TÉCNICO LISBOA

UNIVERSIDADE DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

Conditional Postural Synergies for Dexterous Grasps Using Variational Autoencoders

Dimitrios Dimou

Supervisor	**Doctor** Plinio Moreno
Co-Supervisor	**Doctor** José Alberto Rosado dos Santos-Victor

Thesis approved in public session to obtain the PhD Degree in Electrical and Computer Engineering

Jury final classification: Pass with Distinction

2025

TÉCNICO LISBOA

UNIVERSIDADE DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

Conditional Postural Synergies for Dexterous Grasps Using Variational Autoencoders

Dimitrios Dimou

Supervisor	**Doctor** Plinio Moreno
Co-Supervisor	**Doctor** José Alberto Rosado dos Santos-Victor

Thesis approved in public session to obtain the PhD Degree in Electrical and Computer Engineering

Jury final classification: Pass with Distinction

Jury

Chairperson: Doctor Pedro Manuel Urbano de Almeida Lima, Instituto Superior Técnico, Universidade de Lisboa

Members of the Committee:

Doctor Lino José Forte Marques, Faculdade de Ciências e Tecnologia, Universidade de Coimbra

Doctor Francisco António Chaves Saraiva de Melo, Instituto Superior Técnico, Universidade de Lisboa

Doctor Alexandre José Malheiro Bernardino, Instituto Superior Técnico, Universidade de Lisboa

Doctor Norman Hendrich, Faculty of Mathematics, Informatics and Natural Science, University of Hamburg, Alemanha

Doctor Plinio Moreno López, Instituto Superior Técnico, Universidade de Lisboa

Funding Institution: Fundação para a Ciência e a Tecnologia

2025

Acknowledgments

I would like to begin by thanking my supervisors: Dr. Plinio Moreno and Prof. José Santos-Victor, for their invaluable guidance and support throughout my PhD journey. Their dedication, countless hours spent discussing research ideas, and meticulous feedback on papers have been instrumental to my growth. I am equally grateful to everyone at VisLab for creating such a joyful and inspiring environment during my time there.

I would also like to express my gratitude to my life partner, Konstantina, for her support through all the highs and lows over the years. Her positivity, encouragement, and unwavering belief in me have been a constant source of strength, inspiring me and helping me grow into a better person.

Lastly, I dedicate this work to my mother, whose immense effort, courage, and unwavering support have been the foundation of my journey. She has always been by my side, guiding and inspiring me every step of the way.

This work was funded by a Fundação para a Ciência e a Tecnologia (FCT) doctoral grant [PD/BD/09714/2020].

Abstract

Using our hands we can easily perform precise actions to manipulate and interact with our environment. Our hands are a primary component of our everyday life interactions. Humanoid robots that will collaborate with humans, outside of industrial environments, will require the same level of dexterity and ability to manipulate their surroundings. However, efficiently controlling multi-fingered robotic hands remains a major challenge in robotics.

In this thesis, we investigate the problem of developing computational models, based on recent machine learning methods, to facilitate the control of humanoid hands. We present an approach to control that significantly reduces the required degrees of freedom of the hand. Our approach is inspired by the mechanism that the human brain employs to control the hand, namely using postural synergies. This mechanism couples the degrees of freedom of each finger and effectively controls the hand using only a small number of parameters.

More specifically, we introduce a generative model, based on the Variational Autoencoder framework, that encodes and decodes robotic hand postures using a compact set of variables. This framework abstracts the low-level control of the robotic hand into high-level postural synergies, allowing for more efficient representation and synthesis of grasp postures based on task-specific inputs such as grasp type and object size. By reducing the number of control parameters, our model enables smoother search spaces and improved grasp planning, which we validate by applying it to an in-hand regrasping task.

We further extend our model by integrating tactile feedback through a force controller that utilizes fingertip sensors to regulate grasp forces dynamically. This enables the robotic hand to adapt its grip based on real-time contact information, allowing it to autonomously lift, hold, and release objects during manipulation tasks. By incorporating tactile sensing, our approach improves grasp stability and enhances the robot's ability to interact with objects in unstructured environments.

Additionally, we develop a factorized grasp sampling model that jointly considers postural synergies and the 6DoF pose of the robotic hand. By leveraging this model, we can efficiently sample stable and precise grasps for various objects, particularly for precision grasp types. This approach improves grasp diversity and adaptability, enabling robotic hands to generate feasible grasp configurations across different object shapes and orientations.

Finally, we develop a reinforcement learning-based method that incorporates task-related information into the grasping behavior of a robotic agent. By leveraging insights from human grasping demonstrations and task constraints, our approach enables the robot to adapt its grasping strategy based on the specific requirements of a given task. This enhances the collaborative capabilities of dexterous robots, allowing them to perform task-efficient grasps in complex environments.

Keywords: Robotic Grasping, Robotic Manipulation, Postural Synergies, Deep Learning, Humanoid Robots

Resumo

Usando as nossas mãos podemos facilmente realizar ações precisas para manipular e interagir no ambiente. As nossas mãos são um componente principal das nossas interações na vida quotidiana. Num futuro próximo, os robôs humanoides poderão colaborar com pessoas fora dos ambientes industriais, caso consigam ter um nível avançado de destreza para manusear objetos. No entanto, controlar eficientemente mãos robóticas com vários dedos continua a ser um grande desafio na robótica.

Nesta tese, investigamos o problema do desenvolvimento de modelos computacionais, baseados em métodos recentes de aprendizagem profunda, para facilitar o controlo de mãos humanóides. Apresentamos uma abordagem de controlo que reduz significativamente os graus de liberdade. A nossa abordagem é inspirada no mecanismo que o cérebro humano utiliza para controlar a mão, nomeadamente através de sinergias posturais. Este mecanismo acopla os graus de liberdade de várias junta se controla eficazmente a mão utilizando apenas um número reduzido de parâmetros.

Mais concretamente, introduzimos um modelo generativo, baseado na estrutura do *Variational Autoencoder*, que codifica e descodifica posturas de mãos robóticas utilizando um conjunto compacto de variáveis. Esta estrutura abstrai o controlo de baixo nível da mão robótica em sinergias posturais de alto nível, alcançando uma representação e síntese mais eficiente das posturas de preensão, com base em informações específicas da tarefa, como o tipo de preensão e tamanho do objeto. Ao reduzir o número de parâmetros de controlo, o nosso modelo obtém espaços de pesquisa mais suaves e melhor planeamento da preensão, que validámos aplicando-o a uma tarefa de preensão manual.

Estendemos ainda mais o nosso modelo integrando o feedback tátil através de um controlador de força que utiliza sensores nas pontas dos dedos para regular dinamicamente as forças de preensão. Isto permite que a mão robótica adapte a sua pega com base em informações de contacto em tempo real, permitindo-lhe levantar, segurar e largar objetos de forma autónoma durante tarefas de manipulação. Ao incorporar a deteção tátil, a nossa abordagem melhora a estabilidade da preensão e aumenta a capacidade do robô interagir com objetos em ambientes não estruturados.

Além disso, desenvolvemos um modelo de amostragem de preensão sequêncial que considera conjuntamente as sinergias posturais e a pose 6DoF da mão robótica. Ao aproveitar este modelo, podemos amostrar eficientemente dados estáveis e pegadas precisas para vários objetos, particularmente para tipos de pegada de precisão. Esta abordagem melhora a compreensão diversidade e adaptabilidade, permitindo que as mãos robóticas gerem configurações de preensão viáveis em diferentes formas e orientações dos objetos.

Por fim, desenvolvemos um método baseado na aprendizagem por reforço que incorpora informação relacionada com a tarefa no comportamento de preensão de um agente robótico. Aproveitando as informações dos pontos de contacto e pose da mão relativamente ao objeto durante a manipulação de objetos por humanos para várias tarefas, a nossa abordagem permite que o robô adapte a sua estratégia de preensão com base nas requisitos de uma determinada tarefa. Isto melhora as capacidades de colaboração de robôs colaborativos, permitindo para que realizem tarefas de preensão de forma eficiente em ambientes complexos.

Palavras-chave: *Robotic Grasping*, Manipulação Robótica, Sinergias Posturais, Aprendizagem Profunda, Robôs Humanoides

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The subject matter of this thesis is to investigate the development of computational methods that facilitate the control of multi-fingered robotic hands. More specifically, we seek to imitate the mechanisms that the human brain relies on to control the hand and which enable us to perform dexterous and precise manipulation skills. In this introductory section, we first discuss the context and scope of our research endeavour, we then present our research's objectives and the assumptions behind it, and finally we highlight our contributions.

1.1 Motivation

The human hand evolved over millions of years in a way that constitutes humans the most advanced species on the planet [1]. Hands are the primary apparatus of our interaction with our environment. As infants, we use our hands to explore our environment and the capabilities that they give us by trying to grasp everything put in front of us. As adults, our hands enable us to manipulate our environment in very precise ways, to use tools, and to communicate. Our hands are considered a vehicle for intelligence and the advanced tool use that allow is the foundation of our civilization.

Sensorimotor neuroscientists studying human physiology have made several discoveries on how the brain controls the hand. The framework of postural synergies is the main hypothesis on how the human brain manages to control so efficiently so many different Degrees of Freedom (DoF) [2]. The synergistic hypothesis postulates that couplings in different levels of physiological organization such as neural, muscular, and kinematic facilitate the control of the hand for the brain. These couplings may be defined differently at various levels, but the main idea behind them is that a large number of DoF is controlled by a small number of control variables. This lower control space is called synergy space.

Since the human hand is arguably one of the most critical mechanisms of human physiology, modern robots will need to be equipped with humanoid multi-fingered hands in order to harmoniously co-exist and collaborate with humans outside of industrial environments, where everything is optimized to be handled by humans, such as households [3]. Most everyday objects and spaces are based on a human-centered design. Consequently, we will need to equip our robots with human-like hands and develop software that

allows the robotic hands to demonstrate similar capabilities to humans'. Humanoid robotic hands usually require many DoF in to order to approach the dexterity of human hands. Efficiently controlling such a complex and high DoF system is challenging. Thus roboticists have tried to imitate the organizational structure of human physiology to facilitate the control process.

In this thesis, inspired by this organizational format, we design computational methods that emulate it in order to facilitate the control of robotic multi-fingered hands. We aim to investigate the effectiveness of several computational methods to represent the synergy space and using a generative approach we demonstrate how to efficiently generate new hand configurations. Based on this approach, we develop methods to perform dexterous manipulation tasks using tactile feedback and generate grasp poses for grasping objects. Finally, we integrate the synergy framework with a reinforcement learning (RL) agent to grasp objects based on task-related information.

1.2 Background

The robotics community has developed two main approaches to imitate the organizational structure of the brain with anthropomorphic robotic hands: 1) the mechanical approach, where the synergies are implemented directly on the hardware of the robot, e.g. coupling tendons in the robot's fingers, this approach is more common in soft robotic hands that are hard to have a large number of DoF, and 2) the software approach in which the synergistic framework is implemented using a computational method [4]. In this thesis, we focus on the software approach as it is more modular, it is not invasive, and allows the robot to maintain a large number of degrees of freedom.

This approach is usually implemented using a dimensionality reduction method that extracts a low dimensional representation of the grasp configuration space. The robotic grasp postures can then be encoded in the synergy space as well as points from the synergy space can be decoded into new grasp postures (see Figure 1.1). The lower dimensional representation, i.e. the synergy space, can be used directly in subsequent tasks, for example for more efficient planning in sampling based motion planning methods. This way we avoid having to control independently each joint of each finger and we rely on the synergistic components that are extracted autonomously from the computational method. Dimensionality reduction methods is usually trained on a number of pre-recorded example grasp postures. For example, a human operator can teleoperate a robotic hand to execute successful grasps and record the final grasp postures, i.e. the robotic joint angles, or record entire trajectories of the robotic hand.

1.3 Problem definition and Challenges

Although the extraction of postural synergies has been extensively studied in the past, we identified several shortcomings and opportunities to enhance the framework of synergistic control. We begin with how the different representations for postural synergies are evaluated and what criteria are used for their comparison. For example, in most previous works the reconstruction error was used as an evaluation metric, we challenge this convention and propose a more suitable metric. In addition, past works have

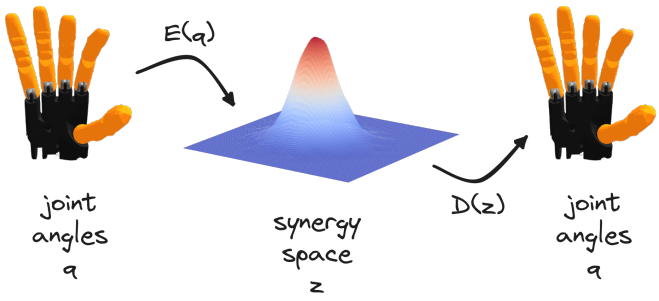

Figure 1.1: Visual representation of the postural synergy framework.

used only implicitly high-level information, such as the grasp type, for the generation of new grasp postures in the latent space. Moreover, this synergistic framework is usually implemented separately from other modalities and information sources. For example, tactile information from the fingers is not considered when controlling humanoid hands using this framework. In this thesis, we aim to expand this control framework for multi-fingered robotic hands.

1.4 Approach and Assumptions

The primary objective of this thesis is to develop a computational framework for implementing synergistic control for robotic hands. We exploit the versatility of this framework, by including additional information about the application in mind: (i) High-level (i.e. semantic) object information such as grasp type, grasp size and object class; (ii) Contact force information from tactile sensors on the fingertips, (iii) Hand pose w.r.t. the object to be grasped, and (iv) Task constraints.

More specifically, in the second chapter of the thesis we discuss previous approaches that were developed for the synergistic control of robotic hands. We explore the several ways that these approaches were applied to various problems, such as object grasping and in-hand manipulation. Additionally, we analyze the limitations and drawbacks associated with these approaches, while also identifying the potential opportunities to extend and enhance synergistic control.

In the third and fourth chapters, we primarily focus on the control of the robotic fingers. In Chapter 3, we explore the generation of grasp postures belonging to several precision grasp types, considering high-level factors such as the grasp type and size. We present a model that takes the desired grasp properties and conditions the synergy space to generate a corresponding grasp, i.e. finger articulation. Additionally, we investigate the effectiveness of using the reconstruction error, which has been commonly employed as the primary evaluation metric in previous works, to compare different methods used for synergy extraction. To illustrate this, we examine the task of in-hand regrasping as our primary application and demonstrate that although certain models may exhibit superior reconstruction performance, their performance in the downstream task may not be equally satisfactory. Moving on to Chapter 4, we extend the synergistic framework to incorporate force feedback obtained from tactile sensors. We design a control architecture that utilizes the force feedback to regulate the high-level grasp properties, namely

the grasp size, enabling the manipulation and placement of objects with diverse physical properties while also avoiding the risk of slipping.

In the fifth and sixth chapters, we concentrate on the hand pose of the robotic hand. In chapter 5, we address the challenge of computing 6DoF hand poses for robotic hands to successfully grasp objects. We present a factorised model, which be generate grasp poses based on the grasp postures provided from the synergistic framework developed before. Finally, in Chapter 6, we present a plan to incorporate task-related information into the grasping process. In this case the goal be to grasp an object in a way that facilitates a downstream task, based on data collected from humans. This integration aim to enhance the effectiveness of grasping and increase the human-robot collaboration capabilities of robots by considering specific constraints derived from human grasping behavior.

Figure 1.2 provides a graphical representation of the thesis structure and its key concepts. In Chapter 3, we introduce the fundamental properties of different grasp types and their integration into the synergy model, analyzing how they influence finger articulation. Chapter 4 focuses on tactile-based control, leveraging the synergy space to regulate finger motions dynamically. In Chapter 5, we present strategies for sampling grasp poses, distinguishing between hand pose and finger synergy space to improve grasp adaptability. Finally, in Chapter 6, we incorporate task constraints into the framework, refining the hand's pose relative to the object to enable task-aware grasping.

To achieve these goals we make some assumptions to facilitate the development of the proposed methods. Firstly we use rigid fully actuated robotic hands that have multiple joints and use position control as the control method. Second, we focus primarily on precision grips, i.e. grips that mainly involve only two or three fingers that come in contact with the object with the fingertips. We also assume that we have recorded a small dataset of grasps executed by a robotic hand, e.g. through teleoperation. Finally, we assume that the robot is equipped with tactile sensors that measure the force applied on its fingertips.

1.5 Contributions

In summary, the contributions of this thesis are the following:

- We introduce a framework for representing conditional postural synergies in robotic hands, which effectively abstracts the low-level control of the robot into high-level characteristics. The foundation of our framework is a conditional generative model capable of encoding and decoding grasp postures based on high-level inputs, such as the grasp type and size [5].

- We validate the effectiveness of our framework by applying it to an in-hand regrasping task. We demonstrate that our framework enables more successful execution of regrasps within its learned synergy space compared to other computational models proposed for synergy extraction [6].

- Additionally, we extend the capabilities of our framework by integrating tactile feedback into the system. We implement a force controller that utilizes the tactile sensors, located on the robot's

fingertips, to control the high-level properties of the synergy space. This allows the hand to autonomously execute object lifting and releasing actions during manipulation tasks [7].

- Furthermore, we develop a factorised grasp sampling model, that integrates both the synergies and the pose of the robotic hand. This model enables as to sample successful 6DoF grasps using precision grasp types [8].

- Finally, we develop a method, relying on reinforcement learning and the synergy framework, that incorporates task-related information, extracted from human data, into the grasping behavior of the agent. By leveraging insights obtained from human grasping behaviors and task requirements, we aim to enhance collaborative capabilities of dexterous robots [9].

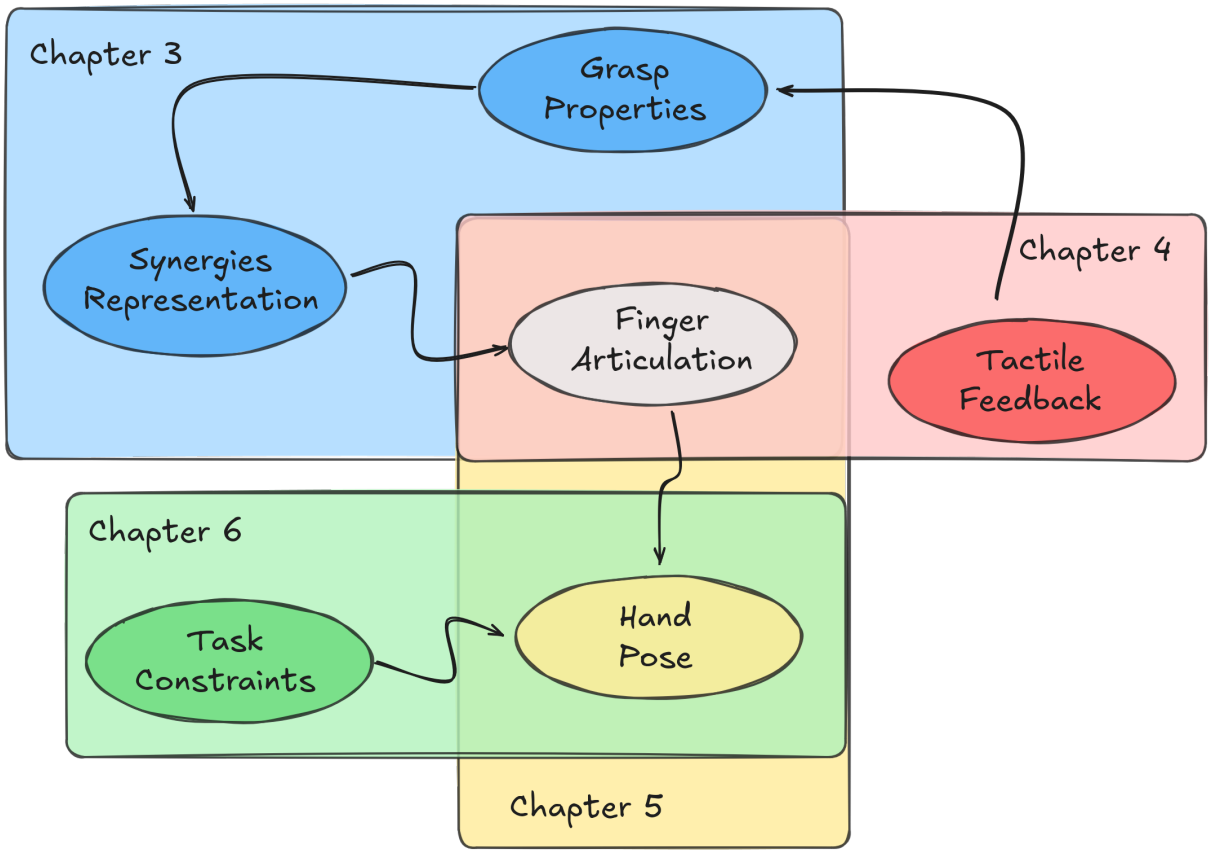

Figure 1.2: Graphical representation of the thesis' outline. Each colored box represents a chapter within the thesis, while each ellipse denotes the main concepts explored in the respective chapter. The arrows indicate the interdependencies between these concepts, illustrating how they influence one another.

1.6 Publications

The research presented here is based on the following publications:

- Dimou, Dimitrios, José Santos-Victor, and Plinio Moreno. "Learning conditional postural synergies for dexterous hands: A generative approach based on variational auto-encoders and conditioned on object size and category." 2021 IEEE international conference on robotics and automation (ICRA). IEEE, 2021.
- Dimou, Dimitrios, José Santos-Victor, and Plinio Moreno. "Robotic hand synergies for in-hand regrasping driven by object information." Autonomous Robots 47.4 (2023): 453-464.
- Dimou, Dimitrios, José Santos-Victor, and Plinio Moreno. "Force Feedback Control For Dexterous Robotic Hands Using Conditional Postural Synergies." 2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids). IEEE, 2022.
- Dimou, Dimitrios, José Santos-Victor, and Plinio Moreno. "Grasp pose sampling for precision grasp types with multi-fingered robotic hands." 2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids). IEEE, 2022.
- Dimou, Dimitrios, José Santos Victor, and Plinio Moreno. "Task-oriented grasping for dexterous robots using postural synergies and reinforcement learning." 2024 Eighth IEEE International Conference on Robotic Computing (IRC). IEEE, 2024.

Chapter 2

Related Work

In this chapter, we review previous works relevant to our research results until writing this proposal and highlight their limitations, which we try to overcome by expanding the synergies framework. The review is in the same order as the chapters after, such that each section in this chapter corresponds to a following chapter. We start by reviewing research on models that extract postural synergies and generate new grasp postures, followed by an overview of regrasping techniques, which serve as the primary application of our work. Next, we examine methods that utilize tactile feedback in humanoid robotic hands for object lifting and stabilization. We then explore various grasp pose sampling strategies for robotic hands. Finally, we review task-oriented grasping approaches, which form the basis of the final section of this thesis.

2.1 Representations of Postural Synergies

2.1.1 Mathematical Background

To begin with, we provide an overview of the mathematical formalism underlying the primary computational methods employed in postural synergy extraction. The fundamental objective in synergy extraction is to encode a grasp posture, typically characterized by a high number of degrees of freedom (DoFs), within a low-dimensional space. To this end, the methods proposed to achieve this objective, predominantly consist of dimensionality reduction algorithms, which are deterministic in nature, and latent variable models, which utilize probabilistic approaches.

The general framework that is used in this context, involves a grasp posture $q \in \mathcal{R}^{d_q}$ and the objective it to discover a mapping function $E_\phi : \mathcal{Q} \to \mathcal{Z}$ that encodes the grasp posture, from the configuration space $\mathcal{Q}$, into a low-dimensional representation $z \in \mathcal{R}^{d_z}$, i.e. the synergy space $\mathcal{Z}$, where $d_z < d_q$, and ϕ are the parameters of the mapping function. Typically, an additional function $D_\theta : \mathcal{Z} \to \mathcal{Q}$ is used to performs the inverse mapping. This inverse mapping function enables the reconstruction of the original grasp posture q from its encoded representation z. Together, the mapping function E_ϕ and its inverse D_θ form a comprehensive framework for encoding and decoding grasp postures within a low-dimensional space.

Deterministic Methods. The first method used to extract the synergies of a set of grasp postures was Principal Component Analysis (PCA). PCA assumes that the map $E(q)$ is linear and deterministic, which means that it can be represented by a matrix equation as $E(q) = z = qW$. PCA optimizes the parameters of the map W such the mean squared error between the original grasp q and the reconstructed grasp $\hat{q}$, is minimized. This problem can be solved analytically and the optimal solution is the matrix W whose columns are the principal components of the grasp dataset Q. In this case, the map that transforms a point in the synergy space back to the configuration space is $D(q) = zW^T$.

Another class of deterministic models used for synergy extraction is the Auto-encoders (AEs) [10]. In Auto-encoders the encoding $E(q)$ and decoding $D(z)$ functions are non-linear deterministic functions, which are approximated with artificial neural networks. Given an input grasp q the encoder network generates a latent point z, while the decoder takes a latent point z and produces a grasp $\hat{q}$ in the configuration space. The parameters of the encoder and the decoder networks are the neural network's weights. The network is trained to minimize the squared error between the input data and the reconstructed ones:

$$L(q, \hat{q}) = ||q - \hat{q}||_2$$

The optimization of the networks' parameters is performed using gradient-based methods, such as Stochastic Gradient Descent. In essence, during training the Auto-Encoder aims to reconstruct the input data using a compressed representation. This compression acts as a bottleneck on the flow of data, forcing the low-dimensional representation (latent representation) to retain the most informative features of the input to successfully reconstruct it.

Latent Variable Models. Latent Variable Models (LVMs) use probability distributions to model the same problem. They assume that the inputs are generated by unobserved latent variables that follow a specific probability distribution. The latent variables are usually low-dimensional vectors that encode the information needed to generate new samples. These assumptions provide a powerful framework for representing high-dimensional datasets. In a mathematical formulation this can be represented with a probability distribution $p(q, z; \theta)$, where q are the observed data points, z are the latent variables and θ are the parameters of the model that need to be computed.

The Gaussian Process Latent Variable Model (GPLVM) [11] uses Gaussian Processes to represent the mapping E, so the encoding procedure becomes a sampling operation from a Gaussian distribution. The optimization criterion in this case is to maximize the likelihood of the data, and is done with a gradient-based method. The mapping E follows the distribution:

$$P(\mathbf{Z} \mid \mathbf{Q}, \theta) = \prod_{j=1}^{D} \frac{1}{(2\pi)^{\frac{N}{2}} |\mathbf{K}|^{\frac{1}{2}}} e^{-\frac{1}{2}\mathbf{z_j}^T \mathbf{K}^{-1} \mathbf{z_j}}$$

To force the model to preserve the relative distances between points in the configuration space in the latent space, a new model with back-constraints was later introduced [12]. These constraints act on the latent points, and ensure that points that are close in the configuration space remain nearby in the latent space. The constraints are simultaneously optimized along with the model parameters.

2.1.2 Methods for Extraction

The framework of postural synergies originated in the field of neuroscience. In the seminal neuroscience study [13], they gathered a dataset of hand postures of human subjects grasping imaginary objects, applying mainly *power* and *precision* grips. They applied the PCA method to analyze the data. In their analysis they demonstrated that the two first principal components were sufficient to account for 80% of the variation in the data. This finding suggested that the brain employs a lower-dimensional space, known as the synergy space, to facilitate the control of hand movements. The study established the synergistic framework as the main candidate to explain how the brain controls effortlessly the many degrees of freedom of the hand.

The robotics community, inspired by this discovery, undertook to replicate this organizational structure in humanoid robotic hands. The PCA method, originally employed in [13], was subsequently utilized in [14, 15] to extract the postural synergies of four hand models: the Barrett hand, the DLR hand, the Robonaut hand, and a human hand model. For the human hand model, the synergies extracted in [13] were used directly. For the robotic hands, hand postures collected in [13], were transferred to each hand by defining a fixed map between the human hand joints and the robotic joints. The extracted two-dimensional synergy space was called *eigengrasps*. The synergy space for each hand was then used to plan stable grasps for several everyday objects. The grasp planning was performed by minimizing an energy function, using simulated annealing, to compute contact points between the hands and the objects. The energy function leveraged the synergy space to control each hand, instead of the original configuration space, effectively reducing the number of control variables. They showed that it is indeed possible to efficiently use the synergy space for grasp planning.

In a later study [16], the PCA method was applied exclusively on precision type grasps. They used two humanoid robotic hands, namely the Shadow Robot Hand and the iCub robot, to execute a series grasps belonging to six distinct precision grip categories. The resulting joint angles for each hand were recorded during these grasping tasks. To facilitate the robot control, a human operator wearing a data glove teleoperated the robots. The study showed that, to describe up to 90% of the variance in the grasp postures, the iCub hand required five synergies, while the Shadow Robot required six synergies.

In a subsequent studies [17, 18], several deterministic non-linear methods for dimensionality reduction were evaluated in controlling a robotic hand within a two-dimensional subspace. In this case, instead of static hand configurations, they utilized recorded finger gait trajectories for the analysis. The evaluation was based on the inconsistency and continuity of the representations. Consistency measured the proximity between two hand postures in the original configuration space, compared to their distance in the low-dimensional representation space. For example, postures similar in the configuration space should be close in the synergy space. On the other hand, continuity described how postures that were adjacent in the motion trajectory should maintain their proximity in the representational space. Their results revealed that the learned representations obtained from the Isomap algorithm, corresponded to more consistent and continuous representations, facilitating the control of a high-DoF hand. However, a fundamental limitation of the Isomap algorithm, is its inability to decode latent points back to the grasp posture space. Consequently, only grasps that are already in the dataset can be executed.

An Auto-Encoder model was employed for extracting the synergy space in [10, 19], They modified the model by incorporating an additional input variable to the decoder network, namely the object size relative to the palm, allowing the model to produce grasps of desired sizes. Furthermore, they augmented the loss function in an attempt to disentangle the grasp categories within the latent space of the model. Consequently, they clustered each grasp type in the latent space and generated type-specific grasps based on these clusters. Their experimental results demonstrated that the AE model outperformed the traditional PCA method in terms of grasp reproduction error. This indicates that the AE model was more effective in accurately reproducing grasp postures compared to the PCA approach.

Finally, a probabilistic approach based on the the Gaussian Process Latent Variable Model (GP-LVM) [11] was proposed in [20, 21] for synergy extraction. In [20], the fingertip positions of several operators were recorded while executing grasps from 31 distinct grasp types. The entire trajectories were recorded and the synergy space was extracted using the GPLVM model. In [21], a dexterous robotic hand was used to collect several hand postures, represented by the joint angles of the robotic hand, while grasping daily-life objects, performing hand signs, and performing in-hand manipulation tasks. Again the synergy space was extracted using the GPLVM model. In both cases, the GPLVM method was compared against the previous deterministic models using the posture reconstruction error as a metric and showed superior performance.

2.1.3 Regrasping.

The ultimate objective of extracting the synergies of robotic hands is to leverage this synergistic framework for improved control in subsequent tasks. An example application of an in-hand manipulation task that this framework has been applied to is generating regrasping trajectories.

For example, in [22], the synergy space, extracted from recorded grasp postures using the PCA method, was used to generate regrasp trajectories for a robotic hand. Five human operators performed 36 reference grasps, and the fingertip positions were recorded. These grasp configurations were then transferred to a dexterous robotic hand using inverse kinematics, enabling the extraction of the synergy space. By selecting two grasps of the same object as base postures, a trajectory was generated directly in the low-dimensional space through linear interpolation between the two configurations. The resulting manipulation trajectories were subsequently executed and evaluated in real-world experiments.

In the study presented in [23], finger trajectories of a dexterous robotic hand executing simple manipulation tasks (such as rotation and translation) were utilized to extract postural synergies using the PCA method. Subsequently, the hand trajectories in the synergy space were parameterized using the Kernelized Movement Primitives method, which employs a Gaussian Mixture Model (GMM) and kernel functions to encode trajectories. Real-world experiments were conducted using the robotic hand, where four different daily in-hand manipulation tasks, previously taught during training, were performed: pouring coffee, opening toolbox latches, sequential grasping of two objects, and playing the board game "carrom".

Both works [22, 23] employed the PCA method to extract the synergy space for the robotic hand, which was then used to generate regrasp trajectories. In [22], regrasps were generated by simple linear interpolation in the synergy space, while in [23], a GMM was used to obtain a probabilistic representation

of the trajectory. In our case, our primary objective is to evaluate the impact of latent space smoothness on the regrasping task. Therefore, we follow the straightforward approach of linear interpolation for generating regrasp trajectories. Additionally, we assess the trajectories generated in the latent spaces of various models to compare their performance in the task.

2.1.4 Limitations

As we discussed the main models that are used for synergy extraction are the PCA, Auto-Encoders, and the Gaussian Process Latent Variable Model. The main metric that was used to compared and evaluate the performance of these approaches was the reconstruction error of each model. We show that the reconstruction error is not the most informative metric for the utility of the learned representation in subsequent tasks. Since the extraction of synergies is done with the intention to use this representation to achieve a downstream task, such as a dexterous manipulation task, we argue that the evaluation metric used to measure the performance of the models needs to be correlated with the performance of the model on the goal task. With this in mind, we propose a new class of models for the extraction of synergies and an evaluation metric that exhibits greater correlation with the performance of the models in an in-hand regrasping task. In addition, the models proposed until now do not provide a mechanism to explicitly generate grasp postures conditioned on high-level information, such as the grasp type or size. We show that the model we propose provides an easy way to conditionally generate new grasp postures based on specific characteristics.

2.2 Tactile Control

The works previously discussed, primarily focused on controlling the angular joint positions of the fingers during object grasping and manipulation, without taking into account any feedback. However, achieving robust and reliable robotic behaviors requires more than just finger position control. Humans rely heavily on tactile feedback to perform complex manipulation tasks [24]. The tactile afferents located in the skin of the fingers provide feedback about contacts and applied forces. Specifically, for tasks requiring precise manipulation, humans heavily rely on feedback from fast-adapting afferents located in the fingertips [25]. The brain processes these tactile signals and translates them into action phases such as grasping, lifting, or releasing. Through the combination of these action phases, humans are able to perform intricate and long-term manipulation tasks [26]. Thus, we argue that incorporating tactile feedback into robotic systems is essential for enhancing their capabilities and enabling them to perform complex manipulation tasks effectively.

The robotics community has equipped robotic hands with tactile sensors that provide force feedback from the fingertips to facilitate object stabilization and slip prevention during the execution of dexterous manipulation tasks. In most cases the force signals are used to predict object slip and prevent it by increasing the applied force on the object. For example, in [27] they use a three-finger robotic hand equipped with the BioTac sensors on its fingertips to detect when slip arises. When slip is detected for the first time the the friction coefficient between the object and the finger is calculated. Then, each

finger is controlled independently to apply a normal force greater that the tangential force divided by the estimated friction coefficient, which according to Coulomb's friction law ensures that slip is avoided. For slip detection they employ two different techniques: a force-derivative method and a pressure-based method. If slip is detected the signal is then classified as linear or rotational slip using a neural network architecture. The controller is applied on real-world object lifting tasks.

In [28], they train a random forest to classify tactile signals into three states: no contact, contact, slip. The tactile feedback is provided by a dexterous robotic hand equipped with BioTac sensors on its fingertips. When a signal is classified as slip the controller of the corresponding finger increases the velocity of the finger in order to increase the applied force and prevent slip. To train the random forest classifier a labeled dataset of slip signals is collected. The controller is applied on stabilising objects in a real-world scenario.

In [29], they train a recurrent neural network to predict slip and adjust the target contact forces in order to avoid it. The tactile feedback is generated from the BioTac sensors located on the fingertips of the Shadow Robot Hand. They collected a dataset of signals labeled with slip information while grasping several household objects. During the execution when slip is detected a PID controller takes the estimated contact force and computes the target joint torques for the finger.

2.2.1 Limitations

All of the previous methods have a common structure in which they employ slip prediction algorithms that inform the system when slip is about to happen. Based on this signal each finger is independently controlled to increase the applied force and avoid slip. The drawbacks of these approaches are threefold: 1) training slip prediction algorithms in most cases requires the collection of labeled data, which is tedious and restricts the range of applications, since collecting data for all possible cases where slip can occur, e.g. for all possible orientations of the hand, is impractical, 2) controlling each finger independently hinders the range of possible grasp types executed since for each new grasp type a different controller for each finger would be need to be used, and 3) the previously developed force controllers have been applied only on stabilizing the objects when grasping and lifting them, while other type of manipulations like transporting or releasing, e.g. by placing them on surfaces, have not been studied. In this thesis we demonstrate how the framework of conditional synergies can be used along with tactile feedback to control a humanoid robotic hand to lift and release objects without requiring a slip prediction module. We achieve this using only one force controller for the entire hand that is also able to handle multiple grasp types, exploiting the synergies of the hand.

2.3 Grasp Pose estimation

Our review until now deals with works that control only the fingers of the hand without considering the hand's position. But the first stage in most manipulation tasks is to reach the object in an appropriate pose. So in this section we review approaches to generating hand poses for grasping.

In general, grasp sampling approaches are usually divided into geometric and data-driven methods.

Geometric approaches use a model of the object and optimize a grasp quality metric in order to find the optimal contact points. They require accurate contact modeling and the definition of cost functions, while the optimization process can be time consuming and subject to local optima. On the other hand, data-driven approaches rely on large datasets of successful grasp examples, which are used to train models to either: 1) approximate some success metric such as the probability of successfully grasping an object, or 2) directly generate a candidate grasp using some kind of generative model, or 3) to learn an end-to-end solution, like reinforcement learning (RL) where the output is a sequence of actions. For an overview of both geometric and data-driven approaches we refer the readers to the reviews [30–33]. In this work we follow a data-driven approach, so we focus our review on this type of methods.

In [34] they aim to directly generate a grasp pose, meaning the finger joint angles and the hand pose, given the object representation. They use a neural network architecture to model this mapping. The input is a voxelized point cloud of the target object and the output is the desired grasp pose. The network is trained on a dataset of successfully executed grasps, computed by a sampling-based motion planner in a simulated environment. The loss function that is used to train the network consists of two terms: a consistency loss, which helps the network choose from the multiple possible grasp poses, and a collision loss that prevents the penetration of the object by the robotic hand. After the grasp is computed from the network a refinement process takes place to further improve the proposed grasp by further minimizing the collision loss term. Later, in [35], they improved upon this work by introducing a differentiable loss function that acts as a grasp quality metric. This loss function consists of a grasp quality metric that evaluates the quality of the grasp based on geometric criteria, a heuristic term that ensures that all the grasp points are close to the object's surface and acts as a guide in cases where the grasp quality metric's gradients do not provide enough information, and finally a collision term, to avoid self-collisions and penetration of the object. The main drawbacks of this approach are that it requires the complete 3D model of the object during runtime in order to accurately compute the introduced loss functions, and since the structure of the produced grasp results from the optimization procedure there is no way to generate grasps with desired characteristics, such as specific grasp types, finally the neural network is deterministic so it generates only one candidate grasp for a given object.

In [36, 37], they propose to model the mapping from the object representation to a grasp using a generative model. Using a generative model allows to sample multiple grasps for a given object. The generative method is based on three probabilistic models: the object model, the contact model, and the hand configuration model. The object model represents the features of the object and is a joint probability function that is derived from the points of the object's point cloud and its curvatures, using the kernel density estimation method. More specifically, in [36] the object model is constructed from a set of multiview images of the target objects, while in [37], the object model is constructed from a point cloud a point cloud representation of the target object. In both cases, the contact model represents the probability function of the object's surface features and the poses of each of the hand's links. The hand configuration model encodes the finger joint angles of the robotic hand and is used to learn a model of the training grasp samples. For each training grasp sample that is provided during training a contact model and a hand configuration model are learned. When an unseen object is presented during runtime

the contact model and the object model are combined to produce a query density model that generates candidate contacts on the new object. Then, a hand configuration is sampled that matches the selected contacts on the object. Finally the grasp is further optimized using simulated annealing to maximize the likelihood of the grasp. In [38], the generative method proposed in [36] was improved by adding an evaluative model at the end of the grasp selection process. The evaluative model takes as input the scene representation and the candidate grasp and estimates the probability of success. This evaluative model can be used to further improve the best ranking grasp in an optimization loop. These approaches are able to sample multiple grasps for a target object but require accurate modelling of the hand and the object in order to learn the object and contact models. In addition, additional high level information such as the desired grasp type are not taken into account.

Explicitly modelling the grasp type of the target grasp has been explored in [39, 40]. In [39], the grasp generation process is modelled using a Bayesian network, which is a probabilistic graphical model. The grasp type is represented as a node in this network, along with the object representation, the grasp configuration, and the probability of grasp success. The Bayesian network is trained using maximum likelihood estimation on a set of training grasp samples. During runtime the a grasp type and a grasp configuration are computed by performing inference on the constructed network, i.e. maximizing the likelihood given the object representation. They demonstrate that explicitly modelling the grasp type increases the number of successful grasps generated. In [40], the grasp generation procedure is broken down in two steps. In the first step multiple candidate grasp poses are generated using a gripper's model, based on geometric criteria. In the second step, each candidate grasp is evaluated by a neural network. The network takes as input the target point cloud and outputs a probability score for each grasp type. The score denotes the probability of success for a grasp of each type given the target object. The grasp type with the highest probability is eventually executed.

Recently several works have developed architectures based on the conditional Auto-Encoder model [41–43] for generating candidate grasps for humanoid robotic hands, following the successful application of this model for grasping with robotic grippers [44]. In [41] they collect a dataset of successful grasps in simulation and train a cVAE model to generate similar grasps. The model is conditioned on the encoding of the partial point cloud observation of the object. In addition, they train a grasp evaluator network to predict the probability of success each grasp and select the one with the highest probability. In [42], a point cloud completion network predicts the missing part of the point cloud observation and a PointNet architecture extracts a representation from it. A cVAE architecture, trained on successful grasps executed in simulation, is conditioned on this representation to generate a grasp. Finally they further refine the grasp to optimize the contact points of the fingers. In [43], a cVAE model, conditioned on the point cloud of object, generates candidate contact points directly on the surface of the object. Then an optimization process computes the optimal finger joint angles to place the fingers on the generated contact points.

2.3.1 Limitations

The works described above usually follow the same pattern in which many grasps, i.e. the hand pose and the finger articulation are generated and then evaluated in order to select the one that scores higher

on a given evaluation metric, such as probability of grasp success or grasp quality metric. In this thesis we model the grasp generation process using the cVAE as our building block. Using this generative model we are able to sample multiple grasps for a target object, and we can condition the generation process on additional information. The works that explicitly model grasp types leave it to the network to choose which grasp type is going to be used as it depends on which one scores higher in each case. In our approach we use the grasp type as a conditional input to the model and generate a grasp to execute the specified grasp type. The work in [39], requires different grasp controllers to execute each grasp type, while in [40] they have predefined hand configurations for each grasp type. Using our approach we can execute multiple grasp types and different configurations within grasp types using only one model/controller that directly outputs the target finger joint angles based on high-level properties. In addition, in the previous approaches the hand pose and the finger articulation was coupled, but in practice the fingers can grasp an object the same way from different hand poses, for example when grasping a cylinder the finger articulation can be the same around the cylinder's body due to its symmetry. In this thesis we explore modelling the different components of a grasp, i.e. the finger articulation, the hand position, and the hand rotation, separately and show that it improves the grasp success rate of the proposed system.

2.4 Task Oriented Grasping

Finally, we explore a more complex topic that integrates human preferences with object functionality and grasping. Task-oriented grasping involves the challenge of grasping an object while taking into account its functional constraints and the intended action after the grasp, such as the purpose for which we plan to use the object. For instance, humans may grasp an object differently depending on whether they intend to use it for a specific task or to pass it to someone else. Adapting grasping behavior to the agent's intention is a crucial skill for humanoid robots working alongside humans.

In the context of robotics, there are two dominant approaches that are employed to model this behavior: (1) Open-loop grasp generation approaches, where a grasp sampling method generates a static grasp pose, taking into account the task constraints to ensure alignment with the task's function. Then, a motion planning algorithm finds a trajectory to execute the proposed grasp. (2) Closed-loop approaches, where a policy takes observations and task constraints as inputs and outputs actions that result in grasping the object in a suitable manner.

Open-loop methods often rely on large labeled datasets for training the grasp generation modules but due to their open-loop nature lack the ability to adjust trajectories based on online observations or account for measurement uncertainties. On the other hand, closed-loop methods typically use reinforcement learning algorithms to train grasping policies from exploration, which requires numerous training samples and complex reward shaping to impose human priors. In the following subsections, we review the current state of the art in both approaches, discussing their advantages and disadvantages, and finally, propose our project plan.

2.4.1 Open-loop Grasp Generation Methods.

In open-loop approaches, the grasping behavior is divided into two stages: first, a grasp is generated, typically represented as a 6DoF pose for the hand and finger configuration and then the target grasp pose is passed to a motion planning algorithm to find a feasible trajectory for execution. Our interest here lies in the grasp generation phase. To generate grasps suitable for a given task, most approaches attempt to model the relationship between grasps, objects, tasks, and other attributes that may be present in the dataset. This modeling is often achieved using probabilistic models, where each feature (i.e., grasp, object, task) is influenced by the other features.

The correlation between the factors influencing human grasping behavior, such as grasp properties (e.g., type or size), object properties (e.g., size or shape), and task attributes, has been investigated in several studies [45, 46]. In [45], data was collected from two housekeepers and two machinists performing various tasks during their professional activities. The tasks were classified based on force requirements, task constraints, and functional task type. Using decision tree and nearest neighbor classifiers, they predicted each attribute based on the others. Their results highlighted a strong correlation between task attributes and grasp and object properties, with potential applications in robotics. Similarly, in [46], human operators manipulated household objects, and features were automatically extracted by segmenting the hand and object. These features included geometric and visual properties of the object (shape, volume, color, etc.), grasp attributes (relative position to the object), and task descriptions. By using the Large Margin Nearest Neighbor Classification algorithm, they successfully classified the extracted features according to the executed task. The classification results underscored a significant correlation between the extracted features and the task being performed.

The fact that humans grasp objects differently according to their post-grasp intention has been exploited by the robotics community to train grasp samplers that use the task attributes to generate task-specific grasps for robotic grippers. These grasp sampling approaches can be divided into two main categories:

1. End-to-end task-grasp modeling: In this approach, the grasp, object, and task relationships are directly modeled, enabling the generation of similar grasps for a given task. The goal is to find correlations between task attributes and grasp-object configurations to create appropriate grasps for specific tasks.

2. Affordance Detection: This approach associates tasks with specific object parts, often referred to as "affordances." By detecting these affordances, the sampler generates grasps tailored to those specific parts of the object, making the grasping process more task-oriented.

Regarding the first type of approaches, in [47], they used a classic geometric grasp sampling method from the GraspIt! simulator to generate grasps for several household objects using both a simulated human hand model and a dexterous robotic gripper. The object's features, such as the bounding box size and convexity and the grasp's properties, such as the position, configuration and stability were recorded. Humans then labeled the grasp-object pairs with task descriptions, such as hand-over, pouring, or tool-use. A Bayesian Neural Network, which is a probabilistic graphical model, was trained using this dataset

to model the relationships between the recorded features. During runtime, the network was queried to output specific features given others. For instance, they sampled grasp positions given the task label and object properties.

In [48], a set of successful power grasps were collected in simulation using a humanoid robotic hand, and a geometric heuristic was employed for grasp sampling. In this case, the task constraints were represented using task-specific contacts for the index finger. A Gaussian Mixture Model was trained from this dataset to learn the correlations between grasps and object properties. New grasps were then predicted using Gaussian Mixture Regression based on the learned model.

In a more recent study [49], a Graph Convolutional Neural Network (GCN) was developed to encode grasp, object, and task relationships. The process involved extracting a low-level representation from the object and the grasp using a PointNet architecture. This encoding was utilized as a node in the GCN to compute a node-level embedding. Subsequently, the embedding was fed into a fully connected network to predict a grasp score. When presented with a new object, multiple grasp poses were sampled using a geometric heuristic, and the grasp-object pair was encoded into the GCN alongside the task description. The grasp evaluator then predicted a grasp score, and the grasp with the highest score was selected.

In the context of affordance detection approaches, the typical procedure involves a vision module that initially detects parts of the object that are compatible with a specific task, such as identifying the handle of a knife for tool use. Subsequently, a grasp sampling method searches for a suitable grasp for that specific part.

For example, in [50], they trained a Convolutional Neural Network to detect task affordance regions on objects from point clouds. Given a task and a voxelized point cloud of an object, the network was trained to predict which voxels of the object are compatible with the given task. The object was then divided into two parts, the functional part used during the task, and the rest that can be grasped. Given the graspable part of the object, an optimization-based grasp sampler computed contact points that result in a high-quality grasp. In [51], they developed a discriminative model that predicted a label indicating whether a grasp is suitable for a given context. The context, in this case, was defined as the label of the task and the object representation (point cloud, spectral reading, and object state). They used a neural network architecture to detect affordances for each part of the object. A geometric heuristic was then used to sample new grasps and for each one the grasp and the context was used to predict a score. The grasp with the highest score was finally selected.

2.4.2 Closed-loop Methods.

Closed-loop methods offer an alternative to open-loop grasp generation by modeling the entire grasping action. Typically, these methods involve training a policy using reinforcement learning to maximize a specified reward. The key challenge lies in appropriately defining the task and designing a suitable reward function to guide the policy towards desired behavior.

For example, in [52], they used data from [53], which consists of 3D models of everyday objects annotated with contact maps captured from humans performing two tasks (using the object and handing it off). Using this data, they trained a CNN to detect affordance regions on objects from RGB images.

The affordance map, along with the state of a humanoid robotic hand, formed the state representation in a reinforcement learning environment. The reward function encouraged the agent to grasp the objects from the same affordance regions that human subjects did. However, the agent's resulting finger configurations were unnatural and did not resemble those of a human hand. To address this limitation and encourage more human-like grasps, the authors later extended their work in [54]. They introduced an auxiliary reward that measures the distance between the robotic hand's posture and a human hand posture grasping the same object. Human hand poses were extracted from internet videos using a hand pose estimation algorithm. This addition led the new agent to learn more natural hand movements, resulting in an increased grasp success rate. However, both methods focus solely on grasping the objects from their functional part, i.e. in order to use it, without taking into account other post-grasp intentions of the agent. So they are not suitable for cases where the robotic agent has more than one post-grasp intentions. Our primary aim is to address this limitation in our approach.

2.4.3 Limitations

Both types of approaches have several advantages and disadvantages. Open-loop grasp methods offer a modular approach to grasp sampling with task constraints, making them versatile and compatible with various applications that can be combined with other techniques. Their probabilistic nature enables the generation of multiple candidate grasps for a given context, and they provide an explainability layer, allowing us to inspect the relationships between different factors. Practically, the machine learning models used for grasp sampling are relatively easier to train. However, a significant drawback is that training such models typically requires large amounts of labeled data, particularly with human-based annotations, which can be challenging to obtain. Additionally, incorporating trajectory planning and collision avoidance adds another layer of complexity. Notably, open-loop grasp generation with task constraints has not yet been widely applied to humanoid hands, possibly due to their increased complexity compared to simpler grippers or robotic hands.

Reinforcement learning approaches do not require successful grasp examples for training, and rely instead on learning from experience. Additionally, these policies can perform the entire grasping task, eliminating the need for separate trajectory planning or collision avoidance steps. Their closed-loop nature also makes them more robust in the face of external perturbations or measurement uncertainties. However, a crucial limitation lies in their dependency on specific reward functions, potentially restricting their applicability to only the tasks defined by these rewards. Consequently, using the grasping behavior in a new task may require retraining the agent with a different reward function. In practice, these algorithms demand vast amounts of samples, making simulation-based training more feasible than real-world applications. Additionally, effective training often requires careful reward shaping to achieve desirable behavior.

To overcome some of these limitations, we develop a solution that uses a closed-loop approach, i.e. a policy trained with reinforcement learning, as its backbone, but also incorporate elements from the open-loop methods. This way we are able to solve the entire grasping task end-to-end without relying on multiple components. In order to accelerate the learning process, which is the main limitation of RL

approaches, we use a small dataset of of task-oriented grasps executed by humans, to guide the policy, and the hand synergy framework developed previously to reduce the action space.

Chapter 3

Representing Postural Synergies for Multi-fingered Hands

In this chapter, our primary focus is on the methods employed to extract postural synergies of multi-fingered robotic hands. A crucial aspect to consider is the evaluation of these methods. Since, the ultimate objective of learning the synergy space is to utilize it in subsequent tasks, it is essential that the evaluation metric reflects the model's performance in those tasks. However, in most cases, the mean squared error is used as the evaluation metric for different synergy models. We demonstrate that this metric is not indicative of the utility of the learned representation. Instead, a metric that is associated with the goal task is required. To this end, we introduce a metric for evaluating the learned representation of synergy models, tailored to an in-hand regrasping task. Moreover, we present a new class of models for learning the synergy space, based on the Variation Auto-Encoder (VAE) model. We also demonstrate the ability of this framework to encode additional high-level information about the grasp posture. Finally, we demonstrate that this new class of models learns better latent spaces, which in turn results in higher performance in the regrasping task.

3.1 Models for Synergy Extraction and Applications

As mentioned in the previous chapter, the majority of models proposed for extracting postural synergies exhibit a common structure, comprising an encoding and a decoding function. The role of the encoder is to take a grasp posture and transform it into a low-dimensional representation known as the synergistic components. Conversely, the decoder operates in reverse, generating a grasp posture from a given low-dimensional point. The specific terminology assigned to these synergistic components may vary depending on the chosen framework, such as principal components in PCA, encodings in AEs, or latent points in latent variable models. However, the underlying principle remains the same across these models. The models are typically trained using a dataset consisting of collected grasp postures, which can contain static postures or snapshots captured during the execution of a manipulation trajectory. The grasp postures are usually represented by the robotic hand's joint angles or the position of the fingertips.

3.1.1 Evaluation of Synergy Models

Previous works, that propose new models for synergy extraction, commonly employ the reconstruction (or reproduction) error of the model, to evaluate the performance of the synergy model. This metric quantifies the dissimilarity between the original grasp posture and its reconstructed counterpart. It is computed by encoding an original grasp posture into a latent point, decoding this latent point to generate a *reconstructed* grasp posture, and then computing the Euclidean norm between the original and the reconstructed grasp postures. The mean squared error (MSE) across all grasps in the dataset is subsequently computed. This metric is assumed to reflect the ability of the model to represent the observed grasp postures. However, in the case of non-linear models, for a grasp posture q encoded in a latent point $z = E(q)$, there exist a neighboring latent point z', such that the decoded grasp $\tilde{q}' = d(q')$ from this point is closer to the original grasp q compared the grasp $\tilde{q} = d(z)$ decoded from the original latent point z. More specifically:

$$||\tilde{q}' - q||_2 < ||\tilde{q} - q||_2$$

This indicates that a model may have a large reconstruction error for a particular grasp posture while still being able to effectively represent it in another region of the latent space. This discrepancy is observed in all non-linear models, while PCA does not exhibit this property. The likely reason behind this is that in PCA we can find a globally optimal solution, whereas in the non-linear models we find only locally optimal solutions since we use gradient based methods.

Moreover, the reconstruction error as a metric fails to provide any relevant information regarding the synergy model's performance in subsequent tasks. We claim that the evaluation metric should be closely aligned with the specific task at hand. In light of this, we propose employing the smoothness of the learned latent space as an alternative evaluation metric for comparing synergy models. Smoothness quantifies the degree of variation in grasp postures as we traverse adjacent points in the latent space. For instance, if we want to generate a manipulation trajectory, e.g. using a sampling-based motion planner, directly in the latent space to solve an in-hand manipulation task, we would want neighboring points to produce similar grasp postures without exhibiting abrupt changes in joint angles between them. This metric shares similarities with the consistency metric introduced in [18]. However, in their case, the metric assesses the quality of the encoding stage in the process, since they are unable to decode new latent points. In our scenario, our objective is to evaluate the distance between grasp postures decoded from new latent points that cover uniformly the entire latent space.

To compute the smoothness of a latent space we average the smoothness in the neighborhood of each latent point. We begin by encoding all grasp postures that are contained in our dataset, we then compute the bounding box of these latent points, and define a Cartesian grid inside it (Figure 3.1.a). For each vertex z_i on the grid (Figure 3.1.b), we compute the difference of the joint angles between the grasp posture $q_i = D(z_i)$, that is decoded from z_i, and the grasp postures $q_j = D(z_j)$, where $j = 1, .., 8$, decoded from the vertices z_j which are the adjacent vertices of z_i (Figure 3.1.c). The smoothness of the latent point z_i is then defined as:

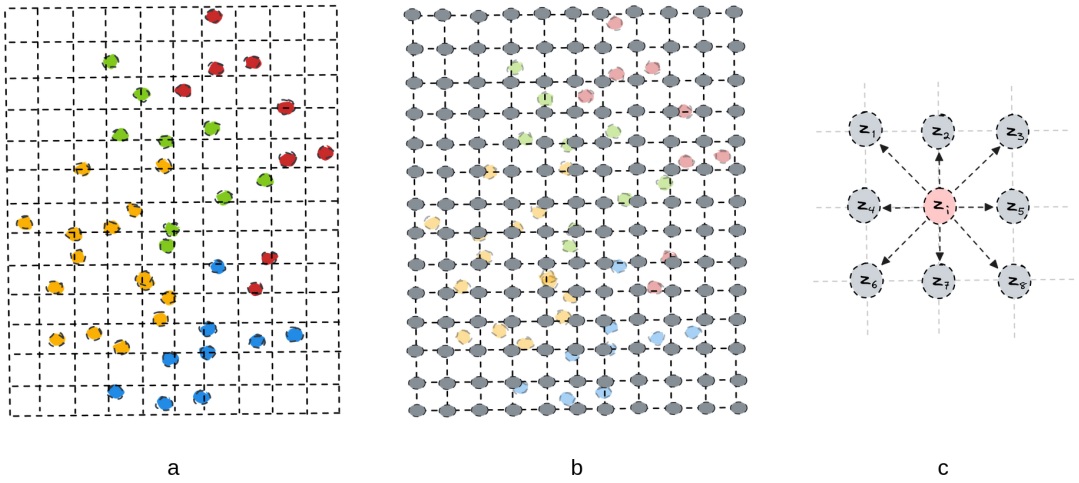

Figure 3.1: Illustration of the smoothness computation procedure for a hypothetical 2D latent space. a) The 2D colorful points correspond the latent points, where each color represents a distinct grasp type. The dashed lines depict to the Cartesian grid within the latent space. b) The gray points indicate the vertices on the latent grid. c) For every point z_i on the latent grid, we calculate the difference in the decoded grasp postures, between that point and its adjacent points z_j where $j = 1, .., 8$.

$$\mathbf{s}_i = \frac{\sum_{j=1}^{8} d_{ij}}{8}, \text{ where } d_{ij} = ||D(z_i) - D(z_j)||_2$$

Finally, the smoothness of the entire latent space is defined as the average smoothness of the latent points on the grid:

$$\mathbf{S} = \frac{\sum_{i=1}^{N} \mathbf{s}_i}{N}$$

where N is the number of vertices on the grid and measures the resolution of the grid and can be arbitrarily defined. Moreover, in order to quantify how smoothness changes in each point's neighborhood we compute the variance $v_i = Var(d_{ij})$ and the entire variance of the latent space $V = Var(s_i)$. We consider the smoothness s_i as a discretized gradient of the latent point z_i, indicating the average difference in joint angles when transitioning to the surrounding points on the grid.

3.1.2 Variational Auto-Encoders for Synergy Extraction

Building on the previous discussion, we propose to utilize a different type of model for synergy extraction, namely the Variational Auto-Encoder (VAE) [55, 56]. VAEs are generative models that can be viewed as an extension of classical AEs. Notably, VAEs have demonstrated the ability to learn smoother and more meaningful latent spaces compared to AEs [57]. To this end, w begin by presenting the formalism of VAEs and how they can be effectively utilized for synergy extraction. By leveraging the advantages offered by VAEs, we aim to learn smoother synergy representations that can be leveraged in downstream tasks.

A Variational Auto-Encoder is a latent variable model that introduces a computationally efficient framework for optimizing the model's parameters. It consists of an encoder network $E_{\boldsymbol{\phi}}(z|q)$, parameterized by $\boldsymbol{\phi}$, and a decoder network $D_{\boldsymbol{\theta}}(q|z)$, parameterized by $\boldsymbol{\theta}$, similar to classical Auto-Encoders. In the

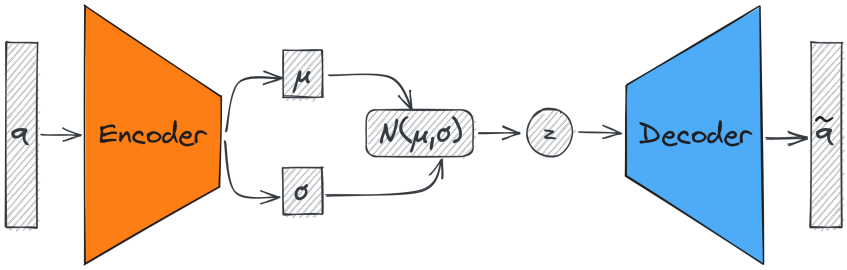

Figure 3.2: Graphical representation of a VAE model.

VAE framework though, the encoder and decoder networks model probability distributions. The encoder is an inference model that takes as input a data point q and infers a latent point z. In practice, the encoder is parameterized by a neural network and produces the mean $\mu \in \mathbb{R}^{d_z}$ and variance $\sigma \in \mathbb{R}^{d_z}$ of a Gaussian distribution. The latent point z is then sampled from this Gaussian distribution, in contrast to the AE model that the latent point is the output of the encoder network. The decoder model, which is also parameterized by a neural network, uses a latent point z to generate a point in the original data space $\mathcal{Q}$. A visual representation of a VAE network can be seen in Figure 3.2. Both networks are trained jointly using variational inference, which aims to minimize the Kullback-Leibler divergence between the true posterior distribution $p(z|x)$ and a variational distribution $q(z|x)$. In this case, instead of directly maximizing the likelihood of the data, which is computationally hard, we maximize the evidence lower bound (ELBO) [55]:

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - D_{KL}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})\right) \tag{3.1}$$

The first term corresponds to the mean squared error between the the input and reconstruction, while the second term plays a regularization role for the generative model. The optimization of the network's parameters is performed with gradient-based algorithms such as ADAM [58].

More specifically in our case, in order to train a VAE model to learn a synergy representation of grasp postures we need a dataset $Q = \{q_1, q_2, .., q_n\}$ of example postures. We can then train the model on this dataset using stochastic gradient decent by taking batches of data passing them through the model, computing the loss, and back-propagating the error. After the model is trained, we can encode grasp postures into the latent space, and generate new grasps by sampling a latent point from the prior distribution of the model and passing it through the decoder.

3.1.3 Conditional Postural Synergies

The VAE framework can be easily extended to generate data explicitly conditioned on discrete and continuous variables. This means that we can model high-level properties, such as the grip type, and quantities, such as the grasp size, using one model. This way we can explicitly generate grasps that have specific desired properties.

In the conditional version of the VAE (cVAE) the encoder and the decoder network are conditioned on an auxiliary variable c, that can be continuous or discrete. In this case, the ELBO takes the following

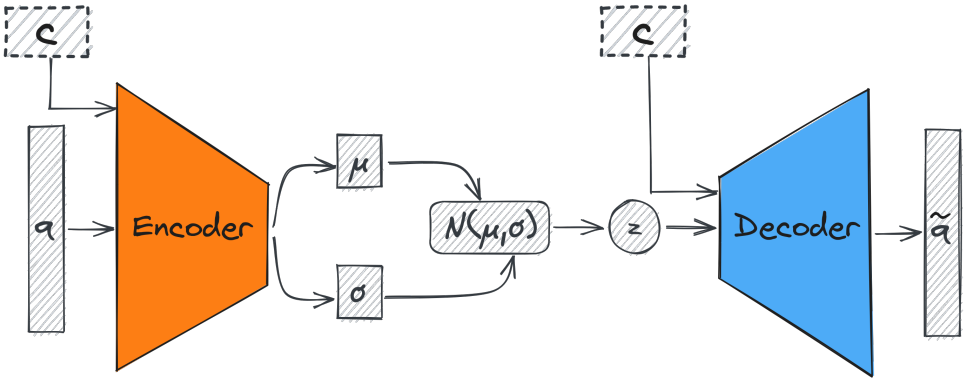

Figure 3.3: Schematic representation of a cVAE model.

form:

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{c})}\left[\log p_\theta(\mathbf{x}|\mathbf{c},\mathbf{z})\right] - D_{KL}\left(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{c})\|p(\mathbf{z})\right)$$

The optimization of the loss function is performed with standard gradient descent algorithms [58]. We use this framework to train a cVAE model to represent the synergies of grasp dataset and we condition the generation process on the grasp size and the grasp type. An example of the architecture of the cVAE can be seen in Figure 3.3.

The conditional variable c can take various forms, either continuous or discrete, and it represents a specific attribute of the grasp posture. In our particular case, we utilize two conditional variables for our model: 1) the shape of the object, which is a discrete variable, and denotes the category to which the grasped object belongs, and 2) the size of the object, a continuous variable ranging from zero to one, that represents the relative size of the object within our dataset. A value of zero corresponds to the smallest object in our dataset, while a value of one corresponds to the largest grasped object. In the next section (3.2) we provide more details on the dataset and the training procedure.

3.1.4 Case Study: In-hand Regrasping

In order to assess the proposed model and evaluation metric in the context of a manipulation task utilizing synergies, we selected an in-hand regrasping task, in which we are generating the regrasp trajectories directly in the latent space of the model. We assume that we have a dataset of grasp postures belonging to distinct grasp types applied to various objects. The objective of the regrasping task is to transition from an initial grasp posture, belonging to one specific grasp type, to a target grasp posture, belonging to another grasp type, while maintaining stability of the object in the hand. We follow a straightforward approach for generating the regrasp trajectories in the latent space of the models. We encode the initial grasp posture $q_{initial}$ and the target grasp posture q_{target} into the latent space, resulting in the respective latent points $z_{initial}$ and z_{target}. Subsequently, we perform linear interpolation between these two points in Euclidean space, sampling N points to represent the trajectory steps, where N corresponds to the desired number of steps in the trajectory. Finally, we decode the sampled latent points into grasp postures $q_1, q_2, .., q_N$, obtaining a trajectory in the configuration space. Figure 3.4 provides a

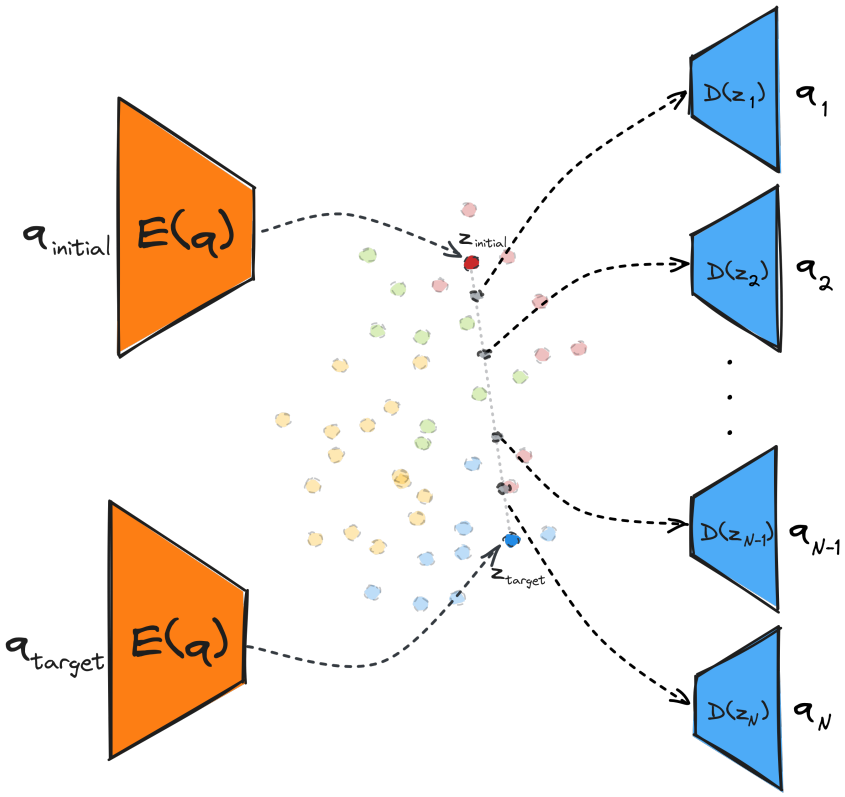

Figure 3.4: The provided figure illustrates the procedure for generating trajectories for in-hand regrasping. Firstly, the initial and target grasp postures are encoded in the latVaticent space, resulting in the respective latent points $z_{initial}$ (red point) and z_{target} (blue point). The color of the points represents the grasp type associated with the original grasp posture. Next, a series of N latent points are sampled by performing linear interpolation between the initial and target points. These sampled latent points are then decoded, yielding corresponding grasp postures. Finally, these grasp postures form a manipulation trajectory in the task space.

visual depiction of the trajectory generation process. In essence, instead of employing a complex planning algorithm to find a sequence of states that can perform the required regrasp we rely on the structure of the latent space of the model. If the latent space is smooth enough we can generate successful trajectories by a simple procedure such as linear interpolation.

3.2 Experiments and Results

3.2.1 Experimental Set-up

Dataset. For our experiments, we trained our models using two datasets consisting of recorded grasps originally presented in [16]. These datasets captured grasp postures performed with two humanoid robotic hands: the Shadow Dexterous Hand and the iCub robot hand. Human operators teleoperated the robots using data gloves. The dataset for the Shadow Hand includes *438* grasps, while the iCub hand dataset contains *536* grasps. During teleoperation, a mapping was applied to transform the joint angles of the human operator's hand to the corresponding joint angles of each robotic hand.

Each recorded grasp is represented by the angle values in degrees of the joints of each robotic hand. The Shadow hand has *21 joints* while the hand of the iCub robot has *7 joints*. Each grasp is classified,

according to the grasp taxonomy [59], in one of the eight following precision-grasp types: tripod, palmar pinch, lateral, writing tripod, parallel extension, adduction grip, tip pinch, lateral tripod. Twelve different objects were grasped, belonging to five distinct categories: ball (three sizes), box (three sizes), cylinder (three sizes), pen and cube. The objects were grasped from different sides adding up to a total of 20 different object configurations. We used a subset of each of these datasets, that had labels for the size of the object. More specifically the grasps for the sphere, the box, and the cylinder. The assigned size labels were: large, medium, and small. We mapped these discrete labels to the continuous range $(0.0, 1.0)$ as follows: large grasps were assigned the value 1.0, medium size grasps were assigned 0.5, and small size grasps were assigned 0.0. Additionally, shape labels indicating the object geometry (sphere, box, cylinder) were one-hot encoded into 1D vectors. The shape vector and the size value were concatenated to form the label vector corresponding to each grasp.

Models. For each grasp dataset we trained six different synergy models: a PCA model, an AE, a BCGP-LVM with a linear mapping, a BCGP-LVM with a multi layer perceptron (MLP), a BCGP-LVM with a radial basis function (RBF), a VAE, and a CVAE conditioned on the object's shape and size.

3.2.2 Quantitative and Qualitative Analysis

Our objective is to determine the utility of the learned latent space for controlling a humanoid robotic hand. To assess the effectiveness of the learned representation, we begin by presenting quantitative results. One option is to compare the reconstruction error, which measures how well the grasps are represented in the latent space. However, as we previously discussed the limitations of this metric, we introduced an alternative: the smoothness of the latent space. We evaluate the models based on both the reconstruction error and the smoothness of the learned representation. Furthermore, we provide qualitative results by decoding grasps from each model's latent space. This allows us to visualize the effects of smoothness on the resulting grasp postures and gain a better understanding of the quality of the learned representations.

Reproduction error. Following previous works, we initially compared the models by evaluating their reconstruction error on each dataset. We trained all models on each dataset and calculated the corresponding reconstruction error. We trained the models with an increasing number of latent dimensions, ranging from 1 to the total number of degrees of freedom of each robotic hand, to show how increasing the number of latent components increases the representation capacity of each model. The results are presented in Figure 3.5. The results indicate that the BCGPLVM model with the RBF kernel as a back constraint consistently achieves the lowest reconstruction error across all latent dimensions, in agreement with previous findings [20]. While the VAE and CVAE models perform better than PCA when the number of latent dimensions is small, the other non-linear methods outperform them in terms of reconstruction accuracy.

As we progressively increase the number of latent dimensions to match the degrees of freedom of the hand, most models exhibit a trend where the reconstruction error approaches zero. However, this is not observed in the VAE and CVAE models, as their loss functions include an additional term accounting for the Kullback-Leibler (KL) divergence between the prior and posterior distributions. This divergence term introduces a regularization effect that prevents the reconstruction error from reaching zero.

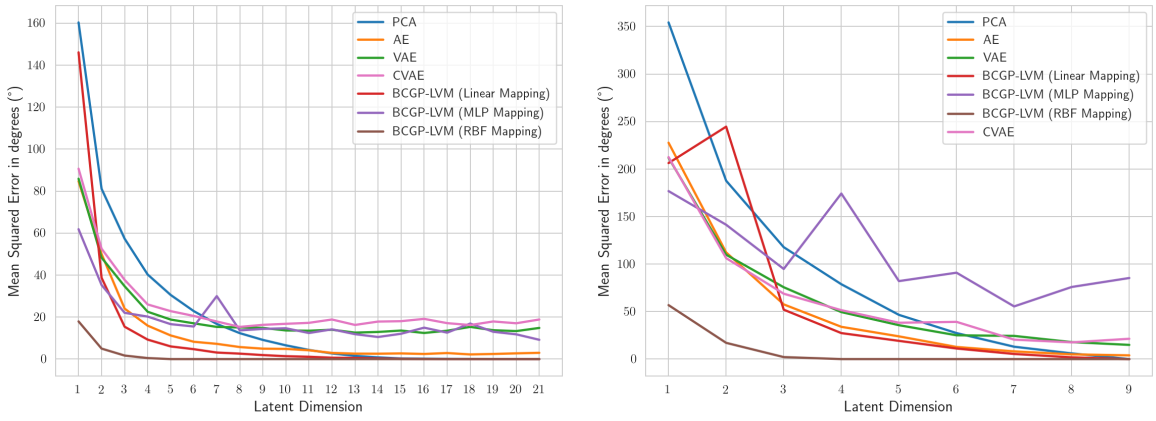

Figure 3.5: Reconstruction results for the models trained on the Shadow Hand dataset (left) and the iCub dataset (right).

Smoothness of latent space. We have previously discussed the limitations of using the reconstruction error as a sole measure to evaluate the effectiveness of the learned synergy representation in downstream tasks. Instead, we propose a different approach by quantifying the smoothness of the latent space. Our rationale is based on the assumption that in a smooth latent space, manipulation trajectories can be generated with minimal abrupt changes in joint angles, thereby increasing the probability of successful execution.

We calculate the smoothness using three different grid resolutions $N = 5, 15, 25$, to observe how the mean and variance of smoothness change with the grid scale. The mean represents the average variation in grasp posture between adjacent points on the grid, while the variance indicates the presence of regions within the latent space exhibiting different levels of smoothness. In the case of the cVAE model, we compute the smoothness for each possible conditional value and report the average. The results for the iCub robot hand can be found in Table 3.2, and the results for the Shadow hand are presented in Table 3.1.

We observe that across both hands and various grid resolutions, the CVAE model demonstrates the smallest average variation between neighboring grasp postures. This indicates that by conditioning the grasps on high-level information, the model clusters similar grasp postures together in the latent space. Consequently, transitioning between adjacent latent points does not result in significant changes in the grasp space. On the other hand, the PCA method exhibits the lowest variance, which can be attributed to its linear nature. As a linear mapping, the distance between two latent points leads to a constant change in the joint angles of the resulting grasp postures along each grid direction. In contrast, the BCGPLVM model with the RBF kernel as a back constraint, despite having the lowest reconstruction error, displays the largest average variation in grasp postures between neighboring points. As we will see next, this implies that decoding grasps from nearby latent points results in substantially different grasp configurations, potentially hindering planning in the latent space of this model.

To gain an intuitive understanding of the impact of latent space smoothness, we can decode the grasp postures from the grid vertices and visually examine them. In Figure 3.6, the first and second rows depict visualizations of the latent spaces learned by the PCA, VAE, and BCGP-LVM (RBF) models for

Figure 3.6: Grasp postures decoded from the a regular $5x5$ grid in the 2D latent spaces of three models: PCA (first column), VAE (second column), and BCGPLVM (RBF) (third column). The first row contains grasp postures using the Shadow Hand and the second row using the iCub hand.

Table 3.1: Smoothness results for the Shadow Hand: The mean and standard deviation calculated from the latent space gradients of each model for three grid resolutions. Lower values suggest a smoother latent space.

	(μ, σ)		
	N=5	N=15	N=25
PCA	(14.0, 2.8)	(4.0, 0.8)	(2.3, 0.5)
AE	(19.3, 7.2)	(6.8, 2.6)	(4.1, 1.6)
VAE	(18.3, 5.2)	(5.7, 1.7)	(3.3, 1.0)
CVAE	**(12.0, 3.2)**	**(3.6, 1.0)**	**(2.1, 0.6)**
BCGPLVM (Linear)	(15.2, 4.6)	(6.7, 2.5)	(4.0, 1.5)
BCGPLVM (MLP)	(15.3, 4.5)	(5.5, 1.8)	(3.2, 1.1)
BCGPLVM (RBF)	(23.3, 8.0)	(14.7, 7.5)	(10.4, 5.8)

Table 3.2: Smoothness results for the iCub Hand: The mean and standard deviation calculated from the latent space gradients of each model for three grid resolutions. Lower values suggest a smoother latent space.

	(μ, σ)		
	N=5	N=15	N=25
PCA	(26.0, 5.0)	(7.4, 1.4)	(4.3, 0.8)
AE	(33.9, 8.9)	(10.8, 2.9)	(6.4, 1.7)
VAE	(29.1, 10.2)	(8.7, 3.2)	(5.1, 1.9)
CVAE	**(21.9, 6.7)**	**(6.3, 2.0)**	**(3.7, 1.2)**
BCGPLVM (Linear)	(28.2, 8.8)	(12.1, 6.0)	(7.3, 3.7)
BCGPLVM (MLP)	(27.1, 8.3)	(8.9, 3.5)	(5.3, 2.1)
BCGPLVM (RBF)	(46.2, 17.5)	(26.6, 14.5)	(17.1, 10.1)

both the Shadow and iCub hands, respectively. The decoded grasp postures originate from a 5×5 grid (similar to the one shown in Figure 3.1.b), offering visual insights into the latent space structure. We can observe that the PCA's latent space exhibits similarities among neighboring grasps, with consistent variations. Moreover, movement along the y axis predominantly affects the thumb, while movement along the x axis primarily influences the other fingers. Similarly, the latent space of the VAE model displays a similar pattern, although the thumb and other fingers appear more interrelated, likely due to the non-linear relationship introduced by the model. In contrast, the BCGP-LVM model with RBF back constraint demonstrates considerable irregularity between neighboring grasp configurations. This irregularity poses challenges for planning algorithms, as small movements in the latent space can lead to significant differences in decoded grasps.

3.2.3 Regrasping Experiments

To demonstrate the impact of latent space smoothness and to highlight the inability of the reconstruction error to provide information about the utility of the learned representation on downstream manipulation tasks we execute regrasping trajectories generated directly from the latent space. The goal of the task is to change the grasp type executed on the object from an initial grasp to a target one while

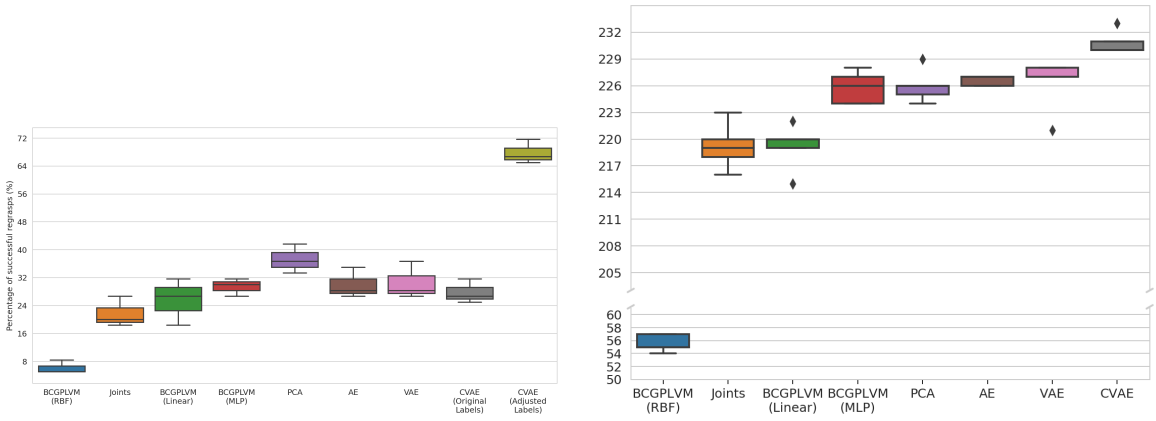

Figure 3.7: Regrasping results.

the object remains stable during the execution. We performed two experiments, one in simulation using the Gazebo simulator for the Shadow hand, and one in a real-world setting with the iCub hand.

For the experiments conducted with the Shadow Hand, we selected 16 grasps from our dataset that corresponded to the red box being grasped from the large side. We generated regrasping trajectories for all possible pairs of these grasps, resulting in a total of 120 trajectories. To account for the influence of the starting grasp on the success of each trajectory, we executed the trajectories in both directions for each pair, leading to a total of 240 trajectories executed. The models employed in these experiments utilized two latent dimensions. As we described before, to generate a regrasp trajectory for a pair of grasps, we encoded them using each model, performed linear interpolation between the latent points, and sampled 10 points on the line connecting the latent points. Finally, we decoded these interpolated points. As a baseline, we also performed the interpolations directly in the joint space.

Table 3.3 presents the time measurements for generating a trajectory using each method on a desktop PC with an Intel®Xeon E5-1603 processor running at 2.80 GHz and 16GiB of memory. Due to the stochastic nature of the simulator, we executed each trajectory five times and plotted the number of successful regrasp trajectories for each model is shown in Figure 3.7 in a box plot format. The vertical axis represents the percentage of successful regrasp trajectories generated by each model. Each box represents the interquartile range between the three runs of each trajectory, while the line in the middle of the box indicates the median. Notably, the models with lower means in Table 3.1 exhibited better performance compared to the models with lower reconstruction errors from Figure 3.5. This result supports our argument that smoothness is a more reliable metric than reconstruction error. Moreover, we observed that the CVAE model conditioned on the object shape and size achieved, on average, a higher number of successful trajectories compared to all other models.

We also conducted real-world experiments with the iCub robot, utilizing all the objects that were used to execute the grasps from the training dataset, as depicted in Figure 3.8. For each object configuration, which corresponds to each side from which an object can be grasped (e.g., a sphere has one object configuration, a cylinder has two configurations: top and side, and a box with three differently sized sides has three configurations), we randomly selected 5 pairs of grasps. Each grasp in the pair was associated with a specific grasp type. For each pair, every model generated a regrasp trajectory, resulting in a total

Table 3.3: Time measurements for generating a 10 step trajectory for each method presented in Figure 3.7.

Method	**Time in milliseconds**
Joint interpolation	0.35
PCA	0.42
AE	3.87
VAE	3.85
CVAE	4.32
BCGPLVM (Linear)	5.72
BCGPLVM (MLP)	5.59
BCGPLVM (RBF)	5.62

of 60 regrasp trajectories.

Similar to the simulation experiments, the number of steps in each trajectory was set to 10. To account for variability in initial conditions, each trajectory was performed three times. In these real-world experiments, the robot was constrained to execute the regrasp trajectory on the same side of the object. This constraint was enforced using the object size conditional variable, which compelled the model to produce tight grasps and ensure continuous contact between the fingertips and the object. Consequently, since no contact was broken or created during the regrasping process, the object was consistently grasped from the initial side.

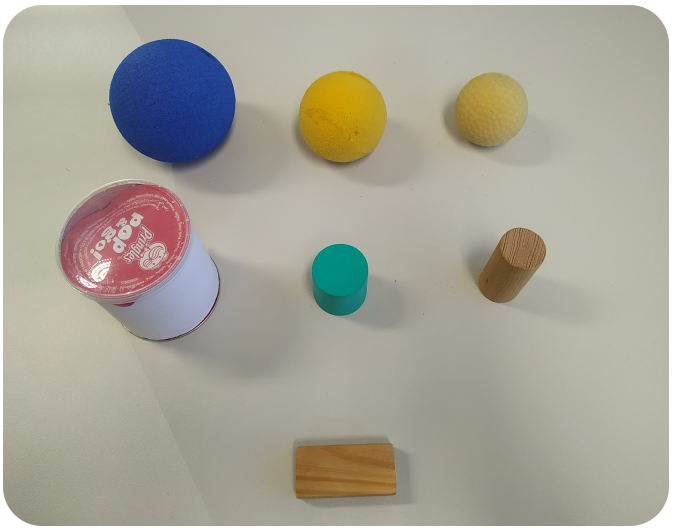

Figure 3.8: Objects used to execute regrasp trajectories.

During our preliminary experiments, we observed that the regrasp trajectories generated by the CVAE model did not outperform the other models as suggested in [5] when executed on the real robot. The reason behind this discrepancy was that while the transitions between states were smooth, the unaccounted properties of the objects, such as their material and mass, were causing instability in some grasp postures due to slippage, particularly in small-sized objects with smooth surfaces.

To address this issue, we leveraged the conditional variables encoded by the CVAE model, specifically the object size. When generating trajectories using the CVAE model, we adjusted the size label to be

lower than the original label. More specifically, during the decoding process of each trajectory's latent points, we reduced the size label of each point by a value of 0.5. By doing so, the model produced firmer grips that were more stable during execution.

It is worth noting that this adjustment is not applicable to the other models since they cannot be conditioned on additional variables. The results of executing the trajectories are presented in Figure 3.7 in the form of a box plot. We observe that most models exhibit a similar pattern to the results reported in [5]. However, the CVAE model utilizing the original size labels for the object does not surpass the performance of the other models. Conversely, the trajectories generated by the CVAE model with the adjusted size labels result in twice as many successful regrasp trajectories. Figure 3.9 showcases real-world examples of the regrasp trajectories generated by the CVAE model with the adjusted size labels and executed using the iCub robot.

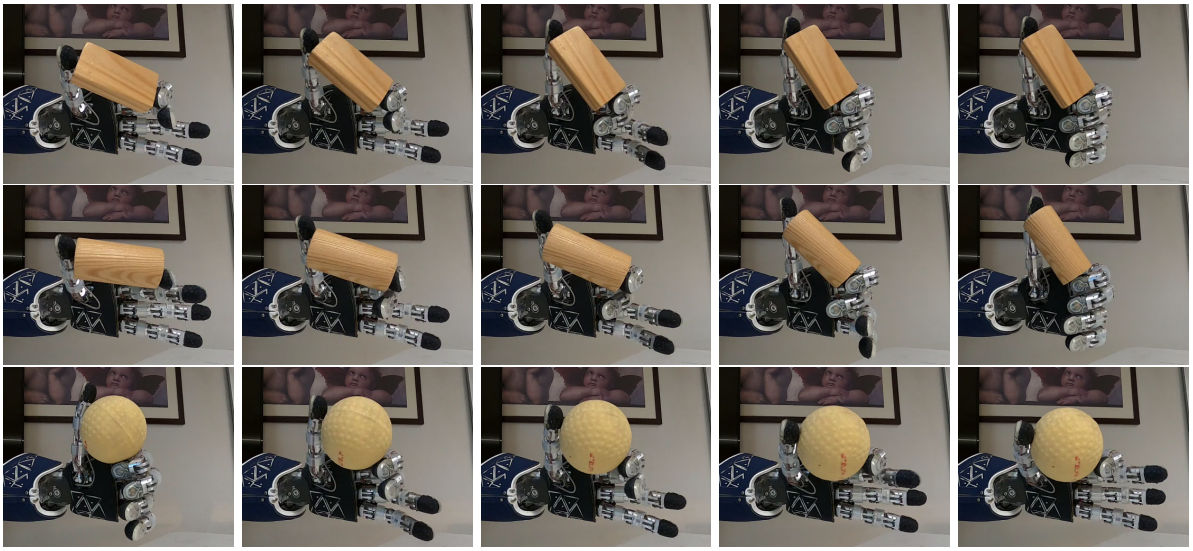

Figure 3.9: Example regrasp trajectories executed on the iCub robot using four different objects. Each trajectory has ten steps. The pictures are taken every two steps. The first trajectory moves from a tripod grasp type to a lateral, the second from a tip pinch to a lateral, and the third from a tip pinch to a lateral.

3.3 Conclusions

In conclusion, we have introduced a novel class of models for extracting postural synergies in humanoid multi-fingered hands, utilizing the framework of Variational Auto-Encoders (VAEs). Our proposed model is capable of learning a representation space for a set of grasp postures and generating new ones. Moreover, we extended the VAE framework to its conditional version to train a model that can generate grasps conditioned on high-level properties such as grasp type and size. While previous approaches may achieve more accurate reconstruction of recorded grasps, we have demonstrated that this does not necessarily correlate with the model's success in subsequent tasks. Instead, we have introduced the smoothness of the latent space as a novel evaluation metric and empirically shown its impact on an in-hand regrasping task, both in simulation and the real world. Our work emphasizes the importance of evaluating postural

synergy representations based on their effectiveness in downstream tasks.

However, it is important to note that our current approach lacks consideration of geometric information regarding the state and shape of the interacting object during trajectory planning. In future research, we aim to explore more intelligent methods for generating trajectories in the latent space. For instance, by leveraging the smoothness of the neighborhood within the latent space, we can avoid regions that result in significant changes in hand configuration, thus enhancing the planning process.

Chapter 4

Integrating tactile feedback with Postural Synergies

In the previous chapter we demonstrated a framework for efficiently controlling a robotic hand's fingers in the synergy space. However, our approach did not incorporate any feedback from the environment, preventing the ability to react to external disturbances and uncertainties. In contrast, humans heavily rely on tactile feedback from the hand's skin to perform complex manipulation tasks that involve contact interactions. This feedback is provided by tactile afferents located in the skin. Particularly, for precise movements, afferents located in the fingertips are utilized, which have high density and adapt fast to pressure changes [25]. These afferents provide information about the characteristics of the exerted contact forces, such as the magnitude and the direction.

In order for humanoid robots to successfully perform complex manipulation tasks in unstructured environments, such as household settings, force signals need to be utilized to facilitate control and provide feedback on the applied forces. Without force feedback, robots using open-loop position control may fail to stably grasp objects due to external forces, such as gravity, or inadvertently damage fragile objects due to the lack of information about the magnitude of the applied forces.

Existing approaches that utilize tactile sensors to modulate finger control during grasping, primarily rely on slip prediction algorithms. These algorithms process force signals from the tactile sensors and employ modern machine learning techniques, such as neural networks, to predict slip occurrence. The predicted slip information is then utilized to determine the desired normal forces required to prevent slippage. Subsequently, each finger is commanded individually to apply the target forces accordingly. However, the development of such algorithms typically requires labeled datasets and is often constrained to specific objects and hand orientations. Moreover, most proposed controllers focus solely on either grasping or releasing objects, lacking the capability to perform both tasks within a unified architecture.

In this chapter, we present a control architecture for robotic hands that incorporates tactile feedback from the fingertips with the conditional postural synergies framework developed in the previous chapter. Our approach enables the execution of both grasping and releasing tasks without relying on slip prediction algorithms. The hand controller takes force readings from the fingertips as input and calculates a target

grasp size, defined as the distance between the thumb's and index finger's fingertips, to achieve the desired applied force. Using the decoder part of the CVAE model introduced in Section 3.1.3, we generate grasp postures, i.e. joint angles, based on the target grasp size and type. These generated grasp postures are then executed by the robot. To prevent object slippage, the desired force is adjusted proportionally to the summation of the tangential contact forces. Essentially, the applied force is controlled using the grasp size as a control variable. Through real-world experiments with a multi-fingered robotic hand, we demonstrate the capabilities of our controller using tactile feedback and high-level properties such as the grasp type and size. Specifically, we showcase the controller's ability to lift objects and maintain a stable grasp even in the presence of external disturbances. Additionally, we demonstrate successful handovers and accurate object placement on surfaces using three precision grasp types.

4.1 Tactile Control

The control of a dexterous multi-fingered hand to achieve stable grasping is typically addressed by formulating it as a problem of position or velocity control for the fingers. The objective is to apply sufficient contact forces to prevent slip between the hand and the grasped object. However, gravity can cause the object to slip if the magnitude of the contact force falls below a certain threshold. This threshold is determined based on the contact modeling framework employed, often utilizing Coulomb's law to model contacts and friction.

According to Coulomb's law, slip does not occur between two surfaces if the product of the normal force f_n acting on the contact surface and the coefficient of friction μ is greater than the tangential force f_t:

$$\mu f_n \geq f_t$$

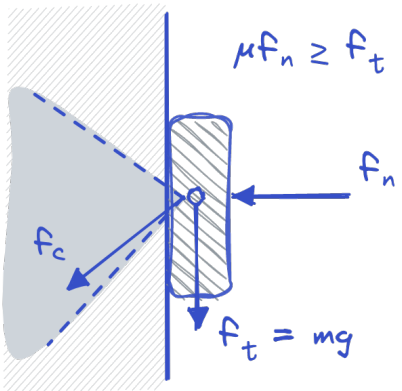

Figure 4.1: Example of modeling the contacts and friction during manipulation.

For example, in Figure 4.1, we observe an object pressed against a wall with an applied normal force f_n, while gravity exerts a tangential force $f_t = mg$. To maintain the stability of the object, the normal force must satisfy the condition:

$$f_n \geq \frac{f_t}{\mu}$$

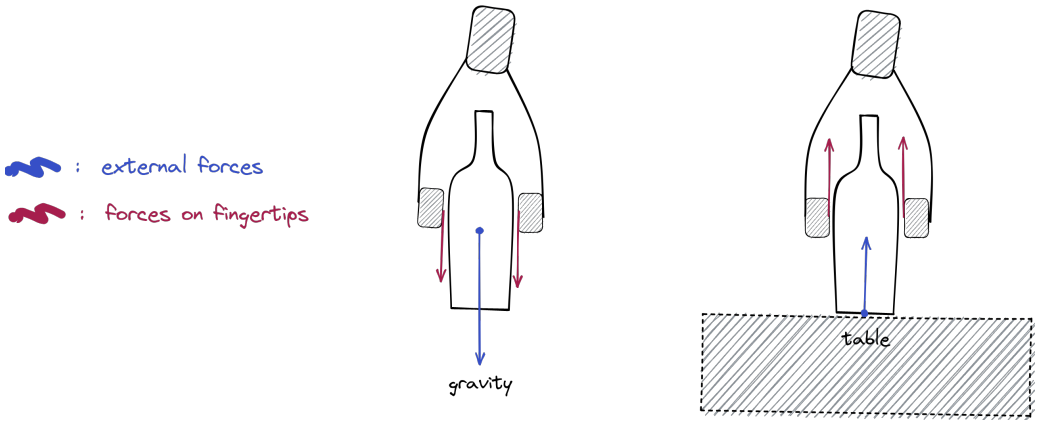

Figure 4.2: Visual representation of the main concept behind our controller.

where μ represents the friction coefficient between the object and the wall.

In the case of a dexterous hand manipulating an object, it is necessary for the normal forces applied by all fingers to exceed the tangential force divided by the friction coefficient between the materials of the object and the fingertip. However, determining the friction coefficients for all possible objects is often impractical. As a result, many existing approaches rely on slip detection. When slip is detected, the controller attempts to increase the applied force measured at the fingertips.

4.2 Tactile Control using Synergies

Next, we outline our proposed control architecture. We begin by assuming that the robotic hand is equipped with force sensors on its fingertips, which provide a 3D force vector representing the forces applied to each fingertip. Additionally, we assume that a CVAE model, capable of modeling the hand's conditional synergies as described in the previous chapter, has been trained and can be conditioned on the desired grasp type and size.

Our primary goal is to design a controller capable of grasping objects in a manner that allows for lifting without slippage, while also being adaptable to external disturbances and capable of releasing the object when necessary. The key idea behind our approach stems from the observation that, when an object is lifted, there is an external force pulling it towards the ground due to gravity, resulting in net forces on the fingertips that are directed downward (as shown in the right part of Figure 4.2). Conversely, when the object is placed on a supporting surface, e.g. a table, there is an external force pushing the object upward (as shown in the left part of Figure 4.2). Building on this assumption, our objective is to design a controller that can switch between two distinct states: grasping and releasing, based on the direction of the external forces.

To accomplish this, we utilize the grasp size variable g_{size} to dynamically adjust the openness or closeness of the grasp as needed. The grasp size represents the distance between the the thumb tip and the index finger tip in a grasp. When the grasp size decreases the normal force applied to the surface of the object increases, since the fingers move closer. The target grasp size, along with the desired grasp type, serves as inputs to the grasp generator, which generates the corresponding grasp posture. The calculation

of the target grasp size is based on the desired normal force f_n^{des}, that we intend the robotic hand to exert on the object, and the currently applied normal force. For instance, to achieve a tighter grasp on the object, we would decrease the grasp size in order to increase the applied normal force. Similarly, to release the object, we would gradually increase the grasp size until no normal force is exerted.

4.2.1 Grasping

In the grasping state, our main goal during lifting and transporting is to avoid object slip. As we saw above to achieve this we want to apply a normal force greater than the tangential force divided by the friction coefficient. Let's imagine a hypothetical scenario, in which we have an object in a fixed position, we have a grasp pose for the hand, and we want to lift the object. Thinking about it step by step, when the hand reaches the target grasp pose we want the hand to start closing its fingers until they come in contact with the objects surface. So, in the beginning the desired normal force takes a small value f_n^{offset}, and the grasp size is reduced until it is achieved and we ensure contact with the object. According to our contact modeling, in order to avoid slip when lifting or holding the object, we want the applied normal force to be equal a gain G times the tangential force f_t. So, the final desired normal force during grasping is the following:

$$f_n^{des} = G \cdot f_t + f_n^{offset} \tag{4.1}$$

where G is the gain that includes the friction coefficient and the additional safety margin as it has been reported in [60] that humans exert an additional force as a safety margin which is proportional to the tangential force: $f_n^{margin} \propto f_t$. We assume that these forces are provided in the reference frame of the fingertip that they are measured.

The applied normal force is controlled using the grasp size variable. Basically, when the applied normal force is lower the desired the grasp size is reduced to tighten the grip, and when it is higher the grasp size is reduced. Because the force readings from the sensors are noisy and can result in oscillations in the grasp size which might cause instability of the object, instead of using the desired normal force as a reference signal, we define a range of values that we want it to be inside. We define a low threshold $f_n^{low} = f_n^{des}$ and a high threshold $f_n^{high} = f_n^{des} + w_{thr}$, where is the width of the threshold. The high threshold helps to avoid the robot applying too much force and potentially damaging the hardware or the object.

After we compute the range for the desired normal force we can calculate if the the actual measured normal force f_n from the fingertip sensors is inside that range and we can compute the difference with respect to the desired range's limits $df_n^{low} = f_n^{low} - f_n$ and $df_n^{high} = f_n^{high} - f_n$. The difference df_n^{low} is greater than zero when the applied force is below the desired one. In this case we want the grasp size to decrease in order to apply greater force, so it decreased proportionally to it. The difference df_n^{high} is greater that zero when the applied force is above the high threshold. In this case we want the grasp size to increase in order to apply less force. So the grasp size g_{size} is calculated using the following equation:

$$g_{size}(n+1) = g_{size}(n) + K \cdot (\mathbb{1}_{df_n^{low}>0} \cdot df_n^{low} - \mathbb{1}_{df_n^{high}>0} \cdot df_n^{high}) \tag{4.2}$$

where K is a parameter that defines the rate of change of the grasp size, i.e. how fast the hand closes and opens, and is determined experimentally. Figure 4.3 shows the change of g_{size} between two time steps, denoted as $dg_{size} = g_{size}(n+1) - g_{size}(n)$, as a function of measured normal force f_n.

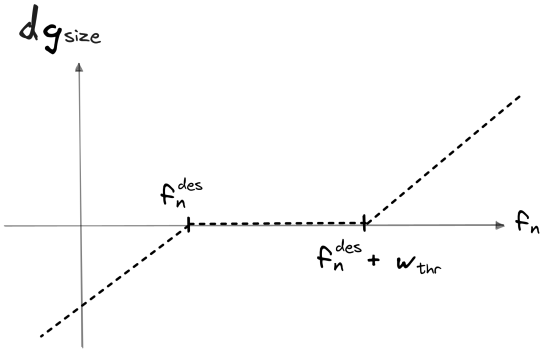

Figure 4.3: The change in target grasp size as a function of the applied normal force. If the applied normal force is below the desired level, the grasp size is reduced to tighten the grasp until the desired force is achieved. Conversely, if the applied normal force exceeds the high threshold, the grasp size increases in order to loosen the grasp.

After calculating the desired grasp size, we apply a saturation function to ensure it falls within the defined bounds:

$$g_{size} = clamp(g_{size}, g_{size}^{min}, g_{size}^{max})$$

This bounded grasp size is then utilized as an input for the grasp generator module, which is implemented using a CVAE model similar to the ones discussed in the previous chapter. The model is trained on a dataset of recorded precision grasp postures, with each grasp labeled with its corresponding grasp size and grasp type. These labels are used as conditional variables in the CVAE model. This approach allows us to control all fingers simultaneously using a single controller. Furthermore, by adjusting the input conditional variable of the model, we can execute various grasp types using the same controller.

4.2.2 Releasing

The controller we have described thus far enables the robotic hand to grasp an object and adjust its posture to prevent slip, utilizing tactile feedback from the fingertips. However, it is also important for the controller to have the capability to release objects when necessary. The robot should release the object when it detects that it is in contact with a support surface, such as a table, or during a handover when someone is pulling the object away. When the object is held in the air, the only external force acting on it is gravity, which pulls it downwards towards the ground. On the other hand, if the object is placed on a support surface, the force of gravity is countered by the support surface, resulting in a net upward external force. Using the tactile reading, the robot can detect the direction of the applied force on the object and determine whether it should release the object or maintain the grasp.

To determine whether the net tangential force on the object is due to gravity or a support surface, we

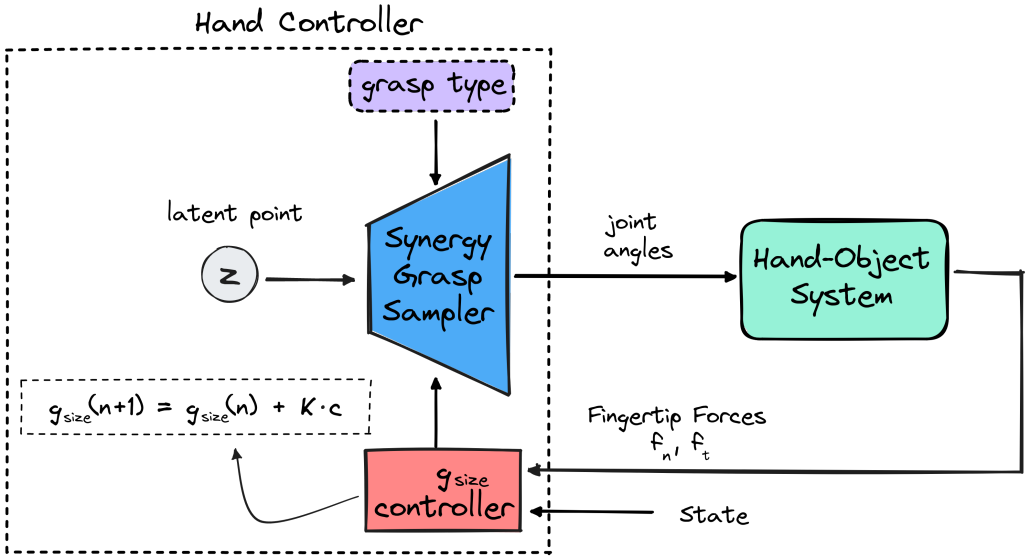

Figure 4.4: Graphical representation of the proposed force controller. The hand controller consists of a grasp posture sampler, modelled using the CVAE model, and a grasp size controller. The grasp size controller takes as input the state (GRASP or RELEASE) and the force readings from the fingertips. Based the target grasp size is computed. The target grasp size along with the desired grasp type are forwarded to the grasp posture sampler. Finally, a grasp posture, represented by the joint angles, is generated and commanded to the robot.

employ a series of steps. Since the force readings from the fingertips are given in the fingertip's coordinate frame, we further transform these readings into the world coordinate frame. This transformation is achieved using the world pose of the hand, obtained from the robotic arm's end-effector. By applying forward kinematics, we can compute the poses of each fingertip in the world reference frame. With this information, we calculate the net tangential force acting on the object. By comparing the direction of this force vector with the direction of gravity, we can determine the hand's current mode of operation. If the angle between the net tangential force vector and the gravity vector is less than 90 degrees, it indicates that the force is directed towards the ground, suggesting that the tangential force is primarily due to gravity pulling the object. In this case, the controller remains in the grasping mode. Conversely, if the angle between the net tangential force vector and the gravity vector exceeds 90 degrees, it implies that the force is directed upwards. This suggests the presence of a support surface pushing the object or someone pulling it from above, triggering the releasing mode of the controller.

To ensure a smooth release of the object, the grasp size controller gradually increases the grasp size variable proportionally to the measured normal force.

$$g_{size}(n+1) = g_{size}(n) + K \cdot f_n \tag{4.3}$$

The control law for adjusting the grasp size during the releasing state is analogous to the one employed in the grasping state (Equation 4.2). However, in this case, the grasp size is incrementally increased to ensure a smooth release of the object.

4.2.3 Final Controller

We can now integrate these two controllers to form a comprehensive control architecture capable of both grasping and releasing objects. Our approach assumes that a trajectory for the robotic arm, consisting of waypoints, is provided to the robot. As the robot follows the trajectory, it enters the *"GRASP"* state once it reaches the target grasp pose, which is included in the trajectory information. At the start, the hand is in an open pre-grasp position. An initial grasp posture is generated by sampling a latent point from the CVAE's prior distribution, together with the desired grasp type and size. The initial grasp size is set to its maximum value. The grasp controller is then activated, gradually reducing the grasp size until the first contact with the object is made. The grasp size is subsequently adjusted to apply the target normal force. Once the robot detects that the tangential force is pointing upwards, indicating the need to release the object, the robot enters the *"RELEASE"* state. It ceases further movement and the hand initiates the release process by increasing the grasp size, thereby opening the grasp. A schematic representation of the final control architecture can be seen in Figure 4.4.

4.3 Experiments and Results

4.3.1 Experimental Set-up

In our experiments, we utilized the Seed Robotics RH8D Hand [61], a human-like robotic hand featuring 7 degrees of freedom (DoFs), show in Figure 4.5. Each fingertip of the hand is equipped with FTS-3 force sensors [62], which are high-resolution tactile sensors that provide the 3D force applied in the fingertip's coordinate frame. The sensors provide data at a rate of 50Hz. To facilitate the experimentation, we mounted the hand on a Kinova Gen3 7DoF robotic arm.

Figure 4.5: The humanoid Seed Robotics 8DOF Hand

To train the grasp posture generator, we followed the same procedure outlined in Section 3.2. Three different operators teleoperated the hand, using the CyberGlove, and collected a dataset consisting of 468 grasps across three precision grasp types: tripod, pinch, and lateral tripod. The grasps were executed

on the objects in Figure 3.8. We trained a CVAE model, conditioned on both the grasp type and the grasp size. The experiments for testing our force controller were executed using 10 household objects, as depicted in Figure 4.6, with weights ranging from 1g to 380g. During the experiments, the arm trajectories were pre-recorded, while the hand was controlled online using the proposed control algorithm.

Figure 4.6: The household objects used to assess the capabilities of the proposed force controller.

4.3.2 Parameter tuning.

In order to determine the parameters for our controller, we conducted preliminary experiments involving the lifting and releasing of various objects with different physical properties.

To determine the value of the offset normal force f_n^{offset}, which represents the minimum force applied to the object, we used an empty plastic cup as a test object. Our goal was to select a value that would prevent the fingers from deforming the cup. After experimentation, we determined the final value for this parameter to be 50 mN.

To determine the values for the gain G and rate of change K of the grasp size, we focused on the heaviest object in our dataset, which was a mustard bottle weighing 380g. The gain G was set to 2.0 to ensure that the desired normal force would be sufficient to securely hold the object. The rate of change K of the grasp size was set to 100.0, taking into consideration the operating frequency of the force sensor and the range of tangential force values.

To mitigate the effect of noise in the measurements of the force sensors, we implemented a running average as a low-pass filter for the measured forces. The filter equation is given by:

$$f'(n) = \alpha f(n) + (1 - \alpha) f'(n - 1)$$

where $f(n)$ is the current measurement, $f'(n)$ is the filtered measurement, and $\alpha \in (0, 1)$ is a parameter that determines the extent to which new measurements contribute to the filtered value. The specific value of α for each force was selected through experimental evaluation.

For the tangential force, we applied the filter to the sum of the x and y component forces measured

in the fingertips. To achieve an appropriate trade-off between responsiveness and noise reduction, we set the averaging parameter $\alpha = 0.7$. This value allows the controller to be sensitive to rapid changes in the tangential force, which may occur during object lifting.

Regarding the normal force, i.e. the z component, we used a parameter value of $\alpha = 0.5$. This choice ensures that the filtering is not unduly influenced by noise, thereby preventing the controller from becoming overconfident in its estimations.

4.3.3 Experiments.

To demonstrate the capabilities of our controller, we conducted a series of four experiments, each increasing in complexity:

- Simple pick and place: We successfully executed a pick and place task using a tripod grasp to grasp and manipulate a bottle. The robot effectively lifted the bottle and accurately placed it at the desired location.
- Pick and place with rotation: Employing a tripod grasp, we picked up a chips can, rotated it, and placed it on top of a box. The controller adeptly adjusted the grasp size throughout the hand rotation, ensuring a secure hold.
- Handover task: This experiment involved picking up a chips can, rotating it, and handing it over to a person while maintaining a tripod grasp.
- Pinch grasp: To highlight the versatility of our controller, we performed a manipulation task using a pinch grasp. The robot picked up a brown foam brick, rotated it, and handed it over to a person.

Furthermore, to assess the controller's capabilities in scenarios where the object's mass changes during execution, we conducted an additional experiment. In this experiment, a person handed a plastic cup to the robot, which was then filled with metal coins to increase its weight. Finally, the robot handed the cup back to the person, employing a tripod grasp.

Figure 4.7 showcases the execution of our first experiment. Alongside the images, we provide recorded signals from the controller for analysis. These signals include: the average normal force applied by all fingers (blue line), the upper and lower thresholds for the normal force (indicated by purple and yellow dashed lines respectively), which are functions of the tangential force, the average tangential force (green line), and the grasp size used to generate the grasp posture in each timestep (red line).

1. Initial grasp: When the hand reaches the target grasp pose, the controller begins by closing the grasp while the object is on the box (indicated by the red section). During this stage, the force controller gradually reduces the grasp size until the applied normal force remains below the offset value, f_n^{offset}.
2. Lifting phase: As the robot attempts to lift the object, the tangential force increases, subsequently adjusting the desired normal force and the threshold (green section). The controller responds by decreasing the grasp size to apply more normal force and maintain a secure grip.

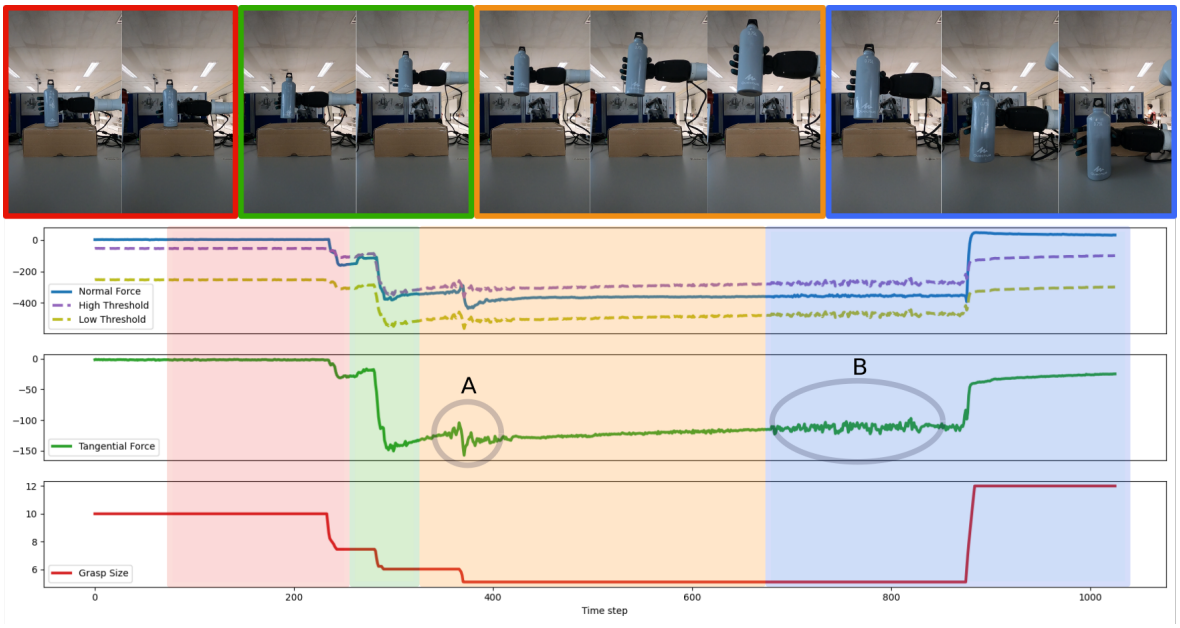

Figure 4.7: In our first experiment, the robot grasps and lifts a bottle, transports it, and places it on the desk. The bottom part of the figure shows see the control signals recorded during this task.

3. Object transportation: During this phase (indicated by the orange section), the robot transports the object. Notably, at point A in the figure, a perturbation in the tangential force occurs when the robot initiates the movement. The controller promptly reacts by decreasing the grasp size, thus enhancing the stability of the object.

4. Release phase: In the final stage (blue section), the arm gradually lowers until the tangential force indicates contact with a support surface. At this point, the robot enters the releasing phase. The arm stops descending, and the controller slowly increases the grasp size, facilitating the controlled release of the object.

In point B of the figure, slight noise is visible in the tangential force, that is also reflected in the desired normal force, likely caused by the arm's movement during the object's placement on the table. However, since we use a desired range for the target normal force rather than a reference signal, this noise does not affect the control of the grasp size.

The execution of the second experiment is depicted in the upper section of Figure 4.8. This experiment demonstrates the controller's capability to handle arbitrary hand poses. The experiment consists of four distinct stages:

1. Object grasping: The robot enters the *GRASP* phase, and the force controller generates grasps to achieve a normal contact force below the predefined threshold, f_n^{offset}.

2. Object lifting: The robot lifts the object while continuously adjusting the grasp size to prevent any potential object slippage.

3. Object rotation: The hand rotates to position the chips can horizontally.

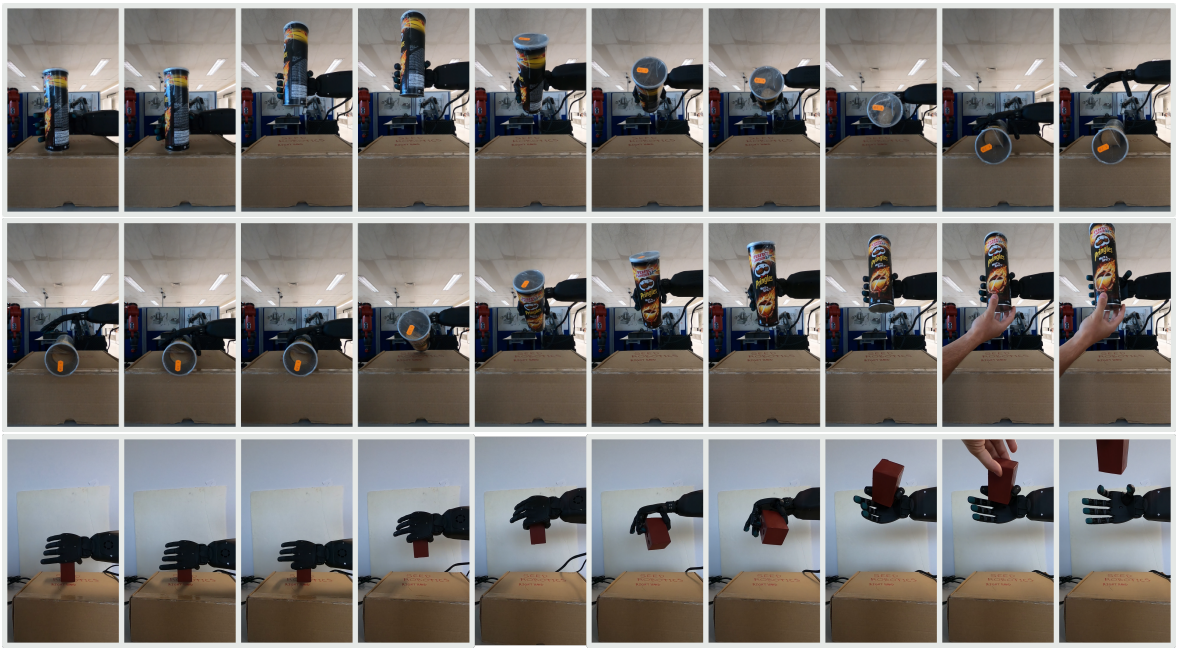

Figure 4.8: In the upper row of images, we present our second experiment where the robot successfully picks up a chips can, rotates it by 90 degrees, and places it back down. Moving to the middle row, our third experiment showcases the robot picking up the chips can, rotating it by 90 degrees, and handing it over to a person. Finally, in the bottom row, our fourth experiment demonstrates the robot picking up a foam brick, rotating it by 180 degrees, and handing it over to a person using a pinch grasp.

4. Object placement: The robotic arm gradually lowers the object. When the hand detects contact with the supporting surface (the box), it enters the *RELEASE* phase and initiates a controlled and gradual release of the object.

Through this experiment, we showcase the controller's ability to handle arbitrary hand poses, maintain grasp stability during object rotation, and ensure a controlled release process.

The execution of the third experiment is illustrated in the middle part of Figure 4.8. This experiment highlights the controller's capability to execute robot-to-human handovers. The experiment is divided into four key stages:

1. Grasping phase: The robot enters the *GRASP* phase, and the force controller generates grasps to achieve a normal contact force below the predefined threshold, f_n^{offset}.

2. Lifting phase: The robot lifts the object while continuously adjusting the grasp size to maintain a secure hold and prevent any potential object slippage.

3. Object rotation: The hand rotates to position the chips can vertically.

4. Robot-to-human handover: During the *RELEASE* phase, the arm remains still as the human grasps the object from the bottom and gently pushes it upward. The hand detects the presence of a supporting surface and gradually releases the object to enable a smooth handover.

The execution of the fourth experiment can be seen in the lower part of Figure 4.8. This experiment shares a similar structure with the previous one, but with a notable distinction in the grasp type employed.

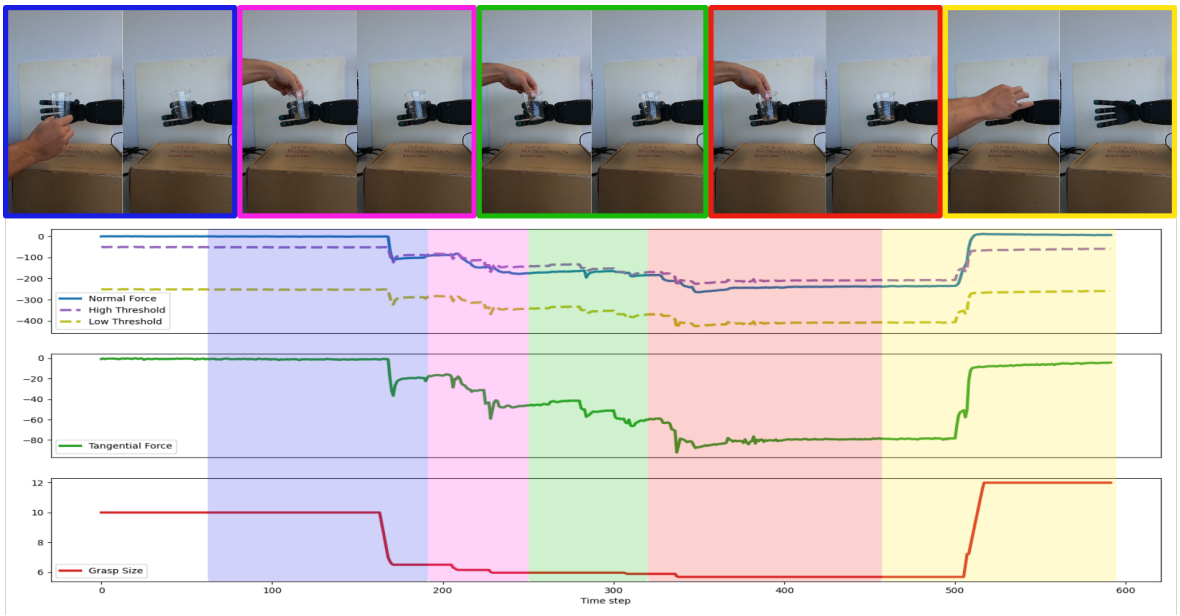

Figure 4.9: In our fifth experiment, we executed a task where a person handed over an empty plastic cup to the robot. The person then proceeded to throw coins into the cup, gradually increasing its weight. During this process, the robot reactively adjusted its grip to stabilize the object and maintain a secure grasp. Finally, the robot successfully handed the cup back to the person, demonstrating its ability to adapt and interact with objects of varying weights.

In this case, the robot utilizes a pinch grasp, involving only the thumb and index finger. To achieve this, we simply modified the grasp type conditional variable provided to the posture sampling model, maintaining the same underlying control architecture and principles.

The execution of the final experiment can be seen in Figure 4.9. In the initial stage (blue), the robot closes its grasp by reducing the grasp size until the normal force falls below the force offset threshold. In the subsequent parts (pink, green, red), the person adds coins to the cup, progressively increasing its weight. As depicted in the signal plots, each time coins are added, the tangential force decreases, causing the normal force threshold to decrease accordingly. Consequently, the grasp size is reduced to apply more normal force. This experiment effectively demonstrates the controller's capability to adapt to perturbations in the weight of the object during the grasping process.

4.4 Conclusions

In this chapter, we have developed a controller specifically designed for humanoid robotic hands equipped with force sensors. Our controller leverages the postural synergies framework to facilitate the execution of grasping and releasing tasks using several precision grasp types. By utilizing feedback from the fingertips' sensors, our force controller determines a target grasp size, which is then used by the conditional synergies decoder to generate the appropriate joint angles for the hand. Through a series of experiments, we have demonstrated the effectiveness of our controller in lifting objects of different weights and materials while effectively preventing slip. Furthermore, our controller showcases real-time adaptability to changes in object weight, enables accurate object placement on surfaces, and facilitates

successful robot-to-human handovers. However, that our current controller does not account for rotational forces applied to the object. Since the force sensors provide only the 3D forces applied on the robot's fingertips we plan to integrate additional sensing modalities, such as vision, to address this limitation. This could be achieved by tracking the pose of the object and detecting sudden pose changes that might be due to rotational forces.

Chapter 5

Sampling 6DoF Grasp Poses for Multi-fingered Hands

In the previous chapters, we explored the representation and generation of grasp postures based on high-level properties exploiting the postural synergies framework. Additionally, we demonstrated the utilization of tactile feedback for controlling the aperture of a humanoid robotic hand to grasp and release objects. However, our focus has primarily been on finger control, assuming that the object has been grasped already. To determine the stable grasp of an object, we have to obtain the hand pose and finger configuration. This involves providing the robot with both a hand pose, which specifies the position and orientation of the end-effector, and a grasp posture, which entails determining the joint angles of the fingers. Subsequently, the robot can proceed by first reaching the designated hand pose and subsequently executing the grasp posture, closing the fingers until they firmly grasp the object.

In this chapter we explore how to sample hand poses with respect to objects, i.e. hand positions and rotations, in order to execute several precision grasps. We develop a model that factors the grasp sampling process in a set of cascaded steps. The proposed model first generates a grasp posture, using the postural synergies framework developed in Sections 3.1.2 and 3.1.3, and then samples a position and an orientation for the wrist of the hand. This way we achieve greater grasp success than sampling the entire grasp in a concurrent manner. We also present a geometric heuristic to generate candidate grasp postures for precision grasp types. By employing the heuristic, we collect a dataset of successful grasp postures that we use as our training samples for the cascaded hand pose generation network. In our experiments, we show the improvement in the percentage of successful grasps generated by the model compared to the grasps computed using the heuristic. We also compare different variants of the model to show how our design choices affect the performance in grasp sampling.

5.1 6DoF Grasp Pose Sampling

We approach the grasp sampling problem by considering a scenario where a free-floating object is positioned at the origin of the coordinate axis. Our objective is to generate a grasp, consisting of a hand's

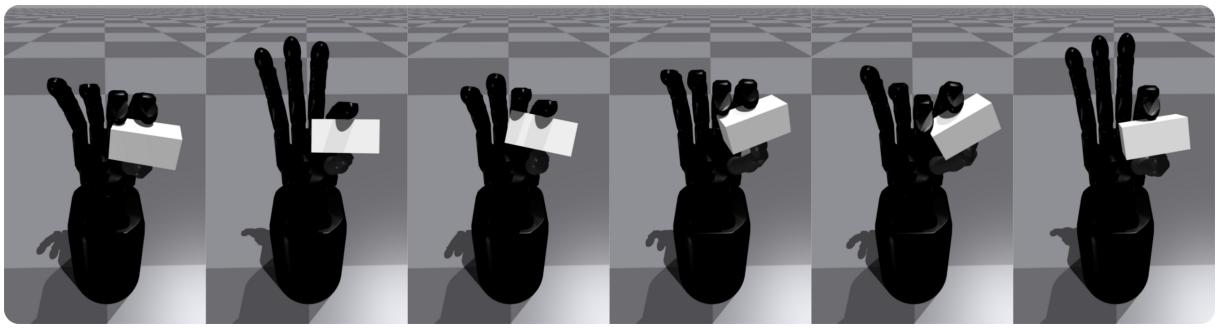

Figure 5.1: Example grasps for each grasp type used in this work, executed with the Shadow Robot Hand. From left to right: tripod, palmar pinch, parallel extension, writing tripod, lateral tripod, and tip pinch.

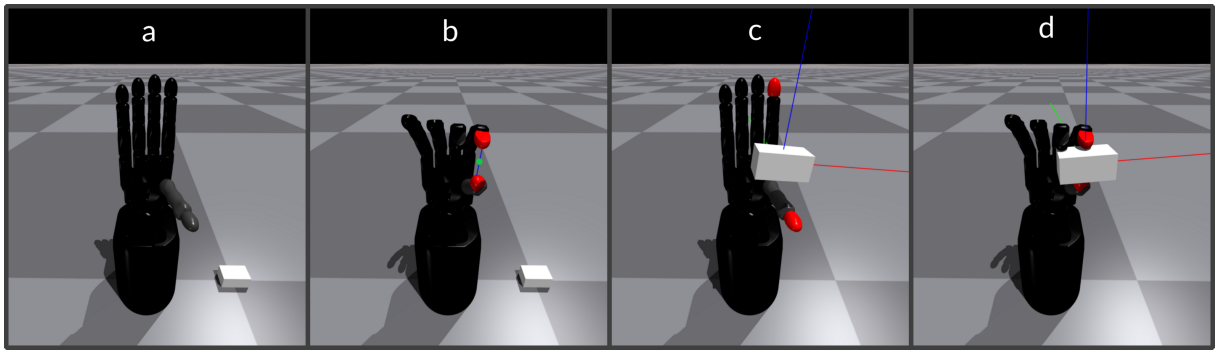

Figure 5.2: Example for the process of generating a candidate grasp. **a**) The initial posture of the hand before grasping. **b**) The tripod grasp type is selected and a grasp posture is sampled from the initial cVAE model. The rigid bodies for the corresponding opposition joints can be seen in red. The blue line connecting them is the opposition axis and the green point is the middle point where the object will be placed. The grasp size recorded is the length of the blue line. **c**) The object is placed in the calculated position and the blue axis of the object is aligned with the opposition axis. **d**) The grasp is executed. After that the hand performs a shaking movement and gravity is activated. If the object remains in the hand the grasp is considered successful. The object size recorded is the current distance between the rigid bodies of the thumb and the index tip.

6DoF pose and the finger's joint angles, with respect to the object's coordinate frame. Additionally, in our approach we assume that the generation of the grasp posture precedes determining the grasp pose. This is because a single grasp posture can be executed from multiple grasp poses, allowing for various directions or orientations on the same object, often owing to symmetries.

Our aim is to model the distribution of successful grasps given the object's category and size. To tackle this task, we adopt a data-driven approach, leveraging a large dataset of successful examples to train our model. To gather the necessary data, we propose a simple geometric heuristic that generates candidate grasp poses based on a sampled grasp posture. We evaluate these generated grasps using a physics-based simulation and record the successful instances. This process enables us to collect the required training data. Finally, we train the three proposed models on the acquired dataset to simulate the distribution of successful grasps. This approach allows us to capture the patterns and characteristics of successful grasps and generate new ones.

5.1.1 Data generation process.

Our proposed data generation process consists of three key steps: (1) generating candidate grasp postures belonging to specific target grasp types, (2) using a geometric heuristic to compute candidate grasp poses for each grasp posture, (3) executing the grasps in a simulated environment and collecting the successful ones.

To generate the candidate grasp postures, we utilize a CVAE model, similar to the one presented in Chapter 3. Initially, we assume that we already have a small dataset consisting of ≈ 500 grasps, labeled with grasp type information. Using this dataset we train the initial CVAE model which we use to generate the candidate grasps postures. This model is exclusively employed during the data generation process and is subsequently discarded. For grasp generation, the model is conditioned on one of the following six precision grasp types, derived from the grasp taxonomy proposed in [59]: tripod, palmar pinch, parallel extension, writing tripod, lateral tripod, and tip pinch. In order to generate the grasp postures we sample latent points from the model's prior distribution and feed them to the decoder along with the desired grasp type. After we sample the grasp postures using this model we have to compute a candidate grasp pose for each posture.

The next step involves determining a candidate grasp pose, which we break down into two phases: calculating the 3D position of the object and determining its rotation. During the data generation process, rather than generating a candidate grasp pose as the position and rotation of the hand, we adopt a different approach. We assume that the hand remains fixed at the origin of the coordinate axis, and we generate a candidate pose for the object within the hand's reference frame. Since our focus is on precision grasp types, we can leverage their inherent structure to generate candidate grasp poses for the object. Precision grasps rely on the opposition between specific finger joints, known as opposition joints, to achieve stability. In particular, the thumb and index fingers are commonly used to create what is known as pad opposition, as described in [63]. The axis connecting these finger segments is referred to as the opposition axis. For each grasp posture, we have a pair of opposition joints that emerges from the grasp type's structure, as well as an opposition axis that depends on the geometric arrangement of the physical finger segments responsible for the opposition. In the case of tripod, writing tripod, lateral tripod, and tip pinch grasp types, the opposition is formed between the thumb tip and the index tip. On the other hand, for palmar pinch and parallel extension grasps, the opposition occurs between the thumb tip and the index middle segment.

Based on the selected opposition joints for the given grasp type, we employ forward kinematics to determine the Cartesian positions of the rigid bodies associated with these joints during the execution of the grasp. By averaging the positions of the two opposition rigid bodies, we compute the position of the object, as it is often observed that humans tend to grasp objects near their center of mass to enhance stability [64, 65]. Furthermore, to establish the rotation of the object, we align one of the three canonical axes of the object with the opposition axis. An example of the candidate grasp generation process can be observed in Figure 5.2.

During the third step, the object is positioned according to the calculated pose, and the grasp is executed. To assess the stability of the grasp, the hand performs a shaking movement, moving the wrist

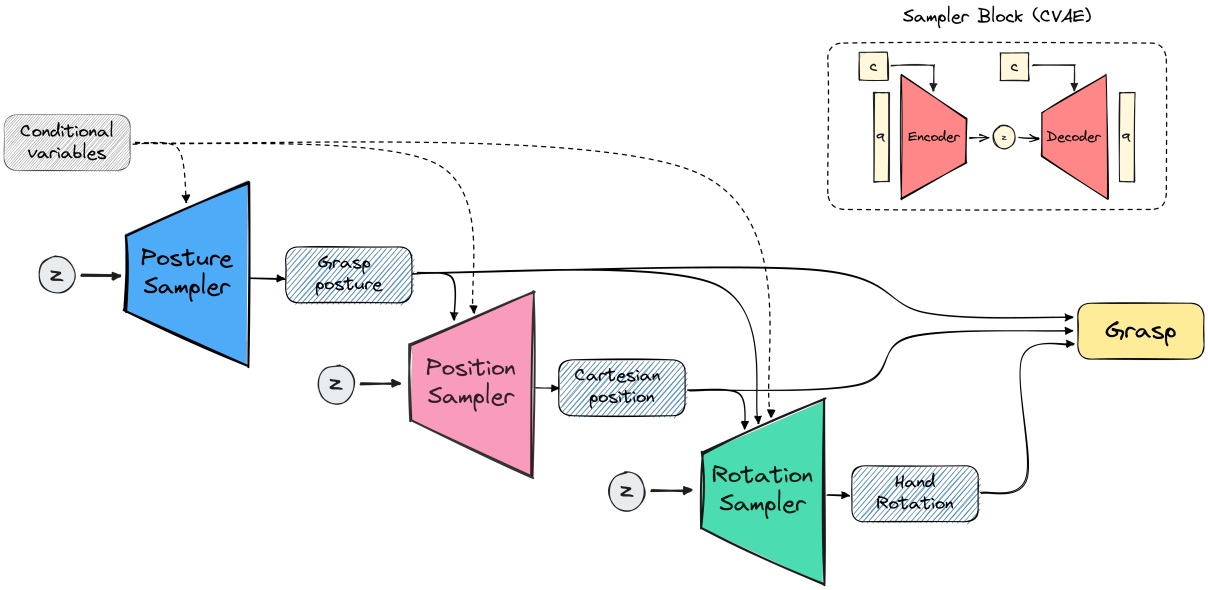

Figure 5.3: Graphical representation of the Grasp Sampling procedure. The model consists of three individual samplers: the Postural Sampler that generates precision grasp postures (finger joint angles), the Positional Sampler that generates the object's Cartesian position for a given grasp posture, and the Rotational Sampler that generates the object's rotation for a given grasp posture and object position.

from left to right and forwards and backwards, to identify any unstable grasp poses. Subsequently, gravity is activated in the environment. If the object remains in the hand for a duration of 5 seconds after gravity is activated, the grasp is considered successful. At this point, the grasp type, grasp posture, grasp size, object pose, object type, and object size are recorded. The grasp size is computed as the distance (in cm) between the thumb tip and the index tip when executing the grasp without an object. On the other hand, the object size is determined as the distance between the thumb tip and the index tip when executing the grasp while holding the object. By following this procedure, we collect an extensive dataset consisting of successful dexterous grasp postures executed on each object using various grasp types.

5.1.2 Grasp sampler.

The main objective of this work is to develop a generative model capable of sampling dexterous precision grasps given the pose, type, and size of an object. Specifically, we aim to learn a sampler for the distribution of successful grasps, denoted as:

$$P(G \mid G_t, G_s, O_t, O_s),$$

where $G = (G_c, G_{pos}, G_{rot})$ represents a successful grasp, consisting of the grasp configuration G_c, represented the finger joint angles, the three-dimensional position of the hand G_{pos}, and the rotation G_{rot} for the hand base. The sampler is conditioned on the properties of the desired grasp, including the type of the grasp G_t and the size of the grasp G_s, as well as on the properties of the target object, such as the type of the object O_t, and the size of the object O_s.

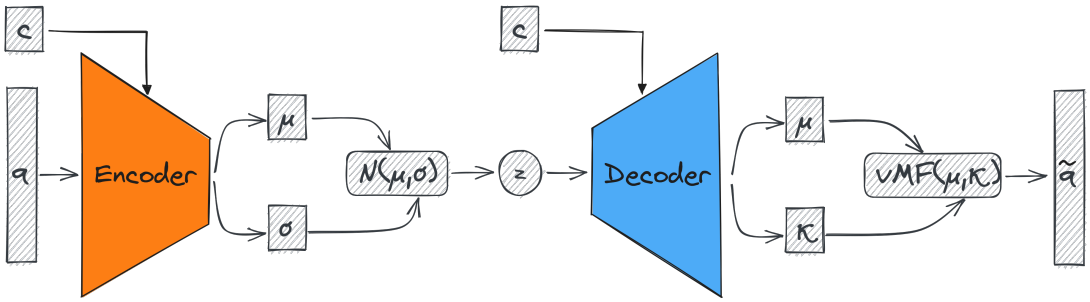

Figure 5.4: Graphical representation of rotation sampler model.

In contrast to previous works that directly modeled the full distribution:

$$P(G \mid G_t, G_s, O_t, O_s) = P(G_c, G_{pos}, G_{rot} \mid G_t, G_s, O_t, O_s),$$

using a single cVAE model, we adopt a different approach by factorising the distribution in its three components. Specifically, we train three individual samplers: the Posture Sampler, the Position Sampler, and the Rotation Sampler, each employing a cVAE architecture. Our experimental results demonstrate that this approach yields a model with superior performance.

The Posture Sampler models the probability distribution:

$$P(G_c \mid G_t, G_s, O_t),$$

and is conditioned on the grasp type label, grasp size, and object type. It is trained on successful grasp postures executed on the objects, building upon the model for learning conditional postural synergies presented in Chapter 3. The key distinction is that we incorporate grasp type information as a conditional variable in the model, enabling the generation of specific grasp types.

The Position Sampler models the probability distribution:

$$P(G_{pos} \mid G_c, G_t, G_s, O_t, O_s)$$

and is conditioned on the grasp posture, grasp type label, grasp size, object type, and object size. It is trained on the 3D Cartesian positions of the objects.

Finally, the Rotation Sampler models the probability distribution:

$$P(G_{rot} \mid G_{pos}, G_c, G_t, G_s, O_t, O_s)$$

and is conditioned on the grasp posture, grasp type label, grasp size, object type, object size, and Cartesian position. It is trained on the rotations of the objects, represented by quaternions. The Rotation Sampler generates the mean direction and concentration parameter of a von Mises-Fisher distribution, which is then used to sample a rotation [66]. The graphical representation of this model can be seen in Figure 5.4.

To obtain grasp samples from the model, first we sample a latent point from the prior distribution of the Posture sampler and along with the corresponding conditional variables we feed them to the

decoder of the model to generate a grasp posture. Then we perform the same procedure for the Position and Rotation samplers, each time adding the output of the previous models to the conditional variables. Finally, the generated grasp posture, hand position, and hand rotation form a complete grasp. A graphical representation of the entire model can be seen in Figure 5.3. In the experimental section we present more details of how each model is trained. The generated grasp pose is in the reference frame of the hand. However, in order to apply the grasp to an object we would need to have the grasp in the reference frame of the object. Given the object's world pose, that we assume is provided by a perception system, can transform the generated grasp to the reference frame of the world frame.

5.2 Experiments and Results

5.2.1 Experimental Set-up.

In our experiments, we utilized the IsaacGym simulator, with the PhysX physics engine [67]. This GPU-accelerated simulator enables parallel execution of multiple environments. To collect our dataset, we ran 500 environments simultaneously, resulting in a total of 270,000 generated grasps. The entire data generation process took approximately 10 hours on an NVIDIA GeForce RTX 3070. The training time for our models averages around 20 minutes, and it only takes 0.05 seconds to sample 1000 grasps. For the robotic hand, we employed the Shadow Robot Hand, which consists of 24 joints. As for the objects, we used a selection of three boxes, three cylinders, and three spheres, each with different sizes, as depicted in Figure 5.5. The initial CVAE model, used in the dataset generation process, was trained on the same dataset as in Chapter 3. In this case, the model was conditioned only on the grasp type information. This model was used exclusively in this stage and it was later discarded.

To generate our dataset, we produced 5000 grasps for each object type and each grasp type, resulting in a total of 270,000 grasps. For each grasp, we recorded the grasp posture (joint angles in degrees), the grasp type, the grasp size (in centimeters), the object type, the object side size (in centimeters), as well as the position and rotation of the object. The grasp size is defined as the Euclidean distance between the thumb tip and the index tip when executing the grasp without an object in the hand. Similarly, the object size is defined as the Euclidean distance between the thumb tip and the index tip when executing the grasp with an object in the hand.

5.2.2 Grasp Sampling Procedure

In order to sample new grasps, during test-time, we provide the model with the following information: (1) the desired grasp type, which is encoded as a six-dimensional one-hot vector, (2) the object type, also represented as a nine-dimensional one-hot vector, (3) the size s, of the specific side of the object that we intend to grasp. Using the provided object size s, we calculate the grasp size g_s, and object size o_s as follows:

$$g_s = s - 0.5(\text{cm}),$$

$$o_s = s + 1.0(\text{cm})$$

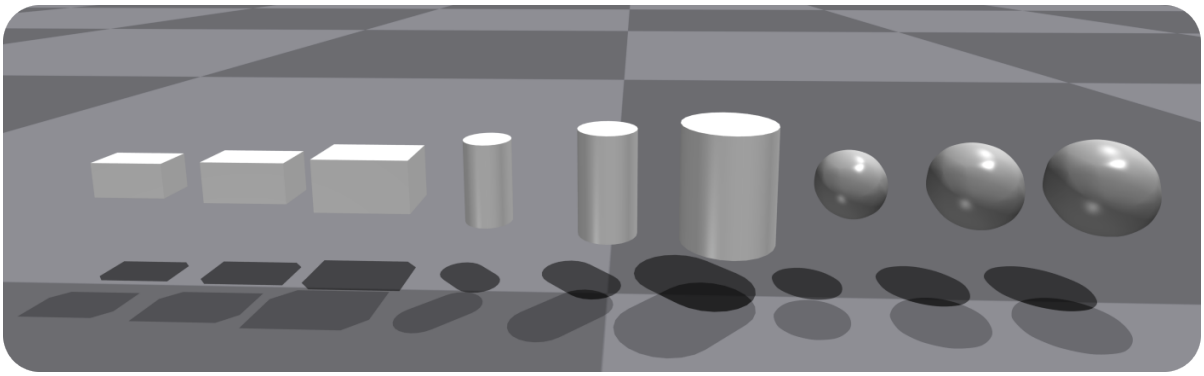

Figure 5.5: The objects use in our experiments.

This adjustment is necessary because when recording the object size, as described in Section 5.1, we measure the Euclidean distance between the thumb tip and the index tip, which includes the size of the simulated robotic hand's rigid bodies. By reducing the grasp size by $0.5cm$, we aim to ensure a firm and stable grasp on the object, considering the absence of force feedback in our setup.

Our model then generates: 1) the joint angles of the robotic hand, which are represented in degrees, 2) the position of the object with respect to the base of the hand, which is in centimeters and normalized to lie inside a unit sphere centered in the origin, 3) the rotation of the object, which is represented by a unit quaternion. Given the actual pose of the object in the world frame, we can compute the rigid transform of the hand such that the relative transform between the hand and the object remains fixed.

5.2.3 Model design and evaluation metric.

The primary evaluation metric we employed to compare our models is the percentage of successful grasps sampled and executed. To establish a baseline, we utilized the heuristic described in Section 5.1, which generates object positions and rotations based on the opposition joints of each grasp. To assess the performance of our grasp sampler models, we present various variants and compare them to support our design decisions leading to the development of our final model, which demonstrates the highest performance. In order to calculate the success rate for each model, we sampled 500 grasps for each object and grasp type, resulting in a total of 27,000 grasps. These grasps were then executed, followed by hand shaking, gravity activation, and computation of the percentage of successful grasps. We present the success rates of each model per object type, along with the average grasp success rate across all objects. These results provide insights into the effectiveness of our models and allow us to compare the performance of each design choice.

5.2.4 Quantitative Results.

In our first experiment, we aimed to examine the benefits of employing separate cVAE models for generating grasp postures, positions, and rotations. This approach was compared against using a single model to generate all modalities, as well as using two models, one for grasp postures and another for poses (position and rotation). Table 5.1 shows the different distributions modelled in each model instance.

Table 5.1: Distributions modelled within each model instance.

	Distributions
Model 1	$P(G_c, G_{pos}, G_{rot}, \|G_t, G_s, O_t, O_s)$
Model 2	$P(G_c\|G_t, G_s, O_t, O_s)$ $P(G_{pos}, G_{rot}, \|G_t, G_s, O_t, O_s)$
Model 3	$P(G_c\|G_t, G_s, O_t, O_s)$ $P(G_{pos}\|G_c, G_t, G_s, O_t, O_s)$ $P(G_{rot}\|G_c, G_{pos}, G_t, G_s, O_t, O_s)$

Table 5.2: Average percentage of successful grasps generated by each model.

Model	**Success Rate**
Model 1	76.89
Model 2	79.41
Model 3	**90.90**
Heuristic	58.10

Model 1 uses one CVAE to represent the entire grasp, Model 2 uses one CVAE for the grasp posture distribution, and one for the pose (position and rotation) distribution, Model 3 uses one CVAE for the grasp posture distribution, one for the position distribution and one for the rotation distribution. All models were provided with the same inputs: the grasp type, grasp size, object type, and object side size. The results, depicting the average grasp success rate for each model, can be found in Table 5.2, including the results achieved by the geometric heuristic presented in Section 5.1. It is evident that the model utilizing separate CVAE models for each modality outperforms all other models. These results indicate that the approach of factorizing the grasp sampling process and independently modeling each distribution better captures the characteristics of our data.

In our second experiment, we investigated the effectiveness of using the von Mises-Fisher (vMF) distribution in the output layer of the rotation sampler network compared to utilizing a non-linear layer and a normalization operation to transform the vector into a unit vector. The vMF distribution allows to sample points on 4D spheres, which is a natural representation for quaternions. The average grasp success rates for each model can be seen in Table 5.3. It is evident that the model utilizing the vMF distribution to sample rotations outperforms the model with the neural network output layer.

Table 5.3: Average percentage of successful grasps generated by each model.

Model	**Success Rate**
Model with NN	86.57
Model with vMF	**90.90**

Table 5.4: Conditional variables used for each of the Position and Rotation samplers for each model variant.

	Position Sampler	**Rotation Sampler**
Variant 1	G_c, G_t, G_s, O_t, O_s	$G_{pos}, G_c, G_t, G_s, O_t, O_s$
Variant 2	G_c, G_t, G_s, O_t, O_s	$G_{pos}, G_c, G_t, O_t, O_s$
Variant 3	G_c, G_t, G_s, O_t	$G_{pos}, G_c, G_t, O_t, O_s$
Variant 4	G_c, G_t, O_t, O_s	$G_{pos}, G_c, G_t, O_t, O_s$
Variant 5	G_c, G_t, O_t	G_{pos}, G_c, G_t, O_t
Variant 6	G_c, G_s, O_t, O_s	$G_{pos}, G_c, G_s, O_t, O_s$
Variant 7	G_c, G_t, G_s, O_s	$G_{pos}, G_c, G_t, G_s, O_s$

Finally, we conducted an investigation into the impact of each conditional variable on the model's performance. For this purpose, we trained the model with the three cVAE samplers using different subsets of conditional variables for the Position and Rotation Samplers. The grasp posture sampler remained the same across all model variants and was conditioned on the grasp type, grasp size, and object type. Table 5.4 presents the conditional variables used for the Position and Rotation Sampler in each model variant. The average grasp success rates for each model can be found in Table 5.5.

Model variants 1-4 demonstrate the effect of different combinations involving the size variables. Notably, the results are similar across all models, indicating that the size information provided to the model, whether it is the grasp size or the object size, does not significantly impact performance. This can be attributed to the linear dependence between these two values. In model variant 5, when the size information is entirely removed from both models, the performance declines. Model variant 6 reveals that the exclusion of grasp type information has minimal effect on the model's performance. This suggests that grasp position is primarily influenced by the finger tips encoded in the grasp posture variables, while rotation is connected to the size of the object's side being grasped. Model variant 7, which lacks any information about the object type, performs the poorest in terms of average successful sampled grasps, underscoring the importance of this parameter. Additionally, it is worth noting that all models perform similarly on the sphere objects due to their complete symmetry, where object rotation has no impact on the results. Nonetheless, model variant 4, which employs grasp and object type information, along with the object size as conditional variables, exhibits the best overall performance.

5.2.5 Qualitative Results.

Figure 5.6 illustrates the 6DoF grasp poses obtained during the data generation process for the medium-sized box across various grasp types. It is evident that each grasp type exhibits a distinct distribution of poses, which arises from the interplay between the object's shape and the characteristics of the grasp type.

Figure 5.7 showcases a collection of grasps generated by our model, featuring different objects from

Table 5.5: Average percentage of successful grasps generated by each model.

Model	Success Rate
Variant 1	90.90
Variant 2	90.57
Variant 3	90.36
Variant 4	**91.54**
Variant 5	81.68
Variant 6	90.78
Variant 7	71.77

the dataset and employing various grasp types. The results indicate that our model can be employed in practical grasping tasks such as object pick-up. In such cases, additional steps would be necessary to ensure collision avoidance by verifying the sampled grasp pose and planning the end-effector's trajectory to reach the intended pose. These tasks can be effectively addressed using trajectory optimization techniques.

Finally, we tested the model using common household objects from the YCB dataset, which typically feature regular geometric shapes such as cubes, spheres, and cylinders, or can be approximated as such. To assess the model's ability to generalize, we selected objects from the dataset that were most similar to the trained categories and adjusted their sizes to match. We then randomly selected several precision grasps for each object and executed them, lifting the objects to verify that the grasps were stable with gravity activated. This evaluation demonstrates the model's capacity to operate effectively on unseen objects with similar shapes to those used during training, highlighting its adaptability to real-world objects not present in the original dataset. Images of the grasp attempts and their outcomes are provided in Figure 5.8, illustrating the model's performance across different object types.

5.3 Conclusions

In conclusion, we have developed a sequential model for generating grasps in multi-fingered robotic hands, specifically targeting six precision grasp types. Our model employs three cascaded samplers: one for finger configuration, one for hand position, and one for hand rotation. Each sampler is implemented using a CVAE, allowing us to generate multiple grasp poses for each finger configuration. Additionally, we introduced a geometric heuristic for calculating candidate grasp poses tailored to each precision grasp type, which served as the basis for our dataset collection. Through our experiments, we have demonstrated the advantages of the cascaded sampling approach and validated our design choices based on quantitative results. It is important to note that our work assumes the applicability of all grasp types to every object. However, in reality, humans consider task-specific information to determine the appropriate grasp type. As part of our future work, we plan to incorporate task-related information into our model, enabling more context-aware grasp generation.

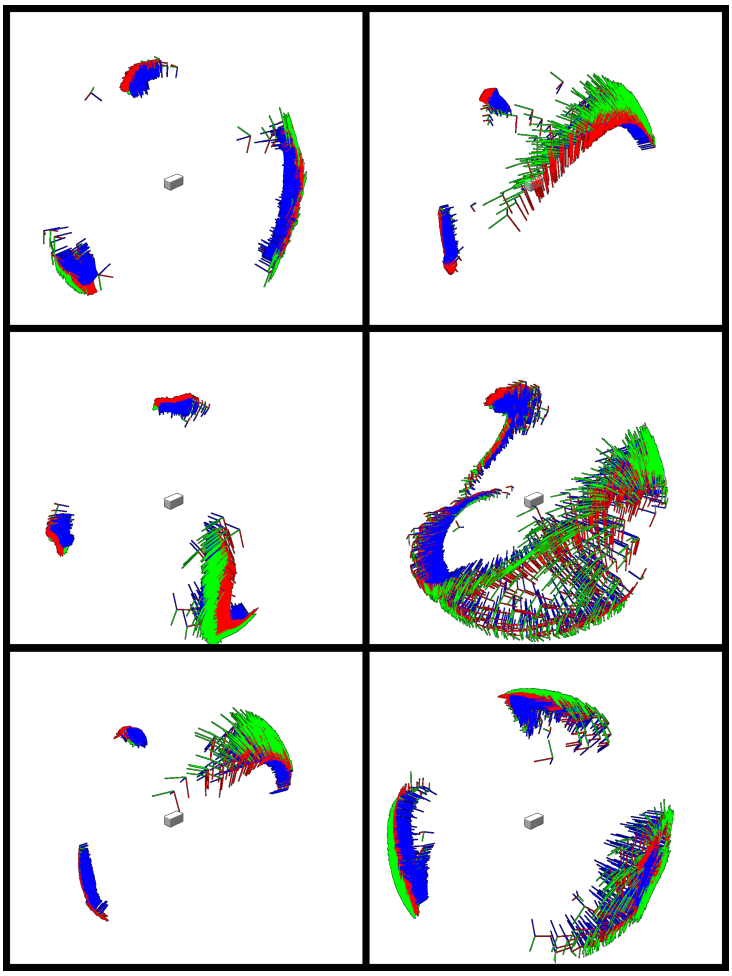

Figure 5.6: The grasp poses collected during the data generation process for the medium box, for each grasp type. Each 6DoF grasp pose is represented by its coordinate frame, where each color represents a different axis.

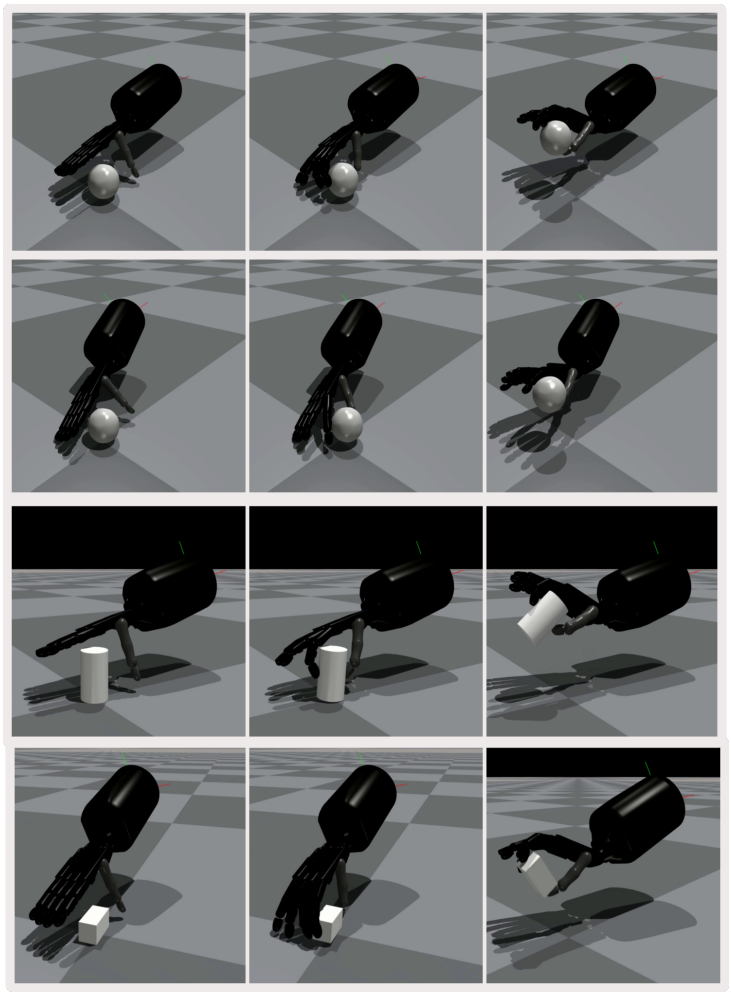

Figure 5.7: Example grasps sampled from our model for different objects and different grasp types. Each row depicts a different grasp attempt. In the first column we see the hand in the pose sampled by our model and the fingers in the pre-grasp position. In the second column the grasp is executed. In the third column the object is lifted to verify that the grasp is stable. The grasp types from the top row to bottom are the following: tripod grasp, palmar pinch, pinch, lateral tripod.

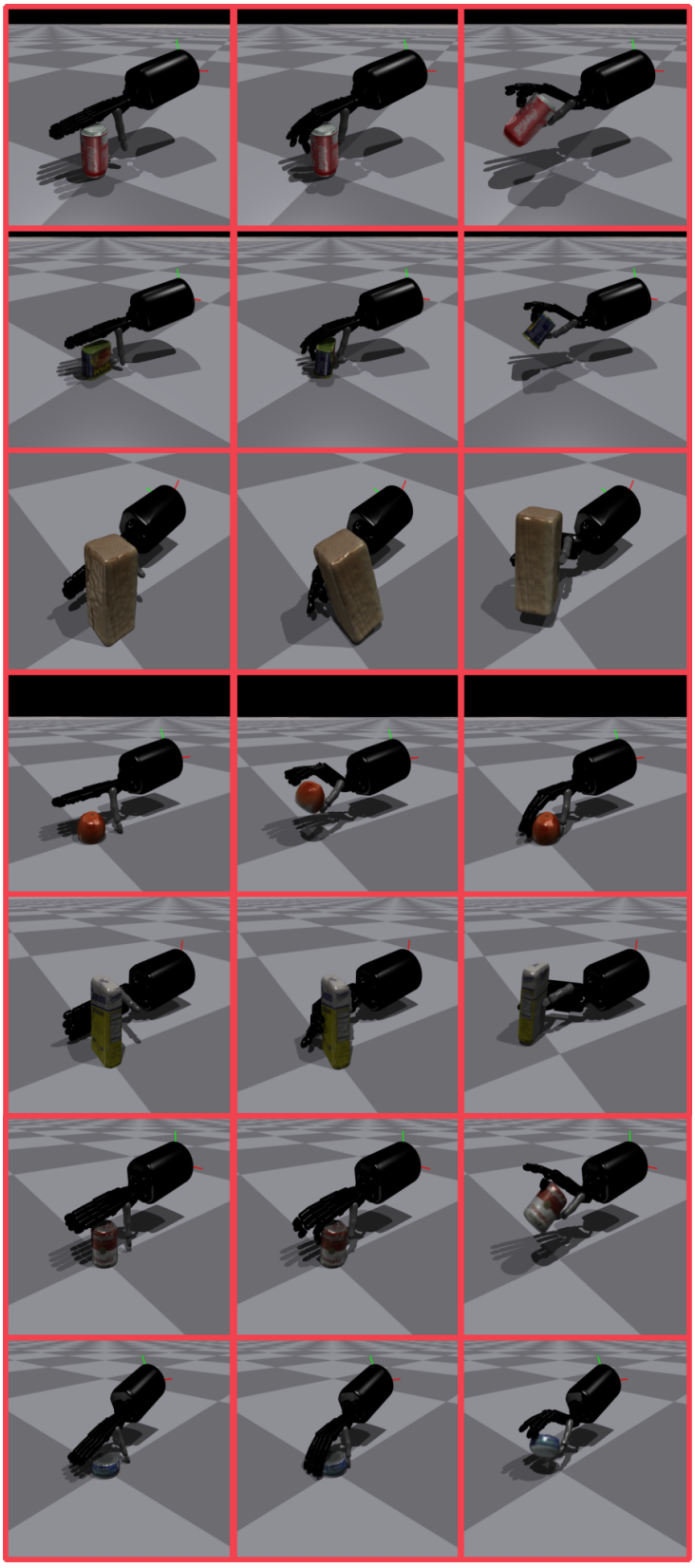

Figure 5.8: Example grasps sampled from our model for different objects from the YCB object dataset.

Chapter 6

Integrating Task Constraints

Typically, human grasping behavior is influenced by the broader context of the manipulation action to be performed. Two primary factors that impact grasping are the properties of the object and the intention or semantics of the task at hand [68, 69]. These factors introduce various constraints, including kinematic constraints (such as the choice of grasp type), geometric constraints (stemming from the object's shape), and functional constraints (ensuring the most suitable way to grasp the object to accomplish the task successfully).

Throughout this thesis, we have delved into the representation and generation of dexterous precision grasp postures, the integration of tactile feedback for grasping and releasing objects, and the optimal hand placement relative to an object for successful grasping. In essence, we have explored the integration of kinematic and geometric constraints within our agent's grasping capabilities.

Naturally, the logical progression of our research now leads us to investigate how we can further enhance the robot's grasping behavior by incorporating task constraints. In this next chapter we focus on conditioning the robot's grasp planning and execution on relevant task-specific factors, enabling more sophisticated and context-aware manipulation actions.

Developing humanoid robots that follow human social norms is crucial for enhancing human-robot collaboration and integrating humanoids into human-centric environments. To accomplish this, we need to develop robotic agents that mimic human grasping behavior. This requires the robot to consider task-specific factors, allowing for more advanced and context-aware manipulation actions. The main challenge of this issue is its intrinsic link to human behavior, preferences, and social norms. Consequently, any proposed solutions must be based on real-world data that reflect these characteristics.

More specifically, in the final stage of our research, we explore generating task-specific grasps based on the intended post-grasp manipulation action. The goal is to develop an agent that can grasp objects in a way that is well-suited for the particular task that wants to accomplish. For instance, just as a person would prefer to grasp a hammer by its handle in order to use it but opt for grasping it by the head when handing it over to someone else (as depicted in Figure 6.1). Robots operating in social environments and collaborating with humans should demonstrate similar adaptive grasping behavior.

Previous works have employed both open-loop and closed-loop grasp generation methods. In open-loop

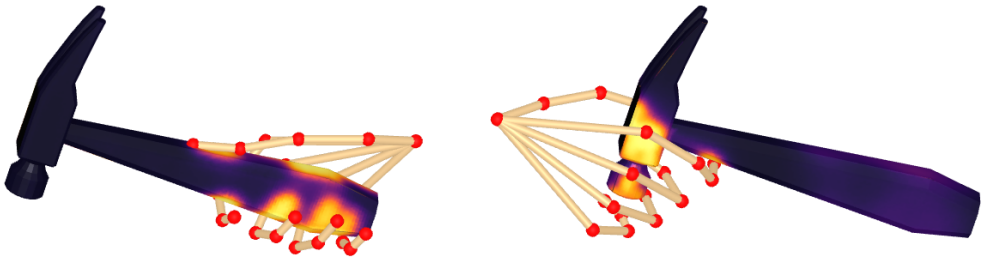

Figure 6.1: Example of different execution of a grasp according to the post-grasp intention as captured in the dataset presented in [53]. In the left figure, the person grasps the hammer in order to use it, in the right it grasps it in order to hand it over to another person.

approaches a grasp sampling method produces a static grasp pose, considering task constraints to ensure alignment with the task's function. Subsequently, a motion planning algorithm devises a trajectory to execute the proposed grasp. Conversely, in closed-loop approaches, a policy utilizes observations and task constraints as inputs, generating actions that result in suitable object grasping. Open-loop methods often depend on extensive labeled datasets for training the grasp generation modules. However, their inherent nature prevents them from adjusting trajectories based on real-time observations or accommodating measurement uncertainties. In contrast, closed-loop methods typically leverage reinforcement learning algorithms to train grasping policies through exploration, necessitating numerous training samples and complex reward shaping to incorporate human priors.

Our approach aims to leverage the postural synergies framework, introduced in the previous chapters, to develop an agent that incorporates post-grasp intention in its grasping behavior. Specifically, our objective is to enable the robot to grasp objects differently based on the intended purpose of the grasping action, in accordance to human social norms. To achieve this, we combine elements from both open-loop and closed-loop approaches, leveraging the strengths of each to create a task-oriented grasping system. By doing so, we aim to develop a more versatile and efficient grasping method that can adapt to various task requirements.

More specifically, we employ a closed-loop approach as our backbone, i.e. train a grasping policy using reinforcement learning. This method does not require large amounts of labeled data, addresses the entire grasping task without additional planning steps, and demonstrates robustness to external disturbances. To incorporate human grasping preferences, we leverage the dataset presented in [70], which consists of objects grasped by several participants for different tasks, as well as the executed grasps (Figure 6.1). Additionally, utilizing the postural synergies framework allows us to both reduce the exploration space for fully actuated humanoid hands, thus accelerating the learning process, and produce finger articulations that are more natural and human-like, potentially leading to superior performance in real-world scenarios.

6.1 Reinforcement learning for task-oriented grasping

The aim of this work is to develop an agent capable of grasping various objects based on the post-grasp intention, provided as a conditional variable to the policy. We assume that the policy can access the

global pose of the object and its own state. The agent's objective is to lift the target object from the table in front of it. To accomplish this, we extract grasp data from a dataset of human grasps with different post-grasp intentions and train a policy using reinforcement learning to exhibit the desired behavior.

6.1.1 Dataset pre-processing.

We utilize the dataset introduced in [70], which includes 3D models of everyday objects annotated with contact maps captured from human participants grasping the objects with two different post-grasp intentions: 1) using the object, and 2) handing the object to another person. The dataset also contains the hand postures, i.e., the 3D positions of keypoints on the fingers of each subject, estimated through a 3D vision pipeline. In our work, we use two sets of data: 1) the grasp postures of the human subjects, and 2) the grasp locations they selected for each task.

Hand postures. Firstly, we transfer the grasp postures executed by the human participants to the robotic hand by designing a fixed mapping function between the two kinematic chains. Specifically, for each joint, we calculate the angle defined by the adjacent keypoints (shown as red points in Figure 6.1) and remap this angle to the joint limits of the robotic hand. This procedure is applied to all grasps and objects in the ContactPose dataset, resulting in a total of 2,596 grasps.

Using this dataset, we train a VAE model to extract the synergy space, following the method detailed in Chapter 3. We utilize a two-dimensional latent space for the synergy space, as our previous results have shown it to be sufficient for generating diverse grasps for different manipulation tasks. This model allows us to sample new grasps by providing new latent points. Consequently, the policy only needs to generate the two-dimensional latent point instead of the full joint angles of the robotic hand, which has 17 degrees of freedom (DoFs), thereby significantly speeding up the optimization process.

Grasp targets. Our next step involves identifying target grasp locations on each object corresponding to specific post-grasp intentions. For instance, consider the grasps on the hammer shown in Figure 6.1, intended for use and handoff. We can extract two target grasp locations from these actions: one on the handle and one on the head of the hammer (Figure 6.2). To automate this process, we calculate a 3D point with repsect to the object for each grasp, representing the average position of the thumb, index, and middle fingertips. Given that multiple participants grasped each object, we have several points indicating the grasp location chosen by each participant.

Subsequently, we cluster these contact points into two groups: one for use and one for handoff. For each cluster, we analyze the grasps to determine whether the majority were for *handoff* or *use* and assign the appropriate label. The center of each cluster is then defined as the grasp target for that object's specific post-grasp intention. These target grasp points are used to guide the reinforcement learning policy in grasping the object near the designated location. By employing the synergy grasp model to reduce the agent's action space and using the grasp targets to direct the agent to the desired grasp location, we can effectively train our policy.

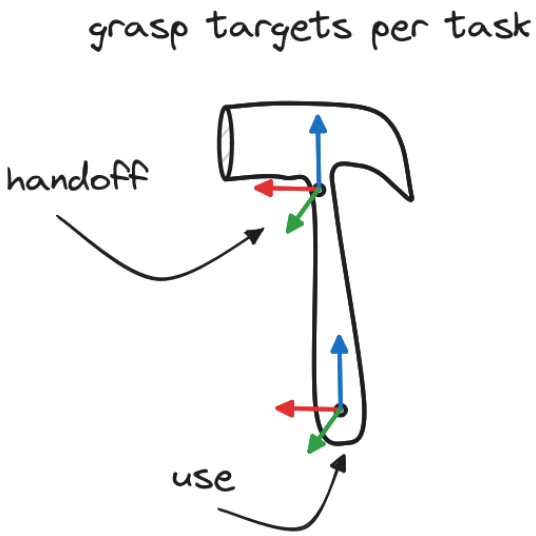

Figure 6.2: Example grasping targets for hammer extracted from the ContactPose dataset [70].

6.1.2 Dexterous task-oriented grasping.

Our objective is to create an agent capable of grasping various objects according to a specified post-grasp intention. To achieve this, we train a reinforcement learning policy that generates the necessary low-level actions for the task. We assume the policy has access to both the object's pose and proprioceptive information. Additionally, the target object and the post-grasp intention are provided to the policy as conditional variables to guide its behavior.

Policy. The policy is implemented using a neural network that takes the current observations, the post-grasp intention, and the object's category as inputs and produces two actions: one for the hand and one for the arm. The hand action is a low-dimensional point in the synergy space, which is then decoded into finger joint values by the synergy model. The arm action specifies the end-effector's displacement in Cartesian space, which is then utilized by an Operational Space Controller (OSC) [71] to compute the desired joint angles. Figure 6.3 provides a graphical representation of the policy structure. To optimize the policy, we employ the Proximal Policy Optimization (PPO) algorithm [72], a widely used method in online reinforcement learning.

Action and state space. The policy receives as observations: the robot's state — represented by the angles of each joint — the object's state — comprising its 3D position and orientation — and its category, which is a one-hot encoded vector. Additionally, it takes a task-conditional variable indicating the post-grasp intent, also encoded as a one-hot vector. Based on these inputs, the policy generates actions for the robot. These actions include a 6-DoF pose for the end-effector, which the Operational Space Controller uses to compute the corresponding joint angles for the arm, and a latent point that the VAE synergy model uses to generate the grasp posture (i.e., joint angles) for the robotic hand. Finally, the robot is controlled via position control, with the computed joint angles applied accordingly.

Reward function. The policy's goal is to lift the specified object from the table while considering the post-grasp intent. To accomplish this, we design the reward function to provide a high reward when

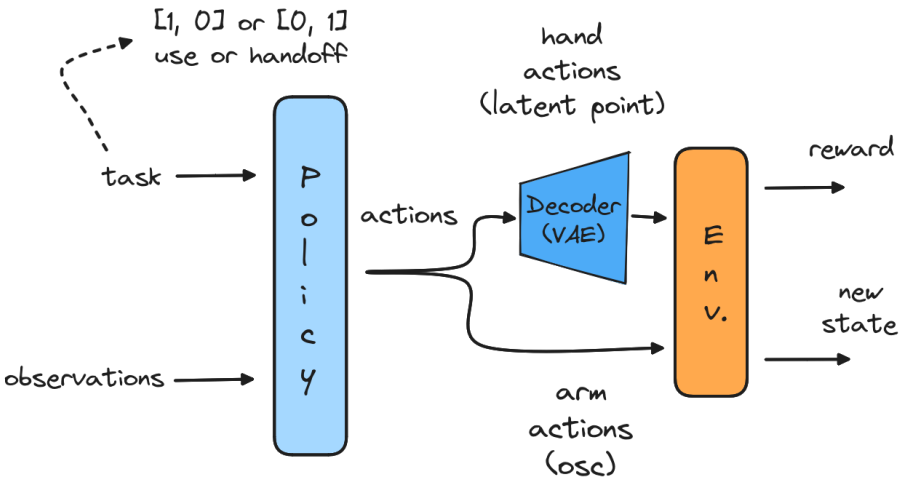

Figure 6.3: Proposed agent structure for task-oriented grasping.

the object is successfully lifted and the grasp location is near the desired grasp location.

$$r(t) = w_1 * r_{hand\_obj\_dist} + w_2 * r_{lift} + w_3 * r_{hand\_rot}$$

$$r_{hand\_obj\_dist} = r_{fingertips\_object\_dist} + r_{palm\_object\_dist}$$

$$r_{lift} = r_{object\_height} + r_{object\_grasped}$$

The reward function comprises three terms designed to achieve specific objectives. The first term evaluates the proximity of the grasp location relative to the desired target. Specifically, $r_{hand\_obj\_dist}$ quantifies this proximity by incorporating two distance measures: $fingertips\_object\_dist$, the distance from the fingertips to the target grasp location, and $r_{palm\_object\_dist}$, the distance from the palm to the target grasp location on the object. The reward increases as these distances decrease, thereby encouraging the policy to position the hand on the target grasp location and close the fingers. By reinforcing a high reward for a close alignment between the chosen grasp location and the specified target, the learning process is fine-tuned to prioritize precision in object manipulation.

Secondly, the reward incentivizes successful lifting of the target object by converging the agent's actions toward achieving this goal. The lifting reward, r_{lift}, consists of two terms: r_{object_height}, which increases as the object's height increases, and a binary term, $r_{object\_grasped}$, which provides a positive reward when the object is lifted above a specified height. Finally, a rotation reward, $r_{rotation}$, is included to encourage the hand to maintain its orientation toward the floor, aiding in task stability and enhancing the exploration phase.

6.2 Experiments

Implementation details. In all our experiments, we use the 7-DoF Kinova Gen3 robotic arm in combination with the Seed Robotics hand, which features 19 DoFs in simulation. We built the simulation environment, using Nvidia's IsaacGym [67], a physics-based simulator that supports parallel training

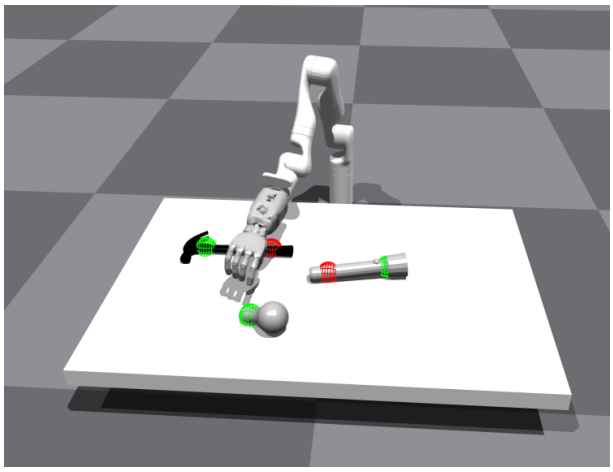

Figure 6.4: Example training environment.

across multiple environments.

We select three objects from the ContactPose database [53] for our simulations which have distinct target grasp locations for each task: a hammer, a flashlight, and a light bulb. We chose these objects due to the fact that the post-grasp intentions were clearly separated on the object's surface. For example, the hammer all grasps associated with the *use* post-grasp intention were on the handle and most of the grasps associated with the *use* post-grasp intention were on the head. In contrast, the grasps on the other objects were more randomly placed on the objects making it hard to separate the use cases.

The policy network is structured as a recurrent neural network (RNN) with a single layer of 768 hidden units, followed by a multi-layer perceptron (MLP) with three layers containing 768, 512, and 256 units, respectively. The MLP layers employ Exponential Linear Unit (ELU) activation functions. Each policy is trained across 2048 parallel environments for 20,000 epochs, with training conducted for two different random seeds to ensure robustness.

In the simulation, the robot is positioned on the floor with all objects placed on a table in front of it, as shown in Figure 6.4. To introduce variability and robustness into the training scenarios, the position and rotation of each object are randomized using additive Gaussian noise.

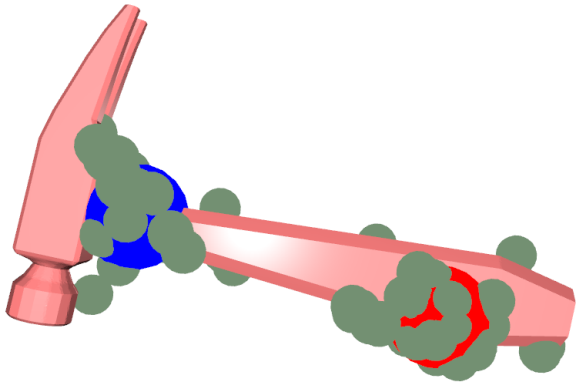

Figure 6.5: Clustering results for grasp points on hammer object.

Evaluation. We evaluated our approach by comparing it to a baseline policy trained with the same

RL algorithm to directly output the finger joint angle values for the hand instead of a low-dimensional point. This approach allows us to assess the performance gains provided by the synergy model. The baseline policy was designed with the same observations, reward functions, and action space for the arm and used the same network architecture as our proposed agent. To assess the performance, we conducted 5000 simulations to determine the average grasp success rate. We also tracked the rewards obtained during training and measured the final distances between the hand and the grasp targets for each task. Additionally, we investigated the impact of increasing the number of latent dimensions in the synergy model on the policy's performance. Lastly, we conducted an ablation study by removing the object category from the policy's observations to evaluate its influence.

6.2.1 Results

Quantitative. Figure 6.6 shows the accumulated rewards achieved by each policy during training. The thick line represents the average reward across the two seeds, while the shaded area denotes the standard deviation. The policies that uses the PCA and the full joint action action space converge faster, at around 2500 iterations, while the the policy that uses the synergy model converges at around 5000 iterations. But all policies achieve similar rewards after 20000 iterations, with low variation.

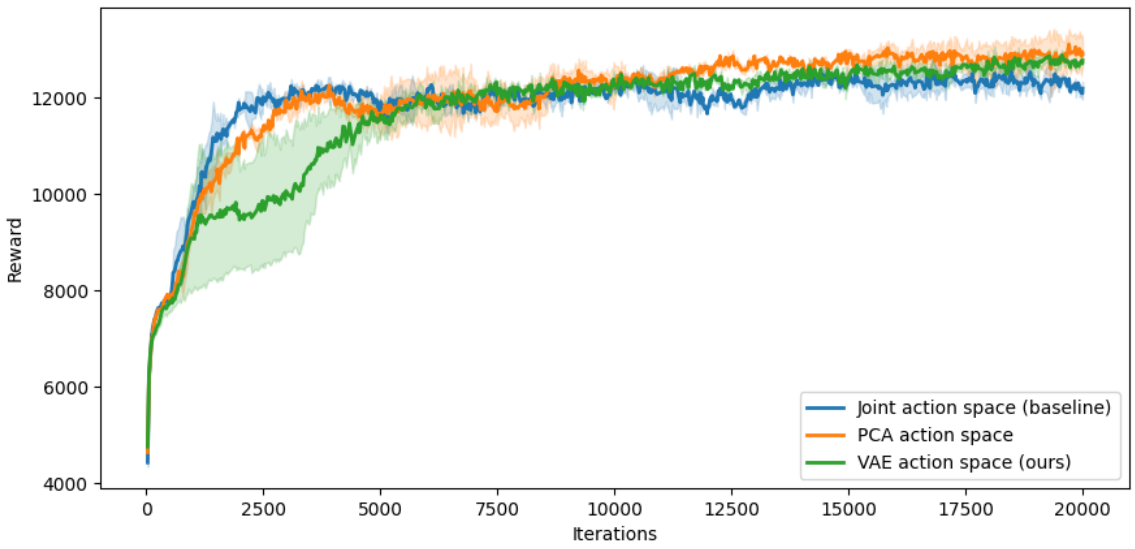

Figure 6.6: Rewards for training policies with 1) full joint control, 2) PCA synergy space, and 3) VAE synergy space.

Nevertheless, higher rewards alone do not always guarantee better task performance. Therefore, we evaluated the best policy from each method by examining its grasp success rate. We tested the policies in 5000 environments and computed the average grasp success rate, defined as the number of times the object was successfully lifted from the table in each episode. The results, summarized in Table 6.1, indicate that the VAE action space policy performs better in terms of grasp success, which is our final objective.

Moreover, we want the agent to grasp the object from a specific location for each post-grasp intention. Figure 6.7 illustrates the distance between the robot's hand and each grasp target (defined for each object

Table 6.1: Average success rate for each policy for 5000 trials.

Method	**Success Rate**
Joint action space	66%
PCA action space	71%
VAE action space (ours)	**83%**

as described in 6.1.1) for the two post-grasp intentions. This distance, measured at the end of each episode, reflects the final position of the hand relative to the object. We conducted 1000 trials for each object and post-grasp intention, retaining the results of successful grasps.

For instance, the right column of figures, representing trials with the handoff post-grasp intention, shows that in every successful trial, the distance between the hand and the handoff target (green dots) on the object was shorter than the distance to the use grasp target. Conversely, in the left figures, corresponding to the use post-grasp intention, the distances are reversed. This demonstrates that the policy effectively learns to associate each grasp position on the object with the appropriate post-grasp intention, and is able to generate actions to lift the object accordingly.

Next, we examine how the number of latent dimensions in the synergy model affects the policy's performance, by training policies that use models with an increasing number of latent dimensions. Figure 6.8 illustrates the rewards obtained by policies using synergy models with latent dimensions ranging from 1 to 5, while Table 6.2 provides the corresponding grasp success rates. Using only one latent dimension results in a lower grasp success rate, likely due to the limited synergy space that fails to adequately capture successful grasps. Latent dimensions greater than two generally produce similar success rates, with the exception of the model with three latent dimensions, which appears to be an experimental anomaly. These results support our choice of using 2 latent dimensions, as it offers a good balance between performance and complexity.

Table 6.2: Average success rate for policies with synergy models of different latent dimensions.

Number of latent dimensions	**Success Rate**
1	59%
2	83%
3	68%
4	84%
5	81%

Additionally, we evaluated the impact of including the object category variable, provided as an observation, on policy performance. We trained the policy both with and without this variable and measured

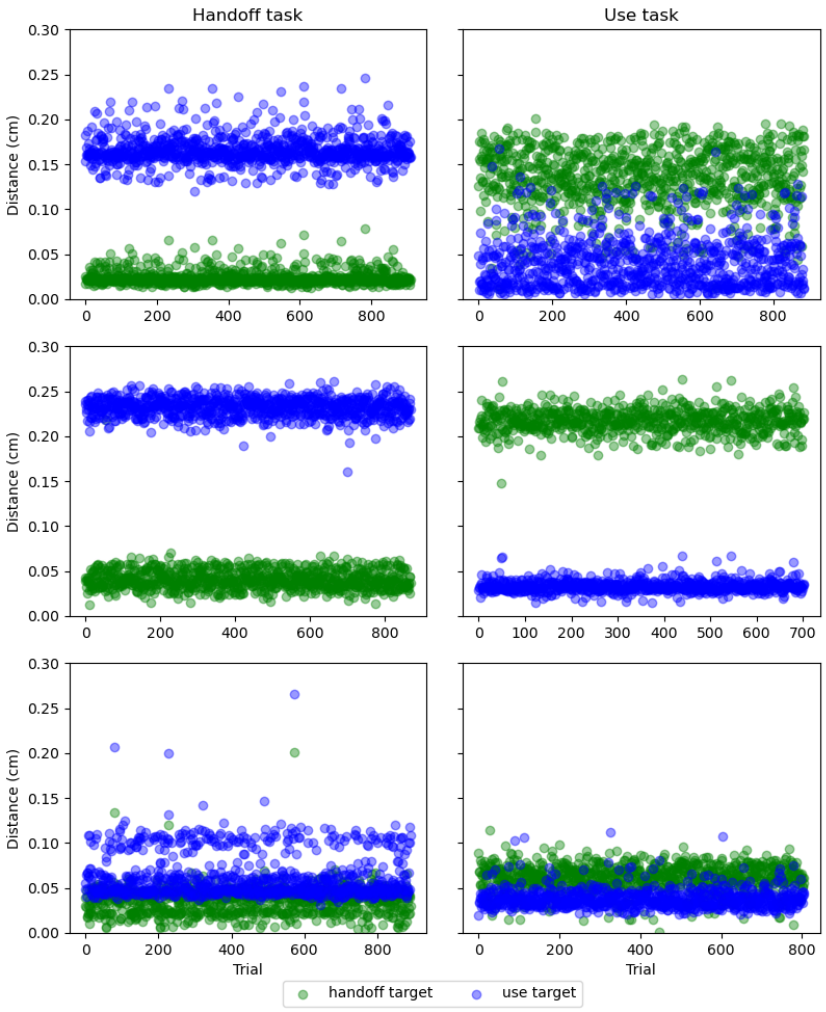

Figure 6.7: Distances of the robot's hand to each grasp target for each task. The results are for a 1000 grasp trials performed for each object and each post-grasp intention performed using our proposed approach.

its grasping performance. Table 6.3 shows the success rates for the policy with and without the object category observation. While removing the object category does not significantly affect the average grasp success rate, Figure 6.9 reveals that the agent struggles to grasp the object from the correct location according to the post-grasp intention. Specifically, for both post-grasp intentions, the robot tends to grasp the object closer to the handoff grasp target rather than the appropriate target for the task, as illustrated in Figure 6.7. This suggests that including the object's category in the observations is essential for the agent to complete the task successfully.

Finally, we evaluated the agent's ability to generalize to unseen objects within the same category as the training objects, but with slightly different shapes. These objects are shown in Figure 6.10, where the first object in each row represents the one used in training, and the next three, labeled 1 to 3, were used for testing generalization. For each object, we conducted 2000 grasp trials in both the *use* and *handoff* tasks, tracking successful grasp attempts. Table 6.4 presents the success rates for each object as well as the average success rate for each category. The results demonstrate that the policy can generalize to unseen objects, although its performance varies slightly depending on the object's shape. Specifically,

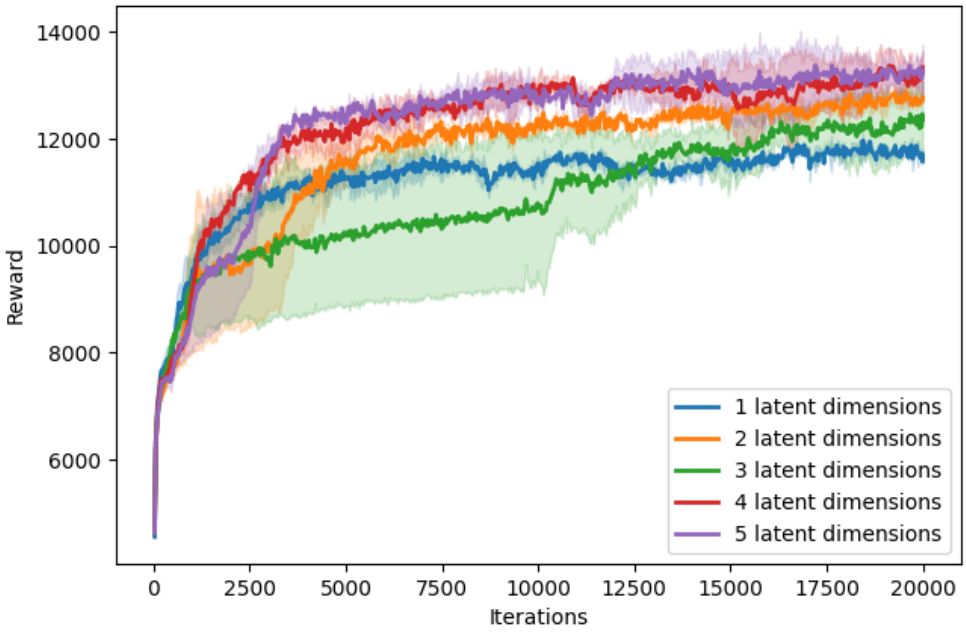

Figure 6.8: Rewards for policies that use synergy models of different latent dimensions.

Table 6.3: Average success rate for each policy for 5000 trials.

Method	**Success Rate**
Policy without object category observation	82%
Policy with object category observation	83%

the success rate for the hammers was noticeably lower compared to other categories, likely due to their distinct shape, particularly at the top. Nonetheless, the findings indicate that the policy is capable of generalizing to new objects, despite being trained on only one object per category.

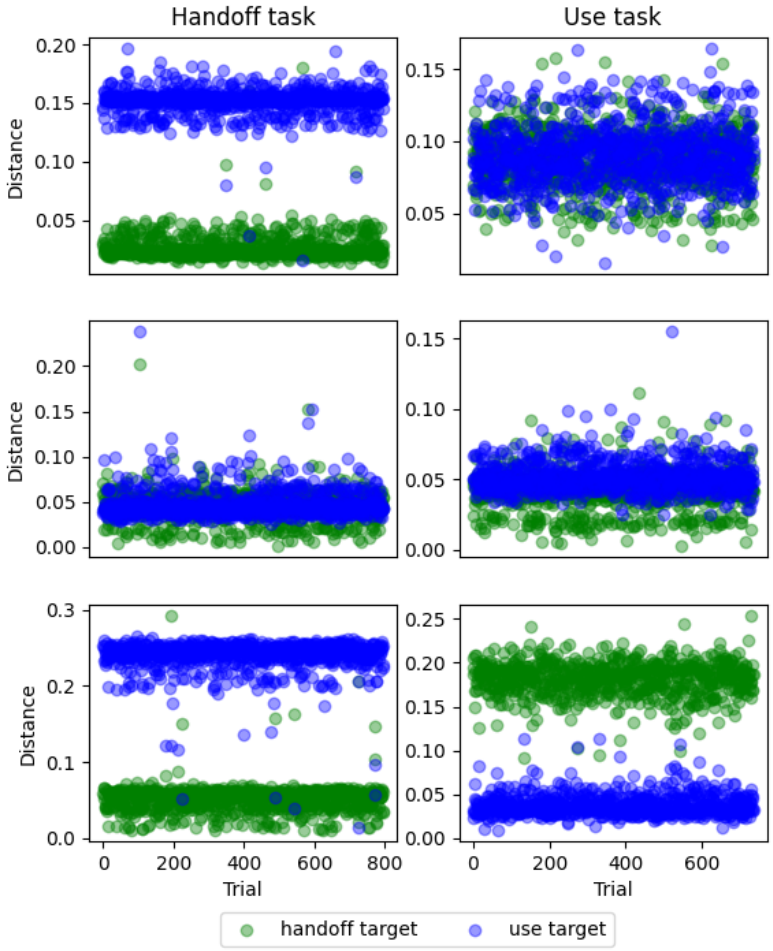

Figure 6.9: Distances of the robot's hand to each grasp target for each task. The results are for a 1000 grasp trials performed for each object and each post-grasp intention performed using a model that does not take as observation the object's category.

Table 6.4: Average grasp success rate of the policy on unseen objects for 2000 trials.

Object	**Success Rate**
Hammer 1	43%
Hammer 2	56%
Hammer 3	50%
Average	**50%**
Flashlight 1	59%
Flashlight 2	65%
Flashlight 3	74%
Average	**66%**
Lightbulb 1	68%
Lightbulb 2	62%
Lightbulb 3	74%
Average	**68%**

Figure 6.10: Additional objects used to evaluate the generalization capabilities of the agent. The first object in each row is the object that was used in training, while the rest are unseen objects from the same category.

Qualitative. In addition to the quantitative benefits of using the synergy model for grasping, there are notable qualitative advantages as well. For instance, when employing the synergy space learned by the VAE model as the action space for the policy, the resulting grasps—i.e., the final finger configurations—resemble those executed by human participants. The first row of Figure 6.11 shows grasps using joint angles as the action space, while the second row illustrates grasps from the policy that uses the synergy space. In the former case, the grasp postures appear unnatural, with some fingers not being utilized and the grasps potentially unstable for subsequent tasks. In contrast, the grasps derived from the synergy space resemble typical power grasps, similar to those performed by humans. This similarity could be advantageous in real-world scenarios, as these grasps may be more robust and effective.

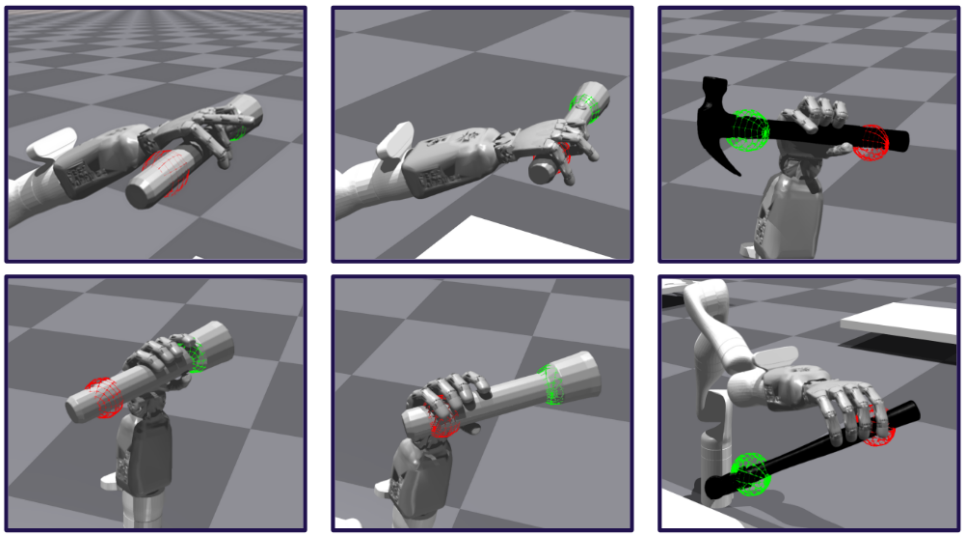

Figure 6.11: The first row of depicts grasps using the joint angles as action space, while the second row depicts grasps from the policy that uses the synergy space as action space

6.3 Conclusions

In summary, we have developed an agent capable of grasping various objects while considering its post-grasp intention. This agent is trained on human grasping data for tasks like object use and handoff, using online reinforcement learning within a simulated environment. By applying the framework of postural synergies, we reduce the action space in reinforcement learning, enabling the agent to produce human-like grasps. Our results show that the agent learns to grasp objects in a manner similar to human preferences, as indicated by the dataset. Moreover, the synergistic approach enhances the grasping success rate compared to using the full joint action space. Despite these advancements, the agent currently only accounts for the position of grasping targets, excluding rotation. Future work should explore incorporating the object's functional characteristics to further improve the agent's effectiveness.

Chapter 7

Conclusions

7.1 Summary

7.1.1 Research Objectives and Questions

This thesis investigates the application of human motor control principles, particularly those related to hand movements, to develop advanced control strategies for dexterous robotic hands. The primary objective was to design and implement methodologies inspired by human motor behavior that significantly improve the versatility, capability, and effectiveness of robotic hands, especially within human-centric environments. Through this research, we aimed to narrow the gap between biological and robotic dexterity, thereby enabling robots to execute complex tasks with greater proficiency in settings where interaction with humans is paramount.

Specifically, our work focused on exploring the synergy framework for controlling robotic hands in various tasks, such as grasping and manipulation. We examined how this framework, which draws on the coordination patterns used by the human hand, can be applied to enhance robotic performance. Additionally, we investigated the potential of integrating synergy models with supplementary feedback modalities, such as tactile feedback, and analyzed the benefits of these integrations for improving robotic manipulation.

Throughout the course of this research, we sought to leverage insights from how the human brain controls hand movements, applying these principles to the development of advanced control techniques for robotic hands. Our goal was to create human-inspired methods that not only facilitate the control of dexterous robots but also enhance their ability to operate effectively and seamlessly in environments designed for human interaction.

7.1.2 Contributions to the Field

In this thesis, we introduced a novel framework for the efficient control of multi-fingered robotic hands, drawing inspiration from the synergistic control strategies employed by the human brain. To replicate this organizational structure in humanoid robots, we developed several models based on the variational

auto-encoder, which enable the generation of new hand postures with specific, predefined characteristics. These models were trained on datasets of human-executed grasps, allowing us to generate human-like hand postures with ease. The effectiveness of our approach was demonstrated through its application in in-hand regrasping tasks, traversing the low-dimensional space to command the hand motions. We demonstrate the successful in-hand manipulation of objects with different shapes using the iCub humanoid robot.

Moreover, we implemented a control system that integrates our synergistic framework with tactile feedback from the robot's fingertips. This integration has proven successful in enabling the robot to perform a variety of object manipulation tasks, while adjusting continuously the grasp size. By controlling the precision grasp size, we show how a humanoid hand lifts and releases objects autonomously, while avoiding slip.

Our work also extended the framework to generate grasp poses for multiple objects, allowing us to create new grasp strategies based solely on the object's pose. More specifically, we developed a sequential model for generating precision grasps in multi-fingered robotic hands using a cascaded sampling approach with CVAE-based samplers. We introduced a geometric heuristic for grasp pose selection and validated our method through experiments.

Finally, we develop a reinforcement learning-based method, grounded in the synergy framework, that integrates task-related information into a robotic agent's grasping behavior. By leveraging insights from human grasping demonstrations and task constraints, our approach enables the robot to adapt its grasping strategy to meet specific task requirements and enhances its ability to collaborate effectively in manipulation tasks. This research provides a strong foundation for advancing the control and autonomy of robotic hands, paving the way for more sophisticated real-world applications.

7.1.3 Summary of Key Findings

In conclusion, our experiments have demonstrated that applying the synergistic approach in the domain of dexterous robotics allows for a significant reduction in the number of control variables required to execute complex manipulation tasks. Specifically, we have shown that generative models, such as Variational Autoencoders (VAEs), are particularly well-suited for modeling the synergy space of robotic hands. The use of a VAE-based model offers several distinct advantages, including the creation of smoother control spaces and the seamless integration of conditional variables that characterize various grasping actions.

Moreover, our findings reveal that synergy models can be effectively integrated with tactile feedback, thereby enhancing the dexterous capabilities of robotic hands. This combination enables more precise and adaptable manipulation, crucial for real-world applications. Lastly, we confirmed that leveraging the synergy space, which is a reduced-dimensional representation compared to the full joint action space of robotic hands, can significantly accelerate and enhance the learning process in reinforcement learning scenarios. This reduction in complexity not only speeds up training but also improves the overall performance and adaptability of the robotic system.

In general, these findings suggest that applying synergistic methods to control complex dexterous

hands is a highly promising approach for addressing the challenges associated with managing a large number of degrees of freedom. By implementing synergies through software rather than embedding them directly into the robot's hardware, we enable the robotic hand to retain its full range of motion while also simplifying control when necessary through the synergistic model. This approach allows for dynamic flexibility, adjusting the level of control complexity based on the task at hand. Moreover, since these synergies are derived from human demonstrations, the robot inherently exhibits more human-like behavior. This characteristic not only enhances the robot's robustness but also significantly improves its potential for effective human-robot collaboration, making interactions more intuitive and seamless.

7.2 Final Remarks

7.2.1 Future Research Directions

Looking forward, several promising avenues for future research emerge from this study. Beyond the specific technical issues discussed at the conclusion of each chapter, there are broader directions that hold significant potential for advancing the field of humanoid robotics.

One such direction is the enhancement of sensor integration. For example, in the realm of tactile sensing, equipping the entire hand with sensors, rather than just the fingertips, could dramatically improve the ability of dexterous robots to perform more complex manipulation tasks. This enhancement would benefit both prehensile tasks, such as object re-orientation, and non-prehensile tasks, like object pushing, by providing more comprehensive and nuanced feedback to the control algorithms.

Another area for future exploration is the incorporation of dynamic demonstrations into the learning process. Currently, our approach to learning from human demonstrations focuses solely on static grasp postures, which are used to train the synergy models. By integrating dynamic demonstrations—full movement trajectories—robots could achieve more complex behaviors and better adapt to a wider variety of tasks and environmental conditions. Additionally, extending the concept of synergies to other body parts, such as the arms or even the whole body, which also operate as high-degree-of-freedom systems, could further enhance the robot's capabilities.

Finally, integrating more sophisticated vision-based feedback systems could significantly enhance the robustness of control algorithms, allowing robots to adapt to an even broader range of tasks and environmental conditions. In this thesis, we primarily relied on geometric information about the objects being manipulated, such as their center of mass or shape category. However, developing agents capable of processing visual inputs, such as images or depth information, presents a substantial opportunity to improve the capabilities and applicability of dexterous robots. This advancement could enable robots to handle a wider variety of objects and scenarios with greater precision and adaptability.

7.2.2 Broader Implications of the Research

The research presented in this thesis represents a small yet significant contribution to the broader efforts within the robotics community to enhance the capabilities of humanoid robots and expand their

range of applications. The technical advancements introduced here hold the potential to make a substantial impact on the field of robotics by accelerating the integration of humanoid robots into human-centered environments. One of the most valuable aspects of humanoid robots lies in their ability to use their hands to perform precise movements and complex manipulation tasks, which are essential for a wide array of real-world applications.

Humanoid robots equipped with dexterous hands can be particularly effective in settings where human-like behavior is required, such as in human-robot collaboration scenarios or in environments that lack the structure and predictability of industrial settings. In collaborative environments, these robots can work alongside humans, taking on tasks that demand fine manipulation and precision. The integration of human-like manipulation capabilities in humanoid robots has the potential to transform industries by increasing productivity and mitigating the effects of labor shortages, particularly in roles that involve repetitive, hazardous, or physically demanding tasks.

Furthermore, the deployment of dexterous robots could significantly enhance the quality of life for individuals in everyday scenarios. For instance, they could take over routine household chores, serve as personal assistants, and improve accessibility for people with disabilities. Finally, the principles of dexterous manipulation explored in this thesis could also be applied to the development of advanced prosthetic limbs that closely mimic natural hand movements, thereby improving the lives of amputees by providing them with greater autonomy and functionality.

Bibliography

[1] T. Kivell, P. Lemelin, B. Richmond, and D. Schmitt. *The Evolution of the Primate Hand: Anatomical, Developmental, Functional, and Paleontological Evidence*. 08 2016.

[2] M. Santello, M. Bianchi, M. Gabiccini, E. Ricciardi, G. Salvietti, D. Prattichizzo, M. Ernst, A. Moscatelli, H. Jörntell, A. M. Kappers, K. Kyriakopoulos, A. Albu-Schäffer, C. Castellini, and A. Bicchi. Hand synergies: Integration of robotics and neuroscience for understanding the control of biological and artificial hands. *Physics of Life Reviews*, 17:1–23, 2016.

[3] A. Bicchi. Hands for dexterous manipulation and robust grasping: a difficult road toward simplicity. *IEEE Transactions on Robotics and Automation*, 16(6):652–662, 2000.

[4] G. Salvietti. Replicating human hand synergies onto robotic hands: A review on software and hardware strategies. *Frontiers in Neurorobotics*, 12:27, 2018.

[5] D. Dimou, J. Santos-Victor, and P. Moreno. Learning conditional postural synergies for dexterous hands: A generative approach based on variational auto-encoders and conditioned on object size and category. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 4710–4716. IEEE, 2021.

[6] D. Dimou, J. Santos-Victor, and P. Moreno. Robotic hand synergies for in-hand regrasping driven by object information. *Autonomous Robots*, 47(4):453–464, 2023.

[7] D. Dimou, J. Santos-Victor, and P. Moreno. Force feedback control for dexterous robotic hands using conditional postural synergies. In *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, pages 674–679. IEEE, 2022.

[8] D. Dimou, J. Santos-Victor, and P. Moreno. Grasp pose sampling for precision grasp types with multi-fingered robotic hands. In *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, pages 773–779. IEEE, 2022.

[9] D. Dimou, J. S. Victor, and P. Moreno. Task-oriented grasping for dexterous robots using postural synergies and reinforcement learning. In *2024 Eighth IEEE International Conference on Robotic Computing (IRC)*, pages 65–71. IEEE, 2024.

[10] J. Starke, C. Eichmann, S. Ottenhaus, and T. Asfour. Synergy-based, data-driven generation of

object-specific grasps for anthropomorphic hands. *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, 2018.

[11] N. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *NIPS*, 2003.

[12] N. Lawrence and J. Q. Candela. Local distance preservation in the gp-lvm through back constraints. In *ICML '06*, 2006.

[13] M. Santello, M. Flanders, and J. F. Soechting. Postural hand synergies for tool use. *Journal of Neuroscience*, 18(23):10105–10115, 1998.

[14] M. Ciocarlie, C. Goldfeder, and P. Allen. Dimensionality reduction for hand-independent dexterous robotic grasping. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[15] M. T. Ciocarlie, C. Goldfeder, and P. K. Allen. Dexterous grasping via eigengrasps : A low-dimensional approach to a high-complexity problem. 2007.

[16] A. Bernardino, M. Henriques, N. Hendrich, and J. Zhang. Precision grasp synergies for dexterous robotic hands. In *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec 2013.

[17] O. C. Jenkins. 2d subspaces for sparse control of high-dof robots. In *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, Aug 2006.

[18] A. Tsoli Odest and O. Jenkins. 2d subspaces for user-driven robot grasping. *Robotics, Science and Systems Conference: Workshop on Robot Manipulation*, 01 2007.

[19] J. Starke, C. Eichmann, S. Ottenhaus, and T. Asfour. Human-inspired representation of object-specific grasps for anthropomorphic hands. *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*, 2020.

[20] J. Romero, T. Feix, C. H. Ek, H. Kjellström, and D. Kragic. Extracting postural synergies for robotic grasping. *IEEE Transactions on Robotics*, 29(6):1342–1352, Dec 2013.

[21] K. Xu, H. Liu, Y. Du, and X. Zhu. A comparative study for postural synergy synthesis using linear and nonlinear methods. *International Journal of Humanoid Robotics*, 13(03):1650009, 2016.

[22] G. Palli, F. Ficuciello, U. Scarcia, C. Melchiorri, and B. Siciliano. Experimental evaluation of synergy-based in-hand manipulation. *IFAC Proceedings Volumes*, 47:299–304, 2014.

[23] S. Katyara, F. Ficuciello, D. G. Caldwell, B. Siciliano, and F. Chen. Leveraging kernelized synergies on shared subspace for precision grasping and dexterous manipulation. *IEEE Transactions on Cognitive and Developmental Systems*, 15(4):2064–2076, 2023.

[24] H. Yousef, M. Boukallel, and K. Althoefer. Tactile sensing for dexterous in-hand manipulation in robotics-a review. *Sensors and Actuators A-physical*, 2011.

[25] R. S. Johansson and J. R. Flanagan. Coding and use of tactile signals from the fingertips in object manipulation tasks. *Nature Reviews Neuroscience*, 10:345–359, 2009.

[26] J. R. Flanagan, M. C. Bowman, and R. S. Johansson. Control strategies in object manipulation tasks. *Current Opinion in Neurobiology*, 16:650–659, 2006.

[27] Z. Su, K. Hausman, Y. Chebotar, A. Molchanov, G. E. Loeb, G. S. Sukhatme, and S. Schaal. Force estimation and slip detection/classification for grip control using a biomimetic tactile sensor. *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 297–303, 2015.

[28] F. Veiga, B. B. Edin, and J. Peters. In-hand object stabilization by independent finger control. *ArXiv*, abs/1806.05031, 2018.

[29] Z. Deng, Y. Jonetzko, L. Zhang, and J. Zhang. Grasping force control of multi-fingered robotic hands through tactile sensing for object stabilization. *Sensors (Basel, Switzerland)*, 20, 2020.

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

[31] J. Bohg, A. Morales, T. Asfour, and D. Kragic. Data-driven grasp synthesis—a survey. *IEEE Transactions on Robotics*, 30:289–309, 2014.

[32] R. Newbury, M. Gu, L. Chumbley, A. Mousavian, C. Eppner, J. Leitner, J. Bohg, A. Morales, T. Asfour, D. Kragic, D. Fox, and A. Cosgun. Deep learning approaches to grasp synthesis: A review. *IEEE Transactions on Robotics*, 39(5):3994–4015, 2023.

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

[34] M. Liu, Z. Pan, K. Xu, K. Ganguly, and D. Manocha. Generating grasp poses for a high-dof gripper using neural networks. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1518–1525, 2019.

[35] M. Liu, Z. Pan, K. Xu, K. Ganguly, and D. Manocha. Deep differentiable grasp planner for high-dof grippers. *ArXiv*, abs/2002.01530, 2020.

[36] M. Kopicki, D. Belter, and J. L. Wyatt. Learning better generative models for dexterous, single-view grasping of novel objects. *The International Journal of Robotics Research*, 38:1246–1267, 2019.

[37] M. Kopicki, R. Detry, M. Adjigble, R. Stolkin, A. Leonardis, and J. L. Wyatt. One-shot learning and generation of dexterous grasps for novel objects. *The International Journal of Robotics Research*, 35 (8):959–976, 2016.

[38] U. 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*, 2022.

[39] Q. Lu and T. Hermans. Modeling grasp type improves learning-based grasp planning. *IEEE Robotics and Automation Letters*, 4:784–791, 2019.

[40] M. Corsaro, S. Tellex, and G. D. Konidaris. Learning to detect multi-modal grasps for dexterous grasping in dense clutter. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4647–4653, 2021.

[41] V. Mayer, Q. Feng, J. Deng, Y. Shi, Z. Chen, and A. Knoll. Ffhnet: Generating multi-fingered robotic grasps for unknown objects in real-time. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 762–769, 2022.

[42] W. Wei, D. Li, P. Wang, Y. Li, W. Li, Y. Luo, and J. Zhong. Dvgg: Deep variational grasp generation for dextrous manipulation. *IEEE Robotics and Automation Letters*, 7(2):1659–1666, 2022.

[43] A. Wu, M. Guo, and C. K. Liu. Learning diverse and physically feasible dexterous grasps with generative model and bilevel optimization. *ArXiv*, abs/2207.00195, 2022.

[44] A. Mousavian, C. Eppner, and D. Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2901–2910, 2019.

[45] T. Feix, I. M. Bullock, and A. M. Dollar. Analysis of human grasping behavior: Correlating tasks, objects and grasps. *IEEE Transactions on Haptics*, 7:430–441, 2014.

[46] M. Hjelm, C. H. Ek, R. Detry, and D. Kragic. Learning human priors for task-constrained grasping. In L. Nalpantidis, V. Krüger, J.-O. Eklundh, and A. Gasteratos, editors, *Computer Vision Systems*, pages 207–217. Springer International Publishing, 2015.

[47] D. Song, K. Huebner, V. Kyrki, and D. Kragic. Learning task constraints for robot grasping using graphical models. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1579–1585, 2010.

[48] M. Li. Learning partial power grasp with task-specific contact. *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 337–343, 2016.

[49] A. Murali, W. Liu, K. Marino, S. Chernova, and A. K. Gupta. Same object, different grasps: Data and semantic knowledge for task-oriented grasping. In *The Conference on Robot Learning*, 2020.

[50] M. Kokic, J. A. Stork, J. A. Haustein, and D. Kragic. Affordance detection for task-specific grasping using deep learning. *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 91–98, 2017.

[51] W. Liu, A. A. Daruna, and S. Chernova. Cage: Context-aware grasping engine. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2550–2556, 2020.

[52] P. Mandikal and K. Grauman. Learning dexterous grasping with object-centric visual affordances. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6169–6176, 2021.

[53] S. Brahmbhatt, C. Ham, C. C. Kemp, and J. Hays. ContactDB: Analyzing and predicting grasp contact via thermal imaging. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6 2019.

[54] P. Mandikal and K. Grauman. DexVIP: Learning dexterous grasping with human hand pose priors from video. In *5th Annual Conference on Robot Learning*, 2021.

[55] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *International Conference on Learning Representations*, abs/1312.6114, 2013.

[56] D. P. Kingma and M. Welling. An introduction to variational autoencoders. *ArXiv*, abs/1906.02691, 2019.

[57] N. Chen, M. Karl, and P. V. D. Smagt. Dynamic movement primitives in latent space of time-dependent variational autoencoders. *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 629–636, 2016.

[58] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

[59] T. Feix, H. bodo Schmiedmayer, J. Romero, and D. Kragić. A comprehensive grasp taxonomy. In *In Robotics, Science And Systems Conference: Workshop On Understanding The Human Hand For Advancing Robotic Manipulation*, 2009.

[60] G. Westling and R. S. Johansson. Factors influencing the force control during precision grip. *Experimental Brain Research*, 53:277–284, 2004.

[61] Seed Robotics - FTS3 3D High Resolution Tactile (Pressure) sensor. `https://kb.seedrobotics.com/doku.php?id=fts:fts3_pressuresensor`.

[62] Seed Robotics - RH8D Adult Robot Hand. `https://www.seedrobotics.com/rh8d-adult-robot-hand`.

[63] T. Iberall and C. L. MacKenzie. *Opposition Space and Human Prehension*, pages 32–54. Springer New York, New York, NY, 1990.

[64] L. Desanghere and J. J. Marotta. "graspability" of objects affects gaze patterns during perception and action tasks. *Experimental Brain Research*, 212:177–187, 2011.

[65] S. L. Prime and J. J. Marotta. Gaze strategies during visually-guided versus memory-guided grasping. *Experimental Brain Research*, 225:291–305, 2012.

[66] S. Sra. Directional statistics in machine learning: a brief review. *arXiv: Machine Learning*, 2016.

[67] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac gym: High performance gpu-based physics simulation for robot learning. *ArXiv*, abs/2108.10470, 2021.

[68] J. R. Napier. The prehensile movements of the human hand. *The Journal of Bone and Joint Surgery. British volume*, 38-B(4):902–913, 1956.

[69] F. Cini, V. Ortenzi, P. Corke, and M. Controzzi. On the choice of grasp type and location when handing over an object. *Science Robotics*, 4, 2019.

[70] S. Brahmbhatt, C. Tang, C. D. Twigg, C. C. Kemp, and J. Hays. ContactPose: A dataset of grasps with object contact and hand pose. In *The European Conference on Computer Vision (ECCV)*, August 2020.

[71] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[72] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.