers, we can now combine them in the switching grasp controller. This controller executes the force residual controller while the force residual is greater than a tolerance  $\beta$ , and the moment residual controller is executed otherwise. The controller can be denoted as:

$$\dot{u}_a = \nabla_{\mathbf{u}} \varepsilon_f + N_a \nabla_{\mathbf{u}} \varepsilon_\tau, \quad (3.32)$$

with the indicator variable  $N_a$  defined as:

$$N_a = \begin{cases} 1, & \|\mathbf{f}\| \leq \beta \\ 0 & \text{otherwise} \end{cases}. \quad (3.33)$$

The switching grasp controller can be implemented by running the moment residual control Algorithm 2 after the moment residual control Algorithm 1. In addition to this, the authors of [PFG10] introduce the Null-Space and Approximate Null-Space Grasp Controllers for contact point optimization. These controllers also combine force and moment residual controllers.

### 3.5 Implementation

In order to implement the contact point position optimization algorithms on a cube, it is necessary to determine the surface normal at a given contact. The authors of [PFG10] use a touch-based approach, which involves measuring surface normals, contact forces, and torques using six-axis load cells at the gripper's fingertips. However, in our case, we assume that we have perfect knowledge of the object's geometry, allowing us to determine the current surface normal from the contact point location. With this geometric knowledge, we can also project points on a unit sphere onto the closest object surface. It should be noted that a touch-based method would require a different approach. While the algorithm presented in this chapter works for contact point optimization on a cube with a two-fingered gripper, it has limitations when used with different grippers and objects. These issues are further discussed in Chapter 6.

Another issue that may arise when using force residual control is the possibility of the grasp failing to reach unit frictionless force equilibrium when both contacts jump over to the next surface simultaneously. This scenario is depicted in Figure 3.1, where the contact points switch surfaces simultaneously, leading to no change in the force residual.

In general, the algorithm will be stuck alternating between these two configurations. This issue can be resolved by reducing the step size  $\gamma_f$ . In that case, such configuration would converge to points in the edge region. However, it can be difficult to clearly define the surface normals in the edge region. In our case, we strictly set the

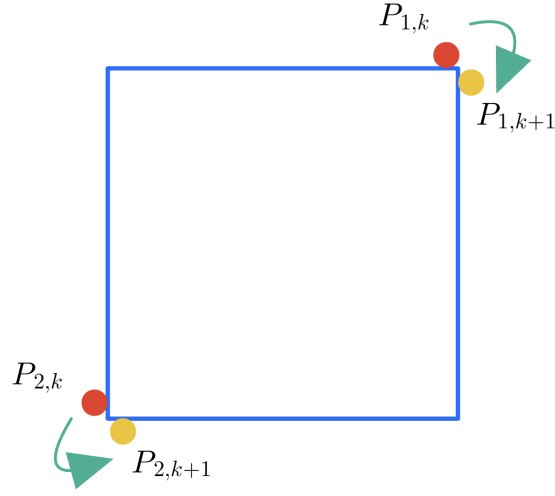


Figure 3.1: Example of alternating configuration when grasping a cube.

surface normal of contact points near the edges to the normal of their corresponding surface. To address the issue of finger placement at the edge region, an ideal solution would be to detect the distances of the fingers to the next surface edge. If the distances are equal, then only one finger should be moved with gradient descent for one iteration while the other remains in its current position.

A similar alternating configuration can also occur during moment residual control on more complex geometries. The issue can be resolved in similar fashion to the previously mentioned solution for force residual control.

---

**Algorithm 1** Force Residual control , strict version

---

```

1: Initialization ( $k = 1$ ):
2: Determine surface normals  $n_{1,1}$  and  $n_{2,1}$  of initial contact points  $r_{1,1}$  and  $r_{2,1}$ 
3: Calculate  $\mathbf{f}_1 = n_{1,1} + n_{2,1}$  and  $\varepsilon_{f,1}$ 
4: while  $\varepsilon_{f,k} > TOL$  or  $k < \text{max. iterations}$  do
5:   if  $n_{i,k} = \pm [0 \ 0 \ 1]^T$  for  $i = 1$  or  $2$  then
6:     Set  $j$  to opposite index of  $i$ 
7:     Calculate  $\varphi_j$ ,  $\theta_j$ ,  $\nabla_{\varphi} r_{j,k}$  and  $\nabla_{\theta} r_{j,k}$ 
8:     if  $\nabla_{\varphi} r_{j,k}^T \mathbf{f} == 0$  then
9:       Set  $v = \varphi$  and  $u = \theta$ 
10:      Set  $\varphi_{i,k} = \varphi_{j,k}$ 
11:      Calculate  $\theta_i$ 
12:     else if  $\nabla_{\theta} r_{j,k}^T \mathbf{f} == 0$  then
13:       Set  $v = \theta$  and  $u = \varphi$ 
14:       Calculate  $\theta_i$ 
15:       Calculate  $\varphi_i$  such that  $\nabla_{\theta} r_{i,k} = \nabla_{\theta} r_{j,k}$ 
16:     end if
17:   else
18:     Calculate  $\varphi_{1,k}$ ,  $\theta_{1,k}$ ,  $\varphi_{2,k}$  and  $\theta_{2,k}$  with (3.11)
19:     Calculate  $\nabla_{\varphi} r_{1,k}$ ,  $\nabla_{\theta} r_{1,k}$ ,  $\nabla_{\varphi} r_{2,k}$  and  $\nabla_{\theta} r_{2,k}$  with (3.10)
20:     if  $\nabla_{\varphi} r_{1,k} = \nabla_{\varphi} r_{2,k}$  then
21:       Set  $v = \varphi$  and  $u = \theta$ 
22:     else if  $\nabla_{\theta} r_{1,k} = \nabla_{\theta} r_{2,k}$  then
23:       Set  $v = \theta$  and  $u = \varphi$ 
24:     end if
25:   end if
26:   Calculate  $\hat{J}_f$  with (3.17) and  $\nabla_{\mathbf{u}} \varepsilon_f$  with (3.21)
27:   Calculate  $\mathbf{u}_{k+1}$  with 3.19
28:   Calculate  $r_{1,k+1}$  and  $r_{2,k+1}$  with (3.9)
29:   Transform  $r_{1,k+1}$  and  $r_{2,k+1}$  to global coordinates
30:   Project point on sphere onto closest object surface
31:   Set  $r_{1,k+1}$  and  $r_{2,k+1}$  to projected points
32:   Determine surface normals  $n_{1,k+1}$  and  $n_{2,k+1}$  of updated contact points
33:   Calculate  $\mathbf{f}_{k+1} = n_{1,k+1} + n_{2,k+1}$  and  $\varepsilon_{f,k+1}$ 
34:    $k = k + 1$ 
35: end while

```

---

**Algorithm 2** Moment Residual Control , strict version

---

```

1: Initialization ( $k = 1$ ):
2: Determine surface normals  $n_{1,1}$  and  $n_{2,1}$  of initial contact points  $r_{1,1}$  and  $r_{2,1}$ 
3: Calculate  $r_{1,1}$  in centroid frame with centroid  $c = 0.5 \cdot (r_{1,1} + r_{2,1})$  for  $l = i, j$ 
4: Calculate  $\tau_1 = r_{1,1} \times n_{1,1} - r_{1,1} \times n_{2,1}$  and  $\varepsilon_{\tau,1}$ 
5: while  $\varepsilon_{\tau,k} > TOL$  or  $k < \text{max. iterations}$  do
6:   if  $n_{i,k} = \pm [0 \ 0 \ 1]^T$  for  $i = 1$  or  $2$  then
7:     Set  $j$  to opposite index of  $i$ 
8:     if  $n_{i,k} = \pm [0 \ 0 \ 1]^T$  for  $i = 1$  and  $2$  then
9:       Define rotation matrix  $R = Rot(270, e_y)$  with (3.31)
10:    else
11:      Define rotation matrix  $R = Rot(270, n_j)$  with (3.31)
12:    end if
13:    Calculate transformed surface normals  $\hat{n}_{l,k} = R \cdot n_{l,k}$  for  $l = i, j$ 
14:    Calculate the angles  $\tilde{\varphi}_{i,k}$ ,  $\tilde{\theta}_{i,k}$ ,  $\tilde{\varphi}_{j,k}$  and  $\tilde{\theta}_{j,k}$  with (3.11)
15:    Calculate gradients  $\nabla_{\varphi} \tilde{r}_{i,k}$ ,  $\nabla_{\theta} \tilde{r}_{i,k}$ ,  $\nabla_{\varphi} \tilde{r}_{j,k}$  and  $\nabla_{\theta} \tilde{r}_{j,k}$  with (3.10)
16:    if  $\nabla_{\varphi} \tilde{r}_{i,k} = \nabla_{\varphi} \tilde{r}_{j,k}$  then
17:      Set  $v = \varphi$  and  $u = \theta$ 
18:    else if  $\nabla_{\theta} \tilde{r}_{i,k} = \nabla_{\theta} \tilde{r}_{j,k}$  then
19:      Set  $v = \theta$  and  $u = \varphi$ 
20:    end if
21:    Calculate  $\hat{J}_{\tau}$  with (3.27) and  $\nabla_{\mathbf{u}} \varepsilon_{\tau}$  with (3.23)
22:    Calculate  $\mathbf{u}_{k+1}$  with 3.28
23:    Calculate new points  $\tilde{r}_{i,k+1}$  and  $\tilde{r}_{j,k+1}$  with (3.9)
24:    Transform  $\tilde{r}_{l,k+1}$  back to  $r_{l,k+1}$  with  $r_{l,k+1} = R^T \cdot \tilde{r}_{l,k+1}$  for  $l = i, j$ 
25:  else
26:    Calculate  $\varphi_{1,k}$ ,  $\theta_{1,k}$ ,  $\varphi_{2,k}$  and  $\theta_{2,k}$  with (3.11)
27:    Calculate  $\nabla_{\varphi} r_{1,k}$ ,  $\nabla_{\theta} r_{1,k}$ ,  $\nabla_{\varphi} r_{2,k}$  and  $\nabla_{\theta} r_{2,k}$  with (3.10)
28:    if  $\nabla_{\varphi} r_{1,k} = \nabla_{\varphi} r_{2,k}$  then
29:      Set  $v = \varphi$  and  $u = \theta$ 
30:    else if  $\nabla_{\theta} r_{1,k} = \nabla_{\theta} r_{2,k}$  then
31:      Set  $v = \theta$  and  $u = \varphi$ 
32:    end if
33:    Calculate  $\hat{J}_{\tau}$  with (3.27) and  $\nabla_{\mathbf{u}} \varepsilon_{\tau}$  with (3.23)
34:    Calculate  $\mathbf{u}_{k+1}$  with 3.28
35:    Calculate new points  $r_{1,k+1}$  and  $r_{2,k+1}$  with (3.9)
36:  end if
37:  Transform  $r_{1,k+1}$  and  $r_{2,k+1}$  to global coordinates
38:  Project point on sphere onto closest object surface
39:  Set  $r_{1,k+1}$  and  $r_{2,k+1}$  to projected points
40:  Determine surface normals  $n_{1,k+1}$  and  $n_{2,k+1}$  of updated contact points
41:  Calculate  $r_{1,k+1}$  in centroid frame
42:  Calculate  $\tau_{k+1} = r_{1,k+1} \times n_{1,k+1} - r_{1,k+1} \times n_{2,k+1}$  and  $\varepsilon_{\tau,k+1}$ 
43:   $k = k + 1$ 
44: end while

```

---

# Chapter 4

## Force Optimization

Grasp force optimization contains different techniques ranging from standard Lagrangian with constraints to Linear Matrix Inequalities (LMI), which can be solved with different numerical methods. In this chapter, we will focus on the Lagrangian method which is based on [XWF04, TW02].

### 4.1 Lagrangian Method

In the Lagrange multiplier method, we attempt to minimize a cost function  $f(x) \in \mathbb{R}$  subject to equality constraint vector  $g(x) = 0$  and inequality constraint vector  $h(x) \leq 0$ , where  $q$  and  $r$  represent the number of constraints on the optimization variable  $x \in \mathcal{X}$  which remains in the convex set  $\mathcal{X} \subset \mathbb{R}^n$ . The optimization problem can be denoted as:

$$\begin{aligned} & \text{minimize } f(x) \\ & \text{subject to } g(x) = 0, \quad g \in \mathbb{R}^q \\ & \quad \quad \quad h(x) \leq 0, \quad h \in \mathbb{R}^r. \end{aligned} \tag{4.1}$$

This problem can be written as a Lagrangian function:

$$L(x, \lambda, \mu) = f(x) + \lambda^T g(x) + \mu^T h(x), \tag{4.2}$$

with Lagrange multipliers  $\lambda \in \mathbb{R}^q$  and  $\mu \in \mathbb{R}^r$ . The optimal solution is found by satisfying the Karush-Kuhn-Tucker (KKT) conditions which are explained in [NW06].

#### 4.1.1 Constraints

When minimizing the exerted force during a grasp, it is necessary to maintain stability. This is achieved through two sets of constraints. The first constraint is a variation of the first force closure requirement, which mandates that any external



wrenches must be resisted while keeping contact forces within their respective friction cones, as shown in (2.29). The second set of constraints are the friction cone inequalities for each contact, as described in Chapter 2.1.

To simplify the problem and avoid numerical complexities, we transform the non-linear constraint into a linear one.

This simplification is demonstrated for a PCWF, where we linearize with four cone edge vectors. The non-linear friction cone constraint is described in (2.3). Based on that we first define the edge vectors via (2.5) :

$$\begin{aligned}
 f_{1,c_i} &= \begin{bmatrix} \mu \cos(0) \\ \mu \sin(0) \\ 1 \end{bmatrix} = \begin{bmatrix} \mu \\ 0 \\ 1 \end{bmatrix}, \\
 f_{2,c_i} &= \begin{bmatrix} \mu \cos\left(\frac{\pi}{2}\right) \\ \mu \sin\left(\frac{\pi}{2}\right) \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \mu \\ 1 \end{bmatrix}, \\
 f_{3,c_i} &= \begin{bmatrix} \mu \cos(\pi) \\ \mu \sin(\pi) \\ 1 \end{bmatrix} = \begin{bmatrix} -\mu \\ 0 \\ 1 \end{bmatrix}, \\
 f_{4,c_i} &= \begin{bmatrix} \mu \cos\left(\frac{3}{2}\pi\right) \\ \mu \sin\left(\frac{3}{2}\pi\right) \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -\mu \\ 1 \end{bmatrix},
 \end{aligned} \tag{4.3}$$

and the matrix F via (2.8):

$$F_{c_i} = [f_{1,c_i} \quad f_{2,c_i} \quad f_{3,c_i} \quad f_{4,c_i}] = \begin{bmatrix} \mu & 0 & -\mu & 0 \\ 0 & \mu & 0 & -\mu \\ 1 & 1 & 1 & 1 \end{bmatrix}. \tag{4.4}$$

Now we get following linearized cone constraint:

$$f_{c_i} = F_{c_i} \cdot \alpha_{c_i} = \begin{bmatrix} \mu & 0 & -\mu & 0 \\ 0 & \mu & 0 & -\mu \\ 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix}, \tag{4.5}$$

and

$$\alpha_{c_i} \geq 0. \tag{4.6}$$

It can be interpreted as a linearized friction cone with four edges which can be visualized as a four-sided pyramid.

To generalize the linearized cone constraint, we define the generalized contact force and parameter vector:

$$f_c = \begin{bmatrix} f_{c_1} \\ \vdots \\ f_{c_{n_c}} \end{bmatrix} \in \mathbb{R}^l, \quad (4.7)$$

$$\alpha_c = \begin{bmatrix} \alpha_{c_1} \\ \vdots \\ \alpha_{c_{n_c}} \end{bmatrix} \in \mathbb{R}^{n_c \cdot s}. \quad (4.8)$$

The linearized friction cone constraint for  $n_c$  contacts can be written as :

$$\begin{aligned} f_c &= F \cdot \alpha_c, \\ \alpha_c &\geq 0, \end{aligned} \quad (4.9)$$

with matrix  $F$  defined as:

$$F = \begin{bmatrix} F_{c_1} & & \\ & \ddots & \\ & & F_{c_{n_c}} \end{bmatrix} \in \mathbb{R}^{3n_c \times sn_c}. \quad (4.10)$$

The friction cone edge vector matrices  $F_{c_i}$  are explained in (2.8) . Since each contact has the same number of linearized edge vectors, and each edge vector is described in a local contact frame with the x-axis pointing in the negative contact normal direction, we can simplify equation (4.10) by using the fact that all the force-closure edge vector matrices are equal.

$$\begin{aligned} F_{c_1} &= F_{c_2} = \dots = F_{c_{n_c}} = F_c, \\ F &= \begin{bmatrix} F_c & & \\ & \ddots & \\ & & F_c \end{bmatrix}. \end{aligned} \quad (4.11)$$

In general, a friction cone constraint can also be represented as a linear matrix inequality (LMI), where the positive definiteness of a matrix is used to denote the constraint. This approach becomes more computationally costly as the matrix dimensions increase with each additional finger. The approximation of the friction cone can be improved by adding additional cone edges.

When using soft finger contacts an additional inequality constraint for each torque component of  $f_{c_i}$  is added. The torque  $\tau_n$  in surface normal direction for contact  $c_i$  is now denoted as  $f_{\tau, c_i}$  as it is the only torque component of the applied contact force vector in the SFC. In this case, the x-axis is in contact normal direction which is why the additional constraint can be denoted as

$$|f_{\tau, c_i}| \leq \gamma f_{x, c_i}, \quad i = 1 \dots n. \quad (4.12)$$

Another constraint for force optimization is needed to assure resistance to external wrenches. Hence, we use the force closure constraint, which is formulated as the matrix equality defined in (2.29).

### 4.1.2 Objective

We choose a convex quadratic objective function such that we can solve this optimization problem with quadratic programming. The general form is:

$$f(x) = 0.5x^T Qx + q^T x, \quad (4.13)$$

with the optimization vector  $x$ , symmetric weight matrix  $Q$  and vector  $q$ . For the non linear case these variables can be defined as:

$$\begin{aligned} x = f_c &= \begin{bmatrix} f_{c_1} \\ \vdots \\ f_{c_{n_c}} \end{bmatrix} \in \mathbb{R}^l, \\ q &= \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^l, \\ \text{and } Q &= I \in \mathbb{R}^{l \times l}. \end{aligned} \quad (4.14)$$

And for the linearized case, these variables have a higher dimension since the optimization variable also contains the parameter vector for each contact. They are noted as:

$$\begin{aligned} x = \begin{bmatrix} f_c \\ \alpha_c \end{bmatrix} &= \begin{bmatrix} f_{c_1} \\ \vdots \\ f_{c_{n_c}} \\ \alpha_{c_{n_1}} \\ \vdots \\ \alpha_{c_{n_c}} \end{bmatrix} \in \mathbb{R}^{l+n_c \cdot s}, \\ q &= \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \in \mathbb{R}^{l+n_c \cdot s} \\ \text{and } Q &= \begin{bmatrix} I_{l \times l} & 0 \\ 0 & 0_{n_c s \times n_c s} \end{bmatrix} \in \mathbb{R}^{(l+n_c \cdot s) \times (l+n_c \cdot s)}, \end{aligned} \quad (4.15)$$

with  $q_1 = [1 \ \dots \ 1]^T \in \mathbb{R}^l$  and  $q_2 = [0 \ \dots \ 0]^T \in \mathbb{R}^{n_c \cdot s}$ .

In this case we only consider contact force components for the cost which is why all the parameter vector components have 0 weight in  $Q$ .

### 4.1.3 Solution

For the implementation, the surface contact normal is set to  $x_{i,1}$  which stands for  $f_{c_i,x}$ , the x component of the  $i$ -th contact force. Hence, the z and x components from the previously mentioned constraints are swapped. We now have the following non-linear optimization problem:

$$\begin{aligned} & \text{minimize } f(x) = 0.5x^T Qx + q^T x \\ & \text{subject to } Gx = -\mathcal{F}_{\text{ext}} \\ & \quad x_{i,2}^2 + x_{i,3}^2 \leq \mu^2 x_{i,1}^2 \\ & \quad x_{i,1} \geq 0 \quad \forall i = 1 \dots n_c. \end{aligned} \tag{4.16}$$

To describe the linearized contact force optimization problem, we need to define two matrices:  $T$  and  $W$ . These matrices are used to extract the contact forces and parameter vectors from the optimization variable  $x$ . This is necessary because these vectors are required for the constraints mentioned in equations (2.29) and (4.9):

$$T = \begin{bmatrix} I_{l \times l} & 0_{l \times n_c s} \end{bmatrix} \in \mathbb{R}^{l \times (l + n_c \cdot s)}, \tag{4.17}$$

$$W = \begin{bmatrix} 0_{n_c s \times l} & I_{n_c s \times n_c s} \end{bmatrix} \in \mathbb{R}^{(n_c \cdot s) \times (l + n_c \cdot s)}, \tag{4.18}$$

$$Tx = f_c, \tag{4.19}$$

$$Wx = \alpha_c. \tag{4.20}$$

#### Force optimization problem for grasp with PCWFs

The linearized optimization problem for PCWFs can now be denoted as follows:

$$\begin{aligned} & \text{minimize } f(x) = 0.5x^T Qx + q^T x \\ & \text{subject to } GTx = -\mathcal{F}_{\text{ext}} \\ & \quad Tx = F \cdot Wx \\ & \quad Wx \geq 0 \\ & \quad x_{i,1} \geq 0 \quad \forall i = 1 \dots n_c. \end{aligned} \tag{4.21}$$

#### Force optimization problem for grasp with SFCs

When using soft finger contact models, the additional constraint  $-\gamma x_{i,1} \leq x_{i,4} \leq \gamma x_{i,1} \quad \forall i = 1 \dots n_c$ , where  $x_{i,4}$  refers to the torque component of each contact force, is added to the problem. It can be rewritten as:

$$\begin{aligned} 0 & \leq \gamma x_{i,1} - x_{i,4} \\ 0 & \leq \gamma x_{i,1} + x_{i,4}. \end{aligned} \tag{4.22}$$

Since the optimization vector  $x$  is larger when using SFCs we need to filter out the force components at each contact in order to use it in the linearized friction cone

constraint. Therefore we define the matrices

$$N_{c_i} = \begin{cases} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} & \text{if } c_i \text{ is SFC,} \\ I_{3 \times 3} & \text{if } c_i \text{ PCWF} \end{cases} \quad (4.23)$$

$$N = \text{Blockdiag}(N_{c_1}, \dots, N_{c_{n_c}}) = \begin{bmatrix} N_{c_1} & & \\ & \ddots & \\ & & N_{c_{n_c}} \end{bmatrix} \in \mathbb{R}^{3n_c \times l}. \quad (4.24)$$

Now we can write the optimization problem with SFCs as:

$$\begin{aligned} & \text{minimize } f(x) = 0.5x^T Qx + q^T x \\ & \text{subject to } GTx = -\mathcal{F}_{\text{ext}} \\ & \quad NTx = F \cdot Wx \\ & \quad Wx \geq 0 \\ & \quad x_{i,1} \geq 0 \\ & \quad \gamma x_{i,1} - x_{i,4} \geq 0 \\ & \quad \gamma x_{i,1} + x_{i,4} \geq 0 \quad \forall i = 1 \dots n_c. \end{aligned} \quad (4.25)$$

The optimization task will be solved using an interior point optimizer, which is suitable for larger problems due to its polynomial complexity.

## 4.2 Implementation

### 4.2.1 Projection on Surface Frame

To start with force optimization, we need the finger forces in the contact frames, as they are required for the initial optimization variable  $x$ . However, in many cases, the finger frame and the surface frame of the object do not align. In MuJoCo, the gripper's finger frames have their contact normal aligned with their x-axis. Hence, we need rotation matrices to transform the finger forces to the contact surface frame with the x-axis pointing towards the negative surface normal direction.

MuJoCo can provide the force direction of the fingers, the position of the object's center of mass in world coordinates and the rotation matrix  ${}^{\text{com}}R_W$  from world to the object's frame. The projection matrix for the surface at contact  $i$  is defined as:

$${}^{c_i}R_W = {}^{c_i}R_{\text{com}} \cdot {}^{\text{com}}R_W. \quad (4.26)$$

Since the second rotational matrix is known, we only need to define a transformation from the object's center of mass frame to the contact frame. The only requirement

for the contact frame is that the x-axis points towards the negative surface normal. Therefore, this frame has ambiguous definitions. Once we detect on which object surface the finger forces are applied, we can choose the corresponding rotation matrix to project the finger forces onto the surface frame.

### Example: cube

For a cube we can define six surface frames: left, right, front, back, top and bottom.

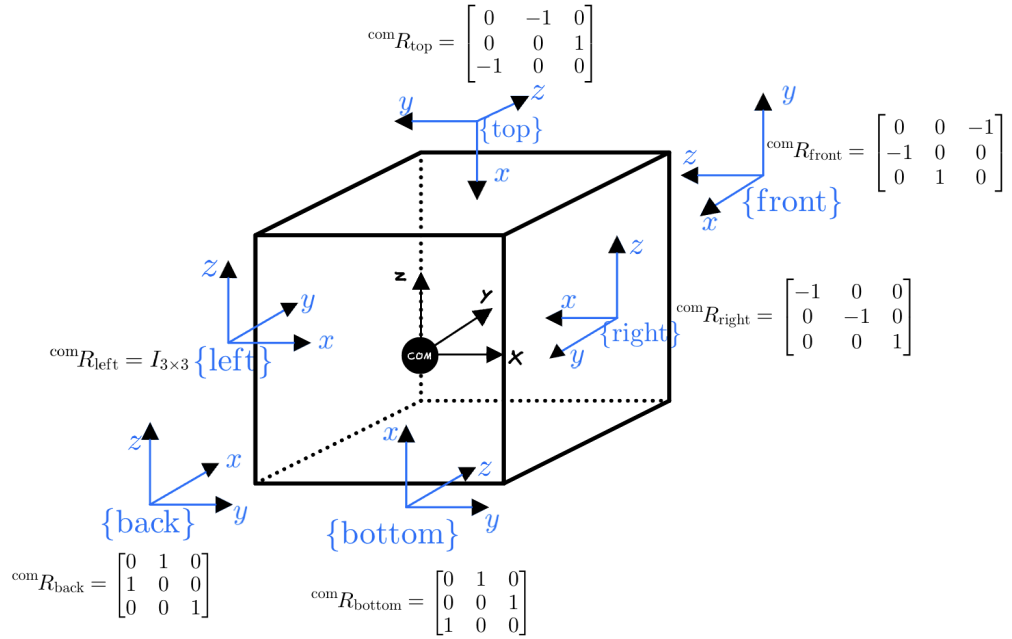


Figure 4.1: All six cube frames and their corresponding rotation matrices.

${}^{\text{com}}R_{c_i} = {}^{c_i}R_{\text{com}}^T$  can be determined by placing the basis vectors relative to the center of mass (com) frame in the columns. Note that in this case, the cube's surface frames can be defined in another way, as they could be rotated around their x-axis. For example,  ${}^{\text{com}}R_{\text{back}}$  can be obtained by expressing the surface frame's basis vectors in terms of the com frame's basis vectors. In this case, the x-axis of the backside frame corresponds to the y-axis of the com frame, which is why the first column of the rotation matrix  ${}^{\text{com}}R_{\text{back}}$  contains the y-axis vector. The same principle can be applied to the other axis vectors as well.

### General procedure for convex objects

The general procedure to calculate the rotation matrix  ${}^{c_i}R_{\text{com}}$  on an object with multiple flat surfaces goes as following:

1. First, define three points on each surface which are not on a line. By doing so we get two surface vectors and determine the normal via cross-product.
2. Redefine second surface basis vector such that all three vectors are orthogonal to each other.
3. Normalize the newly calculated basis vectors.
4. Define the basis vectors of the surface frame in the com frame by multiplying them with  ${}^{\text{com}}R_W$ .
5. Write these surface frame basis vectors as a linear combination of the com frame basis vectors.

$${}^{\text{com}}e_{c_i,x} = \alpha_1 {}^{\text{com}}e_x + \alpha_2 {}^{\text{com}}e_y + \alpha_3 {}^{\text{com}}e_z, \quad (4.27)$$

$${}^{\text{com}}e_{c_i,y} = \beta_1 {}^{\text{com}}e_x + \beta_2 {}^{\text{com}}e_y + \beta_3 {}^{\text{com}}e_z, \quad (4.28)$$

$${}^{\text{com}}e_{c_i,z} = \gamma_1 {}^{\text{com}}e_x + \gamma_2 {}^{\text{com}}e_y + \gamma_3 {}^{\text{com}}e_z. \quad (4.29)$$

Now we can write  ${}^cR_{\text{com}}$  dependent on the calculated parameters:

$${}^cR_{\text{com}} = \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix} \quad (4.30)$$

### 4.2.2 Correlation Cone Edges and Optimal Force

The force optimization is applied to a parallel jaw gripper that is grasping a cube. We can observe that the higher the number of edges for linearized friction cone, the more the optimal grasping force is reduced. The following table shows the optimized finger force in its surface normal direction for a sample grasp, depending on the number of linearized friction cone edges.

$s$	$f_{x,\text{right}}$ in [N]	$f_{x,\text{left}}$ in [N]
4	83.96	84.48
8	83.96	84.48
16	77.12	77.57
32	76.17	76.63
64	75.64	76.09

Table 4.1: contact normal force depending on number of edge vectors for linearized friction cone with  $\mu = 0.33$  and  $\gamma = 0.2$ .

# Chapter 5

## Evaluation

In this chapter, we will be looking at the simulation results after running force and position optimization on datasets of up to 100 sample grasps from reinforcement learning in MuJoCo. The grasped object is a cube with an edge length 0.05m and mass of 2.0kg. We will be using the two-fingered parallel jaw gripper for grasping. In addition, we set the friction coefficient values to  $\mu = 0.33$  and  $\gamma = 0.2$ . Furthermore, the rotation matrices for the cube surfaces as defined in Figure 4.1 will be used.

### 5.1 Position Optimization Results

The position optimization minimizes the moment and force residuals, which overall leads to a force closure grasp. Figure 5.1 illustrates the improvement achieved through contact position optimization by comparing a before-and-after view of a suboptimal initial grasp.

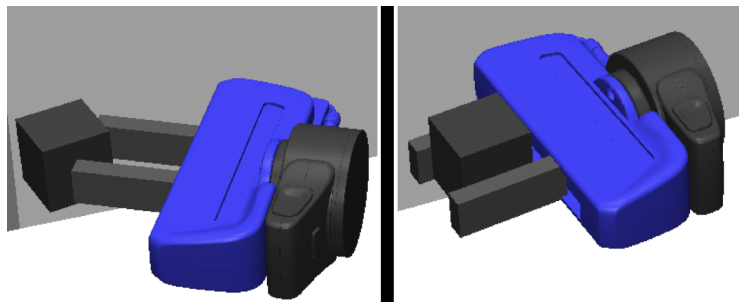


Figure 5.1: Sample grasp before (left) and after (right) contact position optimization in MuJoCo.

To evaluate the success of these algorithms, we can compare force and moment residuals before and after position optimization in figures 5.2 and 5.3.

The simulation results show that the position optimization algorithms are generally successful in reducing force and moment residuals. Numerical inaccuracies may cause uncertainties in the residual values, but we can assume that the grasps achieve



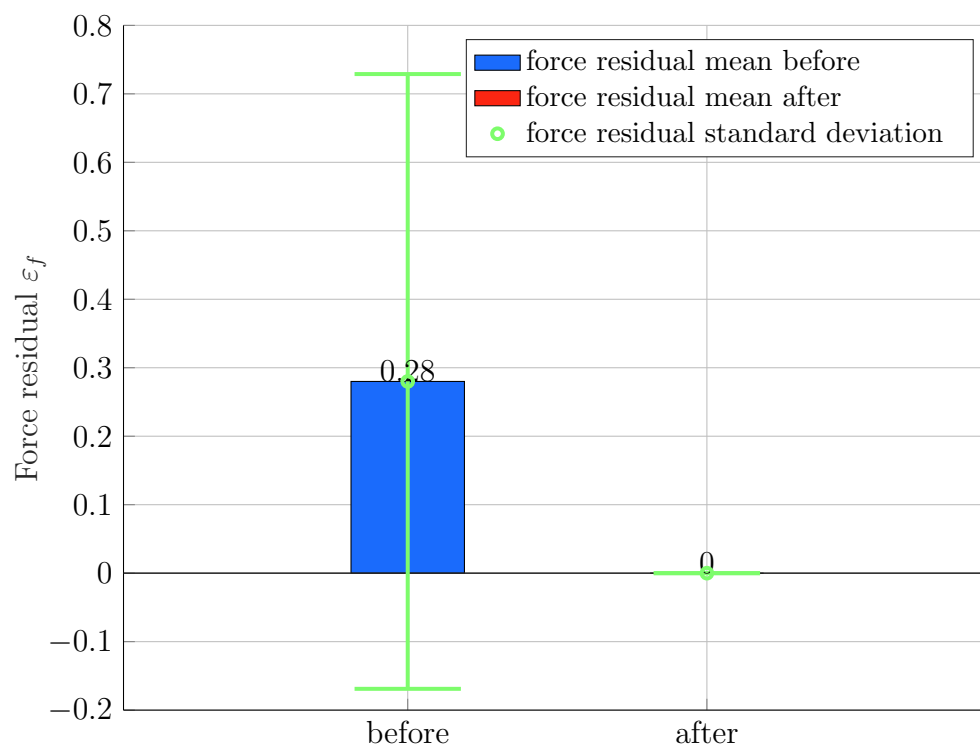


Figure 5.2: Force residual mean+std before and after moment residual control for  $n=100$  sample grasps.

UFE and force closure. Note that the mean values of the force and moment residuals are extremely small since the cube only has an edge length of 0.05m. However, due to the grasped object's small size, we are also unable to evaluate the grasp quality metrics mentioned in section 2.4.1 as there are minimal changes to the grasp matrix based metrics before and after contact point optimization.

## 5.2 Force Optimization Results

In this section, we will be taking a look at the improvements we get by only conducting force optimization on the sample grasps. To visualize the absolute amount of improvement our force optimization is able to achieve, we will take a look at the improvements for a selected number of grasps. The table in Figure 5.2 shows the median contact force of each parallel jaw finger before and after force optimization with SFCs and  $s = 32$  for six different grasps.

It can be seen that force optimization significantly reduces the required contact force for a given grasp. To illustrate this point even more, consider Figure 5.4, which demonstrates that the mean and standard deviation of contact forces after applying force optimization are significantly lower than before. This further high-

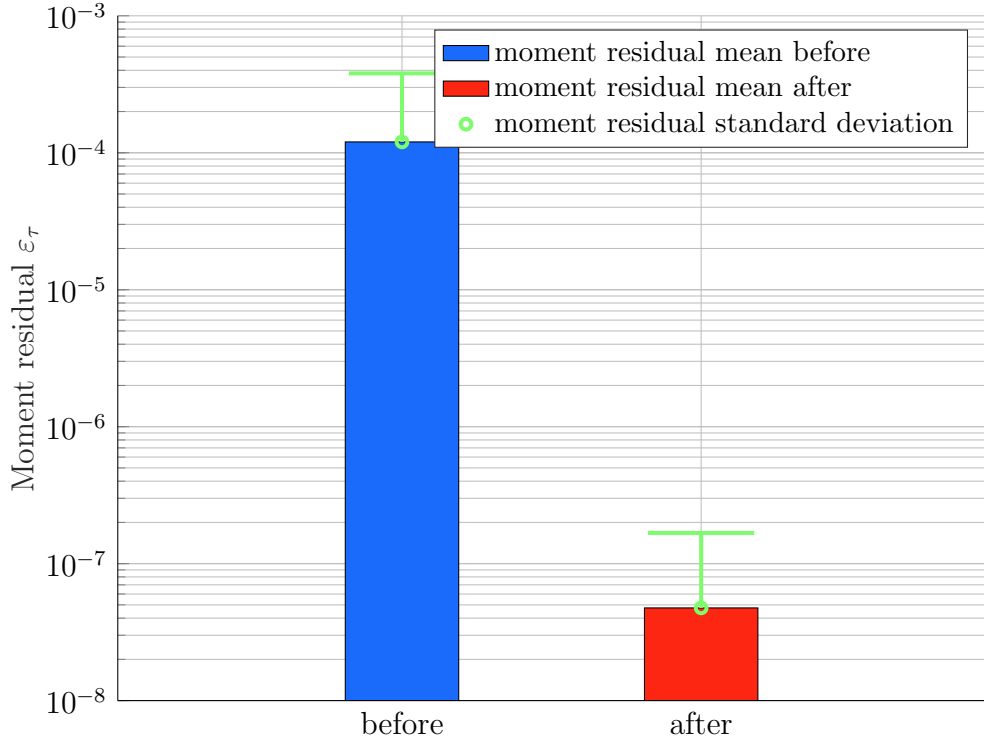


Figure 5.3: Moment residual mean+std before and after moment residual control for n=100 sample grasps.

Grasp nr.	$f_{x,\text{right,before}}$ in [N]	$f_{x,\text{left,before}}$ in [N]	$f_{x,\text{right,opt}}$ in [N]	$f_{x,\text{left,opt}}$ in [N]
0	79.09	76.10	40.09	39.96
1	127.63	129.80	47.82	47.24
2	156.53	158.82	73.88	72.45
3	148.26	149.57	53.91	53.97
4	145.74	145.53	62.54	61.53
5	145.64	146.32	30.70	30.61
6	148.97	150.15	88.10	87.56

Table 5.1: contact normal force for different grasps before and after force optimization with  $\mu = 0.33$  and  $\gamma = 0.2$ .

lights the effectiveness of the algorithm in reducing excessive contact forces and ensuring stable grasping.

### 5.3 Combined Optimization Results

Upon combining position and force optimization, we can observe a significant reduction in the average contact force compared to the initial grasps. Moreover, the average contact force is slightly lower than that achieved with force optimization alone.

Hence, we can conclude that the optimization of the initial grasps were successfully optimized.

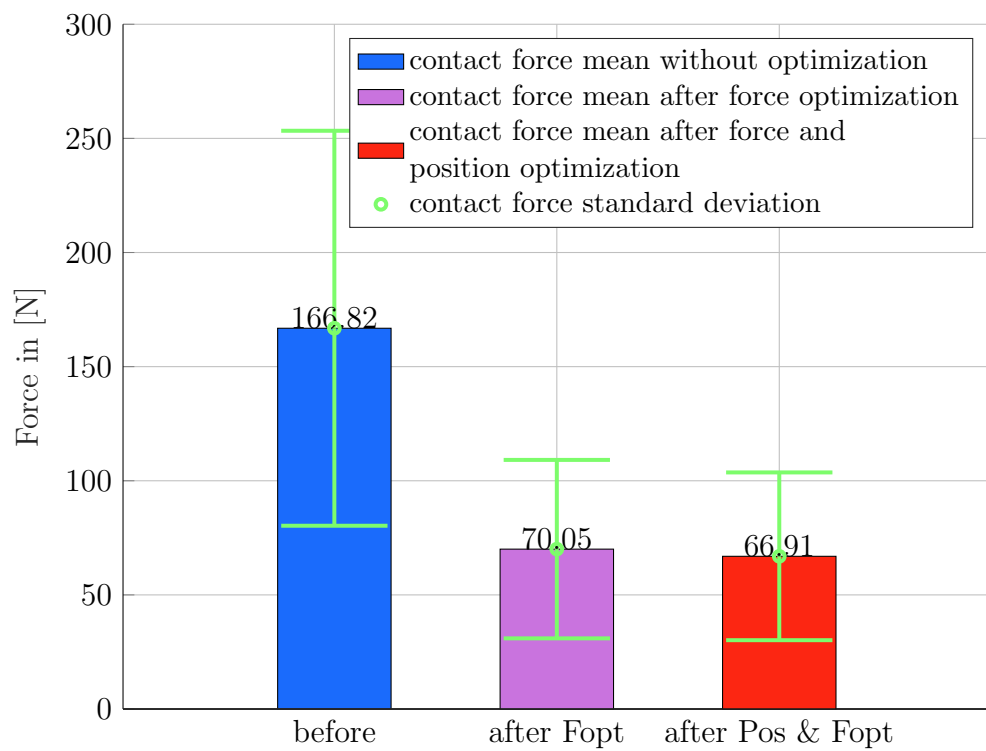


Figure 5.4: Contact force mean+std before and after different optimization methods for  $\mu = 0.33$  and  $\gamma = 0.2$ .



## Chapter 6

### Discussion

In this section, we will be discussing further improvements the applied optimization techniques could use for a more efficient and generalized application.

#### 6.1 Contact Position Optimization

Our contact point optimization method has two prerequisites. First, the condition in (3.20) must be met when choosing the parameters  $u$  and  $v$ . Second, the gradients must define a right-hand coordinate frame as defined in (3.5). However, these prerequisites are not always guaranteed to be met. For instance, in Algorithm 1 of the force residual control, we sometimes select  $u = \theta$  and  $v = \varphi$ , even though there is no right-hand frame in this case. Surprisingly, based on some examples, we observe that these requirements are not strictly necessary for achieving convergence.

##### 6.1.1 Residual Control Algorithm Generalizations

In general it appears that the selection of  $u$  and  $v$  does not matter at all for force residual control since our gradient descent rule for each  $\mathbf{u}_i$  is depicted as:

$$\tilde{\mathbf{u}}_{i,k+1} = \tilde{\mathbf{u}}_{i,k} - \gamma \left( \frac{\partial \mathbf{f}}{\partial \tilde{\mathbf{u}}_{i,k}} \right)^T \mathbf{f} = \begin{bmatrix} u_{i,k} \\ v_{i,k} \end{bmatrix} - \gamma \begin{bmatrix} \nabla_u r_i \\ \nabla_v r_i \end{bmatrix}^T \mathbf{f}, \quad (6.1)$$

where we can see that setting  $u_i = \varphi_i$  or  $u_i = \theta_i$  does not influence the update of variable in any way. Since we cannot always guarantee a right-hand frame at the contact, the condition (3.22) cannot be fulfilled as well. However, the controller still converges, as shown in example 6.1.1.

Moment residual control, on the other hand, requires the correct selection of the parameters  $u$  and  $v$  since swapping them would require gradient ascend instead of descent, which can be noticed in the update rule below:

$$\begin{bmatrix} u_{i,k+1} \\ v_{i,k+1} \end{bmatrix} = \begin{bmatrix} u_{i,k} \\ v_{i,k} \end{bmatrix} - \gamma \begin{bmatrix} -\nabla_v r_i \\ \nabla_u r_i \end{bmatrix}^T \tau. \quad (6.2)$$

Therefore, we will be looking into two examples which show that the force residual controller works without satisfying any of the previously mentioned requirements. Furthermore, a new rule for adequately choosing  $u$  and  $v$  for moment residual control will be proposed.

**Example 1**

For this example we apply force residual control on following object with hexagonal base: The object is grasped at two contact points located on the middle left and

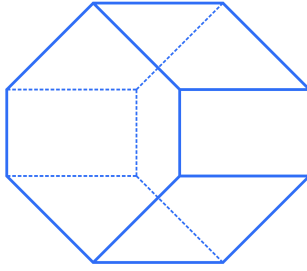


Figure 6.1: Visualization 3d object with hexagonal base.

bottom right sides of the object. To simplify the analysis, we assume that the y-component of the contact position is equal for both fingers. As a result, we can depict the finger movements in the x-z plane, as shown in Figure 6.2. We can now calculate the force residual and its Jacobian:

$$f = n_1 + n_2 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 1 \\ -\frac{1}{\sqrt{2}} \end{bmatrix} + \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 - \sqrt{2} \\ 0 \\ -1 \end{bmatrix}, \quad (6.3)$$

$$J_f = \begin{bmatrix} \nabla_{\varphi} r_1^T \\ \nabla_{\theta} r_1^T \\ \nabla_{\varphi} r_2^T \\ \nabla_{\theta} r_2^T \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 0.7071 & 0 & -0.7071 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}.$$

With these values we can now apply a couple steps of force residual control. Even though (3.20) is not fulfilled, we are still converging. The movement directions are shown in Figure 6.2. It can be observed that once contact  $P_1$  switches to the next surface, the force residual will become zero and we have achieved unit frictionless force equilibrium.

When applying moment residual control with  $u = \theta$  and  $v = \varphi$ , contact point  $P_2$  moves in the opposite direction while  $P_1$ 's movement direction remains the same compared to the ones in figure 6.2. This leads to unit frictionless moment equilibrium after a couple of iterations. Hence, we can propose a new method of determining  $u$

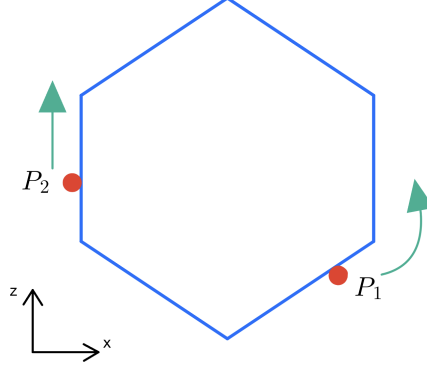


Figure 6.2: Visualization of contact point movement direction on 6.1 during force residual control.

and  $v$  with a new decision variable

$$a = |\nabla_{\varphi} r_1^T \nabla_{\varphi} r_2| - |\nabla_{\theta} r_1^T \nabla_{\theta} r_2|, \quad (6.4)$$

such that

$$v = \begin{cases} \varphi, & a \geq 0 \\ \theta, & a < 0 \end{cases}. \quad (6.5)$$

To check whether this rule works correctly, we will be applying moment residual control on another grasp configuration at the same object in 6.1.1.

### Example 2

In this example, the first contact is on the lower left side while the second one is in the center of the backside as seen in Figure 6.3. By selecting  $v$  and  $u$  through

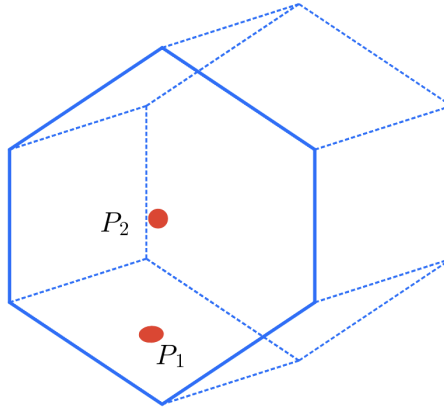


Figure 6.3: Visualization of contact point configuration for example 2 .

(6.5) we are able to reduce the moment residual at each iteration which appears



to support the previous claim. However, the authors of [PFG10] have only proven the convergence of the switching grasp controller when the strict conditions (3.20) and (3.5) are fulfilled. Therefore, further experiments on convex objects need to be conducted before our hypothesis can be fully confirmed.

### **6.1.2 Further Issues**

Another issue is that force residual control is unable to set a finger force to zero, resulting in non-ideal configurations. For example, when using a three-fingered gripper on a cube, it may be impossible to reach unit frictionless force equilibrium, even though it is possible with two fingers. Therefore, the algorithms need to be able to determine whether UFE can be reached with fewer contacts, if the current configuration cannot reach it.

## Chapter 7

# Conclusion

We have presented methods for contact force and position optimization for multi-fingered robotic grasping that attempt to achieve force closure. These optimization methods have been successfully implemented on a parallel jaw gripper while grasping a cube. It was shown that extending the position optimization algorithm for other more complex convex objects demands different and less strict parameter selection requirements than the ones presented in [PFG10].

In conclusion, the presented grasp optimization methods effectively reduce the required grasping force and improve contact locations to achieve lower moment and force residuals. However, further refinement is necessary for successful application to more complex objects and different grippers.



# Appendix A

## A.1 Notation

Symbol	Description
$\mathcal{F}$	Wrench $\in \mathbb{R}^6$
$\mathcal{V}$	Twist $\in \mathbb{R}^6$
$\mu$	Dry friction coefficient
$\gamma$	Torsional friction coefficient
$n_c$	Total number of contacts in grasp
$s$	Number of edges in linearized friction cone
$l_i$	Number of transmitted wrench components at contact i
$l$	Total number of transmitted wrench components in grasp
$f_{c_i}$	Contact force $\in \mathbb{R}^{l_i}$ at contact nr. i
$f_i$	Force $\in \mathbb{R}^3$ at contact nr. i
$\tau_i$	Torque $\in \mathbb{R}^3$ at contact nr. i
$f_c$	Combined contact force $\in \mathbb{R}^l$ applied in the grasp
$f_{k,c_i}$	k th friction cone edge vector $\in \mathbb{R}^3$ at contact nr. i
$F_c$	Friction cone edge vector matrix $\in \mathbb{R}^{3 \times s}$ for arbitrary contact
$F$	Cumulative friction cone edge vector matrix $\in \mathbb{R}^{3n_c \times sn_c}$
$G$	Grasp matrix $\in \mathbb{R}^{6 \times l}$
$T$	Filtermatrix $\in \mathbb{R}^{l \times (l+n_c \cdot s)}$ to extract $f_c$ from input vector $x$
$W$	Filtermatrix $\in \mathbb{R}^{(n_c \cdot s) \times (l+n_c \cdot s)}$ to extract $\alpha_c$ from input vector $x$
$N$	Filtermatrix $\in \mathbb{R}^{3n_c \times l}$ to extract force components from $f_c$
$\mathbf{f}$	Sum of all unit contact forces, $\in \mathbb{R}^3$
$\boldsymbol{\tau}$	Sum of all contact torques for unit forces, $\in \mathbb{R}^3$
$\varepsilon_f$	Unit frictionless force residual, $\in \mathbb{R}$
$\varepsilon_\tau$	Unit frictionless moment residual, $\in \mathbb{R}$
$J_f$	Unit frictionless force residual Jacobian, $\in \mathbb{R}^{3 \times 2n_c}$
	Note: $J_f = \hat{J}_f$ for curvature matrix $K = I_{2n_c}$
$J_\tau$	Unit frictionless moment residual Jacobian $\in \mathbb{R}^{3 \times 2n_c}$
	Note: $J_\tau \approx \hat{J}_\tau$

Table A.1: Notation table

# List of Figures

1.1	Different variants of grasping algorithms according to [CLFVW16]. . . . .	6
2.1	Visualization of nonlinear friction cone [LMSB14], . . . . .	10
2.2	Visualization of linearized friction cone [Boh19]. . . . .	11
2.3	Selection matrices for the common contact types, based on [SK16]. . . . .	14
3.1	Example of alternating configuration when grasping a cube. . . . .	26
4.1	All six cube frames and their corresponding rotation matrices. . . . .	35
5.1	Sample grasp before (left) and after (right) contact position optimization in MuJoCo. . . . .	37
5.2	Force residual mean+std before and after moment residual control for n=100 sample grasps. . . . .	38
5.3	Moment residual mean+std before and after moment residual control for n=100 sample grasps. . . . .	39
5.4	Contact force mean+std before and after different optimization methods for $\mu = 0.33$ and $\gamma = 0.2$ . . . . .	41
6.1	Visualization 3d object with hexagonal base. . . . .	44
6.2	Visualization of contact point movement direction on 6.1 during force residual control. . . . .	45
6.3	Visualization of contact point configuration for example 2 . . . . .	45









## Bibliography

- [BFM97] M. Buss, L. Faybusovich, and J.B. Moore. Recursive algorithms for real-time grasping force optimization. In *Proceedings of International Conference on Robotics and Automation*, volume 1, pages 682–687 vol.1, April 1997. doi:10.1109/ROBOT.1997.620115.
- [BHM96] M. Buss, H. Hashimoto, and J.B. Moore. Dextrous hand grasping force optimization. 12(3):406–418, June 1996. Conference Name: IEEE Transactions on Robotics and Automation. doi:10.1109/70.499823.
- [Boh19] Sadigh Bohg, Pavone. Lecture 5: Fundamentals of Grasping. 2019. URL: [http://web.stanford.edu/class/cs237b/pdfs/lecture/lecture\\_567.pdf](http://web.stanford.edu/class/cs237b/pdfs/lecture/lecture_567.pdf).
- [CLFVW16] Stefano Carpin, Shuo Liu, Joe Falco, and Karl Van Wyk. Multi-fingered robotic grasping: A primer, 2016. URL: <http://arxiv.org/abs/1607.06620>, doi:10.48550/arXiv.1607.06620.
- [DMT18] Hongkai Dai, Anirudha Majumdar, and Russ Tedrake. Synthesis and optimization of force closure grasps via sequential semidefinite programming. In Antonio Bicchi and Wolfram Burgard, editors, *Robotics Research: Volume 1*, Springer Proceedings in Advanced Robotics, pages 285–305. Springer International Publishing, 2018. doi:10.1007/978-3-319-51532-8\_18.
- [HSSR05] R. Haschke, J.J. Steil, I. Steuwer, and H. Ritter. Task-oriented quality measures for dextrous grasping. In *2005 International Symposium on Computational Intelligence in Robotics and Automation*, pages 689–694, June 2005. doi:10.1109/CIRA.2005.1554357.
- [HTL99] L. Han, J.C. Trinkle, and Z. Li. Grasp analysis as linear matrix inequality problems. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, volume 2, pages 1261–1268 vol.2, 1999. ISSN: 1050-4729. doi:10.1109/ROBOT.1999.772534.

- [JC10] Hyunhwan Jeong and Joono Cheong. Evaluation of grasps for 3d objects with physical interpretations using object wrench space. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 604–609, October 2010. ISSN: 2153-0866. doi:10.1109/IRoS.2010.5649123.
- [LHBR12] Qiang Li, Robert Haschke, Bram Bolder, and Helge Ritter. Grasp point optimization by online exploration of unknown object surface. In *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pages 417–422, 2012. ISSN: 2164-0580. doi:10.1109/HUMANOIDS.2012.6651553.
- [Li94] Sastry Li, Murray. A Mathematical Introduction to Robotic Manipulation - 1st Edition - Ri, 1994. URL: <https://www.ce.cit.tum.de/fileadmin/w00cgn/rm/pdf/murray-li-sastry-94-complete.pdf>.
- [LL04] Guanfeng Liu and Zexiang Li. Real-time grasping-force optimization for multifingered manipulation: theory and experiments. 9(1):65–77, March 2004. Conference Name: IEEE/ASME Transactions on Mechatronics. doi:10.1109/TMECH.2004.823879.
- [LMSB14] Beatriz León, Antonio Morales, and Joaquin Sancho-Bru. Robot Grasping Foundations. In Beatriz León, Antonio Morales, and Joaquín Sancho-Bru, editors, *From Robot to Human Grasping Simulation*, Cognitive Systems Monographs, pages 15–31. Springer International Publishing, Cham, 2014. doi:10.1007/978-3-319-01833-1\_2.
- [LP17] Kevin M. Lynch and Frank C. Park. *Modern robotics: mechanics, planning, and control*. Cambridge University Press, 2017.
- [NW06] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer Science & Business Media, 2006. Google-Books-ID: VbHYoSylFcC.
- [PFG10] Robert Platt, Andrew H. Fagg, and Roderic A. Grupen. Null-Space Grasp Control: Theory and Experiments. *IEEE Transactions on Robotics*, 26(2):282–295, April 2010. Conference Name: IEEE Transactions on Robotics. doi:10.1109/TR0.2010.2042754.
- [Pre10] Andrew Pressley. *Elementary Differential Geometry*. Springer Undergraduate Mathematics Series. Springer, 2010. URL: <http://link.springer.com/10.1007/978-1-84882-891-9>, doi:10.1007/978-1-84882-891-9.
- [RB19] Elon Rimon and Joel Burdick. *The Mechanics of Robot Grasping*. Cambridge University Press, Cambridge, 2019. URL: <https://www.cambridge.org/core/books/mechanics-of-robot-grasping/>

- OD79AD1A613397256CB1AEA67F3B3A6E, doi:10.1017/9781108552011.
- [RS15] Máximo A. Roa and Raúl Suárez. Grasp quality measures: review and performance. *Autonomous Robots*, 38(1):65–88, January 2015. doi:10.1007/s10514-014-9402-3.
- [SBC<sup>+</sup>22] Martin Schuck, Jan Brüdigam, Alexandre Capone, Stefan Sosnowski, and Sandra Hirche. *Dext-Gen: Dexterous Grasping in Sparse Reward Environments with Full Orientation Control*. 2022. doi:10.48550/arXiv.2206.13966.
- [SK16] Bruno Siciliano and Oussama Khatib. Springer handbook of robotics, 2016. URL: <https://link.springer.com/10.1007/978-3-319-32552-1>, doi:10.1007/978-3-319-32552-1.
- [TW02] Wai Sum Tang and Jun Wang. A Lagrangian network for multifingered hand grasping force optimization. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, volume 1, pages 177–182 vol.1, May 2002. ISSN: 1098-7576. doi:10.1109/IJCNN.2002.1005465.
- [XWF04] Youshen Xia, Jun Wang, and Lo-Ming Fok. Grasping-force optimization for multifingered robotic hands using a recurrent neural network. *IEEE Transactions on Robotics and Automation*, 20(3):549–554, June 2004. Conference Name: IEEE Transactions on Robotics and Automation. doi:10.1109/TRA.2004.824946.
- [ZW03] Xiangyang Zhu and Jun Wang. Synthesis of force-closure grasps on 3-D objects based on the Q distance. *IEEE Transactions on Robotics and Automation*, 19(4):669–679, August 2003. Conference Name: IEEE Transactions on Robotics and Automation. doi:10.1109/TRA.2003.814499.
- [ZWH05] Yu Zheng and Qian W.-H. Coping with the grasping uncertainties in force-closure analysis. 24:311–327, 2005. doi:10.1177/0278364905049469.



# License

This work is licensed under the Creative Commons Attribution 3.0 Germany License. To view a copy of this license, visit <http://creativecommons.org> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.