University of Liège
Faculty of Applied Sciences
Department of Electrical Engineering & Computer Science

SIMULATION-BASED INFERENCE FOR ROBOTIC GRASPING

a PhD dissertation
by Norman Marlier

Advisor: Prof. Gilles Louppe
Co-Advisor: Prof. Olivier Brüls
March 2024

This dissertation has been submitted and accepted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy in Computer Science.

JURY MEMBERS

Pierre Geurts, Professor at the Université de Liège (President);

Gilles Louppe, Professor at the Université de Liège (Advisor);

Olivier Brüls, Professor at the Université de Liège (Co-Advisor);

Pierre Sacré, Professor at the Université de Liège;

Renaud Detry, Professor at Katholieke Universiteit Leuven;

Estelle Massart, Professor at the Université Catholique de Louvain;

Robotic grasping is the task of picking up an object by applying forces and torques at a set of contacts with a gripper. This well-known problem, with more than four decades of research, still poses significant challenges mainly for three reasons: the rich nonsmooth contact dynamics of grasping, the high dimensional search space of grippers, and the sensor noise. Physical modeling provides a mathematical description of the contact dynamics, leading to the first approach in robotic grasping. Then, the profusion of data due to the rapid progress of information technology has made machine learning, a sub-field of artificial intelligence, an uncontested approach for solving real-world problems. It was quickly applied to robotic grasping with incredible results. However, both approaches neglect one of the most fundamental issues: dealing with uncertainties.

In this thesis, we aim to develop a general probabilistic framework, describing the grasping problem with random variables, to infer the grasp pose thanks to the Bayes' rule, which is a principled approach for dealing with uncertainties: our prior beliefs are updated based on new available observations. To this end, we structure our work with four contributions.

The first contribution is to lay down the foundation of our approach. We develop and explain the key components. The first component consists of the probabilistic modeling of the variables used in robotic grasping. We describe the quantities needed to solve the task by incorporating human-based knowledge of grasping while being general enough to make our modeling applicable in many situations. The second component is the method used to carry out the inference. Due to the intractability of the likelihood function, we use a family of methods known as simulation-based inference. These methods learn a part of the Bayes' rule using neural networks as surrogate models. However, they typically require lots of data, which is very costly to gather in robotics. Therefore, we leverage robotic simulators to generate samples representing the task of grasping. The last component consists of using geometric methods to search for the grasp pose by performing a Riemannian gradient descent scheme, which preserves the geometry of the space of some variables.

The next three contributions consist of designing priors relevant to the constraints of the task. While a very simple prior is used in the first contribution, bringing almost no information about the task, we propose to use invariance as a good property to inject into the prior in the second contribution. Thus, it becomes possible to complexify the task by adding more unknown parameters. These priors, however, are based on the object. To overcome this issue, we use implicit neural representations to model a prior based on the scene in the third contribution, which allows one to reason about the environment and not only one object. We extend these capabilities by providing a systematic way of designing priors in the last contribution, thus achieving a high success rate on complex tasks.

ACKNOWLEDGMENTS

When my family asked me a long time ago if I would do a PhD, my answer was "Certainly not! I want to work in the industry!". Then, I did my Master's thesis in robotics and machine learning and slowly my opinion changed to "It may be great to do a PhD". And here I am, five years later, finishing my PhD thesis. This change in my mind was possible thanks to my incredible advisors, Gilles and Olivier.

And for that, I would like to thank you. You both support and advise me along my PhD journey. You both teach me a lot of things, with your different points of view. Gilles, I like the conversations we had and the meetings which always started with "I have good news and a problem". Thank you for challenging my ideas and supporting my ideas. Olivier, I enjoyed the conversations we had as well as the BBQs in summer and Christmas dinners. You bring me the hindsight needed in robotics to embrace the whole problem. I feel fortunate to have you both as advisors, as you help me to grow as a roboticist, a data scientist, and more importantly, as a researcher.

I would like to thank all the members of the lab, from the older to the latest. Arthur and Robin, thank you for all the moments we shared. I miss all the small talks we had, the conversions as well as the after works. Juliano, you bring with you the sun from Brazil. I really appreciate all the conversations we had about mathematical tools while drinking different beers. I would to thank the robotic team from Montefiore. Pierre, I really appreciate giving the robotic courses with you, as well as all our conversions. Sven, thank you for showing me the ping pong table! These matches were very enjoyable and productive. Speaking about ping pong, I would like to thank Louis, for all the moments we shared. The journey we had in New York with Tom, Audrey, and Loïc was memorable. I would like to thank other members with whom I had the chance to spend lunch breaks: Ghilsain, Gaetan, Martin, and Olivier. I really appreciate your friendship and support.

I would like to thank my oldest road partner, Nicolas, for all the moments we shared. We achieved our bachelor's together, we achieved our master's together, we achieved our second master's together and we did our Ph.D. together. That was awesome!

I would like to thank my best friend, Guillaume. All the moments we had were greatly enjoyable and helped me to balance my life during my PhD. Moreover, you made me discover squash, which is by far the best sport I have done in my life. Thank you for all, bro.

Even if they don't fully understand all the insights of my research, I would like to warmly thank all my family for supporting me and encouraging me throughout my PhD journey. Special thanks to Isis, my little cat, who was an incredible source of warmth by sitting on my knees and a source of distraction by tapping on my keyboard with her little paws.

Finally, I would like to thank you, Emilie, for all your love, support and motivation. You help me to overcome so many challenges. Your daily presence is the best thing that has happened in my life.

CONTENTS

A robot may not injure a human being or, through inaction, allow a human being to come to harm.

A robot must obey orders given it by human beings except where such orders would conflict with the First Law.

A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

Isaac Asimov

1

INTRODUCTION

Robots, in the collective imagination, are machines capable of performing tasks at a superhuman level. In movies, robots are often depicted as entities that can surpass humans in speed, strength, and precision, accomplishing tasks with unparalleled accuracy. These remarkable capabilities stem from their possession of cybernetic brains, which *model* the world from perception to action. These cybernetic brains are, in fact, powerful algorithms that learn from their experiences of the world and rapidly adapt to novel situations. Although some of these algorithms exist, they are still in their infancy. In this context, this thesis aims to explore and contribute to modern techniques for enhancing robots' capabilities.

In the present day, robots find their primary applications in industrial production lines. They excel at performing repetitive, straightforward tasks within highly structured environments, leaving no room for uncertainties. These robots operate using a single program that executes motion and waits for instructions in a continuous loop. This approach to robotics originated during the third industrial revolution in the 1970s, improving manufacturing processes and enabling the mass production of various technological devices like smartphones, monitors, personal computers, and more. However, despite their success in industrial applications, robots face limitations when it comes to tasks with high complexity and substantial potential value addition. The current programming paradigm is inadequate to handle such intricacy, as there are countless possibilities, making it challenging to select a single action from an infinity of options.

This is where machine learning comes into play, presenting a whole new paradigm that enables us to construct programs that can automatically *learn* complex decision functions without explicit programming. Through machine learning models, robots can now extract valuable information from a diverse range of sensors, even those not directly associated with a predefined physical variable. This breakthrough has led to successful applications in various tasks, including autonomous driving, quadrupedal-legged robots, navigation, grasping, and manipulation.

While machine learning has undoubtedly revolutionized the potential of robotics in the future, there are still numerous challenges that remain unsolved. Machine learning methods exhibit a strong dependence on data, which poses difficulties in acquiring sufficient data for robotic tasks. In practice, gathering the necessary amount of samples often demands hundreds of robots executing the same tasks for extended periods, rendering them impractical for many industrial applications. Furthermore, these methods still struggle with generalization capabilities, limiting their adaptability across varying scenarios.

1.1 RESEARCH QUESTION

The primary challenge in robotics lies in accomplishing complex tasks that demand sophisticated algorithms. Robotic grasping exemplifies such a problem, warranting a closer investigation due to its need to comprehend nonsmooth and nonlinear contact mechanics. Moreover, the algorithm must adapt to a diverse array of objects, each varying in shape and physical properties, often obscured by occlusions in perception. Additionally, it must compute specific trajectories to avoid collisions between the robot and the objects. The amalgamation of these requirements makes robotic grasping a compelling benchmark in the field of robotics.

This thesis aims to address the research question: **How can robots autonomously grasp any objects in unstructured environments?** To achieve this objective, we explore innovative methods known as *simulation-based inference*. These methods leverage physical simulators to generate data and use Bayes' rule to reduce uncertainties about the parameters of interest. We investigate the general applicability of these methods in robotics by initially focusing on a simple grasping task and progressively increasing the task's complexity. Our approach begins with a modest level of prior knowledge, based on *object representation*, and then advances to a more advanced prior, based on *scene representation*.

1.2 OUTLINE AND STRUCTURE

This thesis is divided into three parts. Part i provides the necessary foundational knowledge to understand the core contributions. Chapter 2 introduces fundamental notions of robotic and more particularly robot trajectory planning. In Chapter 3, we delve into probabilistic modeling and simulation-based inference methods. The background is further enriched by exploring optimization methods on Riemannian manifolds in Chapter 4. Part ii (Chapter 5) provides a comprehensive review of the different approaches used in robotic grasping. Finally, Part iii (Chapters 7 to 10) contains all the contributions of the thesis, explaining the framework, the modeling, and the results.

Moving forward, Chapter 7 focuses on object-based priors and introduces a straightforward grasping benchmark. We establish the fundamental basis of our framework by elaborating on the modeling of the grasping problem, transforming it into a Bayesian inference problem. Subsequently, in Chapter 8, we explore the utilization of various sensor formats to enhance decision-making.

Advancing our framework, we extend its capabilities by improving the given building blocks. In Chapter 9, we explore neural implicit representations and introduce a new type of prior, enabling us to grasp a wide range of objects on a planar tabletop. In Chapter 10, we demonstrate how our method can be sequentially enhanced, enabling its utilization for new tasks.

1.3 PUBLICATIONS

While Part i and Part ii provide background content, the scientific content of this thesis is limited to four papers, each with its original contribution. Each paper constitutes its own chapter, supplemented with additional prologue and epilogue sections.

[Marlier et al., 2021] *Simulation-based Bayesian inference for multi-fingered robotic grasping*, **Marlier Norman**, Brüls Olivier and Louppe Gilles.
Preprint on Arxiv
→ Chapter 7.

[Marlier et al., 2022] *Simulation-based Bayesian inference for robotic grasping*, **Marlier Norman**, Brüls Olivier and Louppe Gilles.
International Conference on Intelligent Robots and Systems, Workshop on Probabilistic Robotics at the Age of Deep Learning, 2022
→ Chapter 8.

[Marlier et al., 2023] *Implicit representation priors meet Riemannian geometry for Bayesian robotic grasping*, **Marlier Norman**, Gustin Julien, Brüls Olivier and Louppe Gilles.
International Conference on Robotics and Automation, Workshop on Geometric Representations, 2023
→ Chapter 9.

Marlier Norman, Brüls Olivier and Louppe Gilles, (2024), *Grasping under uncertainties: Sequential Neural Ratio Estimation for 6 DoF robotic grasping* [Manuscript submitted for publication]
→ Chapter 10.

Marlier Norman, Brüls Olivier and Louppe Gilles, (2024), *Riemannian optimization for robotic grasping* [Manuscript submitted for publication]

This scientific contribution is not included in this thesis but it feels for the author as a starting point because several key ingredients exist: robots, real-world experiments, and predicting a probability of success.

[Marlier et al., 2019] *Robotic throwing controller for accelerating a recycling line*, **Marlier Norman**, Brüls Olivier, Dislaire Godefroid and Louppe Gilles.
Proceedings of the Robotix Academy Conference for Industrial Robotics (RACIR), 2019

Part I

BACKGROUND

Hands-on experience is the best way to learn about all the interdisciplinary aspects of robotics.

Rodney Brooks

2

TRAJECTORY PLANNING IN ROBOTICS

Outline

This chapter provides an in-depth introduction to trajectory planning in robotics. We begin with a description of kinematic variables, covering essential concepts such as configuration space, coordinate systems, frames, and rotation representation. Building upon this understanding, we progress to both forward and inverse kinematic analyses, exploring methods dealing with kinematic variables. Concluding this chapter, we showcase the synthesis of these fundamental tools in tackling trajectory planning problems.

2.1 INTRODUCTION

Robotic arms are mechanical machines composed of multiple bodies referred to as **links**, interconnected by **joints**. The motion of these links is performed by **actuators**, typically *electrical motors*, which generate forces and torques. In many cases, an **end-effector** is attached to a specific link, often taking the form of a gripper for grasping and a hand for manipulation.

Grasping vs Manipulation

The distinction between grasping and manipulating often arises in the robotic community. In grasping, contact points between the gripper and the object will not change during the motion. It acts like a rigid constraint between the gripper and the object. In manipulation, called *dexterous* manipulation to enhance the level of accuracy needed, the robotic hand will apply new contact forces to change the pose of the object with respect to its wrist. Furthermore, manipulating involves pushing, pulling, sliding, and so on, which is not the case in grasping.

The primary function of a robotic arm is to perform tasks by moving its body and, if equipped, its end-effector. For instance, in a pick-and-place operation, the robot moves toward an object, grasps it with its end-effector, and transports it to a different location. Similarly, in a welding task, the robot follows a precise trajectory along the parts to join the materials.

All these tasks are defined through trajectories, which describe how the position, and possibly the orientation, of the end-effector vary with time. Most of the time, the trajectories are generated by passing through specific locations, called *via-points*. It is more

convenient to describe a motion relatively to accurate locations than computing by how much the angle of the joints needs to be turned. Furthermore, these trajectories can easily avoid collisions with the objects in the environment. However, motor torques are the only variables we can control in a robotic arm. Mappings from positions to angles are thus needed to describe trajectories in terms of motor positions.

> *Kinematics vs Dynamics*
>
> Kinematics describes the motion of points, bodies, and systems of bodies without taking into account the forces to cause them to move.
> Dynamics is the study of forces and their effect on motion.

To illustrate all the concepts explained in this chapter, we introduce a toy example, which is a little robotic arm called Esbi-1 (Fig 2.1).

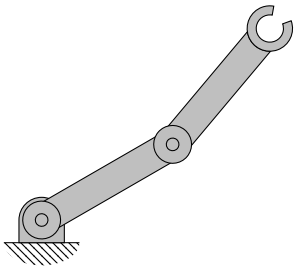
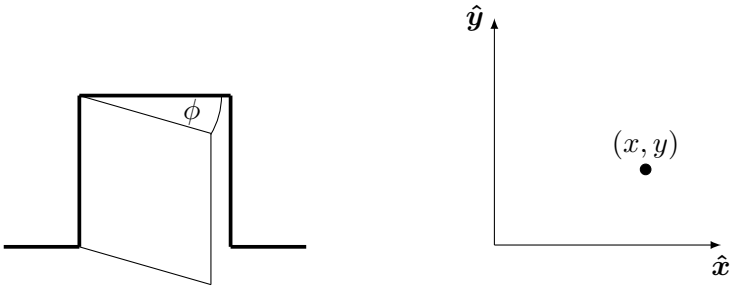


Figure 2.1: Esbi-1, our little robotic arm. We will progressively add description of this model through this chapter, to illustrate important concepts in robotics.

2.2 CONFIGURATION SPACE

The configuration space is the set of all possible positions for points within the robot. The minimum number of variables in this space corresponds to the degrees of freedom (DoF) of the robot. For instance, a door has one degree of freedom, represented by the rotation around its axis, while a point on a plane possesses two degrees of freedom (see Figure 2.2). The configuration space is an n-dimensional space, where n denotes the DoF of the robot, and it is referred to as the C-space.

In this section, we focus on the C-space and the degrees of freedom of robots. To begin, we delve into the concept of degrees of freedom, discussing the general assumptions made in robotics and exploring automated methods for computing them. Moving on, we thoroughly examine the C-space and its representation. Additionally, we explore other spaces that are closely related to the C-space, which plays a crucial role in robotics.



(a) (b)

Figure 2.2: The configuration space is the set of all possible positions for points within the system. The minimum number of variables in this space corresponds to the degrees of freedom (DoF) of the system. Example of configuration spaces (a) a door has one DoF (the rotation) while (b) a point in a plane has 2 DoF.

2.2.1 *Degrees of freedom*

The rigid body assumption asserts that the distance between any two points in a rigid body remains constant and unaffected by time or external forces. Based on this assumption, a 2D rigid body possesses exactly 3 degrees of freedom (DoF), with 2 DoF allocated to the position (x, y) of a reference point (most of the time, the center of gravity) and 1 DoF to its orientation ϕ_{xy}. Similarly, a 3D rigid body has 6 DoF, with 3 DoF pertaining to its position (x, y, z) and the remaining 3 DoF related to its orientation. The former is referred to as a **planar rigid body**, while the latter is called a **spatial rigid body**.

Since robots consist of rigid bodies, their degrees of freedom can be computed thanks to the Grübler formula [Grübler, 1884] as

$$\begin{aligned}\text{degree of freedom} = \text{sum of freedoms of the bodies} - \\ \text{number of independent constraints.}\end{aligned} \tag{2.1}$$

Constraints in a robot arise from the joints that connect its links. Various types of joints exist, enabling different kinds of motions and degrees of freedom (see Table 2.1). The revolute joint (R), also known as the *hinge*, allows rotational motion around the joint axis, making it the most common joint in robotic arms. The prismatic joint (P) allows translational motion along the joint axis. Additionally, the helical joint (H) enables both translational and rotational motion along the screw axis. Notably, these three joints possess only one degree of freedom each.

Joints can possess several degrees of freedom, offering diverse motion capabilities. The cylindrical joint (C) allows independent translational and rotational motions along a single fixed joint axis. The universal joint (U) consists of two revolute joints with orthogonal axes. Meanwhile, the spherical joint (S) offers three degrees of freedom, functioning similarly to a shoulder joint.

Joints	Degree of Freedom	Constraints	
		Planar Rigid Bodies	Spatial Rigid Bodies
Revolute (R)	1	2	5
Prismatic (P)	1	2	5
Helical (H)	1	N/A	5
Cylindrical (C)	2	N/A	4
Universal (U)	2	N/A	4
Spherical (S)	3	N/A	3

Table 2.1: Joints with their respective degrees of freedom and constraints for planar and spatial rigid bodies.

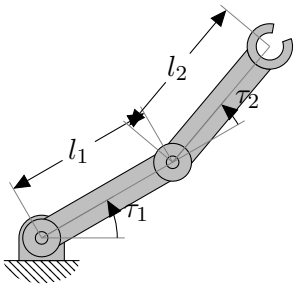


Figure 2.3: The robotic arm Esbi-1 has 2 two hinges which are revolute joints. Therefore, it possesses 4 four constraints, leading to 2 degrees of freedom, which are the angles of each hinge.

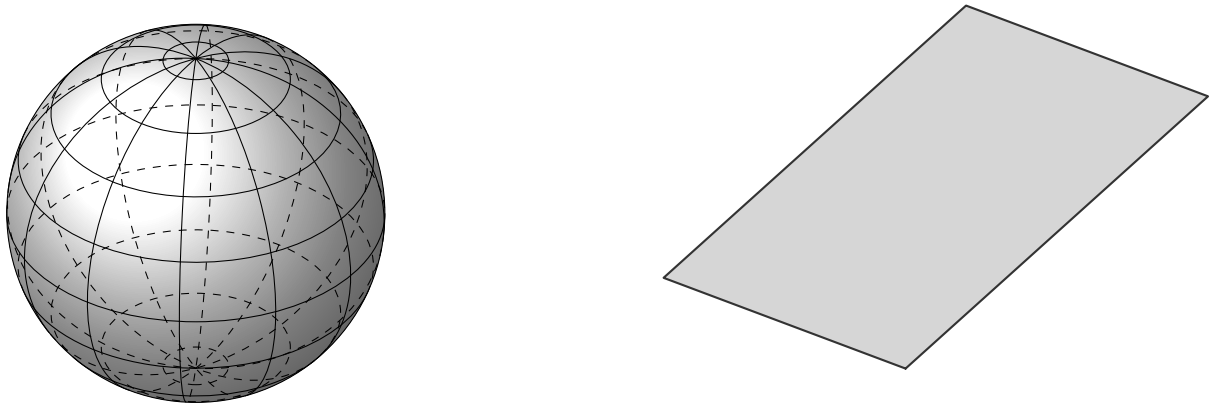
Our little robot Esbi-1 is a planar arm with two revolute joints as illustrated in Fig 2.3. Because it has two hinges, the degree of freedom of its C-space is 2. Its joint configuration is described by $\boldsymbol{\tau} = [\tau_1, \tau_2]$.

2.2.2 *Topology of the configuration space*

While the degree of freedom determines the dimension of the C-space, another important aspect is its **topology**.

Consider a sphere and a plane, both possessing two degrees of freedom. The plane can be extended infinitely in all directions, whereas the sphere cannot. Consequently, they have distinct topology. An oval-shaped American football exemplifies this concept, as it warps the sphere, stretching it in one direction to form the ball's unique shape. This idea of different geometry for spaces is referred to as topology.

For instance, one-dimensional spaces include the line, the circle, and a closed interval of a line. The circle is denoted by $\mathbb{S}^1$ or $\mathbb{S}$ since it represents a one-dimensional sphere. The



(a) (b)

Figure 2.4: Topology concerns the study of properties of geometric object that are preserved under continuous deformations. Examples of different topological spaces are the sphere and the plane. It is not possible to continuously deform a plane to obtain a sphere and vice versa.

line is denoted as $\mathbb{E}$ or $\mathbb{E}^1$, indicating a one-dimensional Euclidean space. Alternatively, it can be represented as $\mathbb{R}$ or $\mathbb{R}^1$ using real numbers once an origin, a direction, and a length scale is selected. A closed interval is written as $[a, b] \in \mathbb{R}$. These spaces also exist in higher dimensions, with $\mathbb{E}^n$ being an n-dimensional Euclidean space with the associated vector space $\mathbb{R}^n$, and $\mathbb{S}^n$ being an n-dimensional sphere embedded in an $(n + 1)$-dimensional Euclidean space.

Spaces can be constructed by expressing them as a product of lower-dimensional spaces; this is referred to as a **Cartesian product** and is denoted by $\times$. For instance, the configuration space of a planar rigid body can be expressed as $\mathbb{R}^2 \times \mathbb{S}^1$. In the case of a spatial rigid body, the configuration space becomes $\mathbb{R}^3 \times \mathbb{S}^3$. Furthermore, a torus is an n-dimensional space obtained by the product of n circles, represented as $\mathbb{T}^n = \mathbb{S}^1 \times \mathbb{S}^1 \times ... \times \mathbb{S}^1$.

It is important to note that the topology of a space remains independent of the choice of coordinates used to represent points within that space. When the number of coordinates, or parameters, is equal to the dimension of the space, we refer to this representation as a *minimal representation*. For instance, using latitude and longitude to represent points on a sphere is a minimal representation. However, such a representation can be unsatisfactory for certain regions of the space. For example, in the case of $\mathbb{S}^2$ (the 2-dimensional sphere), regions near the poles are highly sensitive. A small step in one direction results in a significant change in coordinates. The poles themselves are singularities, as one coordinate can take all possible values. These singularities can be problematic because the time derivative of the coordinates becomes infinite near them while the physical quantity is finite. To address this issue, one can use multiple **coordinate charts**, forming an **atlas**. When near a singularity in one chart, switching to another chart can help

avoid it. This solution reduces the number of parameters required, but it comes with the drawback of having to keep track of all the charts.

Another solution is to use a **redundant representation**. In this approach, more than n coordinates are used to describe an element of the space, which means that the space is considered to be embedded in a higher-dimensional Euclidean space. For instance, $\mathbb{S}^2$ is embedded in the three-dimensional Euclidean space $\mathbb{R}^3$ and can be described with (x, y, z) coordinates, *subject to a* ***constraint***: $x^2+y^2+z^2 = 1$. This constraint reduces the degrees of freedom to 2. Redundant representations have the drawback of having more coordinates than needed. However, they do not suffer from singularities. For example, small changes occur in coordinates around the North and South poles of the sphere.

Let's take back our little robot, Esbi-1. It possesses two revolute joints, each joint belonging to the unit circle. Thus, its C-space is equivalent to $\mathbb{T}^2 = \mathbb{S}^1 \times \mathbb{S}^1$. The minimal representation uses two real variables τ_0, τ_1 while the redundant representation uses many. For example, two complex numbers can be used to represent the C-space.

In summary, the non-Euclidean shape of many C-spaces motivates the use of a redundant representation. More information about manifolds, charts, and atlas can be found in Chapter 4.

2.2.3 *Task space and workspace*

The task space is the space where the robot's task can be naturally expressed and is directly related to the end-effector. For instance, drawing on a sheet of paper with a pen is expressed in $\mathbb{R}^2$, describing the position of the pen on the sheet. Grasping and manipulating a spatial rigid body is expressed in $\mathbb{R}^3 \times \mathbb{S}^3$, describing a combination of three-dimensional positional space and three-dimensional orientation space. Moreover, additional restrictions can be applied according to the specific task. For instance, the task space for grasping may be confined to $\mathbb{R}^3 \times \mathbb{S}^1$, thereby limiting the number of degrees of freedom for the orientation.

The task space is chosen by the user to represent the desired task, and as such, it may differ from the configuration space. It may not fully describe the robot's entire configuration. For instance, a 7-DoF robotic arm can have several configurations given the position and orientation of its end-effector. Additionally, some points in the task space may not be reachable by the end-effector due to physical constraints or limitations.

The task space is often confused with the workspace. Moreover, the workspace is a very overloaded term nowadays, making the distinction even more difficult. Lynch and Park [2017] define the workspace as a specification of the configurations that the end-effector can reach. It is defined by the user but depends only on the robot structure and not the task. Breyer et al. [2021] define the workspace as a 3D volume where the task happens. It is more viewed as bounds for positions in the task space than a space related to the robot itself. The definition used in the contributions of this thesis is the last one.

2.3 TRANSFORMATION MATRICES

A transformation matrix is a 4×4 matrix that provides a systematic representation of the motion, translation, and rotation of a spatial rigid body about an inertial frame. It serves as an implicit representation, comprising 16 parameters subject to 10 constraints.

These matrices are versatile tools capable of performing multiple functions: (1) translating and rotating a vector or frame and (2) expressing a vector or frame from one coordinate frame to another. All of these operations can be accomplished using linear algebra, which is the primary reason for utilizing this representation.

A transformation matrix consists of two key components: the translation part $\boldsymbol{t}$ and the rotation part $\boldsymbol{R}$. The translation component, denoted as $\boldsymbol{t}$, is a vector in $\mathbb{R}^3$, representing the spatial displacement between two points. It is written as $\boldsymbol{t}_{\mathrm{AB}}$ to signify the translation from point A to point B. Notably, this vector denotes a distance regardless of the coordinate frame it is expressed in. For clarity, we include a subscript to indicate the specific frame in which the vector is expressed, as ${}_{\mathrm{W}}\boldsymbol{t}_{\mathrm{AB}}$, where W refers to the reference frame $\underrightarrow{\mathcal{F}}_{\mathrm{W}}$. The rotation component, denoted as $\boldsymbol{R}$, is a 3×3 rotation matrix in $\mathbb{SO}(3)$, the special orthogonal group. It is written as $\boldsymbol{R}_{\mathrm{AB}}$ representing the rotation from the frame $\underrightarrow{\mathcal{F}}_{\mathrm{A}}$ to the frame $\underrightarrow{\mathcal{F}}_{\mathrm{B}}$. The transformation matrix $\boldsymbol{T}_{\mathrm{AB}}$ can be built as

$$
{}_{\mathrm{A}}\boldsymbol{t}_{\mathrm{AB}} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad \boldsymbol{R}_{\mathrm{AB}} = \begin{pmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{pmatrix} \quad \boldsymbol{T}_{\mathrm{AB}} = \begin{pmatrix} \mathbf{R}_{\mathrm{AB}} & {}_{\mathrm{A}}\boldsymbol{t}_{\mathrm{AB}} \\ \mathbf{0} & 1 \end{pmatrix}. \tag{2.2}
$$

Let us illustrate how to practically use transformation matrices. Let $\boldsymbol{p} = (p_x, p_y, p_z) \in \mathbb{R}^3$ be a point and let $\boldsymbol{p}_{\mathrm{A}}$ be the same point expressed in the coordinate system $\underrightarrow{\mathcal{F}}_{\mathrm{A}}$. We can express $\boldsymbol{p}$ in $\underrightarrow{\mathcal{F}}_{\mathrm{B}}$ using the transformation matrix $\boldsymbol{T}_{\mathrm{BA}}$ as $\boldsymbol{p}_{\mathrm{B}} = \boldsymbol{T}_{\mathrm{BA}}\boldsymbol{p}_{\mathrm{A}}$ (Fig 2.5).

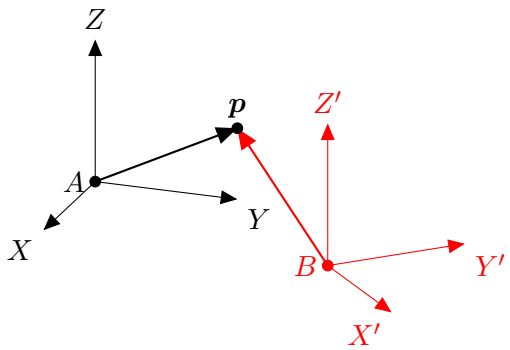
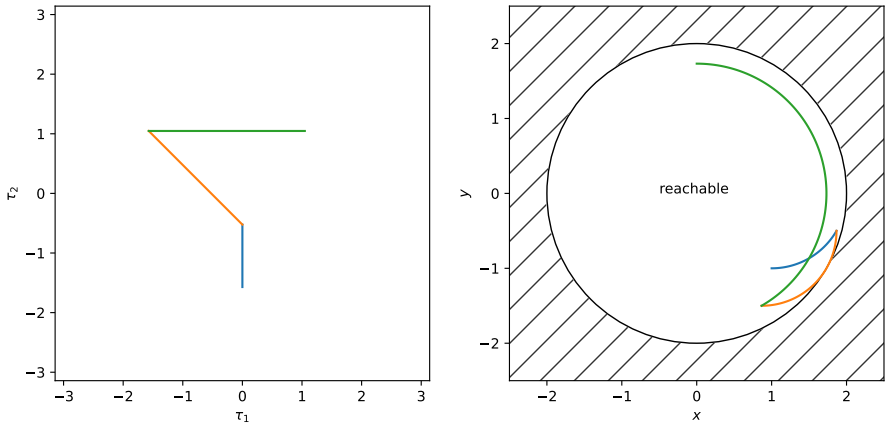


Figure 2.5: A point $\boldsymbol{p}$ exists independently of a coordinate system. Therefore, it can expressed in any reference coordinate systems. For example, it can be described by $\underrightarrow{\mathcal{F}}_{\mathrm{A}}$ or $\underrightarrow{\mathcal{F}}_{\mathrm{B}}$.

2.4 FORWARD AND INVERSE KINEMATICS

This section offers a brief introduction to forward and inverse kinematics for open-loop kinematics chains. These concepts frequently emerge in robotics, particularly when dealing with the positions and orientations of the end-effector. The configuration space does not inherently share the same topology as Euclidean spaces. Challenges can arise when transitioning between these spaces because the mappings between them are nonlinear, as illustrated in Fig 2.6. These problems are called *forward* and *inverse* kinematics.



(a) Joint configuration (b) Positions in 2D

Figure 2.6: The mapping of a given trajectory for Esbi-1 between its joints (a) and its positions (b) is nonlinear. The area inside the circle represents the reachable workspace of the robot.

2.4.1 *Forward kinematics*

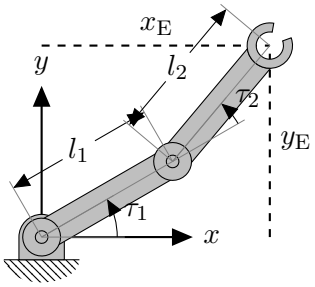
Forward kinematics, or direct kinematics, refers to the evaluation of the position and orientation of the end-effector of a robot given its joint configuration. Formally, this can be described as

$$\boldsymbol{T}_{\mathrm{ER}} = \mathrm{fkine}(\boldsymbol{\tau}), \tag{2.3}$$

with $\boldsymbol{\tau}$ being the joint configuration, $\boldsymbol{T}_{\mathrm{ER}}$ the end-effector pose E in the robot base coordinate system $\underrightarrow{\mathcal{F}}_{\mathrm{R}}$. Forward kinematics can give the same position and orientation for different joint configurations.

There are many ways to evaluate the mapping fkine. For simple cases, closed-form solutions exist. The Danevit-Hartenberg matrices [Denavit and Hartenberg, 1955] provide a systematic procedure to build transformation matrices between links and thus solve equation (2.3). But most of the time, the kinematics chain of robots is too complex to provide analytical solutions and requires numerical methods to solve (2.3).

In the case of Esbi-1, closed-form solutions exist. We can expressed the position of the end-effector $(x_{\mathrm{E}}, y_{\mathrm{E}})$ as



$$\begin{pmatrix} x_{\mathrm{E}} \\ y_{\mathrm{E}} \end{pmatrix} = \begin{pmatrix} l_1 \cos(\tau_1) + l_2 \cos(\tau_1 + \tau_2) \\ l_1 \sin(\tau_1) + l_2 \sin(\tau_1 + \tau_2) \end{pmatrix}. \tag{2.4}$$

2.4.2 *Inverse kinematics*

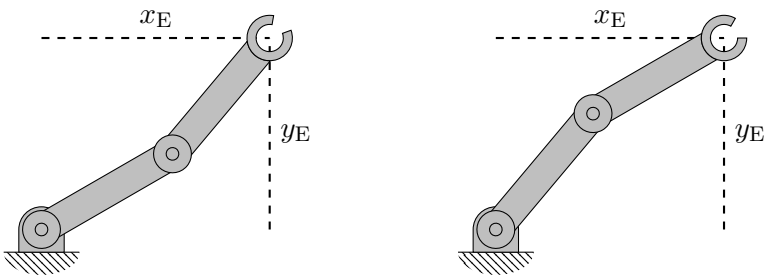
Inverse kinematics refers to the evaluation of the robot's configuration given the position and the orientation of the end-effector. Formally, this can be described as

$$\boldsymbol{\tau} = \mathrm{ikine}(\boldsymbol{T}^D_{\mathrm{ER}}), \tag{2.5}$$

with $\boldsymbol{\tau}$ being the joint configuration, $\boldsymbol{T}^D_{\mathrm{ER}}$ the desired end-effector pose E in the robot base coordinate system $\underrightarrow{\mathcal{F}}_{\mathrm{R}}$. Inverse kinematics do not always have a solution or can have more than one solution (see Fig 2.7). For example, asking for a position way too far from the robot base will result in a nonreachable position and thus an empty solution. There are two general approaches to solving equation (2.5): analytical ones and numerical ones. Analytical approaches derive closed-form solutions through symbolic manipulations. This approach is very fast and computes all the possible solutions. However, it can be hard to derive and needs to be done for every robot that has different kinematic structures. In some cases, an analytical solution may not be available. Numerical approaches compute a sequence of configurations $\boldsymbol{\tau_1}, \cdots, \boldsymbol{\tau_n}$ such that the error $\|\boldsymbol{T}^D_{\mathrm{ER}} - \mathrm{fkine}(\boldsymbol{\tau})\|$ decreases to 0. They can be grouped based on the method used: Jacobian, Newton, or heuristic methods [Aristidou et al., 2018].

Even if a closed-form solution exists for the forward kinematics, the inverse kinematics may behave in an extremely complex manner and have many solutions. In the case of Esbi-1, let $\boldsymbol{x}_D \approx \boldsymbol{T}^D_{\mathrm{ER}}$ (no rotation goal). The point $\boldsymbol{x}_D$ evolves on a circle centered at the base of the robot and located at a distance $\|\boldsymbol{x}_D\|$. Because we control the norm of $\boldsymbol{x}_D$, we can set the condition $\|\boldsymbol{x}(\tau_1, \tau_2)\| = \|\boldsymbol{x}(0, \tau_2)\|$ (see Fig 2.8). Using equation (2.3), we have

$$\begin{aligned} \|\boldsymbol{x}_D\|^2 &= l_1^2 + 2\cos(\tau_2) l_1 l_2 + l_2^2 \\ \cos(\tau_2) &= \frac{\|\boldsymbol{x}_D\|^2 - l_1^2 - l_2^2}{2 l_1 l_2} \end{aligned}$$



(a) Solution with the elbow down (b) Solution with the elbow up

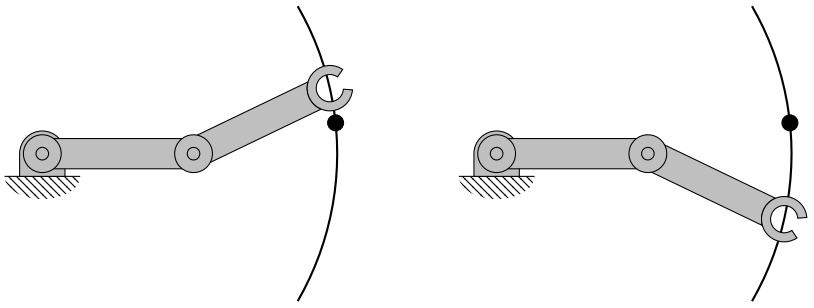
Figure 2.7: Several solutions may exist for the inverse kinematic problem, even for simple cases. Examples (a) and (b) show two different joint configurations leading to the same position.

$$\tau_2 = \pm \arccos\left(\frac{\|\boldsymbol{x}_D\|^2 - l_1^2 - l_2^2}{2l_1 l_2}\right).$$

The positive and negative solutions come from the two positions (Fig 2.8a and 2.8b). Moreover, if $\|\boldsymbol{x}_D\|^2 - l_1^2 - l_2^2 > 2l_1 l_2 \iff \|\boldsymbol{x}_D\|^2 > (l_1 + l_2)^2$, the position is unreachable and the inverse cosine has no solution. Otherwise, we note τ_2^k the two solutions with $k = \{1, 2\}$. The solutions for τ_1 are derived by introducing $\alpha = \arctan(y_D, x_D)$ and noticing that if $\tau_1 = 0$, then $\alpha^k = \arctan 2(l_2 \sin(\tau_2^k), l_1 + l_2 \cos(\tau_2^k))$. Therefore,

$$\tau_1^k = \alpha - \alpha^k,$$

showing that even for simple cases, inverse kinematics is quite complex.

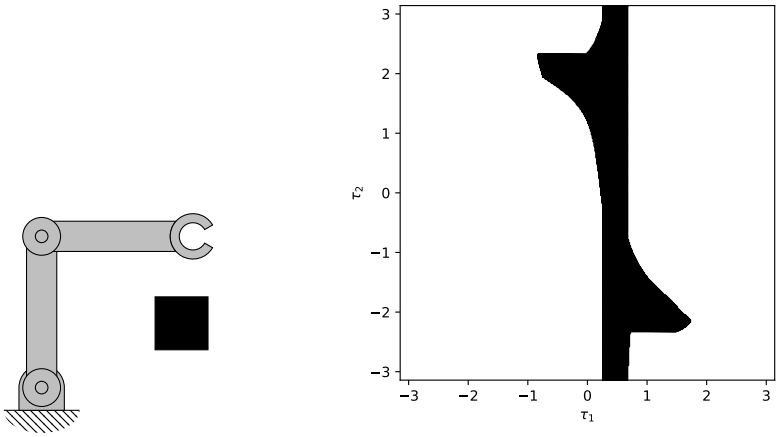


(a) Configuration corresponding to the position solution of the inverse cosine for τ_2. (b) Configuration corresponding to the negative solution of the inverse cosine for τ_2.

Figure 2.8: Procedure for computing τ_2. The robotic arm is put on the circle so that the radius is equal to the distance to the desired point. Then, we adjust the position using τ_1 to fit the corresponding point.

2.5 PATH PLANNING

Path planning involves the task of finding trajectories within the C-space, connecting an initial state to a desired goal state while avoiding obstacles in the environment and adhering to various constraints, including joint limits. Formally, the problem can be stated as searching $\tau(s), s \in [0, 1]$ with $\tau(0) = \tau_{start}$ and $\tau(1) = \tau_{goal}$. Furthermore, $\tau(s) \in C_{\text{free}}$, the set of configurations that avoid collisions. Path planning is about geometric trajectories without concerns about dynamics. There exist other planning problems in robotics such as motion planning, related to the dynamics of the robot.



(a) Workspace. (b) Configuration space.

Figure 2.9: The configuration space exhibits black regions where an object is in the workspace. This black region corresponds to collision with the object and must be avoided by the path planner.

Path planners can have several properties. These are not always fulfilled with one path planner. Some properties may be irrelevant depending on the application.

Completeness: A path planner is said to be complete if it is guaranteed to find a solution in finite time if one exits, and can report failure if there is no feasible path. This can be extended to **probabilistic completeness** if the probability of finding a solution if one exists, tends to 1 as the planning time goes to infinity.

Computational complexity: The amount of time or memory requires to solve a problem as a function of the dimension of the problem.

"Anytime" planner: A planner which finds for a better solution once it has already found one, such in [Likhachev et al., 2003].

There may exist other properties but they are not always relevant.

2.5.1 *Path planner methods*

Several methods exist to find a valid path from an initial state to a goal state. They are based on many different principles and they do not have all the same properties. The library OMPL [Şucan et al., 2012], for Open Motion Planning Library, implements all these planners in a unified framework.

Complete methods: These methods use an exact representation of the C-space, which ensures to make the optimal decision. However, they are computationally or mathematically intractable.

Grid methods: These methods discretize the C-space into a grid for searching the best path. Due to the discretization, searching is easier as well as the implementation. However, the discretization may introduce artificial issues and these methods require a big memory cost for a fixed resolution. They are usually limited to low-dimensional spaces.

Sampling methods: These methods explore the C-space by sampling randomly from one state to another one. Then, a function evaluates if the new state belongs to C_{free} or not. Finally, a kind of local planner tries to connect this new free state to the "closest" previous one. They often use a tree structure. These methods are easy to implement, tend to be probabilistically complete and work well in high dimensions. The downside is the optimality of the found path, which is most cases satisfying and not optimal.

Virtual potential fields methods: These methods create artificial forces that pull the robot towards the goal state [Khatib, 1986]. Additional repulsive forces are created near obstacles to avoid collisions. The advantages are an easy implementation and a low computational cost allowing online evaluation even for high dimensional systems. The major drawback is that local minima may appear in the potential function: the robot gets stuck in configurations that are not the goal state.

Additionally to all these methods, some smoothing may be required at the post-processing level because trajectories found by planners can be quite jerky.

2.5.2 *Example: Rapidly exploring Random Tree*

We illustrated the problem of path planning with Esbi-1. It must avoid an object in its workspace. The algorithm used for the collision avoidance is a rapidly exploring random tree (RRT) [LaValle, 1998; LaValle et al., 2001]. It starts from the initial state and explores the configuration space through random explorations. The complete algorithm is depicted in 2.1. RRTs are easily implemented and work well for low-dimensional problems. In our problem, Esbi-1 has to avoid the square to go from its initial configuration $\boldsymbol{\tau}_{\text{init}}$ to a target point $\boldsymbol{x}_D \in \mathbb{R}^2$. By using the inverse kinematic solver, we compute the

target configuration $\boldsymbol{\tau}_{\mathrm{D}} = \mathrm{ikine}(\boldsymbol{x}_D)$. These initial and target quantities are shown in Fig 2.10a, 2.10b and 2.10c. The path planner has to find a path with no collisions in the C-space. RRTs build a graph to generate a valid path (Fig 2.10d) and find among its nodes the shortest path to the target configuration. Because RRTs work by using linear interpolation, the paths found are straight lines in the configuration space (Fig 2.10e), which is not ideal for motion constraints. Fig 2.10f illustrates the cartesian trajectory, which is curved lines because of the nonlinear mapping between the configuration space and the workspace.

Algorithm 2.1 BuildRRT

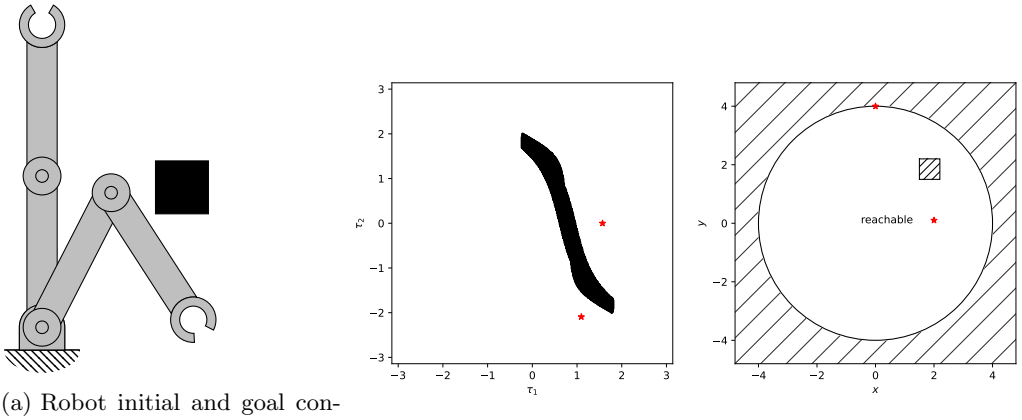
Input: Initial state τ_{init}, number of vertices in the graph K, incremental distance $\Delta\tau$.
Output: RRT graph G.

G.append(τ_{init})
for $k = 1, \ldots, K$ **do**
    $\tau_{\mathrm{rand}} \leftarrow$ RAND_CONF()
    $\tau_{\mathrm{near}} \leftarrow$ NEAREST_VERTEX(τ_{rand}, G)
    $\tau_{\mathrm{new}} \leftarrow$ NEW_CONF(τ_{near}, τ_{rand}, $\Delta\tau$)
    G.add_vertex(τ_{new})
    G.add_edge(τ_{near}, τ_{new})
end for
return G

2.6 SUMMARY

This section has introduced basic notions of robotics. Notably, the configuration and the workspace do have not the same topology and their topology is not necessarily vector spaces, which will be relevant for this thesis. Moreover, we illustrated the nonlinear mapping between configuration space and workspace through forward and inverse kinematics and introduced their intrinsic challenges. Finally, we showed an example of trajectory planning with a simple planner to avoid collision with its environment.

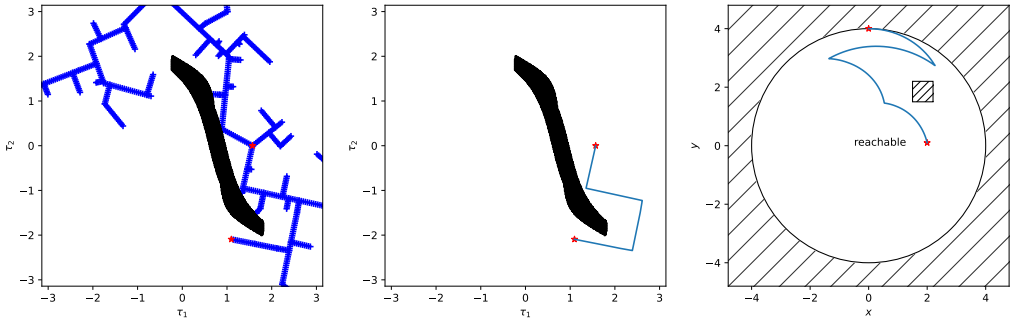
This chapter is mainly based on the book [Lynch and Park, 2017], which introduces robotics from a mechanical point of view. The notation of transformation matrices is based on a talk from a workshop [Furgale, 2014], illustrating the problem of notations in robotics where many authors have their own notations that are not enough accurate.



(a) Robot initial and goal configurations.

(b) Configuration space.

(c) Workspace.



(d) Graph in the configuration space.

(e) Trajectory in the configuration space.

(f) Trajectory in the workspace.

Figure 2.10: Illustration of the path planning problem. (a) Initial and goal configurations. The goal configuration is often computed from the position in the workspace via inverse kinematics. It can be a position to grasp an object for example. (b) the two red dots represent the initial and goal configurations. The black region represents the collisions with the object. (c) the two configurations represented in the workspace, computed via forward kinematics. (d) the graph generated by the RRT. (e) the trajectory, which consists in linear interpolations, in the configuration space. (f) the trajectory in the workspace, computed by forward kinematics.

The machine has no feelings, it feels no fear and no hope ... it operates according to the pure logic of probability. For this reason, I assert that the robot perceives more accurately than man.

Max Frisch

3

PROBABILISTIC MODELING

Outline

This chapter introduces concepts of probability theory with a focus on methods applied in this thesis. We first start by briefly explaining random variables and probability distributions. Then, we continue with inference and learning, the key ingredients of all systems based on probability. This leads to graphical models, which are convenient representations to perform inference with domain knowledge and therefore are very useful in practical applications. We finish this chapter with simulation-based inference, the core family of methods of this thesis.

3.1 INTRODUCTION

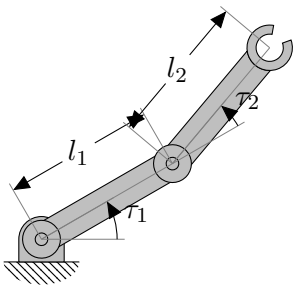
Robots are meant to perform complex tasks in unstructured environments, far from the very predictable situation of assembly lines. Thus, it becomes critical to handle the many challenges of these highly dynamic environments. *Uncertainty* is a key element in robotics that arises from different sources. First, it comes from the environment. The physical world is very complex and its dynamic can hardly be captured in a model. Thus, it is nearly impossible to model all the phenomena. Sensors are also prone to noise and do not provide perfect information about the parameters of interest. This is problematic if the decision-making process of robots relies heavily on deterministic models because small changes in input variables may result in a completely different response. If uncertainty may come from the environment, it arises also from the robots themselves. Motor actuation can result in inaccurate motion, due to friction or wear-or-tear, leading to difficult control. While all these sources are mainly from hardware and physical interactions, uncertainty arises also in the decision process. Robots are meant to make fast decisions, as humans do when they move or grasp objects, to evolve in dynamic environments. Therefore, decisions are mainly based on approximations, introducing uncertainty and inaccuracy in the decision process. All these factors make uncertainty very important for robots and cannot be ignored. Thus, it has to be properly modeled in decision-making processes. This is achieved by using the laws of probability.

3.2 PROBABILISTIC MODELS

Before talking about probabilistic models, let us introduce first what a random variable is. A random variable X is a *function* that maps possible outcomes x in the sample space Ω to a measurable space E. The triple (Ω, F, P) is the probability space where Ω is the sample space, F is a subset of possible outcomes from Ω and P is the probability function which assigns values in $[0, 1]$ for each possible outcomes. This triple satisfies the three Kolmogorov' axioms: *non-negativity*, *unitarity*, and *σ-additivity*. When the sample space is discrete, we denote $P_X(x) = P(X = x) : \Omega \mapsto [0, 1]$ the probability function. When the sample space is continuous, the probability function defines a probability that the possible outcomes x lie in a subset $A \subset \Omega$ which is expressed as $\int_{x \in A} p_X(x)dx$. The probability density function $p_X(x) = p(X = x)$ is a function that integrates to 1 if the subset is the set itself. When it is possible, we omit the subscript X for more readability. While the probability function $P(x)$ is bounded by 1, the density function $p(x)$ is not.

3.2.1 *Models*

Mathematical models are abstractions of real systems present in nature. They try to capture their intrinsic properties and allow us to make predictions about them. Let's take our little robot, Esbi-1, as an example with its mathematical modeling [Lynch and Park, 2017]



$$\boldsymbol{F} = M(\boldsymbol{\tau})\ddot{\boldsymbol{\tau}} + C(\boldsymbol{\tau}, \dot{\boldsymbol{\tau}})\dot{\boldsymbol{\tau}} + g(\boldsymbol{\tau}) \tag{3.1}$$

$$M(\boldsymbol{\tau}) = \begin{bmatrix} m_1 l_1^2 + m_2(l_1^2 + 2l_1 l_2 \cos(\tau_1) + l_2^2) & m_2(l_1 l_2 \cos(\tau_2) + l_2^2) \\ m_2(l_1 l_2 \cos(\tau_2) + l_2^2) & m_2 l_2^2 \end{bmatrix} \tag{3.2}$$

$$C(\boldsymbol{\tau}, \dot{\boldsymbol{\tau}}) = \begin{bmatrix} -m_2 l_1 l_2 \sin(\tau_2)(2\dot{\tau}_1\dot{\tau}_2 + \dot{\tau}_2^2) \\ m_2 l_1 l_2 \dot{\tau}_1^2 \sin(\tau_2) \end{bmatrix} \tag{3.3}$$

$$g(\boldsymbol{\tau}) = \begin{bmatrix} (m_1 + m_2) l_1 g \cos(\tau_1) + m_2 l_2 g \cos(\tau_1 + \tau_2) \\ m_2 l_2 g \cos(\tau_1 + \tau_2) \end{bmatrix} \tag{3.4}$$

describing how the system evolves with time and thus taking into account the *dynamics* of the system. By giving the external forces $\boldsymbol{F}$ and the parameters of the system l_1, l_2, m_1 and m_2, we can predict the evolution of the positions of the joints $\boldsymbol{\tau}$.

These models rely on the strong assumption that parameters are known with infinite precision to make predictions. They are *deterministic*. This is barely the case in practice because measurement tools have always a finite accuracy with some noise. Therefore, we need to consider the stochasticity or the randomness of the system by using *probabilistic models*. They treat the variables of interest as random variables (e.g X, Y, Z) and use probability distributions to describe phenomena; it can be joint distribution, e.g., $P(X, Y, Z)$ or conditional distributions, e.g., $P(X \mid Y, Z)$. Once the model is built, we can perform *inference* and reason about the variables. For example, what is the probability of X given Y and Z? In our example, an inference problem can be: what is the probability of the external forces $\boldsymbol{F}$ given an observation of the positions of the robot $\boldsymbol{\tau}$, $p(\boldsymbol{F} \mid \boldsymbol{\tau})$. In the deterministic case, the answer will be given by the equation (3.1), which can be complicated to solve. However, the measured positions $\boldsymbol{\tau}$ are not the true ones because of the sensor noise. Then, we can put a probability distribution on the plausible values of $\boldsymbol{F}$. From that, deterministic models are a special case of probabilistic models assuming a delta Dirac distribution for all the variables. Moreover, we may want to sample from this distribution which is different from evaluating the probability. Then, probabilistic models may handle different queries: *evaluation*, *sampling*, *marginalization*, and *conditioning*.

3.2.2 *Probabilistic graphical models*

Probabilistic graphical models (PGM) are graph-based representations that compactly encode complex probabilistic interactions between random variables [Koller and Friedman, 2009]. They take advantage of the fact that random variables interact directly with only a few others in practice. This can simplify drastically the expression of the joint distribution. The two main kinds of PGM are Bayesian Networks (BN) and Markov Random Fields (MRF). Markov Random Fields can have cyclic dependencies whereas it is not the case for Bayesian Networks. Because we will not use MRF in this thesis, we only focus on BN for this subsection.

A Bayesian network is a directed acyclic graph (DAG) in which each node represents a random variable and each edge represents the conditional dependencies between two nodes or the absence of edge represents independence (Fig 3.1). Let X_i denotes the random variable of node i. The full joint distribution in the Bayesian Network is encoded as the product of the local distributions

$$P(x_1, \ldots, x_n) = \Pi_{i=1}^{n} P(x_i \mid \text{parents}(X_i)) \tag{3.5}$$

The structure of the network expresses the domain knowledge of the problem by stating the causal dependencies between the variables. In Fig 3.1, x depends on z and θ so x can

be sampled according to the conditional density $x \sim p(x \mid \theta, z)$. The sampling can be done through computer programs that define implicitly local probability densities. Evaluating this conditional density can be done by using tables for simple random variables, such as binary variables.

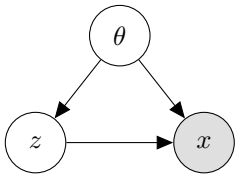


Figure 3.1: Example of a Bayesian network. White nodes are latent variables and grey nodes are observed variables. The full joint distribution is computed via equation (3.5) as $p(x, \theta, z) = p(\theta)p(z \mid \theta)p(x \mid \theta, z)$.

3.3 INFERENCE

Inference is the problem of computing a marginal and/or a conditional probability distribution from a joint probability distribution. Bayesian inference treats parameters of interest as random variables to capture subjective uncertainties. This method derives the posterior distribution from a prior distribution and the likelihood of the observed data. Formally, Bayesian inference uses Bayes's rule to update the *posterior distribution* $p(\theta \mid x)$ of parameters θ given a new observation x as

$$p(\theta \mid x) = \frac{p(x \mid \theta)}{p(x)} p(\theta), \tag{3.6}$$

with $p(x \mid \theta)$ being the *likelihood* function or the conditional probability, $p(x)$ being the marginal distribution or the evidence and $p(\theta)$ being the *prior*. These four quantities are essential in Bayesian inference which expressed probability distribution as a degree of beliefs.

The prior express knowledge we have about parameters θ without observing x. Because the Bayesian point of view assigns probability as a degree of beliefs, thus being subjective, there are many ways to construct priors [Lemoine, 2019; Sarma and Kay, 2020]. It can be an *informative prior*. For example, a Gaussian can be used with a mean and standard deviation as prior distribution for the position of the center of mass of an object to grasp. It can also be a *weakly informative prior*, which is much more permissive than informative ones. In the example of a Gaussian as a prior, a weakly informative prior will have a greater standard deviation than an informative prior, meaning that we are less aware of the position of the object. Finally, it can be an *uninformative prior* which gives little to no information about the parameters θ, or the object to grasp can be everywhere. It mostly encodes information such as "the variable is positive" or "the variable is in

the interval of values". Practically, a uniform distribution with bounds is used for such priors.

The likelihood is a special quantity. When it is viewed as a function of x with θ fixed, it is a probability distribution. When it is viewed as a function of θ with x fixed, it is the likelihood function, also noted $\mathcal{L}(\theta \mid x)$. That is said, the likelihood function is not a distribution probability of θ, which can lead to misleading conclusions. From a Bayesian point of view, this quantity describes how much information is brought by the observation x about parameters θ.

The marginal distribution is the prior predictive distribution of observation x. It can also be viewed as the likelihood function integrated over the parameters space. The distribution serves as a normalizing constant, making the integral of the posterior over the parameters equals to 1. We can also note that Bayes' rule constrains the marginal distribution to be non-zero, which implies that x must be plausible.

The Bayes' rule can be generalized to more than two random variables. This generalization will be useful for the understanding of the thesis. For example, with three random variables A, B and C, the Bayes' rule becomes

$$p(A \mid B, C) = \frac{p(B \mid A, C)}{p(B \mid C)} p(A \mid C). \tag{3.7}$$

Posterior distribution modifies the prior distribution to take into account new information from the observation. Once the posterior distribution is computed, we can extract information of interest: credible intervals, Bayesian estimators, or maximum a posteriori.

The *maximum a posterior* (MAP) is a point estimator, meaning we restrict our random variable θ to a single value $\hat{\theta} = \theta(x)$ of the parameter space. This contrasts with credible intervals, which are regions of the parameter space. The MAP is the posterior mode and is computed by

$$\hat{\theta}(x) = \arg\max_{\theta} p(\theta \mid x) \tag{3.8}$$

$$= \arg\max_{\theta} p(x \mid \theta) p(\theta) \tag{3.9}$$

because the marginal distribution does not depend on θ. Other point estimators exist such as the posterior mean or the posterior median.

All Bayesian inference methods aim to compute the posterior distribution. However, except for simple tractable problems, the marginal distribution is **intractable**. Considering a simple PGM (Fig 3.2) with $x \in \mathcal{X}$ the observations and $z \in \mathcal{Z}$ the latent variables, the marginal distribution is

$$p(x) = \int_{\mathcal{Z}} p(x, z) dz. \tag{3.10}$$

Equation (3.10) is often intractable because of the dimension of the latent space, which is possibly very large. Thus, *approximate inference* methods are developed to carry out

Figure 3.2: Latent variables model. It relates a set of observable variables x (in grey) to a set of latent variables z (in white).

inference despite the intractability of the marginal distribution.

Example of intractable marginal We consider the problem of Bayesian mixture of unit Gaussians. Let be $\boldsymbol{\mu} \in \mathbb{R}^K$ and distributed as $\boldsymbol{\mu} \sim \mathcal{N}(\mathbf{0}, \mathbb{I}_K)$. To generate an observation x_i, we first sample a cluster assignment c_i distributed as a categorical variable with a one-hot encoding. We then draw x_i from the tractable likelihood function, corresponding to a Gaussian $\mathcal{N}(c_i^T \boldsymbol{\mu}, 1)$. The full model is

$$\boldsymbol{\mu} \sim \mathcal{N}(\mathbf{0}, \mathbb{I}_K) \tag{3.11}$$

$$c_i \sim \text{Categorical}(\frac{1}{K}, \cdots, \frac{1}{K}) \tag{3.12}$$

$$x_i \mid c_i, \boldsymbol{\mu} \sim \mathcal{N}(c_i^T \boldsymbol{\mu}, 1) \tag{3.13}$$

For a sample of size n, the joint distribution of the latent and observation variables is

$$p(\boldsymbol{x}, c, \boldsymbol{\mu}) = p(\boldsymbol{\mu}) \prod_i^n p(c_i) p(x_i \mid c_i, \boldsymbol{\mu}) \tag{3.14}$$

The latent variables are $\boldsymbol{z} = \{\boldsymbol{\mu}, c\}$. Thus, the marginal distribution is

$$p(\boldsymbol{x}) = \int p(\boldsymbol{\mu}) \prod_i^n \sum_{c_i} p(c_i) p(x_i \mid c_i, \boldsymbol{\mu}) d\boldsymbol{\mu} \tag{3.15}$$

The integral in equation (3.15) is intractable because the time complexity is $O(K^n)$. This example is taken from [Blei et al., 2017] Section 2.1.

If we are only interested in the MAP, a common strategy is to drop the marginal distribution term and thus using the product of the likelihood and the prior, $p(\theta \mid x) \propto p(x \mid \theta) p(\theta)$ which will lead to the same parameters $\hat{\theta}(x)$ as explained in equation (3.8).

If the full distribution is needed, several methods exist to approximate the posterior distribution. We briefly introduce here two methods, one based on sampling, and the other based on optimization.

Markov chain Monte Carlo It formulates the Bayesian inference as a sampling problem [Hastings, 1970; Geyer, 1992; Gilks et al., 1995; Neal et al., 2011]. This family of methods aims to generate samples from a target density function, up to a constant, and extract from these samples meaningful statistics such as mean, variance, and others. There are no models of the posterior distribution leading to a low bias but a high variance of the estimators. Markov chain Monte Carlo uses two ingredients: (i) A proposal distribution $q(\theta' \mid \theta)$ which proposes new states θ' from current state θ, and (ii) an acceptance step which decides if the proposed state θ' is accepted or not with the probability to accept $\alpha(\theta, \theta') = \min\{1, \frac{p(x|\theta')p(\theta')q(\theta'|\theta)}{p(x|\theta)p(\theta)q(\theta|\theta')}\}$. While Monte Carlo methods use independent and identically distributed samples, Markov chain Monte Carlo use autocorrelated samples, needing advance techniques to reduce the correlation between samples.

Variational inference It formulates the Bayesian inference as an optimization problem [Jordan et al., 1999; Wainwright et al., 2008; Blei et al., 2017]. The posterior distribution $p(\theta \mid x)$ is approximated by the variational distribution $q_\phi(\theta) \approx p(\theta \mid x)$. The distribution $q_\phi(\theta)$ is chosen among a family of distributions simpler than the true posterior, such as Gaussian. The notion of distance between these two distributions can be expressed by the Kullback-Leibler divergence, denoted by $KL(P||Q) = \int_{x \in \mathcal{X}} \log \frac{p(x)}{q(x)} p(x) dx$. Then, we choose the parameters ϕ which minimize the KL-divergence of $q_\phi(\theta)$ and $p(\theta \mid x)$,

$$\phi^* = \arg\min_\phi KL(q_\phi(\theta)||p(\theta \mid x)). \tag{3.16}$$

Equation (3.16) is still complicated to evaluate because the evaluation of the posterior density is required. It can be written as

$$KL(q_\phi(\theta)||p(\theta \mid x)) = \mathbb{E}_{\theta \sim q_\phi(\theta)}[\log q_\phi(\theta) - \log p(x, \theta)] + \log p(x). \tag{3.17}$$

$$KL(q_\phi(\theta)||p(\theta \mid x)) = -\mathbb{E}_{\theta \sim q_\phi(\theta)}[\log p(x, \theta) - \log q_\phi(\theta)] + \log p(x). \tag{3.18}$$

$$KL(q_\phi(\theta)||p(\theta \mid x)) = -\text{ELBO}(x; \phi) + \log p(x), \tag{3.19}$$

where ELBO is the **evidence lower bound objective**. Because $p(x)$ does not depend on ϕ, Equation (3.16) can be solved by maximizing the ELBO instead of minimizing the KL-divergence. However, ELBO requires to evaluation of the likelihood function because $p(x, \theta) = p(x \mid \theta)p(\theta)$.

All these methods require to evaluate the likelihood function, whether it is for the MAP or the full distribution. However, real systems rarely exhibit simple dynamics such as Gaussian, meaning that the likelihood function does not come in a closed-form. So, the methods explained above can not be applied and we need new methods to approximate the posterior distribution.

3.4 SIMULATION-BASED INFERENCE

3.4.1 *Context*

As explained in the previous section, Bayesian inference is computationally intractable because of the marginal distribution so approximate methods are usually needed to carry out inference. However, for an increasing range of scientific problems, the likelihood function $p(x \mid \theta)$ is either computationally prohibitive to evaluate or intractable [Sisson et al., 2018]. The first case arises when the observed dataset, x_{obs}, is too big to evaluate the likelihood function, which occurs a lot in the era of Big Data. The likelihood function can be written as $p(x_{obs} \mid \theta) = \frac{1}{Z_\theta}\tilde{p}(x_{obs} \mid \theta)$ where $\tilde{p}(x_{obs} \mid \theta)$ is a function that can be easily evaluated. The normalizing constant $Z_\theta = \sum_{\mathcal{X}} \tilde{p}(x_{obs} \mid \theta)$ cannot be evaluated by brute-force enumeration due to the number of possible data configuration of $\mathcal{X}$. The second case arises when the likelihood function is defined implicitly, through quantile or characteristic functions [Drovandi and Pettitt, 2011; Peters et al., 2012], or as a data generation process [Cranmer et al., 2020]. This last case is the most interesting one for robotics because robotic simulators work as a data generation process. Let the scenario where $\theta \in \Theta$ are the parameters of the model, $z \in \mathcal{Z}$ are the latent variables and $x \in \mathcal{X}$ are the observations.

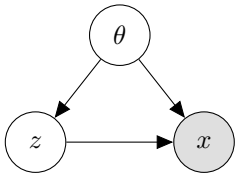


Figure 3.3: Bayesian network of the simulation-based inference setting. θ are the parameters of the model, z are the latent variables, conditioned by the parameters and x is the observation, conditioned by both the parameters and the latent variables.

The likelihood function and the marginal distribution are both intractable because the latent variables z have potentially a very high dimension

$$p(x \mid \theta) = \int_{\mathcal{Z}} p(x, z \mid \theta) dz \tag{3.20}$$

$$p(x) = \int_{\Theta} \int_{\mathcal{Z}} p(x, z \mid \theta) dz \; p(\theta) d\theta. \tag{3.21}$$

If the likelihood is intractable, it remains feasible to sample from it with the help of computer simulators (see the definition below). They define implicitly the likelihood function $p(x \mid \theta)$. For example, in the case of Esbi-1, the parameters θ are the forces $\boldsymbol{F}$, the observations x are the positions of the joints $\boldsymbol{\tau}$ and latent variables z are lengths and the masses of the links l_1, l_2, m_1 and m_2. Equation (3.1) produces samples $\boldsymbol{\tau}$ from $\boldsymbol{F}$ and l_1, l_2, m_1, m_2 according to $p(\boldsymbol{\tau} \mid \boldsymbol{F}, l_1, l_2, m_1, m_2)$ while not providing direct evaluation

of $p(\boldsymbol{\tau} \mid \boldsymbol{F}, , l_1, l_2, m_1, m_2)$. Therefore, posterior inference is still complicated because we only have access to samples of the likelihood function. Thus, we need new methods to perform inference despite the intractability of the likelihood and the marginal distribution.

Simulators: A simulator is a computer program that takes as input a vector of parameters θ, samples a series of random latent variables $z \sim p(z \mid \theta)$, and produces an observation $x \sim p(x \mid \theta, z)$.

3.4.2 *Machine learning in simulation-based inference*

The machine learning revolution has recently arisen over the last decade, showing incredible performance on various tasks, such as classification or regression. It begins with the use of neural networks, which are powerful and versatile parametric functions. Neural networks can directly work with a variety of raw data and particularly high-dimensional data, such as images. Their high capacity allows them to be trained on a huge amount of data, which is available due to new capabilities of storage. Combining them with new computation devices, such as GPU, makes neural networks the new pocketknife to solve complex problems.

In particular, neural networks are universal approximator functions [Cybenko, 1989; Funahashi, 1989], making them powerful tools for estimating probability densities. Therefore, we can make use of these tools to learn the components of Bayes's rule: it can be the likelihood function, the posterior itself, or the likelihood-ratio function.

Learning the likelihood function The intractable likelihood function $p(x \mid \theta)$ can be approximated by neural networks, which is called neural likelihood estimation (NLE) [Papamakarios et al., 2019]. For example, *Synthetic Likelihood* [Everitt, 2017; Ong et al., 2018] use a Gaussian density for a observation x_0 such that $p(x_0 \mid \theta) \approx \mathcal{N}(x_0 \mid m_\theta, S_\theta)$ where the mean m_θ and the covariance matrix S_θ are estimated from a batch of data $x_n \sim p(x \mid \theta)$ sampled at a given θ. Non-Gaussian likelihood approximations are also possible [Fasiolo et al., 2018]. Once the likelihood function is approximated, we are in the case where only the evidence is intractable so standard procedures such as MCMC or VI can be used for posterior inference.

Learning the posterior Instead of learning the likelihood function, we can directly learn the posterior density $p(\theta \mid x)$, which is called neural posterior estimation (NPE) [Papamakarios and Murray, 2016; Wehenkel and Louppe, 2019; Papamakarios et al., 2021]. Approximation models are based on *normalizing flows* [Rezende and Mohamed, 2015; Kobyzev et al., 2020], a class of invertible neural networks, which provide density estimation as well as sampling procedure.

Learning the likelihood-ratio function Another possibility is learning the likelihood-to-evidence ratio, called neural ratio estimation (NRE). NRE learn the likelihood-ratio function [Cranmer et al., 2015; Tran et al., 2017; Hermans et al., 2020; Durkan et al., 2020; Miller et al., 2021; Hermans et al., 2021; Miller et al., 2022; Thomas et al., 2022],

$$r(x \mid \theta) = \frac{p(x \mid \theta)}{p(x)} \tag{3.22}$$

by using the likelihood-ratio trick. A surrogate model, $d_\phi(x, \theta)$, can be trained as a classifier to distinguish samples coming from the joint distribution, or forward model, $x, \theta \sim p(x \mid \theta)p(\theta)$ labeled $y = 1$ against samples coming from the product of the marginals $x, \theta \sim p(x)p(\theta)$ labeled $y = 0$. The binary cross-entropy (BCE) loss is

$$L_{\text{BCE}} = -\mathbb{E}[p(x, \theta)log(d(x, \theta)) + p(x)p(\theta)log(1 - d(x, \theta))]. \tag{3.23}$$

Therefore, the minimizer of the BCE is the optimal Bayes classifier $d^*(x, \theta)$ given by

$$\frac{\partial L}{\partial d} = 0 \iff \frac{p(x, \theta)}{d^*(x, \theta)} - \frac{p(x)p(\theta)}{1 - d^*(x, \theta)} = 0 \tag{3.24}$$

$$\iff d^*(x, \theta) = \frac{p(x, \theta)}{p(x, \theta) + p(x)p(\theta)}. \tag{3.25}$$

The likelihood-ratio function (3.22) can be recovered from (3.25) by dividing the numerator and the denominator with $p(x, \theta)$

$$d^*(x, \theta) = \frac{1}{1 + r(x, \theta)^{-1}} = \frac{1}{1 + \exp^{-\log r(x,\theta)}} = \boldsymbol{\sigma}(\log r(x, \theta)) \tag{3.26}$$

with $\boldsymbol{\sigma}$ being the sigmoid function. Neural networks can be used as surrogate models of the likelihood-ratio function, providing a differentiable approximation of it. Moreover, this approach is **amortized**, meaning that inference can be carried out for any observation x without retraining the network. This makes the inference very fast but needs a larger dataset for training than non-amortized methods.

However, amortized approaches need a model with a large capacity to generalize well. This is not always a viable strategy. To overcome this issue, a sequential ratio estimation procedure can be used, in which the posterior for a given observation $x = x_0$ will be iteratively refined. The procedure starts with the prior $p_0(\theta) =: p(\theta)$, the posterior is improved by setting it the prior at the next round, $p_{t+1}(\theta) =: p_t(\theta \mid x = x_0)$. Then, a new ratio is trained at each round with data generated from the prior $p_t(\theta)$.

3.5 SUMMARY

Dealing with uncertainty is a key issue in robotics. To this end, probabilistic modeling is a formidable tool for managing unknowns and uncertainty. To effectively use the information contained in sensor data, Bayesian inference provides an update rule based on

prior, likelihood function, and posterior. However, this is computationally intractable in most cases and thus needs approximation methods. Machine learning algorithms demonstrate their capabilities to perform complex tasks, particularly density estimation. Among them, simulation-based inference methods arise and perform well even in the most restricted settings, when the likelihood function is not available, which is always the case in robotics.

True optimization is the revolutionary contribution of modern research to decision processes.

George Dantzig

4

OPTIMIZATION ON MANIFOLDS

Outline

This chapter serves as an introduction to the fundamental concepts of manifold optimization. Our exploration commences by establishing the mathematical foundation of manifolds, delving into their definition, including the concepts of tangent spaces and metrics, while also highlighting their intrinsic relationship with Euclidean spaces. We present well-known manifolds that are within the scope of robotics. Then, we explore optimization algorithms specifically designed to deal with smooth manifolds.

4.1 INTRODUCTION

Optimization problems are frequent in robotics. The problem of trajectory planning can be formulated as an optimization problem using the method of virtual potentials where the best path minimizes potential energy between the initial state and the final state. In optimal control, the motor controls should minimize a cost function along a predefined trajectory in the state space. The mechanical design of grippers and robotic arms may be improved a lot with topology optimization to maximize the stiffness or the success rate of a task given constraints.

Optimization involves the process of either maximizing or minimizing a real function by identifying the optimal input value. However, real-world scenarios often give rise to *constrained* optimization, where inputs must adhere to specific limitations, restricting them to a finite subset of the input space. These constraints are often the driving factors behind the existence of optimization problems. While navigating an unconstrained space may be relatively straightforward, accommodating constraints can prove challenging, contingent on their linearity or lack thereof. This complexity has given rise to a diverse array of solvers tailored for handling constrained optimization.

In this chapter, we embark on an exploration of a family of numerical methods designed to tackle optimization problems posed on smooth search spaces. These methods leverage the symmetry and invariance inherent in the structure of the search space. To begin, we delve into the concept of *manifolds*, which are topological spaces that bear local resemblances to Euclidean spaces. Subsequently, we delve into optimization procedures that intricately incorporate the geometrical characteristics of manifolds to effectively conduct optimization.

Non-Euclidean parameters spaces often arise in robotics, such as the stiffness matrix or the orientation. As applications, Boumal [2013] obtain the best curve of rotations which belong to the special orthogonal group to estimate a trajectory of poses of a rigid body. Jaquier et al. [2020] use Bayesian Optimization [Shahriari et al., 2015] on Riemannian manifolds to learn orientation and impedance parameters for manipulation skills. Saveriano et al. [2023] learn stable dynamical systems that evolve on Riemannian manifolds. Klein et al. [2023] use Riemannian methods for motion planning.

This chapter is mainly based on the reference book [Absil et al., 2008].

4.2 MANIFOLDS

4.2.1 *Manifolds, charts, atlas*

Informally, a manifold is a topological space with the property that each point has a neighborhood that is homeomorphic to an open subset of n-dimensional Euclidean space. For example, you can identify a portion of a circle to a segment of a line with a one-to-one correspondence. However, it is not true for the whole circle because the extremities identified on the segment of the line will lead to the same point, breaking the one-to-one relation. To define a manifold formally, we have to introduce two more concepts: *charts* and *atlas*.

Definition 4.1 (charts)**.** *Let $\mathcal{M}$ be a set. A chart φ of dimension d is a bijection of a subset $\mathcal{U}$ of $\mathcal{M}$ onto an open subset of $\mathbb{R}^d$, denoted by $(\mathcal{U}, \varphi)$.*

These charts make it possible to study objects associated with $\mathcal{U}$ by putting them to the subset $\varphi(\mathcal{U})$ of $\mathbb{R}^n$. However, it is only a subset of $\mathcal{M}$. To cover entirely $\mathcal{M}$, several charts have to be used. This collection of charts is called an *atlas*. These charts have to be consistent, meaning if a real-value function f is defined on $\mathcal{U}_1 \cap \mathcal{U}_2$, then $f \circ \varphi_1^{-1}$ and $f \circ \varphi_2^{-1}$ should have the same differentiability properties on $\mathcal{U}_1 \cap \mathcal{U}_2$.

Definition 4.2 (atlas)**.** *A (C^∞) atlas $\mathcal{A}$ of $\mathcal{M}$ into $\mathbb{R}^d$ is a collection of charts $(\mathcal{U}_\alpha, \varphi_\alpha)$ of the set $\mathcal{M}$ such that*

(a) $\bigcup_\alpha \mathcal{U}_\alpha = \mathcal{M}$

(b) for any pair α, β with $\mathcal{U}_\alpha \cap \mathcal{U}_\beta \neq \emptyset$, the sets $\varphi_\alpha(\mathcal{U}_\alpha \cap \mathcal{U}_\beta)$ are open sets in $\mathbb{R}^d$ and the change of coordinates

$$\varphi_\beta \circ \varphi_\alpha^{-1} : \mathbb{R}^d \to \mathbb{R}^d \tag{4.1}$$

is C^∞.

We need an additional property to define a manifold and its structure. Given an atlas $\mathcal{A}$, let $\mathcal{A}^+$ be the set of all charts $(\mathcal{U}, \varphi)$ such that $\mathcal{A} \cup \{(\mathcal{U}, \varphi)\}$ is also an atlas. This atlas is called *maximal atlas* or *complete atlas*. We have now all the ingredients to define properly a manifold

Definition 4.3 (manifold)**.** *A d-dimensional manifold is a couple $(\mathcal{M}, \mathcal{A}^+)$, where $\mathcal{M}$ is a set and $\mathcal{A}^+$ is a maximal atlas of $\mathcal{M}$ into $\mathbb{R}^d$, such that the topology induced by $\mathcal{A}^+$ is Hausdorff and second-countable.*

Vector spaces

Every vector space is a *linear manifold*. Let $\mathcal{E}$ be a d-dimensional vector space. Then, given a basis $(\boldsymbol{e}_i), i = 1, ..., d$ of $\mathcal{E}$, the function

$$\varphi : \mathcal{E} \to \mathbb{R}^d : \boldsymbol{x} \mapsto \begin{bmatrix} x^1 \\ \vdots \\ x^d \end{bmatrix} \tag{4.2}$$

such that $\boldsymbol{x} = \sum_i^d x^i \boldsymbol{e}_i$ is a chart of the set $\mathcal{E}$.

4.2.2 *Tangent vectors and differentiable maps*

A fundamental notion used in optimization algorithms is the idea of *directional derivatives*

$$\mathrm{D}f(x)[\eta] = \lim_{t \to 0} \frac{f(x + t\eta) - f(x)}{t}. \tag{4.3}$$

However, *nonlinear manifolds* do not have a vector space structure. Thus, the generalization of the notion of directional derivatives necessitates the development of new concepts. Let $\mathcal{M}$ be a manifold. A smooth mapping $\gamma : \mathbb{R} \to \mathcal{M} : t \mapsto \gamma(t)$ is named a curve in $\mathcal{M}$. One may define the derivative $\gamma'(t)$ as

$$\gamma'(t) =: \lim_{h \to 0} \frac{\gamma(t + h) - \gamma(t)}{h}, \tag{4.4}$$

requiring a vector space structure to compute the difference $\gamma(t+h)-\gamma(t)$, which fails for abstract nonlinear manifolds. By introducing a smooth real-value function f on $\mathcal{M}$, the function $f \circ \gamma : t \mapsto f(\gamma(t))$ is a smooth function from $\mathbb{R}$ to $\mathbb{R}$ with well-defined classical derivatives. This property can be exploited in the following definition. Let $x \in \mathcal{M}$ and γ a curve through x at $t = 0$. We denote $\mathfrak{F}_x(\mathcal{M})$ the set of smooth real-valued functions defined on a neighborhood of x. Thus, the mapping $\dot{\gamma}(0)$ from $\mathfrak{F}_x(\mathcal{M})$ to $\mathbb{R}$ is defined by

$$\dot{\gamma}(0)f =: \frac{d(f(\gamma(t)))}{dt}\Big|_{t=0}, f \in \mathfrak{F}_x(\mathcal{M}), \tag{4.5}$$

and is called *tangent vector to the curve γ at $t = 0$*.

Definition 4.4 (tangent vector)**.** *A tangent vector ξ_x to a manifold $\mathcal{M}$ at a point x is a mapping from $\mathfrak{F}_x(\mathcal{M})$ to $\mathbb{R}$ such that there exists a curve γ on $\mathcal{M}$ with $\gamma(0) = x$, satisfying,*

$$\xi_x f = \dot{\gamma}(0)f =: \frac{d(f(\gamma(t)))}{dt}\Big|_{t=0} \tag{4.6}$$

for all $f \in \mathfrak{F}_x(\mathcal{M})$. Such a curve γ is said to realize the tangent vector $\xi_x f$.

The tangent space to $\mathcal{M}$ at x, which is denoted by $\mathcal{T}_x\mathcal{M}$, is the set of all tangent vectors to $\mathcal{M}$ at x. An important feature of the tangent space is its vector space structure. Indeed, given two tangent vectors $\dot{\gamma}_1(0), \dot{\gamma}_2(0) \in \mathcal{T}_x\mathcal{M}$ and two reals $a, b \in \mathbb{R}$, the linear combination $(a\dot{\gamma}_1(0) + b\dot{\gamma}_2(0))f =: a(\dot{\gamma}_1(0)f) + b()\dot{\gamma}_2(0)f)$ belongs to the tangent space $\mathcal{T}_x\mathcal{M}$. This feature provides a local vector space approximation of the manifold, as the derivative of a real-value function provides a linear approximation. Manifold optimization algorithms are based on this vector space approximation which is more easily handled.

It is important to note that $\dot{\gamma}(0)$ is a mapping from $\mathfrak{F}_x(\mathcal{M})$ to $\mathbb{R}$ and is not seen as a time derivative $\gamma'(0)$. However, for manifolds which are embedded manifolds of a vector space, the mapping $\dot{\gamma}(0)$ from $\mathfrak{F}_x(\mathcal{M})$ to $\mathbb{R}$ and the derivative $\gamma'(0) =: \lim_{t\to\infty} \frac{1}{t}(\gamma(t) - \gamma(0))$ are closely related. For all functions $\overline{f}$ defined on subset $\mathcal{U}$ of $\mathcal{E}$, we have:

$$\dot{\gamma}(0)f = \mathrm{D}\overline{f}(\gamma(0))[\gamma'(0)], \tag{4.7}$$

where f denotes the restriction of $\overline{f}$ on $\mathcal{U} \cap \mathcal{M}$ and D denotes the *differential* of $\overline{f}(\gamma(0))$ at $\gamma'(0)$.

Most of the manifolds met in robotics are embedded manifolds of vector spaces. A submanifold of a manifold $\mathcal{M}$ is a subset $\mathcal{S}$ which itself has the structure of a manifold. Thus, $\gamma'(0)$ is well defined because $\gamma(t)$ belongs to a vector space $\mathcal{E}$ for all t. Graphically, this can be viewed as an "arrow" tangent to the manifold. When $\mathcal{M}$ is defined as a level set of a constant rank function $F : \mathcal{E} \mapsto \mathbb{R}^n$, we have

$$\mathcal{T}_x\mathcal{M} = \ker(\mathrm{D}F(x)), \tag{4.8}$$

which correspond to the vectors ξ that satisfy $\mathrm{D}F(x)[\xi] = 0$.

Tangent space to a sphere The n-dimensional sphere is the set $\mathbb{S}^n = \{x \in \mathbb{R}^{n+1} : \|x\| = 1\}$. Thus, the sphere is defined through a level set of constant rank $F : x^T x - 1 = 0$. Let $t \mapsto \gamma(t)$ a curve on the sphere through x at $t = 0$. Since $\gamma(t)$ belongs to the sphere, we have

$$\begin{aligned} F(\gamma(t)) &= 0 \\ \gamma^T(t)\gamma(t) - 1 &= 0 \end{aligned}$$

for all t. We can obtain the kernel $\ker(\mathrm{D}F(x))$ by differentiating this equation with respect to t (with $\gamma(0) = x$), which yields to

$$\mathcal{T}_x\mathbb{S}^n = \ker(\mathrm{D}F(x)) = \{z \in \mathbb{R}^{n+1} : x^T z + z^T x = 0\} = \{z \in \mathbb{R}^{n+1} : x^T z = 0\}$$

which is the set of all vectors orthogonal to x in $\mathbb{R}^{n+1}$.

To finish this section about tangent space, we introduce also the concept of *tangent bundle*, which is the set of all tangent vectors to $\mathcal{M}$:

$$\mathcal{T}\mathcal{M} =: \bigcup_{x\in\mathcal{M}} \mathcal{T}_x\mathcal{M} \tag{4.9}$$

4.2.3 *Riemannian metrics, distances, and gradients*

We have seen that tangent vectors on manifold generalize the notion of directional derivatives. However, it is still needed to have the notion of distances and lengths to effectively compute the steepest descent of an optimization problem. To do so, an *inner product* $\langle \cdot\ ,\ \cdot \rangle_x$ can be introduced on every tangent space $\mathcal{T}_x\mathcal{M}$. This inner product is bilinear and symmetric positive-definite. It induces a norm

$$\|\xi_x\|_x =: \sqrt{\langle \xi_x, \xi_x \rangle_x}.$$

If the inner product varies smoothly, the manifold is said to be a **Riemannian manifold**, and the inner product is said to be a **Riemannian metric**. We denote the Riemannian metric with $g(\xi_x, \zeta_x) = \langle \xi_x, \zeta_x \rangle_x, \xi_x, \zeta_x \in \mathcal{T}_x\mathcal{M}$. Thanks to this notion of distance, we can compute the length of the curve $\gamma : [a, b] \to \mathcal{M}$ by

$$L(\gamma) = \int_a^b \sqrt{g(\dot{\gamma}(t), \dot{\gamma}(t))} dt. \tag{4.10}$$

Therefore, the *Riemannian distance* on a connected Riemannian manifold $(\mathcal{M}, g)$ is

$$\text{dist} : \mathcal{M} \times \mathcal{M} \to \mathbb{R} : \text{dist}(x, y) = \inf_{\Gamma} L(\gamma),$$

where Γ is the set of all curves in $\mathcal{M}$ joining x to y. These new properties are very important for optimization. Indeed, given a smooth scalar field f on a Riemannian manifold $\mathcal{M}$, the **gradient** of f at x, denoted by $\text{grad} f(x)$, is defined as the unique element of $\mathcal{T}_x\mathcal{M}$ that satisfies

$$\langle \text{grad} f(x), \xi \rangle_x = \mathrm{D} f(x)[\xi], \xi \in \mathcal{T}_x\mathcal{M} \tag{4.11}$$

This gradient has several remarkable properties:

(a) The direction of $\text{grad} f(x)$ is the steepest-ascent direction of f at x.

(b) The norm of $\text{grad} f(x)$ gives the steepest slope of f at x.

Most of the Riemannian manifolds of interest for us are submanifolds of Euclidean spaces, which are also Riemannian manifolds. Therefore, they inherit the Riemannian metric of their "parent" manifolds. Let $\mathcal{M}$ be an embedded submanifold of a Riemannian manifold $\overline{\mathcal{M}}$. All the tangent spaces $\mathcal{T}_x\mathcal{M}$ are a subset of $\mathcal{T}_x\overline{\mathcal{M}}$ (they are vector spaces). Thus, the metric on $\mathcal{M}$ can be chosen as

$$g_x(\xi, \zeta) = \overline{g}_x(\xi, \zeta),\ \xi, \zeta \in \mathcal{T}_x\mathcal{M},$$

with $\overline{g}_x(\xi, \zeta)$ being the Riemannian metric of $\overline{\mathcal{M}}$. This is possible because $\xi, \zeta \in \mathcal{T}_x\mathcal{M} \subset \mathcal{T}_x\overline{\mathcal{M}}$. The difference $\mathcal{T}_x\overline{\mathcal{M}} \setminus \mathcal{T}_x\mathcal{M}$ is the orthogonal complement of $\mathcal{T}_x\mathcal{M}$ in $\mathcal{T}_x\overline{\mathcal{M}}$. It is the normal space to $\mathcal{M}$ at x and is denoted by

$$(\mathcal{T}_x\mathcal{M})^{\perp} = \{\xi \in \mathcal{T}_x\overline{\mathcal{M}} : g_x(\xi, \zeta) = 0\ \ \forall \zeta \in \mathcal{T}_x\mathcal{M}\}. \tag{4.12}$$

Therefore, any element $\xi \in \mathcal{T}_x\overline{\mathcal{M}}$ can be uniquely decomposed into the sum of an element of $\mathcal{T}_x\mathcal{M}$ and an element of $(\mathcal{T}_x\mathcal{M})^\perp$

$$\xi = P_x\zeta + P_x^\perp\zeta \tag{4.13}$$

where P_x denotes the orthogonal projection onto $\mathcal{T}_x\mathcal{M}$ and $P_x^\perp$ denotes the orthogonal projection onto $(\mathcal{T}_x\mathcal{M})^\perp$. As an introduction to the next section, we can extend our reflection from tangent spaces to functions. Let $\overline{f}$ be a cost function defined on a Riemannian manifold $\overline{\mathcal{M}}$ and let f denote the restriction of $\overline{f}$ to a Riemannian submanifold $\mathcal{M}$. We can recover the gradient of f, grad $f(x)$, by projecting the gradient of $\overline{f}$ onto $\mathcal{T}_x\mathcal{M}$:

$$\text{grad } f(x) = P_x\text{grad } \overline{f}(x). \tag{4.14}$$

In fact, the orthogonal projection of the gradient of $\overline{f}$, $P_x\text{grad } \overline{f}(x)$ belongs to $\mathcal{T}_x\mathcal{M}$ and satisfies (4.11) for all $\zeta \in \mathcal{T}_x\mathcal{M}$ because $\langle P_x\text{grad } \overline{f}(x), \zeta\rangle_x = \langle \text{grad } \overline{f}(x) - P_x^\perp\text{grad } \overline{f}(x), \zeta\rangle_x = \langle \text{grad } \overline{f}(x), \zeta\rangle_x = \mathrm{D}\overline{f}(x)[\zeta] = \mathrm{D}f(x)[\zeta]$ using (4.12).

Example of the sphere The unit sphere $\mathbb{S}^n$ is a submanifold of $\mathbb{R}^{n+1}$. Its inner product, inherited from the standard inner product on $\mathbb{R}^{n+1}$, is given by

$$\|\xi, \eta\|_x =: \xi^T\eta. \tag{4.15}$$

The normal space is

$$(\mathcal{T}_x\mathcal{M})^\perp = \{x\alpha : \alpha \in \mathbb{R}\}$$

and projections are given by

$$P_x\xi = (I - xx^T)\xi, \quad P_x^\perp\xi = xx^T\xi$$

for $x \in \mathbb{S}^n$.

4.3 FIRST-ORDER OPTIMIZATION ON MANIFOLDS

We consider the problem of minimizing a cost function $f : S \mapsto \mathbb{R}$ with S being the *search space*. The goal is to find $x \in S$ such as

$$\min_{x \in S} f(x). \tag{4.16}$$

In our case, $S = \mathcal{M}$ is a Riemannian manifold. First-order optimization algorithms in $\mathbb{R}^n$ are based on gradients to make small steps to the local minimum of the function. The general idea follows this update rule

$$x_{k+1} = x_k + \alpha_k\eta_k, \tag{4.17}$$

where $\eta_k \in \mathbb{R}^n$ is the *steepest direction* and $\alpha_k \in \mathbb{R}$ is the *step size* or *learning rate*. The generalization to a Riemannian manifold consists of selecting the steepest descent in the tangent space and moving along a particular curve in $\mathcal{M}$ to search a local minimum. This motion relies on the concept of *retraction*, a continuous mapping from a topological space to a subspace of it.

4.3.1 *Retraction*

Optimizing a differentiable function on a manifold can be described as follows; we first start from a point $x_k \in \mathcal{M}$, take the steepest descent $-\text{grad } f(x_k)$ and move along it until reaching a stationary point x where grad $f(x) = 0$. If in $\mathbb{R}^n$, the concept of moving along a curve is straightforward with a simple addition operation, in Riemannian manifold, this is generalized with *retraction*. A retraction can be viewed as a mapping from a tangent space $\mathcal{T}_x\mathcal{M}$ to a manifold $\mathcal{M}$.

Definition 4.1 (retraction)**.** *A retraction on a manifold $\mathcal{M}$ is a smooth mapping R from the tangent bundle $\mathcal{T}\mathcal{M}$ with the following properties. Let R_x denote the restriction of R to $\mathcal{T}_x\mathcal{M}$.*

(a) $R_x(0_x) = x$, where 0_x denotes the zero element of $\mathcal{T}_x\mathcal{M}$.

(b) With the canonical identification $\mathcal{T}_{0_x}(\mathcal{T}_x\mathcal{M})$, R_x satisfies

$$\mathrm{D}R_x(0_x) = \mathrm{id}_{\mathcal{T}_x\mathcal{M}}, \tag{4.18}$$

where $\mathrm{id}_{\mathcal{T}_x\mathcal{M}}$ denotes the identity mapping on $\mathcal{T}_x\mathcal{M}$.

The curve $\gamma_\xi : t \mapsto R_x(t\xi)$ satisfies $\gamma(0) = x$ and $\dot{\gamma}(0) = \xi$. In other words, moving along γ_ξ can be viewed as moving in the direction ξ while staying on the manifold $\mathcal{M}$.

As we did for other concepts in manifolds, special properties hold for retractions on embedded submanifolds. Let $\mathcal{M}$ be an embedded submanifold of a vector space $\mathcal{E}$. The tangent space of $\mathcal{M}$ at x, $\mathcal{T}_x\mathcal{M}$, is a subspace of $\mathcal{T}_x\mathcal{E} \simeq \mathcal{E}$. Therefore, we move along the sum $x + \xi,\ x \in \mathcal{M}, \xi \in \mathcal{T}_x\mathcal{M}$. Then, we project this point from the tangent space back to the manifold. This projection needs to be well-defined and computationally efficient to be used in an optimization routine. A special retraction is the *exponential map*, noted as

$$\mathrm{Exp}_x : \mathcal{T}_x\mathcal{M} \mapsto \mathcal{M} : \xi \mapsto \mathrm{Exp}_x\xi. \tag{4.19}$$

If the exponential map is computationally intensive for general manifolds, it is very straightforward to compute for most of the embedded manifolds met in robotics.

Retraction and Exponential map on the sphere

A retraction on $\mathbb{S}^n = \{x \in \mathbb{R}^{n+1} : \|x\| = 1\}$ is

$$R_x(\xi) = \frac{x + \xi}{\|x + \xi\|}, \tag{4.20}$$

defined $\forall \xi \in \mathcal{T}_x\mathbb{S}^n$. This retraction minimizes the distance to $x+\xi$. The exponential map on $\mathbb{S}^n$ is

$$\mathrm{Exp}_x(\xi) = x\cos(\|\xi\|^2) + \frac{\xi}{\|\xi\|^2}\sin(\|\xi\|^2). \tag{4.21}$$

Fig 4.1 shows the retraction and the exponential map on the circle $\mathbb{S}^1$.

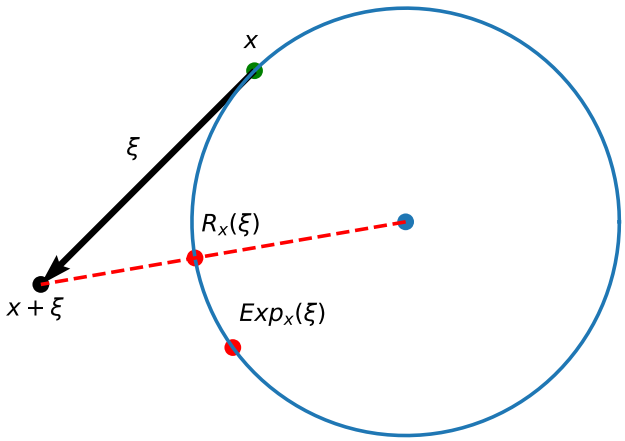


Figure 4.1: The retraction $R_x(\xi)$ and the exponential map $\mathrm{Exp}_x(\xi)$ illustrated on $\mathbb{S}^1$. The retraction $R_x(\xi)$ minimizes the distance to $x + \xi$.

4.3.2 *Line-search algorithms*

Line-search algorithms on Riemannian manifolds are based on the update rule:

$$x_{k+1} = R_{x_k}(\alpha_k \eta_k) \tag{4.22}$$

where $\eta_k \in \mathcal{T}_x\mathcal{M}$ and $\alpha_k \in \mathbb{R}$. R is a retraction from $\mathcal{T}_x\mathcal{M}$ to $\mathcal{M}$. The two remaining issues are (i) how to choose the right direction? (ii) how to choose the right step size? For that, two definitions exist, providing rules to choose the direction and the step size.

Definition 4.2 (gradient-related sequence)**.** *Given a cost function f defined on a Riemannian manifold $\mathcal{M}$, a sequence $\{\eta_k\}$, $\eta_k \in \mathcal{T}_{x_k}\mathcal{M}$, is gradient-related if, for any subsequence $\{x_k\}_{k\in\mathcal{K}}$ of $\{x_k\}$ that converges to a noncritical point of f, the corresponding subsequence $\{\eta_k\}$ is bounded and satisfies*

$$\limsup_{k\to\infty, k\in\mathcal{K}} \langle \mathrm{grad}\ f(x_k), \eta_k \rangle < 0. \tag{4.23}$$

Regarding the stepsize, one possibility is to use the Armijo point.

Definition 4.3 (Armijo point)**.** *Given a cost function f defined on a Riemannian manifold $\mathcal{M}$ with a retraction R, a point $x \in \mathcal{M}$, a tangent vector $\eta \in \mathcal{T}_x\mathcal{M}$, and scalars $\bar{t} > 0, \beta, \sigma \in (0, 1)$, the Armijo point is $\eta^A = \alpha^A \eta = \beta^m \bar{t}\eta$, where m is the smallest nonnegative integer such that*

$$f(x) - f(R_x(\beta^m \bar{t}\eta)) \geq -\sigma \langle \mathrm{grad}\ f(x), \beta^m \bar{t}\eta \rangle_x. \tag{4.24}$$

The real α^A is the Armijo step size.

Therefore, we can derive an accelerated Riemannian line-search Algorithm 4.1. In practice, we can do the same as gradient descent without a line search in vector spaces, that is the direction is given by the Euclidean gradient projected onto the tangent space and the step size is constant for all the iterations. This procedure is Riemannian gradient descent (Algorithm 4.2).

Algorithm 4.1 Accelerated Line Search (ALS)

Require: Riemannian manifold $\mathcal{M}$; continuously differentiable scalar field f on $\mathcal{M}$; retraction R from $\mathcal{TM}$ to $\mathcal{M}$, scalars $\bar{t} > 0, c, \beta, \sigma \in (0, 1)$.
Input: Initial state $x_0 \in \mathcal{M}$.
Output: Sequence of iterates x_k.

1: **for** $k = 0, 1, 2, \ldots$ **do**
2: Pick η_k in $\mathcal{T}_{x_k}\mathcal{M}$ such that the sequence $\{\eta_i\}_{i=0,1,\ldots}$ is gradient-related.
3: Select x_{k+1} such that

$$f(x_k) - f(x_{k+1}) \geq c(f(x_k) - f(R_{x_k}(\alpha_k^A \eta_k))), \tag{4.25}$$

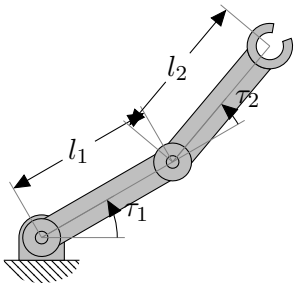
where α_k^A is the Armijo step size for the given $\bar{t}, \beta, \sigma, \eta_k$.
4: **end for**

Algorithm 4.2 Riemannian gradient descent

Require: Riemannian manifold $\mathcal{M}$; continuously differentiable scalar field f on $\mathcal{M}$; retraction R from $\mathcal{TM}$ to $\mathcal{M}$, step size α, maximum number of iterations N.
Input: Initial state $x_0 \in \mathcal{M}$.
Output: Sequence of iterates x_k.

1: **for** $k = 0, 1, 2, \ldots, N$ **do**
2: Compute Euclidean gradients $\nabla f(x_k)$.
3: Compute the Riemannian gradients $\text{grad} f(x_k) = P_{x_k}(\nabla f(x_k))$.
4: Set $x_{k+1} = R_{x_k}(-\alpha \text{grad} f(x_k))$.
5: **end for**

Example of optimization on manifold Let's take the inverse kinematics of our little robot Esbi-1. Instead of using an analytical solution as viewed in Chapter 2, we can frame the problem as an optimization problem to find the joints. We first solve this problem by considering the variables which belong to a vector space. The joints



$\boldsymbol{\tau} = [\tau_1, \tau_2] \in [0, 2\pi]^2$ and we would like to minimize the distance between the current position of the end-effector and the target position

$$\boldsymbol{\tau}^* = \arg\min_{\boldsymbol{\tau}} L(\boldsymbol{\tau}, \boldsymbol{x}_{\text{target}}) \tag{4.26}$$

$$= \arg\min_{\boldsymbol{\tau}} \|\text{fkine}(\boldsymbol{\tau}) - \boldsymbol{x}_{\text{target}}\|^2. \tag{4.27}$$

This equation has two solutions because analytical solutions give two solutions. The desired position is $\boldsymbol{x}_{\text{target}} = [1., 1.5]$. Analytical solution give

$$\boldsymbol{\tau}^1 = [5.35e{-}1, 8.96e{-}1]$$
$$\boldsymbol{\tau}^2 = [1.43, 5.39],$$

if we restricted the solutions to $[0, 2\pi]$. The loss function $L(\boldsymbol{\tau}, \boldsymbol{x}_{\text{target}})$ can be depicted in Fig 4.2. The analytical solutions are shown with a star marker in red. Depending on the starting, gradient descent will eventually converge to one of the two optimal solutions. However, if we choose a starting point from the bottom right area, we may fail to converge to analytical solutions. If we choose a larger domain, for example $[0, 4\pi]$, a repeated pattern emerges (Fig 4.3). This is because torus $\mathbb{T}$ does not share the same topology as $\mathbb{R}^2$ and cannot be parametrized with only two real numbers.

A solution will be to use a variable $\boldsymbol{\tau} \in \mathbb{T}$ and optimize the loss function $L_{\mathcal{M}}(\boldsymbol{\tau}, \boldsymbol{x}_{\text{target}}) : \mathbb{T}^1 \times \mathbb{R}^2 \mapsto \mathbb{R}$ which has only two minimizers: the two solutions given by analytical solutions. This loss function is illustrated in Fig 4.4. Therefore, we can use Riemannian gradient descent to solve

$$\boldsymbol{\tau}^* = \arg\min_{\boldsymbol{\tau} \in \mathbb{T}} L_{\mathcal{M}}(\boldsymbol{\tau}, \boldsymbol{x}_{\text{target}}). \tag{4.28}$$

The algorithm converges toward one minimizer equivalent to one of the analytical solutions. The solution obtained depends on the starting point because we use a gradient descent-like scheme. Fig 4.5 illustrates an optimization trajectory on the torus while Fig 4.6a shows the value of the components of $\boldsymbol{\tau} \in \mathbb{T}$ with respect to the number of iterations. Fig 4.6b shows the converge toward the minimal value of $L_{\mathcal{M}}(\boldsymbol{\tau}, \boldsymbol{x}_{\text{target}})$, which is 0 leading to $\text{fkine}(\boldsymbol{\tau}^*) = \boldsymbol{x}_{\text{target}}$.

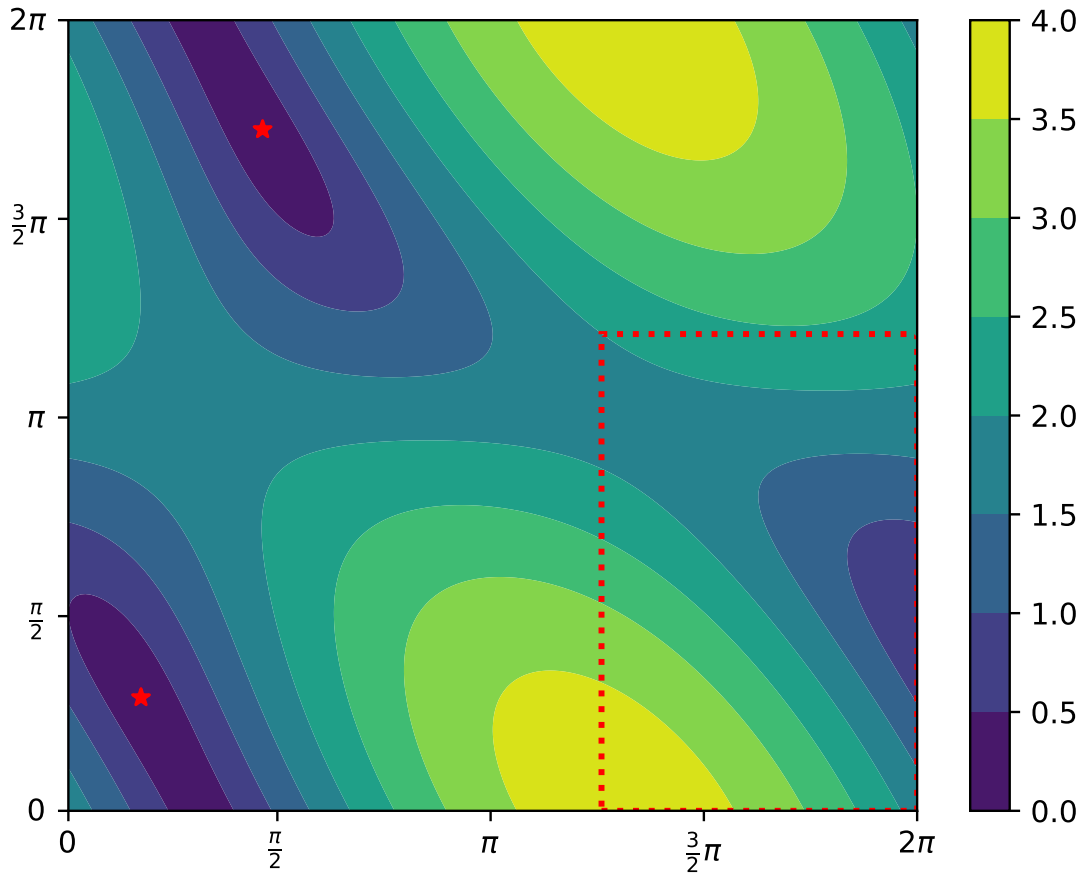


Figure 4.2: Value of the loss function $L(\boldsymbol{\tau}) = \|\text{fkine}(\boldsymbol{\tau}) - \boldsymbol{x}_{\text{target}}\|^2$ on the domain $[0, 2\pi]$. Optimal solutions are depicted with a star marker in red. The red rectangle represents the approximated region which leads to non-optimal solutions if the starting point is chosen in.

4.4 SUMMARY

Manifold optimization extends gradient-based algorithms in Euclidean spaces to Riemannian manifolds. It generalizes the notion of directional derivatives used in the optimization scheme. As we have seen in Chapter 2, robots have variables that belong to Riemannian manifolds such as tori or spheres. While Euclidean gradient descent can be applied to these variables, it does not preserve their geometric properties. Therefore, manifold optimization appears to be the proper way to optimize robotic variables.

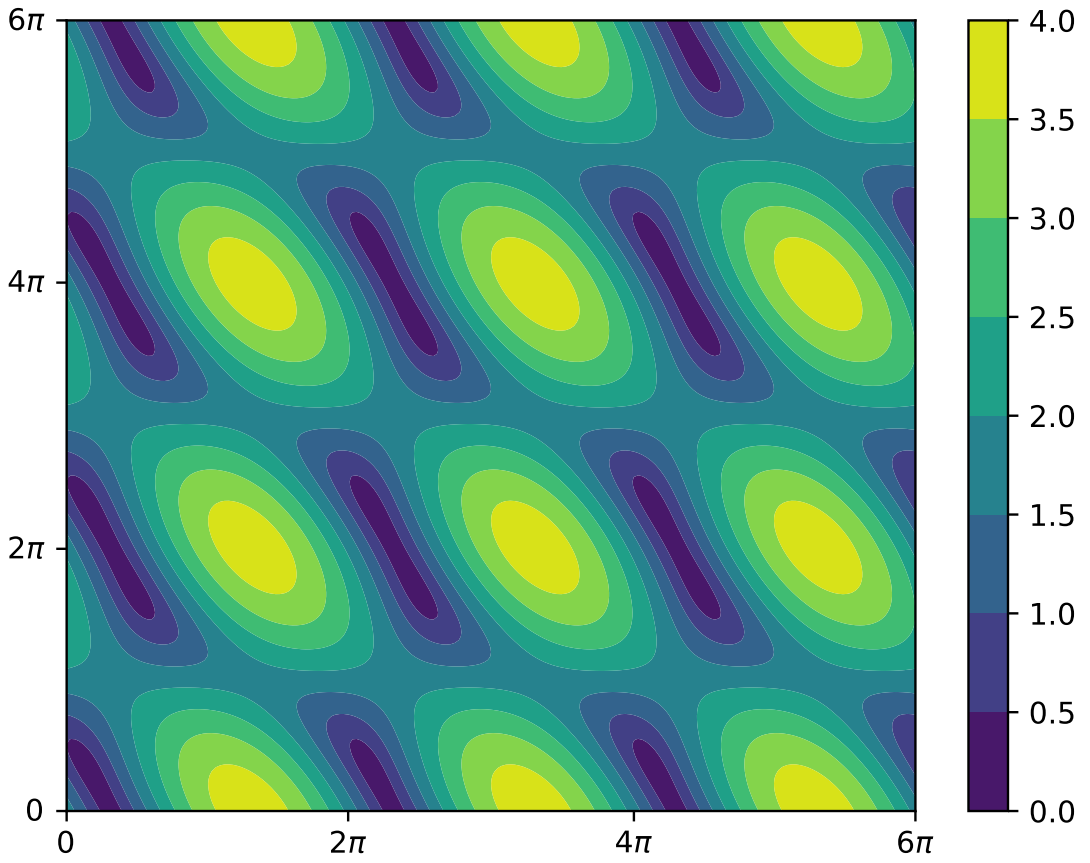


Figure 4.3: Value of the loss function $L(\boldsymbol{\tau}) = \|\text{fkine}(\boldsymbol{\tau}) - \boldsymbol{x}_{\text{target}}\|^2$ on the domain $[0, 6\pi]$. A pattern emerges as the true domain is a torus $\mathbb{T}$, leading to multiple minimizers.

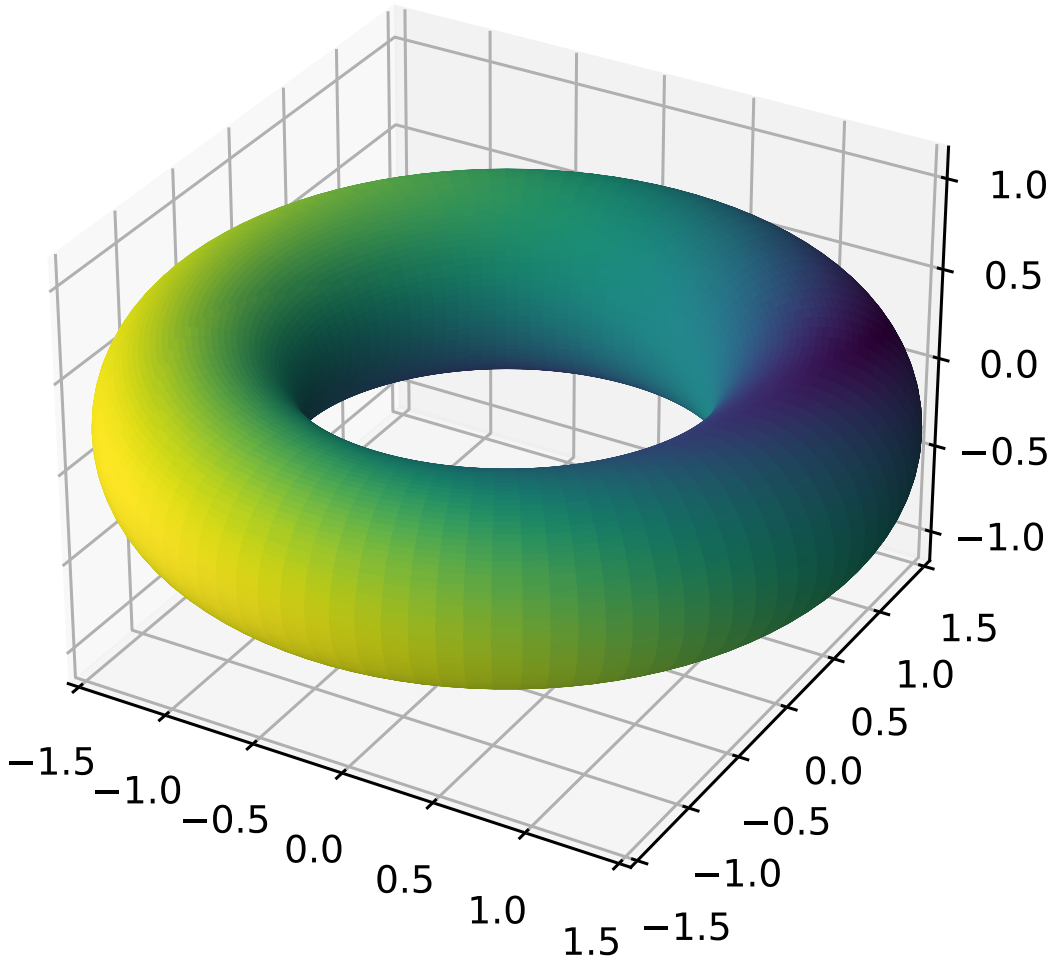


Figure 4.4: Loss function $L_{\mathcal{M}}(\boldsymbol{\tau}, \boldsymbol{x}_{\text{target}})$ on the torus $\mathbb{T}$. This loss function possesses only two minimizers which correspond to analytical solutions.

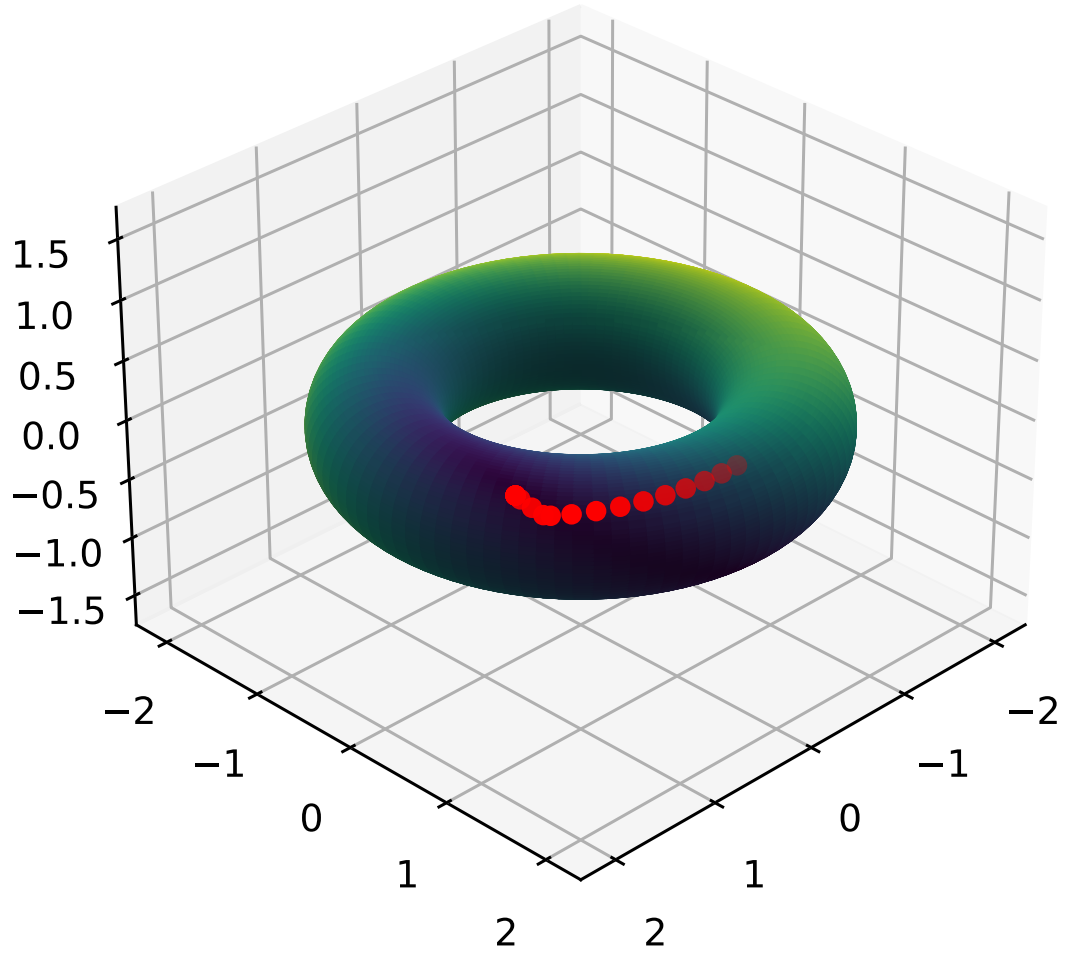
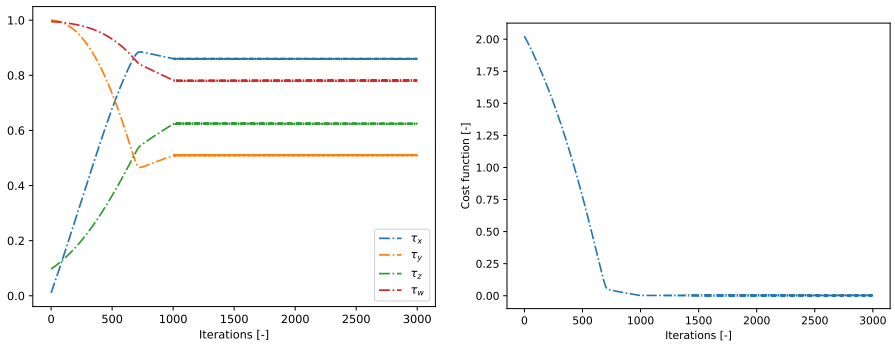


Figure 4.5: Optimization trajectory on loss function $L_{\mathcal{M}}(\boldsymbol{\tau}, \boldsymbol{x}_{\text{target}})$ on the torus $\mathbb{T}$. We perform 3000 iterations with a constant step size of $\alpha = 10^{-3}$. The retraction used is the exponential map of the torus.



(a) Optimization trajectory of the variable $\boldsymbol{\tau} = [\tau_x, \tau_y, \tau_z, \tau_w]$.

(b) Optimization trajectory of the cost function.

Figure 4.6: Manifold optimization on $\mathbb{T}$. We perform 3000 iterations with a constant step size of $\alpha = 10^{-3}$. The retraction used is the exponential map of the torus. The cost function $L_{\mathcal{M}}(\boldsymbol{\tau}, \boldsymbol{x}_{\text{target}})$ is the distance between the current position of the end-effector and the target position.

Part II

ROBOTIC GRASPING: A REVIEW

5

ROBOTIC GRASPING: A REVIEW

Outline

This chapter provides a literature review of robotic grasping. Starting with deterministic approaches based on Newtonian mechanics, we continue the review of recent developments of machine learning-based approaches. Basic concepts and notations are described as well as the formulation of the grasping problem.

5.1 INTRODUCTION

Grasping is the process of picking up an object by applying forces and torques at a set of contacts [Newbury et al., 2023]. It is a fundamental skill to control the object's motion. Humans can handle a large variety of objects and reach nearly perfect grasps at each try.

While grasping is a natural skill for humans, this task is very difficult for robots. Finding a good grasp pose is a high-dimensional search in the space of object-gripper poses and gripper joint configurations. The complexity of the search increases with many factors; the number of degrees of freedom of the pose (4 DoF or 6 DoF), the complexity of the gripper (parallel jaw or multi-fingered grippers), the physical mechanism to apply forces (friction, magnetism, etc.) but also the shape of the objects, the number of objects to grasp and how the objects are placed in the scene (structured or cluttered scene).

Robotic grasping is a well-known problem in robotics with more than four decades of research. We shall briefly present the evolution of methods to solve this problem.

Analytical approaches vs data-driven approaches

We denote approaches based on mechanical principles as *analytical approaches* in contrast to *data-driven approaches*. While both use mathematical models and analytical expressions, they differ in the sense that data-driven approaches are mathematical models in which the relations between inputs and outputs are based on the user's choices while analytical approaches are based on physics principles. While the term "physics-based approaches" is more appropriate, we stick to "analytical approaches" because it is the more common term in the literature.

5.2 EARLY DAYS: ANALYTICAL APPROACHES

In the early days of robotic grasping, a framework based on contact models was proposed to form the basis of analytical approaches [Murray et al., 1994; Shimoga, 1996; Bicchi and Kumar, 2000]. Let us introduce the framework. A force $\boldsymbol{f}_i$ applied on an object at the contact point p_i by the fingertips generates a torque $\boldsymbol{\tau}_i = p_i \times \boldsymbol{f}_i$ with respect to the object's center of mass (CM). Forces and torques can be concatenated into a *wrench* $w_i = (\boldsymbol{f}_i, \boldsymbol{\tau}_i/\rho)$, ρ being a constant that defines the metric of the wrench space, allowing to express a wrench as forces in Newton. There are several possible choices for ρ with different implications [Roa and Suárez, 2009].

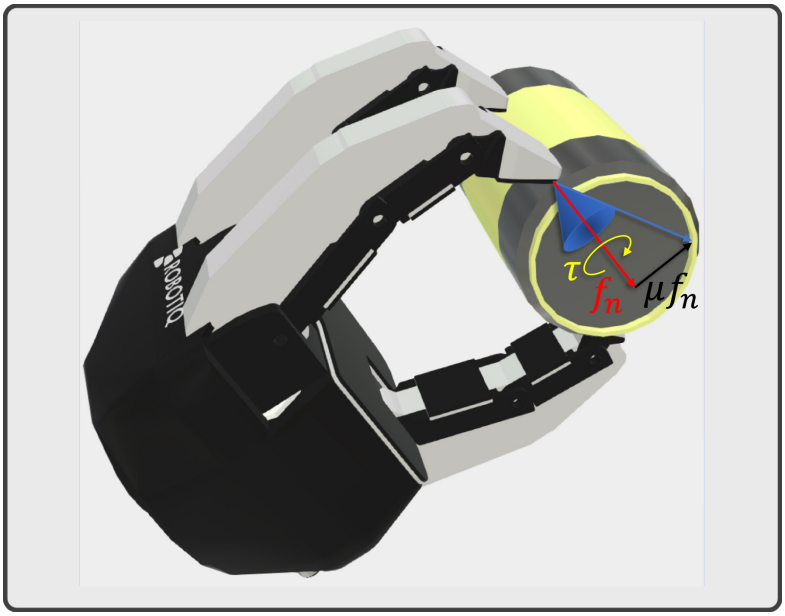


Figure 5.1: Model of the soft contact [de Souza et al., 2021]. Normal component f_n, tangential component due to friction $f_t = \mu f_n$ and torque $\boldsymbol{\tau}$.

The components of the force applied at a contact point will depend on the contact model. *Frictionless* contacts rely on the assumption that the force will always be normal to the contact surface. *Frictional* contacts rely on the assumption that the force has a normal component to the contact surface and may have a tangent component due to friction. According to Coulomb's dry friction model, the tangential force should belong to the friction cone. In *soft* contacts, a torque around the direction normal to the contact surface is included as a corrective term to better model the fact that contact points are contact areas in practice (Fig 5.1). This model only applies to 3D objects. We can now introduce the fundamental equation of grasping, which relates the forces applied at the contact points to the total wrench applied at the object's CM w_o as

$$w_o = G\boldsymbol{f}, \tag{5.1}$$

where $\boldsymbol{f} = [f_{1k}^T, \ldots, f_{ik}^T]$ with i the number of contact points and $k = 1, \ldots, r$ with $r = 1$ for frictionless contact, $r = 2$ and $r = 3$ for frictional contact in 2D and 3D space, $r = 4$ for soft contact. G is the grasp matrix or the grasp map. It is a fundamental quantity because it allows us to directly compute the influence of a force applied at a contact point to the wrench on the object's CM. Equation (5.1) is the basis of analytical approaches. We can choose the contact and thus the gripper configurations to maximize or minimize quality metrics [Roa and Suárez, 2015] based on this equation. The most famous one is the ε-metric [Canny and Ferrari, 1992], which measures the largest perturbation wrench w_{ext} that the grasp can resist in any direction,

$$w_{\text{o}} + w_{\text{ext}} = 0. \tag{5.2}$$

and is directly related to the *force-closure* property.

Surprisingly, early work focused more on multi-fingered robotic hands than parallel-jaw grippers. This choice is motivated by the fact that parallel-jaw grippers limit the possibility of grasping objects with complex shapes [Shimoga, 1996].

While this approach is appealing because of the theoretical guarantees, it quickly shows its limitations. These methods assume the perfect knowledge of objects, their geometrical shape, and physical properties. However, these quantities are rarely directly observable in practice and can only be inferred from partial observations. Moreover, quality metrics are not good predictors for grasping success in the real world [Balasubramanian et al., 2012; Weisz and Allen, 2012]. More specifically, Weisz and Allen [2012] show that quality metrics perform poorly with grasp pose uncertainty. These limitations motivated the development of new methods based on learning algorithms.

5.3 THE RISE OF MACHINE LEARNING: DATA-DRIVEN APPROACHES

Data-driven approaches, or empirical approaches, use data to construct object representations, and make processing such as feature extraction, similarity metrics, pose estimation and so on [Bohg et al., 2013]. Because these methods require lots of data, they were not so much popular until the release of the GraspIt! simulator in 2004 [Miller and Allen, 2004]. For example, Miller et al. [2003] use grasp configurations that have a good quality metric for an object with a given preshape and try to fit new shapes into preshapes. Pelossof et al. [2004] use a support vector machine to estimate the grasp quality from object shape parameters and grasp pose and optimize the learned function to maximize the grasp quality metric given object shape parameters.

However, as said before, grasp quality metrics perform poorly in real-world setups. Several researchers proposed to use empirical experience to learn how to grasp objects [Morales et al., 2004; Montesano et al., 2008; Detry et al., 2009]. The question addressed in these studies was to determine the object features that are necessary to infer accurately grasp poses. When the object to grasp is known, the goal is to estimate the object pose and then find grasp poses by their reachability within a database. For ex-

ample, Detry et al. [2009] use human grasps to build object empirical grasp density from which grasp poses can be sampled. After 2010, the progress of data-driven approaches was also boosted thanks to developments in the area of 3D sensing. The Kinect [Zhang, 2012] was released in 2010 and became very popular in this field due to its low price and simple usage. RGB-D images, color, and depth images started to be used in robotic grasping [Jiang et al., 2011].

This period revealed methods that are capable of grasping *unknown* objects, which was impossible for analytical approaches. These methods mainly use hand-crafted features [El-Khoury et al., 2007; Kyota et al., 2005; Michel et al., 2006] and still have some limitations: only one object can be grasped, the unknown object should remain very similar to objects used for training, etc. In 2012, AlexNet [Krizhevsky et al., 2012] was published, making deep learning [LeCun et al., 2015] the backbone of all machine learning methods.

5.4 DEEP LEARNING FOR ROBOTIC GRASPING

Deep learning has become a core method for robotic grasping. We can divide its use into three methods: *sampling*, *direct regression*, and *reinforcement learning* [Newbury et al., 2023].

5.4.1 *Sampling*

Sampling approaches use the information contained in a sample, *i.e* containing the observation, the grasp pose, and its quality(I, g, Q), to decide on the optimal grasp pose. These methods typically work by predicting the quality of the grasp pose given an observation and then optimizing the quality with respect to the grasp pose. Thus, these methods require (i) a sampling procedure (ii) a model to predict the quality (iii) an optimization scheme. Then, the approach can be summarized with these three steps:

(a) Sample a grasp pose $g \sim p(g)$

(b) Evaluate the quality of the grasp pose g given the observation I with the model $Q = f_\theta(g, I)$

(c) Optimizing the quality $g^* = \arg\max_g f_\theta(g, I)$

with g being the grasp pose, f_θ being the function approximator (neural networks in most cases), Q being the quality of grasp, and I being the observation.

5.4.1.1 *Generating samples*

Sampling grasp poses is mostly done in the workspace, meaning $\mathbb{R}^3 \times \mathbb{SO}(2)$ or $\mathbb{R}^3 \times \mathbb{SO}(3)$ [Eppner et al., 2019]. Grasping is complex because only a very small region of the

grasp space will lead to a successful grasp. Therefore, several sampling strategies were used to generate quickly useful grasp poses and cover all successful grasp poses. Many taxonomies exist to classify grasp sampling strategies. We split strategies into two main categories: analytical ones, which provide the density of the grasp poses, and heuristic ones, which use handcrafted procedures to generate samples.

Analytical samplers are mostly represented by uniform samplers. They sample from the bounded $\mathbb{R}^3 \times \mathbb{SO}(3)$ space uniformly. They are not effective in generating useful samples but they cover all the space, making them unbiased.

Heuristic samplers are based on the surface information of the object. They parameterize the grasp pose with the hand's approach vector. Eppner et al. [2019] show that they are effective in generating useful samples but are biased, meaning that they do not fully cover all possible grasps of an object.

In this category, we can again make a distinction between approach-based samplers and antipodal samplers. *Approach-based* samplers work as follows: a point is sampled on the object surface and the normal to this point is evaluated. This normal vector corresponds to the approach vector of the gripper. From that, many variations exist [Diankov, 2010; Kappler et al., 2015; Veres et al., 2017]. *Antipodal* samplers try to sample directly contact points between the object and the hand as in [Smith et al., 1999; Mahler et al., 2017; Ten Pas and Platt, 2018](Fig 5.2). They are effective in generating useful grasp poses but they are biased. However, antipodal samplers do not scale well with multi-fingered grasping.

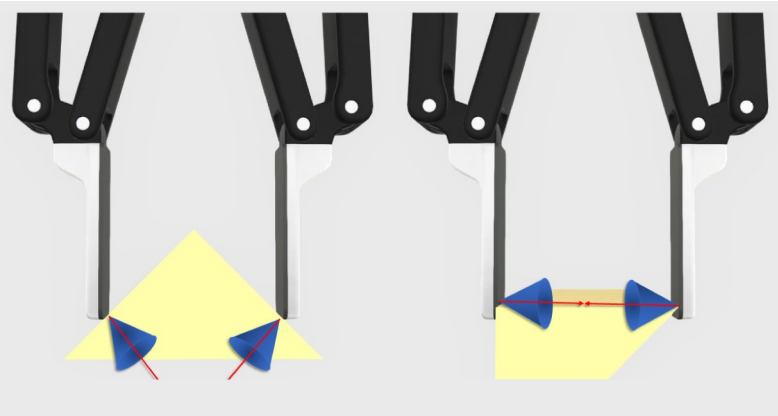


Figure 5.2: Antipodal grasps [de Souza et al., 2021]. The line that connects the two contact points must lay in both friction cones. It generates grasp poses from depth images, as explained in [Mahler et al., 2017].

All these strategies became viable when robotic simulators became accessible. Common simulators are GraspIt! [Miller and Allen, 2004] and Simox [Vahrenkamp et al., 2013]. Moreover, simulators may be used to test a new algorithm.

5.4.1.2 *Evaluating samples*

Sampling approaches use a function to evaluate the "quality" of the sampled grasp poses. This grasp quality can be approximated by a grasp quality metric from analytical approaches or the likelihood of a grasp being successful. Some approaches can directly sample from a region with a high grasp quality while other approaches refine the solution using optimization-based techniques.

Binary classification These methods predict the success of a grasp pose as a binary problem, meaning that successful grasps are classified as 1 and failed grasps as 0. For example, a Convolutional Neural Network [LeCun et al., 1995] can be trained to classify if grasp parameters are successful or not [Aktaş et al., 2022; Kappler et al., 2015; Mousavian et al., 2019].

Learning a metric Instead of predicting a probability of success, learning metrics of a grasp allows one to have more information about the quality of the grasp. For example, Varley et al. [2015] learn a series of heatmaps, in which each pixel represents the quality of grasp.

5.4.1.3 *Optimizing sample*

Because deep neural networks are fully differentiable with respect to their inputs, the gradient of the quality of grasps with respect to grasp poses can be computed efficiently. Therefore, starting from an initial sample, we can maximize the quality of grasp with gradient-based optimization and refine iteratively the grasp parameters. Zhou and Hauser [2017] train a CNN to predict the grasp quality given the grasp poses and a depth map and use a quasi-Newton method to optimize the grasp poses. Lu et al. [2020] find the grasp pose that maximizes the probability of success, starting with a sample drawn from the prior.

5.4.2 *Direct Regression*

Direct regression approaches tackle the problem of computing the optimal grasp by directly outputting it from sensor data. They are often framed as an end-to-end solution. Compared to sampling approaches, they are faster because they require only one forward pass of the learned model. The downside is that they only provide point estimates for the grasp pose and do not capture the full distribution. Thus, this approach can be stuck if the outputted grasp pose leads to an unreachable pose. Most of the time, they approximate the maximum likelihood estimate (MLE) of the probability of success given a grasp pose and an observation $p(S \mid g, I)$. Because it is a complex problem to infer

the optimal grasp pose for 6 DoF, these approaches tend to use multi-stage pipelines. In contrast to sampling approaches, they required only one step:

(a) Compute $g = f_\theta(I)$

with g being the grasp parameters, f_θ being the function approximator (neural networks in most cases), and I is the observation.

5.4.2.1 *Direct regression of the pose*

These approaches output the 6 DoF grasp pose leading to a successful grasp. For example, Schmidt et al. [2018] train a CNN to output the translation part and the rotation part from depth images. Yang et al. [2021] train a neural network to estimate the end-effector pose represented as a 4×4 transformation matrix. These approaches benefit from pose estimation but face another problem: the multi-modality behavior of grasping. Some authors try to avoid it by learning the best grasp poses which are the nearest from the current output [Liu et al., 2019; Min et al., 2020].

5.4.2.2 *Multi-stage approach*

Multi-stage approaches split the problem of inferring the grasp parameters into several stages. Each stage has its own loss function and processes one specific task. Most multi-stage approaches consider three stages [Fang et al., 2020; Wang et al., 2021; Zhao et al., 2021; Zhu et al., 2021] while Wei et al. [2021] consider only two stages.

The first stage aims at producing a first estimation of the grasp parameters by predicting grasp quality for subsampled points, estimating the grasp for each point, or estimating reduced DoF. The middle stage creates grasp proposals or further estimates a subset of the DoF. The last stage acts as a refinement stage which improves the grasps generated at the previous stage.

5.4.3 *Reinforcement Learning*

Reinforcement learning (RL) aims at learning an optimal, or nearly optimal, policy that maximizes a reward function. The policy π gives the best action to do in a given state with a given observation. In robotics grasping, actions usually represent a relative displacement for the end-effector. Moreover, the opening or closing of the gripper is also predicted by the network. Some additional information such as contact points can also be given to the network. RL completely differs from sampling and direct approaches because it produces a trajectory of actions and not just the final state. A comprehensive review of RL in grasping explaining its open challenges can be found in [Mohammed et al., 2020] with notable works [Wu et al., 2020, 2019; Berscheid et al., 2021; Merzić et al., 2019; Song et al., 2020; Tang et al., 2021].

5.4.4 *Large language models for robotics*

Language modeling is a widely used approach for language understanding and generation [Zhao et al., 2023]. Large language models (LLM) are very large neural networks that are pretrained on large datasets. Research has shown that scaling the size of the model leads to better performance as well as new capabilities exhibited by the model. ChatGPT-3 [Brown et al., 2020] shows impressive *in-context learning*.

Because LLM encode semantic information, this can be used in robotics to perform tasks expressed in a high-level language. Ichter et al. [2023] use such methods and achieve a very high success rate for various tasks. Ichter et al. [2023] open the use of LLM in robotics and are followed by [Singh et al., 2023; Huang et al., 2023, 2022; Zeng et al., 2022]. This is still an active area of research.

5.5 SUMMARY

Robotic grasping has a rich history. Starting from analytical approaches, we showed that these methods are not suitable for many practical applications due to their need for precise knowledge of the interactions between the gripper and the object. To overcome this limitation, the community has explored data-driven approaches to replace deterministic functions with approximations based on collected data. Then, deep learning turned out to be very effective in performing some tasks and it became quickly a new standard in robotic grasping. Today, two approaches coexist: deep learning approaches which predict the final grasp pose, and reinforcement learning approaches, which provide a series of actions to perform to grasp objects.

However, robotic grasping is not yet solved and there are still open questions. Dealing with uncertainties becomes a major concern since the working environments of the robots become more and more stochastic. While direct regression approaches are fully deterministic and thus do not deal with uncertainties, sampling approaches model the result of a grasp as a random variable. However, the grasp pose is still treated as parameters, which cannot capture the uncertainties about the grasp pose itself. This issue can be overcome by modeling the grasp pose as a random variable and using Bayesian inference to propagate uncertainties while acquiring new observations. Furthermore, most of the current works choose the parametrization of the orientation based on the DoF and the type of their grippers, meaning that their method may be difficult to use with other hardware. As a result, their modeling does not take into account the geometry of the rotation space, leading to potential issues for learning surrogates of the likelihood and difficulties in the optimization step.

Part III

SIMULATION-BASED INFERENCE FOR ROBOTIC GRASPING

6

HARDWARE

Outline

This chapter reviews and illustrates the hardware used in the contributions. Hardware makes robotics a unique playground for researchers because it integrates many different elements into one system: software, sensors, communications, actuators, motion, and so on. While many algorithms work in simulated environments, a percentage of them fail to bridge the gap to the real world.

6.1 ROBOTIC ARM

The robotic arm used in our setup is a Universal Robot 5, called the UR5 (Fig.6.1). It has 6 degrees of freedom (DoF), like most industrial robots. The Universal Robots company names their robots with a number corresponding to their payload capacity. In our case, the UR5 has a payload of 5 kg. The reach of the UR5 is 850 mm when all the links of the arm are aligned. Surprisingly, the arm only weighs 18.4 kg, which is quite low given the payload of 5 kg.

The URScript Programming Language serves as an interface to code the robot through a touchscreen teach pendant. The TCP/IP interface allows external programs to communicate with the controller and send instructions. This is particularly useful when the application needs to access other devices or services. For example, the Robot Operating System (ROS) communicates with Universal Robots through this channel.

6.2 GRIPPER

In all our experiments, we used a Robotiq 3-finger adaptive robot gripper (Fig.6.2). This gripper weighs 2.3 kg and has a large opening of 155 mm. It produces a grip force between 30 and 70 N, which helps to grasp heavy objects. This robotic hand has three fingers: a thumb and two fingers. Each finger has three links but only one motor at the base to close them, making them underactuated. While the thumb stays fixed, the two fingers can move sideways, with the addition of one motor.

It is important to mention that the internal mechanism for closing the fingers makes it very difficult to simulate the controller in virtual environments. This leads to poor simulation fidelity and challenges in transferring from simulation to the real setup.

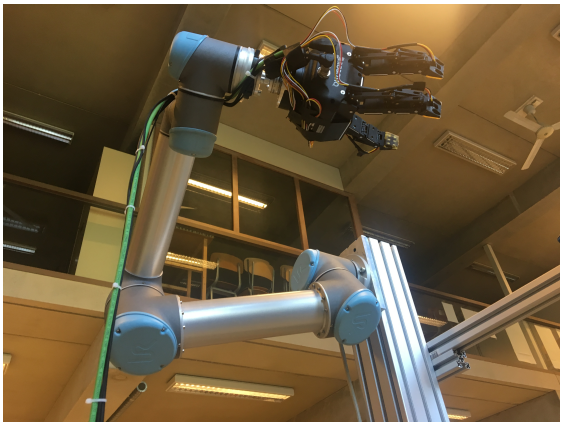


Figure 6.1: The UR5. It is a 6 DoF robotic arm with 5kg of payload and 850mm of reachability.

The gripper is controlled via ROS by a TCP/IP connection.

Figure 6.2: The Robotiq 3-finger adaptive robot gripper. It possesses several control modes such as force, position and speed. This gripper provides a lot of flexibility and versatility.

6.3 DEPTH CAMERAS

Depth cameras are commonly used in robotics nowadays. They provide 3D information presented as depth images and point clouds, and most of the time RGB images as well. These features make them the perfect sensors for robots because segmentation, object recognition, pose estimation, etc., are possible with this sensor data. In addition, depth images are well simulated in virtual robotics environments, unlike RGB images.

We used two depth cameras in our experiments. We used the Kinect for the first contribution and the Intel RealSense D435i for the next three contributions. In fact, the Intel RealSense provides much more accurate depth images.

6.3.1 *Kinect*

The Kinect was first introduced by Microsoft in 2008 and released in Europe in 2010. The Kinect was designed for video games but quickly attracted the interest of robotics researchers. It provides depth images with a resolution of 640×480 at a rate of 30 images per second. The range of the Kinect is about 1.5 to 3.5 meters. The Kinect is designed to work indoor and not outdoor. Open-source drivers exist, notably used by ROS.

Figure 6.3: The Kinect. It produces 640×480 depth images at the rate of 30 images per second.

6.3.2 *Intel Realsense D435i*

The Intel RealSense D435i (Fig.6.4) offers many new features compared to the Kinect. An IMU is integrated into the camera to perform SLAM (simultaneous localization and mapping) or tracking. This depth camera provides up to 1280×720 depth images at a rate of 60 images per second. Its range is about 0.3 to 3 meters. The software for the Intel RealSense is well developed and integrated into ROS.

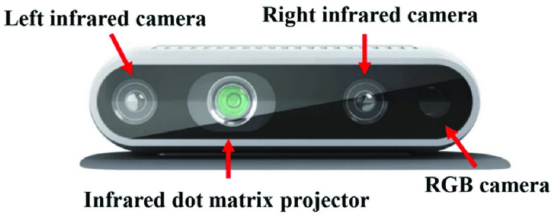


Figure 6.4: The Intel Realsense D435i. This depth camera provides accurate depth images at higher resolution than the Kinect at a highest rate.

7

GRASPING A SINGLE OBJECT IN A FIXED POSE

Outline

We demonstrate the usefulness of simulation-based inference algorithms in robotics. We introduce our modeling of grasping and how to turn it into a Bayesian posterior inference. While current approaches only approximate the likelihood function, and thus do not put probability distribution on the grasp pose, we model the grasp pose as a random variable and leverage robotics simulators to learn the likelihood-ratio function with fully differentiable neural networks. We then compute the maximum a posteriori (MAP) of the posterior distribution with gradient descent. However, the search space is not a vector space because of the rotation component. Therefore, we use Riemannian manifold optimization, thus preserving the geometry of the rotation space. Experiments in simulation and a real setup show promising results.

7.1 PROLOGUE

Machine learning in robotics has progressed a lot due to the recent development of robotics simulators [Coumans and Bai, 2016–2023; Todorov et al., 2012; Liang et al., 2018]. They allow practitioners to generate data and test their methods and algorithms in simulation without risking hardware issues and costs. Reinforcement learning has also pushed toward the development of benchmarks in simulation. This facilitates the use of robotics simulators.

Simulation-based inference algorithms on the other hand have enhanced due to progress in deep learning. It becomes possible to perform the full Bayesian posterior inference, which contrasts with the most common approach, *i.e.* learning a surrogate of the likelihood. Therefore, the maximum a posteriori can used as a point estimator instead of the maximum likelihood estimator.

In robotics, rotations have been handled most of the time with Euler angles which may lead to gimbal lock and singularities. Our method deals nicely with rotations and does not assume that it is a 2D or 3D rotation.

We unified all these concepts into a global framework and demonstrated its usefulness for robotic grasping.

Reading tips Section 2 may be skipped because most of the material was introduced in the background part. The reader interested in the origin of the methodology should

look at Hermans et al. [2020] for the neural ratio estimation, Detry et al. [2017] for the robotic modeling, and Boumal [2013] for manifold optimization.

7.2 THE PAPER: SIMULATION-BASED BAYESIAN INFERENCE FOR MULTI-FINGERED ROBOTIC GRASPING

SIMULATION-BASED BAYESIAN INFERENCE FOR MULTI-FINGERED ROBOTIC GRASPING

Norman Marlier
University of Liège
norman.marlier@uliege.be

Olivier Brüls
University of Liège
o.bruls@uliege.be

Gilles Louppe
University of Liège
g.louppe@uliege.be

ABSTRACT

Multi-fingered robotic grasping is an undeniable stepping stone to universal picking and dexterous manipulation. Yet, multi-fingered grippers remain challenging to control because of their rich nonsmooth contact dynamics or because of sensor noise. In this work, we aim to plan hand configurations by performing Bayesian posterior inference through the full stochastic forward simulation of the robot in its environment, hence robustly accounting for many of the uncertainties in the system. While previous methods either relied on simplified surrogates of the likelihood function or attempted to learn to directly predict maximum likelihood estimates, we bring a novel simulation-based approach for full Bayesian inference based on a deep neural network surrogate of the likelihood-to-evidence ratio. Hand configurations are found by directly optimizing through the resulting amortized and differentiable expression for the posterior. The geometry of the configuration space is accounted for by proposing a Riemannian manifold optimization procedure through the neural posterior. Simulation and physical benchmarks demonstrate the high success rate of the procedure.

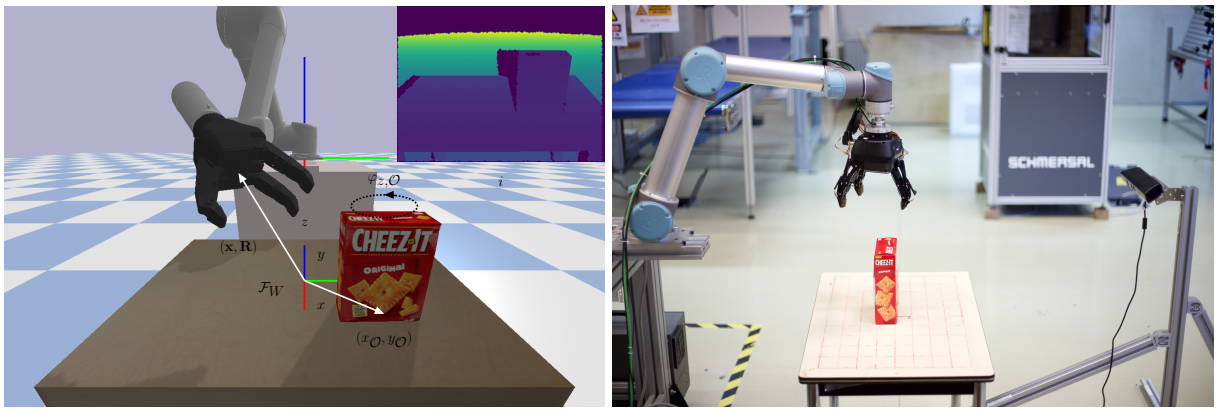
***K*eywords** Multi-fingered grasping, Bayesian inference, Robot learning

Two of the grand challenges for the deployment of robots in warehouses, assembly lines or homes are universal picking, *i.e* the ability to robustly grasp a large variety of objects in diverse environments, and dexterous manipulation, *i.e* the ability to manipulate objects to perform useful actions. Multi-fingered robotic grasping represents a promising avenue towards these objectives and has the potential to greatly improve and facilitate human-machine interactions. However, due to the wide variety of object shapes, sensor noise and nonsmooth contact dynamics, multi-fingered grippers remain challenging to control. In addition, multi-fingered grasps entail high dimensional configuration spaces compared to two-fingered grasps, making them difficult to optimize and plan.

Early analytical approaches [1, 2] for planning multi-fingered grasps rely on force analysis where a metric is optimized based on the laws of mechanics and 3D models. These analytical methods require the accurate knowledge of many model and world parameters, which is practically difficult to achieve in real settings. By contrast, learning-based methods are getting more and more established because they can extract useful information from raw sensor data [3]. These methods typically make use of machine learning or deep learning models for learning a mapping between grasp configurations and their success. This strategy has been applied on top-down grasping with a parallel jaw gripper and has demonstrated excellent results [4, 5, 6, 7, 8], including its extension to 6 DOF [9, 10].

In this work, we consider the more challenging setting of robust multi-fingered grasping plans, including the 6 DOF grasp poses and the finger configuration. By framing the problem as an inference task, we demonstrate the generic usefulness and applicability of likelihood-free Bayesian inference algorithms to difficult robotic tasks such as multi-fingered grasping considered here. We summarize our contributions as follow:

- We bring simulation-based Bayesian inference methods [11] to multi-fingered grasping. By learning a model for the likelihood-to-evidence ratio and using an analytical prior, we derive an amortized and differentiable posterior for the hand configurations.
- We make use of Riemannian manifold optimization to deal with the nonlinearity of the configuration space, in particular of the 3D rotation group tied to the hand pose.



(a) (b)

Figure 1: Our benchmark scene for multi-fingered grasping. The pose of the hand $(\mathbf{x}, \mathbf{R})$ is defined in the local object frame. The depth camera produces an image i of the scene.

- We validate our approach on a realistic and challenging 3-finger gripper setup, through both simulation and real-world physical benchmarks. Results demonstrate its high success rate.

1 Problem statement

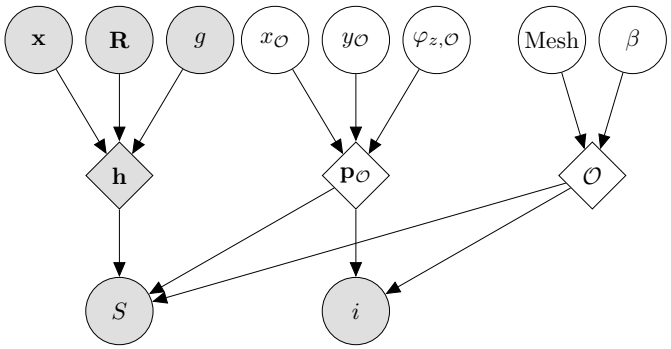
Description We consider the problem of grasping a rigid and stable body with a multi-fingered gripper, as illustrated in Fig. 1. The object $\mathcal{O}$ is modelled as a 3D surface mesh and its centroid stands on a table at a location $(x_\mathcal{O}, y_\mathcal{O}, z_\mathcal{O})$ with a rotation $\varphi_{z,\mathcal{O}}$ around the z-axis in the world reference frame $\mathcal{F}_W$. We refer to its 2D pose $(x_\mathcal{O}, y_\mathcal{O}, \varphi_{z,\mathcal{O}})$ as $\mathbf{p}_\mathcal{O} \in \mathbb{R}^2 \times \text{SO}(2)$. The hand configuration $\mathbf{h} \in \mathcal{H} = \mathbb{R}^3 \times \text{SO}(3) \times \mathcal{G}$ is defined as the combination of the pose $(\mathbf{x}, \mathbf{R}) \in \mathbb{R}^3 \times \text{SO}(3)$ of the hand and the type $g \in \mathcal{G} = \{\text{basic}, \text{wide}, \text{pinch}\}$ of the grasp. The hand pose $(\mathbf{x}, \mathbf{R})$ is defined with respect to the world frame coordinate. The robot evolves in a 3D workspace observed with a fixed depth camera producing images $i \in \mathcal{I}$. The goal is to find a robust hand configuration $\mathbf{h}^*$ with respect to a given binary metric $S = \{0, 1\}$, where $S = 1$ indicates a successful grasp.

Probabilistic modeling We model the scene and the grasping task according to the Bayesian network shown in Fig. 2a. The variables $S, i, \mathbf{h}, \mathcal{O}$ and $\mathbf{p}_\mathcal{O}$ are modelled as random variables in order to capture the noise in the robot or in the depth camera, as well as our prior beliefs about the hand configuration, the object or its pose. The structure of the Bayesian network is motivated by the fact that $\mathbf{h}$, $\mathcal{O}$ and $\mathbf{p}_\mathcal{O}$ are independent, while S is dependent on $\mathbf{h}$, $\mathcal{O}$ and $\mathbf{p}_\mathcal{O}$ and i is dependent on $\mathcal{O}$ and $\mathbf{p}_\mathcal{O}$. This structure also enables a direct and straightforward way to generate data: $\mathbf{h}$, $\mathcal{O}$ and $\mathbf{p}_\mathcal{O}$ are sampled from their respective prior distributions while S and i can be generated using forward physical simulators for the grasping and the camera.

The prior distribution $p(\mathbf{x})$ of the spatial position is uniformly distributed between the extreme values $\mathbf{x}_{\text{lim}} = (x_{\text{low}}, y_{\text{low}}, z_{\text{low}}, x_{\text{high}}, y_{\text{high}}, z_{\text{high}})$, chosen to be within the range of physical dimensions of the gripper and the biggest object. It emphasizes our ignorance about interesting regions of space for grasping. The rotation $\mathbf{R}$ is parameterized with a quaternion. A quaternion $\mathbf{q}$ is an element of the quaternion group $\mathbb{H}$, in the form $\mathbf{q} = q_0\mathbf{1} + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k} = (q_0, q_1, q_2, q_3)^T$ with $(q_0, q_1, q_2, q_3)^T \in \mathbb{R}^4$ and $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$. The conjugate $\bar{\mathbf{q}}$ of quaterion $\mathbf{q}$ is given by $\bar{\mathbf{q}} := q_0\mathbf{1} - q_1\mathbf{i} - q_2\mathbf{j} - q_3\mathbf{k}$. A unit quaternion, called *versor*, $\mathbf{q}_1 \in \mathbb{H}_1$ has a unit norm defined as $\|\mathbf{q}\| = \sqrt{\mathbf{q}\bar{\mathbf{q}}} = 1$. They give a more compact representation than rotation matrices and avoid gimbal lock and singularities. Unit quaternions can be identified with the elements of a hyperspherical manifold $\mathbb{S}^3$ embedded into $\mathbb{R}^4$. Moreover, $\mathbb{S}^3$ is a double covering of SO(3), meaning that antipodal points $\pm\mathbf{q}$ represent the same rotation, which implies that $p(\mathbf{q}; \cdot) = p(-\mathbf{q}; \cdot)$. We define the prior $p(\mathbf{q})$ as a mixture of *power-spherical* distributions [12] with 4 modes μ_i. Each mode is a mixture that satisfies $p(\mathbf{q}; \cdot) = p(-\mathbf{q}; \cdot)$. In total, we have

$$p(\mathbf{q}) = \frac{1}{N}\sum_{i=1}^{N=4} \frac{\text{PowerSpherical}(\mathbf{q}; \mu_i, \kappa)}{2} + \frac{\text{PowerSpherical}(\mathbf{q}; -\mu_i, \kappa)}{2}. \tag{1}$$

These modes μ_i encode information about the orientation of the gripper and share the same concentration factor $\kappa = 30$. To grasp an object, the gripper point toward the table and thus toward the object – an informed prior which indeed



(a)

Variable	Prior
x	uniform$(-0.15, 0.15)$
y	uniform$(-0.15, 0.15)$
z	uniform$(0.12, 0.34)$
$\mathbf{R}$	mixture of power spherical(μ_i, κ)
g	categorical$(\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\})$
$x_\mathcal{O}$	uniform$(-0.05, 0.05)$
$y_\mathcal{O}$	uniform$(-0.05, 0.05)$
$\varphi_{z,\mathcal{O}}$	uniform$(-\pi, \pi)$
Mesh	uniform in the set of objects
β	uniform(0.9, 1.1)

(b)

Figure 2: (a) Probabilistic graphical model of the environment. Gray nodes correspond to observed variables and white nodes to unobserved variables. (b) Prior distributions.

results in sufficiently many successful grasps. We then define four rotations, separated by a rotation of $\frac{\pi}{2}$ around the z-axis (see Fig. 5 in Appendix B). In this way, our prior covers a large part of the rotation space and is sufficiently informative by contrast to a uniform prior over the unit sphere $\mathbb{S}^3$. The grasp type g is uniformly distributed between the three types basic, wide and pinch. These three modes modulates the spacing between the fingers in the opposite side of the thumb. Finally, $p(\mathbf{h}) = p(\mathbf{x})p(\mathbf{R})p(g)$.

The prior $p(\mathcal{O}) = p(\text{Mesh})p(\beta)$ is a discrete uniform distribution over a fixed set of object meshes and a uniform distribution for the scaling factor β. Finally, the prior $p(\mathbf{p}_\mathcal{O})$ captures our belief that the object can be everywhere on the table with any rotation around the vertical axis. For this reason, uniform distributions are used for all three parameters $x_\mathcal{O}, y_\mathcal{O}, \varphi_{z,\mathcal{O}}$. Table 2b summarizes the prior distributions.

Given our probabilistic graphical model, we finally formulate the problem of grasping as the Bayesian inference of the hand configuration $\mathbf{h}^*$ that is a posteriori the most likely given a successful grasp and an observation i. That is, we are seeking for the maximum a posteriori (MAP) estimate

$$\mathbf{h}^* = \arg\max_{\mathbf{h}} \; p(\mathbf{h}|S = 1, i). \tag{2}$$

2 Likelihood-free Bayesian inference for multi-fingered grasping

2.1 Density ratio estimation

From the Bayes's rule, the posterior of the hand configuration is

$$p(\mathbf{h}|S, i) = \frac{p(S, i|\mathbf{h})}{p(S, i)} p(\mathbf{h}). \tag{3}$$

The likelihood function $p(S, i|\mathbf{h})$ and the evidence $p(S, i)$ are both intractable, which makes standard Bayesian inference procedures such as Markov chain Monte Carlo unusable. However, drawing samples from forward models remains feasible with physical simulators, hence enabling likelihood-free Bayesian inference algorithms.

First, we express the likelihood-to-evidence ratio as a product of two individual ratios,

$$r(S, i|\mathbf{h}) = \frac{p(S, i|\mathbf{h})}{p(S, i)} = \frac{p(S|\mathbf{h})}{p(S)} \frac{p(i|S, \mathbf{h})}{p(i|S)} = r(S|\mathbf{h}) r(i|S, \mathbf{h}). \tag{4}$$

By adapting the approach described in [13, 14] for likelihood ratio estimation, we train two neural network classifiers d_ϕ and d_θ that we will use to approximate $r(S|\mathbf{h})$ and $r(i|S, \mathbf{h})$. The first network d_ϕ is trained to distinguish positive tuples $(S, \mathbf{h})$ (labeled $y = 1$) sampled from the joint distribution $p(S, \mathbf{h})$ against negative tuples (labeled $y = 0$) sampled from the product of marginals $p(S)p(\mathbf{h})$. The Bayes optimal classifier $d^*(S, \mathbf{h})$ that minimizes the cross-entropy loss is given by

$$d^*(S, \mathbf{h}) = \frac{p(S, \mathbf{h})}{p(S)p(\mathbf{h}) + p(S, \mathbf{h})}, \tag{5}$$

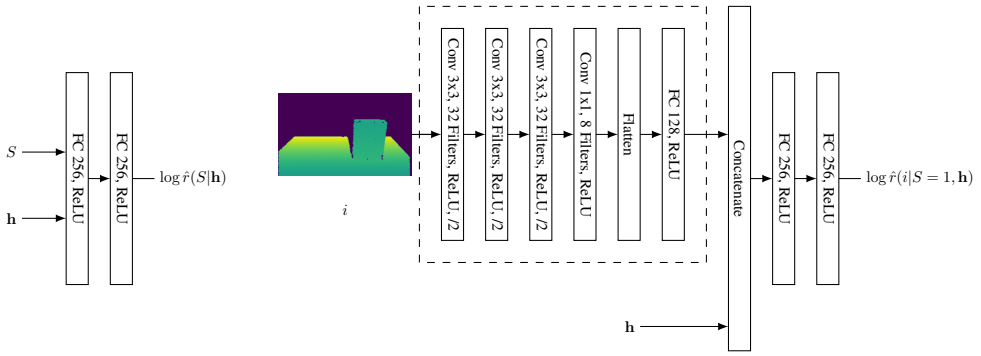


Figure 3: Neural network architectures of the classifiers d_ϕ and d_θ used to respectively approximate the likelihood ratios $r(S|\mathbf{h})$ and $r(i|S=1,\mathbf{h})$.

which recovers the likelihood ratio $r(S|\mathbf{h})$ as

$$\frac{d^*(S,\mathbf{h})}{1-d^*(S,\mathbf{h})} = \frac{p(S,\mathbf{h})}{p(S)p(\mathbf{h})} = \frac{p(S|\mathbf{h})}{p(S)}. \tag{6}$$

Therefore, by modelling the classifier with a neural network d_ϕ trained on the binary classification problem, we obtain an approximate but amortized and differentiable likelihood ratio

$$\hat{r}(S|\mathbf{h}) = \frac{d_\phi(S,\mathbf{h})}{1-d_\phi(S,\mathbf{h})}. \tag{7}$$

The second network d_θ is trained similarly, over positive tuples $(i,\mathbf{h})$ (labeled $(y=1)$ sampled from the conditional joint distribution $p(i,\mathbf{h}|S=1)$ against negative tuples $(i,\mathbf{h})$ (labeled $y=0$) sampled from the product of marginals $p(i|S=1)p(\mathbf{h}|S=1)$. Using the same likelihood ratio trick, we obtain

$$\hat{r}(i|S=1,\mathbf{h}) = \frac{d_\theta(i,\mathbf{h})}{1-d_\theta(i,\mathbf{h})}. \tag{8}$$

Finally, the likelihood ratios are combined with the prior to approximate the posterior as

$$\hat{p}(\mathbf{h}|S=1,i) = \hat{r}(i|S=1,\mathbf{h})\hat{r}(S=1|\mathbf{h})p(\mathbf{h}), \tag{9}$$

which enables immediate posterior inference despite the initial intractability of the likelihood function $p(S,i|\mathbf{h})$ and of the evidence $p(S,i)$.

The neural network classifiers d_ϕ and d_θ are architectured as in Fig. 3. In d_θ, the camera image i of size $640\times480\times1$ is pre-processed by scaling the depths in the interval $\{0\}\cup[0.45,1]$ and resized to $256\times160\times1$. Then, i is fed to a convolutional network made of four convolutional layers followed by a fully connected layer and which goal is to produce a vector embedding of the image. The image embedding and $\mathbf{h}$ are then both fed to a subsequent network made of 2 fully connected layers. The hand configuration $\mathbf{h}$ enters the neural network as a 1×13 vector where the rotation matrix $\mathbf{R}$ is flattened [15, 16] and the grasp type g is passed through an embedding. In d_ϕ, both S and $\mathbf{h}$ are directly fed to a network made of 2 fully connected layers. The parameters ϕ and θ are optimized by following Algorithm 1 (Appendix A), using Adam as optimizer.

Finally, we note that the factorization of the likelihood-to-evidence ratio forces the two ratio estimators to focus their respective capacity on the information brought by S and i. Because of the high discriminative power of S, training instead a single ratio taking both S and i as inputs would indeed lead to an estimator that usually discards the smaller information brought in i.

2.2 Maximum a posteriori estimation

Due to the intractability of the likelihood function and of the evidence, Eq. (2) cannot be solved analytically nor numerically. We rely instead on the approximation given by the likelihood-to-evidence ratio $\hat{r}$ to find an approximation of the maximum a posteriori (MAP) estimate as

$$\hat{\mathbf{h}}^* = \arg\max_{\mathbf{h}} \hat{r}(S=1,i|\mathbf{h})p(\mathbf{h}) \tag{10}$$

$$= \arg\min_{\mathbf{h}} -\log \hat{r}(S=1,i|\mathbf{h})p(\mathbf{h}), \tag{11}$$

which we solve using gradient descent. For a given g, the gradient of Eq. (11) decomposes as

$$-\nabla_{(\mathbf{x},\mathbf{R})} \log \hat{r}(S, i|\mathbf{h}) p(\mathbf{h}) = -\nabla_{(\mathbf{x},\mathbf{R})} \log \hat{r}(S, i|\mathbf{h}) - \nabla_{(\mathbf{x},\mathbf{R})} \log p(\mathbf{h}). \tag{12}$$

Our closed-form prior $p(\mathbf{h})$ has analytical gradients. In fact, uniform distributions are set to have null gradient everywhere in the domain. Therefore, $\nabla_{\mathbf{x}} p(\mathbf{h}) = \mathbf{0}$. By contrast, $p(\mathbf{R})$ is a weakly informative prior and has a non null gradient from the power spherical distribution. Its derivative with respect to $\mathbf{q}$ is

$$\begin{aligned} \nabla_{\mathbf{q}} p(\mathbf{q}; \mu, \kappa) &= C(\kappa)\kappa(1 + \mu^T \mathbf{q})^{\kappa-1} \nabla_{\mathbf{q}}(1 + \mu^T \mathbf{q}) \\ &= C(\kappa)\kappa\mu(1 + \mu^T \mathbf{q})^{\kappa-1}, \end{aligned} \tag{13}$$

where $C(\kappa)$ is the normalization term. Since the likelihood-to-evidence ratio estimator $\hat{r}$ is modelled by a neural network, it is fully differentiable with respect to its inputs and its gradients can be computed by automatic differentiation. However, not all variables of the problem are Euclidean variables and naively performing gradient descent would violate our geometric assumptions. Let us consider a variable $\mathcal{Z}$ on the smooth Riemannian manifold $\mathcal{M} = \mathbb{R}^3 \times \text{SO(3)}$ with tangent space $\mathcal{T}_{\mathcal{Z}}\mathcal{M}$ and a function $f : \mathcal{M} \to \mathbb{R}$. Since SO(3) is embedded in the set of 3×3 matrices $\mathbb{R}^{3\times3}$, f can be evaluated on $\mathbb{R}^3 \times \mathbb{R}^{3\times3}$, leading to the definition of the Euclidean gradients $\nabla f(\mathcal{Z}) \in \mathbb{R}^3 \times \mathbb{R}^{3\times3}$. In turn, these Euclidean gradients can be transformed into their Riemannian counterparts $\text{grad} f(\mathcal{Z})$ via orthogonal projection $\mathbf{P}_{\mathcal{Z}}$ into the tangent space $\mathcal{T}_{\mathcal{Z}}\mathcal{M}$ [17, 18]. Therefore,

$$\text{grad} f(\mathcal{Z}) = \mathbf{P}_{\mathcal{Z}}(\nabla f(\mathcal{Z})) \tag{14}$$

where the orthogonal projection onto $\mathbb{R}^3$ is the identity $\mathbb{I}_3$ and the orthogonal projection onto SO(3) at $\xi \in \text{SO(3)}$ is $\xi\text{skew}(\xi^T \nabla f(\xi))$ where $\text{skew}(A) \coloneqq \frac{1}{2}(A - A^T)$. Thus, we can solve Eq. (11) by projecting Euclidean gradients of Eq. (12) to the tangent space $\mathcal{T}_{\mathcal{Z}}\mathcal{M}$ and plug them to a manifold optimization procedure. In our experiments, we use the geometrical conjugate gradient method [17] implemented in Pymanopt [19] to perform 20 optimization steps and we scan for the best value of g. For completeness, the full optimization algorithm is provided in Algorithm 2 (Appendix A).

3 Experiments

For training, we use 19 objects from the YCB data [20] (see Fig. 6 in Appendix C) together with 5 objects from the ShapeNet dataset [21], for a total of 24 types of objects. We selected a diverse range of objects compatible with the geometry of our gripper. In simulation, the success rate was evaluated on the 19 objects used for training, as well as on 5 new unseen objects from the YCB data (see Fig. 7 in Appendix C). Only the 19 objects from YCB are used in the real setup.

3.1 Data generation

Grasp generative model A physical simulator is used to sample from $p(S|\mathbf{h}, \mathcal{O}, \mathbf{p}_{\mathcal{O}})$. Hand configurations, objects and object poses are sampled from their priors $p(\mathbf{h})$, $p(\mathcal{O})$ and $p(\mathbf{p}_{\mathcal{O}})$, and are then submitted to a lift test. First, a planner generates a trajectory in the joint space to bring the gripper to the hand configuration $\mathbf{h}$. If the pose is not reachable, the test fails. Otherwise, the gripper closes its fingers until contact with the object or the gripper itself. Then, the robot lifts possibly the object to a given height. If the object is held in the gripper, the grasp is considered as successful. Simulations are performed using Pybullet [22]. We use volumetric hierarchical approximate decomposition to get convex meshes of objects from *obj* files for collision detection [23].

Sensor generative model The sensor generative model $p(i|\mathbf{p}_{\mathcal{O}}, \mathcal{O})$ is implemented in Pybullet, with an approach similar to the Blensor sensor framework [24] used to render depth images from a Kinect model sensor. Simulating a structured-light sensor allows a better transfer to the real setup. Objects and poses are sampled from their priors, $\mathcal{O} \sim p(\mathcal{O}), \mathbf{p}_{\mathcal{O}} \sim p(\mathbf{p}_{\mathcal{O}})$. Then, the object is placed and an image $i \sim p(i|\mathbf{p}_{\mathcal{O}}, \mathcal{O})$ is generated.

Domain randomization for sim-to-real transfer Generative models in simulation differ from their real world counterparts due to incorrect physical modelling and inaccurate physical parameters. This *reality gap* may lead to failures from our model because the synthetic data and the real data distributions are different. To overcome this issue, we use *domain randomization* [25] with nuisance parameters on the position and the orientation of the camera, the minimum and maximum distance of the depth, the field of view, and the coefficient of lateral and spinning frictions μ and γ. Domain randomization avoids the precise calibration of both the grasp and the image simulators, which can be very difficult and costly. We use uniform distributions for the nuisance parameters which are difficult to measure with $\pm 2\%$ error and Gaussian distributions for easily measurable parameters. For the orientation of the camera, a multivariate normal variable $\eta \sim \mathcal{N}(\mathbf{0}, \Sigma), \Sigma = \text{diag}(\sigma_\alpha = 0.002, \sigma_\beta = 0.01, \sigma_\gamma = 0.002)$ is drawn and then mapped to $\text{SO}(3)$ using the exponential map.

Grasping inference strategy		Success rate	
		Sim	*Real*
Prior based	$\mathbf{h} \sim p(\mathbf{h})$	0.8%	-
	$\mathbf{h} = \arg\max p(\mathbf{h})$	0.6%	-
Metric based	$\mathbf{h} = \arg\max_{\mathbf{h}} \hat{p}(S=1\|\mathbf{h})$	43%	-
	$\mathbf{h} = \arg\max_{\mathbf{h}} \hat{p}(\mathbf{h}\|S=1)$	44%	46%
Partial observation based	$\mathbf{h} = \arg\max_{\mathbf{h}} \hat{p}(S=1,i\|\mathbf{h})$	$64\%/71\%$	-
	$\mathbf{h} = \arg\max_{\mathbf{h}} \hat{p}(\mathbf{h}\|S=1,i)$	$71\%/\mathbf{75}\%$	70%
Full observation based (ideal)	$\mathbf{h} = \arg\max_{\mathbf{h}} \hat{p}(S=1,\mathcal{O},\mu,\beta\|\mathbf{h})$	80%	-
	$\mathbf{h} = \arg\max_{\mathbf{h}} \hat{p}(\mathbf{h}\|S=1,\mathcal{O},\mu,\beta)$	85%	-

Table 1: Grasping success rate for various inference strategies of the hand configuration. The success rate obtained by performing Bayesian posterior inference through the full forward simulation reaches 71% for objects seen during training and 75% for 5 new objects. In real experiments, the success rate reaches 70%.

3.2 Simulation benchmarks

We evaluate the performance of our approach incrementally, adding algorithmic components one by one to assess their respective marginal benefits. For each inference strategy, we estimate the success rate over 1000 grasping attempts for randomly sampled objects and camera images. Nuisances parameters are resampled in simulation when evaluating the success. Our general results are summarized in Table 1, while supplementary results for each individual category of objects can be found in Appendix D. We first report results for strategies maximizing the (conditional) densities $p(\mathbf{h}|\cdot)$ of hand configurations. Optimizing for the maximum a priori estimate $\mathbf{h} = \arg\max p(\mathbf{h})$, without conditioning on success or an observation of the scene, leads to a very low success rate of 0.6%. As expected, these results are too poor to be usable but they should underline the informativeness of the prior. Sampling hand configurations from a uniform prior would instead result in a much smaller success rate, by about one order of magnitude (less than 0.1%). When conditioning on the expected success $S = 1$, performance increases to 44%. Taking both the expected success $S = 1$ and the image i into account and following the inference methodology proposed in Section 2, leads to an even larger success rate of 71% for the maximum a posteriori estimates. For the 5 new objects, we reach a comparable success rate of 75%, which demonstrates the good generalization of the approach. In comparison, had the properties $\mathcal{O}$, μ, and β of the object been perfectly known, the success rate would reach 85%, which shows that the convolutional network manages to extract most of the relevant information from the observation i. Table 1 also reports results for maximum likelihood estimates, achieving success rates of 43%, 64% and 80% when maximizing the likelihoods $p(S = 1|\mathbf{h})$, $p(S = 1, i|\mathbf{h})$, and $p(S = 1, \mathcal{O}, \mu, \beta|\mathbf{h})$ respectively. Note that maximizing $p(S = 1|i, \mathbf{h})$ and $p(S = 1|\mathcal{O}, \mu, \beta, \mathbf{h})$ would give the same result since i and $\mathcal{O}, \mu, \beta$ are independent from $\mathbf{h}$. Our informative prior can explain the difference in success rates between the MAP and the MLE estimates and motivates the use of a Bayesian approach.

Not only our framework can be used for grasp planning, it also provides immediate access to the full posterior $p(\mathbf{h}|S = 1, i)$ of hand configurations. As an illustrating example, we extract the marginal posterior densities $p(\mathbf{x}|S = 1, u)$, $p(\mathbf{R}|S = 1, i)$ and $p(g|S = 1, i)$ for the simulated scene of Fig. 1a, with the box centered at $(0, 0)$ without any rotation. The resulting posterior is shown in Fig. 4. First, $p(\mathbf{x}|S = 1, i)$ shows the distribution in space of the hand configuration $\mathbf{h}$. The concentration along the x-axis and z-axis are high, meaning that high density regions are located slightly behind and in front of the box, at a given height related to the geometrical dimensions of the box. Concerning the y-axis, the posterior fails to capture the symmetry and places all the density at the right of the box. Overall, the positions $\mathbf{x}$ which are the most likely to give a successful grasp are on the right corner of the box. This is underlined by the posterior of $p(\mathbf{R}|S = 1, i)$. The red dots correspond to the density of the x-axis, the green dots to the y-axis and the blue dots to the z-axis. The x-axis has one mode, directed toward the table, inherited from the prior and slightly deviates to the right. The y-axis, however, has only two antipodal modes by contrast to the prior. These modes correspond to the situation in which the fingers are placed on the front surface or the back surface. The z-axis can be constructed by taking the cross product between x and y. Uncertainties from x and y are propagated, leading to two antipodal modes with lower concentration than y. Finally, the posterior $p(g|S = 1, i)$ for the grasp type indicates a preference towards pinch and wide modes over the basic mode. Pinch mode is preferred when the position $\mathbf{x}$ is far from the right corner while the wide mode is mainly used when $\mathbf{x}$ is located near the right corner.

3.3 Physical benchmarks

We carried out experiments with a Robotiq 3-finger gripper attached to a UR5 robotic arm, as shown in Figure 1b. A Kinect v1 provides depth images and is rigidly linked to the robot base. The robotic arm is controlled in position in

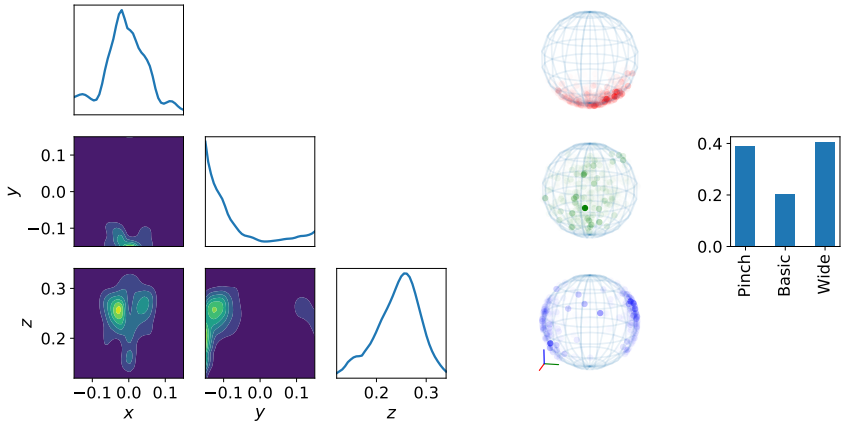


Figure 4: Posterior $p(\mathbf{h}|S=1,i)$ for the setup in Fig. 1a with the box centered at $(0,0)$. (Left) Posterior $p(\mathbf{x}|S=1,i)$, (Middle) Posterior $p(\mathbf{R}|S=1,i)$, (Right) Posterior $p(g|S=1,i)$.

the joint space. Communications are performed within the ROS framework. We calibrate the centre of the table by computing the trajectory in the simulator and then sending it to the real robot. We perform 10 trials per object, for a total of 190 grasps. Objects are placed at best at $(x_{\mathcal{O}}=0, y_{\mathcal{O}}=0, \theta_{\mathcal{O}}=0)$. As shown in Table 1, our success rate of 70% is similar than in simulation, which indicates that the simulation-to-reality transfer works well, at least on average. These results also demonstrate competitive performance with respect to related works (see Section 4), although this remains difficult to establish because of the distinct hardware setups. We observe that failures are mainly behavioral and geometric. Behavioral failures arise when the simulation does not model the physics correctly. For example, in the real setup, the bowl slides on the table when the gripper closes its fingers, while in simulation, the bowl is just lifted. We could reduce these errors by using a more accurate simulator. Geometric failures arise when there is a shift in the location or in the orientation of the object. Most of the time, the robot either collides with the object or misses it for smaller objects. These failures could be avoided using a more precise calibration, additional sensors, or more extensive domain randomization. Finally, the time of computation is reasonable (from 5 to 10s), but could be decreased by tuning the architecture of the neural network, lowering the number of optimization steps, or using more powerful hardware. We leave this for future work.

4 Related work

Over the last decade, progress in multi-fingered robotic grasping has been steady [26, 27, 28, 29, 30, 31, 32] thanks to differentiable models such as neural networks. Unfortunately, the variety of robotic hands, their actuation modes and sensor inputs, and the lack of standard benchmarks, make it difficult to compare these advances fairly against one another.

From early works, [26] identify poses and fingertip positions of stable grasps with a deep neural network from RGB-D images and use a planner (GraspIt!) based on simulated annealing to determine the best hand configurations. They reach a success rate of 75% over a set of 8 objects but suffer from slow execution times (16.6s on average). By contrast, our method is faster and reaches comparable performance over a larger set of objects. [29] use generative adversarial networks to sample both grasp poses and finger joints efficiently based on RGB-D images. While being a fast approach, they reach only 60% of success rate in real experiments. The work of [27] is the most similar to ours. They perform grasp planning as probabilistic inference via a classifier trained to predict the success of a grasp. They retrieve maximum a posteriori estimates using gradient ascent and the fact that the classifier is fully differentiable. The prior distribution is fitted with a Gaussian mixture model from the dataset. In contrast, our method uses an analytical prior based on power-spherical distributions, does not require an external grasp sampler, and relies on a neural classifier to approximate the likelihood-to-evidence ratio. Similarly, [28] compute maximum likelihood estimates of the hand configuration by making use of gradients provided by a neural network. Finally, both of these works treat the rotations with Euler angles and optimize them as real numbers with boundary constraints. This representation is not suitable for a neural network according to [16]. Instead, our optimization relies on Riemannian conjugate gradients, which preserve the geometrical structure of the rotation group. Other interesting approaches to multi-fingered grasping include the use of deep reinforcement learning based on vision and tactile sensors [31], or the use of tactile information only for learning a closed-loop controller [32].

From a statistical perspective, several Bayesian likelihood-free inference algorithms [33, 34, 35, 36, 37, 38, 13] have been developed to carry out inference when the likelihood function is implicit and intractable. These methods operate by approximating the posterior through rejection sampling or by learning parts of the Bayes' rule, such as the likelihood function, the likelihood-to-evidence ratio, or the posterior itself. These algorithms have been used across a wide range of scientific disciplines such as particle physics, neuroscience, biology, or cosmology [11]. To the best of our knowledge, our work is one of the first to apply one of those for the direct planning successful grasps. More specifically, we rely here on the amortized inference approach of [13] to carry out inference within seconds for any new observation i. In contrast, an approach such as ABC [33, 34] could take up to hours to determine a single hand configuration $\mathbf{h}$ since data would need to be simulated on-the-fly for each observation i due to the lack of amortization of ABC. Neural posterior estimation [38] is also amortizable but would have required new methodological developments to be applicable on distributions defined on manifolds, such as those needed here for the rotational part of the pose.

5 Summary and future work

We demonstrate the usefulness and applicability of simulation-based Bayesian inference to multi-fingered grasping. The approach is generic yet powerful because it can work with any simulator, thereby incorporating from the simplest to the more sophisticated piece of domain knowledge, while leveraging all recent developments from deep learning to solve the Bayesian inference problem. Maximum a posteriori hand configurations are found by directly optimizing through the resulting amortized and differentiable expression for the posterior. The geometry of the configuration space is accounted for by proposing a Riemannian manifold optimization procedure through the neural posterior. We demonstrate a working proof-of-concept achieving robust multi-fingered grasping, both in simulation and in real experiments thanks to domain randomization. Our success rate is comparable to previous works.

Acknowledgments

Norman Marlier would like to acknowledge the Belgian Fund for Research training in Industry and Agriculture for its financial support (FRIA grant). Computational resources have been provided by the Consortium des Équipements de Calcul Intensif (CÉCI), funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under Grant No. 2.5020.11 and by the Walloon Region. Gilles Louppe is recipient of the ULiège - NRB Chair on Big data and is thankful for the support of NRB.

References

[1] Domenico Prattichizzo and Jeffrey C. Trinkle. *Grasping*, pages 671–700. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[2] Carlo Ferrari and John F Canny. Planning optimal grasps. In *ICRA*, volume 3, pages 2290–2295, 1992.

[3] Kilian Kleeberger, Richard Bormann, Werner Kraus, and Marco F Huber. A survey on learning-based robotic grasping. *Current Robotics Reports*, pages 1–11, 2020.

[4] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017.

[5] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673, 2018.

[6] Ulrich Viereck, Andreas Pas, Kate Saenko, and Robert Platt. Learning a visuomotor controller for real world robotic grasping using simulated depth images. In *Conference on Robot Learning*, pages 291–300. PMLR, 2017.

[7] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016.

[8] Douglas Morrison, Peter Corke, and Jurgen Leitner. Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. *Robotics: Science and Systems XIV*, pages 1–10, 2018.

[9] Andreas ten Pas, Marcus Gualtieri, Kate Saenko, and Robert Platt. Grasp pose detection in point clouds. *The International Journal of Robotics Research*, 36(13-14):1455–1473, 2017.

[10] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2901–2910, 2019.

[11] Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 2020.

[12] Nicola De Cao and Wilker Aziz. The power spherical distribution. *arXiv preprint arXiv:2006.04437*, 2020.

[13] Joeri Hermans, Volodimir Begy, and Gilles Louppe. Likelihood-free MCMC with amortized approximate ratio estimators. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4239–4248. PMLR, 13–18 Jul 2020.

[14] Johann Brehmer, Siddharth Mishra-Sharma, Joeri Hermans, Gilles Louppe, and Kyle Cranmer. Mining for Dark Matter Substructure: Inferring subhalo population properties from strong lenses with machine learning. *Astrophys. J.*, 886(1):49, 2019.

[15] Kieran Murphy, Carlos Esteves, Varun Jampani, Srikumar Ramalingam, and Ameesh Makadia. Implicit-pdf: Non-parametric representation of probability distributions on the rotation manifold. *arXiv preprint arXiv:2106.05965*, 2021.

[16] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019.

[17] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[18] Jiang Hu, Xin Liu, Zai-Wen Wen, and Ya-Xiang Yuan. A brief introduction to manifold optimization. *Journal of the Operations Research Society of China*, pages 1–50, 2019.

[19] James Townsend, Niklas Koep, and Sebastian Weichwald. Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation. *The Journal of Machine Learning Research*, 17(1):4755–4759, 2016.

[20] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 510–517, 2015.

[21] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[22] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016–2020.

[23] Khaled Mamou and Faouzi Ghorbel. A simple and efficient approach for 3d mesh approximate convex decomposition. In *2009 16th IEEE international conference on image processing (ICIP)*, pages 3501–3504. IEEE, 2009.

[24] Michael Gschwandtner, Roland Kwitt, Andreas Uhl, and Wolfgang Pree. Blensor: Blender sensor simulation toolbox. In *International Symposium on Visual Computing*, pages 199–208. Springer, 2011.

[25] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[26] Jacob Varley, Jonathan Weisz, Jared Weiss, and Peter Allen. Generating multi-fingered robotic grasps via deep learning. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4415–4420. IEEE, 2015.

[27] Qingkai Lu and Tucker Hermans. Modeling grasp type improves learning-based grasp planning. *IEEE Robotics and Automation Letters*, 4(2):784–791, 2019.

[28] Qingkai Lu, Kautilya Chenna, Balakumar Sundaralingam, and Tucker Hermans. Planning multi-fingered grasps as probabilistic inference in a learned deep network. In *Robotics Research*, pages 455–472. Springer, 2020.

[29] Jens Lundell, Enric Corona, Tran Nguyen Le, Francesco Verdoja, Philippe Weinzaepfel, Gregory Rogez, Francesc Moreno-Noguer, and Ville Kyrki. Multi-fingan: Generative coarse-to-fine sampling of multi-finger grasps. *arXiv preprint arXiv:2012.09696*, 2020.

[30] Jens Lundell, Francesco Verdoja, and Ville Kyrki. Ddgc: Generative deep dexterous grasping in clutter. *arXiv preprint arXiv:2103.04783*, 2021.

[31] Bohan Wu, Iretiayo Akinola, Abhi Gupta, Feng Xu, Jacob Varley, David Watkins-Valls, and Peter K Allen. Generative attention learning: a "general" framework for high-performance multi-fingered grasping in clutter. *Autonomous Robots*, pages 1–20, 2020.

[32] Hamza Merzić, Miroslav Bogdanović, Daniel Kappler, Ludovic Righetti, and Jeannette Bohg. Leveraging contact forces for learning to grasp. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3615–3621. IEEE, 2019.

[33] Jean-Michel Marin, Pierre Pudlo, Christian P Robert, and Robin J Ryder. Approximate bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180, 2012.

[34] Mark A Beaumont, Wenyang Zhang, and David J Balding. Approximate bayesian computation in population genetics. *Genetics*, 162(4):2025–2035, 2002.

[35] George Papamakarios, David Sterratt, and Iain Murray. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 837–848. PMLR, 2019.

[36] George Papamakarios and Iain Murray. Fast ϵ-free inference of simulation models with bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*. 2016.

[37] Jan-Matthis Lueckmann, Pedro J. Goncalves, Giacomo Bassetto, Kaan Öcal, Marcel Nonnenmacher, and Jakob H. Macke. Flexible statistical inference for mechanistic models of neural dynamics. In *Advances in Neural Information Processing Systems*, 2017.

[38] David Greenberg, Marcel Nonnenmacher, and Jakob Macke. Automatic posterior transformation for likelihood-free inference. In *International Conference on Machine Learning*, pages 2404–2414. PMLR, 2019.

A Algorithms

Algorithm 1: Training procedure for $d_\phi(S, \mathbf{h})$ and $d_\theta(i, \mathbf{h})$.

Input: Priors $p(\mathbf{h})$, $p(\mathcal{O}), p(\mathbf{p}_\mathcal{O})$
Sensor generative model $p(i|\mathcal{O}, \mathbf{p}_\mathcal{O})$
Grasp generative model $p(S|\mathbf{h}, \mathcal{O}, \mathbf{p}_\mathcal{O})$
Criterion ℓ (e.g, the binary cross-entropy)
Output: Trained classifiers $d_\phi(S, \mathbf{h})$, $d_\theta(i, \mathbf{h})$

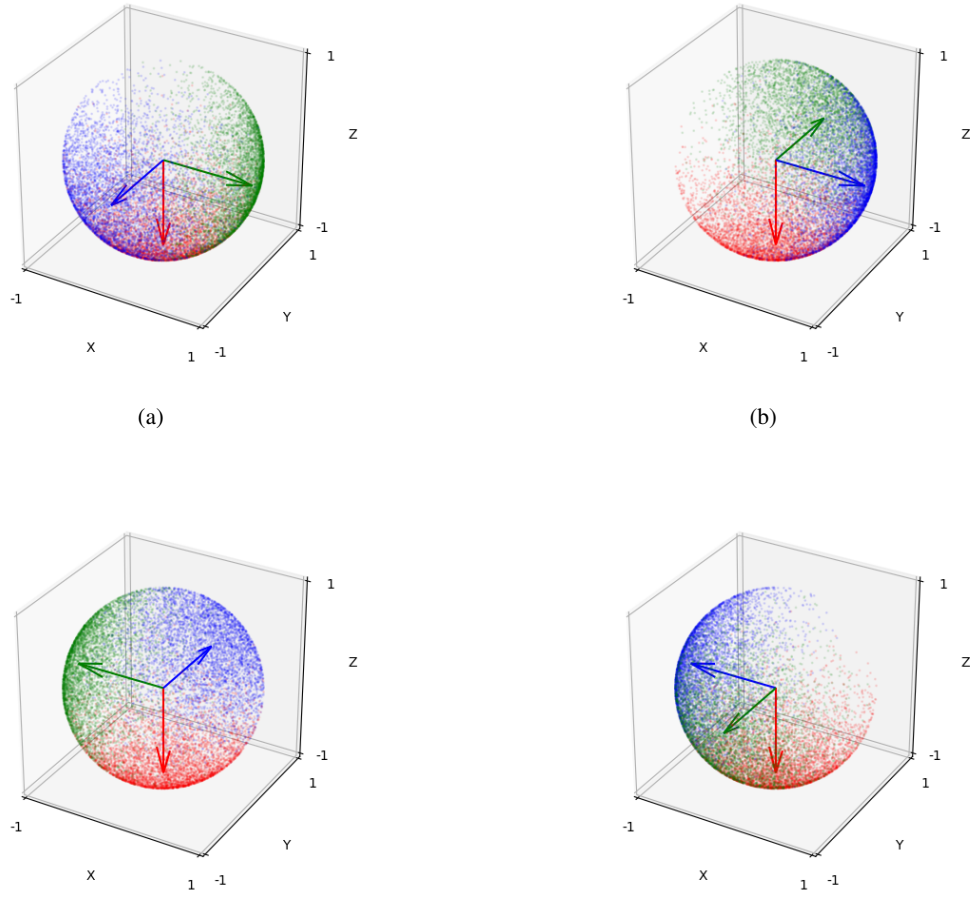
while *not converged* **do**
  Sample $\mathbf{h} \leftarrow \{\mathbf{h}_m \sim p(\mathbf{h})\}_{m=1}^M$
  Sample $\mathbf{h}' \leftarrow \{\mathbf{h}'_m \sim p(\mathbf{h})\}_{m=1}^M$
  Sample $\mathcal{O}, \mathbf{p}_\mathcal{O} \leftarrow \{\mathcal{O}_m, \mathbf{p}_{\mathcal{O},m} \sim p(\mathcal{O})p(\mathbf{p}_\mathcal{O})\}_{m=1}^M$
  Simulate $S \leftarrow \{S_m \sim p(S|\mathbf{h}_m, \mathcal{O}_m, \mathbf{p}_{\mathcal{O},m})\}_{m=1}^M$
  $\mathcal{L} \leftarrow \ell(d_\phi(S, \mathbf{h}), 1) + \ell(d_\phi(S, \mathbf{h}'), 0)$
  $\phi \leftarrow \texttt{OPTIMIZER}(\phi, \nabla_\phi \mathcal{L})$
end
while *not converged* **do**
  Sample $\mathbf{h} \leftarrow \{\mathbf{h}_m \sim p(\mathbf{h}|S = 1)\}_{m=1}^M$
  Sample $\mathbf{h}' \leftarrow \{\mathbf{h}'_m \sim p(\mathbf{h}|S = 1)\}_{m=1}^M$
  Sample $\mathcal{O}, \mathbf{p}_\mathcal{O} \leftarrow \{\mathcal{O}_m, \mathbf{p}_{\mathcal{O},m} \sim p(\mathcal{O}, \mathbf{p}_\mathcal{O}|S = 1, \mathbf{h})\}_{m=1}^M$
  Simulate $i \leftarrow \{i_m \sim p(i|\mathcal{O}_m, \mathbf{p}_{\mathcal{O}m})\}_{m=1}^M$
  $\mathcal{L} \leftarrow \ell(d_\theta(i, \mathbf{h}), 1) + \ell(d_\theta(i, \mathbf{h}'), 0)$
  $\theta \leftarrow \texttt{OPTIMIZER}(\theta, \nabla_\theta \mathcal{L})$
end
return d_ϕ, d_θ

Algorithm 2: Manifold optimization procedure to obtain the MAP estimate $\hat{\mathbf{h}}^*$

Input: Differentiable priors $p(\mathbf{h})$
Differentiable likelihood-to-evidence ratio $\hat{r}$
Depth image i
Output: Approximate MAP estimate $\hat{\mathbf{h}}^*$

MAP cost function $f(i, \mathbf{h}) = -\log \hat{r}(S = 1, i|\mathbf{h}) - \log p(\mathbf{h})$
Sample an initial subset $\mathcal{S} = \{\mathbf{x}, \mathbf{q}\} \sim \{p(\mathbf{x})p(\mathbf{q})\}_1^{1000}$
foreach $g_k \in \mathcal{G}$ **do**
  $\mathbf{h} = (\mathbf{x}, \mathbf{R}(\mathbf{q}), g_k)$
  Initial iterate $\mathbf{h}_0 = \arg\min_{\mathbf{h} \in \mathcal{S}} f(i, \mathbf{h})$
  $\hat{\mathbf{x}}_k^*, \hat{\mathbf{R}}_k^* =$ Geometric CG method$(f, i, \mathbf{h}_0)$
  $\hat{\mathbf{h}}_k^* = (\hat{\mathbf{x}}_k^*, \hat{\mathbf{R}}_k^*, g_k)$
end
return $\arg\min_{\mathbf{h}_k} f(i, \hat{\mathbf{h}}_k^*)$

B Prior Distribution



(a) (b)

(c) (d)

Figure 5: Prior distribution $p(\mathbf{q})$. (a)-(d) correspond to the four modes of the mixture.

C YCB objects

Figure 6: Objects from YCB used for training and testing.

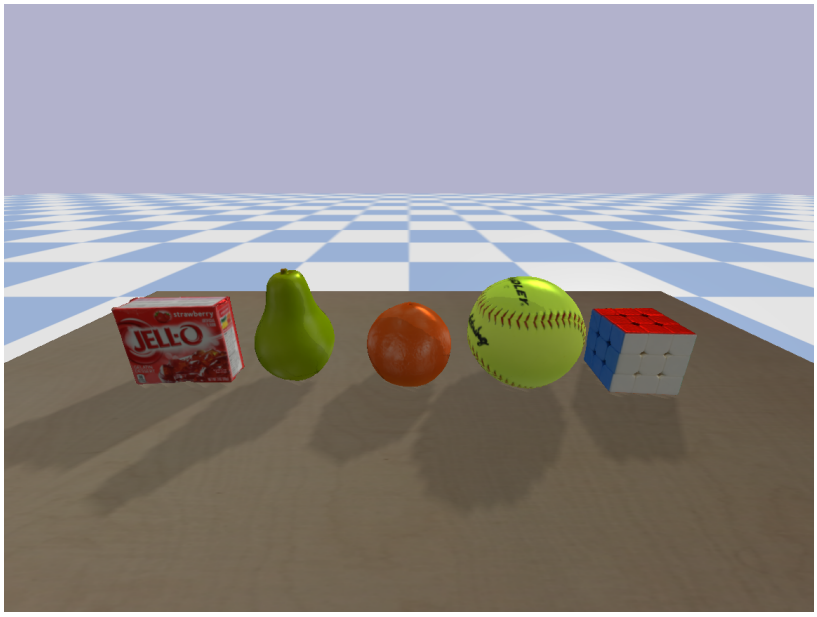


Figure 7: Additional unseen objects from YCB used for testing in simulation.

D Success rate by object category

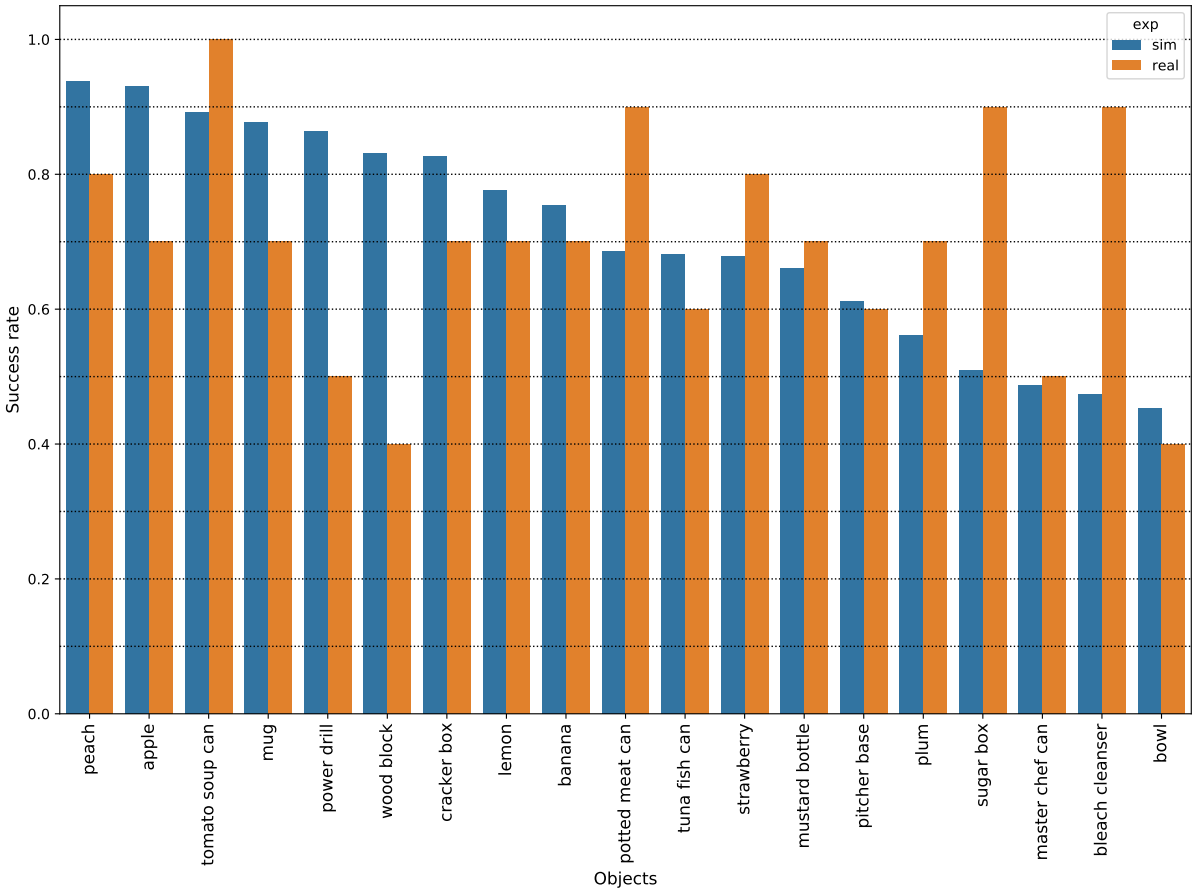


Figure 8: Success rate obtained from simulation and real experiments for each object.

7.3 EPILOGUE

7.3.1 *Advantages*

Our framework offers several advantages. The most notable one is treating the grasp pose as a random variable, thus enabling uncertainty management which is barely the case for other methods. Even if we use point estimates, we have access to the full posterior distribution, understanding which regions are relevant to the task. Our modeling also allows us to perform nearly all grasping tasks, thanks to the success variable. We can change its definition to a specific goal, which will modify the posterior distribution of the grasp pose. For example, we can condition the grasp to be successful only if the object is grasped by the handle. Another advantage of Bayesian methods is that prior knowledge is injected through the prior distribution. This can discard grasp poses that are not relevant to a given task. Furthermore, our framework deals with rotations without any problem of representation or singularities, thanks to the use of geometrical methods. Finally, the sim-to-real gap is overcome without any drop in performances.

7.3.2 *Limitations*

Even if the experiments show promising results, we face some limitations. Bayesian inference makes use of priors, a very important component of the Bayes's rule. Uninformative priors are very ineffective in generating useful samples. In our case, the prior is split into two parts: the position and the orientation. Our prior for the orientation is relevant because it is weakly informative but relies on the fact that objects in this paper are placed at the same location in the same orientation. This limitation will be addressed in the next papers. For the position, it is complicated. In this case, we use a uniform distribution over an important part of the workspace, which is very inefficient. We stressed out that it is a fundamental limitation because this needs to be manually tuned. In the next papers, we will see how to address and overcome this limitation.

7.3.3 *Conclusion and opportunities*

This first paper introduced simulation-based inference for robotic grasping. The results demonstrate the usefulness of the method while the experiments remain in a restricted setup. Opportunities for better priors will show that simulation-based inference methods can perform well on more complex benchmarks.

8

GRASPING A SINGLE OBJECT IN ANY POSE

Outline

We explore more complex prior distributions by conditioning them by the observation. Chapter 7 has shown that uninformative priors are very inefficient in generating useful grasp parameters for simulation-based inference algorithms. The observation gives prior information about the object, such as its position and orientation. We exploit this information to scan interesting areas near the object in the workspace of the robot. Experiments show a high success rate in simulation and a real setup, demonstrating the usefulness of the method.

8.1 PROLOGUE

Priors are fundamental components of simulation-based inference algorithms. In robotics, priors can be inefficient in generating data or require procedures that do not provide density estimation (see Chapter 5). In Chapter 7, we sample the position of the grasp pose from a uniform distribution over the whole workspace, and this strategy turned out to be inefficient, generating less than 1% of successful grasp poses.

In this chapter, we acquire 3D information about the scene through a truncated signed distance function [Curless and Levoy, 1996], instead of using a single depth image. This provides information about the position and 2D orientation of the object to grasp. Therefore, we can define our prior in the local frame of the object and use the observation to condition it, improving a lot the sampling efficiency. Furthermore, we design a specific prior for the orientation, based on a mixture of simple distributions, to increase the expressiveness of our prior.

While not mentioned in the paper, the formulation of our prior was guided by recent work [Dax et al., 2022] on *invariance* and *equivariance* properties of probability distributions with respect to group action. Formally, a G-invariant element $x \in X$ is such that $g \cdot x = x,\ \forall g \in G$. Thus, the probability density function has the following property: $p(g \cdot x) = p(x)$. Equivariance is defined as follows: a function $f : X \mapsto Y$ is said to be equivariant to the group G if $f(g \cdot x) = g \cdot f(x),\ \forall g \in G$. In our case, the prior defined locally exhibits *invariance* properties with respect to a group action. Indeed, because our prior of the position and orientation are defined in the local frame of the object, it is invariant to any transformation $g \in \mathbb{SE}(2)$ applied to the object on the table.

Update In the chapter Chapter 7, we expressed the posterior as

$$p(\boldsymbol{h} \mid S, i) = \frac{p(i \mid S, \boldsymbol{h})}{p(i \mid S)} \frac{p(S \mid \boldsymbol{h})}{p(S)} p(\boldsymbol{h}). \tag{8.1}$$

The prior $p(\boldsymbol{h})$ was proven to be inefficient to sample interesting grasp poses, *i.e* leading to successful grasps. The first ratio $\frac{p(S|\boldsymbol{h})}{p(S)}$ times the prior leads to the posterior distribution $p(\boldsymbol{h} \mid S)$. However, this posterior brings little information about interesting grasps for a particular observation i, because it represents the most likely grasp pose given a success, **marginalized** over all the objects. The ratio $\frac{p(i|S,\boldsymbol{h})}{p(i|S)}$ is the only term that brings information that linked the observation, the success and the grasp pose. But when expressed with this form, the learning is very difficult and may require a great amount of data.

In this new contribution, we express the posterior as

$$p(\boldsymbol{h} \mid S, \boldsymbol{V}) = \frac{p(S \mid \boldsymbol{h}, \boldsymbol{V})}{p(S \mid \boldsymbol{V})} p(\boldsymbol{h} \mid \boldsymbol{V}). \tag{8.2}$$

In this form, the prior $p(\boldsymbol{h} \mid \boldsymbol{V})$ uses the information contained in the observation to sample interesting grasp poses. In addition, the ratio $\frac{p(S|\boldsymbol{h},\boldsymbol{V})}{p(S|\boldsymbol{V})}$ is more meaningful, because it links the probability of success with the observation and the grasp pose. As results, the success rate is highly increased.

Reading tips The reader should at least read the Chapter 7, to understand the whole methodology as well as the contributions. The reader interested in invariant and equivariant properties should look at Dax et al. [2022].

8.2 THE PAPER: SIMULATION-BASED BAYESIAN INFERENCE FOR ROBOTIC GRASPING

Simulation-based Bayesian inference for robotic grasping

Norman Marlier[1] Olivier Brüls[2] Gilles Louppe[3]

***Abstract*— General robotic grippers are challenging to control because of their rich nonsmooth contact dynamics and the many sources of uncertainties due to the environment or sensor noise. In this work, we demonstrate how to compute 6-DoF grasp poses using simulation-based Bayesian inference through the full stochastic forward simulation of the robot in its environment while robustly accounting for many of the uncertainties in the system. A Riemannian manifold optimization procedure preserving the nonlinearity of the rotation space is used to compute the maximum a posteriori grasp pose. Simulation and physical benchmarks show the promising high success rate of the approach.**

I. Introduction

Industrial grasping works very well in highly structured environments with few uncertainties. However, complex applications requiring great flexibility have recently gained a lot of interest. For such tasks, dealing with uncertainties becomes key to robust performance.

While previous methods relied on simplified surrogates of the likelihood function, we bring a novel simulation-based approach for full Bayesian inference based on a deep neural network surrogate of the likelihood-to-evidence ratio. By framing robotic grasping as an inference task, we demonstrate the general applicability of simulation-based inference algorithms to complex robotic tasks and their usefulness to deal with uncertainties.

We summarize our contributions as follow:

- We bring simulation-based Bayesian inference methods [1] to robotic grasping.
- We make use of Riemannian manifold optimization to deal with the nonlinearity of the rotation space.
- We validate our method on simulated and real experiments. Results show promising grasping performances.

II. Problem statement

We consider the problem of planning 6-DoF hand configurations of a general robotic gripper for unknown rigid objects placed on a table and observed through multi-view depth images (Fig. 1).

A. Description

The robot arm (6 or 7 DoF) evolves in a cubic workspace with a planar tabletop. It is equipped with a robotic gripper and observes the scene with a depth camera mounted on its flange. Depth images, captured along a predefined trajectory, are fused into a Truncated Signed Distance Function (TSDF) voxel grid [2]. Then, we search for the most plausible hand configuration given a successful grasp and the TSDF voxel grid. Finally, a joint trajectory is computed by a path planner based on the TSDF to reach the hand pose and grasp the object in order to remove it from the table.

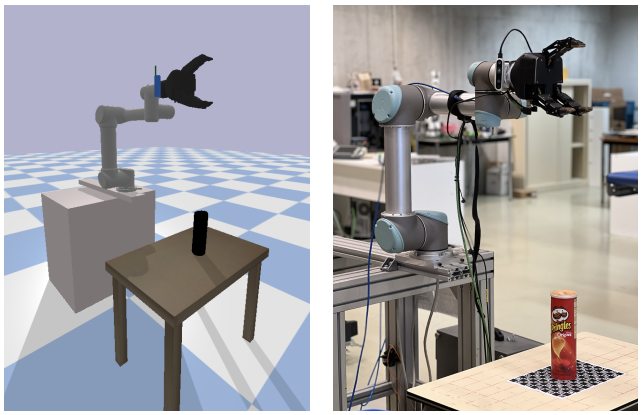


Fig. 1: Our benchmark scene. (left) The simulated environment. (right) The real setup.

B. Notations

Frames We use several reference frames in our work. The world frame $\underrightarrow{\mathcal{F}}_{\mathrm{W}}$ and the workspace frame $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$ can be choosen freely and are not tied to a physical location. $\underrightarrow{\mathcal{F}}_{\mathrm{B}}$, $\underrightarrow{\mathcal{F}}_{\mathrm{C}}$, $\underrightarrow{\mathcal{F}}_{\mathrm{F}}$, $\underrightarrow{\mathcal{F}}_{\mathrm{E}}$ correspond respectively to the robot base, the camera, the flange and the tool center point (TCP).

Hand configuration The hand configuration $\mathbf{h} \in \mathcal{H} = \mathbb{R}^3 \times \mathrm{SO}(3)$ is defined as the combination of the pose $\mathbf{T}_{\mathrm{SE}} = ({}_{\mathrm{S}}\mathbf{t}_{\mathrm{SE}}, \mathbf{R}_{\mathrm{SE}}) \in \mathbb{R}^3 \times \mathrm{SO}(3)$ of the hand, where ${}_{\mathrm{S}}\mathbf{t}_{\mathrm{SE}}$ is the vector $\overrightarrow{SE}$ expressed in $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$. We parametrize the rotation $\mathbf{R}_{\mathrm{SE}}$ with quaternions.

Binary metric A binary variable $S \in \{0, 1\}$ indicates if the grasp fails ($S = 0$) or succeeds ($S = 1$).

Observation Given the depth images $\mathcal{I}_k = \{I_0, ..., I_k\}$ with their corresponding transformations camera to world $\Gamma_k = \{\mathbf{T}^0_{\mathrm{WC}}, ..., \mathbf{T}^k_{\mathrm{WC}}\}$ and camera intrinsic matrix K, we construct a TSDF voxel grid $\mathbf{V}$ with N^3 voxels, representing the workspace of size l.

Latent variables Unobserved variables $\mathbf{z}$ capture uncertainties about the nonsmooth dynamics of contact, the sensor noise, as well as the geometry of the object (see Section.V-A).

C. Probabilistic modeling

We model the scene and the grasping task according to the Bayesian network shown in Fig. 2. The variables $S, \mathbf{V}$ and

*The authors come from the Univeristy of Liège, Belgium
[1]norman.marlier@uliege.be
[2]o.bruls@uliege.be
[3]g.louppe@uliege.be

$\mathbf{h}$ are modelled as random variables to capture the noise in sensors, uncertainties in the dynamics, as well as our prior beliefs about the hand configuration. The structure of the Bayesian network is motivated by the fact that S is dependent on $\mathbf{h}$, $\mathbf{V}$ and $\mathbf{z}$, $\mathbf{h}$ is dependent of $\mathbf{V}$ and $\mathbf{V}$ is dependent on $\mathbf{z}$. This structure also enables a direct and straightforward way to generate data: $\mathbf{h}$ and $\mathbf{z}$ are sampled from their respective prior distributions while S and $\mathbf{V}$ can be generated using forward physical simulators.

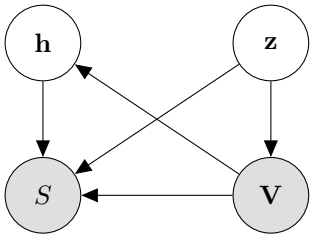


Fig. 2: Probabilistic graphical model of the environment. Gray nodes correspond to observed variables and white nodes to unobserved variables.

D. Objectives

Given our probabilistic graphical model, we formulate the problem of grasping as the Bayesian inference of the hand configuration $\mathbf{h}^*$ that is a posteriori the most likely given a successful grasp and a TSDF voxel grid $\mathbf{V}$. That is, we are seeking for the maximum a posteriori (MAP) estimate

$$\mathbf{h}^* = \arg\max_{\mathbf{h}} \; p(\mathbf{h}|S=1,\mathbf{V}), \tag{1}$$

from which we then compute the joint trajectory

$$\tau_{1:m} = \Lambda(\tau_0, \text{IK}(\mathbf{h}^*), \mathbf{V}) \tag{2}$$

where IK is an inverse kinematic solver, $\tau_{1:m}$ are waypoints in the joint space, $\tau_m = \text{IK}(\mathbf{h}^*)$ and Λ is a path planner.

III. Related work

Probabilistic approaches for grasping problems are usually based on likelihood functions which model the probability of success or a grasp quality metric with respect to an observation and a grasp pose. Then, different methods can be used to find the maximum likelihood estimate (MLE) which corresponds to the final grasp pose. Numerical optimization can be used when the likelihood is modeled by differentiable models [3]. Direct regression of the MLE with a learnt model generates quick output but without capturing the full distribution [4]. Other approaches identify the maximum likelihood estimate based on a list of candidates computed through a grasp map on the sensor space [5], [6]. Similar to our work, [7] learn models respectively for the likelihood and the prior. Then, they can optimize via gradient descent the posterior density. Contrary to our work, they use Euler angles which can lead to gimbal lock and singularities. Our method preserves the topology by using Riemannian gradient descent.

From a statistical perspective, several Bayesian likelihood-free inference algorithms [8], [9], [10], [11], [12], [13], [14] have been developed to carry out inference when the likelihood function is implicit and intractable. These methods operate by approximating the posterior through rejection sampling or by learning parts of the Bayes' rule, such as the likelihood function, the likelihood-to-evidence ratio, or the posterior itself. These algorithms have been used across a wide range of scientific disciplines such as particle physics, neuroscience, biology, or cosmology [1]. To the best of our knowledge, our work is one of the first to apply one of those for the direct planning successful grasps. More specifically, we rely here on amortized neural ratio estimation [14] to carry out inference within seconds for any new observation $\mathbf{V}$. In contrast, an approach such as ABC [8], [9] could take up to hours to determine a single hand configuration $\mathbf{h}$ since data would need to be simulated on-the-fly for each observation $\mathbf{V}$ due to the lack of amortization of ABC. Neural posterior estimation [13] is also amortizable but would have required new methodological developments to be applicable on distributions defined on manifolds, such as those needed here for the rotational part of the pose.

IV. Likelihood-free Bayesian inference for multi-fingered grasping

From the Bayes's rule, the posterior of the hand configuration is

$$p(\mathbf{h}|S,\mathbf{V}) = \frac{p(S \mid \mathbf{h},\mathbf{V})}{p(S \mid \mathbf{V})} p(\mathbf{h} \mid \mathbf{V}). \tag{3}$$

A. Priors

Position The prior over the position ${}_{\text{S}}\mathbf{t}_{\text{SE}} \coloneqq \mathbf{x}_{\text{E}}$ is a uniform distribution over all the dimensions. We first use a uniform distribution over the cube of length $[-1,1]^3$, called $p(\mathbf{u})$ and then use the bijection $\text{B}(\mathbf{u};\mathbf{V}) : [-1,1]^3 \rightarrow [x_{\text{low}}, x_{\text{high}}] \times [y_{\text{low}}, y_{\text{high}}] \times [z_{\text{low}}, z_{\text{high}}]$ to compute $\mathbf{x}_{\text{E}}$, where the bounds are chosen to be the dimensions of the object voxel axis aligned bounding box. Then, $p(\mathbf{x}_{\text{E}} \mid \mathbf{V}) = (\text{B}(\mathbf{V}) \circ p)(\mathbf{u})$. It ensures that the position and orientation are within the same numerical values for estimating the density and the bijection emphasizes our ignorance about interesting regions of space for grasping.

Orientation The prior over the orientation $\mathbf{R}_{\text{SE}} \coloneqq \mathbf{q}_{\text{E}}$ is defined as a mixture of *power-spherical* (PS) distributions [15] with 20 modes ν_i (Fig. 3). Each mode is itself a mixture that satisfies $p(\mathbf{q}_{\text{E}};\cdot) = p(-\mathbf{q}_{\text{E}};\cdot)$. In total, we have

$$p(\mathbf{q}_{\text{E}}) = \frac{1}{20}\sum_{i=1}^{20} \frac{\text{PS}(\mathbf{q}_{\text{E}};\nu_i,\kappa)}{2} + \frac{\text{PS}(\mathbf{q}_{\text{E}};-\nu_i,\kappa)}{2}. \tag{4}$$

This prior encodes a top-down approach as well as side approaches by its 5 main modes ν_i. The 4 additional modes, rotated by $\frac{\pi}{2}$, allows us to explore various orientations. We set the concentration factor $\kappa = 8$ for all modes, which keeps the prior gradients low and not hightly regularizes the MAP. In this way, our prior covers a large part of the rotation space

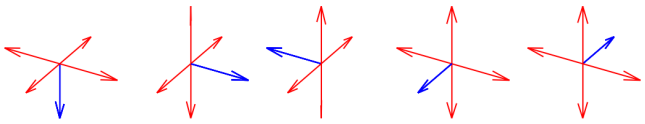


Fig. 3: The modes of the orientation distribution. (left) Encode a top-down approach. (others) Encode side approach.

and is sufficiently informative by contrast to a uniform prior over the unit sphere $\mathbb{S}^3$.

Finally, $p(\mathbf{h} \mid \mathbf{V}) = p(\mathbf{x}_{\mathrm{E}} \mid \mathbf{V})p(\mathbf{q}_{\mathrm{E}})$.

B. Density ratio estimation

The likelihood function $p(S \mid \mathbf{h}, \mathbf{V})$ and the evidence $p(S \mid \mathbf{V})$ are both intractable, which makes standard Bayesian inference procedures such as Markov chain Monte Carlo unusable. However, drawing samples from forward models remains feasible with physical simulators, hence enabling likelihood-free Bayesian inference algorithms.

First, we express the likelihood-to-evidence ratio as,

$$r(S \mid \mathbf{h}, \mathbf{V}) = \frac{p(S \mid \mathbf{h}, \mathbf{V})}{p(S \mid \mathbf{V})} = \frac{p(S, \mathbf{h} \mid \mathbf{V})}{p(S \mid \mathbf{V})p(\mathbf{h} \mid \mathbf{V})}. \quad (5)$$

By adapting the approach described in [14] for likelihood ratio estimation, we train a neural network classifier d_ϕ that we will use to approximate $r(S|\mathbf{h}, \mathbf{V})$. The network d_ϕ is trained to distinguish positive tuples $(S, \mathbf{h}, \mathbf{V})$ (labeled $y = 1$) sampled from the joint distribution $p(S, \mathbf{h} \mid \mathbf{V})$ against negative tuples (labeled $y = 0$) sampled from the product of marginals $p(S \mid \mathbf{V})p(\mathbf{h} \mid \mathbf{V})$. The Bayes optimal classifier $d^*(S, \mathbf{h}, \mathbf{V})$ that minimizes the cross-entropy loss is given by

$$d^*(S, \mathbf{h}, \mathbf{V}) = \frac{p(S, \mathbf{h} \mid \mathbf{V})}{p(S \mid \mathbf{V})p(\mathbf{h} \mid \mathbf{V}) + p(S, \mathbf{h} \mid \mathbf{V})}, \quad (6)$$

which recovers the likelihood ratio $r(S|\mathbf{h})$ as

$$\frac{d^*(S, \mathbf{h}, \mathbf{V})}{1 - d^*(S, \mathbf{h}, \mathbf{V})} = \frac{p(S, \mathbf{h} \mid \mathbf{V})}{p(S \mid \mathbf{V})p(\mathbf{h} \mid \mathbf{V})} = \frac{p(S|\mathbf{h}, \mathbf{V})}{p(S \mid \mathbf{V})}. \quad (7)$$

Therefore, by modelling the classifier with a neural network d_ϕ trained on the binary classification problem, we obtain an approximate but amortized and differentiable likelihood ratio

$$\hat{r}(S \mid \mathbf{h}, \mathbf{V}) = \frac{d_\phi(S, \mathbf{h}, \mathbf{V})}{1 - d_\phi(S, \mathbf{h}, \mathbf{V})}. \quad (8)$$

Finally, the likelihood ratio is combined with the prior to approximate the posterior as

$$\hat{p}(\mathbf{h}|S = 1, \mathbf{V}) = \hat{r}(S = 1 \mid \mathbf{h}, \mathbf{V})p(\mathbf{h} \mid \mathbf{V}), \quad (9)$$

which enables immediate posterior inference despite the initial intractability of the likelihood function $p(S \mid \mathbf{h}, \mathbf{V})$ and of the evidence $p(S \mid \mathbf{V})$.

Ensembles tend to produce more conservative posteriors [16]. In our case, we take 4 models and compute the ratio as

$$\log \hat{r} = \log \frac{1}{4} \Sigma_{i=1}^{4} \exp \log \hat{r}_i \quad (10)$$

The neural network classifiers d_ϕ is architectured as follows. The hand configuration $\mathbf{h}$ enters the neural network as a tuple of $(\mathrm{N_B} \times 3, \mathrm{N_B} \times 4)$ vector where $\mathrm{N_B}$ is the batch size. The position is rescaled into a cube of $[-1, 1]$ thanks to a bijection. In d_ϕ, $\mathbf{V}$ is fed to a 3D convolutional network made of four convolutional layers followed by a fully connected layer, as in [5], and which goal is to produce a vector embedding of the voxel grid. The voxel embedding, the 4D pose (position and 2D rotation) of the object point cloud $\mathbf{p} = f(\mathbf{V})$ obtained via the TSDF, S and $\mathbf{h}$ are then fed to a subsequent network made of 2 fully connected layers of 256 neurons. The parameters ϕ are optimized using Adam as optimizer.

C. Maximum a posteriori estimation

Due to the intractability of the likelihood function and of the evidence, Eq. (1) cannot be solved analytically nor numerically. We rely instead on the approximation given by the likelihood-to-evidence ratio $\hat{r}$ to find an approximation of the maximum a posteriori (MAP) estimate as

$$\hat{\mathbf{h}}^* = \arg\max_{\mathbf{h}} \hat{r}(S = 1 \mid \mathbf{h}, \mathbf{V})p(\mathbf{h} \mid \mathbf{V}) \quad (11)$$

$$= \arg\min_{\mathbf{h}} -\log \hat{r}(S = 1 \mid \mathbf{h}, \mathbf{V})p(\mathbf{h} \mid \mathbf{V}), \quad (12)$$

which we solve using gradient descent. The gradient of Eq. (12) decomposes as

$$-\nabla_{(\mathbf{x},\mathbf{q})} \log \hat{r}(S \mid \mathbf{h}, \mathbf{V})p(\mathbf{h} \mid \mathbf{V}) = -\nabla_{(\mathbf{x},\mathbf{q})} \log \hat{r}(S \mid \mathbf{h}, \mathbf{V}) - \nabla_{(\mathbf{x},\mathbf{q})} \log p(\mathbf{h} \mid \mathbf{V}). \quad (13)$$

Our prior $p(\mathbf{h} \mid \mathbf{V})$ has analytical gradients. In fact, uniform distributions are set to have null gradient everywhere in the domain. Therefore, $\nabla_{\mathbf{x}} p(\mathbf{h}) = \mathbf{0}$. By contrast, $p(\mathbf{q}_{\mathrm{E}})$ is a weakly informative prior and has a non null gradient from the power spherical distribution. Its derivative with respect to $\mathbf{q}$ is

$$\begin{aligned} \nabla_{\mathbf{q}} p(\mathbf{q}; \nu, \kappa) &= C(\kappa)\kappa(1 + \nu^T\mathbf{q})^{\kappa-1}\nabla_{\mathbf{q}}(1 + \nu^T\mathbf{q}) \\ &= C(\kappa)\kappa\nu(1 + \nu^T\mathbf{q})^{\kappa-1}, \end{aligned} \quad (14)$$

where $C(\kappa)$ is the normalization term. Since the likelihood-to-evidence ratio estimator $\hat{r}$ is modelled by a neural network, it is fully differentiable with respect to its inputs and its gradients can be computed by automatic differentiation. However, not all variables of the problem are Euclidean variables and naively performing gradient descent would violate our geometric assumptions. Let us consider a variable $\mathcal{Z}$ on the smooth Riemannian manifold $\mathcal{M} = \mathbb{R}^3 \times \mathbb{S}^3$ with tangent space $\mathcal{T}_{\mathcal{Z}}\mathcal{M}$ and a function $f : \mathcal{M} \to \mathbb{R}$. Since $\mathbb{S}^3$ is embedded in $\mathbb{R}^4$, f can be evaluated on $\mathbb{R}^3 \times \mathbb{R}^4$, leading to the definition of the Euclidean gradients $\nabla f(\mathcal{Z}) \in \mathbb{R}^3 \times \mathbb{R}^4$. In turn, these Euclidean gradients can be transformed into their Riemannian counterparts $\mathrm{grad} f(\mathcal{Z})$ via orthogonal projection $\mathbf{P}_{\mathcal{Z}}$ into the tangent space $\mathcal{T}_{\mathcal{Z}}\mathcal{M}$. Therefore,

$$\mathrm{grad} f(\mathcal{Z}) = \mathbf{P}_{\mathcal{Z}}(\nabla f(\mathcal{Z})) \quad (15)$$

where the orthogonal projection onto $\mathbb{R}^3$ is the identity $\mathbb{I}_3$ and the orthogonal projection onto $\mathbb{S}^3$ is $\mathbf{P}_{\xi}(\nabla f) = (\mathbb{I}_4 -$

$\xi\xi^T)\nabla f$ at $\xi \in \mathbb{S}^3$. Thus, we can solve Eq. (12) by projecting Euclidean gradients of Eq. (13) to the tangent space $\mathcal{T}_{\mathcal{Z}}\mathcal{M}$ and use it in the following update rule

$$\mathbf{h}_{k+1} = \exp_{\mathbf{h}_k}(-\alpha_k \text{grad} f(\mathbf{h}_k)) \tag{16}$$

with $\exp_x(v) : \mathcal{T}_x\mathcal{M} \to \mathcal{M}$ is the exponential map.

V. Experiments

To validate our approach, we perform a series of experiments in simulation as well as in the real setup. We evaluate the performance of our method and determine the transfer capabilities of our network without any fine-tuning.

A. Data generation

The data generating procedure is defined as follow:

$$\mathbf{z} \sim p(\mathbf{z}) \tag{17}$$
$$I_k \sim p(I \mid \mathbf{z}, \mathbf{T}^k_{\text{WC}}) \tag{18}$$
$$\mathbf{V} = f(\mathcal{I}_k, \Gamma_k) \tag{19}$$
$$\{\mathbf{h} \sim p(\mathbf{h} \mid \mathbf{V})\} \tag{20}$$
$$\{\tau_{1:m} \sim \Lambda(\tau_0, \text{IK}(\mathbf{h}), \mathbf{V})\} \tag{21}$$
$$\{S \sim p(S \mid \tau_{1:m}, \mathbf{z})\} \tag{22}$$

We use Pybullet [17] for implementing these functions. We use the same object assets than VGN [5] for the training and testing. The latent variables $\mathbf{z}$ are described as follow:

Object mesh We sample uniformly an object mesh from an asset of objects.

Pose of the table $\mathbf{T}_{\text{ST}}$ We randomize the position $(x, y) \sim \mathcal{N}(0, 0.008)$ and the rotation $q_{\text{T}} = (0., 0., \sin(\frac{\theta_{\text{Table}}}{2}), \cos(\frac{\theta_{\text{Table}}}{2}))$, $\theta_{\text{Table}} \sim \mathcal{U}(-5, 5)$ of the table with respect to $\underrightarrow{\mathcal{F}}_{\text{S}}$.

Pose of the object $\mathbf{T}_{\text{TO}}$ We randomize the position $(x, y) \sim \mathcal{U}(\frac{-l}{2}, \frac{l}{2})$ and the orientation $q_{\text{O}} = (0., 0., \sin(\frac{\theta_{\text{O}}}{2}), \cos(\frac{\theta_{\text{O}}}{2}))$, $\theta_{\text{O}} \sim \mathcal{U}(0, 2\pi)$ of the object with respect to $\underrightarrow{\mathcal{F}}_{\text{T}}$.

Torque applied by the fingers We randomize the final torque applied by the fingers $\tau \sim \mathcal{U}(35, 40)$.

Lateral friction coefficient We randomize the lateral friction coefficient $\mu \sim \mathcal{U}(1, 2)$.

Spinning friction coefficient We randomize the spinning friction coefficient $\gamma = \eta\mu, \eta \sim \mathcal{N}(0.002, 0.0001)$.

Depth images We add noise to the rendered depth images in simulation using the additive noise model of [18] with the same parameters.

B. Simulated Experiments

We evaluate the performance of our method with the success rate (%). For one round, procedures from (17) to (19) are done. We find the MAP or the MLE by sampling 1000 initial hand configurations from the prior and we take the best one. Then, we perform 300 optimization steps with a step size of 0.005 for the orientation and 0.008 for the position. Because of the stochastic nature of our MAP estimate, we recompute the MLE/MAP at a maximum of 3 times if the path planner fails to find a valid path. Our method reaches a success rate of nearly $\mathbf{91}\%$ with the MAP, demonstrating the capabilities to adapt to new objects and correctly lift object. Moreover, the MLE performs slightly lower (87.3%) than the MAP. Our weakly informative prior explains the difference in success rates and motivates the use of a Bayesian approach.

Fig. 4: (left) Object assets used in the real setup. (right) Example of side grasp.

C. Real Robot Experiments

We carry out experiments with a Robotiq 3-finger gripper attached to a UR5 robotic arm, as shown in Fig. 1. A Intel Realsense D435i depth sensor is mounted to the flange of the robotic arm. It produces 848×480 depth images which are integrated into a TSDF with a resolution of $N = 40$ for the network and a resolution of $N = 120$ for collision detection using Open3D [19]. The transformation $\mathbf{T}_{\text{FC}}$ is calibrated using hand-eye calibration from OpenCV [20]. All the devices are handled within the ROS framework. We performs 100 rounds with a protocol similar to the simulation experiments. We randomly select 1 object from the 10 test objects and put it randomly on the table by hand. The objects are chosen between seen and unseen objects during training and for their availability in the lab. Our success rate of $\mathbf{90}\%$ is similar than in simulation, which indicates that the simulation-to-reality transfer works well. Our approximate ratio learnt successfully several modes to grasp an object and can switch most of the time between them if the path planner fails (Fig .4).

In simulation as well as in the real setup, half of the failure cases are due to the path planner and half are due to wrong hand configurations making the object slip. We leave the improvement of these parts as future work.

VI. Conclusion

We demonstrate the usefulness and applicability of simulation-based Bayesian inference to robotic grasping. Our results show promising performance for determining 6 DoF grasp poses. Nevertheless, our task is rather simple compared to others benchmarks. In the next step, we plan to challenge our method to more complex tasks such as grasping in cluttered environments.

ACKNOWLEDGEMENT

Norman Marlier would like to acknowledge the Belgian Fund for Research training in Industry and Agriculture for its financial support (FRIA grant). Computational resources have been provided by the Consortium des Équipements de Calcul Intensif (CÉCI), funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under Grant No. 2.5020.11 and by the Walloon Region. Gilles Louppe is recipient of the ULiège - NRB Chair on Big Data and is thankful for the support of the NRB

REFERENCES

[1] K. Cranmer, J. Brehmer, and G. Louppe, "The frontier of simulation-based inference," *Proceedings of the National Academy of Sciences*, 2020.

[2] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996, pp. 303–312.

[3] Q. Lu, K. Chenna, B. Sundaralingam, and T. Hermans, "Planning multi-fingered grasps as probabilistic inference in a learned deep network," in *Robotics Research*. Springer, 2020, pp. 455–472.

[4] J. Cai, J. Cen, H. Wang, and M. Y. Wang, "Real-time collision-free grasp pose detection with geometry-aware refinement using high-resolution volume," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1888–1895, 2022.

[5] M. Breyer, J. J. Chung, L. Ott, S. Roland, and N. Juan, "Volumetric grasping network: Real-time 6 dof grasp detection in clutter," in *Conference on Robot Learning*, 2020.

[6] D. Morrison, P. Corke, and J. Leitner, "Learning robust, real-time, reactive robotic grasping," *The International Journal of Robotics Research*, vol. 39, p. 027836491985906, 06 2019.

[7] M. Van der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans, "Learning continuous 3d reconstructions for geometrically aware grasping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 11 516–11 522.

[8] J.-M. Marin, P. Pudlo, C. P. Robert, and R. J. Ryder, "Approximate bayesian computational methods," *Statistics and Computing*, vol. 22, no. 6, pp. 1167–1180, 2012.

[9] M. A. Beaumont, W. Zhang, and D. J. Balding, "Approximate bayesian computation in population genetics," *Genetics*, vol. 162, no. 4, pp. 2025–2035, 2002.

[10] G. Papamakarios, D. Sterratt, and I. Murray, "Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 837–848.

[11] G. Papamakarios and I. Murray, "Fast ϵ-free inference of simulation models with bayesian conditional density estimation," in *Advances in Neural Information Processing Systems*, 2016.

[12] J.-M. Lueckmann, P. J. Goncalves, G. Bassetto, K. Öcal, M. Nonnenmacher, and J. H. Macke, "Flexible statistical inference for mechanistic models of neural dynamics," in *Advances in Neural Information Processing Systems*, 2017.

[13] D. Greenberg, M. Nonnenmacher, and J. Macke, "Automatic posterior transformation for likelihood-free inference," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2404–2414.

[14] J. Hermans, V. Begy, and G. Louppe, "Likelihood-free MCMC with amortized approximate ratio estimators," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020, pp. 4239–4248. [Online]. Available: http://proceedings.mlr.press/v119/hermans20a.html

[15] N. De Cao and W. Aziz, "The power spherical distribution," *arXiv preprint arXiv:2006.04437*, 2020.

[16] J. Hermans, A. Delaunoy, F. Rozet, A. Wehenkel, and G. Louppe, "Averting a crisis in simulation-based inference," *arXiv preprint arXiv:2110.06581*, 2021.

[17] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2020.

[18] Z. Jiang, Y. Zhu, M. Svetlik, K. Fang, Y. Zhu, J. Tremblay, S. Birchfield, G. Shi, F. Ramos, A. Anandkumar, *et al.*, "Synergies between affordance and geometry: 6-dof grasp detection via implicit representations," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[19] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[20] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

8.3 EPILOGUE

8.3.1 *Advantages*

Our framework offers great flexibility through the prior. In this paper, we demonstrate how a simple prior with invariant property can greatly increase the usefulness of our method for some robotics applications. These properties help a lot in learning the posterior because latent variables contain the position of the object on the table as well as its orientation. Furthermore, by using invariant priors, we can learn an equivariant posterior and thus reduce the amount of data needed to reach a high success rate. This is a desirable property for limiting the time required to generate data. Contrary to the benchmark used in Chapter 7, the object can be placed anywhere, making the task more difficult, and yet our method overcomes the previous performance.

8.3.2 *Limitations*

While our prior is very effective for a single object on the table, this is complicated to extend it to many objects. When several objects are placed on the table, our prior cannot make the difference and consider all the objects as one. This implies that it will put density where there is no object, thus dropping a lot the success rate of the method. A clustering algorithm may be used to split objects into clusters of point cloud but this is not accurate and may lead to errors. Furthermore, this prior is not flexible enough to grasp objects in any 3D orientation. It works only for objects that are placed in an upright configuration. This main limitation restraints the complexity of the robotic task, which will be addressed in the next chapter.

8.3.3 *Conclusion and opportunities*

Our new contribution increases the success rate of the task by about 15%, which justifies the use of a better prior. We further increase the complexity of the task, by grasping an object in any 2D pose on the table while keeping the 6 DoF of the grasp pose. We will see in the next chapter how to automate the building of prior useful for grasping several objects.

9

GRASPING MANY OBJECTS IN A RESTRICTED SETUP

Outline

We automate the building of the prior by learning an implicit representation of the scene. While Chapter 8 used heuristic priors conditioned by observation, we leverage the development of 3D reconstruction methods to model accurately the scene. Furthermore, we use a point cloud representation instead of a voxel one, decreasing the sensor data acquisition time. We slightly modify the modeling of the grasping task to integrate this learned prior. We validate the results on more complex benchmarks while we keep the dimension of the search space lower, by computing a top-down approach and not the full 6 DoF.

9.1 PROLOGUE

Machine learning brings a way to learn automatically how to perform robotic tasks. However, as we saw in the two previous chapters, it is quite complicated to build an efficient prior for robotics tasks. While keeping the prior simple for practitioners, we can use neural networks to learn efficient priors by adding a new random variable to our robotic modeling. By using the observation of the scene, we can improve a lot the sample efficiency of our method, thus performing grasping in a cluttered environment, with many objects.

Sampling from density estimators can be difficult, and even more so when the variables belong to Riemannian manifolds. To overcome this issue, we adapt the famous Hamiltonian Monte Carlo sampling scheme to work with Riemannian manifolds. This allows us to generate samples from the approximate posterior and thus visualize which position and orientation will lead to a successful grasp.

Update In the chapter Chapter 8, we expressed the posterior as

$$p(\boldsymbol{h} \mid S, \boldsymbol{V}) = \frac{p(S \mid \boldsymbol{h}, \boldsymbol{V})}{p(S \mid \boldsymbol{V})} p(\boldsymbol{h} \mid \boldsymbol{V}). \tag{9.1}$$

However, the prior $p(\boldsymbol{h} \mid \boldsymbol{V})$ is based on a heuristic, meaning it is difficult to extend to new tasks. More importantly, it cannot handle several objects being in the workspace

by assumption. Therefore, we introduce a new variable, the occupancy o, to be able to grasp several objects. To do so, we express the posterior as

$$p(\boldsymbol{h} \mid S, o, \boldsymbol{V}) = \frac{p(S \mid \boldsymbol{h}, o, \boldsymbol{V})}{p(S \mid o, \boldsymbol{V})} p(\boldsymbol{h} \mid o, \boldsymbol{V}). \tag{9.2}$$

The occupancy removed the assumption of only one object standing on the table. Instead of reasoning at the object level, we reason at position level, giving more flexibility because several objects can be on the table.

Reading tips As this paper is more oriented toward probability density on manifolds and sampling, the reader should read Chapter 3 and Chapter 4. The rest of the paper should flow smoothly.

9.2 THE PAPER: IMPLICIT REPRESENTATION PRIORS MEET RIEMANNIAN GEOMETRY FOR BAYESIAN ROBOTIC GRASPING

Implicit representation priors meet Riemannian geometry for Bayesian robotic grasping

Norman Marlier[1*] Julien Gustin[2] Olivier Brüls[3] Gilles Louppe[4]

***Abstract*— Robotic grasping in highly noisy environments presents complex challenges, especially with limited prior knowledge about the scene. In particular, identifying good grasping poses with Bayesian inference becomes difficult due to two reasons: i) generating data from uninformative priors proves to be inefficient, and ii) the posterior often entails a complex distribution defined on a Riemannian manifold. In this study, we explore the use of implicit representations to construct scene-dependent priors, thereby enabling the application of efficient simulation-based Bayesian inference algorithms for determining successful grasp poses in unstructured environments. Results from both simulation and physical benchmarks showcase the high success rate and promising potential of this approach.**

I. Introduction

Grasping is a fundamental skill for any robotic system. While current methods are effective for highly constrained tasks in structured environments, new and complex applications require increased flexibility and more advanced algorithms to account for the uncertainties that emerge in unstructured and noisy environments. Bayesian inference offers a well-principled approach to address these uncertainties; however, robotic tasks present unique challenges that make Bayesian inference difficult to apply, particularly for sampling-based algorithms. Firstly, many Bayesian approaches assume that the likelihood is tractable and can be evaluated, which is seldom the case in robotics. Secondly, parameters may span a vast space, leading to inefficient sampling strategies. Lastly, parameters of interest often belong to smooth Riemannian manifolds, further complicating the inference procedure. In this paper, we address these challenges by designing informative scene-dependent priors and using simulation-based inference algorithms combined with geometric sampling methods. Our contributions are summarized as follows:

- We integrate simulation-based Bayesian inference methods [1] with 3D implicit representations for robotic grasping.
- We adapt geodesic Monte Carlo [2] with a neural ratio estimator to sample on Riemannian manifolds with an intractable likelihood.
- We validate our method through simulated and real experiments, demonstrating promising grasping performance.

*The authors come from the University of Liège, Belgium
[1]norman.marlier@uliege.be

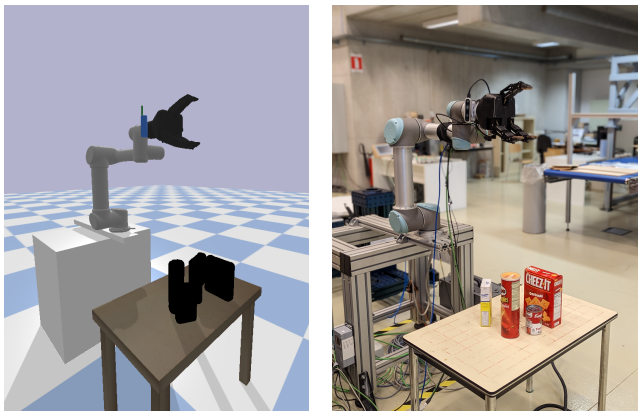


Fig. 1: Our benchmark scene. (left) The simulated environment. (right) The real setup.

II. Problem statement

We consider the problem of planning 4-DoF hand configurations for a robotic gripper handling various unknown objects on a table, observed with a depth camera. A benchmark scene is shown in Fig. 1.

A. Notations

Frames We use several reference frames in our work. The world frame $\underrightarrow{\mathcal{F}}_{\mathrm{W}}$ and the workspace frame $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$ can be chosen freely and are not tied to a physical location. The world frame is used for the robot and the sensor, while the workspace frame is used for our inference system. $\underrightarrow{\mathcal{F}}_{\mathrm{C}}$ and $\underrightarrow{\mathcal{F}}_{\mathrm{E}}$ correspond respectively to the camera and the tool centre point.

Hand configuration The hand configuration $\mathbf{h} \in \mathcal{H} = \mathbb{R}^3 \times \mathbb{S}^1$ is defined as the pose $(\mathbf{x}, \mathbf{q}) \in \mathbb{R}^3 \times \mathbb{S}^1$ of the hand, where $\mathbf{x}$ is the vector $\vec{SE}$ expressed in $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$ and $\mathbf{q}$ is the planar rotation represented with complex numbers defined in $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$.

Binary metric A binary variable $S \in \{0, 1\}$ indicates if the grasp fails $(S = 0)$ or succeeds $(S = 1)$.

Observation Given the depth image I with its corresponding transformation camera to world $\mathbf{T}_{\mathrm{WC}}$ and camera intrinsic matrix K, we construct a point cloud $\mathbf{P} \in \mathbb{R}^{2048\times 3}$ expressed in $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$.

Occupancy A binary variable $o \in \{0, 1\}$ indicates if a point $\mathbf{p} \in \mathbb{R}^3$ is occupied by any object of the scene.

Latent variables Unobserved variables $\mathbf{z}$ capture uncertainties about the nonsmooth dynamics of contact, the sensor noise, as well as the number of objects and their geometry.

B. Grasping as inference

We formulate the problem of grasping as the Bayesian inference of the hand configuration $\mathbf{h}^*$ that is a posteriori the most likely given a successful grasp, an occupancy o and a point cloud $\mathbf{P}$. That is, we are seeking the maximum a posteriori (MAP) estimate

$$\mathbf{h}^* = \arg\max_{\mathbf{h}} \; p(\mathbf{h}|S=1, o=1, \mathbf{P}), \tag{1}$$

from which we then compute the joint trajectory

$$\tau_{1:m} = \Lambda(\tau_0, \text{IK}(\mathbf{h}^*), \mathbf{P}), \tag{2}$$

where IK is an inverse kinematic solver, $\tau_{1:m}$ are waypoints in the joint space, $\tau_m = \text{IK}(\mathbf{h}^*)$ with $\mathbf{h}^*$ expressed in $\underset{\rightarrow}{\mathcal{F}}_{\text{W}}$ and Λ is a path planner.

III. Implicit representation of priors and posteriors for robotic grasping

From the Bayes rule, the posterior of the hand configuration is

$$p(\mathbf{h}|S, o, \mathbf{P}) = \frac{p(S \mid \mathbf{h}, o, \mathbf{P})}{p(S \mid o, \mathbf{P})} p(\mathbf{h} \mid o, \mathbf{P}), \tag{3}$$

which can be rewritten as the product of the likelihood-to-evidence ratio r and a scene-dependent prior

$$p(\mathbf{h}|S, o, \mathbf{P}) = r(S \mid \mathbf{h}, o, \mathbf{P}) p(\mathbf{h} \mid o, \mathbf{P}). \tag{4}$$

A. Priors

Position The scene-dependent prior over the position $\mathbf{x}$ is the distribution $p(\mathbf{x}|o, \mathbf{P}) = \frac{p(o|\mathbf{x}, \mathbf{P})}{p(o|\mathbf{P})} p(\mathbf{x})$, where $p(o|\mathbf{x}, \mathbf{P})$ is the likelihood of the occupancy o, $p(\mathbf{x})$ is uniform over the workspace, and $p(\mathbf{x}|\mathbf{P})$ is simplified to $p(\mathbf{x})$ by independence.

We model the occupancy likelihood $p(o|\mathbf{x}, \mathbf{P})$ using a Convolutional Occupancy Network [3]. This network computes the occupancy by first producing three canonical features planes $\mathbf{c}_{xy}(\mathbf{P}), \mathbf{c}_{xz}(\mathbf{P})$ and $\mathbf{c}_{yz}(\mathbf{P})$. Then, the bilinear interpolations of the three planes are used to compute $\psi(\mathbf{P}, \mathbf{x}) = \mathbf{c}_{xy}(\mathbf{P})(\mathbf{x}) + \mathbf{c}_{xz}(\mathbf{P})(\mathbf{x}) + \mathbf{c}_{yz}(\mathbf{P})(\mathbf{x})$. These point-wise features at point $\mathbf{x}$ are finally processed by a fully connected network, outputting the occupancy probability.

This implicit representation allows us to sample interesting grasping positions from $p(\mathbf{x}|o, \mathbf{P}) \propto p(o|\mathbf{x}, \mathbf{P}) p(\mathbf{x})$. We use Hamiltonian Monte Carlo (HMC) to take advantage of the differentiability of the occupancy network.

Orientation The prior of the orientation $\mathbf{q}$ is a uniform distribution over the unit circle $\mathbb{S}^1$. This prior is *invariant* to any rotation $\mathbf{R} \in \text{SO}(2)$ applied to $\mathbf{q}$, satisfying $p(\mathbf{q}) = p(\mathbf{R}\mathbf{q})$. This property enables free selection of the reference frame on the table. Additionally, the prior can be extended to SO(3) by using quaternions on $\mathbb{S}^3$.

Hand configuration Finally, the prior of the hand configuration is $p(\mathbf{h} \mid o, \mathbf{P}) = p(\mathbf{x} \mid o, \mathbf{P}) p(\mathbf{q})$.

B. Ratio

The likelihood function $p(S \mid \mathbf{h}, o, \mathbf{P})$ and the evidence $p(S \mid o, \mathbf{P})$ are both intractable. However, drawing samples from forward models remains feasible with physical simulators, hence enabling likelihood-free Bayesian inference algorithms. In particular, the likelihood-to-evidence ratio $r(S \mid \mathbf{h}, o, \mathbf{P})$ (Eq. (4)) can be approximated by a neural network $r_\phi(S \mid \mathbf{h}, o, \mathbf{P})$ using amortized neural ratio estimation [5]. Here, instead of using only point-wise features $\psi(\mathbf{P}, \mathbf{x})$ as input for the ratio, we add a crop of the features plane $\mathbf{c}_{xy}(\mathbf{P})$, centred at $\mathbf{x}$, sized by the gripper and rotated by $\mathbf{q}$. These features $\Psi(\mathbf{P}, \mathbf{h})$ local in the neighbourhood of the grasping point are nearly equivariant to a 2D transformation applied to the object, *i.e* $\mathbf{T}\Psi(\mathbf{P}, \mathbf{h}) \approx \Psi(\mathbf{T}\mathbf{P}, \mathbf{h}), \mathbf{T} \in \text{SE}(2)$.

C. Posteriors

Given our scene-dependent prior and our likelihood-to-evidence ratio, we approximate the posterior over the hand configurations as

$$\hat{p}(\mathbf{h} \mid S, o, \mathbf{P}) = r_\phi(S \mid \mathbf{h}, o, \mathbf{P}) p(\mathbf{h} \mid o, \mathbf{P}). \tag{5}$$

This approximation defines an implicit function [6] on the product of manifolds $\mathbb{R}^3 \times \mathbb{S}^1$ that is both fully tractable and differentiable, allowing the use of gradient-based methods for computing the MAP and sampling. Therefore, we can use Markov Chain Monte Carlo methods to sample from our posterior approximation $\hat{p}(\mathbf{h} \mid S, o, \mathbf{P})$. In particular, based on [5], we use a likelihood-free version of HMC by replacing the intractable likelihood with the ratio. The potential energy function is defined as $U(\mathbf{h}) \triangleq -\log p(S \mid \mathbf{h}, o, \mathbf{P})$ and its difference is $U(\mathbf{h}_t) - U(\mathbf{h}') = \log r(S \mid \mathbf{h}_t, \mathbf{h}')$. The gradient used in the integration step is given by $\nabla_{\mathbf{h}} U(\mathbf{h}) = -\nabla_{\mathbf{h}} \log r(S \mid \mathbf{h}, o, \mathbf{P})$. To account for the geometry of the parameter space, we then further extend our likelihood-free HMC with a geodesic integrator. The geodesic Monte Carlo scheme uses geodesic flow to perform the integration while staying on the manifold. To this end, orthogonal projections and geodesics are needed in a closed form. Finally, geodesic Monte Carlo can be applied to a product of manifolds $\mathcal{M}_1 \times \mathcal{M}_2 : \{(x_1, x_2) : x_1 \in \mathcal{M}_1, x_2 \in \mathcal{M}_2\}$, such as $\mathbb{R}^3 \times \mathbb{S}^1$ in our specific case. Geodesic flow can be executed in parallel; only the evaluation of the ratio requires both variables $\mathbf{x}$ and $\mathbf{q}$. In this manner, we can sample from the posterior density defined on a smooth manifold with closed-form geodesic. The full sampling procedure is summarized in Algorithm 1 of Appendix A.

IV. Experiments

We assess our approach on a robotic grasping task in both simulation and real-world settings. We generate data in a *packed* scenario, as defined in [7]. Additional experiments can be found in Appendix B and C.

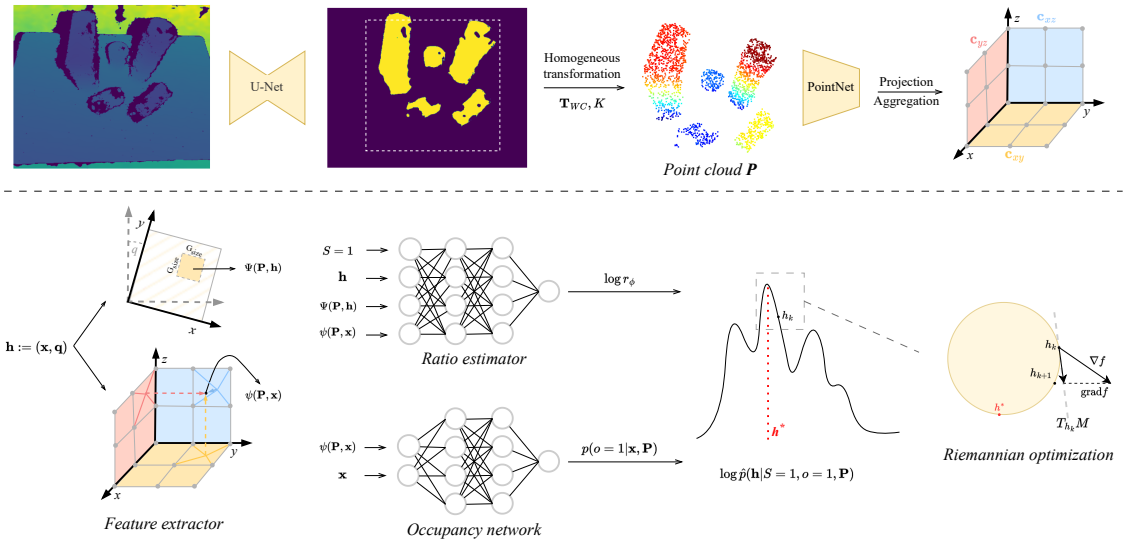


Fig. 2: Our grasp inference pipeline, as run on the scene of Fig. 1. It begins with a noisy depth image of the scene, from which we first separate the objects from the background using a U-Net [4]. We then generate three canonical feature planes following the approach in [3]. To evaluate a given $\mathbf{h}$, we extract point-wise $\psi(\mathbf{P}, \mathbf{x})$ and local $\Psi(\mathbf{P}, \mathbf{h})$ features and feed them to the ratio and occupancy networks. Using the resulting differentiable posterior and Riemannian optimization, we finally identify the most plausible hand configuration $\mathbf{h}^*$.

A. Grasp inference pipeline

Starting from the depth image I, we remove the background and extract only pixels of the objects with a segmentation model based on a U-Net architecture [4]. Then, we convert I to a point cloud $\mathbf{P}$ with 2048 points, which passes through an encoder and produces three canonical feature planes. We then extract point-wise $\psi(\mathbf{P}, \mathbf{x})$ and local $\Psi(\mathbf{P}, \mathbf{h})$ features to evaluate the occupancy and the ratio networks. To smooth the posterior approximation, we use an ensemble of 6 ratio models. Finally, we compute the MAP by maximizing the log posterior density [8]. To this end, we use a Riemannian gradient ascent which preserves the nonlinearity of $\mathbb{S}^1$. A visual summary of our method is given in Fig. 2.

Our likelihood-free geodesic Monte Carlo is used to sample plausible hand configurations $\mathbf{h} \sim \hat{p}(\mathbf{h} \mid S = 1, o = 1, \mathbf{P})$ for successful grasps, as shown in Fig. 3. Although our conditional prior distributes density across everywhere on the objects, the posterior assigns minimal density to the bottom of objects when multiple objects are present on the table. This occurs due to potential collisions between the gripper and the table, or the gripper and other objects. Regarding rotation, the posterior resembles the prior because multiple objects on the table, some with axial symmetry, allow for a wide range of grasp orientations. When only a single object is used, distinct modes can be observed, indicating that our posterior captures meaningful orientations, as shown in Fig. 4.

B. Simulation results

To compare our approach, we evaluated it against Grasp Pose Detection (GPD) [9] and Volumetric Grasping Network (VGN) [7] in terms of success rate and percent cleared [7] using the same dataset and a similar scenario (Table I). Our model achieved a high success rate of 91.1%, which is very close to VGN's best success rate of 91.5%. However, our model operated in a more constrained setup with 4 DoF instead of 6, limiting

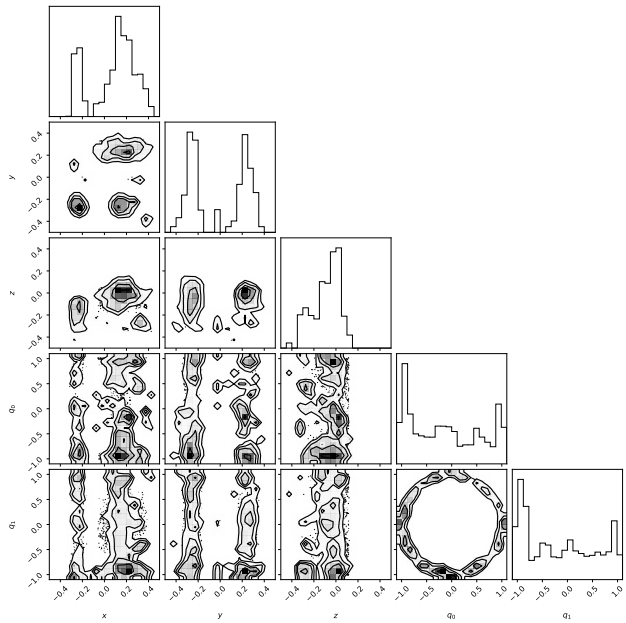


Fig. 3: Estimated posterior distribution $\hat{p}(\mathbf{h} \mid S = 1, o = 1, \mathbf{P})$ of plausible successful hand configurations, for the scene shown in Fig. 1.

Fig. 4: (left) Estimated posterior distribution of the orientation $\hat{p}(\mathbf{q} \mid S = 1, o = 1, \mathbf{P})$. (right) Single object.

TABLE I: Success rates (%) and % cleared for picking experiments for the packed scenario with 5 objects over 200 rounds.

Method	Success rate	% cleared
Simulation results		
GPD [9]	73.7	72.8
VGN ($\varepsilon = 0.95$) [7]	91.5	79
VGN ($\varepsilon = 0.9$) [7]	87.6	80.4
VGN ($\varepsilon = 0.85$) [7]	80.4	79.9
Ours	91.1	77
Real-world results		
Ours	95.6	88

the gripper's movement and increasing the risk of collisions. This limitation resulted in a slightly lower percent cleared compared to VGN. Furthermore, the gripper used was primarily designed for larger objects, making it challenging to handle smaller ones.

C. Real-world results

We adopt the setup from [8]. We perform 20 rounds with 5 objects among 13 (see Fig. 5), using a protocol similar to the simulation experiments. The objects are chosen based on their availability in the lab and whether they were seen or unseen during training. For novel objects, we achieve a success rate of **95.6**% and a percent cleared of **88**%, showing the strong adaptability and performance of our approach. The discrepancy between the simulation and real-world setup is overcome without any decrease in performance. In both simulation and real-world settings, the majority of failure cases are due to insufficient friction forces, causing the objects to slip.

V. Related Work

Grasp sampling strategies that generate data for machine learning methods can be categorized based on their coverage of the hand configuration space $\mathcal{H}$ [10]. Simple strategies like uniform sampling provide direct density estimation but are highly inefficient. Heuristic methods, on the other hand, mainly rely on object geometry. Notably, the most efficient strategies [11], [12] are not suitable

Fig. 5: (left) Object assets used in the real setup. (right) Successful grasp of a mug.

for complex settings such as multi-fingered grippers. Our occupancy networks-based approach offers direct density estimation, efficient sampling, and does not depend on specific object or gripper assumptions.

Representing a 3D scene as a parameterized function using neural networks has recently gained substantial interest. Occupancy networks [13] determine whether a point is occupied or not. However, they lack *equivariance* for translations and rotations. To overcome this limitation, translation-equivariant feature planes are derived from convolutional networks, resulting in Convolutional Occupancy Networks [3]. Achieving rotation-equivariance is more challenging, but innovative solutions have recently emerged [14], [15].

Various methods exist for sampling from a distribution defined on a Riemannian manifold. Adapting Markov Chain Monte Carlo methods [16] to Riemannian manifolds allows sampling from a differentiable and tractable likelihood. Our approach eliminates the need for an explicit likelihood. Normalizing flows [17] which target density defined on manifolds can rapidly sample from the posterior distribution. However, it remains unclear how to use their gradients for Riemannian optimization.

Finally, probabilistic approaches for grasping problems typically depend on explicit likelihood functions that model the probability of success or a grasp quality metric related to an observation and a grasp pose [7], [18]–[20]. Closer to our work, a similar study [8] uses simulation-based inference to compute the maximum a posteriori through Riemannian gradient ascent. However, this approach uses a heuristic prior and cannot accommodate multiple objects.

VI. Conclusion

We have shown that Bayesian inference can be effectively applied to robotic grasping in complex and noisy environments. Through innovative enhancements, we have improved the sample efficiency of the inference pipeline. Our approach can manage tasks with escalating complexity and proves valuable for real-world robotic applications. Future research will focus on controlling the complete 6-DoF of the robotic hand.

Acknowledgement

Norman Marlier acknowledges the Belgian Fund for Research training in Industry and Agriculture for its financial support (FRIA grant). Computational resources have been provided by the Consortium des Équipements de Calcul Intensif (CÉCI), funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under Grant No. 2.5020.11 and by the Walloon Region.

References

[1] K. Cranmer, J. Brehmer, and G. Louppe, "The frontier of simulation-based inference," *Proceedings of the National Academy of Sciences*, 2020.

[2] S. Byrne and M. Girolami, "Geodesic monte carlo on embedded manifolds," *Scandinavian Journal of Statistics*, vol. 40, no. 4, pp. 825–845, 2013.

[3] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, 2020, pp. 523–540.

[4] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[5] J. Hermans, V. Begy, and G. Louppe, "Likelihood-free MCMC with amortized approximate ratio estimators," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020, pp. 4239–4248. [Online]. Available: http://proceedings.mlr.press/v119/hermans20a.html

[6] K. Murphy, C. Esteves, V. Jampani, S. Ramalingam, and A. Makadia, "Implicit-pdf: Non-parametric representation of probability distributions on the rotation manifold," *arXiv preprint arXiv:2106.05965*, 2021.

[7] M. Breyer, J. J. Chung, L. Ott, S. Roland, and N. Juan, "Volumetric grasping network: Real-time 6 dof grasp detection in clutter," in *Conference on Robot Learning*, 2020.

[8] N. Marlier, O. Brüls, and G. Louppe, "Simulation-based bayesian inference for robotic grasping," *arXiv preprint arXiv:2303.05873*, 2023.

[9] A. Ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1455–1473, 2017.

[10] C. Eppner, A. Mousavian, and D. Fox, "A billion ways to grasp: An evaluation of grasp sampling schemes on a dense, physics-based grasp data set," in *Robotics Research: The 19th International Symposium ISRR*. Springer, 2022, pp. 890–905.

[11] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," 2017.

[12] X. Yan, J. Hsu, M. Khansari, Y. Bai, A. Pathak, A. Gupta, J. Davidson, and H. Lee, "Learning 6-dof grasping interaction via deep geometry-aware 3d representations," 2018.

[13] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4460–4470.

[14] Y. Chen, B. Fernando, H. Bilen, M. Nießner, and E. Gavves, "3d equivariant graph implicit functions," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*. Springer, 2022, pp. 485–502.

[15] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. J. Guibas, "Vector neurons: A general framework for so (3)-equivariant networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 200–12 209.

[16] C. Liua and J. Zhub, "Geometry in sampling methods: A review on manifold mcmc and particle-based variational inference methods," *Advancements in Bayesian Methods and Implementations*, vol. 47, p. 239, 2022.

[17] D. J. Rezende, G. Papamakarios, S. Racaniere, M. Albergo, G. Kanwar, P. Shanahan, and K. Cranmer, "Normalizing flows on tori and spheres," in *International Conference on Machine Learning*. PMLR, 2020, pp. 8083–8092.

[18] Q. Lu, K. Chenna, B. Sundaralingam, and T. Hermans, "Planning multi-fingered grasps as probabilistic inference in a learned deep network," in *Robotics Research*. Springer, 2020, pp. 455–472.

[19] J. Cai, J. Cen, H. Wang, and M. Y. Wang, "Real-time collision-free grasp pose detection with geometry-aware refinement using high-resolution volume," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1888–1895, 2022.

[20] M. Van der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans, "Learning continuous 3d reconstructions for geometrically aware grasping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 11 516–11 522.

Appendix

A. *Likelihood-free geodesic Monte Carlo*

Algorithm 1: Likelihood-free geodesic Hamiltonian Monte Carlo

Input: A Manifold $\mathcal{M}$
Initial parameters $\mathbf{h}_0$
Prior $p(\mathbf{h} \mid \mathbf{P})$
Momentum distribution $p(\mathbf{v})$
Trained classifier $d_\phi(S, \mathbf{h}, \mathbf{P})$
Observations $S, \mathbf{P}$
Output: Markov chain $\mathbf{h}_{1:T}$

$t \leftarrow 0$
$\mathbf{h}_t \leftarrow \mathbf{h}_0$
for $t < T$ **do**
  $\mathbf{v}_t \sim p(\mathbf{v})$
  $\mathbf{v}_t \leftarrow \boldsymbol{\pi}_{\mathbf{h}_t}(\mathbf{v}_t)$
  $k \leftarrow 0$
  $\mathbf{v}_k \leftarrow \mathbf{v}_t$
  $\mathbf{h}_k \leftarrow \mathbf{h}_t$
  for $k < L$ **do**
    $\mathbf{v}_k \leftarrow \mathbf{v}_k + \frac{\epsilon}{2} \nabla_{\mathbf{h}_k} \log r(S \mid \mathbf{h}_k, \mathbf{P})$
    $\mathbf{v}_k \leftarrow \boldsymbol{\pi}_{\mathbf{h}_k}(\mathbf{v}_k)$
    $\mathbf{h}_k \leftarrow \gamma(\epsilon), \gamma(0) = \mathbf{h}_k$
    $\mathbf{v}_k \leftarrow \dot{\gamma}(\epsilon), \dot{\gamma}(0) = \mathbf{v}_k$
    $\mathbf{v}_k \leftarrow \mathbf{v}_k + \frac{\epsilon}{2} \nabla_{\mathbf{h}_k} \log r(S \mid \mathbf{h}_k, \mathbf{P})$
    $\mathbf{v}_k \leftarrow \boldsymbol{\pi}_{\mathbf{h}_k}(\mathbf{v}_k)$
    $k \leftarrow k + 1$
  end
  $\lambda_k \leftarrow \log r(S \mid \mathbf{h}_k, \mathbf{P}) + \log p(\mathbf{h}_k \mid \mathbf{P}) - \frac{1}{2} \mathbf{v}_k^T \mathbf{v}_k$
  $\lambda_t \leftarrow \log r(S \mid \mathbf{h}_t, \mathbf{P}) + \log p(\mathbf{h}_t \mid \mathbf{P}) - \frac{1}{2} \mathbf{v}_t^T \mathbf{v}_t$
  $\rho \leftarrow \min(\exp(\lambda_k - \lambda_t), 1)$
  $\mathbf{h}_{t+1} \leftarrow \begin{cases} \mathbf{h}_k & \text{with a probability } \rho \\ \mathbf{h}_t & \text{with a probability } 1 - \rho \end{cases}$
  $t \leftarrow t + 1$
end
return $\mathbf{h}_{1:T}$

B. Sampling the orientation: toy problem

Given a model parameter sample $q_\theta \in \mathbb{S}^d$, the forward generative process is defined as:

$$\nu = q_\theta \tag{6}$$

$$\kappa = 20 \tag{7}$$

$$q_x \sim \exp(\kappa \nu^T q_x) \tag{8}$$

with the prior $p(q_\theta) \overset{\Delta}{=} \texttt{SphericalUniform}(d)$. It follows the true posterior $p(q_\theta \mid q_x) \propto \exp(\kappa q_x^T q_\theta)$. We use a MLP of 3 layers with 64 neurons to approximate the likelihood-to-evidence ratio. All the activation functions are ReLU except the last one which is linear. We train the ratio with 1000000 samples with a batch size of 8000 for 50 epochs. For the geodesic HMC, we use 100 chains and 2000 transitions with a burn in of 1000. The integration parameters are $\epsilon = 0.01, L = 20$. The approximate posterior shares the same structure that the true posterior, demonstrating its accuracy (Fig. 6). Moreover, we conduct a quantitative analysis by computing the Maximal Mean Discrepancy (MMD) for $\mathbb{S}^1$ and $\mathbb{S}^3$. We obtain respectively $0.0028 \pm 5.84e^{-6}$ and $0.01 \pm 3e^{-5}$ for an identity kernel between the two geodesic means for 10 different observations q_x.

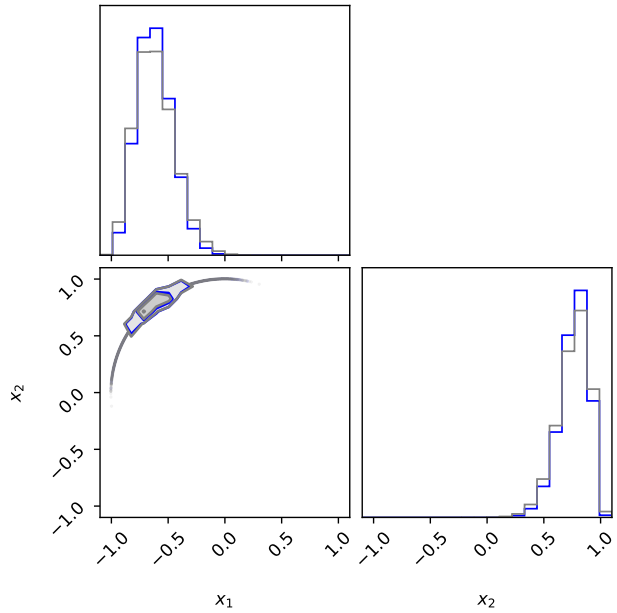


Fig. 6: Posterior for $\mathbb{S}^1$. In blue, the distribution obtained through GMC with the ratio and in grey, the ground truth posterior.

C. Sampling the position: multiple objects scene

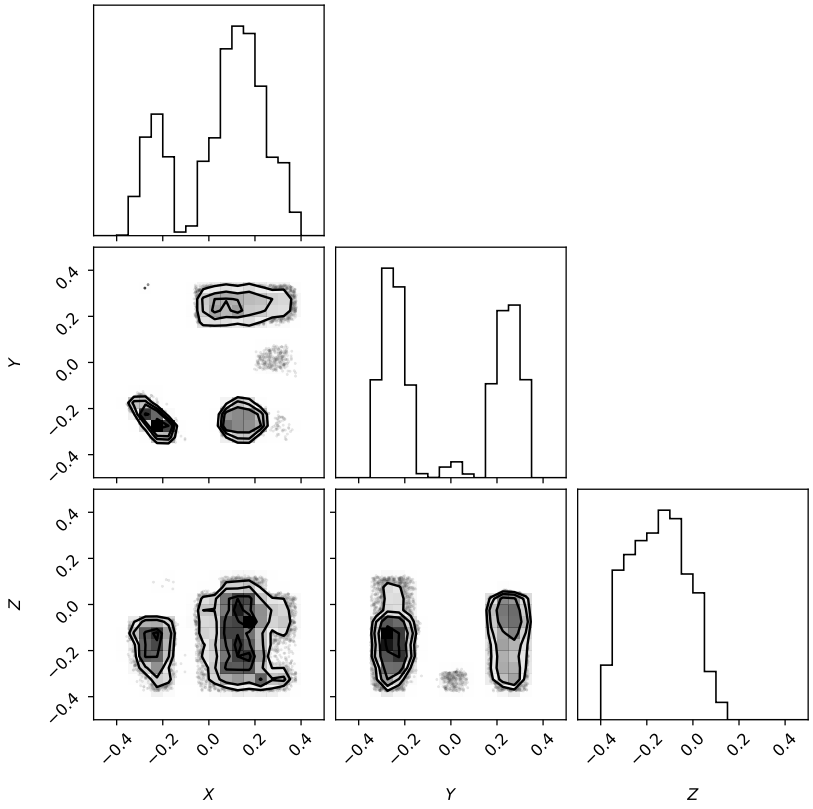


Fig. 7: Approximate posterior $p(\mathbf{x}|o=1,\mathbf{P})$ of a scene with 5 different objects.

In this experiment, we have a scene containing five objects with varying levels of difficulty, as shown in Fig. 2. We use a convolutional occupancy network [3] that is trained for 120,000 iterations with a batch size of 32 samples. The network has a resolution of 128 and a feature dimension of 32. To sample from the posterior $p(\mathbf{x}|o=1,\mathbf{P}) \propto p(o=1|\mathbf{x},\mathbf{P})p(\mathbf{x})$, as we previously explained, we employed HMC with specific hyper-parameters. These include 100 chains of 5000 transitions and a burn-in of 1000. The integration parameters are $\epsilon = 0.01$, $L = 20$. The chains' initials point are sampled uniformly in the bounding box of the objects. The corner plot in Fig. 7 illustrates the resulting posterior, which aligns with the location and shape of four out of the five objects and the potential grasping point. However, the occupancy of the fifth object cannot be recovered due to either being regarded as noise or having a much smaller density compared to the other objects.

9.3 EPILOGUE

9.3.1 *Advantages*

Our new prior, which represents implicitly the scene, brings high flexibility to our framework. By using a new variable in our modeling, the occupancy, we can integrate nicely the occupancy network in our pipeline. The first requirement of neural ratio estimation is that the prior can be evaluated to compute the posterior, which is possible because occupancy networks are density estimators. They accurately localize the positions occupied by the objects and put no restriction on the number of objects present on the table. The second requirement necessitates sampling from prior to generate samples from the joint distribution. Occupancy networks provide a target density and are fully differentiable, thus providing gradients. Therefore, they can be plugged into a Markov Chain Monte Carlo scheme to generate samples with respect to their density. The last requirement requires gradients of the posterior with respect to the grasp pose to compute the maximum a posteriori, which is already the case. However, occupancy networks also possess several advantages. Firstly, they are powerful feature extractors because they implicitly represent the scene, while simpler networks can struggle for this task. These features can be in turn used by the neural ratio estimator, facilitating the learning of the likelihood ratio function. Secondly, only a single point cloud is needed to predict the occupancy of the scene. This drastically reduces the sensor data acquiring time compared to the truncated signed distance function used in Chapter 8.

9.3.2 *Limitations*

While our newly learned prior shows huge advantages compared to a heuristic prior, it has some limitations and drawbacks. The first straightforward drawback is the computation time. Using a big model requires more time for the optimization procedure and slows down this part of the pipeline. Another limitation comes from the assumption that the grasping contact point has to be **inside** the object. This strong assumption does not take advantage of hanse or holes in object shapes, making the prior irrelevant for some robotic tasks. Furthermore, the prior only applies to the position of the grasp pose and not the orientation. Finally, sampling from this prior using MCMC is quite slow which is not very interesting because simulation-based inference methods require a lot of data, especially in the case of rare events (successful grasps in our case).

9.3.3 *Conclusion and opportunities*

Our newly learned prior increases the complexity of the task and still has a very high success rate, about 95%. Even if the grasp pose has 4 DoF, we still improve the framework. Moreover, because we developed an algorithm to sample from a distribution defined on

manifolds, we unlock new possibilities, such as sequential neural ratio estimation. We leave this for the next chapter.

10

GRASPING MANY OBJECTS IN AN UNRESTRICTED SETUP

Outline

We propose a sequential approach to approximate the posterior distribution. While Chapter 8 introduced a scene-based prior designed only to capture relevant positions for grasping and simplifies the orientation by limiting its DoF, this chapter takes a step further by sequentially refining our posterior approximation, hereby releasing the constraints put on the orientation. This allows us to explore the high-dimensional space of gripper poses. We validate the results on realistic benchmarks and show that 6 DoF approaches converge toward 4 DoF.

10.1 PROLOGUE

All the previous chapters of Part iii overcome the task of robotic grasping of increased complexity by using new components in the framework. This complexity can be more degrees of freedom to infer for the grasp pose or a larger diversity and variability for the task itself: the object can be anywhere on the table or several objects can be present on the table. This last chapter introduces a systematic method to learn the full grasp pose with several objects present on the table.

Our method is based on Chapter 9 but instead uses a sequential approach to approximate the posterior density by sampling from a previous approximation of the posterior, which acts as a new prior, and learned from the newly generated samples until the convergence criteria are met. This sequential approximation allows us to refine the posterior, by decreasing low-density areas and increasing high-density areas. This provides great flexibility, compared to analytical densities. Among them, only the Bingham distribution [Bingham, 1974] for quaternion gives enough flexibility but comes with many issues such as sampling or fine-tuning the parameters of the distribution. In conclusion, this paper can be viewed as a synthesis of all the contributions.

Update For this contribution, there is no change in the modeling. However, the previous Section 9.3 only uses one degree of freedom for the orientation, thus staying on $\mathbb{S}^1$, which facilitated the task. This contribution tackles the problem of inferring the full grasp pose.

Reading tips In this article, we use different visualizations to represent distribution over the rotation from the grasp pose. While we use the joint plot to illustrate the

distribution over quaternions, [Murphy et al., 2021] provides another visualization based on the Mollweide projection.

10.2 THE PAPER: GRASPING UNDER UNCERTAINTIES: SEQUENTIAL NEURAL RATIO ESTIMATION FOR 6-DOF ROBOTIC GRASPING

Grasping under uncertainties: Sequential Neural Ratio Estimation for 6-DoF robotic grasping

Norman Marlier[1*] Olivier Brüls[2] Gilles Louppe[3]

***Abstract*— We introduce a novel approach to 6-DoF robotic grasping based on simulation-based inference. Our approach combines sequential neural ratio estimation with a neural implicit representation for the Bayesian inference of hand configurations in cluttered environments. We propose to compute the maximum a posteriori by gradient descent, more specifically using Riemannian gradient descent, to preserve the geometry of the rotation space and capitalize on the full differentiability of our model. We demonstrate the capabilities of our approach on a grasping benchmark both in simulation and on a real robot. Our performance generalizes well across different scenarios, achieving high success rates.**

I. Introduction

Grasping is a fundamental skill for robots. While industrial robots perform tasks in very controlled environments, novel applications require robots to adapt to new unstructured environments. Determining robust grasp poses from raw perception data is, for this reason, essential to the wider deployment of robots in the real world. However, dealing with uncertainties arising from rich nonsmooth contact dynamics and sensor noise is still an open problem. To address these challenges, Bayesian inference provides a principled framework to recast grasping as an inference problem under uncertainties. The nature of robotic grasping, however, often involves highly complex dynamics relating the tentative grasp pose to the final outcome, making the necessary likelihood function intractable. Additionally, grasp poses in these tasks come with hard constraints from the robot's kinematics, further complicating the inference procedure.

In this paper, we tackle the problem of generating robust grasp poses by proposing an original approach based on simulation-based inference algorithms [1], a family of methods that learn a component of the Bayes rule through stochastic forward simulations. Our contributions are summarized as follows:

- We bring simulation-based Bayesian inference methods to robotic grasping. By sequentially learning a model for the likelihood-to-evidence ratio and using an implicit neural representation, we derive an amortized and differentiable posterior for grasp poses.

*The authors come from the University of Liège, Belgium
[1]norman.marlier@uliege.be
[2]o.bruls@uliege.be
[3]g.louppe@uliege.be

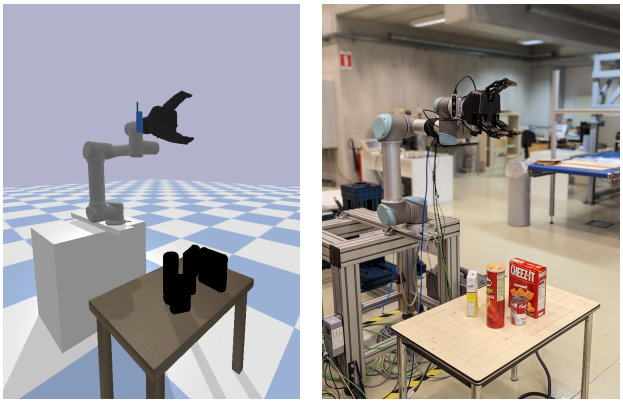


Fig. 1: Our benchmark scene. (left) The simulated environment. (right) The real setup.

- We make use of Riemannian manifold methods to sample from densities defined on smooth Riemannian manifolds and optimize the grasp poses.
- We validate the effectiveness of our method through both simulated and real experiments, demonstrating remarkable grasping performance.

II. Problem statement

We consider the problem of planning 6-DoF hand configurations for a robotic gripper handling various unknown objects on a table, observed with a depth camera. A benchmark scene is shown in Fig. 1.

A. Description

The robot arm, which consists of 6 or 7 DoF, operates a gripper within a cubic workspace of size l, featuring a planar tabletop. The scene is observed with a depth camera mounted on the robot flange. Our objective is to identify the most plausible hand configuration conditioned by a grasp success and the observed point cloud. Then, a path planner computes a collision-free joint trajectory, enabling the robot to reach the desired grasp pose and safely remove the selected object from the tabletop.

B. Notations

Frames We use several reference frames. The world frame $\underrightarrow{\mathcal{F}}_{\mathrm{W}}$ and the workspace frame $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$ can be chosen freely and are not tied to a physical location. The world frame is used for the robot and the sensor, while the workspace frame is used for our inference system. $\underrightarrow{\mathcal{F}}_{\mathrm{C}}$ and $\underrightarrow{\mathcal{F}}_{\mathrm{E}}$ correspond respectively to the camera and the end-effector frames.

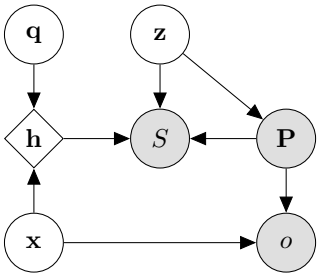


Fig. 2: Probabilistic graphical model of the environment. Gray nodes represent observed variables, white nodes represent unobserved variables, and diamond nodes represent deterministic functions.

Hand configuration The hand configuration $\mathbf{h} \in \mathcal{H} = \mathbb{R}^3 \times \mathbb{S}^3$ is defined as the pose $(\mathbf{x}, \mathbf{q}) \in \mathbb{R}^3 \times \mathbb{S}^3$ of the hand, where $\mathbf{x}$ is the vector $\vec{SE}$ expressed in $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$ and $\mathbf{q}$ is the 3D rotation from $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$ to $\underrightarrow{\mathcal{F}}_{\mathrm{E}}$ represented using unit quaternions with scalar last format.

Binary metric A binary variable $S \in \{0, 1\}$ indicates if the grasp fails ($S = 0$) or succeeds ($S = 1$).

Observation Given the depth image I with its corresponding transformation camera to world $\mathbf{T}_{\mathrm{WC}}$ and camera intrinsic matrix K, we construct a point cloud $\mathbf{P} \in \mathbb{R}^{2048\times 3}$ expressed in $\underrightarrow{\mathcal{F}}_{\mathrm{S}}$.

Occupancy A binary variable $o \in \{0, 1\}$ indicates if a point $\mathbf{p} \in \mathbb{R}^3$ is occupied by any object of the scene.

Latent variables Unobserved variables $\mathbf{z}$ capture uncertainties about the nonsmooth dynamics of contact, the sensor noise, as well as the number of objects and their geometry.

C. Probabilistic modelling

We model the scene and the grasping task according to the Bayesian network shown in Fig. 2. This choice of structure is motivated by the dependencies observed in the system: S depends on $\mathbf{z}$, $\mathbf{P}$, and $\mathbf{h}$; o depends on $\mathbf{P}$ and $\mathbf{x}$; and $\mathbf{P}$ depends on $\mathbf{z}$. Such a structure facilitates straightforward generation procedures: $\mathbf{z}$ and $\mathbf{h}$ are sampled from their respective priors, while $\mathbf{P}$ and S are obtained through forward physical simulations.

D. Grasping as inference

We formulate the problem of grasping as the Bayesian inference of the hand configuration $\mathbf{h}^*$ that is a posteriori the most likely given a successful grasp, an occupancy o at $\mathbf{x}$ and a point cloud $\mathbf{P}$. That is, we are seeking the maximum a posteriori (MAP) estimate

$$\mathbf{h}^* = \arg\max_{\mathbf{h}} \; p(\mathbf{h}|S = 1, o = 1, \mathbf{P}), \tag{1}$$

from which we then compute the joint trajectory

$$\tau_{1:m} = \Lambda(\tau_0, \textsc{ik}(\mathbf{h}^*), \mathbf{P}), \tag{2}$$

where IK is an inverse kinematic solver, $\tau_{1:m}$ are waypoints in the joint space, $\tau_m = \mathrm{IK}(\mathbf{h}^*)$ with $\mathbf{h}^*$ expressed in $\underrightarrow{\mathcal{F}}_{\mathrm{W}}$ and Λ is a path planner.

III. Simulation-based inference

From the Bayes rule, the posterior of the hand configuration is

$$p(\mathbf{h}|S, o, \mathbf{P}) = \frac{p(S \mid \mathbf{h}, o, \mathbf{P})}{p(S \mid o, \mathbf{P})} p(\mathbf{h} \mid o, \mathbf{P}) \tag{3}$$

A. Priors

The prior $p(\mathbf{h} \mid o, \mathbf{P})$ represents our domain knowledge about the hand configuration without knowing if it leads to a successful grasp or not (S is not observed).

Position Objects present in the workspace occupy a small volume of the whole workspace. In order to facilitate the exploration of potential grasp poses, we make the assumption that, at a successful grasp pose, the position of the finger tips when the gripper is closed lies inside the object. This information is modelled by the occupancy variable o. Formally, we defined our prior as

$$p(\mathbf{x}|o = 1, \mathbf{P}) = \frac{p(o = 1|\mathbf{x}, \mathbf{P})}{p(o|\mathbf{P})} p(\mathbf{x}). \tag{4}$$

Here, $p(o|\mathbf{x}, \mathbf{P})$ represents the likelihood of occupancy o, $p(\mathbf{x})$ denotes a uniform distribution over the workspace, and $p(\mathbf{x}|\mathbf{P})$ simplifies to $p(\mathbf{x})$ due to independence.

We model the likelihood of occupancy $p(o|\mathbf{x}, \mathbf{P})$ with a convolutional occupancy network [2]. This network first generates an embedding on the three canonical planes $\mathbf{c}_{xy}(\mathbf{P}), \mathbf{c}_{xz}(\mathbf{P})$ and $\mathbf{c}_{yz}(\mathbf{P})$. Subsequently, a feature vector is obtained by summing point-wise features computed via bilinear interpolation at the desired position $\mathbf{x}_{\mathrm{D}}$, *i.e* $\psi(\mathbf{P}, \mathbf{x}_{\mathrm{D}}) = \mathbf{c}_{xy}(\mathbf{P})(\mathbf{x}_{\mathrm{D}}) + \mathbf{c}_{xz}(\mathbf{P})(\mathbf{x}_{\mathrm{D}}) + \mathbf{c}_{yz}(\mathbf{P})(\mathbf{x}_{\mathrm{D}})$. Finally, this embedding is passed through a fully connected network to produce the occupancy output.

We use a Markov chain Monte Carlo (MCMC) scheme to sample from the position prior because we have access to the target density $p(\mathbf{x}|o = 1, \mathbf{P}) \propto p(o = 1|\mathbf{x}, \mathbf{P})p(\mathbf{x})$ with the convolutional occupancy network. Furthermore, it is fully differentiable, making possible the use of the Hamiltonian Monte Carlo [3], a variant of MCMC using gradients to efficiently explore the position prior space.

Orientation The prior of the orientation $\mathbf{q}$ is a uniform distribution over the hypersphere $\mathbb{S}^3$. This prior is *invariant* to any rotation $\mathbf{R} \in \mathbb{SO}(2)$ applied to individual objects. This is a desirable property as it does not assume any preferred orientation.

B. Neural ratio estimation

The likelihood function $p(S \mid \mathbf{h}, o, \mathbf{P})$ and the evidence $p(S \mid o, \mathbf{P})$ are both intractable because no closed-form relations exist to express the success of a grasp as it derives from nonsmooth contact dynamics. It makes standard Bayesian inference procedures such as Markov chain Monte Carlo unusable. However, drawing samples from forward models remains feasible with physical simulators, hence enabling simulation-based Bayesian inference algorithms.

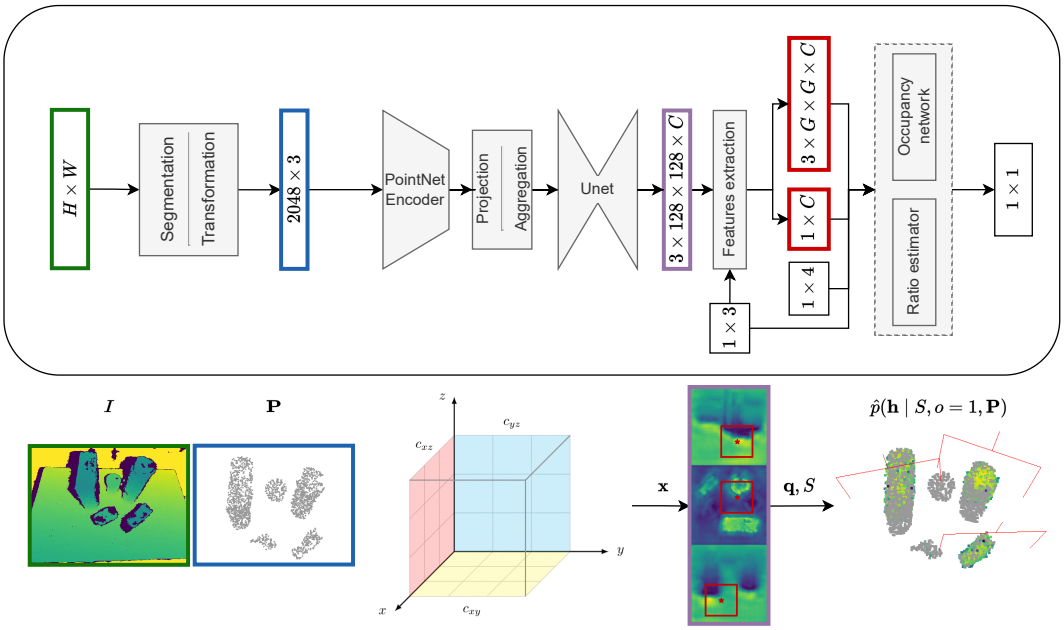


Fig. 3: Posterior inference. The depth image I (in green) passes through a segmentation model which extracts pixels belonging to objects. Then, these pixels are transformed into point cloud $\mathbf{P}$ (in blue) with intrinsic and extrinsic matrices of the depth camera. We then generate three canonical feature planes $\mathbf{c}_{xy}(\mathbf{P}), \mathbf{c}_{xz}(\mathbf{P})$ and $\mathbf{c}_{yz}(\mathbf{P})$ and extract for a given $\mathbf{h}$ point-wise $\psi(\mathbf{P}, \mathbf{x})$ and local $\Psi(\mathbf{P}, \mathbf{h})$ features. They are feed to the ratio and occupancy networks which output the posterior density $\hat{p}(\mathbf{h} \mid S, o, \mathbf{P})$.

First, we express the likelihood-to-evidence ratio as

$$r(S \mid \mathbf{h}, o, \mathbf{P}) = \frac{p(S \mid \mathbf{h}, o, \mathbf{P})}{p(S \mid o, \mathbf{P})} \tag{5}$$

$$= \frac{p(S, \mathbf{h} \mid o, \mathbf{P})}{p(S \mid o, \mathbf{P}) p(\mathbf{h} \mid o, \mathbf{P})} \tag{6}$$

By adapting the approach described in [4] for likelihood ratio estimation, we train a neural network classifier d_ϕ which approximates $r(S|\mathbf{h}, o, \mathbf{P})$. This network is trained to distinguish tuples $(S, \mathbf{h}, o, \mathbf{P})$ (labeled $y = 1$) sampled from the joint distribution $p(S, \mathbf{h} \mid o, \mathbf{P})$ and tuples $(S, \mathbf{h}, o, \mathbf{P})$ (labeled $y = 0$) sampled from the product of the marginals $p(S \mid o, \mathbf{P})p(\mathbf{h} \mid o, \mathbf{P})$. The Bayes optimal classifier that minimizes the cross entropy is given by

$$d^*(S, \mathbf{h}, o, \mathbf{P}) = \frac{p(S, \mathbf{h} \mid o, \mathbf{P})}{p(S, \mathbf{h} \mid o, \mathbf{P}) + p(S \mid o, \mathbf{P}) p(\mathbf{h} \mid o, \mathbf{P})} \tag{7}$$

which recovers the likelihood-to-evidence ratio as

$$\frac{d^*}{1 - d^*} = \frac{p(S, \mathbf{h} \mid o, \mathbf{P})}{p(S \mid o, \mathbf{P}) p(\mathbf{h} \mid o, \mathbf{P})} = \frac{p(S \mid \mathbf{h}, o, \mathbf{P})}{p(S \mid o, \mathbf{P})} \tag{8}$$

Therefore, by modelling the classifier with a neural network d_ϕ trained on the binary classification problem, we obtain an approximate but amortized and differentiable likelihood ratio

$$r_\phi(S \mid \mathbf{h}, o, \mathbf{P}) = \frac{d_\phi(S, \mathbf{h}, o, \mathbf{P})}{1 - d_\phi(S, \mathbf{h}, o, \mathbf{P})}. \tag{9}$$

Finally, the likelihood ratio is combined with the prior to approximate the posterior as

$$\hat{p}(\mathbf{h} | S, o, \mathbf{P}) = r_\phi(S \mid \mathbf{h}, o, \mathbf{P}) p(\mathbf{h} \mid o, \mathbf{P}), \tag{10}$$

which enables immediate posterior inference despite the initial intractability of the likelihood function $p(S \mid \mathbf{h}, o, \mathbf{P})$ and the evidence $p(S \mid o, \mathbf{P})$.

Instead of passing the raw point cloud into the ratio estimator, we use the point-wise features vector from the convolutional occupancy network $\psi(\mathbf{P}, \mathbf{x})$. Furthermore, we extract local features centered around the point $\mathbf{x}$, cropped from the features planes $\mathbf{c}_{xy}(\mathbf{P}), \mathbf{c}_{xz}(\mathbf{P})$ and $\mathbf{c}_{yz}(\mathbf{P})$ and scaled to be within the gripper's size. These local features $\Psi(\mathbf{P}, \mathbf{x})$ bring information about collision and objects present in the area targeted by $\mathbf{x}$.

Using only a single neural network produces a posterior with very high frequencies. Ensembles tend to produce more conservative posteriors [5], making them suitable for optimization purposes. In our case, we take 5 models and compute the ratio as

$$\hat{r} = \frac{1}{5} \Sigma_{i=1}^{5} \hat{r}_i. \tag{11}$$

The whole pipeline is illustrated in Fig 3.

C. Sequential neural ratio estimation

Despite the use of an informative position prior, the success rate a priori remains below 1%, resulting in an imbalanced dataset that favors failure grasps ($S = 0$). To address this issue, we iteratively refine our ratio

estimator by using a sequential neural estimation scheme. Starting with our prior $p_0(\mathbf{h} \mid o, \mathbf{P}) \coloneqq p(\mathbf{h} \mid o, \mathbf{P})$, we refine the posterior from the previous iterate by setting it as the prior for the next iteration, $p_{t+1}(\mathbf{h} \mid o, \mathbf{P}) \coloneqq \hat{p}_t(\mathbf{h}|S = 1, o, \mathbf{P})$. In each iteration, we repeat the training procedure described in [4].

Sampling from the new prior p_{t+1} poses significant challenges because $\mathbf{q}$ is defined on a manifold. The geodesic Monte Carlo scheme [6] generates samples similarly to Hamiltonian Monte Carlo for distributions defined on smooth manifolds. Furthermore, geodesic Monte Carlo is applicable to a product of manifolds $\mathcal{M}1 \times \mathcal{M}2 : (x_1, x_2) : x_1 \in \mathcal{M}1, x_2 \in \mathcal{M}_2$, which, in our specific case, corresponds to $\mathbb{R}^3 \times \mathbb{S}^3$. It requires to have access to closed-form expressions of the target density, orthogonal projection and geodesics. While orthogonal projections and geodesics are available in closed-form for $\mathbb{R}^3$ and $\mathbb{S}^3$, the target density is intractable here. As described in [4], a likelihood-free variant of Hamiltonian Monte Carlo exists, where the intractable likelihood is replaced by the ratio. Therefore, by replacing the likelihood by the ratio in the geodesic Monte Carlo scheme, we can draw samples from our posterior density $\hat{p}_t(\mathbf{h}|S = 1, o, \mathbf{P})$ defined as a product of smooth manifolds.

D. Maximum a posteriori

Due to the intractability of the likelihood function and of the evidence, Eq. (1) cannot be solved analytically nor numerically. We rely instead on the approximation given by the likelihood-to-evidence ratio $\hat{r}$ to find an approximation of the maximum a posteriori (MAP) estimate as

$$\hat{\mathbf{h}}^* = \arg\max_{\mathbf{h}} \hat{r}(S = 1 \mid \mathbf{h}, o, \mathbf{P}) p(\mathbf{h} \mid o, \mathbf{P}) \tag{12}$$

$$= \arg\min_{\mathbf{h}} -\log\big(\hat{r}(S = 1 \mid \mathbf{h}, o, \mathbf{P}) p(\mathbf{h} \mid o, \mathbf{P})\big), \tag{13}$$

which we solve using gradient descent. The gradient of Eq. (13) decomposes as

$$\begin{aligned} &-\nabla_{(\mathbf{x},\mathbf{q})} \log\big(\hat{r}(S \mid \mathbf{h}, o, \mathbf{P}) p(\mathbf{h} \mid o, \mathbf{P})\big) \\ &= -\nabla_{(\mathbf{x},\mathbf{q})} \log \hat{r}(S \mid \mathbf{h}, o, \mathbf{P}) \\ &\quad - \nabla_{(\mathbf{x},\mathbf{q})} \log p(\mathbf{h} \mid o, \mathbf{P}). \end{aligned} \tag{14}$$

Since the likelihood-to-evidence ratio estimator $\hat{r}$ and our informative prior $p(\mathbf{h} \mid o, \mathbf{P})$ are modelled by a neural network, they are fully differentiable with respect to their inputs and their gradients can be computed by automatic differentiation. However, the orientation $\mathbf{q}$ belongs to a Riemannian manifold. Thus, performing gradient descent would violate our geometric assumptions (Fig. 4). Let us consider a variable $\mathcal{Z}$ on the smooth Riemannian manifold $\mathcal{M} = \mathbb{R}^3 \times \mathbb{S}^3$ with tangent space $\mathcal{T}_{\mathcal{Z}}\mathcal{M}$ and a function $f : \mathcal{M} \to \mathbb{R}$. Since $\mathbb{S}^3$ is embedded in $\mathbb{R}^4$, f can be evaluated on $\mathbb{R}^3 \times \mathbb{R}^4$, leading to the definition of the Euclidean gradients $\nabla f(\mathcal{Z}) \in \mathbb{R}^3 \times \mathbb{R}^4$. In turn, these Euclidean gradients can be transformed into

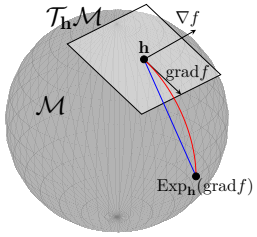


Fig. 4: Sphere manifold $\mathbb{S}^2$. $\mathbf{h}$ and $\text{Exp}_{\mathbf{h}}(\text{grad}f(\mathbf{h}))$ are points on the surface of the sphere. The red curve corresponds to the geodesic and the blue curve corresponds to the straight line in Euclidean space.

their Riemannian counterparts $\text{grad}f(\mathcal{Z})$ via orthogonal projection $\mathbf{P}_{\mathcal{Z}}$ into the tangent space $\mathcal{T}_{\mathcal{Z}}\mathcal{M}$. Therefore,

$$\text{grad}f(\mathcal{Z}) = \mathbf{P}_{\mathcal{Z}}(\nabla f(\mathcal{Z})) \tag{15}$$

where the orthogonal projection onto $\mathbb{R}^3$ is the identity $\mathbb{I}_3$ and the orthogonal projection onto $\mathbb{S}^3$ is $\mathbf{P}_{\xi}(\nabla f) = (\mathbb{I}_4 - \xi\xi^T)\nabla f$ at $\xi \in \mathbb{S}^3$. Thus, we can solve Eq. (13) by projecting Euclidean gradients of Eq. (14) to the tangent space $\mathcal{T}_{\mathcal{Z}}\mathcal{M}$ and use it in the following update rule [7]

$$\mathbf{h}_{k+1} = \text{Exp}_{\mathbf{h}_k}(-\alpha_k \text{grad}f(\mathbf{h}_k)) \tag{16}$$

with $\text{Exp}_x(v) : \mathcal{T}_x\mathcal{M} \to \mathcal{M}$ is the exponential map and α_k is the step size.

IV. Experiments: Design and Setup

We assess our approach on a robotic grasping task in both simulation and real-world settings. Specifically, we investigate three questions: 1) Does the method transfer well from simulation to real-world? 2) How the sequential improvement impacts the grasping success rate? 3) What kind of posterior distribution are learned?

A. Data generation

The data generating procedure is defined as follow:

$$\mathbf{z} \sim p(\mathbf{z}) \tag{17}$$

$$I \sim p(I \mid \mathbf{z}, \mathbf{T}_{\text{WC}}) \tag{18}$$

$$\mathbf{P} = f(I, \mathbf{T}_{\text{WC}}, K) \tag{19}$$

$$\{\mathbf{h} \sim p(\mathbf{h} \mid o = 1, \mathbf{P})\} \tag{20}$$

$$\{\tau_{1:m} \sim \Lambda(\tau_0, \text{IK}(\mathbf{h}), \mathbf{P})\} \tag{21}$$

$$\{S \sim p(S \mid \tau_{1:m}, \mathbf{z})\} \tag{22}$$

We use Pybullet [8] for implementing these functions. We use the same object assets than VGN [9] for the training and testing and we placed the objects in a *packed* scenario, as defined in [9]. The latent variables $\mathbf{z}$ are described as follow:

Number of objects We sample according to a Poisson law $N \sim Pois(4) + 1$ [9]

Object mesh We sample uniformly an object mesh from an asset of objects.

Pose of the table $\mathbf{T}_{\text{ST}}$ We randomize the position $(x, y) \sim \mathcal{N}(0, 0.008)$ and the rotation $q_{\text{T}} =$

Fig. 5: (left) Object assets used in the real setup. (right) First object removed is often the tallest one.

$(0., 0., \sin(\frac{\theta_{\text{Table}}}{2}), \cos(\frac{\theta_{\text{Table}}}{2}))$, $\theta_{\text{Table}} \sim \mathcal{U}(-5, 5)$ of the table with respect to $\underrightarrow{\mathcal{F}}_{\text{S}}$.

Pose of the object $\mathbf{T}_{\text{TO}}$ We randomize the position $(x, y) \sim \mathcal{U}(\frac{-l}{2}, \frac{l}{2})$ and the orientation $q_{\text{O}} = (0., 0., \sin(\frac{\theta_{\text{O}}}{2}), \cos(\frac{\theta_{\text{O}}}{2}))$, $\theta_{\text{O}} \sim \mathcal{U}(0, 2\pi)$ of the object with respect to $\underrightarrow{\mathcal{F}}_{\text{T}}$.

Torque applied by the fingers We randomize the final torque applied by the fingers $\tau \sim \mathcal{U}(35, 40)$.

Lateral friction coefficient We randomize the lateral friction coefficient $\mu \sim \mathcal{U}(1, 2)$.

Spinning friction coefficient We randomize the spinning friction coefficient $\gamma = \eta\mu, \eta \sim \mathcal{N}(0.002, 0.0001)$.

Depth images We add noise to the rendered depth images in simulation using the additive noise model of [10] with the same parameters.

B. Robotic setup

We carry out experiments with a Robotiq 3-finger gripper attached to a UR5 robotic arm, as shown in Fig. 1. A Intel Realsense D435i depth sensor is mounted to the flange of the robotic arm. It produces 848×480 depth images. The transformation $\mathbf{T}_{\text{FC}}$ is calibrated using hand-eye calibration from OpenCV [11]. All the devices are handled within the ROS framework. The objects, used for testing and unseen during training, are chosen based on their availability in the lab (Fig 5).

V. Experiments: Results

A. Grasping results

We compared our results with [10], as they used also an implicit representation and only one depth image. However, we benchmark only on *packed* scenario because our gripper is too big to fit small objects of the *pile* scenario and our robotic arm has only 6 DoF and not 7, making harder to find a valid joint trajectory.

Because our modelling treats the rotation as a random variable belonging to the n-sphere manifold, we also train a neural ratio estimator to predict the 4 DoF pose (we enforce a top-down approach) with $\mathbf{q} \in \mathbb{S}^1$ but we only use the $\mathbf{c}_{xy}(\mathbf{P})$ features plane. We train 5 ratios on 7500 different scenes $\mathbf{z}$ and 4000 tuples $(\mathbf{h}, S)$.

We evaluate the grasp success rate (GSR), the ratio of success grasp executions, and the declutter rate (DR), the average ratio of objects removed. Results are reported in Table I. To compute the MAP, we first take the 20 best candidates among 1800 hand configurations sampled from the prior. Then, we perform 100 optimization steps and keep the best candidate. If our method fails three times in row to generate a reachable hand configuration, we note it as a fail. Globally, our method performs similar to or better than other approaches. Our 4 DoF model performs better than the 6 DoF, mainly due to the lower dimensional space. While DR of GIGA and VGN are similar to their GSR, our DR is significantly lower than our GSR, meaning that our model mostly fails to pick the first object but once it is removed, it can successfully grasp all the objects. The discrepancy between the simulation and real-world setup is overcome without any decrease in performance. In real-world settings, the majority of failure cases are due to insufficient friction forces, causing the objects to slip. We believe that these failure scenario happen because the simulator used to generate the training set does not accurately model friction forces.

TABLE I: Grasp success rate and declutter rate for picking experiments for the packed scenario with 5 objects over 100 rounds. Real-world results are performed with 5 objects over 15 rounds. Results of GPD [12], VGN [9] and GIGA [10] are reported from [10].

Method	GSR ↑	DR ↑
Simulation results		
GPD [12]	35.4	30.7
VGN [9]	74.5	79.2
GIGA [10]	87.9	**86**
Ours (4DoF)	**91.1**	77
Ours (6DoF)	79.71	58.56
Real-world results		
VGN [9]	77.2	81.3
GIGA [10]	83.3	86.6
Ours (4DoF)	**95.6**	**88**
Ours (6DoF)	86.3	62.1

B. Sample efficiency

We generate 2000 different scenes $(\mathbf{z})$ and for each scene, we sample 2000 tuples $(\mathbf{h}, S)$ from the prior and the sequential posteriors, each one trained on 7500 different scenes $\mathbf{z}$ with 4000 tuples $(\mathbf{h}, S)$. We report the results in Table II. With no surprise, sampling for the sequential posteriors leads to an increasing GSR. The use of an occupancy network as prior conditionned by observation for the position increases the GSR by a factor 10. However, this rate is still far from an acceptable one, mainly due to the orientation prior, which is uniform. As shown in Fig 6, the distribution $p(S)$ shifts from a concentrated distribution near 0% (meaning that for one scene $\mathbf{z}$, the sampled hand configurations $\mathbf{h}$ will lead to a very low grasping success rate in expectation) and spreads toward higher grasping success rate areas,

meaning that each iteration of the sequential procedure generalizes better than the previous one.

TABLE II: GSR of sampling strategies.

Sampling strategy	GSR
Uniform prior	0.05
$p_0(\mathbf{h} \mid o, \mathbf{P})$	0.96
$p_1(\mathbf{h} \mid o, \mathbf{P})$	14
$p_2(\mathbf{h} \mid o, \mathbf{P})$	19.62

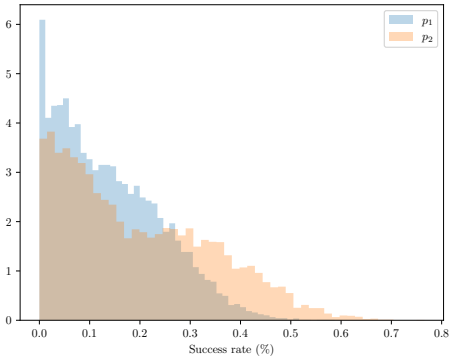


Fig. 6: Empirical distribution $p(S)$ evaluated on $10,000$ scenes $\mathbf{z}$. Iterations of the sequential procedure shifts the distribution from low grasping success rate regions to higher grasping success rate regions.

C. Posteriors

Our method has the advantage to offer access to the full posterior distribution over $\mathcal{H}$. Thus, with our geodesic Hamiltonian Monte Carlo scheme, we can sample hand configurations from the posterior $\mathbf{h} \sim p(\mathbf{h} \mid S = 1, o, \mathbf{P})$, as depicted in Fig 7. While our conditional prior distributes density across everywhere on the objects, the posterior assigns maximal density to the top of objects and minimal density to the bottom of objects when multiple objects are present on the table. This occurs due to potential collisions between the gripper and the table, or the gripper and other objects. In practice, our method will first pick the tallest object present on the table and then move to the next tallest object or to an isolated object (Fig 5). Regarding orientation, the posterior converges to a top-down grasping approach while having some deviations from a strict 4 DoF setup. In fact, top-down approaches seem optimal in our setup (small workspace size and large gripper) because it avoids many collisions between the gripper and the table or the other objects.

VI. Related Work

A. Robotic grasping

Probabilistic approaches for grasping problems typically rely on likelihood functions that model the probability of grasp success or a grasp quality metric given an observation and grasp pose. Various methods can be employed to determine the maximum likelihood estimate (MLE) corresponding to the final grasp pose. Numerical optimization techniques can be utilized when the likelihood is represented by differentiable models [14]. Direct regression of the MLE using a learned model yields quick output but fails to capture the complete distribution [15]. Other approaches determine the MLE based on a list of candidates computed through a grasp map in sensor space [9]. In a similar vein to our work, in [16], models for the likelihood and the prior are trained and the posterior density is optimized using gradient descent. However, unlike our approach, they employ Euler angles, which can be susceptible to gimbal lock and singularities. Our method preserves the geometry of the rotation space by performing Riemannian gradient descent.

B. Neural scene representation

Our work capitalizes on the latest advancements in neural implicit representations, which parameterize a 3D scene as a continuous function [17]–[19]. This representation has proven valuable in various domains, including 3D reconstruction. Implicit representations possess the advantage of offering 'infinite resolution' compared to discrete approaches. Furthermore, their rich latent space endows them with powerful feature extraction capabilities for diverse purposes. Additionally, they exhibit flexibility by being conditioned on different sensor inputs, such as point clouds or voxel grids, thus making them highly adaptable for robotic applications [10], [20], [21].

VII. Conclusion

We have shown that simulation-based Bayesian inference can be applied effectively to robotic grasping in complex and noisy environments. The proposed innovative method improves the sample efficiency of the inference pipeline. Our approach can manage tasks with escalating complexity and proves valuable for real-world robotic applications. Future research will focus on the real-time constraints for inferring the optimal grasp pose.

Acknowledgement

Norman Marlier acknowledges the Belgian Fund for Research training in Industry and Agriculture for its financial support (FRIA grant). Computational resources have been provided by the Consortium des Équipements de Calcul Intensif (CÉCI), funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under Grant No. 2.5020.11 and by the Walloon Region.

References

[1] K. Cranmer, J. Brehmer, and G. Louppe, "The frontier of simulation-based inference," *Proceedings of the National Academy of Sciences*, 2020.

[2] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, 2020, pp. 523–540.

[3] R. M. Neal *et al.*, "Mcmc using hamiltonian dynamics," *Handbook of markov chain monte carlo*, vol. 2, no. 11, p. 2, 2011.

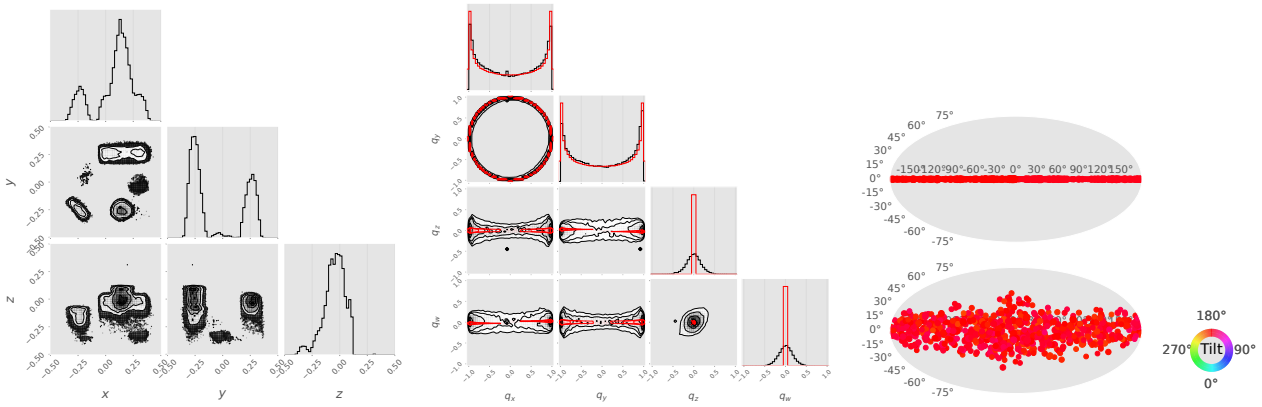


Fig. 7: Posterior distribution $\hat{p}(\mathbf{h} \mid S = 1, o = 1, \mathbf{P})$ of the grasp pose estimated by geodesic Hamiltonian Monte Carlo. The scene and thus the point cloud are the same as in Fig 3. (left) The marginal distribution of the position $\hat{p}(\mathbf{x} \mid S = 1, o = 1, \mathbf{P})$. (middle) The marginal distribution of the orientation $\hat{p}(\mathbf{q} \mid S = 1, o = 1, \mathbf{P})$ in black compared to a top-down approach in red by components of the quaternion. (right top) Distribution $p(\mathbf{q})$ along a top-down approach shown with a Mollweide projection as in [13]. (right bottom) The marginal distribution of the orientation $\hat{p}(\mathbf{q} \mid S = 1, o = 1, \mathbf{P})$.

[4] J. Hermans, V. Begy, and G. Louppe, "Likelihood-free MCMC with amortized approximate ratio estimators," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020, pp. 4239–4248. [Online]. Available: http://proceedings.mlr.press/v119/hermans20a.html

[5] J. Hermans, A. Delaunoy, F. Rozet, A. Wehenkel, V. Begy, and G. Louppe, "A crisis in simulation-based inference? beware, your posterior approximations can be unfaithful," *Transactions on Machine Learning Research*, 2022.

[6] S. Byrne and M. Girolami, "Geodesic monte carlo on embedded manifolds," *Scandinavian Journal of Statistics*, vol. 40, no. 4, pp. 825–845, 2013.

[7] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[8] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2020.

[9] M. Breyer, J. J. Chung, L. Ott, S. Roland, and N. Juan, "Volumetric grasping network: Real-time 6 dof grasp detection in clutter," in *Conference on Robot Learning*, 2020.

[10] Z. Jiang, Y. Zhu, M. Svetlik, K. Fang, and Y. Zhu, "Synergies between affordance and geometry: 6-dof grasp detection via implicit representations," *arXiv preprint arXiv:2104.01542*, 2021.

[11] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[12] A. Ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1455–1473, 2017.

[13] K. Murphy, C. Esteves, V. Jampani, S. Ramalingam, and A. Makadia, "Implicit-pdf: Non-parametric representation of probability distributions on the rotation manifold," *arXiv preprint arXiv:2106.05965*, 2021.

[14] Q. Lu, K. Chenna, B. Sundaralingam, and T. Hermans, "Planning multi-fingered grasps as probabilistic inference in a learned deep network," in *Robotics Research*. Springer, 2020, pp. 455–472.

[15] J. Cai, J. Cen, H. Wang, and M. Y. Wang, "Real-time collision-free grasp pose detection with geometry-aware refinement using high-resolution volume," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1888–1895, 2022.

[16] M. Van der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans, "Learning continuous 3d reconstructions for geometrically aware grasping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 11 516–11 522.

[17] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 165–174.

[18] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[19] A. Simeonov, Y. Du, A. Tagliasacchi, J. B. Tenenbaum, A. Rodriguez, P. Agrawal, and V. Sitzmann, "Neural descriptor fields: Se (3)-equivariant object representations for manipulation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6394–6400.

[20] K. Karunratanakul, J. Yang, Y. Zhang, M. J. Black, K. Muandet, and S. Tang, "Grasping field: Learning implicit representations for human grasps," in *2020 International Conference on 3D Vision (3DV)*. IEEE, 2020, pp. 333–344.

[21] D. Driess, J.-S. Ha, M. Toussaint, and R. Tedrake, "Learning models as functionals of signed-distance fields for manipulation planning," in *Conference on Robot Learning*. PMLR, 2022, pp. 245–255.

10.3 EPILOGUE

10.3.1 *Advantages*

Our sequential approach allows us to overcome the main issue arising in simulation-based inference for robotics grasping: very few grasp poses sampled from the prior lead to successful grasps. This sequential approach provides an automatic procedure to generate data that will converge to an approximate posterior useful for robotics tasks. All the previous chapters show that the prior plays an important role and it can be difficult to handcraft a relevant prior to generate successful grasp poses. With the sequential approach, even an inefficient first prior leads to an approximate posterior which reaches a high success rate in practice.

10.3.2 *Limitations*

While our sequential approach showed significant performance on real benchmarks for full DoF grasp pose estimation, it has some limitations and drawbacks. First of all, the computation time is prohibitive, making it even slower than the 4 DoF. Furthermore, it requires more memory because of the dimension of the grasp pose. Another limitation concerns the data generation. While Chapters 7 to 9 generated only one dataset, the sequential approach requires generating new samples for each iteration of the sequential procedure. Because sampling from the manifold is quite slow, the whole procedure is very time-consuming. Finally, the sequential approach shares the same limitations as the previous work. We keep the assumption that the grasping contact point lies inside the object which makes the approximation irrelevant for some robotics tasks.

10.3.3 *Conclusion and opportunities*

Our sequential approach to approximate the posterior reached a high success rate, about 85%, for the full grasp pose. It allows us to combine the high degree of freedom from Chapter 8 and the high complexity of the task from Chapter 9. We can still improve the framework by decreasing the computation time. It can be possible to train a model to directly infer the maximum a posteriori by using the approximate posterior as a loss function. Thus, only one forward pass will be needed which reduces the inference to less than one second. We leave this for future work.

Part IV

CONCLUSION

11

CONCLUSION

Robotic grasping is a challenging task due to its intrinsic complexity: the wide variety of object shapes, the nonsmooth and nonlinear contact mechanics, and the noise inherent in the perception of the environment. Furthermore, the complexity comes from the robot itself: the nonlinear mapping between configuration space and task space alongside the high dimensional multi-DoF gripper. All these ingredients make robotic grasping a perfect benchmark to exploit the power of simulation-based inference algorithms. This thesis has demonstrated their usefulness and how to use them in practice.

In Part i, we motivated the use of probabilistic modeling to infer grasp poses and we took into account the topology of the task space through manifold optimization. We introduced notions of robotic trajectory planning in Chapter 2 and how to relate a robotic task to an effective motion. Then, Chapter 3 discussed probabilistic modeling and why it is useful for robotics applications. We showed how to use machine learning and neural networks to perform approximate Bayesian posterior inference. Finally, in Chapter 4 we introduced manifold optimization which links probabilistic modeling and the topology of the robotic task space. This completed our framework for robotic grasping.

In Part ii, we provided a review of existing approaches for robotic grasping. We first started with the approaches based on mechanical principles, called analytical approaches. We showed why they are rarely used in practice because they do not deal with uncertainties and require perfect knowledge of their environment, which is almost impossible to achieve. Then, roboticists started to use more and more data-driven methods to overcome the principal issue of analytical approaches. However, they continue to use quality metrics from mechanical principles to assess the success or failure of grasps. During this period, deep learning has appeared and shown incredible performance on complex tasks. It became quickly the new state-of-the-art approach for robotic grasping. From this time, novel deep learning methods found successful applications in the field of robotics.

Part iii explained all the contributions of the thesis. In Chapter 7, we started from a simple robotic benchmark, which is the grasping of a single object with very few non-observable variables. This prototype demonstrated the interest of simulation-based inference methods for robotics. Then, in Chapter 8, we increased the complexity of the task and overcame previous limitations by improving the prior and slightly incorporating new properties in it. We moved a step further in Chapter 9 in the complexity of the task by learning priors from observation while reducing the dimension of the search space. These contributions unlock new possibilities used in Chapter 10, providing the full benchmark without constraints on the DoF of the gripper.

The use of probabilistic modeling in robotics is a central element in the evolution of robotics. Over four decades of research in robotic grasping have led roboticists from a deterministic approach to a stochastic one. Sensors always produce noisy observations of the environment, and dynamics is often stochastic, thereby complicating the decision-making process. Although probabilistic modeling found its initial application in localization and navigation tasks with linear approximations, the deep learning revolution has provided many tools to accomplish complex nonlinear tasks. Robotic grasping is no longer a task requiring complex models with many equations to solve but rather methods that exploit at best outcomes from experience to make decisions.

We bring one solution to the research question: **How can robots autonomously grasp objects in unstructured environments?** *By performing Bayesian posterior inference with the posterior surrogate learned from stochastic simulations.* While many different possible solutions exist, our framework based on simulation-based inference methods nicely handles the issues faced in the research question.

Our contributions offer original solutions to several challenges by bringing simulation-based inference methods to robotic grasping. These methods offer numerous advantages. Firstly, they allow us to perform Bayesian inference despite the intractability of both the likelihood function and the marginal. Secondly, they leverage robotic simulators to generate samples. They have gained widespread adoption due to extensive use in research, thereby boosting simulation-based inference methods by bridging the gap between simulations and the real world. Thirdly, they provide amortized and differentiable approximations, enabling to generalize well across a wide variety of situations encountered in robotic grasping without retraining for each new observation. Subsequently, we introduce appropriate modeling of the task to perform Bayesian inference. This step is crucial as it involves the design of a prior for the variables of interest, addressing the primary challenge encountered throughout our contributions. Each contribution overcomes a more complex task by using a more sophisticated prior. Finally, we model the variables of interest by considering their geometry and enhancing the modeling with greater generalization capabilities. The integration of learning and geometrical methods unlocks novel possibilities for robotic applications.

Nevertheless, simulation-based inference methods do not solve all the problems faced in robotics. First of all, simulation-based inference methods rely on simulations to learn a surrogate of the posterior. However, robotics simulators may lack precision because of the rich nonsmooth dynamics of contact. Thus, they can only simulate basic scenarios. This discrepancy between the simulation and the real world may introduce some errors that impact the success of the method. Secondly, amortized methods require a large amount of data that can be very time-consuming, depending on the task, to generate a valid dataset. These two issues may be solved by learning the posterior surrogate from a small bunch of real demonstrations, made by an expert. Given that these demonstrations take place in real-world settings, the surrogate posterior will closely approximate the actual posterior distribution rather than its simulated counterpart. In addition, the expert efficiently

explores the state space, thus providing very informative demonstrations. Lastly, the inference time of our method can be long for robotic applications, due to the many optimization steps we performed. Future solutions may come from hardware acceleration but most likely from new methods discovered in machine learning, such as knowledge distillation, that hold promise for quickly inferring commands transmitted to the robot.

Machine learning has changed many aspects of robotics. It has unlocked so many possibilities that were unimaginable ten years ago. But as it is said in the robotic community, robot learning is more about how a robot learns rather than applying naive machine learning methods. This dissertation, along with its contributions, illustrates the multidisciplinary nature of robot learning, demonstrating the diverse array of tools necessary for its efficacy.

The robotic revolution is underway, driven by progress in learning methods.

Part V

APPENDIX

A

REFERENCES

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

Ü. R. Aktaş, C. Zhao, M. Kopicki, and J. L. Wyatt. Deep dexterous grasping of novel objects from a single view. *International Journal of Humanoid Robotics*, 19(02):2250011, 2022.

A. Aristidou, J. Lasenby, Y. Chrysanthou, and A. Shamir. Inverse kinematics techniques in computer graphics: A survey. In *Computer graphics forum*, volume 37, pages 35–58. Wiley Online Library, 2018.

R. Balasubramanian, L. Xu, P. D. Brook, J. R. Smith, and Y. Matsuoka. Physical human interactive guidance: Identifying grasping principles from human-planned grasps. *IEEE Transactions on Robotics*, 28(4):899–910, 2012.

L. Berscheid, C. Friedrich, and T. Kröger. Robot learning of 6 dof grasping using model-based adaptive primitives. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4474–4480. IEEE, 2021.

A. Bicchi and V. Kumar. Robotic grasping and contact: A review. In *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, volume 1, pages 348–353. IEEE, 2000.

C. Bingham. An antipodally symmetric distribution on the sphere. *The Annals of Statistics*, pages 1201–1225, 1974.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

J. Bohg, A. Morales, T. Asfour, and D. Kragic. Data-driven grasp synthesis - a survey. *IEEE Transactions on robotics*, 30(2):289–309, 2013.

N. Boumal. Interpolation and regression of rotation matrices. In *International Conference on Geometric Science of Information*, pages 345–352. Springer, 2013.

M. Breyer, J. J. Chung, L. Ott, R. Siegwart, and J. Nieto. Volumetric grasping network: Real-time 6 dof grasp detection in clutter. In *Conference on Robot Learning*, pages 1602–1611. PMLR, 2021.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

J. Canny and C. Ferrari. Planning optimal grasps. In *Proc. Conf. on Robotics and Automation (ICRA)*, volume 1992, pages 2290–2295, 1992.

E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2023.

K. Cranmer, J. Pavez, and G. Louppe. Approximating likelihood ratios with calibrated discriminative classifiers. *arXiv preprint arXiv:1506.02169*, 2015.

K. Cranmer, J. Brehmer, and G. Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020.

B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

M. Dax, S. Green, J. Gair, M. Deistler, B. Schölkopf, and J. H. Macke. Group equivariant neural posterior estimation. In *ICLR 2022*, 2022.

J. P. C. de Souza, L. F. Rocha, P. M. Oliveira, A. P. Moreira, and J. Boaventura-Cunha. Robotic grasping: from wrench space heuristics to deep learning policies. *Robotics and Computer-Integrated Manufacturing*, 71:102176, 2021. ISSN 0736-5845. doi: https://doi.org/10.1016/j.rcim.2021.102176. URL https://www.sciencedirect.com/science/article/pii/S0736584521000594.

J. Denavit and R. S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. 1955.

R. Detry, O. Kroemer, M. Popovic, Y. Touati, E. Baseski, N. Krueger, J. Peters, and J. Piater. Object-specific grasp affordance densities. In *Proceedings of ICDL*, volume 10, 2009.

R. Detry, J. Papon, and L. Matthies. Task-oriented grasping with semantic and geometric scene understanding. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3266–3273. IEEE, 2017.

R. Diankov. *Automated construction of robotic manipulation programs*. PhD thesis, Carnegie Mellon University, The Robotics Institute Pittsburgh, 2010.

C. C. Drovandi and A. N. Pettitt. Likelihood-free bayesian estimation of multivariate quantile distributions. *Computational Statistics & Data Analysis*, 55(9):2541–2556, 2011.

C. Durkan, I. Murray, and G. Papamakarios. On contrastive learning for likelihood-free inference. In *International conference on machine learning*, pages 2771–2781. PMLR, 2020.

S. El-Khoury, A. Sahbani, and V. Perdereau. Learning the natural grasping component of an unknown object. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2957–2962. IEEE, 2007.

C. Eppner, A. Mousavian, and D. Fox. A billion ways to grasp: An evaluation of grasp sampling schemes on a dense, physics-based grasp data set. In *The International Symposium of Robotics Research*, pages 890–905. Springer, 2019.

R. G. Everitt. Bootstrapped synthetic likelihood. *arXiv preprint arXiv:1711.05825*, 2017.

H.-S. Fang, C. Wang, M. Gou, and C. Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11444–11453, 2020.

M. Fasiolo, S. N. Wood, F. Hartig, and M. V. Bravington. An extended empirical saddlepoint approximation for intractable likelihoods. 2018.

K.-I. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural networks*, 2(3):183–192, 1989.

P. Furgale. Representing robot pose: The good, the bad, and the ugly. In *workshop on Lessons Learned from Building Complex Systems, IEEE International Conference on Robotics and Automation (ICRA). http://paulfurgale. info/news/2*, volume 14, page 9, 2014.

C. J. Geyer. Practical markov chain monte carlo. *Statistical science*, pages 473–483, 1992.

W. R. Gilks, S. Richardson, and D. Spiegelhalter. *Markov chain Monte Carlo in practice*. CRC press, 1995.

M. Grübler. *Allgemeine Eigenschaften der zwangläufigen ebenen kinematischen Ketten*. L. Simion, 1884.

W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

J. Hermans, V. Begy, and G. Louppe. Likelihood-free mcmc with amortized approximate ratio estimators. In *International conference on machine learning*, pages 4239–4248. PMLR, 2020.

J. Hermans, N. Banik, C. Weniger, G. Bertone, and G. Louppe. Towards constraining warm dark matter with stellar streams through neural simulation-based inference. *Monthly Notices of the Royal Astronomical Society*, 507(2):1999–2011, 2021.

W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR, 2022.

W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. In *Conference on Robot Learning*, pages 1769–1782. PMLR, 2023.

b. Ichter, A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, D. Kalashnikov, S. Levine, Y. Lu, C. Parada, K. Rao, P. Sermanet, A. T. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, M. Yan, N. Brown, M. Ahn, O. Cortes, N. Sievers, C. Tan, S. Xu, D. Reyes, J. Rettinghouse, J. Quiambao, P. Pastor, L. Luu, K.-H. Lee, Y. Kuang, S. Jesmonth, N. J. Joshi, K. Jeffrey, R. J. Ruano, J. Hsu, K. Gopalakrishnan, B. David, A. Zeng, and C. K. Fu. Do as i can, not as i say: Grounding language in robotic affordances. In K. Liu, D. Kulic, and J. Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 287–318. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/ichter23a.html.

N. Jaquier, L. Rozo, S. Calinon, and M. Bürger. Bayesian optimization meets riemannian manifolds in robot learning. In *Conference on Robot Learning*, pages 233–246. PMLR, 2020.

Y. Jiang, S. Moseson, and A. Saxena. Efficient grasping from rgbd images: Learning using a new rectangle representation. In *2011 IEEE International conference on robotics and automation*, pages 3304–3311. IEEE, 2011.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37:183–233, 1999.

D. Kappler, J. Bohg, and S. Schaal. Leveraging big data for grasp planning. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 4304–4311. IEEE, 2015.

O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1):90–98, 1986.

H. Klein, N. Jaquier, A. Meixner, and T. Asfour. On the design of region-avoiding metrics for collision-safe motion generation on riemannian manifolds. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2346–2353. IEEE, 2023.

I. Kobyzev, S. J. Prince, and M. A. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020.

D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

F. Kyota, T. Watabe, S. Saito, and M. Nakajima. Detection and evaluation of grasping positions for autonomous agents. In *2005 International Conference on Cyberworlds (CW'05)*, pages 8–pp. IEEE, 2005.

S. LaValle. Rapidly-exploring random trees: A new tool for path planning. *Research Report 9811*, 1998.

S. M. LaValle, J. J. Kuffner, B. Donald, et al. Rapidly-exploring random trees: Progress and prospects. *Algorithmic and computational robotics: new directions*, 5:293–308, 2001.

Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

N. P. Lemoine. Moving beyond noninformative priors: why and how to choose weakly informative priors in bayesian analyses. *Oikos*, 128(7):912–928, 2019.

J. Liang, V. Makoviychuk, A. Handa, N. Chentanez, M. Macklin, and D. Fox. Gpu-accelerated robotic simulation for distributed reinforcement learning. In *Conference on Robot Learning*, pages 270–282. PMLR, 2018.

M. Likhachev, G. J. Gordon, and S. Thrun. Ara*: Anytime a* with provable bounds on sub-optimality. *Advances in neural information processing systems*, 16, 2003.

M. Liu, Z. Pan, K. Xu, K. Ganguly, and D. Manocha. Generating grasp poses for a high-dof gripper using neural networks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1518–1525. IEEE, 2019.

Q. Lu, M. Van der Merwe, B. Sundaralingam, and T. Hermans. Multifingered grasp planning via inference in deep neural networks: Outperforming sampling by learning differentiable models. *IEEE Robotics & Automation Magazine*, 27(2):55–65, 2020.

K. M. Lynch and F. C. Park. *Modern robotics*. Cambridge University Press, 2017.

J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. 2017.

N. Marlier, G. Louppe, O. Bruls, and G. Dislaire. Robotic throwing controller for accelerating a recycling line. In *Proceedings of the Robotix Academy Conference for Industrial Robotics (RACIR) 2019*. F.R.S.-FNRS - Fonds de la Recherche Scientifique [BE], Robotix Academy, 2019.

N. Marlier, O. Brüls, and G. Louppe. Simulation-based bayesian inference for multi-fingered robotic grasping. *arXiv preprint arXiv:2109.14275*, 2021.

N. Marlier, O. Brüls, and G. Louppe. Simulation-based bayesian inference for robotic grasping. *arXiv preprint arXiv:2303.05873*, 2022.

N. Marlier, J. Gustin, G. Louppe, and O. Brüls. Implicit representation priors meet riemannian geometry for bayesian robotic grasping. *arXiv preprint arXiv:2304.08805*, 2023.

H. Merzić, M. Bogdanović, D. Kappler, L. Righetti, and J. Bohg. Leveraging contact forces for learning to grasp. In *2019 international conference on robotics and automation (ICRA)*, pages 3615–3621. IEEE, 2019.

C. Michel, V. Perdereau, and M. Drouin. An approach to extract natural grasping axes with a real 3d vision system. In *2006 IEEE International Symposium on Industrial Electronics*, volume 4, pages 3130–3135. IEEE, 2006.

A. T. Miller and P. K. Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004.

A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen. Automatic grasp planning using shape primitives. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 2, pages 1824–1829. IEEE, 2003.

B. K. Miller, A. Cole, P. Forré, G. Louppe, and C. Weniger. Truncated marginal neural ratio estimation. *Advances in Neural Information Processing Systems*, 34:129–143, 2021.

B. K. Miller, C. Weniger, and P. Forré. Contrastive neural ratio estimation. *Advances in Neural Information Processing Systems*, 35:3262–3278, 2022.

L. Min, P. Zherong, X. Kai, G. Kanishka, and M. Dinesh. Deep differentiable grasp planner for high-dof grippers. *CoRR*, abs/2002.01530, 2020. URL https://arxiv.org/abs/2002.01530.

M. Q. Mohammed, K. L. Chung, and C. S. Chyi. Review of deep reinforcement learning-based object grasping: Techniques, open challenges, and recommendations. *IEEE Access*, 8:178450–178481, 2020.

L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor. Learning object affordances: from sensory–motor coordination to imitation. *IEEE Transactions on Robotics*, 24(1):15–26, 2008.

A. Morales, E. Chinellato, A. H. Fagg, and A. P. Del Pobil. Using experience for assessing grasp reliability. *International Journal of Humanoid Robotics*, 1(04):671–691, 2004.

A. Mousavian, C. Eppner, and D. Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2901–2910, 2019.

K. A. Murphy, C. Esteves, V. Jampani, S. Ramalingam, and A. Makadia. Implicit-pdf: Non-parametric representation of probability distributions on the rotation manifold. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 7882–7893. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/murphy21a.html.

R. M. Murray, Z. Li, S. S. Sastry, and S. S. Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

R. M. Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

R. Newbury, M. Gu, L. Chumbley, A. Mousavian, C. Eppner, J. Leitner, J. Bohg, A. Morales, T. Asfour, D. Kragic, et al. Deep learning approaches to grasp synthesis: A review. *IEEE Transactions on Robotics*, 2023.

V. M. Ong, D. J. Nott, M.-N. Tran, S. A. Sisson, and C. C. Drovandi. Variational bayes with synthetic likelihood. *Statistics and Computing*, 28:971–988, 2018.

G. Papamakarios and I. Murray. Fast ε-free inference of simulation models with bayesian conditional density estimation. *Advances in neural information processing systems*, 29, 2016.

G. Papamakarios, D. Sterratt, and I. Murray. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 837–848. PMLR, 2019.

G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *The Journal of Machine Learning Research*, 22(1):2617–2680, 2021.

R. Pelossof, A. Miller, P. Allen, and T. Jebara. An svm learning approach to robotic grasping. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 4, pages 3512–3518. IEEE, 2004.

G. W. Peters, S. A. Sisson, and Y. Fan. Likelihood-free bayesian inference for α-stable models. *Computational Statistics & Data Analysis*, 56(11):3743–3756, 2012.

D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

M. A. Roa and R. Suárez. Computation of independent contact regions for grasping 3-d objects. *IEEE Transactions on Robotics*, 25(4):839–850, 2009.

M. A. Roa and R. Suárez. Grasp quality measures: review and performance. *Autonomous robots*, 38:65–88, 2015.

A. Sarma and M. Kay. Prior setting in practice: Strategies and rationales used in choosing prior distributions for bayesian analysis. In *Proceedings of the 2020 chi conference on human factors in computing systems*, pages 1–12, 2020.

M. Saveriano, F. J. Abu-Dakka, and V. Kyrki. Learning stable robotic skills on riemannian manifolds. *Robotics and Autonomous Systems*, 169:104510, 2023.

P. Schmidt, N. Vahrenkamp, M. Wächter, and T. Asfour. Grasping of unknown objects using deep convolutional neural networks based on depth images. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 6831–6838. IEEE, 2018.

B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2015.

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

I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023.

S. A. Sisson, Y. Fan, and M. Beaumont. *Handbook of approximate Bayesian computation*. CRC Press, 2018.

G. Smith, E. Lee, K. Goldberg, K. Bohringer, and J. Craig. Computing parallel-jaw grips. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, volume 3, pages 1897–1903. IEEE, 1999.

S. Song, A. Zeng, J. Lee, and T. Funkhouser. Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. *IEEE Robotics and Automation Letters*, 5(3):4978–4985, 2020.

I. A. Şucan, M. Moll, and L. E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012. doi: 10.1109/MRA.2012.2205651. https://ompl.kavrakilab.org.

B. Tang, M. Corsaro, G. Konidaris, S. Nikolaidis, and S. Tellex. Learning collaborative pushing and grasping policies in dense clutter. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6177–6184. IEEE, 2021.

A. Ten Pas and R. Platt. Using geometry to detect grasp poses in 3d point clouds. *Robotics Research: Volume 1*, pages 307–324, 2018.

O. Thomas, R. Dutta, J. Corander, S. Kaski, and M. U. Gutmann. Likelihood-free inference by ratio estimation. *Bayesian Analysis*, 17(1):1–31, 2022.

E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

D. Tran, R. Ranganath, and D. Blei. Hierarchical implicit models and likelihood-free variational inference. *Advances in Neural Information Processing Systems*, 30, 2017.

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. In *Intelligent Autonomous Systems 12: Volume 1 Proceedings of the 12th International Conference IAS-12, held June 26-29, 2012, Jeju Island, Korea*, pages 585–594. Springer, 2013.

J. Varley, J. Weisz, J. Weiss, and P. Allen. Generating multi-fingered robotic grasps via deep learning. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 4415–4420. IEEE, 2015.

M. Veres, M. Moussa, and G. W. Taylor. An integrated simulator and dataset that combines grasping and vision for deep learning. *arXiv preprint arXiv:1702.02103*, 2017.

M. J. Wainwright, M. I. Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

C. Wang, H.-S. Fang, M. Gou, H. Fang, J. Gao, and C. Lu. Graspness discovery in clutters for fast and accurate grasp detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15964–15973, 2021.

A. Wehenkel and G. Louppe. Unconstrained monotonic neural networks. *Advances in neural information processing systems*, 32, 2019.

W. Wei, Y. Luo, F. Li, G. Xu, J. Zhong, W. Li, and P. Wang. Gpr: Grasp pose refinement network for cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4295–4302. IEEE, 2021.

J. Weisz and P. K. Allen. Pose error robust grasping from contact wrench space metrics. In *2012 IEEE international conference on robotics and automation*, pages 557–562. IEEE, 2012.

B. Wu, I. Akinola, and P. K. Allen. Pixel-attentive policy gradient for multi-fingered grasping in cluttered scenes. In *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 1789–1796. IEEE, 2019.

B. Wu, I. Akinola, A. Gupta, F. Xu, J. Varley, D. Watkins-Valls, and P. K. Allen. Generative attention learning: A "general" framework for high-performance multi-fingered grasping in clutter. *Autonomous Robots*, 44(6):971–990, 2020.

D. Yang, T. Tosun, B. Eisner, V. Isler, and D. Lee. Robotic grasping through combined image-based grasp proposal and 3d reconstruction. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6350–6356. IEEE, 2021.

A. Zeng, M. Attarian, B. Ichter, K. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. Ryoo, V. Sindhwani, et al. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

Z. Zhang. Microsoft kinect sensor and its effect. *IEEE multimedia*, 19(2):4–10, 2012.

B. Zhao, H. Zhang, X. Lan, H. Wang, Z. Tian, and N. Zheng. Regnet: Region-based grasp network for end-to-end grasp detection in point clouds. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13474–13480. IEEE, 2021.

W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

Y. Zhou and K. Hauser. 6dof grasp planning by optimizing a deep learning scoring function. In *Robotics: Science and systems (RSS) workshop on revisiting contact-turning a problem into a solution*, volume 2, page 6, 2017.

X. Zhu, L. Sun, Y. Fan, and M. Tomizuka. 6-dof contrastive grasp proposal network. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6371–6377. IEEE, 2021.