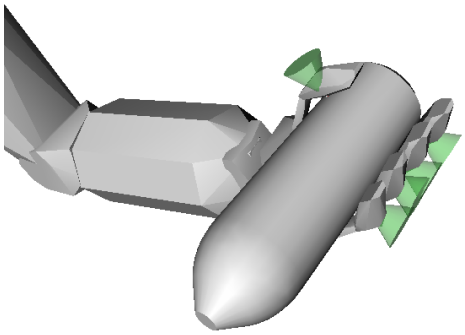
TÉCNICO LISBOA



Robot grasping in 3D through efficient haptic exploration with unscented Bayesian optimization and collision penalty

João Pedro Morais Castanheira

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor(s): Prof. Alexandre José Malheiro Bernardino

Examination Committee

Chairperson: Prof. João Fernando Cardoso Silva Sequeira
Supervisor: Prof. Alexandre José Malheiro Bernardino
Member of the Committee: Prof. Rodrigo Martins de Matos Ventura

June 2018

To my dear grandmother...

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

I would like to thank all the time and support provided by Prof. Alexandre Bernardino and Pedro Vicente throughout the duration of the project. All your expertise and guidance allowed for a clearer path to the final objective. Also, I want to thank Lorenzo Jamone and Ruben Martinez-Cantin for taking the time to review the paper, thus contributing for a more complete technical review.

To all my friends that were additional sources motivation and support during the entirety of my studies.

A special note of appreciation to my family, my parents and my siblings, for providing me with all I could wish for and all the love one could hope.

Resumo

O agarre robusto de objectos é ainda um problema por resolver na robótica. Informação global 3D acerca do objecto pode ser obtida através de informação já conhecida (*e.g.*, modelos precisos de objectos conhecidos ou modelos aproximados de objectos que nos são familiares) ou sensores em tempo-real (*e.g.*, nuvens de pontos parciais de objectos desconhecidos) e pode ser usada para identificar potenciais bons agarres. Contudo, devido a imprecisões de modelação e limitações sensoriais, a exploração local é normalmente necessária para complementar os tais agarres e aplicá-los a objectos no mundo real. Nomeadamente, a técnica de optimização Bayesiana "*unscented*", recentemente proposta, é capaz de tornar essa exploração mais segura por favorecer a escolha de agarres que são robustos à incerteza no espaço de entrada (*e.g.*, imprecisões na execução do agarre).

Extendendo o trabalho prévio feito para a optimização em 2D, nesta dissertação propomos uma estratégia de exploração táctil 3D que combina optimização unscented Bayesiana com uma nova heurística. Esta penaliza colisões inesperadas entre a mão e objecto durante a exploração, para encontrar agarres mais seguros em 3D de forma muito eficiente. Ao expandir o espaço de exploração de 2D para 3D somos capazes de encontrar melhores agarres e a introdução da penalização de colisões permite fazê-lo sem ter de aumentar o número de explorações, portanto ajudando a combater o desafio do aumento de dimensões.

Palavras-chave: Agarre robótico, Optimização Bayesiana, Exploração táctil, Penalização de colisão, Optimização Bayesiana "*unscented*", iCub.

Abstract

Robust grasping of arbitrary objects is a major, and still unsolved problem in robotics. Global 3D information about the object can be obtained from previous knowledge (*e.g.*, accurate models of known objects or approximate models of familiar objects) or real-time sensing (*e.g.*, partial point clouds of unknown objects) and can be used to identify good potential grasps; however, due to modeling inaccuracies and sensing limitations, local exploration is often needed to refine such grasps and to successfully apply them to the objects in the real world. Notably, the recently proposed unscented Bayesian optimization technique can make such exploration safer by favoring the selection of grasps that are robust to uncertainty in the input space (*e.g.*, inaccuracies in the execution of the grasp). By finding a safer grasp, we assure that a slight error in grasp position due to execution noise won't cause a dramatic decrease of the grasp quality.

Extending the previous work done on 2D optimization, in this thesis we propose a 3D haptic exploration strategy that combines unscented Bayesian optimization with a novel collision penalty heuristic to find safe 3D grasps in a very efficient way. By expanding the search-space from 2D to 3D we are able to find better grasps and the introduction of the collision penalty heuristic allows to do so without having to increase the number of exploration steps, therefore battling the curse of dimensionality.

Keywords: Robotic Grasp, Bayesian optimization, Haptic exploration, Collision Penalty, Unscented Bayesian Optimization, iCub.

Contents

1 Introduction **1**

1.1 Motivation . . . 2
1.2 Objectives . . . 3
1.3 State-of-the-Art . . . 3
1.4 Contributions . . . 4
1.5 Thesis Outline . . . 5

2 Robotic Grasp **7**

2.1 Wrenches . . . 8
2.2 Contact Model . . . 8
2.3 Grasp Representation . . . 11
2.4 Grasp stability . . . 12
2.5 Grasp Quality Metric . . . 12

3 Optimization **14**

3.1 Bayesian optimization . . . 15
3.1.1 Gaussian Process . . . 15
3.1.2 Learning hyperparameters . . . 17
3.1.3 Surrogate model estimations . . . 17
3.1.4 Decision using acquisition function . . . 19
3.2 Unscented Bayesian optimization . . . 20
3.2.1 Unscented expected improvement . . . 20
3.2.2 Unscented optimal incumbent . . . 21
3.3 DIRECT algorithm . . . 22
3.4 Grasp position optimization problem . . . 23
3.4.1 Collision Penalty . . . 24

4 Experimental Setup and Results **27**

4.1 Experimental Setup . . . 28
4.1.1 Simox Simulator . . . 28
4.1.2 BayesOpt . . . 29

4.1.3 Experimental Design 29
4.2 Experimental Results 30
4.2.1 Collision Penalty tuning 30
4.2.2 Benefits of the Collision Penalty 31
4.2.3 Generalization to 3D 32
4.2.4 Advantages of UBO over BO 37

5 Conclusions and Future work **41**

Bibliography **43**

List of Tables

4.1 Results at the last iteration ($n = 160$) of the optimization process (means over all runs) . . 37

List of Figures

1.1 Examples of objects to perform grasp optimization on simulation. Initial pose for each test object . . . 2

2.1 Contacts and friction cones . . . 9
2.2 Friction cone approximation . . . 11
2.3 On the left, the convex hull of the grasp wrenches known as the GWS (simplified example). On the right, the Largest minimum Resisted Wrench metric. . . . 12

3.1 Five one-dimensional functions randomly sampled from a GP prior with kernel k_{M52} . . . 16
3.2 On the left, five one-dimensional functions randomly sampled from a GP posterior. On the right, the predictive approximation which is the GP posterior mean . . . 18
3.3 Finding the best quality grasp requires the maximization of an black-box function using a limited number of evaluations. The goal is to optimize the hand Cartesian position so that it maximizes the quality of the grasp. . . . 23
3.4 Grasp optimization process . . . 23
3.5 Collision examples . . . 24
3.6 Influence of parameter λ on the Collision Penalty function . . . 24

4.1 Simox simulation window . . . 28
4.2 Flowchart of the program. The boxes in blue correspond to stages that are not necessary for the functioning of the system, since they only take part for its evaluation. . . . 30
4.3 Performance of the 3D UBO CP on the Mug with different with different tuning parameters λ. Left: expected outcome of current optimum $\bar{\mathrm{y}}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})$, Right: Variability of the outcome $\mathrm{std}\left(\mathrm{y}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})\right)$ . . . 31
4.4 **Glass: CP vs $\overline{\mathbf{CP}}$** (UBO 3D). Left: expected outcome of current optimum $\bar{\mathrm{y}}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})$, Right: Variability of the outcome $\mathrm{std}\left(\mathrm{y}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})\right)$ . . . 32
4.5 **Bottle: CP vs $\overline{\mathbf{CP}}$** (UBO 3D). Left: expected outcome of current optimum $\bar{\mathrm{y}}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})$, Right: Variability of the outcome $\mathrm{std}\left(\mathrm{y}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})\right)$ . . . 32
4.6 **Mug: CP vs $\overline{\mathbf{CP}}$** (UBO 3D). Left: expected outcome of current optimum $\bar{\mathrm{y}}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})$, Right: Variability of the outcome $\mathrm{std}\left(\mathrm{y}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})\right)$ . . . 33
4.7 **Glass: 2D vs 3D** (UBO CP). Left: expected outcome of current optimum $\bar{\mathrm{y}}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})$, Right: Variability of the outcome $\mathrm{std}\left(\mathrm{y}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})\right)$ . . . 33

4.8 **Bottle: 2D vs 3D** (UBO CP). Left: expected outcome of current optimum $\bar{y}_{mc}(\mathbf{x}_n^{ubopt})$, Right: Variability of the outcome std$\left(y_{mc}(\mathbf{x}_n^{ubopt})\right)$ 34
4.9 **Mug: 2D vs 3D** (UBO CP). Left: expected outcome of current optimum $\bar{y}_{mc}(\mathbf{x}_n^{ubopt})$, Right: Variability of the outcome std$\left(y_{mc}(\mathbf{x}_n^{ubopt})\right)$ 34
4.10 **Glass**: Optimal parameters $\mathbf{x}_n^{ubopt}$ for each of the 20 runs using 3D UBO CP strategy. . . . 35
4.11 **Mug**: Optimal parameters $\mathbf{x}_n^{ubopt}$ for each of the 20 runs using 3D UBO CP strategy. . . . 35
4.12 Quality of best grasp in one of the runs. 2D UBO CP (a) and 3D UBO CP (b) converge almost to the same grasp 36
4.13 **Bottle**: Optimal parameters $\mathbf{x}_n^{ubopt}$ for each of the 20 runs using 3D UBO CP strategy. . . 36
4.14 **Glass: BO vs UBO** (3D CP). Left: expected outcome of current optimum $\bar{y}_{mc}(\mathbf{x}_n^{opt})$, Right: Variability of the outcome std$\left(y_{mc}(\mathbf{x}_n^{opt})\right)$ 37
4.15 **Bottle: BO vs UBO** (3D CP). Left: expected outcome of current optimum $\bar{y}_{mc}(\mathbf{x}_n^{opt})$, Right: Variability of the outcome std$\left(y_{mc}(\mathbf{x}_n^{opt})\right)$ 38
4.16 **Mug: BO vs UBO** (3D CP). Left: expected outcome of current optimum $\bar{y}_{mc}(\mathbf{x}_n^{opt})$, Right: Variability of the outcome std$\left(y_{mc}(\mathbf{x}_n^{opt})\right)$ 38
4.17 Quality of best grasp in one of the runs. The best grasp achieved by BO is in an unsafe zone (a). The UBO's best grasp has a lower observation outcome but it is more robust to input noise (b) 39

List of Symbols

$\boldsymbol{B}_{c_i}$	Wrench basis at contact i
$\boldsymbol{d}_{c_i}$	Distance vector from the center of mass of the object to the contact point $\mathbf{p}_{c_i}$
$\delta_x, \delta_y, \delta_z$	Simox input variables
d	Number of dimensions of the problem
$\boldsymbol{f}$	Force
$\boldsymbol{f}_{c_i}$	Force magnitudes at contact i
$\boldsymbol{f}_{c_{ij}}$	Primitive forces of contact i
f_{min}	Global minimum resultant from DIRECT
f_n	Magnitude of the normal force component
f_t	Magnitude of the tangential force component
FC_{c_i}	Friction cone resultant from contact i
$\mathbf{G}$	Complete grasp map
$\mathbf{G}_i$	Partial grasp map of contact i
$\mathcal{GP}$	Gaussian Process
k_{M52}	$\mathcal{GP}$ ARD Matérn 5/2 kernel function
λ	Tuning parameter for CP
l_i	Kernel's length on dimension i hyperparameter
μ	$\mathcal{GP}$ mean function
μ_f	Static friction coefficient
n_c	Number of contacts
n_j	Number of colliding joints
$\bar{\mathbf{\Omega}}$	Function domain
$\mathbf{p}_{ba}$	Position of the origin of frame B seen from the origin of frame A
$\mathbf{p}_{c_i}$	Location of contact i
Φ	Gaussian probability cumulative function
ϕ	Gaussian probability density function
$[\mathbf{p}_{ba}]_\times$	Screw-symmetric matrix of $\mathbf{p}_{ba}$
Q_{LRW}	Largest minimum Resisted Wrench quality metric
$\mathbb{R}$	Set of real numbers
$\mathbf{R}_{ab}$	Rotation matrix from coordinate frame B to A

$\Sigma'_{\mathbf{x}}$	Input space noise covariance matrix
σ_n	Observation noise
σ_x	Input noise
$\boldsymbol{\tau}$	Torque
$\boldsymbol{\theta}$	Kernel's hyperparameters
θ_0	Kernel's overall amplitude hyperparameter
$\boldsymbol{w}$	Wrench
$\boldsymbol{w}_e$	External wrench
$\boldsymbol{w}_o$	Total object wrench
$\boldsymbol{w}_{c_i}$	Wrench applied at contact i
$\boldsymbol{w}_{c_{ij}}$	Primitive wrenches of contact i
$w^0, w_+^{(i)}, w_-^{(i)}$	Sigma points' weights
W	Wrench Space
W_{gws}	Grasp Wrench Space
∂W_{gws}	Boundary of the Grasp Wrench Space
$\mathbf{X}$	Observation queries
$\mathbf{x}^0, \mathbf{x}_+^{(i)}, \mathbf{x}_-^{(i)}$	Sigma points
$\mathcal{X}$	Input space of objective function f
$\mathbf{x}_n^{\text{b}opt}$	Optimal query of BO
$\mathbf{x}_n^{\text{ub}opt}$	Optimal query of UBO
$\mathbf{x}_n^{opt}$	Optimal query at iteration n
ξ	Trade-off parameter between exploration and exploitation
$\hat{y}$	Objective function estimation
$\mathbf{y}$	Outcomes of observation queries
y_{mc}	Outcome of Monte Carlo sample
$\bar{\mathsf{y}}_{\text{mc}}$	Mean outcome of Monte Carlo samples
$\mathsf{y}_n^{\text{b}opt}$	Incumbent in BO
$\mathsf{y}_n^{\text{ub}opt}$	Incumbent in UBO

Acronyms

3D UBO CP 3D unscented Bayesian optimization with collision penalty. 30

ARD Automatic Relevance Determination. 16

BO Bayesian optimization. 3, 4, 5, 15, 16, 21, 29, 30, 37

CP Collision penalty. 5, 25, 28, 30, 31, 32, 33, 36

GP Gaussian Process. 15, 16, 17, 18, 21, 23

GWS Grasp Wrench Space. 12

LHS Latin Hypercube Sampling. 19, 29

LRW Largest minimum Resisted Wrench. 4, 12, 23, 28

MCMC Markov chain Monte Carlo. 17

UBO unscented Bayesian optimization. 4, 5, 21, 22, 29, 30, 32, 36, 37

UEI unscented expected improvement. 21

1 Introduction

Contents

1.1 Motivation 2
1.2 Objectives 3
1.3 State-of-the-Art 3
1.4 Contributions 4
1.5 Thesis Outline 5

1.1 Motivation

Robotic grasping is a very important research area in robotics that has been around for many years now [1, 2, 3, 4]. Its early applications were designed for the industrial level, where usually robots are part of assembly lines and perform grasping tasks like moving components, lifting objects or simply hold tools to perform an action. These robots have typically 6 degrees of freedom and perform repetitive tasks, therefore the problem is well formulated and accurately modelled, from the perspective of the object that is being grasped but also relative to the robot kinematics.

In parallel, humanoid robots are also an extensive field of application. These robots present a more complex problem when it comes to grasping, since mostly they use anthropomorphic hands, which have more degrees of freedom than typical industrial manipulators [5, 6]. Fine manipulation tasks, like grasping an object with precision or pushing small buttons with dexterity, are quite simple for humans, who deal with these activities on a daily basis, but complex and challenging for autonomous robots. To obtain similar dexterity with robotic hands, the robot has to be able to deal with uncertainty and

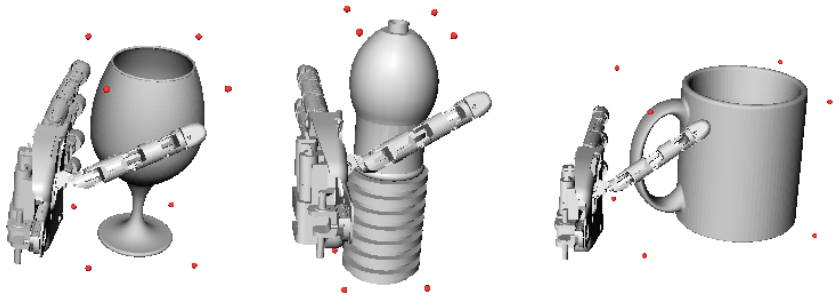


Figure 1.1: Examples of objects to perform grasp optimization on simulation. Initial pose for each test object

incrementally incorporate knowledge from previous grasp trials to adapt to different environments and tasks. Meanwhile, the human hand naturally provides sensory feedback (sensing slip, object weight, object stability, etc.) and intuitively weights the importance of all these parameters to decide on what is a safe grasp. Robots on the other hand are equipped with a limited number of sensors that try to capture as much information as possible, yet they are subjected to limitations that increase the difficulty of the grasping problem. The robot can not blindly accept sensory information as deterministic, otherwise it will incur in errors. It must account and deal with the uncertainty of sensor data to make its decisions. Overall, the robust grasping of arbitrary objects is still an open problem.

Naturally, an approach inspired by human learning, is for the robot to learn how to grasp by teaching how to mimic human behaviour. Thus, there are several approaches which rely on learning by human demonstration [7, 8]. An equally interesting approach is to let the robot learn independently, learning from several trials and correcting its on behaviour to be able to complete a task [9, 10, 11]. This approach is known as active learning, a technique for online machine learning, where both prior data and samples acquired during the task execution are used to learn a certain objective. The objective is set so that the robot learns, for arbitrary objects, the function that relates a certain end-effector configuration to its

grasp quality. In this case, the objective function is a black-box function, *i.e.*, we do not have access to its mathematical expression, only to the input-output pair. Therefore, the only way to learn it, it is by querying a certain input and obtaining a noisy output. When we query a certain position, the grasp quality is calculated based on tactile forces resultant from the contacts between the hand and the object. Thus, the goal is to learn and optimize the function to find the configuration that maximizes the grasp quality.

Bayesian optimization (BO) is a technique used for the efficient optimization of expensive black-box functions. BO works by fitting a model to black-box function data and then using the model's predictions to decide where to collect data next, so that the optimization problem can be solved using only a small number of function evaluations. It is characterized by its high sample-efficiency when compared to alternative black-box optimization algorithms, enabling us to find the solution of new challenging problems.

1.2 Objectives

This dissertation aims to tackle the problem of grasping optimization for an anthropomorphic hand. Starting with some knowledge about an arbitrary object, our intention is to allow the robot to find the best pose that can grasp that object. We propose to follow a trial-and-error approach based on contact information, while using a BO framework to find such grasps by an efficient exploration that will minimize the number of required explorative actions.

In the end, we expect to have a method that can lead the robot to grasp arbitrary objects safely, *i.e.*, find a grasp configuration that is robust to execution noise and sensory limitations; with just a small set of exploration trials.

1.3 State-of-the-Art

The optimum grasping pose, which is the end-effector pose that yields the best grasp quality, is usually based on the available information about the target object. According to Bohg et al [4], the object can be inserted into one of three categories: known, familiar or unknown object. On the first category, accurate models of the object are available and can be exploited transforming a grasping problem into a pose estimation one [12, 13, 14], since the optimum grasp could be learned *a priori* and retrieved from a database of possible grasping poses.

On the second case, the object shares some features (*e.g.*, visual or 3D features) with previous known objects and it can fall back on the previous learned models adjusting the final target pose with online learning methods [15]. Finally, when grasping unknown objects we should resort to real-time sensing (*e.g.*, partial point clouds) using exploration for retrieving the optimum grasp pose. However, and even in the case of known objects, it is not always possible to achieve a robust grasp since small errors on the pose estimation algorithm or during motor execution may turn optimal grasps into bad grasps. Indeed, many object grasping controllers rely on a complete open-loop approach to achieve the pre-computed optimal grasp [16, 17, 18, 19, 20], where the robot looks once to the scene, computes the

pose of the object, and then drives the arm to the grasping pose without using any sensory feedback during the whole process. However, due to modeling inaccuracies and sensing limitations, such open-loop approach might not permit robust grasping performance. As a matter of fact, according to Bohg et al. [4], very few grasping pipelines exploit tactile or visual feedback to achieve a robust grasp execution. The robustness consists in: i) accuracy, *i.e.*, how far is the selected grasp from the optimal grasp for an object and ii) precision or repeatability, *i.e.*, how precisely a desired grasp can be executed; the latter can be seen as uncertainty in the input space of the robotic platform.

In general, some form of local adjustment or exploration is often needed to refine the desired grasps and to successfully apply them to the objects in the real world. For instance, the task-space placement errors can be mitigated by means of visual servoing [21], by closing the control-loop exploiting force and torque sensors [22] or by learning the optimum grasping position with a trial-and-error approach [23].

BO has been employed before to guide haptic exploration in robot grasping [10, 11]. The unscented Bayesian optimization (UBO) algorithm introduced in [11], is a variation of the classical BO, which finds a safe optimum. This approach was tested to search for safe robot grasps in a 2D search-space. The concept of a safe grasp was defined as a grasp where the presence of noise during the execution of a task won't result in a dramatic decrease in its quality. Thus, this approach assures that our selected optimal grasp can be once more executed and yield a similar grasp quality.

When evaluating the grasp stability there are several metrics that have been presented [24]. Namely, the Largest minimum Resisted Wrench (LRW) metric is widely implemented [25] and computes the magnitude of the largest disturbance that can be resisted by a grasp in any direction. More recently the Bimodal Wrench Space Analysis Metric [10] was introduced, this metric provides information about stable grasps but also provides information about non-stable grasps. It combines two evaluation modes: the first assesses the quality of stable grasps using LRW; the second calculates how close to being stable the grasp is.

During the optimization there are instances where the function can not be evaluated because of collisions. These can be interpreted as hidden constraints which are not part of the problem specification/formulation, and their manifestation comes in the form of some indication that the objective function could not be evaluated [26]. The introduction of a penalty function is a simpler approach that makes the problem unconstrained but forces the optimization to the feasible region. This approach has explored for a long time and a complete review can be found in [27]. These functions assume the constraints are known *a priori* and add a penalization according to the severity of the constraint violation. There has been some work with black-box function constraints (*i.e.* unknown constraints) in Bayesian optimization [28], using surrogate models to represent them. However, this approach can be very time consuming during the optimization of the acquisition function [29].

1.4 Contributions

This dissertation provides the following contributions:

- **Generalization to a 3D search-space optimization problem** : following the work done by Nogueira

et al. [11], we scale the previously introduced UBO in 2D search-space to a 3D grasping problem. We prove that better optimum grasps can be found compared to the 2D case, confirming also the better performance of the UBO against the BO.

- **Introduction of the collision penalty** : we consider that the problem is in fact a constrained optimization problem because of the existence of hidden constraints unspecified in the problem formulation, *i.e.*, the collisions during exploration, which significantly hinder the convergence speed of the optimization. The introduction of a Collision penalty (CP) helps tackling that problem, increasing the convergence speed and proves to be vital for the superior outcome of the 3D UBO. This different approach to deal with occurring collisions during the exploration is to our knowledge a novel approach to the BO problem in the grasping domain.
- **Paper accepted for the IROS conference**: the work developed in this dissertation also led to the acceptance of an article for the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems.
- **Implementation of UBO in the BayesOpt library** : the code from the UBO implementation was integrated in the Bayesian optimization framework BayesOpt.

1.5 Thesis Outline

The remainder of the dissertation aims to give the reader a background overview of the grasping optimization problem, how it was implemented and present the results and conclusions of the experiences performed. Thus, the document is structured as follows:

- Chapter 2, introduces the fundamentals of robotic grasping, presenting the contact model employed in this work and how of grasp quality metric is calculated from these contacts;
- Chapter 3, introduces the BO framework, the UBO variation and the CP heuristic;
- Chapter **??**, describes the experimental setup of our experiences;
- Chapter 4, presents and analyses our experimental results on grasping simulations of test objects;
- Chapter 5, we draw our conclusions and sketch future work.

2 Robotic Grasp

Contents

2.1 Wrenches 8
2.2 Contact Model 8
2.3 Grasp Representation 11
2.4 Grasp stability 12
2.5 Grasp Quality Metric 12

2.1 Wrenches

A grasp is commonly defined as a set of contacts on the surface of the object, with the purpose to constrain the potential movements of the object in the event of external disturbances. Each of these contacts exerts a linear system of forces, which can be replaced by a single force applied along a line, $\boldsymbol{f}$, combined with a torque about that same line, $\boldsymbol{\tau}$. Such a force is referred to as a wrench [30].

$$\boldsymbol{w} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{\tau} \end{bmatrix} \qquad \begin{matrix} \boldsymbol{f} \in \mathbb{R}^3 \\ \boldsymbol{\tau} \in \mathbb{R}^3 \end{matrix} \tag{2.1}$$

The values of the wrench vector $\boldsymbol{w} \in \mathbb{R}^6$ depend on the coordinate frame in which the force and moment are represented. If B is a coordinate frame attached to a rigid body, then we write $\boldsymbol{w}_b = (\boldsymbol{f}_b, \boldsymbol{\tau}_b)$ for a wrench applied at the origin of B, with $\boldsymbol{f}_b$ and $\boldsymbol{\tau}_b$ specified with respect to the B coordinate frame. If there are multiple contacts acting on an object, the total set of wrenches $\boldsymbol{w}_o$ that can be transmitted to the object through the n_c contacts is the linear combination of all individual wrenches:

$$\boldsymbol{w}_o = \sum_{i=1}^{n_c} \boldsymbol{w}_i \tag{2.2}$$

However, this only makes sense if every individual wrench is written in respect to the same coordinate frame, therefore all wrenches must be rewritten to a single coordinate frame before their sum. If B and A are two different coordinate frames, the transformation of a wrench $\boldsymbol{w}_b$ applied at the origin of B to the equivalent $\boldsymbol{w}_a$ applied at the origin of A is given by

$$\boldsymbol{w}_a = \begin{bmatrix} \mathbf{R}_{ab} & 0 \\ [\mathbf{p}_{ba}]_\times \mathbf{R}_{ab} & \mathbf{R}_{ab} \end{bmatrix} \boldsymbol{w}_b \tag{2.3}$$

where $\mathbf{R}_{ab}$ is the rotation matrix from coordinate frame B to frame A. The position vector $\mathbf{p}_{ba} \in \mathbb{R}^3$ represents the position of the origin of frame B seen from the origin of frame A and its screw-symmetric matrix $[\mathbf{p}_{ba}]_\times$ is defined by

$$[\mathbf{p}_{ba}]_\times = \begin{bmatrix} 0 & -p_3 & p_2 \\ p_3 & 0 & -p_1 \\ -p_2 & p_1 & 0 \end{bmatrix} \tag{2.4}$$

where p_1, p_2 and p_3 are the components of vector $\mathbf{p}_{ba}$. This transformation of frames includes an additional torque of the form $\mathbf{p}_{ba} \times \boldsymbol{f}_b$ which is the torque generated by applying a force $\boldsymbol{f}_b$ at $\mathbf{p}_{ba}$.

2.2 Contact Model

The question now arises, how can we mathematically describe a contact using wrenches as building blocks. We need a model that maps the forces exerted by the finger at each contact point to the resultant wrenches at some reference point in the object.

This mapping is determined by the geometry of the contacting surfaces and the material properties

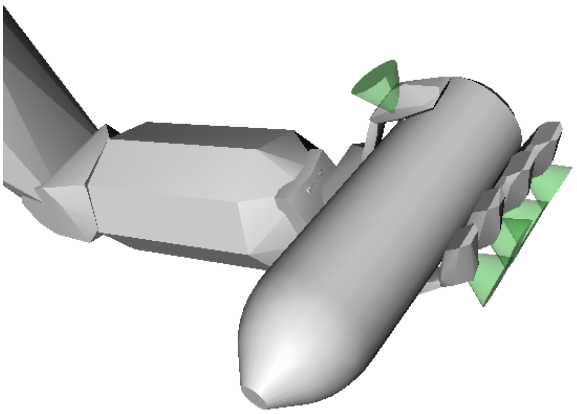


Figure 2.1: Contacts and friction cones

of the objects, which dictate friction and possible contact deformation [30]. As the object's reference point we use its center of mass, O. The forces at the contacts and on the object are represented in terms of a set of coordinate frames, C_i, attached to each contact location $\mathbf{p}_{c_i}$. It is assumed that the location of the contact point on the object is fixed. The coordinate frame C_i is chosen such that its z-axis points in the direction of the surface normal at the point of contact. The force applied by a contact is modeled as a wrench $\boldsymbol{w}_{c_i}$ applied at the origin of the contact frame, C_i.

The simplest representation is the frictionless point contact model, which considers that the finger can only transmit forces along the normal of the object's surface at the contact point. Thus, the applied wrench is defined as

$$\boldsymbol{w}_{c_i} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} f_{c_i}, \qquad f_{c_i} \geq 0 \tag{2.5}$$

where $f_{c_i} \in \mathbb{R}$ is the magnitude of the applied normal force. This model admits no deformations are allowed between the two rigid bodies, therefore the contact force results from the constraints of incompressibility and impenetrability. These assumptions are suitable for a situation where the contact patch is very small and the surfaces of the hand and object are slippery. Even though the frictionless point contact model might look like an attractive option because of its simplicity, it is a very basic assessment of reality. In most practical cases, friction is of significant importance, ignoring it will lead to a misrepresentation of reality.

A point contact with friction admits there can be forces in both normal and tangential directions to the object's surface. It is able to describe how much force a contact can apply in the tangent directions to a surface as a function of the applied normal force. However, it assumes that the contact patch is too small for a friction moment to exist about the normal direction. The classical friction model is the Coulomb model, it asserts that the allowed tangential force, f_t, is proportional to the applied normal

force, f_n, by

$$|f_t| \leq \mu_f |f_n| \tag{2.6}$$

where μ_f is the static coefficient of friction between the two contacting materials. This implies that in case this condition is not respected, where the tangential forces are actually larger, then there will be slippage. The geometric interpretation of this condition is that any applied force has to lie inside a cone, called friction cone, centered about the surface normal at the point of contact, Fig. 2.1. Using this friction model, the derived wrench at the contact point is

$$\boldsymbol{w}_{c_i} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{\boldsymbol{B}_{c_i}} \boldsymbol{f}_{c_i}, \qquad \boldsymbol{f}_{c_i} \in FC_{c_i} \tag{2.7}$$

where the friction cone FC_{c_i} is

$$FC_{c_i} = \{\boldsymbol{f}_{c_i} \in \mathbb{R}^3 : \sqrt{f_1^2 + f_2^2} \leq \mu_f f_3,\ f_3 \geq 0\}. \tag{2.8}$$

The components f_1,f_2 and f_3 are the force components along the x, y and z coordinate axis respectively. The matrix $\boldsymbol{B}_{c_i}$ is often described as the wrench basis. Even though there are other more complex contact models as described in [30], this is the model implemented in our simulator, thus the one employed during our experiments.

In practice the friction cone is discretized to a k-sided pyramid, so it can be described by a finite set of vectors. One other hypothesis that is often assumed is that the individual finger force $\boldsymbol{f}_c$ has a unit upper bound [25]. Thus, every contact force $\boldsymbol{f}_{c_i}$ applied by the hand is in the friction cone and it can be expressed as a positive linear combination of forces $\boldsymbol{f}_{c_{ij}}, j = 1, ..., k$ (usually called primitive forces) along the pyramid edges, equally spaced around the boundary of the cone, as displayed in Fig. 2.2. This means $\boldsymbol{f}_{c_i}$ can be rewritten as

$$\boldsymbol{f}_{c_i} = \sum_{j=1}^{k} \alpha_{ij} \boldsymbol{f}_{c_{ij}} \tag{2.9}$$

where $\alpha_{ij} \geq 0, \sum_{j=1}^{k} \alpha_{ij} \leq 1$.

Similarly, the wrench of each contact is actually the combination of k so-called primitive wrenches, which can be transformed from the force vectors by

$$\begin{aligned} \boldsymbol{w}_{c_i} &= \sum_{i=1}^{k} \alpha_{ij} \begin{bmatrix} \boldsymbol{f}_{c_{ij}} \\ (\boldsymbol{d}_{c_i} \times \boldsymbol{f}_{c_{ij}}) \end{bmatrix} \\ &= \sum_{i=1}^{k} \alpha_{ij} \boldsymbol{w}_{c_{ij}} \end{aligned} \tag{2.10}$$

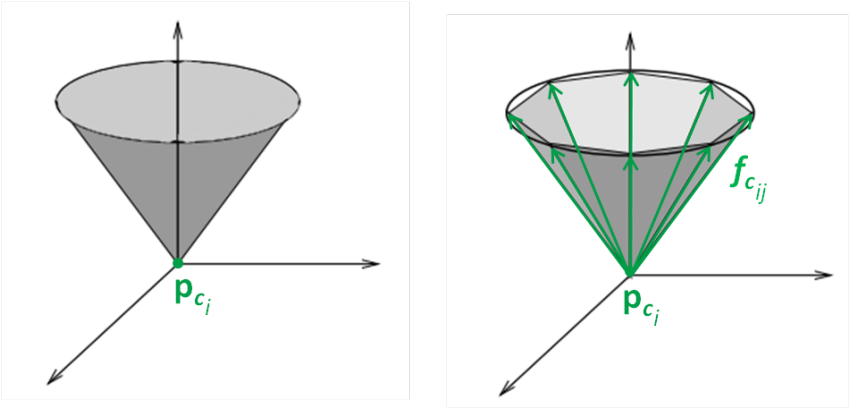


Figure 2.2: Friction cone approximation

where $\boldsymbol{w}_{c_{ij}}$ is one of the boundary wrenches of contact point $\mathbf{p}_{c_i}$ and $\boldsymbol{d}_{c_i}$ is the distance vector from the center of mass of the object (O) to contact point $\mathbf{p}_{c_i}$.

2.3 Grasp Representation

Depending on the number of contacts n_c, there is the equivalent number of set of wrenches which are represented in order to the correspondent contact reference frame C_i. Before we can add them to get the total wrench applied on the object, they all need to be transformed to the object's reference frame, set to its center of mass and previously defined as O. Hence, we define the partial grasp map, which linearly maps the contact forces from each contact c_i with respect to the $\boldsymbol{B}_{c_i}$, to the object's reference frame O, as

$$\mathbf{G}_i = \begin{bmatrix} \boldsymbol{R}_{c_i} & 0 \\ [\mathbf{p}_{c_i}]_\times & \boldsymbol{R}_{c_i} \end{bmatrix} \boldsymbol{B}_{c_i}, \quad i \in [1, ..n_c] \tag{2.11}$$

where $\boldsymbol{R}_{c_i}$ is the rotation matrix from the contact reference frame to the the object's reference frame.

After all wrenches are transformed to the object reference frame, the object net wrench can be defined as a linear combination of the partial grasps for each of the n_c contact points. The total object wrench is given by

$$\begin{aligned} \boldsymbol{w}_o &= [\mathbf{G}_1, \ \dots \ , \mathbf{G}_{n_c}] \begin{bmatrix} \boldsymbol{f}_{c_1} \\ \vdots \\ \boldsymbol{f}_{c_{n_c}} \end{bmatrix}, \qquad \boldsymbol{f}_{c_i} \in FC_{c_i}. \\ &= \mathbf{G}\boldsymbol{f}_c \end{aligned} \tag{2.12}$$

where $\mathbf{G}$ is the map between the contact forces and the total object force, which is called the grasp map. Thus, a grasp can simply be described by its grasp map $\mathbf{G}$ and its friction cone FC, even though the latter is omitted most of the time.

2.4 Grasp stability

To assess the stability of grasps it is often considered how it behaves under exterior disturbances, thus most quality metrics are based around the grasp resistance to exterior forces. Force-closure provides a binary qualitative analysis of grasp safety, where a grasp is said to be in force-closure if the fingers can apply, with their set of contacts, arbitrary wrenches on the object, assuring that any motion of the object is resisted by the contact forces [31]. From the previously described definition force-closure is mathematically described as

$$\mathbf{G}\boldsymbol{f}_c = -\boldsymbol{w}_e \tag{2.13}$$

where $\boldsymbol{w}_e \in \mathbb{R}^6$ represents an external wrench.

A simple way to evaluate if a grasp $\mathbf{G}$ is force-closure is through the analysis of the Grasp Wrench Space (GWS). The GWS is the minimum convex region spanned by $\mathbf{G}$ in the wrench space W. It can be constructed by generating a convex hull of the Minkowski sum of all the primitive wrenches $\boldsymbol{w}_{c_{ij}}$ at all contact points as

$$W_{\text{gws}} = ConvexHull\left(\bigoplus_{i=1}^{n_c}\{\boldsymbol{w}_{i1},\ \ldots\ \boldsymbol{w}_{ik}\}\right) \tag{2.14}$$

In order for the grasp to be force-closure, the origin of the wrench space W has to be inside the convex hull W_{gws}. Knowing if a grasp is force-closure provides a qualitative measure of stability, yet in order to analyze the quality of a stable grasp, we need a quantitative quality metric.

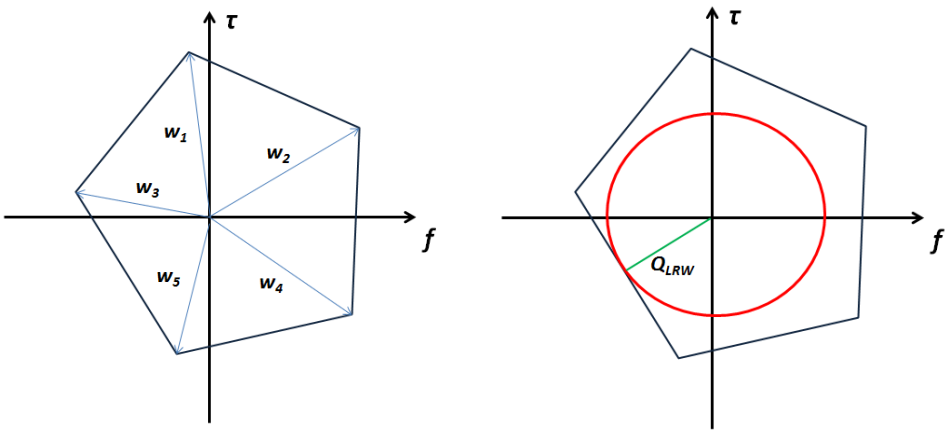


Figure 2.3: On the left, the convex hull of the grasp wrenches known as the GWS (simplified example). On the right, the Largest minimum Resisted Wrench metric.

2.5 Grasp Quality Metric

There are several grasp quality metrics related to the position of contact points on the object, some are solely based on the algebraic properties of the grasp map $\mathbf{G}$, others on geometrical relations, but the in our case we are interested in the ones that analyze the GWS [24]. By analyzing W_{gws} they get a quantitative quality measure Q.

The LRW or epsilon-quality, is widely implemented and defines the quality as the largest perturbation wrench that the grasp can resist in any direction *i.e.*, the distance to the nearest facet of the convex hull from the origin [25]. It is given by

$$Q_{LRW} = \min_{\boldsymbol{w} \in \partial W_{\text{gws}}} \|\boldsymbol{w}\| \tag{2.15}$$

where ∂W_{gws} denotes the boundary of W_{gws}. Geometrically, it can be interpreted as the largest radius sphere centered on the origin of W that is fully contained in the convex hull W_{gws}, as seen in Figure 2.3.

3 Optimization

Contents

3.1 Bayesian optimization 15

3.1.1 Gaussian Process 15

3.1.2 Learning hyperparameters 17

3.1.3 Surrogate model estimations 17

3.1.4 Decision using acquisition function 19

3.2 Unscented Bayesian optimization 20

3.2.1 Unscented expected improvement 20

3.2.2 Unscented optimal incumbent 21

3.3 DIRECT algorithm 22

3.4 Grasp position optimization problem 23

3.4.1 Collision Penalty 24

3.1 Bayesian optimization

Our objective is to find the grasp position with the best quality. However, we have no previous knowledge on our objective function $f()$, which relates a position to its grasp quality, and we can not observe $f()$ directly. This description fits the case of a global optimization problem where our objective function is a black box function, *i.e.*, we do not have its expression and we do not know its derivatives [32]. Therefore, the only way to evaluate the function is by querying a position and obtaining an noisy observation. We employ BO to guide the haptic exploration, so that after each evaluation we can decrease the distance between our estimated global maximum and the true global maximum. The Bayesian approach uses the memory of all previous observations to decide on the next point to sample, which allows for a more efficient search strategy that requires a lower number of iterations when compared to other nonlinear optimization algorithms.

The Bayesian optimization algorithm consists of two stages. First, the update of the probabilistic surrogate model, a distribution over the family of functions $P(f)$, where the target function $f()$ belongs. A very popular choice for this model is a Gaussian Process (GP). The GP captures our updated beliefs about the unknown target function and is built incrementally sampling over the input-space, therefore providing a better estimation of $f()$. Second, a Bayesian decision process, where an acquisition function uses the information gathered in the GP to decide on the best point (*i.e.*, a sample) to query next. The goal is to guide the search to the optimum, while balancing the trade-off: exploration *vs* exploitation.

In the following sections, we follow the Bayesian optimization notation presented in [11, 32, 33].

3.1.1 Gaussian Process

A GP provides a way to represent known information and updates our current knowledge with every new observation. In the case of robot grasping, the robot is trying to learn how to grasp through a black-box function. The GP can be used as a surrogate model in a BO framework to provide an estimation of the objective function, by representing this function based on its uncertainty and known values.

GPs are a state-of-the-art probabilistic non-parametric regression method [33]. A Gaussian is used as a means to describe a distribution over functions. As a more formal definition, a GP is a collection of random variables, any finite number of which have a joint Gaussian distribution. It is completely defined by a mean and covariance function pair where

$$\mu(x) = \mathbb{E}[f(x)] \tag{3.1}$$

$$k(x, x') = \mathbb{E}[(f(x) - \mu(x))(f(x') - \mu(x'))] \tag{3.2}$$

therefore we can denote that our unknown function $f(x)$ can be approximately represent by

$$f(x) \sim \mathcal{GP}(\mu(x), k(x, x')) \tag{3.3}$$

It is more intuitive to think of a GP as analogous to a function but instead of returning a scalar $f(x)$

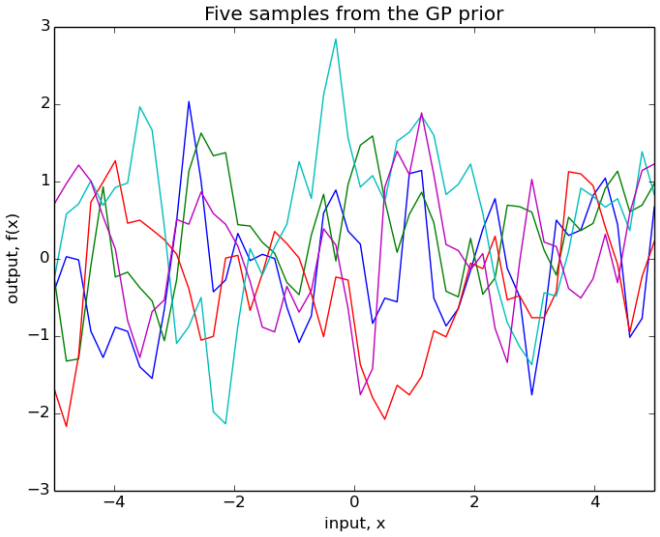


Figure 3.1: Five one-dimensional functions randomly sampled from a GP prior with kernel k_{M52}

for an arbitrary x, it returns the mean and variance of a normal distribution over the possible values of f at x [32].

In our context the random values represent values from the grasp quality metric function Q_{LRW} that the robot wishes to learn. Also, the mean function is initially considered as a zero function, since there is no relevant information at the start of the process.

The covariance function provides a measure of similarity or proximity between two points in the input parameter space. A pair of points close to each other must have a high covariance seeing as they have a larger influence on each other or are more similar. On the other hand, a low covariance is associated with unrelated points. There are various kernel functions that can be employed, thus it must be careful selected according to the problem in hands, since an ill chosen kernel can seriously hinder the performance of the BO algorithm. In this work, the chosen kernel function was the Automatic Relevance Determination (ARD) Matérn 5/2 [33], which uses independent parameters for every dimension of the problem. Its expression is given as follows

$$k_{M52}(\mathbf{x}, \mathbf{x}') = \theta_0 \left(1 + \sqrt{5r^2(\mathbf{x}, \mathbf{x}')} + \frac{5}{3}r^2(\mathbf{x}, \mathbf{x}')\right) exp\left\{-\sqrt{5r^2(\mathbf{x}, \mathbf{x}')}\right\} \tag{3.4}$$

where,

$$r^2(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{d} \frac{(\mathbf{x}_i - \mathbf{x}'_i)^2}{{l_i}^2} \tag{3.5}$$

Also, $\boldsymbol{\theta} = (\theta_0, l_i)$ are the so called hyperparameters. This kernel has $d + 1$ hyperparameters in d dimensions: an overall amplitude θ_0 and one characteristic length scale $l_{1:d}$ per dimension.

The covariance function is what implies the definition of distribution over functions. Therefore, with a GP prior one can draw several randomly generated sample functions at a number of test points. A demo result can be seen in Fig. 3.1, this is not conditioned on data and therefore represents our prior assumption of the function space from which the data may be generated. These random functions do not provide any insight to the objective without having observations.

One advantage of using GPs as a prior is that new observations of the target function (x_i, y_i) can be easily used to update the distribution over functions. Furthermore, the posterior distribution, conditioned on previous observations is also a GP, whose mean function will provide a much better approximation of $f()$.

3.1.2 Learning hyperparameters

The kernel described above in Equation (3.4) has hyperparameters $\boldsymbol{\theta}$. These hyperparameters greatly affect the behaviour of the GP, for instance, if the length scales are too large, then the GP prior will overlook the higher frequency variations in the true function; on the other hand, if the length scales are too small, the GP will fail to generalize across meaningful distances [33].

One way of choosing these hyperparameters is by fixing them *a priori* and keeping them unchanged as more data is acquired. However, this approach can lead to a very poor performance if the hyperparameters aren't suitable to the data. A more interesting approach is to learn the kernel hyperparameters from the data. This can be achieved by maximizing the marginal likelihood of the GP given the kernel hyperparameters. One can then perform maximum likelihood estimation by maximizing this quantity with respect to the hyperparameters $\boldsymbol{\theta}$. It is possible to analytically compute the gradient of the log-likelihood and perform gradient descent optimization to get to find its maximum.

An even more sophisticated approach is a fully Bayesian treatment of the kernel hyperparameters. This is achieved by placing a prior on these hyperparameters and marginalizing them out. This marginalization can be performed using Markov chain Monte Carlo (MCMC) methods such as slice sampling [34]. The slice sampling algorithm unlike other MCMC algorithms like the Hamiltonian Monte Carlo [35], is not dependent on other parameters. It does have a step size parameter but even a bad step size choice can be compensated by the algorithm at the cost of some extra computations. By using slice sampling on the posterior distribution of $\boldsymbol{\theta}$ given the data, we acquire a set of m different hyperparameters,

$$\boldsymbol{\Theta} \sim p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{y}). \tag{3.6}$$

3.1.3 Surrogate model estimations

Formally, the problem is based around finding the optimum (maximum) of an unknown real valued function $f : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is a compact space, $\mathcal{X} \subset \mathbb{R}^d$, and $d \geq 1$ its dimension, with a maximum budget of N evaluations of the target function $f()$. The Gaussian process $\mathcal{GP}(\mathbf{x}|\mu, \sigma^2, \boldsymbol{\theta})$ has inputs $\mathbf{x} \in \mathcal{X}$, scalar outputs $y \in \mathbb{R}$ and an associated kernel function $k(\cdot, \cdot)$ with hyperparameters $\boldsymbol{\theta}$ (as in Sec. 3.1.1). The hyperparameters are also optimized during the process (as in Sec. 3.1.2), resulting in m samples $\boldsymbol{\Theta} = [\boldsymbol{\theta}_i]_{i=1}^m$.

From the GP we can get an estimate, $\hat{y}()$, of our target function $f()$ based on known values. This can be achieved by a simple matter of conditioning distribution over functions to what is already known.

Assuming our optimization is at a step n, where we have a dataset of observations $\mathcal{D}_n = (\mathbf{X}, \mathbf{y})$, represented by all the queries until that step, $\mathbf{X} = (\mathbf{x}_{1:n})$, and their respective outcomes, $\mathbf{y} = (y_{1:n})$, then the prediction, $y_{n+1} = \hat{y}(\mathbf{x}_{n+1})$, at an arbitrary new query point $\mathbf{x}_{n+1}$, with kernel k_i conditioned on

the i-th hyperparameter sample $k_i = k(\cdot, \cdot|\boldsymbol{\theta}_i)$, is normally distributed and given by:

$$\hat{y}(\mathbf{x}_{n+1}) \sim \frac{1}{m}\sum_{i=1}^{m}\mathcal{N}(\mu_i(\mathbf{x}_{n+1}), \sigma_i^2(\mathbf{x}_{n+1})) \tag{3.7}$$

where

$$\begin{aligned}\mu_i(\mathbf{x}_{n+1}) &= \mathbf{k}_i^T\mathbf{K}_i^{-1}\mathbf{y} \\ \sigma_i^2(\mathbf{x}_{n+1}) &= k_i(\mathbf{x}_{n+1}, \mathbf{x}_{n+1}) - \mathbf{k}_i^T\mathbf{K}_i^{-1}\mathbf{k}_i.\end{aligned} \tag{3.8}$$

The vector $\mathbf{k}_i$ is one of the sample kernels evaluated at the arbitrary query point $\mathbf{x}_{n+1}$ with respect to the dataset $\mathbf{X}$,

$$\mathbf{k}_i = \begin{bmatrix} k_i(\mathbf{x}_{n+1}, \mathbf{x}_1) & k_i(\mathbf{x}_{n+1}, \mathbf{x}_2) & \cdots & k_i(\mathbf{x}_{n+1}, \mathbf{x}_n) \end{bmatrix} \tag{3.9}$$

and $\mathbf{K}_i$ is the Gram matrix corresponding to the self-correlation of dataset $\mathbf{X}$, with noise variance σ_n^2.

$$\mathbf{K}_i = \begin{bmatrix} k_i(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k_i(\mathbf{x}_1, \mathbf{x}_n) \\ \vdots & \ddots & \vdots \\ k_i(\mathbf{x}_n, \mathbf{x}_1) & \cdots & k_i(\mathbf{x}_n, \mathbf{x}_n) \end{bmatrix} + \mathbf{I} \cdot \sigma_n^2 \tag{3.10}$$

Note that, because we use a sampling distribution of $\boldsymbol{\theta}$, the predictive distribution at any point $\mathbf{x}$ is a mixture of Gaussians [33]. As we can observe in Fig. 3.2, conditioning the GP by the acquired observations allows to obtain better random samples from the GP posterior, thus the GP surrogate mean also provides a better good estimation of the objective function.

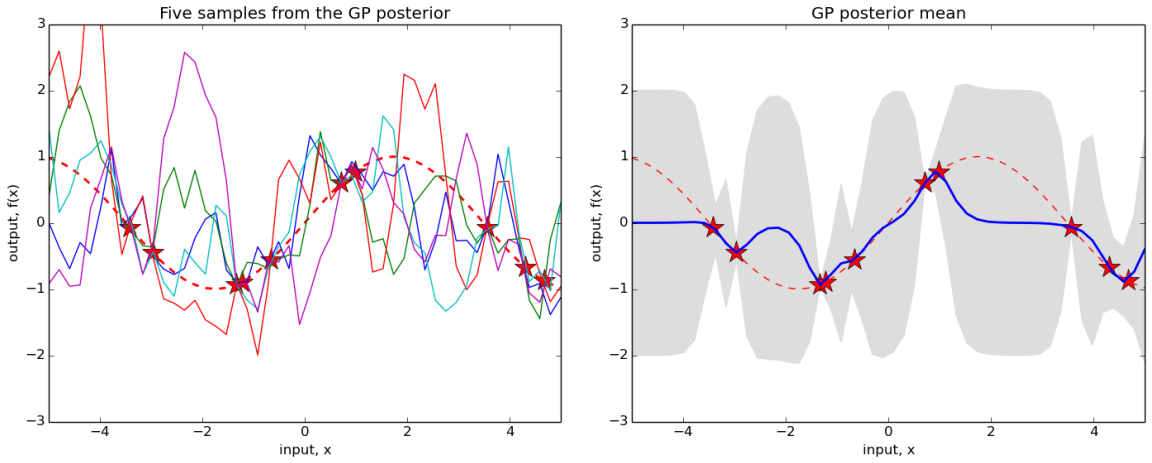


Figure 3.2: On the left, five one-dimensional functions randomly sampled from a GP posterior. On the right, the predictive approximation which is the GP posterior mean (solid blue line), given by Equation (3.8). The shaded area represents the pointwise mean plus and minus two times the standard deviation for each input value. In both plots, the real objective function is displayed with the dotted line and the stars are the observations already acquired.

3.1.4 Decision using acquisition function

To select the next query point at each iteration, we use the expected improvement criterion as the acquisition function. This function takes into consideration the predictive distribution for each point in $\mathcal{X}$, whose mean and variance are stated in equation (3.8), to decide the next query point. The expected improvement enables us to balance between exploration and exploitation. This dilemma is based around whether we should look to obtain a new sample in regions of the input-space where the surrogate mean is high, *i.e.*, exploiting known information about that region, or explore unknown regions where no previous evaluations where done and the surrogate variance is high [32].

The expected improvement is the expectation of the improvement function $I(\mathbf{x}) = \mathsf{max}(0, \hat{y}(\mathbf{x}) - \mathsf{y}_n^{bopt})$, where the incumbent is

$$\mathsf{y}_n^{bopt} = \mathsf{max}(y_{1:n}) \tag{3.11}$$

the best outcome found until now (iteration n). In other words, if the prediction for an input point is higher than the incumbent, then $I(\mathbf{x})$ gets a positive score, corresponding to the amount by which one expects to improve over the function value at the current best solution. Then, the optimum value corresponds to its associated query on the dataset and is denoted as $\mathbf{x}_n^{bopt}$.

Taking the expectation over the mixture of Gaussians of the predictive distribution [33], the expected improvement can be computed as

$$\begin{aligned} EI(\mathbf{x}) &= \mathbb{E}_{(\hat{y}|\mathcal{D}_n,\boldsymbol{\theta},\mathbf{x})}[\mathsf{max}(\mathbf{0}, \hat{y}(\mathbf{x}) - \mathsf{y}_n^{bopt})] \\ &= \begin{cases} \sum_{i=1}^{m}[(\mu_i(\mathbf{x}) - \mathsf{y}_n^{bopt} - \xi)\Phi(z_i) + \sigma_i(\mathbf{x})\phi(z_i)], & \text{if } \sigma_i(\mathbf{x}) > 0 \\ 0, & \text{if } \sigma_i(\mathbf{x}) = 0 \end{cases} \end{aligned} \tag{3.12}$$

where ϕ corresponds to the Gaussian probability density function (PDF), Φ to the cumulative density function (CDF) and

$$z_i = \begin{cases} \frac{\mu_i(\mathbf{x}) - \mathsf{y}_n^{bopt} - \xi}{\sigma_i(\mathbf{x})}, & \text{if } \sigma_i(\mathbf{x}) > 0 \\ 0, & \text{if } \sigma_i(\mathbf{x}) = 0 \end{cases} \tag{3.13}$$

The parameter $\xi \geq 0$ is what allows to regulate between exploration and exploitation. According to literature [32], if set to $\xi = 0.01$ we can get a good performance in most cases. Also, the pair (μ_i, σ_i^2) are the predictions computed in equation (3.8). Then, the new query point is selected by maximizing the expected improvement

$$\mathbf{x}_{n+1} = \underset{\mathbf{x} \in \mathcal{X}}{\operatorname{argmax}}\, EI(\mathbf{x}). \tag{3.14}$$

Lastly, in order to reduce initialization bias and improve global optimality, we rely on an initial design of p points based on Latin Hypercube Sampling (LHS).

3.2 Unscented Bayesian optimization

When determining the most interesting point to query at each iteration, acquisition functions, like the expected improvement criterion, make that selection assuming the query is deterministic [11]. However, when considering input noise our query is in fact a probability distribution, so the acquisition function should, for each specific query, also account for its uncertainty. Indeed, if taken into consideration the query's vicinity in input space, a better notion and estimation of the expected outcome can be achieved. The size of vicinity to be considered depends on the input noise estimation.

Thus, instead of analyzing the outcome of the expected improvement criterion to select the next query, we are going to analyze the posterior distribution that results from propagating the query distribution through the acquisition function.

The unscented transformation is a method used to propagate probability distributions through nonlinear functions with a trade-off between computational cost and accuracy [36]. The unscented transform uses selected samples from the input distribution designated sigma points, $\mathbf{x}^{(i)}$, and calculates the value of the nonlinear function $g()$ at each of these points. Subsequently, the transformed distribution is computed based on the weighted combination of the transformed sigma points.

For a d-dimensional input space, the unscented transformation only requires a set of $2d+1$ sigma points. If the the input distribution is a Gaussian, then the transformed distribution is simply

$$\mathbf{x}' \sim \mathcal{N}\left(\sum_{i=0}^{2d} w^{(i)} g(\mathbf{x}^{(i)}), \Sigma'_{\mathbf{x}}\right) \tag{3.15}$$

where $w^{(i)}$ is the weight corresponding to the i-sigma point.

The unscented transformation provides mean and covariance estimates of the new distribution that are accurate to the third order of the Taylor series expansions of $g()$ provided that the original distribution is a Gaussian prior. Another advantage of the unscented transformation is its computational cost. The $2d+1$ sigma points make the computational cost almost negligible compared to other alternatives to distribution approximation.

3.2.1 Unscented expected improvement

Considering that our prior distribution is a Gaussian distribution $\mathbf{x} \sim \mathcal{N}(\bar{\mathbf{x}}, \mathbf{I}\sigma_x)$, then the set of $2d+1$ sigma points of the unscented transform are computed as

$$\begin{aligned}
&\mathbf{x}^0 = \bar{\mathbf{x}} \\
&\mathbf{x}_+^{(i)} = \bar{\mathbf{x}} + \left(\sqrt{(d+k)\,\sigma_x}\right)_i \qquad \forall i = 1...d \\
&\mathbf{x}_-^{(i)} = \bar{\mathbf{x}} - \left(\sqrt{(d+k)\,\sigma_x}\right)_i \qquad \forall i = 1...d,
\end{aligned} \tag{3.16}$$

where $(\sqrt{(\cdot)})_i$ is the i-th row or column of the corresponding matrix square root. In this case, k is a free parameter that can be used to tune the scale of the sigma points. For more information on choosing the

optimal values for k, refer to [36]. For these sigma points, the corresponding weights are

$$\begin{aligned} w^0 &= \frac{k}{d+k} \\ w_+^{(i)} &= \frac{1}{2(d+k)} \qquad \forall i = 1...d \\ w_-^{(i)} &= \frac{1}{2(d+k)} \qquad \forall i = 1...d \end{aligned} \tag{3.17}$$

If we consider the expected improvement criterion as the nonlinear function $g()$, then we are making a decision on the next query considering that there is input noise. This new decision can be interpreted as a new acquisition function, the unscented expected improvement (UEI). It corresponds to the expected value of the transformed distribution and is defined by

$$UEI(\mathbf{x}) = \sum_{i=0}^{2d} w^{(i)} EI(\mathbf{x}^{(i)}), \qquad \mathbf{x} \in \mathcal{X}. \tag{3.18}$$

This strategy, by also evaluating the sigma points around the query, reduces the chance that the next query point is located in an unsafe region, *i.e.*, where a small change on the input (induced by noise) implies a bad outcome.

3.2.2 Unscented optimal incumbent

By employing the UEI we are driving the search towards safe regions, yet in BO the final decision for what we consider the optimum still does not depend on the acquisition function. We defined the incumbent for BO in Equation (3.11), as the best observation outcome until the current iteration. However, when using the UEI each query point is evaluated considering its small vicinity, so as we incrementally obtain more observations and get a better GP fit, we might observe that our optimum is actually inserted in a unsafe region. Therefore, based on the UEI criterion, the current incumbent would no longer be resistant to input noise and should be changed.

Thus, instead of considering the best observation outcome as the incumbent, we also apply the unscented transformation to select incumbent at each iteration, based on the outcome at the sigma points of each query that belongs to the dataset of observations ($\mathcal{D}_n$) . Obviously, we do not want to perform any additional evaluations of $f()$ because that would defeat the purpose of Bayesian optimization. Alternatively, we evaluate the sigma points with our estimation $\hat{y}()$, which is the GP surrogate average prediction $\mu()$. Therefore, we define the unscented outcome as:

$$u(\mathbf{x}) = \sum_{i=0}^{2d} w^{(i)} \sum_{j=1}^{m} \mu_j(\mathbf{x}^{(i)}), \qquad \mathbf{x} \in \mathbf{X} \tag{3.19}$$

where $\sum_{j=1}^{m} \mu_j(\mathbf{x}^{(i)})$ is the prediction of the GP according to equation (3.8) integrated over the kernel hyperparameters and at sigma points of equation (3.16). Under these conditions, the incumbent for the UBO is defined as:

$$\mathsf{y}_n^{\mathsf{ubopt}} = u(\mathbf{x}_n^{\mathsf{ubopt}}), \tag{3.20}$$

where $\mathbf{x}_n^{\mathsf{ubopt}} = \mathrm{argmax}_{\mathbf{x}_i \in \mathbf{x}_{1:n}} u(\mathbf{x}_i)$ is the optimal query until that iteration according to the unscented outcome. For further information on the performance of the UBO on synthetic functions, refer to [11].

3.3 DIRECT algorithm

The maximization of the Expected Improvement presented in Equation (3.14) is performed using the global optimization algorithm DIRECT [37, 38]. This is a derivative free method that represents an alternative to the usual gradient descent method. DIRECT works by iteratively dividing the search domain into hyper-rectangles and evaluating the unknown function at particular locations within the hyper-rectangles. It uses a small number of initial predictions to decide how to DIvide the feasible space into smaller RECTangles. The end result is a high discretization of the target function near the function minimum and a low discretization elsewhere.

The DIRECT algorithm starts by normalizing the function domain into a unit hyper-cube with center c_1. Thefore, the domain is

$$\bar{\mathbf{\Omega}} = \{\mathbf{x} \in \mathbb{R}^d : 0 \leq x_i \leq 1, \quad i = 1, ..., d\}. \tag{3.21}$$

After evaluating the function at $f(c_1)$ it makes the first division of the hyper-cube. The cube is divided into three smaller parts centered at $c_1 \pm \delta \mathbf{e}_i, i = 1, ..., d$, where δ is one third of the cube length and $\mathbf{e}_i$ is the i-th unit vector. DIRECT chooses to leave the best function values in the largest space. As such the first dimension to be divided is chosen by means of

$$w_i = \min(f(c_i + \delta \mathbf{e}_i), f(c_i - \delta \mathbf{e}_i)), \qquad 1 \leq i \leq d. \tag{3.22}$$

The dimension with the smallest w_i is divided into thirds and the process is repeated for all dimensions on the resulting center hyper-rectangles, evaluating the function at all the resulting center points. At this stage the initialization is concluded.

Afterwards, we need to identify which of those hyper-rectangles are potentially optimal. In order to do that, the following conditions are tested:

- if hyper-rectangle i is potentially optimal, then $f(c_i) \leq f(c_j)$ for all hyper-rectangles that are of the same size as i (*i.e.*, $d_i = d_j$);
- if hyper-rectangle i has the largest dimension (*i.e.*, $d_i \geq d_k, \forall k$), and $f(c_i) \leq f(c_j)$ for all hyper-rectangles such that $d_i = d_j$, then i is potentially optimal;
- if hyper-rectangle i has the smallest dimension (*i.e.*, $d_i \leq d_k, \forall k$), and i is potentially optimal, then the current minimum is $f(c_i) = f_{\mathsf{min}}$.

At each iteration, the function is evaluated at the set of potentially optimal values, *i.e.*, the center of the resulting hyper-rectangles, thus updating f_{min} and converging to a solution. From the previous

conditions, we conclude that the hyper-rectangles are divided further if they are deemed likely to contain the solution, or if they are large as stated by the second condition. These criteria allow the method to perform both local and global search.

The algorithm continues repeats the test until it can not find any more potentially optimal hyper-rectangles, *i.e.* until there are no more divisions of interest, or the optimization budget is complete. At the end, f_{min} is the global function minimum.

3.4 Grasp position optimization problem

In the grasping optimization context, the target function $f()$ is relates a given hand pose to its quality metric (as exemplified in Fig. 3.3), which should be maximized to achieve the optimal grasp.

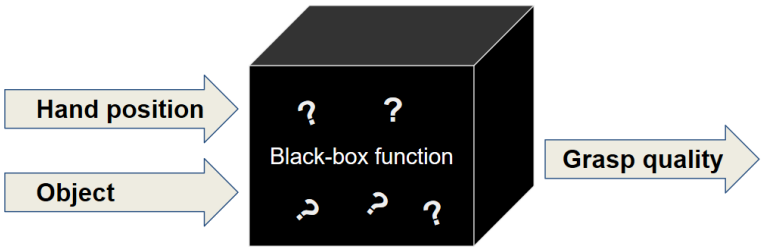


Figure 3.3: Finding the best quality grasp requires the maximization of an black-box function using a limited number of evaluations. The goal is to optimize the hand Cartesian position so that it maximizes the quality of the grasp.

In our work, as mentioned in Sec. 2.5, $f(\mathbf{x})$ is the LRW metric achieved according to Equation (2.15) when, starting the hand with an initial pose $\mathbf{x}$ (as in Fig.1.1), the fingers are closed until touching the object.

Thus, the knowledge of the black-box target function is incrementally built by querying different positions around the object and updating the GP according to the new observations. The process of the grasp optimization framework, as previously described, is presented in Fig. 3.4.

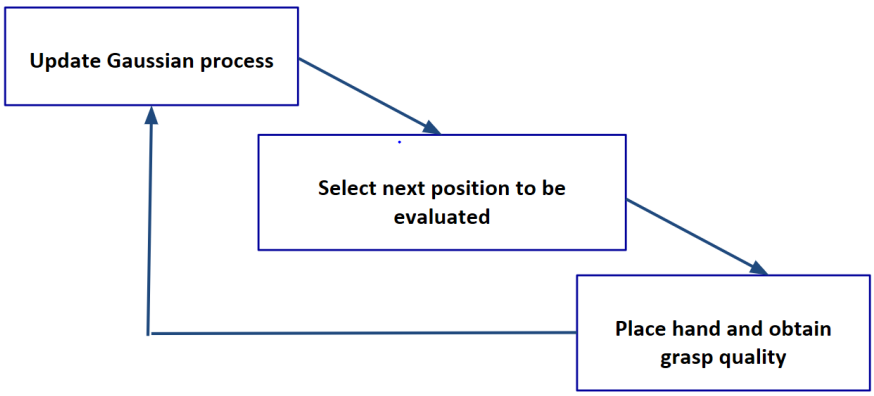


Figure 3.4: Grasp optimization process

3.4.1 Collision Penalty

We assume that the exploration is limited to a region next to the object, and we are therefore limiting the input space ($\mathcal{X}$) of our function $f()$. In practical applications, we may assume that approximate information about the object size and location is available, and such information can be used to limit the exploration space.

However, there are exploration queries that result in unfeasible grasps where the target pose of the robot's hand collides with the object, even before attempting to close the hand (one can see such examples in Fig. 3.5). This indicates that the problem has constraints. Although there has been some recent

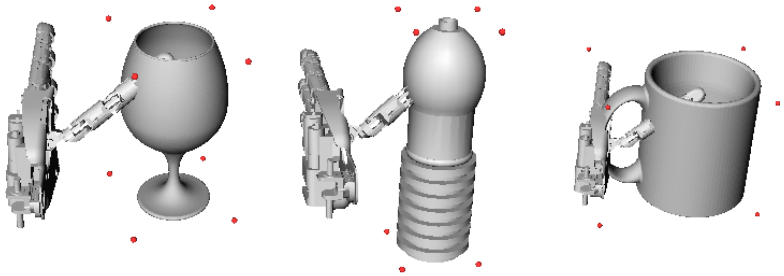


Figure 3.5: Collision examples

work on Bayesian optimization with constraints [39, 28], we opted for the simpler approach of adding a penalty as described in [29]. This approach means that the input space remains unconstrained, improving the performance of the acquisition function's optimization. Additionally, due to kernel smoothing, we also get a safety area around the collision query where the function is only partly penalized.

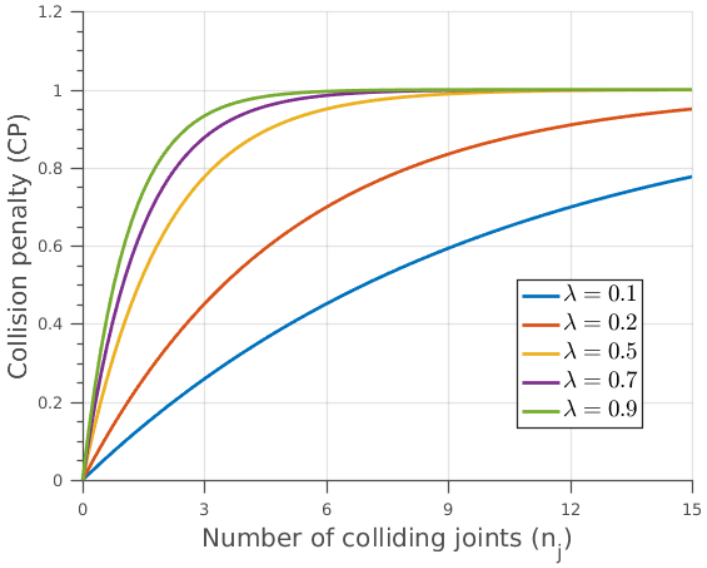


Figure 3.6: Influence of parameter λ on the Collision Penalty function

Other research works on robotic grasping optimization had different approaches to deal with collisions. Some would skip the collision query (*e.g.*, [40]) and others would give it a grasp quality of zero (*e.g.*, [11]). In the first case, by ignoring the query we are losing information about the target function,

hence not reducing uncertainty. In the second case, although uncertainty is reduced (*i.e.*, the query is incorporated in the function estimation), the zero value is usually associated to a position where the hand does not touch the object at all. Therefore, by modelling a collision in a similar way (with a zero value), one does not incorporate useful information for further queries since there is an arguably difference between the two situations. Moreover, collisions can throw down the object, changing its pose and damaging it, and can also harm robot's parts, thus these configurations should be avoided. In an ultimate analysis, the approaches presented so far slow down convergence to the optimum value, meaning that a larger budget for the optimization is needed.

Instead, we propose using an heuristic to improve the optimization convergence, which is based on the information retrieved from the collision. The heuristic indicates a regret or a penalization according to the level of penetration of the hand in the object. The penalization factor will drive the search away from collision locations, ensuing a reduction of explored area and consequently leading to faster convergence.

The CP is calculated by finding the number of joints in the robot's hand that collide with the object, $n_j \in \mathbb{N}$, which indicates a measure of penetration in the object. Therefore, we define the CP as follows:

$$\text{CP}(n_j) = 1 - \mathrm{e}^{-\lambda n_j}, \tag{3.23}$$

where λ is a tuning parameter used to smooth the penalty as shown in Fig. 3.6. As the value of λ increases, we get closer to a static penalty, losing the ability of penalizing a collision according to its penetration level. The penalty function was designed so that it's value is limited by 1, since the grasp quality is also normalized.

The CP is an heuristic used only to improve the convergence speed, therefore during the optimization we redefine the target function as $f' = f - CP$, however on the evaluation process we resort to the original f.

4 Experimental Setup and Results

Contents

4.1 Experimental Setup 28

4.1.1 Simox Simulator 28

4.1.2 BayesOpt 29

4.1.3 Experimental Design 29

4.2 Experimental Results 30

4.2.1 Collision Penalty tuning 30

4.2.2 Benefits of the Collision Penalty 31

4.2.3 Generalization to 3D 32

4.2.4 Advantages of UBO over BO 37

4.1 Experimental Setup

4.1.1 Simox Simulator

All the results were obtained from simulations using the Simox simulation toolbox [41, 42]. This toolbox allows to simulate the iCub's hand in a grasping task performed on arbitrary objects as seen in Fig. 4.1. By setting an initial pose for the hand and a motion trajectory for the finger joints, we can simulate a grasp movement. In particular, at the beginning of each optimization procedure, the left hand of the iCub is placed in an initial pose parallel to one of the object's facets with the thumb aligned with one of the neighbour facets. The facets where the hand is placed are chosen at the start of the simulation. This defines uniquely the pose of the hand with respect to the object. At each optimization step (i.e. each grasping attempt) the new hand pose is then defined with respect to the initial pose by incremental translations: $(\delta_x, \delta_y, \delta_z)$.

In these simulations we perform the optimization in either 2D or 3D search-space, only focusing on translation parameters; all other parameters (e.g. hand orientation, finger joints trajectories) are set in the initial pose and remain the same throughout the optimization. In 2D, we optimize (δ_x, δ_y), while δ_z is kept fixed. In 3D, we optimize $(\delta_x, \delta_y, \delta_z)$. For dimensions x and y, the exploration bounds are set to the object's dimensions, as for z, which is the approach direction, the bounds extend from the surface of the object's facet to the plane where the hand is no longer able to touch the object when it closes.

At each exploration (*i.e.*, optimization) step, the hand is placed in a new pose, and the fingers joints will move following predefined motion trajectories; the trajectories are set so that the hand performs a power grasp on the object, in which all fingers closed at the same time, following a movement synergy that has been defined in previous work [43]. Each finger stops when a local contact with the object is felt.

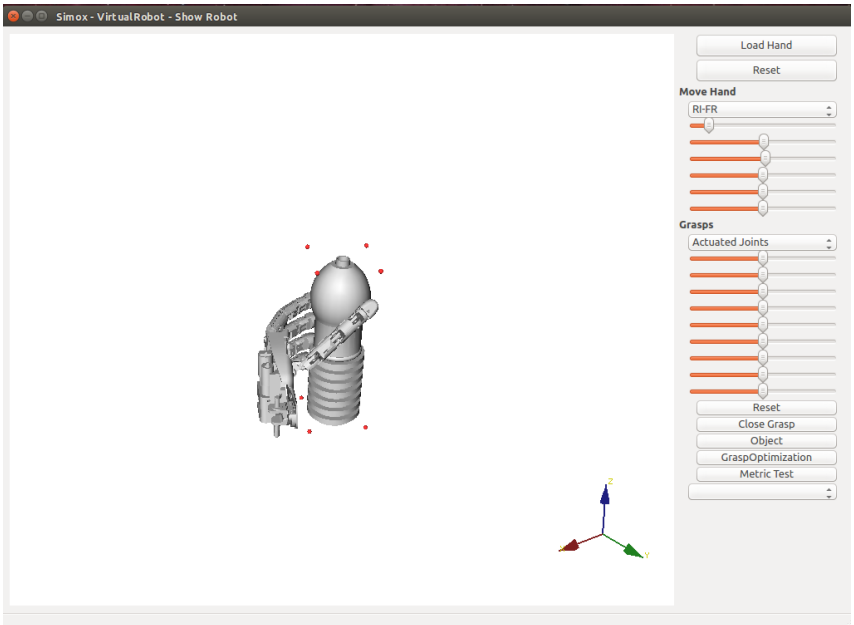


Figure 4.1: Simox simulation window

The robot in the simulator is equipped with 15 force sensors, 3 for each finger, meaning each phalanx is

equipped with a sensor. When the fingers motion is finished, the quality of the grasp is calculated based on the LRW defined in Sec. 2.5. However, if a collision between the hand (either palm or fingers) and the object is detected when positioning the hand to the new pose, the CP is applied, and the grasping motion will not be executed.

4.1.2 BayesOpt

The *BayesOpt* library [44] is used as the framework to perform both methods of Bayesian optimization (*i.e.*, both BO and UBO). It performs the rather well in terms of efficiency when compared to other popular Bayesian optimization software like SMAC or Spearmint [34], reducing the computation time. Additionally, it is very flexible allowing the user to easily tweak parameters, like surrogate models, kernels, acquisition functions, etc. It takes advantage of the structure of the Bayesian algorithm, where new information arrives sequentially to implement a more efficient calculation of matrix $\mathbf{K}$ from Equation (3.10) and its inverse $\mathbf{K}^{-1}$, which is a big reason behind the more efficient implementation. For the inner optimization tasks, like the maximization of the Expected improvement it uses the nonlinear optimization library NLopt [45].

Because this framework by default performs a minimization of the target function $f()$, we modified the problem formulation accordingly to solve our maximization problem; still, the results presented in Chapter 4 are consistent with a maximization problem.

4.1.3 Experimental Design

To reproduce the effect of the input noise, we obtain Monte Carlo samples at the optimum in each iteration, $\mathsf{y}_{\mathsf{mc}}(\mathbf{x}_n^{opt})$, according to the input noise distribution $\mathcal{N}(0, \mathbf{I}\sigma_{\mathbf{x}})$. Remember that, $\mathbf{x}_n^{opt}$ corresponds to $\mathbf{x}_n^{\mathsf{b}opt}$ when performing BO, and to $\mathbf{x}_n^{\mathsf{ub}opt}$ for the UBO strategy. By analyzing the outcome of the samples we can estimate the expected outcome from the current optimum $\bar{\mathsf{y}}_{\mathsf{mc}}(\mathbf{x}_n^{opt})$, and the variability of the outcomes $\mathrm{std}\Big(\mathsf{y}_{\mathsf{mc}}(\mathbf{x}_n^{opt})\Big)$. These metrics allow us to assess if the optimum belongs to a safe region. Indeed, if $\bar{\mathsf{y}}_{\mathsf{mc}}(\mathbf{x}_n^{opt})$ decreases over time/iterations (something which cannot occur in classical BO) it should be correlated with the fact that the optimum ($\mathbf{x}_n^{\mathsf{b}opt}$) is inside an unsafe area and is not a robust grasp.

The input noise at each query point is assumed to be white Gaussian, $\mathcal{N}(0, \mathbf{I}\sigma_{\mathbf{x}})$, with $\sigma_{\mathbf{x}} = 0.03$ (note that the input space was normalized in advance to the unit hypercube $[0, 1]^d$). We assume the grasp quality metric to be stochastic, due to small simulation errors and inconsistencies, thus we set $\sigma_n = 10^{-4}$.

For each experiment, we performed 20 runs of the robotic grasp simulation for all test objects. The robot hand posture for each object is initialized as shown in Fig. 1.1. Every time a new optimum is found we collect 10 Monte Carlo samples at its location to get $\mathsf{y}_{\mathsf{mc}}(\mathbf{x}_n^{opt})$. Each run starts with 20 initial iterations using LHS, followed by 140 iterations of optimization. The shaded region in each plot represents a 95% confidence interval. All the quantitative results from each experience, at its last iteration, are presented in Table 4.1.

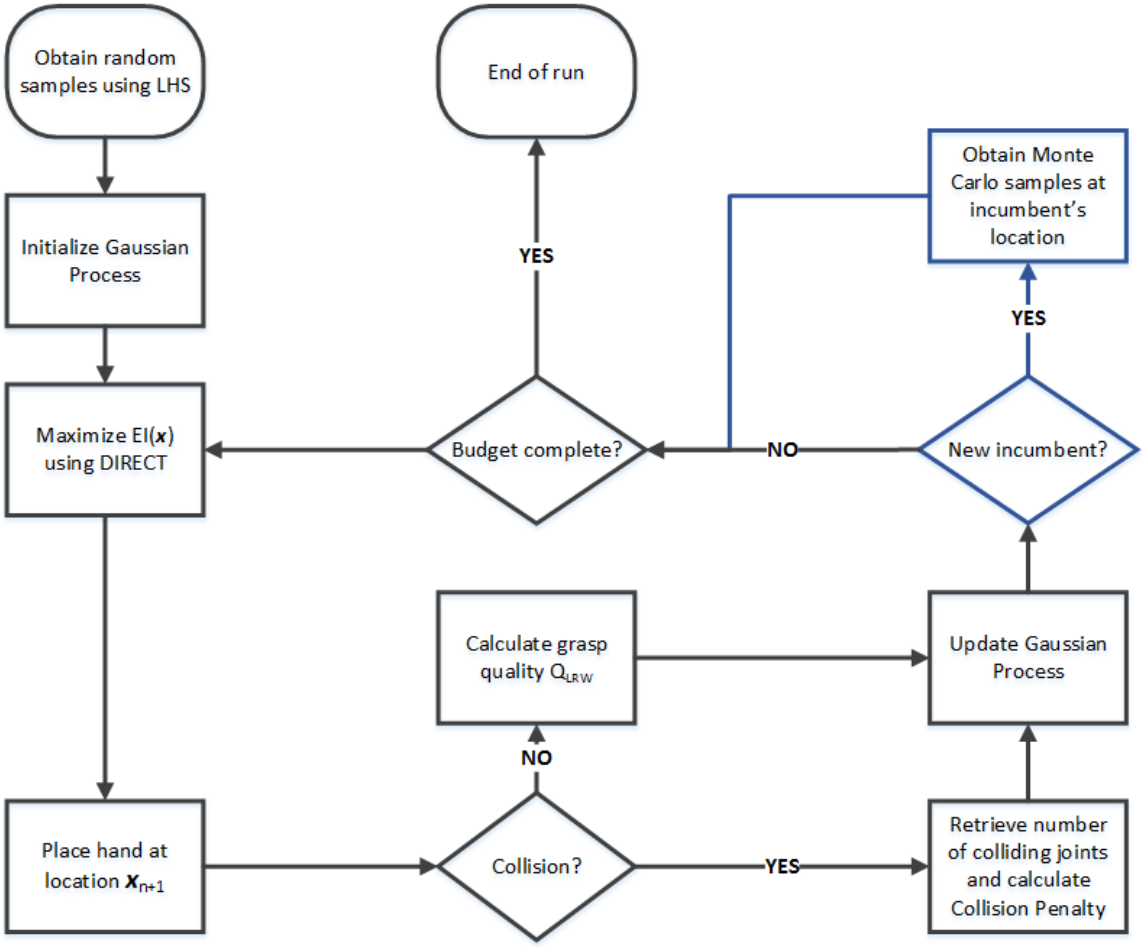


Figure 4.2: Flowchart of the program. The boxes in blue correspond to stages that are not necessary for the functioning of the system, since they only take part for its evaluation.

The flowchart of the program is displayed in Fig. 4.2.

4.2 Experimental Results

In this section, we start by tuning the CP so it produces the best possible results (Sec. 4.2.1). Then, we describe the experiments performed to evaluate the benefits of adding the CP into the optimization process (Sec. 4.2.2). Subsequently, we present the results of the UBO generalization to a 3D search-space and compare them to those obtained in the 2D case scenario (Sec. 4.2.3). Lastly, we investigate and corroborate the results achieved in [11] but on a 3D search-space, *i.e.*, we prove the UBO outperforms the classical BO in finding a safer grasp also in a higher dimension (Sec. 4.2.4).

4.2.1 Collision Penalty tuning

As introduced in Sec. 3.4.1, the collision penalty depends on a tuning parameter λ. A higher value for λ will result in a more aggressive penalty, while a lower one allows for a milder penalization. Even though we might intuitively assume that a more aggressive penalty will be more beneficial to reduce the search-space, that notion wasn't confirmed with the empirical results obtained.

The results displayed in Fig. 4.3, correspond to three different experiments on the mug with different λ values while adopting a 3D unscented Bayesian optimization with collision penalty (3D UBO CP) strategy. The experiment with $\lambda = 0.1$, managed to find the safest grasp and the results deteriorated as the value of λ increased. This is a consequence of the best grasps being most likely really close to the

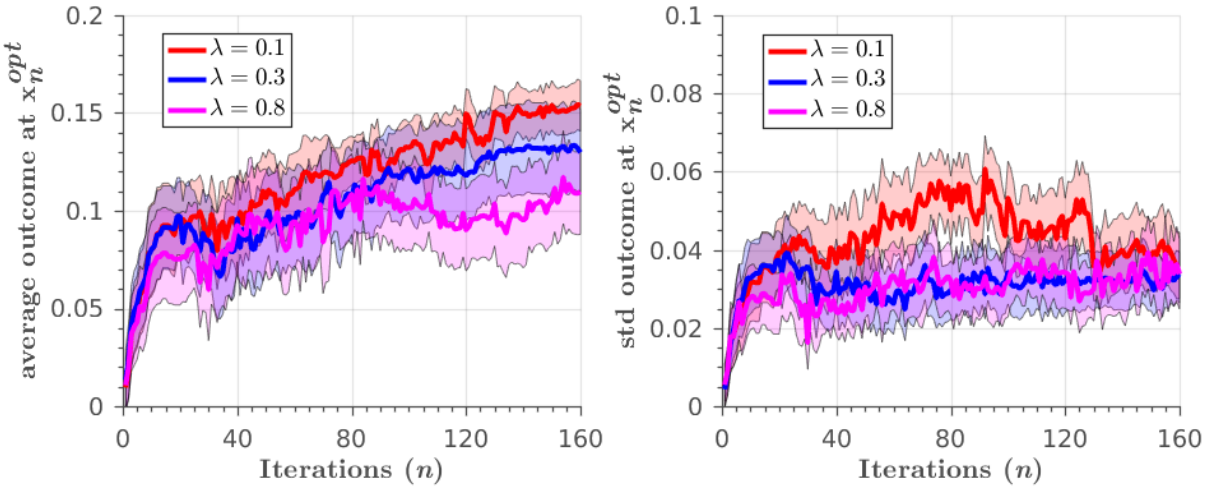


Figure 4.3: Performance of the 3D UBO CP on the Mug with different with different tuning parameters λ. Left: expected outcome of current optimum $\bar{y}_{mc}(\mathbf{x}_n^{ubopt})$, Right: Variability of the outcome std$\left(y_{mc}(\mathbf{x}_n^{ubopt})\right)$

boundary of the unfeasible region, *i.e.*, there is a collision configuration in the immediate vicinity of the best grasp. Hence, the higher penalization and sharp cliffs in the predictive function drove the search excessively away from the boundary region, consequently away from the best grasp location.

Based on these results, for the remaining experiments the tuning parameter will take the value of $\lambda = 0.1$.

4.2.2 Benefits of the Collision Penalty

To assess the benefits of CP, we performed two types of experiences for each object, a 3D UBO with and without CP. As we can see in Fig. 4.4 and 4.5, the addition of CP to the optimization process provides a boost in convergence speed for both the glass and the bottle. Also, by penalizing collisions we are reducing the regions that are worth exploring, meaning the robot is actually able to find a better grasp at the end, both in terms of mean and variance.

The mug is the most challenging object to learn a good grasp, since the optimization is performed on the mug's facet that includes the handle. In 3D, the handle is inside the search-space, leading to a large number of configurations that result in collisions, consequently undermining the convergence to the optimum. This is a situation where the CP really thrives. By penalizing these collisions, we are driving our search away from the inside of the handle and finding a safer grasp outside the handle. The results in Fig. 4.6 show how dramatic the improvement is, obtaining at the end of the process results with CP that are 50% better in terms of mean and also achieving those higher mean values with great confidence level (*i.e.*, a smaller shaded region).

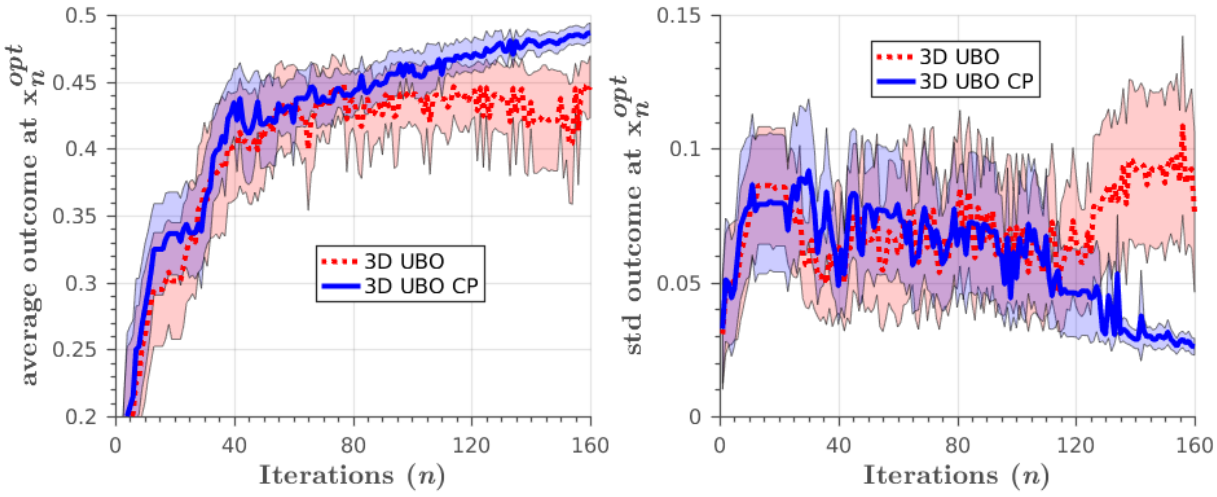


Figure 4.4: **Glass: CP vs $\overline{\text{CP}}$** (UBO 3D). Left: expected outcome of current optimum $\bar{y}_{mc}(\mathbf{x}_n^{ubopt})$, Right: Variability of the outcome std$\left(y_{mc}(\mathbf{x}_n^{ubopt})\right)$

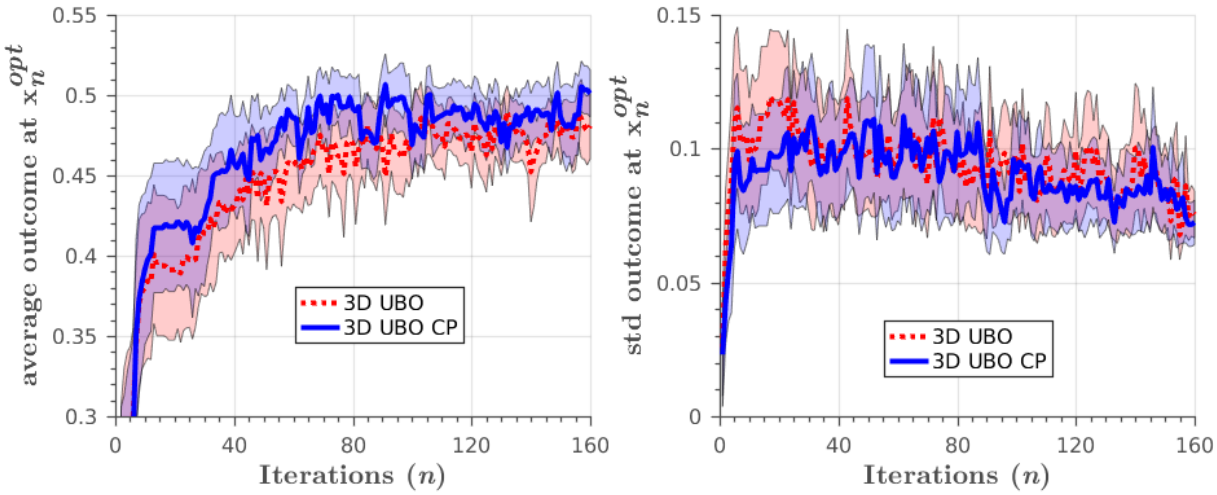


Figure 4.5: **Bottle: CP vs $\overline{\text{CP}}$** (UBO 3D). Left: expected outcome of current optimum $\bar{y}_{mc}(\mathbf{x}_n^{ubopt})$, Right: Variability of the outcome std$\left(y_{mc}(\mathbf{x}_n^{ubopt})\right)$

4.2.3 Generalization to 3D

We performed UBO with CP in both 2D and 3D to provide evidence that UBO generalizes well into a higher dimension space, *i.e.*, in 3D we only need a few extra evaluations to reach the same results obtained in 2D.

In Fig. 4.7, we observe that for the glass, even though 2D reaches better mean values right after the learning starts (iteration 20), 3D is able to reach the same level around iteration 40 and proceeds to surpass it achieving better results. As for the bottle, in Fig. 4.8, the mean value of the 3D case trails

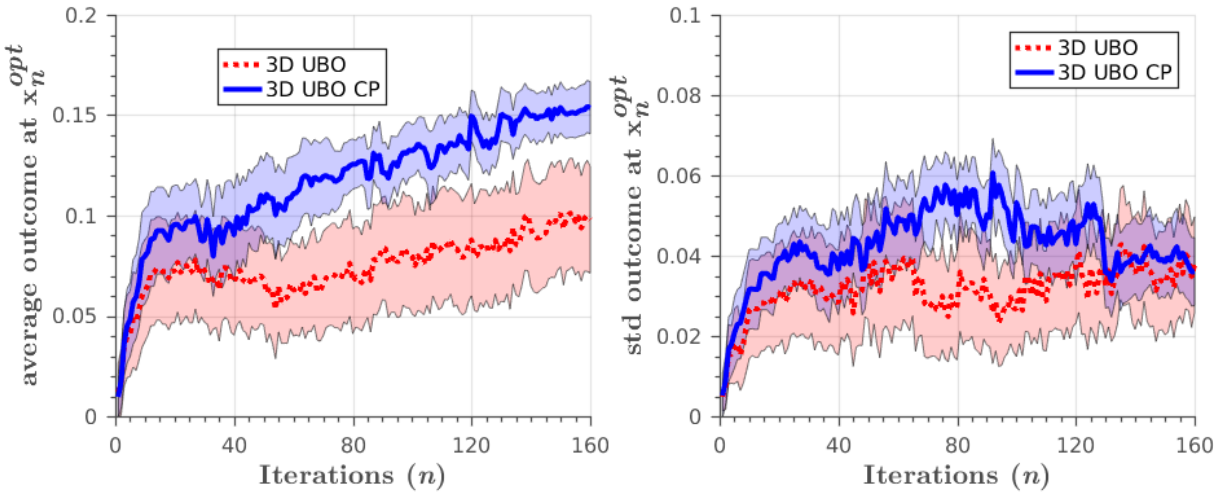


Figure 4.6: **Mug: CP vs $\overline{\text{CP}}$** (UBO 3D). Left: expected outcome of current optimum $\bar{y}_{mc}(x_n^{ubopt})$, Right: Variability of the outcome std$\left(y_{mc}(x_n^{ubopt})\right)$

almost the whole process only edging out the 2D results close to the end of the budget.

For the mug, Fig. 4.9, the 3D optimization only manages to reach similar mean values at around iteration 65. As explained in Sec. 4.2.2, this is due to the high amount of queries that result in collisions when we are optimizing in a 3D search-space. Still, it comes to show how the boost in convergence speed provided by the CP makes it possible to generalize the optimization to 3D, since without it the 3D exploration wouldn't even reach the 2D values within the budget, as we can conclude from the quantitative values in Table 4.1.

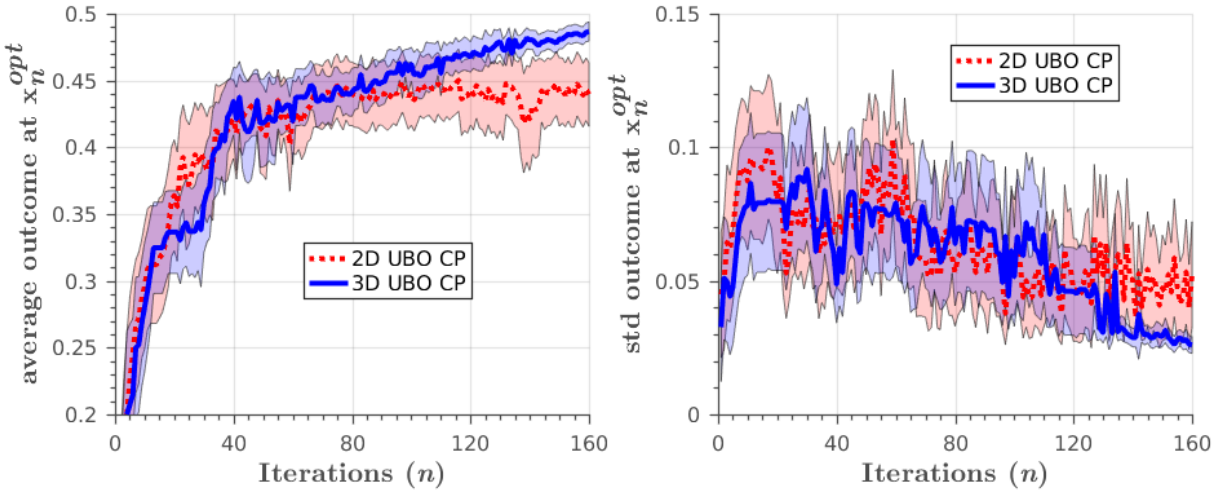


Figure 4.7: **Glass: 2D vs 3D** (UBO CP). Left: expected outcome of current optimum $\bar{y}_{mc}(x_n^{ubopt})$, Right: Variability of the outcome std$\left(y_{mc}(x_n^{ubopt})\right)$

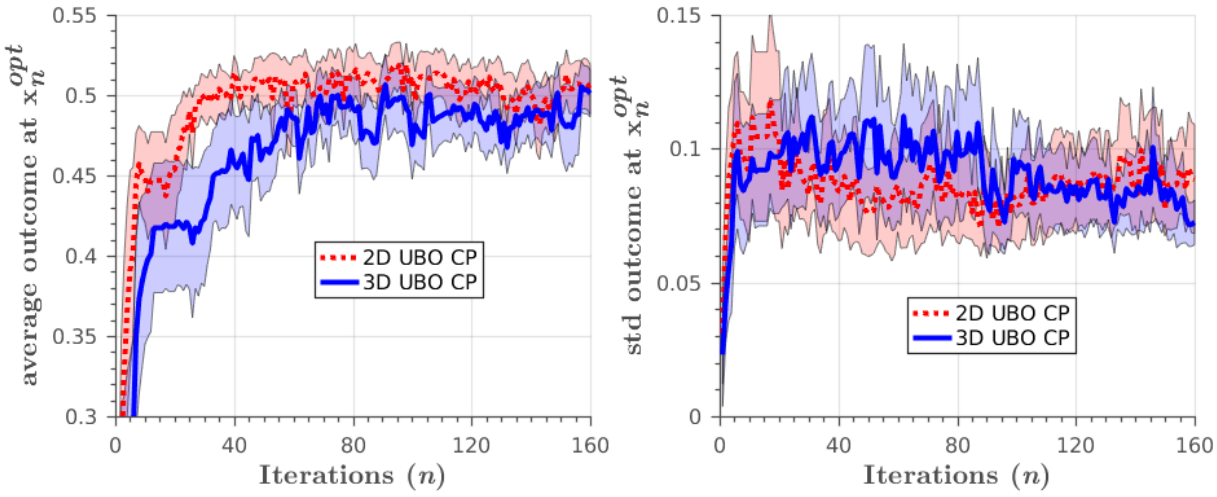


Figure 4.8: **Bottle: 2D vs 3D** (UBO CP). Left: expected outcome of current optimum $\bar{y}_{mc}(x_n^{ubopt})$, Right: Variability of the outcome std$\left(y_{mc}(x_n^{ubopt})\right)$

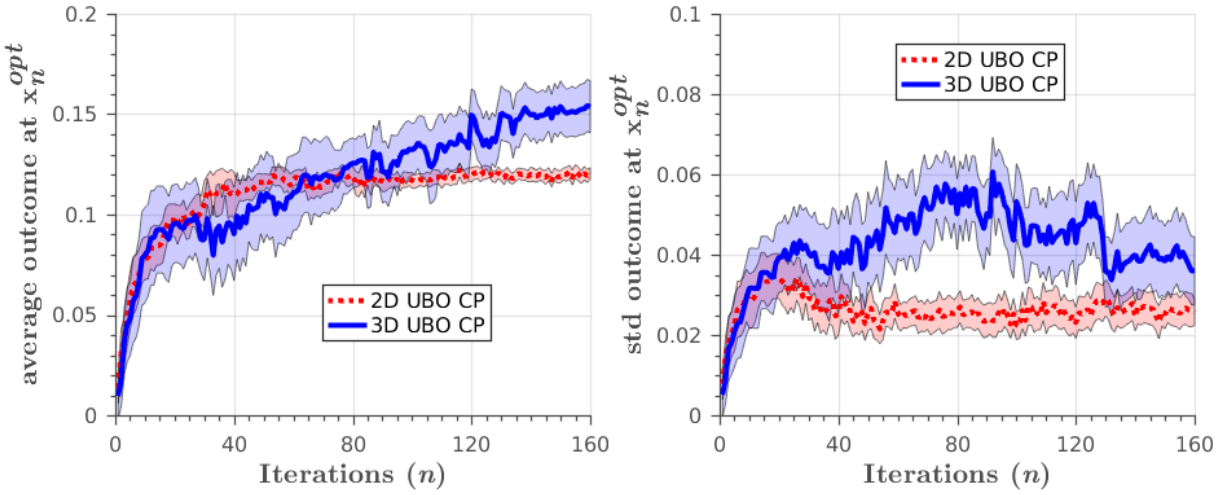


Figure 4.9: **Mug: 2D vs 3D** (UBO CP). Left: expected outcome of current optimum $\bar{y}_{mc}(x_n^{ubopt})$, Right: Variability of the outcome std$\left(y_{mc}(x_n^{ubopt})\right)$

We must point out that the z coordinate in the 2D optimization was chosen to ensure a fair comparison with the 3D, setting it to the parallel plane where the optimal grasp should be. However, the better results obtained for both glass and mug in 3D indicate that the optimum z was somewhere else, as shown in Figs. 4.10, 4.11.

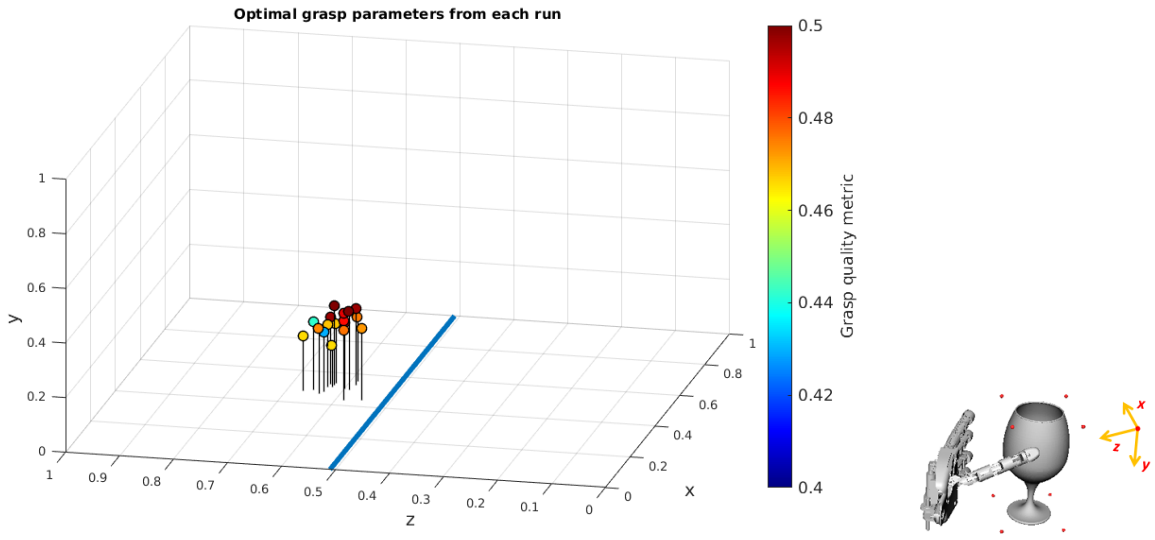


Figure 4.10: **Glass**: Optimal parameters $\mathbf{x}_n^{ubopt}$ at the final iteration ($n = 160$), for each of the 20 runs using 3D UBO CP strategy (input space normalized to the hypercube $[0, 1]^3$). The solid blue line represents the value to which coordinate z was fixed in 2D ($z = 0.51$).

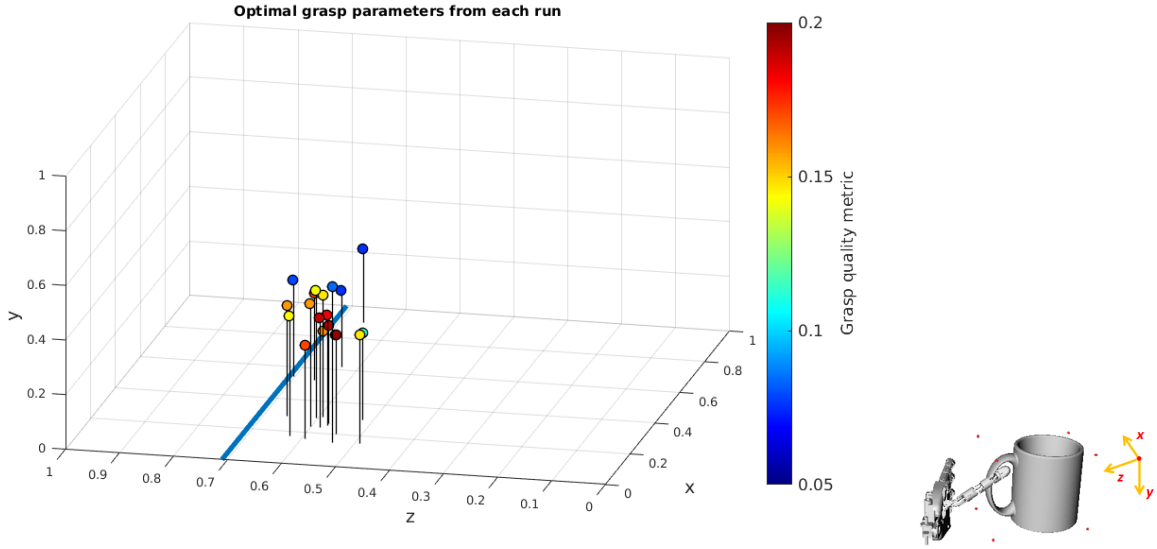
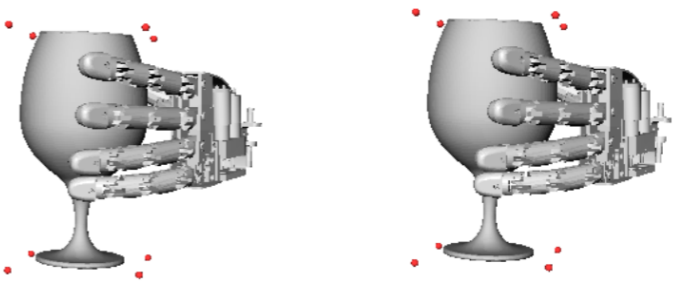


Figure 4.11: **Mug**: Optimal parameters $\mathbf{x}_n^{ubopt}$ at the final iteration ($n = 160$), for each of the 20 runs in using 3D UBO CP strategy (input space normalized to the hypercube $[0, 1]^3$). The solid blue line represents the value to which coordinate z was fixed in 2D ($z = 0.71$).

Then again, we can observe in Fig. 4.12, that the visual difference between the best grasps in 2D and 3D for the glass isn't quite noticeable, even though 3D still achieves better results.

On the bottle, as seen in Fig. 4.13, the initial z coordinate for 2D exploration was set closer to the optimum parameters achieved in 3D. This explains why the 3D results took more evaluations to reach



(a) $\mathrm{y}_n^{\mathrm{ub}opt} = 0.467$
$\bar{\mathrm{y}}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt}) = 0.4424$
$\mathrm{std}\left(\mathrm{y}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})\right) = 0.0471$

(b) $\mathrm{y}_n^{\mathrm{ub}opt} = 0.497$
$\bar{\mathrm{y}}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt}) = 0.472$
$\mathrm{std}\left(\mathrm{y}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})\right) = 0.0203$

Figure 4.12: Quality of best grasp in one of the runs. 2D UBO CP (a) and 3D UBO CP (b) converge almost to the same grasp

the same kind of results. Also in Fig. 4.13, we observe an outlier run that found a good quality grasp around $z \approx 0.53$. The existence of another region in search-space with good quality grasps is also a contributing factor to the slower convergence to 2D values.

Overall, the generalization to the 3D search-space is arguably needed since better grasp parameters were found during the 3D optimization and obviously because it is another step towards the real high-dimensional problem.

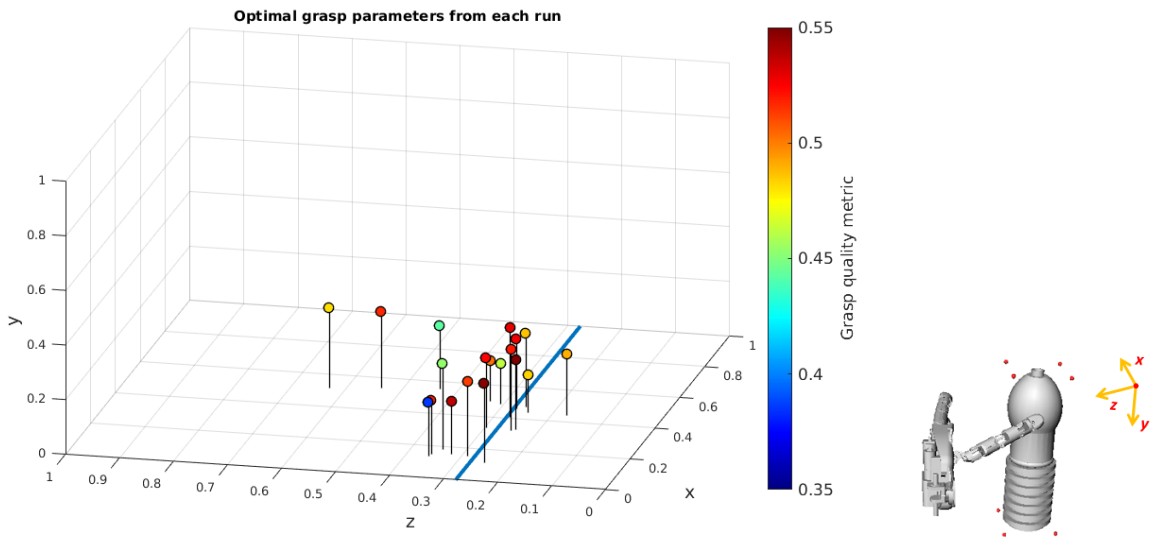


Figure 4.13: **Bottle**: Optimal parameters $\mathbf{x}_n^{\mathrm{ub}opt}$ at the final iteration ($n = 160$), for each of the 20 runs using 3D UBO CP strategy (input space normalized to the hypercube $[0, 1]^3$). The solid blue line represents the value to which coordinate z was fixed in 2D ($z = 0.27$).

4.2.4 Advantages of UBO over BO

Since we already proved that UBO generalizes well to 3D, we will compare BO against UBO in 3D using CP, to conclude if the advantages described by Nogueira et al. [11] in 2D are still attainable in 3D.

The results collected from both methods show that we are still able to learn safer grasps using UBO. The advantage is clear for both the glass, Fig. 4.14, and bottle, Fig. 4.15, where the UBO achieves higher mean values and lower variance for its optimum. In the mug, Fig. 4.16, we get competitive mean values using BO, but the UBO still finds an optimum with lower variance, *i.e.*, a safer grasp. These competitive values are due to the very limited amount of grasp configurations that yield a good quality. The visual comparison between the two optimization strategies (BO and UBO) for the glass is displayed in Fig. 4.17, where we can observe the best grasps achieved in one of the 20 runs.

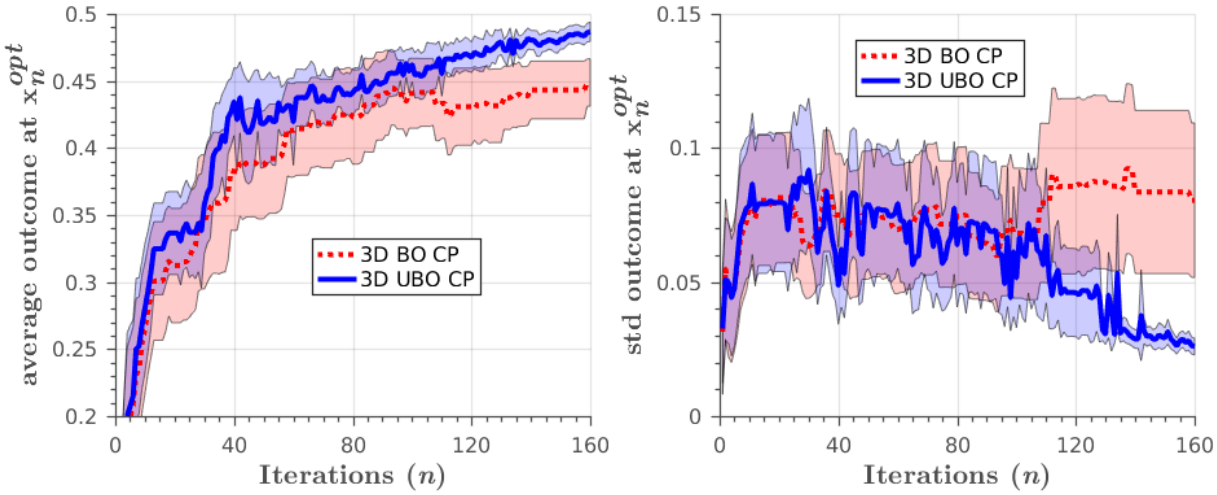


Figure 4.14: **Glass: BO vs UBO** (3D CP). Left: expected outcome of current optimum $\bar{y}_{mc}(\mathbf{x}_n^{opt})$, Right: Variability of the outcome $\text{std}\left(y_{mc}(\mathbf{x}_n^{opt})\right)$

	Glass		Bottle		Mug	
Exp.	$\bar{y}_{mc}(\mathbf{x}_n^{opt})$	$\text{std}\left(y_{mc}(\mathbf{x}_n^{opt})\right)$	$\bar{y}_{mc}(\mathbf{x}_n^{opt})$	$\text{std}\left(y_{mc}(\mathbf{x}_n^{opt})\right)$	$\bar{y}_{mc}(\mathbf{x}_n^{opt})$	$\text{std}\left(y_{mc}(\mathbf{x}_n^{opt})\right)$
2D UBO CP	0.4396	0.0536	**0.5026**	0.0887	0.1205	**0.0256**
3D UBO	0.4462	0.0761	0.4810	0.0754	0.0979	0.0378
3D UBO CP	**0.4867**	**0.0260**	0.5011	**0.0725**	**0.1567**	0.0361
3D BO CP	0.4489	0.0805	0.4767	0.1097	0.1551	0.0415

Table 4.1: Results at the last iteration ($n = 160$) of the optimization process (means over all runs)

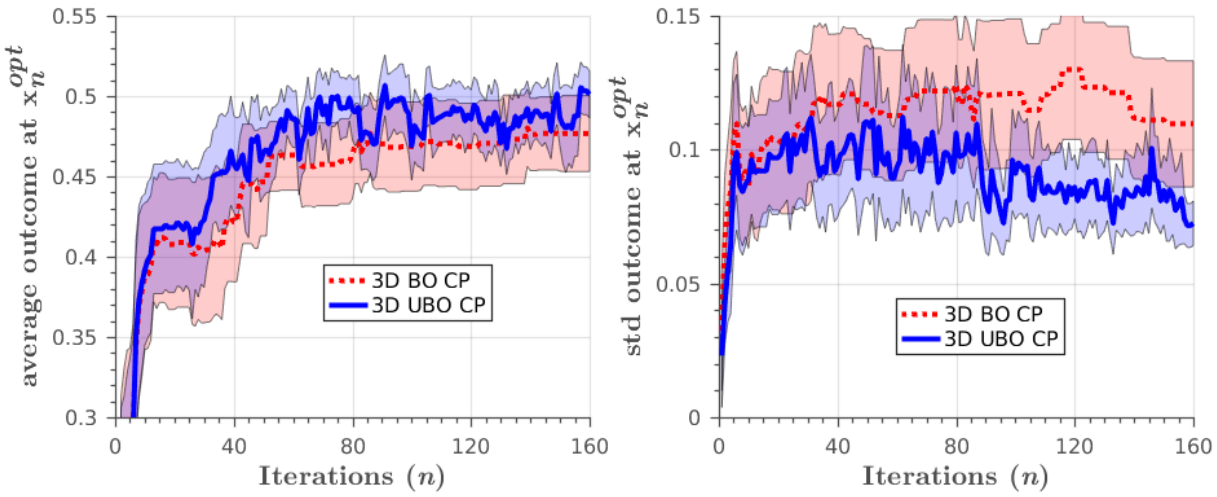


Figure 4.15: **Bottle: BO vs UBO** (3D CP). Left: expected outcome of current optimum $\bar{y}_{mc}(x_n^{opt})$, Right: Variability of the outcome std$\left(y_{mc}(x_n^{opt})\right)$

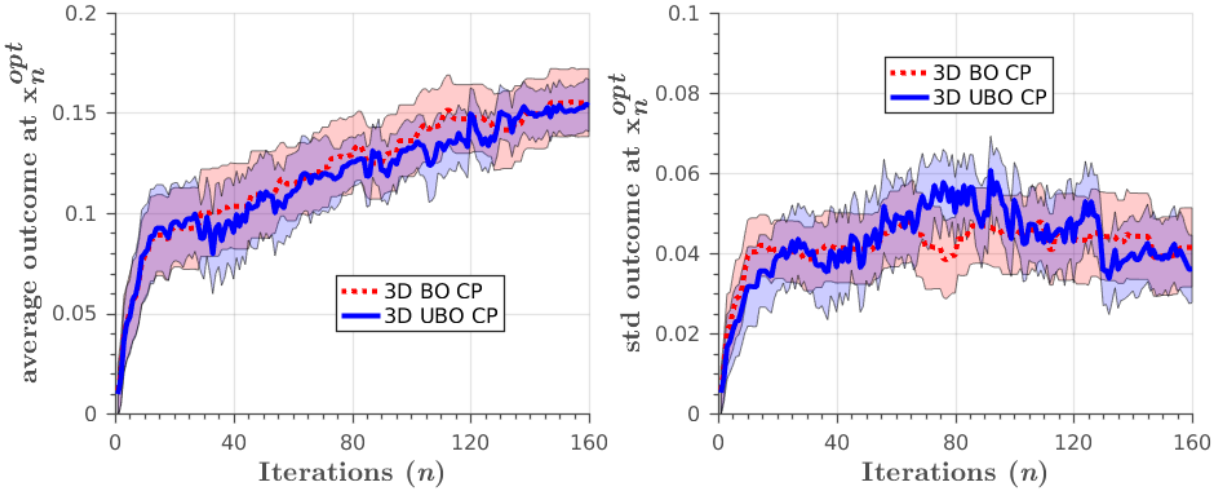
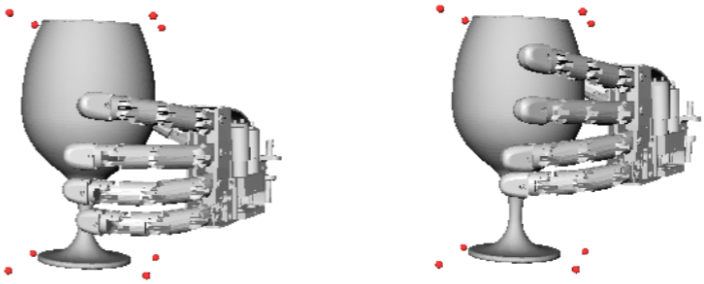


Figure 4.16: **Mug: BO vs UBO** (3D CP). Left: expected outcome of current optimum $\bar{y}_{mc}(x_n^{opt})$, Right: Variability of the outcome std$\left(y_{mc}(x_n^{opt})\right)$



(a) $\mathrm{y}_n^{\mathrm{b}opt} = 0.649$
$\bar{\mathrm{y}}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{b}opt}) = 0.383$
$\mathrm{std}\Big(\mathrm{y}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{b}opt})\Big) = 0.207$

(b) $\mathrm{y}_n^{\mathrm{ub}opt} = 0.497$
$\bar{\mathrm{y}}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt}) = 0.472$
$\mathrm{std}\Big(\mathrm{y}_{\mathrm{mc}}(\mathbf{x}_n^{\mathrm{ub}opt})\Big) = 0.0203$

Figure 4.17: Quality of best grasp in one of the runs. The best grasp achieved by BO is in an unsafe zone (a). The UBO's best grasp has a lower observation outcome but it is more robust to input noise (b)

5 Conclusions and Future work

This work has validated the application of Unscented Bayesian Optimization to 3D grasp optimization. We confirmed once more that it outperforms the classical Bayesian optimization, being able to find safer grasps under input noise. Furthermore, it generalizes well from the existing results in 2D search to a more challenging 3D search, without compromising the optimization budget. The upgrade to 3D allowed to achieve better grasps and obviously take one step closer towards a reality with many more degrees of freedom.

The expansion to 3D proved to be difficult as a result of all the queries that would result in collision configurations. These slowed down the optimization process, since a lot more observations needed to be performed to reach the same kind of results obtained in 2D. That defeated the main purpose of using a Bayesian optimization framework that strives for sample efficiency.

We propose a collision penalty function to force the search algorithm away from potential collision configurations, thus speeding up the convergence of the method. The collision penalty proved to be very important weapon to fight the curse of dimensionality, thus facilitating the transition from 2D to 3D. Naturally, as the optimization process continues expanding to further dimensions, the collision penalty will prove to be even more valuable because the number of potential collision configurations will always increase.

From here, the future roadmap is very diverse. A natural first step is studying how to extend the method to the full 6D (translation+rotation) optimization and the application of the method in a real robotic anthropomorphic hand with 3D force sensors in the finger's phalanges [46]. The implementation of the collision penalty in the real robot is also a subject that will require some further adaptations. It must be decided if the robot should run a background collision simulation with Simox before approaching the grasping pose, to verify if it the grasp is viable; or should the process be independent from a simulator and simply use the tactile forces sensors to detect any contact before the robot closes its hand. Also, the use of other grasp synergies other than the first synergy grasp can be modeled and tested.

Concerning unscented Bayesian optimization, as of right now the unscented expected improvement criterion is calculated assuming we have a perfect estimation of the input noise. However, in a practical application we do not have a perfectly accurate estimation, so it would be interesting to see how it behaves when the input noise under those circumstances and whether it still achieves better results than the classical approach. Moreover, although we assume input noise and noise corrupted outputs, we associate our observation to a noiseless input location, *i.e.*, we do not account for uncertain location estimates in observations. Therefore, an approach where our observation dataset contains noisy outcomes and also noisy incomes, can also be a subject of further research.

From a different perspective, the grasping optimization process can also become more autonomous and improve if visual exploration is incorporated. A suggested first step, would be for the robot to do a visual pre-screening before initiating the haptic exploration, identifying areas that are potentially better to explore. Thus, once the exploration started we would be able to initiate our Gaussian process prior with already some information, therefore we could once again reduce the number of exploration steps.

Bibliography

[1] M. T. Mason and J. K. Salisbury. *Robot Hands and the Mechanics of Manipulation*, chapter Manipulator grasping and pushing operations. The MIT Press, Cambridge, MA, 1985.

[2] K. Shimoga. Robot grasp synthesis algorithms: A survey. *The International Journal of Robotics Research*, 15(3):230–266, 1996.

[3] A. Sahbani, S. El-Khoury, and P. Bidaud. An overview of 3d object grasp synthesis algorithms. *Robotics and Autonomous Systems*, 60(3):326–336, 2012.

[4] J. Bohg, A. Morales, T. Asfour, and D. Kragic. Data-driven grasp synthesis - a survey. *IEEE Transactions on Robotics*, 30(2):289–309, April 2014. ISSN 1552-3098. doi: 10.1109/TRO.2013.2289018.

[5] A. Schmitz, U. Pattacini, F. Nori, L. Natale, G. Metta, and G. Sandini. Design, realization and sensorization of the dexterous icub hand. In *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pages 186–191, Dec 2010. doi: 10.1109/ICHR.2010.5686825.

[6] C. S. Lovchik and M. A. Diftler. The robonaut hand: a dexterous robot hand for space. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, volume 2, pages 907–912 vol.2, 1999. doi: 10.1109/ROBOT.1999.772420.

[7] D. R. Faria, P. Trindade, J. Lobo, and J. Dias. Knowledge-based reasoning from human grasp demonstrations for robot grasp synthesis. *Robotics and Autonomous Systems*, 62(6):794 – 817, 2014. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2014.02.003. URL `http://www.sciencedirect.com/science/article/pii/S0921889014000347`.

[8] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469 – 483, 2009. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2008.10.024. URL `http://www.sciencedirect.com/science/article/pii/S0921889008001772`.

[9] Z. Yi, R. Calandra, F. Veiga, H. van Hoof, T. Hermans, Y. Zhang, and J. Peters. Active tactile object exploration with gaussian processes. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4925–4930, Oct 2016. doi: 10.1109/IROS.2016.7759723.

[10] F. Veiga and A. Bernardino. Towards Bayesian grasp optimization with wrench space analysis. In *IROS 2012 Workshop: Beyond Robot Grasping – Modern Approaches for Learning Dynamic Manipulation*, 2012.

[11] J. Nogueira, R. Martinez-Cantin, A. Bernardino, and L. Jamone. Unscented Bayesian optimization for safe robot grasping. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1967–1972, Oct 2016. doi: 10.1109/IROS.2016.7759310.

[12] K. Huebner, K. Welke, M. Przybylski, N. Vahrenkamp, T. Asfour, D. Kragic, and R. Dillmann. Grasping known objects with humanoid robots: A box-based approach. In *2009 International Conference on Advanced Robotics*, pages 1–6, June 2009.

[13] C. Papazov, S. Haddadin, S. Parusel, K. Krieger, and D. Burschka. Rigid 3d geometry matching for grasping of known objects in cluttered scenes. *Int. J. Rob. Res.*, 31(4):538–553, Apr. 2012. ISSN 0278-3649. doi: 10.1177/0278364911436019. URL `http://dx.doi.org/10.1177/0278364911436019`.

[14] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Şucan. *Towards Reliable Grasping and Manipulation in Household Environments*, pages 241–252. Springer Berlin Heidelberg, 2014. ISBN 978-3-642-28572-1.

[15] L. Montesano and M. Lopes. Active learning of visual descriptors for grasping using non-parametric smoothed beta distributions. *Robotics and Autonomous Systems*, 60(3):452 – 462, 2012. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2011.07.013. Autonomous Grasping.

[16] R. Figueiredo, A. Shukla, D. Aragao, P. Moreno, A. Bernardino, J. Santos-Victor, and A. Billard. Reaching and grasping kitchenware objects. In *Proc. Int. Symp. Syst. Integration (SII)*, 2012.

[17] A. S. Seungsu Kim and A. Billard. Catching objects in flight. *IEEE Trans. Robot.*, 2014.

[18] J. Stückler, M. Schwarz, M. Schadler, A. Topalidou-Kyniazopoulou, and S. Behnke. Nimbro explorer: Semiautonomous exploration and mobile manipulation in rough terrain. *J. Field Robotics*, 33(4):411–430, 2015. ISSN 1556-4967. doi: 10.1002/rob.21592.

[19] D. Leidner, W. Bejjani, A. Albu-Schaeffer, and M. Beetz. Robotic agents representing, reasoning, and executing wiping tasks for daily household chores. In *Proc. Int. Conf. Autonomous Agents & Multiagent Systems (AAMAS)*, pages 1006–1014, 2016. ISBN 978-1-4503-4239-1.

[20] D. Leidner, A. Dietrich, M. Beetz, and A. Albu-Schäffer. Knowledge-enabled parameterization of whole-body control strategies for compliant service robots. *Auton. Robots*, 40(3):519–536, 2016. ISSN 1573-7527. doi: 10.1007/s10514-015-9523-3.

[21] P. Vicente, L. Jamone, and A. Bernardino. Towards markerless visual servoing of grasping tasks for humanoid robots. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3811–3816, May 2017. doi: 10.1109/ICRA.2017.7989441.

[22] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal. Online movement adaptation based on previous sensor experiences. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 365–371, Sept 2011. doi: 10.1109/IROS.2011.6095059.

[23] O. Kroemer, R. Detry, J. Piater, and J. Peters. Active learning using mean shift optimization for robot grasping. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2610–2615, Oct 2009. doi: 10.1109/IROS.2009.5354345.

[24] M. A. Roa and R. Suárez. Grasp quality measures: review and performance. *Autonomous Robots*, 38(1):65–88, 2015.

[25] C. Ferrari and J. Canny. Planning optimal grasps. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 2290–2295 vol.3, May 1992. doi: 10.1109/ROBOT.1992.219918.

[26] A. R. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to Derivative-Free Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009. ISBN 0898716683, 9780898716689.

[27] A. Smith and D. Coit. *Handbook of Evolutionary Computation*, chapter "Constraint-Handling Techniques - Penalty Function". Institute of Physics Publishing and Oxford University Press, 1997.

[28] J. S. Michael A. Gelbart and R. P. Adams. Bayesian optimization with unknown constraints. In *Uncertainty in Artificial Intelligence (UAI)*, 2014.

[29] R. Martinez-Cantin. Funneled bayesian optimization for design, tuning and control of autonomous systems. *IEEE Transactions on Cybernetics*, 2018.

[30] R. M. Murray, S. S. Sastry, and L. Zexiang. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., Boca Raton, FL, USA, 1994. ISBN 0849379814.

[31] B. León, A. Morales, and J. Sancho-Bru. *Robot Grasping Foundations*, pages 15–31. Springer International Publishing, Cham, 2014. ISBN 978-3-319-01833-1. doi: 10.1007/978-3-319-01833-1_2. URL `https://doi.org/10.1007/978-3-319-01833-1_2`.

[32] E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *CoRR*, abs/1012.2599, 2010.

[33] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[34] H. L. J. Snoek and R. Adams. Practical Bayesian optimization of machine learning algorithms. *NIPS*, page 2960–2968, 2012.

[35] R. Neal. *MCMC Using Hamiltonian Dynamics*. CRC Press, May 2011. doi: 10.1201/b10905-6. URL `http://dx.doi.org/10.1201/b10905-6`.

[36] S.Julier and J.Uhlmann. Unscented filtering and nonlinear estimation. *Proc. IEEE*, 92(3):401–422, March 2004.

[37] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, Oct 1993. ISSN 1573-2878. doi: 10.1007/BF00941892. URL `https://doi.org/10.1007/BF00941892`.

[38] D. E. Finkel. Direct optimization algorithm user guide, 2003.

[39] J. Gardner, M. Kusner, Zhixiang, K. Weinberger, and J. Cunningham. Bayesian optimization with inequality constraints. In *Proceedings of the 31st International Conference on Machine Learning*, pages 937–945, 2014.

[40] P. Allen, M. Ciocarlie, and C. Goldfeder. Grasp planning using low dimensional subspaces. In V. S. R. Balasubramanian, editor, *The Human Hand as an Inspiration for Robot Hand Development*. Springer, Cham, 2014.

[41] V. Vahrenkamp. Simox - a lightweight simulation and motion planning toolbox for c++. Accessed: 2017-12-02.

[42] N. Vahrenkamp, M. Kröhnert, S. Ulbrich, T. Asfour, G. Metta, R. Dillmann, and G. Sandini. *Simox: A Robotics Toolbox for Simulation, Motion and Grasp Planning*, pages 585–594. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-33926-4.

[43] A. Bernardino, M. Henriques, N. Hendrich, and J. Zhang. Precision grasp synergies for dexterous robotic hands. In *IEEE ROBIO*, 2013.

[44] R.Martinez-Cantin. BayesOpt: A Bayesian optimization library for nonlinear optimization, experimental design and bandits. *Journal Of Machine Learning Research*, 15:3735–3739, 2014.

[45] S. G. Johnson. *The NLopt nonlinear-optimization package*, 2014. URL `http://ab-initio.mit.edu/nlopt`.

[46] T. Paulino, P. Ribeiro, M. Neto, S. Cardoso, A. Schmitz, J. Santos-Victor, A. Bernardino, and L. Jamone. Low-cost 3-axis soft tactile sensors for the human-friendly robot vizzy. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 966–971, May 2017.