UNIVERSITÉ DU LUXEMBOURG

PhD-FSTM-2025-022
The Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on 31/01/2025 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN INFORMATIQUE

by

Kuldeep Rambhai BARAD
Born on 26 November 1995 in Veraval (INDIA)

VISUAL INTELLIGENCE FOR 6-DOF ROBOTIC GRASPING OF UNKNOWN OBJECTS FROM EARTH TO SPACE

Dissertation defence committee

Dr. Miguel Olivares-Mendez, Supervisor
Associate Professor, Université du Luxembourg

Dr Carol Martinez Luna, Vice-Chair
Research Scientist, Université du Luxembourg

Dr Jan Dentler,
Lead Software Architect, Redwire Space Europe

Dr. Holger Voos, Chairman
Professor, Université du Luxembourg

Dr Renaud Detry,
Professor, KU Leuven


uni.lu
UNIVERSITÉ DU
LUXEMBOURG

Doctoral School in Science and Engineering

Dissertation Defence Committee:

Committee members:	Prof. Holger Voos, University of Luxembourg (Chair)
	Prof. Renaud Detry, KU Leuven (External)
	Dr. Jan Dentler, Redwire Space Europe (External, Industry)
Supervisor:	Prof. Dr. Miguel Olivares-Mendez, University of Luxembourg
Co-supervisor:	Dr. Carol Martinez, University of Luxembourg

Affidavit

I hereby confirm that the PhD thesis entitled **"VISUAL INTELLIGENCE FOR 6-DOF ROBOTIC GRASPING OF UNKNOWN OBJECTS FROM EARTH TO SPACE"** has been written independently and without any other sources than cited.

Luxembourg, _____ _____

Name

Acknowledgements

This dissertation is a result of a long journey, during which I had grand imaginations of its conclusion. I thought, surely, the countless struggles, rare successes, long nights and painful rejections would lead to a grand crescendo of emotions. But here I am, sitting at my desk, realizing that the finish line has come and gone. So swiftly that I missed the magic moment everything was supposed to build up to. I failed to realize until now that it was never meant to be just one moment, but every moment that I learned something new, navigated a difficult day, cracked a bad joke, made the robot move, shared a meal, completed a milestone or missed a deadline. The magic of these moments was in the people I shared them with, for I never found myself alone; not really. For someone like me in a foreign land and in a foreign field, this was a true privilege.

I am most thankful to my supervisors, who have supported me unwaveringly. I still fail to comprehend the amount of energy and optimism that Carol brings to work. Carol has had a profound impact on my transformation from a naive aerospace engineer to a robotics researcher. What I'll cherish most is our spontaneous chats, after which no problem seemed insurmountable. I greatly appreciate her ability to let her students debate and brainstorm as equals, despite her years of experience. It has been truly empowering. In the last four years, I developed both academic and industrial acumen, thanks largely to the Jan's supervision under the industrial collaboration. I still remember my first conversation with Jan in the company offices, which was still a garage. His words stuck with me throughout my Ph.D.—"The most important thing is that you develop the skills you need to succeed once you are done". It was simple yet invaluable advice that helped me gain foothold in the industry by the end of the four years. Jan's ability to cut through complexity and ask prudent questions has been an invaluable asset throughout the Ph.D.. Finally, it would be remiss to say any of this was possible without Miguel. From materializing the industrial partnership to placing disproportionate trust in an inexperienced student, Miguel provided me with the foundation for good research. Despite the challenges of building a new research group, he has fostered

an environment of immense trust, empathy and respect. Above all, Miguel has taught me how to truly appreciate people despite circumstances.

While my supervisors were obligated to stick with me, Antoine voluntarily took on the role of a mentor, for which I am eternally grateful. His relentless optimism and deep hands-on knowledge proved invaluable. I am grateful and, at the same time, apologetic for the many hours he spent convincing me of the value of my own work. I'm also thankful to my colleagues at SpaceR—particularly Andrej, Matteo, Mohtashem, and Jota, with whom I've shared far more than just an office. Our conversations spanned from the most futile to the profound. I especially thank Andrej for co-authoring the GraspLDM work with me. His relentless pursuit of quality, unmistakable eye for detail, and willingness to help in any situation have been a great source of learning.

My work family extended to Made In Space Europe and later Redwire Space through industrial collaboration. I thank JJ and Vinayak for initiating this industrial collaboration. I remain grateful to all the colleagues at Redwire, especially to the software team, who have been an absolute delight to work with. I am very thankful to Mohamed, who has been a great friend, collaborator, and mentor, and who imparts more practical knowledge per unit time than anyone I have met. I'm thankful to Al for trusting me to lead the CAESAR collaboration with Stanford University and to Prof. Simone D'Amico and the SLAB team for graciously hosting me for a research visit.

Beyond work, I am eternally grateful to Supriya who has been an unfaltering source of support, kindness, and comfort these past four years. I am deeply thankful to Ester and Daniele for the Friday night climbing and dinner ritual which was a much-needed source of comfort and respite last year. I'm very thankful to Menelaos for letting me crash on his couch during difficult experiment days, and to the 'Bravi Tutti!' and BelVal folks for their friendship and fun.

Finally, I'm grateful for my parents and my brother, Arjun, who remain my strongest pillars of support. Without their steadfast belief in me, I would not have had the courage to see this through.

Index

List of Figures **xii**

List of Tables **xiii**

Abstract **1**

1 Introduction **3**

1.1 Problem Statement 6

1.2 Contributions 7

1.3 Thesis Outline 9

2 Literature Review **11**

2.1 Introduction 11

2.2 Grasp Synthesis 13

2.3 Vision-based 6-DoF Grasp Synthesis 24

2.4 Dynamic Grasping 41

2.5 Vision-based Manipulation and Grasping in Space 45

2.6 Summary and Research Questions 50

3 Generative 6-DoF Grasp Synthesis **53**

3.1 Introduction 53

3.2 Generative Modeling for Grasp Synthesis 54

3.3 GraspLDM: Grasp Latent Diffusion Models 64

3.4 GraspLDM: Implementation 70
3.5 Experiments 77
3.6 Discussion 92
3.7 Summary 94

4 Object-centric System for Real-world Grasping 97
4.1 Introduction 97
4.2 Grasp-O: Object-centric Grasping System 98
4.3 Pipeline 100
4.4 Hardware and Software Architecture 110
4.5 Real-world Experiments 113
4.6 Discussion 118
4.7 Summary 121

5 6-DoF Tracking and Reconstruction of Unknown Objects 123
5.1 Introduction 123
5.2 Structure and Motion from Visual Observations 124
5.3 Object-centric 3D Gaussian Splatting 128
5.4 Incremental 3D Reconstruction and Tracking 134
5.5 Experiments 136
5.6 Discussion 139
5.7 Summary 143

6 Conclusions 145
6.1 Generative Modeling for 6-DoF Grasp Synthesis 145
6.2 Real-world Object-centric Grasping 146
6.3 Perception of Dynamic Unknown Objects 146

Appendices 181

A Real-world Experiment Logs **181**

A.1 Test Setup 1 181

A.2 Test Setup 2 182

List of Figures

1.1 Examples of multi-modalilty and discontinuity in grasp distributions. 4

2.1 Organization of the literature review 12
2.2 Classification of grasps based on degrees of freedom of the hand/wrist . . . 13
2.3 Types of contact models used in robotic grasping 14
2.4 Illustration of closure properties of grasps 16
2.5 4-DoF grasp representation 22
2.6 Grasp parameterizations for 6-DoF grasps of a parallel jaw gripper 28
2.7 Point-based grasp synthesis pipeline in Contact-GraspNet 30
2.8 Volumetric grasp synthesis pipeline in GIGA 31
2.9 Representations of grasp sampling schemes used for dataset generation . . 38
2.10 Examples of diverse object shapes in ACRONYM dataset 40
2.11 Visualization of Launch Adapter Ring capture 46

3.1 Architecture and computation graphs of generative models 57
3.2 Illustration of encoding and decoding of distributions with a VAE 60
3.3 Illustration of unconditional and task-conditional generation from a prior . . . 63
3.4 Grasp Latent Diffusion Model Architecture 65
3.5 Visualization of point-voxel convolution operation 71
3.6 Comparison of latency and memory usage of point cloud learning architectures 72
3.7 Examples of objects and their grasp annotations in ACRONYM dataset . . . 75
3.8 Distribution of training and test object instances in the 63C category set. . . . 76

3.9 GraspLDM performance comparison on full object point clouds 77
3.10 Multi-object grasping environments in Isaac Gym for success rate evaluation. 78
3.11 Visualization of latent space de-noising in GraspLDM 80
3.12 Visualization of task-conditional de-noising with GraspLDM 81
3.13 Grasp generation performance of GraspLDM models 83
3.14 Special scenarios affecting GraspLDM performance 84
3.15 GraspVAE-63C: KL Divergence and Reconstruction Loss during training with different KL weights. 87
3.16 Point cloud reconstruction from latents with different structure. 90

4.1 Overview of grasp execution in the real world using Grasp-O 99
4.2 Illustration of positive and negative grasp samples used in training 102
4.3 Illustration of cluttered scene segmentation using SAM 105
4.4 Software architecture for object-centric grasping system 111
4.5 Objects used for real-world grasping tests 113
4.6 Real-world experiment setups 115
4.7 Examples of object placement on support during real-world testing 116
4.8 Visualization of grasp generation and selection in real world tests 119
4.9 Real-world grasp execution examples on the two setups. 120

5.1 3D reconstruction and pose tracking pipeline using 3D Gaussian Splatting . . 128
5.2 Illustration of projecting 3D Gaussians to 2D splats on the image plane 130
5.3 137
5.4 Spacecraft 3D models used for data generation 137
5.5 Target-relative camera trajectory used for dataset generation 138
5.6 Qualitative reconstruction and tracking results 140
5.7 Pose tracking results for object-centric SLAM using 3D Gaussian Splatting . . 142

List of Tables

2.1 Comparison of datasets for 6-DoF grasp synthesis 36

3.1 Reverse diffusion sampling speed-up in GraspLDM 82
3.2 Impact of KL weight and annealing on GraspLDM performance 86
3.3 Impact of FiLM conditioning on model performance 88
3.4 Impact of shape latent size on model performance 89
3.5 Impact of conditional grasp latent size on model performance 91

4.1 GraspClassifier performance metrics on validation set 103
4.2 Hardware and software compatibility matrix for object-centric grasping system 112
4.3 Real-world 6-DoF grasping success rate comparison on 16 evaluation objects in five random poses 117
4.4 Inference latency for segmentation, grasp generation and classification models in the pipeline 118

5.1 Evaluation of 3D reconstruction performance using bi-directional chamfer distance 141

Abstract

Autonomous robotic manipulation is fundamental to realizing robots that will assist in homes and support space operations. This dissertation explores one of the elementary challenges in robotic manipulation– the ability of a robot to grasp objects that it must manipulate. While grasping known objects is well-studied, real-world manipulation often involves unknown objects. This work proposes an object-centric approach to learning vision-based 6-DoF grasp synthesis, sim-to-real transfer, and perception for dynamic grasping. The first major contribution is GraspLDM, a generative framework that effectively learns and samples from the complex distribution of object-centric 6-DoF grasp poses for unknown objects. By combining Variational Autoencoders (VAE) with latent diffusion models, GraspLDM retains benefits of VAEs while overcoming challenges like prior and posterior gap. GraspLDM achieves superior grasp quality compared to existing generative models, while affording flexibility of task-conditional generation unavailable in the latter. The second contribution is Grasp-O, a modular object-centric grasping system that enables reliable transfer of simulation-trained grasp synthesis models to physical robots. The system is used to validate sim-to-real transfer of GraspLDM, demonstrating approximately 80% success rate on 16 unknown objects in different poses across two robotic setups. The third contribution is a novel approach to simultaneous 3D reconstruction and 6-DoF pose tracking of unknown objects that can assist dynamic grasping. Using object-centric 3D Gaussian Splatting, reconstruction and tracking are accomplished under a single representation that also interfaces with grasp synthesis models like GraspLDM. The approach is validated on challenging scenarios involving spacecraft in unconstrained relative motion. Together, these contributions take a step towards general-purpose robotic manipulation that will benefit both terrestrial and space applications.

Chapter 1

Introduction

> *"... we wish to let the computer deal directly with the real world by itself, beginning with perception of the real world and the appreciation of it and ending with the performance of a purposeful active task in the real world."*
>
> – Heinrich Ernst [1], 1961

Robotic grasping research has a rich history in both academia and industry dating back to the 1960s when the first industrial robotic grippers were built. Fascinatingly, the original idea of the first robotic gripper 'MH-1' [1] traces back to 1958 to a seminar by Claude Shannon and Marvin Minsky. This was two years after the two, among others, founded the 'Dartmouth summer research project on AI' [2] - a landmark workshop that formalized the field of artificial intelligence. MH-1 was realized under Shannon, in a period inter-weaved with rapid advancements in robotics and AI between 1957-1974, which also witnessed the birth of the first artificial neural network [3]. Separately from robotic grippers, the foremost work on analytical techniques for hand-object interactions that shaped much of the grasp synthesis research was done on the famed 3-finger hand designed in collaboration with NASA JPL, called the Stanford-JPL hand or the Salisbury hand [4]. Evidently, robotic grasping, artificial intelligence, and space exploration have been inter-connected from their earliest days.

Today, this intersection takes on renewed importance as we enter an era of expanded space operations. The new paradigm of space robotics will require the robots to move towards interaction in semi-structured and un-structured environments. Structure refers to the

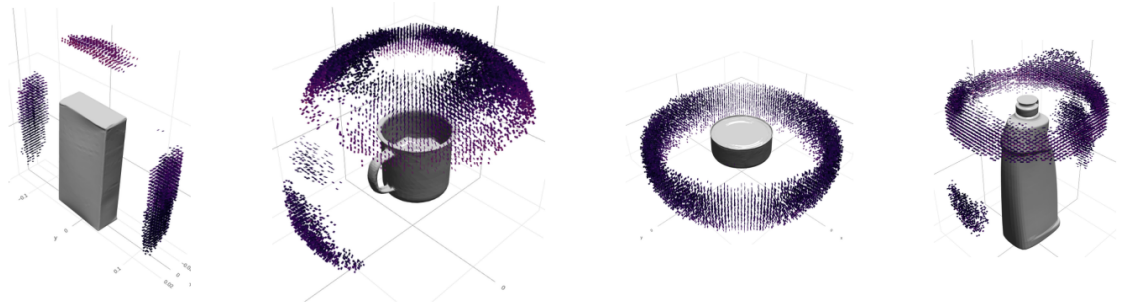


Figure 1.1: Examples of multi-modalilty and discontinuity in grasp distributions around arbitrary objects. Each point represents a successful grasp pose. (Image adapted from [5])

extent of knowledge of the environment models (geometry, materials, dynamics, illumination etc.) and the level of determinism in predicting its evolution. Semi-structured environments have some known structure, while unstructured environments have little to no known structure. From servicing aging satellites to constructing large space structures, from assisting astronauts to supporting long-term surface operations— the need for autonomous robotic manipulation in increasingly unstructured environments is critical. The challenges of robotic operations are further amplified in the space environment, which features sensitive dynamics, harsh visual conditions, communication delays and high reliability requirements.

A fundamental task primitive underpinning robotic manipulation is grasping, which requires a robot to generate and execute restraining contacts on an object. In the real world, this must be done from noisy observations of the object. When the object is known apriori, the task is primarily of localizing the object and contacts. Systems designed only for known objects are inherently limited in both capability and practical use. If robots are to permeate terrestrial and space-borne environments, from kitchens to space stations, they will need to work in an environment that is not completely defined and the objects they encounter will not be a fixed set. This motivates the problem of building grasping systems that will work on real-world objects.

The complexity of robotic grasping in the real world arises from three key factors that span both terrestrial and space applications. These factors are primary and precede specific environmental challenges.

1. **Grasp Distribution**: There is rarely a single ”correct” way to grasp an object and a robot must reason about all possible ways. Objects typically afford infinitely many restraining grasp configurations, with a complex distribution that depends on geometry and material properties. Fig. 1.1 shows examples grasp distribution around objects.

2. **Open-set of Real-world Objects**: Robots operating in unstructured environments must generalize grasping strategies to novel objects never encountered during training.

3. **Task Context**: For long-horizon manipulation grasps must be task-relevant, satisfying both physical constraints of the manipulation and semantic requirements of the task. The robot needs to reason about the subset of task grasps from all possible grasps.

The task of reasoning about the distribution of all possible grasps on an object is called grasp synthesis. Between 1980s to early 2000s, robotic grasp synthesis research was focused on analytical approaches rooted in contact mechanics and closure analysis. These methods provided formal guarantees about equilibrium as a function of contact locations. However, analytical methods proved to be impractical for real-world scenarios due to their reliance on exact models and simplified assumptions on contact dynamics, material properties, and environmental conditions. The limitations of analytical approaches led to increased interest in data-driven methods for grasp synthesis in late 2000s. These methods leverage prior grasping experience, either from trial and error, human demonstration or simulation.

However, storing and transferring grasp experience to uncountable novel objects requires learning complex conditional distributions of grasp configurations on local and global geometry. Initially, data-driven grasp synthesis methods employed manually designed representations and heuristics, which were hard to scale and effective only in limited settings. With the advent of deep learning and availability of large-scale grasp datasets, learning abstract representations and models of arbitrary capacity has allowed a new direction for accelerated progress in grasping. Modern approaches can generate successful grasps directly from partial visual observations of unknown objects, without requiring explicit 3D models or physical property estimates. This capability is especially valuable for space applications where detailed models of targets may be unavailable by relying on learned representations

that maximize the likelihood of a successful grasp given the observation.

1.1 Problem Statement

In this thesis we focus on the problem of vision based 6-DoF grasp synthesis. Given a visual observation V, we want to generate grasps $g \in \mathcal{G}$, where $\mathcal{G}$ is the space of all stable grasps. This space is a non-trivial subset $\mathcal{G} \subset SE(3) \times R^n$ in the space of all possible grasp configurations, where $SE(3)$ is the space of 6-DoF poses of the wrist and n is the number of internal degrees of freedom available to the gripper fingers. We are interested in building models of form $f_\theta : V \rightarrow \mathcal{G}$ or $p_\theta(\, g \,|\, V\,)$, where θ are the parameters of the model. A rich and rapidly evolving body of literature has emerged around this problem with a variety of solutions. Recent advances in the field have leveraged deep learning which provides a remarkably effective framework for parametric modelling of complex functions from data. Despite this progress, several fundamental challenges remain in developing truly general-purpose robotic grasping capabilities:

1. **Representation Learning for 6-DoF Grasp Synthesis**: How can we effectively learn and represent the complex distribution of good grasps for an open-set of objects? Deep generative models are promising for efficiently learning such distributions and availing better generalization outside of training data distribution but are currently challenging to train and underperform in comparison to other approaches.

2. **Sim-to-Real Transfer**: How can we ensure that grasping strategies learned in simulation transfer effectively to real-world scenarios? The reality gap in contact dynamics, sensor noise, and environmental conditions makes this particularly challenging. Moreover, real-world manipulation may require grasping in different environments or subject to a task. Most current methods are applicable only for table-top and bin-picking applications.

3. **Dynamic Environments**: How should grasping systems handle non-static environments with unknown objects? This is crucial for various tasks from human-robot col-

laboration to capturing and servicing tumbling spacecraft. Most solutions for grasping in dynamic environment assume known objects or planar motions.

We focus on the following two research questions aligned with these challenges. These are developed further in Chapter 2 through a comprehensive review of the literature. Detailed research questions are available in Section 2.6.

1. **RQ-1: How can a robot reason about the distribution of successful grasps on objects from imperfect visual inputs in the real world?**
2. **RQ-2: How can a robot reason about an unknown object moving in 6D space without prior knowledge of its structure or motion?**

1.2 Contributions

This thesis proposes an ***object-centric framework for unknown object grasping*** to address these challenges with three main contributions:

1. **A novel generative modeling framework for 6-DoF grasp synthesis using latent diffusion that captures the distribution of good grasps.** This framework enables higher quality sampling compared to other generative models while also providing greater flexibility. The flexibility enables task-conditional generation without re-training the model from scratch. This framework is designed to be environment-agnostic, with evaluation done in zero-gravity environments.
2. **An open-source system that enables reliable transfer of object-centric grasp synthesis models trained in simulation to real-world robotic platforms.** We integrate necessary elements of the pipeline like instance segmentation, motion planning and grasp classification for a modern object-centric grasping stack that is effective and modular. While the stack is designed as ready-to-use application for benchmarking grasp synthesis with real robots, it is readily adaptable other real-world settings.

3. **An approach for simultaneous 3D reconstruction and pose tracking of dynamic unknown objects using 3D Gaussian Splatting.** Our approach uses a unified scene representation that ties reconstruction, tracking and grasp synthesis task seamlessly. The approach is validated on pose tracking of unknown spacecraft in non-natural relative motion, addressing foremost application of grasping in orbital robotics. The approach is general and provides building blocks for moving towards practical 6-DoF grasping systems for dynamic environments.

Our work builds upon recent advances in several key areas. First, the emergence of powerful generative models like diffusion models has enabled learning complex data distributions with unprecedented fidelity. We leverage these advances to better model grasp pose distributions, moving beyond direct regression approaches. Second, development of foundation models have greatly advanced zero-shot performance on visual recognition tasks like segmentation in the wild. We use these advancements to support our object-centric pipelines. Finally, progress in 3D vision, particularly in differentiable rendering and novel view synthesis, has enabled new approaches to high-fidelity 3D reconstruction and camera pose estimation. These can be leveraged to move towards grasping and manipulation of unknown objects in dynamic environments.

Publications

This thesis is based on the following peer-reviewed publications:

1. Barad, K. R., Orsula, A., Richard, A., Dentler, J., Olivares-Mendez, M. A. and Martinez, C. **"GraspLDM: Generative 6-DoF Grasp Synthesis Using Latent Diffusion Models"**, IEEE Access, vol. 12, pp. 164621-164633, 2024,
   arXiv: 2312.11243 , 2023.

2. Barad, K. R., Orsula, A., Richard, A., Dentler, J., Olivares-Mendez, M. A. and Martinez, C. **"Grasp-O: A Generative System for Object-centric 6-DoF Grasping of Unknown Objects"**, Springer Proceedings in Advanced Robotics, 2024. (In Press)

3. Barad, K. R., Richard, A., Dentler, J., Olivares-Mendez, M. A. and Martinez, C. **"Object-centric Reconstruction and Tracking of Dynamic Unknown Objects using 3D Gaussian Splatting"**, in Proceedings of the 1st International Space Robotics Conference, Luxembourg, 2024. arXiv:2405.20104.

Other publications not included in this thesis:

1. Muralidharan, V., Makhdoomi, M. R., Barad, K. R., Amaya-Mejía, L. M., Howell, K. C., Martinez, C. and Olivares-Mendez, M. **"Rendezvous in cislunar halo orbits: Hardware-in-the-loop simulation with coupled orbit and attitude dynamics"**, Acta Astronautica, vol. 211, pp. 556-573, 2023.

2. Makhdoomi, M. R., Muralidharan, V., Barad, K. R., Sandoval, J., Olivares-Mendez, M. and Martinez, C. **"Emulating on-orbit interactions using forward dynamics based Cartesian motion"**, arXiv:2209.15406, 2022.

3. Barad, K. R., Martinez Luna, C., Dentler, J. and Olivares-Mendez, M. A. **"Towards incremental autonomy framework for on-orbit vision-based grasping"**, in Proceedings of the International Astronautical Congress, 2021.

4. Olivares-Mendez, M. A. et al. **"Zero-G Lab: A multi-purpose facility for emulating space operations"**, Journal of Space Safety Engineering, vol. 10, no. 4, pp. 509-521, 2023.

1.3 Thesis Outline

The remainder of this thesis is organized as follows:

- Chapter 2 derives the research questions addressed in thesis through a comprehensive review of robotic grasping literature, from its historical connections to space exploration through modern learning-based approaches. It examines key developments in grasp synthesis, focusing particularly on vision-based 6-DoF methods and the challenges of sim-to-real transfer in both terrestrial and space contexts.

- Chapter 3 introduces GraspLDM, our generative modeling framework for 6-DoF grasp synthesis. We present a novel architecture combining variational autoencoders with latent diffusion models to learn and sample from the distribution of viable grasps. The chapter includes detailed experimental validation in simulation and analysis of the model's capabilities.

- Chapter 4 details our system architecture for deploying grasp synthesis models in real-world settings. We present a modular framework that addresses practical challenges in perception, planning, and control. The chapter demonstrates successful transfer of simulation-trained models to physical robots, with extensive experimental validation.

- Chapter 5 presents our approach to tracking and reconstructing dynamic unknown objects, with particular emphasis on space applications. We introduce a novel framework using 3D Gaussian Splatting that unifies representations between dense reconstruction, 6-DoF pose tracking and GraspLDM-based grasp synthesis. The chapter includes experimental validation in dynamic scenarios of spacecraft in unknown non-natural relative motion.

- Chapter 6 closes thesis with a summary of the research along with a discussion of future directions that extend the work presented here towards real-world grasping in unstructured and dynamic environments.

Chapter 2

Literature Review

2.1 Introduction

Grasping is the fundamental manipulation task of generating restraining contacts on an object. The primary challenge in grasping lies in generating contacts that constrain the object from moving or slipping under unknown external forces. Successful execution of real-world manipulation tasks requires robots to establish secure grasps on objects. The Salisbury hand [4] introduced the first comprehensive framework for modeling robotic grasping kinematics using a three-fingered design. This work led to the development of theoretical foundations based on analytical kinematic, dynamic, and contact models. These analytical approaches primarily addressed the problem of grasp synthesis—generating object contacts given the hand kinematics and object geometry to ensure security under manipulation forces. As robotics moved away from strictly controlled environments, limitations of analytical models were revealed and the research focus shifted towards data-driven methods, which outperformed previous methods in real-world scenarios. A key feature of modern grasp synthesis is the absence of explicit object models and the use of sensor-based observations of objects. This shift in perspective away from theoretical guarantees to pragmatic solutions for tackling the complexity of the real world has been a significant driver of progress in robotic grasping. More recently, deep learning, generative models and large datasets have allowed this effort to scale robot grasping in a manner that tackles transfer of grasping knowledge

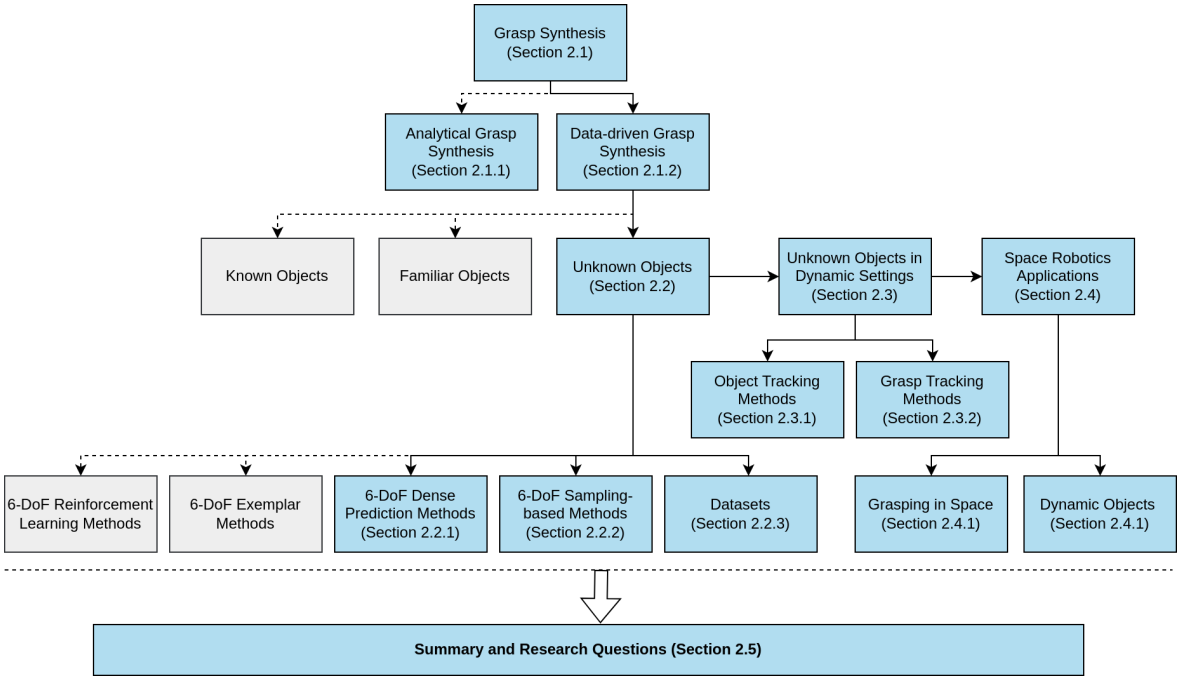
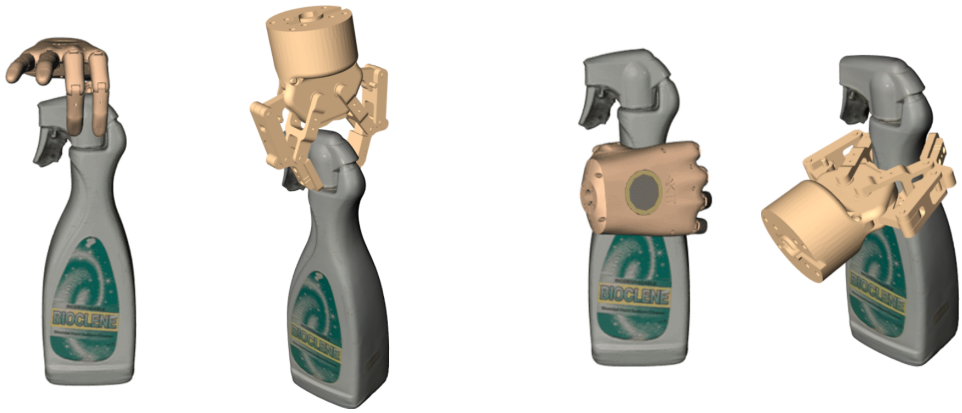


Figure 2.1: Organization of the literature review. Blue boxes represent the topics covered in detail in this review. Gray boxes complete the context of the classifications but are not covered in detail.

or experience to novel and unknown objects. Yet, several aspects of grasp synthesis for autonomous robotic manipulation remain challenging.

This chapter examines the evolution of robotic grasp synthesis as shown in Fig. 2.1, with particular emphasis on vision-based 6-DoF approaches. Section 2.2 provides a broad overview of grasp synthesis methods, examining their classification, historical development, and fundamental limitations. Section 2.3 focuses specifically on visual grasp synthesis for unknown objects, highlighting recent advances in deep learning approaches. This chapter builds upon seminal reviews of analytical methods [6, 7] and data-driven approaches [8, 9], and extends the discussion on relevant works. Readers are encouraged to refer to these works for a more complete coverage of the literature. Section 2.5 examines the state of grasping and perception in space robotics, identifying both parallels with terrestrial applications and unique challenges that remain to be addressed.



(a) (b)

Figure 2.2: Classification of grasps based on degrees of freedom of the hand/wrist [9]. (a) 4-DoF or top-down grasps. (b) 6-DoF or free-space grasps

2.2 Grasp Synthesis

Grasp synthesis methods are broadly classified into two categories based on their approach to generating hand configurations on objects: *analytical* and *data-driven* methods. Analytical methods rely on the kinematics, dynamics, and contact models of the hand, while data-driven methods learn a model from relevant grasping experience data. A secondary classification, particularly prevalent in modern literature, considers the degrees of freedom (six degrees-of-freedom (DoF)) available to the hand/wrist, independent of the finger mechanics. This classification distinguishes between 4-DoF and 6-DoF grasps. A 4-DoF grasp defines a three-dimensional position and an in-plane rotation angle, typically used for top-down grasping scenarios as shown in Fig. 2.2a. In contrast, a 6-DoF grasp represents the general case, defining the complete six-dimensional pose of the gripper as shown in Fig. 2.2b.

The following sections are organized primarily according to the fundamental analytical versus data-driven classification. Section 2.2.1 discusses analytical methods, while Section 2.2.2 examines data-driven approaches. Within these sections, distinctions between 4-DoF and 6-DoF representations are addressed where relevant to the discussion.

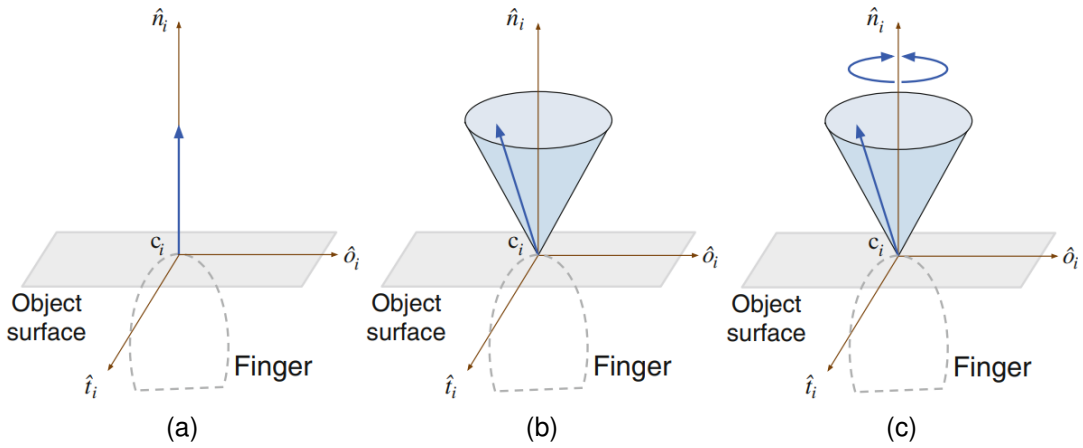


Figure 2.3: Types of contact models used in robotic grasping [10]. (a) Frictionless point contact. (b) Hard frictional contact. (c) Soft frictional contact

2.2.1 Analytical grasp synthesis

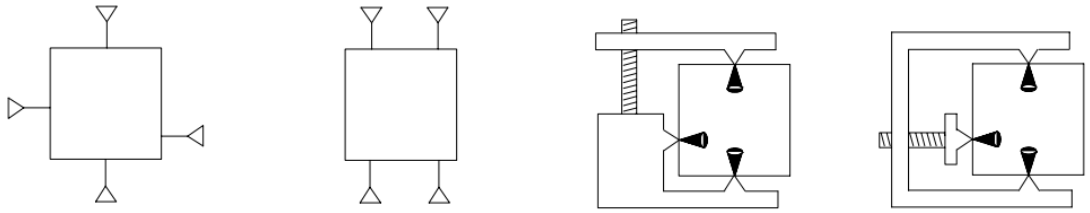
Analytical methods for grasp synthesis are rooted in the fact that desired physical behavior for an object under grasp can be derived from kinematic, dynamic and contact models. The early work in robotic grasping drew inspiration from the analysis of constraining parts in machine design [11]. The focus was on characterizing grasp contacts and their *closure* properties [12]. Each contact of a robot hand or a gripper imparts a set of forces and torques, which are collectively referred to as wrenches. The wrenches that can be applied through a contact depend on the frictional properties at the contact interface. The three types of contact models used for robotic grasping are shown in Fig. 2.3. The simplest is the frictionless point contact model, where only normal forces can be applied. More complex models include frictional point contacts, which can also apply tangential forces according to Coulomb's friction law, and soft contacts which additionally transmit rotational moments around the contact normal.

Closure Properties

Under these wrenches, *closure* properties mathematically characterize the nature of object restraint. A grasp under closure can be maintained against any possible disturbance. The

two types of closure properties investigated in robotic grasping literature are *form-closure* and *force-closure* [6]. A grasp under form-closure is geometrically immobilized, rendering object motion impossible. Form-closure analysis disregards the kinematics and dynamics of the hand, instead assuming the gripper and its finger positions are ideally locked in space. On the other hand, a grasp under force-closure resists disturbances by applying forces at the contacts. Assuming frictional contacts, the forces applied by the fingers generate friction forces that avoid motion or slippage under an external wrench up to a limit. Unlike form-closure's assumption of static locked joints, force-closure assumes the capability to modulate internal forces to influence friction forces and balance external wrenches. In force-closure analysis, bodies are typically assumed to be rigid, and the Coulomb friction model is used. The Coulomb friction model stipulates that maintaining a point contact requires the total force at the contact to lie within a cone, determined by its static friction coefficient, with its apex at the contact point and its axis aligned with the contact normal. For computational efficiency, the friction cone is approximated by a pyramid with a finite number of faces.

Every force-closure grasp is also form-closure, but not vice versa. For form-closure, a minimum of four contacts is required for planar configurations and seven contacts for 6D configurations [13]. Generally, form-closure requires n+1 contacts, where n represents the degrees of freedom of the object. Force-closure can be achieved with fewer contact points than form-closure by exploiting frictional constraints at the contacts. These properties are illustrated for a planar case in Fig. 2.4. In Fig. 2.4a, the object is form-closed as the object is immobilized by four contacts. In Fig. 2.4b, the object is not form-closed as the object still has one degree of freedom. This case is also called partial form-closure. In Fig. 2.4c, the object can be squeezed between the top and the bottom contact using the screw, thus increasing the friction and preventing an external force from moving the object to the right. Therefore, achieving closure with one contact less than in the form-closure case. On the other Fig. 2.4b with identical contact location is not force-closed as the movable contact does not alter frictional forces to prevent rightward motion. For force-closure grasps in 6D configurations, a minimum of three hard friction contacts or two soft friction contacts [12].



(a) (b) (c) (d)

Figure 2.4: Illustration of closure properties of grasps [12]. (a) Static contacts with form-closure. (b) Static contacts with partial form-closure. (c) Controllable contacts with force-closure. (d) Controllable contacts without force-closure

Quality Metrics

Two grasp configurations under force-closure can resist external wrenches very differently. Various analytical quality metrics have been proposed in the literature to evaluate and compare different grasp configurations [14]. Metrics may associate quality of a grasp with the magnitude of external wrenches that can be resisted by that grasp. The quality measures can be associated with the algebraic properties of the grasp matrix [15]. The grasp matrix relates the external wrenches and contact forces, as well as between contact point velocities and object twists [6]. Other metrics can be based on desirable geometry of contacts on objects which include shape of grasp polygon [16, 17], distance of object center of mass from the polygon enclosed by the contacts [18], the effect of finger positioning uncertainty [19] or sizes of independent contact regions [20]. An important category of metrics for real world grasping consider the force limits on finger actuation and seek to minimize the forces that fingers need to apply for various force-closure grasps. These approaches analyze the *Grasp Wrench Space (GWS)*, which is a sub-space of the wrench space and defines the convex hull of all resistible external wrenches under a grasp configuration. The larger the GWS, the larger the external wrench that can be resisted in any possible direction and better the grasp quality. This can be quantified using the radius (ϵ) of the largest sphere in wrench space fully contained in the GWS [21] or with the volume of the GWS [22].

Other metrics may consider the dexterity of the grasp, which is the ability to move the

object in a desired manner post-grasping. Dexterity can be evaluated using the smallest singular value of the grasp matrix quantifying the distance to a singular configuration [23] or the volume of the manipulability ellipsoid [24]. Metrics may also evaluate how well-suited a grasp is for specific tasks, such as by considering the ability to resist task-specific forces or provide fine control in task-relevant directions. A common approach is to analyze the wrench space (the space of all resistible wrenches) through its volume. A more popular metric called ϵ-metric analyzes the radius of the largest inscribed sphere centered at the origin [21].

Grasp Synthesis

Using these closure properties and quality metrics, analytical grasp synthesis is a constrained optimization problem, subject to kinemodynamics of the hand, object geometry and properties, and the desired task specification. One class of analytical grasp synthesis approaches test for possible hypotheses using closure tests [6]. Another class of techniques try to find a single grasp under force or form closure constraints using linear programming [25, 26, 27] or cone programming [28, 29], without considering grasp quality. To incorporate the notion of grasp quality try to find the optimal grasps subject to a quality metric [22]. As this process is computationally expensive, several heursitics are included to make it more efficient [30]. Task relevant constraints can also be included in the optimization.

Limitations

Analytical grasp synthesis methods provide guarantees that are generally desirable for safe manipulation. However, their applicability is limited by the following factors.

1. **Simplifying assumptions**: Dynamics of hand-object contacts can be complex and computationally intractable to solve online. Simplifying assumptions like rigid bodies, point contacts, Coulomb friction and simplified shape models can be brittle in representing the real world [31].

2. **Computation Cost**: The effectiveness of analytical methods is conditional on the fidelity of the models and constraints. For complex objects and tasks, the optimization

problem can be computationally impractical to solve online.

3. **Knowledge of Models**: Analytical methods assume a priori and often exact knowledge of the object properties, geometry and state. This is rarely accessible in the real world.

4. **Performance transfer to the real world**: The robotic tasks in the real world are dynamic in nature, observed through noisy sensors and controlled with limited precision. In addition, force equilibrium through closure is often misconstrued as stability [7, 6]. As a result, grasps scoring high with analytical quality metrics can perform poorly in the real world and human labelled grasps can perform better despite reporting lower quality [32].

2.2.2 Data-driven grasp synthesis

Data-driven grasp synthesis methods address the limitations of analytical approaches by leveraging prior experience to overcome explicit modeling requirements. These methods emphasize robust perception and generalizable representations to maximize success with noisy real-world sensor data and diverse objects.

The progression toward data-driven approaches began with early work on visual perception for known object localization [33]. Visual feature extraction with heuristics and rules was then used to generate grasps on similar objects [34]. A parallel line of work used learning with physical trial and error to develop an online grasping database [35, 36]. Visual recognition was used to identify similar objects and transfer grasps from the database [34], while trial and error were initialized for every new object. Learning was also used to predict grasp quality in [37], where feature-space was pre-defined and the grasp configurations were selected using heuristics. In these works, the testing was reasonably structured and often limited in shape and object diversity. Collecting experience data with a physical robot also limited the density of generated experience data. Data-driven grasp synthesis research was noticeably accelerated by the advent of grasp simulation tools like GraspIt! [38], which allowed for physics-based generation of grasp databases for arbitrary robots and objects.

Bohg et al. [8] classify the huge body of work into three categories based on what these approaches assume to know apriori about the object. The categorization is based on the object instance being grasped: known, familiar or unknown. Below, we discuss the literature in these categories in detail.

Grasping Known Objects

The first category of data-driven grasp synthesis methods assume that the object instance to be grasped has been seen before. In this case, a database of grasps on the known object can be generated apriori. Object recognition and pose estimation is used to localize the object frame. The stored grasps can be transferred to the new object instance by simply transforming the grasp poses to the object frame. The approaches differ in how they generate the grasp database and how they estimate the object pose. Several works tackle the problem of sampling a finite subset of good grasps from an infinite set of possible grasp configurations [39, 40, 41, 42, 31]. These samples are then evaluated in simulations using force-closure conditions and ϵ-metric. This brings up the limitations of performance transfer to the real world [32, 31]. Another alternative is to use human demonstrations to generate the grasp database [43, 44, 45, 46]. Finally, the grasp database can also be generated by trial and error in the real world and combining it with human demonstrations for efficient exploration [45, 46]. Once grasp samples are available for object instances, pose estimation is used to localize the object frame, especially in cluttered scenarios. Pose estimation has been tackled using 2D/3D feature correspondences [47, 48, 49] or learning part-based models [50].

Grasping Familiar Objects

The second category of data-driven grasp synthesis methods assume familiar objects i.e. the query instance isn't known but it shares some characteristics with instances seen in the grasp database. In this case, the goal is to establish similarity with the known object and then transfer grasps by comparison. As object instances may have deviations in geome-

try from the instances in the dataset, discrimination of good and bad grasps also becomes important. The approaches in this category differ in how they abstract similarity to generate grasps. One set of techniques establish discriminative relation between features and graspability. The features used to discriminate good grasp regions from 2D/3D observations may encode local geometry [51, 41], local appearance [52, 53] or global shape [54]. All of these approaches use hand-designed feature space and use the features to classify graspable regions of the object. Alternative set of approaches transfer grasp configurations from a previously seen object to a similar object using part-level representations [55, 56, 57, 58]. Instead of local features based on geometry or appearance, grasps can also be transferred between objects of the same category [59, 60]. These techniques assume that objects with different shapes and appearance within a category afford similar grasps and that category is either known apriori or is retrieved using object detection or semantic segmentation. All of the aforementioned methods relied on manually designed feature spaces with limited generalization. With breakthroughs in deep neural networks with high model capacity, feature extraction could be made more robust.In particular, category-specific semantic keypoints [61, 62] or dense correspondences [63, 64] matching has been used effectively to transfer grasp experience from training data with high success rates. Category-level grasp synthesis leveraging semantic features has the advantage that it can be readily employed for task-specific manipulation beyond pick and place. However, the methods are limited to the trained categories and may not generalize to novel objects that may not be easily resolved into categories.

Grasping Unknown Objects

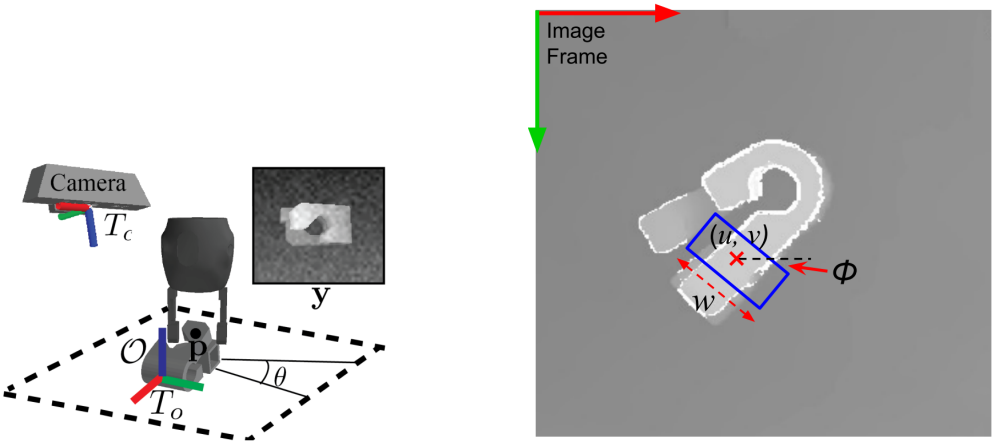
The third category addresses the most general case: grasping completely unknown objects without prior models or established similarities to known objects. These approaches target grasp synthesis for an open-set of objects. The goal is to learn local or global representations that reveal useful structural information about the object from partial and noisy sensor data. These representations are expected to then transfer to novel objects. Early methods relied on heuristic mappings between visual features and grasp candidates [65, 66, 67], often

augmented with auxiliary tasks like shape completion [68, 69, 70] or primitive-based shape approximation [71] to improve grasp synthesis.

For grasp synthesis on unknown objects, learning effective representations from data is a key challenge. Consequently, machine learning has been widely used in this category of approaches to learn arbitrary parametric functions. With proliferation of deep learning in computer vision, it has become clear that (1) deep neural networks are powerful function approximators and (2) the representational power and generalization of such neural networks scale with the capacity and the amount of good quality data. This has spawned a new wave of research where models for tasks like pose estimation, grasp generation, grasp discrimination, grasp execution policy are parameterized as deep neural networks. These models learn abstract representations subject to a task-specific objective or loss function. A significant portion of this research considers parallel-jaw grippers due to their wide-spread availability, simplicity and surprising effectivenes in a broad range of prehensile and non-prehensile manipulation tasks [72]. Consequently, the gripper state is most commonly parameterized as the 6- DoF $SE(3)$ pose of the gripper palm, with optional grasp width as the seventh degree of freedom.

The emergence of deep learning has transformed this approach through two key insights: (1) deep neural networks are powerful function approximators and (2) the representational power and generalization of such neural networks scale with the capacity and the amount of good quality data. This has led to deep neural network parameterization of various tasks: object pose estimation, grasp generation, grasp quality assessment, and grasp execution policies. These models learn abstract representations subject to a task-specific objective or loss function. Much of this research focuses on parallel-jaw grippers, given their widespread deployment and effectiveness across both prehensile and non-prehensile manipulation tasks [72]. Consequently, the gripper state is parameterized as the 6- DoF $SE(3)$ pose of the gripper palm, with optional grasp width as the seventh degree of freedom.

In table-top and bin-picking scenarios, which are prominent grasping applications, the pose parameterization can be restricted to 4-DoF (three for position and one for orientation) by considering only top-down grasps. As most objects in these settings can be conveniently



(a) (b)

Figure 2.5: 4-DoF grasp representation (a) Task space visualization of grasp (Image from [73]). T_c is camera pose, T_o is the object pose, $\mathbf{p}$ is the grasp center point and θ is the angle to the y-axis in the image plane (b) Image space representation of equivalent grasp rectangle in image space (Image from [74]). (u, v) is the 2D image coordinate and w is the grasp width

grasped from the top, restricting the degrees of freedom of the output pose considerably simplifies the learning problem. The grasp is most commonly parameterized as a rectangle in the image space first proposed in [75]. In the rectangle representation, the four degrees of freedom are expressed using 2D image coordinate of the grasp center, grasp width and the angle between the major axis of the rectangle and the y axis in the image space as shown in Fig. 2.5. The grasp synthesis problem can then be framed as a discrimination or generation task. Discriminative approaches try to learn the decision boundary between good and bad grasps or a continuous grasp quality metric corresponding to the probability of success. Several early works sample grasp candidates in the image space or from known object models and use a deep neural network to classify or rank them [76, 73, 77]. Dex-Net [73] is a popular framework which uses a large dataset of 3D object models and parallel-jaw grasps labeled with analytical quality (ϵ-metric [21]) to train a Grasp Quality Convolutional Neural Network (GQCNN). GQCNN model is trained to predict the quality labels in a supervised manner. The grasp candidates to be evaluated are sampled heuristically in image space using normals computed from the depth image. One key limitation of Dex-Net

and similar discriminative methods is the need for expensive sampling in image space to generate grasp candidates for evaluation. In the second class of approaches, one can learn to generate grasps by either directly regress grasp configurations [78, 79] or by generating dense affordance/quality maps [80, 74, 81]. Generative Grasping Convolutional Neural Network (GGCNN) [74] uses fully convolutional network to generate three dense maps for each depth input - quality, angle and width. The availability of per-pixel quality, angle and width values allows selection of grasp location and inference of grasp configuration in a single forward pass. The speed allows this architecture to be used effectively for planar dynamic grasping.

Discussion

It is important to note that supervised learning from large and diverse offline data has superseded reinforcement and imitation learning in 4-DoF grasping works. This may be due to short-horizon nature of this task and relative efficiency of supervised learning as opposed to reinforcement learning which can be sample inefficient and imitation learning which can be brittle to domain shift. This efficiency owes itself partly to architectures that exploit inductive biases offered by the observation space (like CNNs for images) and to the simplicity of objective functions. Another important observation is the increasing success of transferability of models trained on simulated data to physical robots in the real world, moving away from reliance on expensive real-world data collection [82].

We also observe inconsistencies in the literature. Firstly, the term ”generative” is often used to describe models that generate dense affordance maps or quality maps. However, these models are not generative in the sense that they do not model a generative process, rather they are discriminative models making dense predictions.

Rapid advances in 4-DoF robotic grasping have been driven by its economic impact in the fulfillment and logistics industries. The prevalent scenarios involves reliable top-down grasping of objects from cluttered environments, such as bins or table surfaces. Major competitions, notably the Amazon Robotics Challenge, have catalyzed significant algorithmic developments in this field [83, 84]. These advances have been further accelerated by the devel-

opment of comprehensive datasets, including Cornell [76], Dex-Net [73], and Jacquard [85], which have steadily improved in size, quality, and object diversity with a shift towards simulated data generation. With these key ingredients, architectural innovation for efficient, generalizable and sim-to-real transferrable representation learning has enabled greater than 95% success rate in these scenarios [78, 86]. Nonetheless, these systems remain limited for general purpose grasping in countless grasping applications where a canonical view and grasp approach direction cannot be assumed. Even in table-top settings, not all objects in all stable poses are graspable from the top using near-vertical approach. Qin et al. [87] showed that only around 64% of the objects could be grasped from the top (0°-15°) in their evaluation of over 6000 cluttered scenes. We would therefore like to move towards general 6-DoF grasping, which remains an open problem [9]. In the next section we take a detailed look at vision-based 6-DoF grasp synthesis.

2.3 Vision-based 6-DoF Grasp Synthesis

In vision-based 6-DoF grasp synthesis, we are interested in reasoning about all possible grasp configurations around a 3D object. Nominally, this concerns $SE(3)$ pose of the gripper with respect to the object or another frame rigidly attached to the object. 6-DoF grasp synthesis problem concerns generating a subset of poses $\mathbb{H}_g \subset SE(3)$ that generate secure grasps around a reference frame attached to the object ($\mathcal{O}$), given a view of the object $V \in \mathcal{V}$. In the case of dextrous grippers, the grasp configurations may have additional joint or contact parameters. Here, we restrict the discussion to methods designed for grasping unknown objects with parallel jaw gripper. We focus on the learning problem and primarily review deep learning methods that benefit from learning useful representations automatically from data. A key challenge for learning grasp synthesis is non-trivial distribution of grasp poses in SE(3) space for each object, which can have multiple modes and discontinuities.

It is helpful at this point to outline desirable properties of grasp synthesis for an ideal grasping system for unknown objects. We are interested in five key properties:

1. ***Success***: The system should generate grasps that are likely to be successful in the

real world. This is the primary metric for such a system.

2. ***Generalization***: The system should be able to generalize to an open-set of novel objects. It is also desirable to have a system that can generalize to unseen object poses, other illumination conditions, adversarial noise and occlusions.

3. ***Diversity and Mode Coverage***: The system should be able to generate grasps covering all possible modes around the object. As grasp synthesis needs to integrate into the broader manipulation pipeline with reachability and other constraints, it is desirable to have all possible grasp options available.

4. ***Efficiency***: The system should be able to generate grasps within reasonable memory usage and latency. It is desirable to have a system that does not impose extensive preprocessing requirements like multi-view reconstruction, which can be computationally and operationally restrictive. It is also desirable to have an architecture that allows easy control over computational efficiency for different applications. Efficiency also extends to offline development of such a system like training and data collection, wherein leveraging simulation is desirable for scalability.

5. ***Flexibility***: The system architecture should be adaptable to task and operational variations. It is desirable to avoid training and developing distinct models for different tasks with the same gripper. At the same time, we are interested in the ability to extend the architecture to different gripper types and grasp representations.

A large number of approaches have been proposed to tackle these problems with numerous variations in the choice of input, grasp representations, architecture, hardware and validation. The input can be a 2D image, 2.5D RGB-D image, 3D point cloud, voxel grid, neural field or a combination of these with additional information (like normals or semantic features). In terms of grasp representations, the output can be the $SE(3)$ pose of the gripper, a contact-based representation or an abstract multi-view grasp region representation as shown in Fig. 2.6. The architecture can be a convolutional neural network, point-set network or a transformer. The validation may be done in simulation, or on a real robot, with

singulated objects or in cluttered scenes. Classifying these methods based on the aforementioned variations is challenging. Instead, a more meaningful classification can be made based on the formulation of the learning problem.

We can classify deep learning-based 6-DoF grasp generation methods into one of four following formulations, where θ will generally represent the learnable parameters in the respective models.

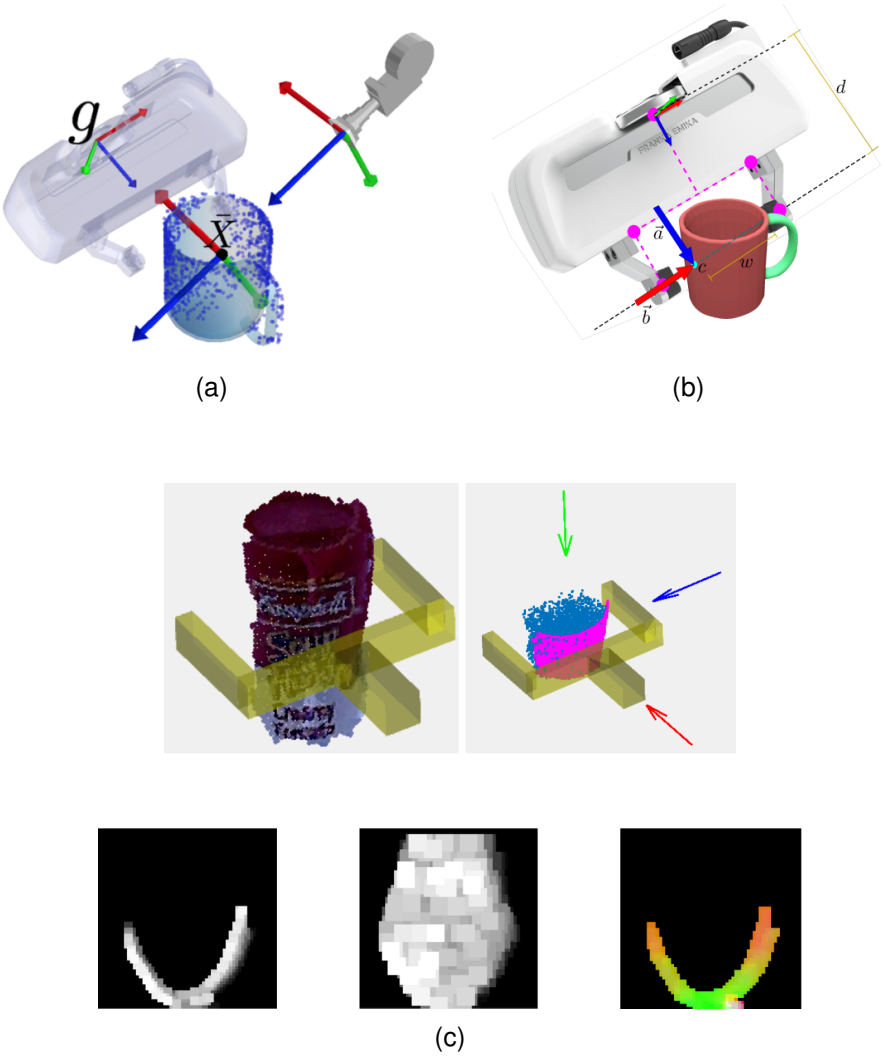
1. **Dense Prediction Methods**: These methods learn a model $f_\theta : \mathcal{V} \rightarrow \mathcal{A}$, where $\mathcal{A}$ is the affordance space. The model predicts some measure of graspability or affordance in the observation, like grasp score per pixel in an image. Alternatively, they may directly learn the feed-forward map $f_\theta : \mathcal{V} \rightarrow \mathcal{P}_{set}(\mathcal{H}_g)$, from the observation space to a power-set of possible grasp poses. In the former case, the affordance is typically expressed using a single or multiple scalar values assigned densely in the discrete observation space i.e. $\mathcal{A} = \mathbb{R}^{D \times M}$ where D represents the dimensions of a discrete observation space e.g pixels, voxels or points and M represents the affordance dimensions. Affordance can also be stored as a continuous field $g_\theta : \Omega \rightarrow \mathbb{R}^M$ using neural fields [88]. The values generally represent how graspable an object is at a query location but may encode other features. These dense maps can then be used to search for remaining gripper parameters like orientation and width for best possible grasp at each location. This process may be single stage or multi-staged. Such dense prediction methods generally produce one grasp per query location and are therefore limited in their coverage of the grasp distribution.

2. **Sampling-based Methods**: These methods generate grasp poses by sampling from a distribution and evaluating the quality of the grasp. Sampling can be done explicitly from a desired distribution, e.g. in the Euclidean space or SE(3) space, and optionally using heuristics to guide the sampling. Alternatively, sampling can be done more efficiently from a learned latent space with generative model $p_\theta(H_g|z, V)$, where z is latent that can be sampled from a prior. If the latent space is structured, the grasp distribution is stored efficiently in a low-dimensional space which can be sampled flexi-

bly. These models also implicitly ignore regions of observations that are not graspable, making them more efficient for generation. The grasp generator stage is followed by a grasp evaluator which is typically a learned discriminator $f_\phi(s|H_g, V)$ that scores the grasp for quality or probability of success ($s \in [0, 1]$). The challenge in these methods is training a regularized and structured latent space.

3. **Reinforcement Learning Methods**: Reinforcement Learning (RL) methods typically focus on control and grasp execution under unstructured observations rather than synthesizing multiple candidate grasps. The goal is to learn a control policy by interacting with a real or simulated environment through trial and error. The policy $\pi_\theta : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ maps from state space to action space ($\mathcal{A}$ or the space of probability distributions over $\mathcal{A}$) to generate a sequence of actions towards a successful grasps. The state space S is the set of all possible configurations of the environment. The action space $\mathcal{A}$ is the set of all possible actions/controls that the agent can take. The view V represents a partial observation of the state. In manipulation scenarios like grasping, the action space commonly corresponds to the reachable pose space ($\mathcal{A} \subset SE(3)$). However, state and action parameterization vary greatly across approaches. System dynamics evolve according to a partially observable Markov decision process, and the policy is optimized to maximize expected rewards over sampled grasping trajectories. The learning process can be sample inefficient and requires intricate reward design. They also don't offer explicit control over grasp selection and are difficult to evaluate for coverage.

4. **Imitation Learning Methods**: These methods learn grasping policies from expert demonstrations $\tau = (s_t, a_t)_{t=1}^T$. The policy $\pi_\theta : \mathcal{V} \to \mathcal{A}$ can be learned through direct behavioral cloning or by first inferring a reward function $r_\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ via inverse reinforcement learning. Actions typically parameterize end-effector trajectories in $SE(3)$, with policies trained to replicate expert grasping strategies while managing compounding errors and generalization challenges. These methods are limited by the availability of expert demonstrations and brittle generalization to novel scenarios.



(a) (b)

(c)

Figure 2.6: Grasp parameterizations for 6-DoF grasps of a parallel jaw gripper. (a) $SE(3)$ pose of the gripper [89]. The object frame is set as the camera aligned frame ($\bar{X}$) at the centroid of the partial point cloud. The grasp is described by the relative pose between the gripper frame g and $\bar{X}$. (b) Contact-based reduced dimension parameterization [90]. Assuming a contact point is visible Vectors $\vec{a}$ and $\vec{b}$ represent the approach (c) Three image representation aligned with the Darboux frame of the grasp [91]

In the following discussion, we exclude reinforcement learning and imitation learning methods from further discussion. In both cases, explicit control over grasp selection is not available or limited. Reinforcement learning methods also suffer from sample inefficiency and reward design challenges during training. Imitation learning methods on the other hand are limited by the narrow domain of expert demonstrations and brittle generalization to novel scenarios. We review the state-of-the-art in dense prediction and sampling based methods.

2.3.1 Dense Prediction Methods

Regression of 6D grasp pose directly from RGB-D images was demonstrated in [92] using a convolutional neural network, albeit for a single grasp pose. Qin et al. [87] demonstrated regression of multiple grasps of a parallel jaw gripper from a single view of an object clutter. They use PointNet++ [93] as the backbone to simultaneously regress grasp poses and predict per-point grasp quality map. Their method is able to generate grasps with 77% success rate on four cluttered scenes of up to 10 out of 30 unseen objects in the training set. Zhao et al. [94] extend this work by adding additional stages that predict approach direction and approach angles. Fang et al. [95] propose a new dataset (GraspNet-1billion) containing 1.2 billion grasp annotations of 88 objects in clutter along with a multi-stage network to generate 6-DoF grasps from a single-view point cloud. PointNet [96] backbone is used to extract per-point features and predict graspability of the point and quality of the pre-defined set of approach vectors relative to that point. The approach vectors are then grouped into cylindrical neighbourhood regions and the in-plane rotation, gripper width and grasp quality are predicted for each region using multi-label classification. Additionally they predict tolerance of each grasp to predictions. The evaluation reports average precision in the test scenes of their dataset with respect to the ground truth labels. No real tests or success rate is reported. Such multi-stage networks require supervision on intermediate outputs in addition to the grasp pose. Gou et al. [97] use the same dataset and take a similar approach of decomposing orientation prediction into two prediction stages, one each for an approach vector and an in-plane rotation. The latter stage uses heuristic search instead of learned network. They suggest that using an RGB images makes grasp predictions more robust in

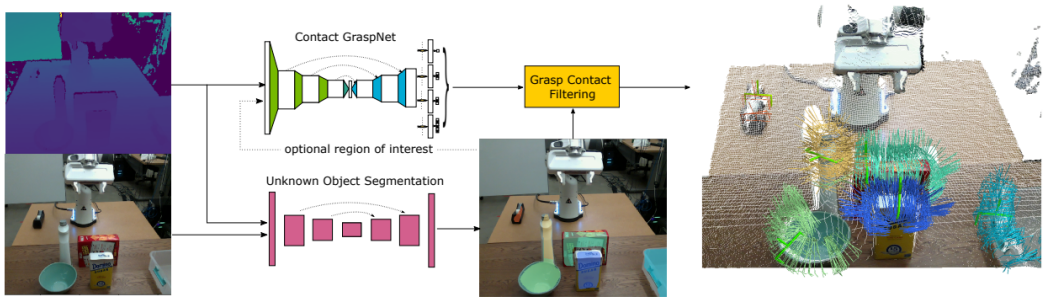


Figure 2.7: Point-based grasp synthesis pipeline in Contact-GraspNet [90]. The network directly regresses a reduced dimension representation of the grasp pose using a PointNet++ backbone.

the presence of noise in depth images compared to [98].

Sundermeyer et al. [90] highlight that approach vector predictions are ambiguous per contact and may not capture thin, curved and hollow regions on the objects well. Instead, they propose a contact-based reduced dimension representation that is directly regressed using a PointNet++ [93] backbone as shown in Fig. 2.6b. This representation is shown in Fig. 2.6b. They directly supervise positive contact points and gripper width, while using additional control point supervision as a proxy for the grasp pose. The network is trained on table-top scenes synthesized from the ACRONYM [99] dataset, which is generated in simulation. While meant for cluttered point cloud as an input, it can use object masks to generate higher density grasp candidates around individual objects. They demonstrate 90% success rate in real world tests significantly outperforming the relevant sampling based methods on 9 cluttered scenes containing a total of 51 novel objects.

Apart from point-based representation, volumetric representations have also been used for dense prediction. Breyer et al. [101] propose Volumetric Grasping Network (VGN) that takes a voxel grid containing truncated distance to the nearest surface. This TSDF representation is integrated from six views around the clutter and predict grasp quality, orientation and width per voxel. The training data is generated in physics-based simulation and transfer to real world demonstrating 80% success rate in tests containing 12 unseen objects in 10 random cluttered scenes. As opposed to explicit volumetric representations, one can also encode shape information with implicit representations. When parameterized as neural net-

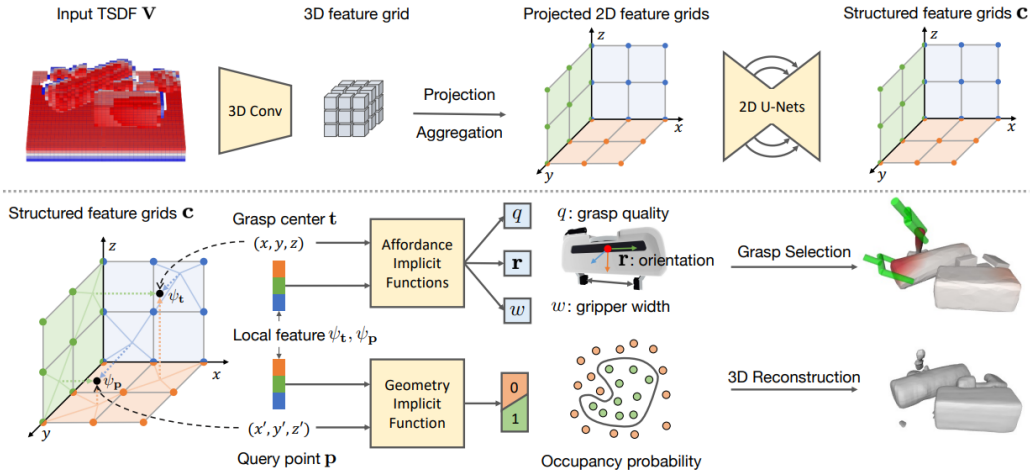


Figure 2.8: Volumetric grasp synthesis pipeline in Grasp detection via Implicit Geometry and Affordance (GIGA) [100] that learns to extract structured features in a 3D grid from a TSDF of the scene. The features are used to predict grasp quality, orientation and width per voxel using neural fields.

works, they are called neural fields as they represent a continuous field of values. Neural fields are appealing are appealing for 3D reconstruction and geometric understanding as they are continuous and differentiable. Neural fields have gained popularity in various visual computing tasks recently [88] Van der Merwe et al. [102] first proposed learning a neural Signed Distance Function (SDF) to aid grasp quality prediction for a multi-fingered hand. They use synthetic data to first train a neural SDF model. The global shape embedding from the feature extractor is then used to predict grasp quality, for a sampled configuration. They configuration for which they predict grasp quality contains gripper pose and arm joint angles. Since the neural SDF is continuous and differentiable, configuration in collision can be moved to a non-collision region using gradient descent on desired configuration parameters.

Instead of learning a global descriptor, Jiang et al. [100] propose Grasp detection via Implicit Geometry and Affordance (GIGA) which extract a feature per voxel in a voxel grid enhancing local context. They suggest that simultaneously learning geometry prediction and grasp prediction using neural fields conditioned on local descriptors improves grasp generation. The pipeline is shown in Fig. 2.8. The TSDF is generated from a single-view

image and 3D features are extracted per voxel using a 3D convolution network. Neural fields are then used to (1) predict grasp quality, orientation and width and (2) predict occupancy, per voxel using the positional embedding of the local 3D feature and the voxel's 3D location.

The main limitation of using neural fields to encode geometry is that the highest affordance locations are not explicitly available and need to be sampled. The computational cost of architectures like GIGA grows significantly with the resolution of the input feature grid, both for feature extraction and for evaluating implicit functions at each location. This limitation extends to other methods that use voxel-grids and 3D convolution architectures. Overall, all the methods discussed in this section are limited in their coverage of the grasp distribution as they generate only one grasp per query location. More recent works like Implicit Grasp Diffusion (IGD) [103] has sought to improve the multi-modal coverage of grasps in architectures like GIGA. Instead of directly regressing orientations per voxel, IGD using a diffusion model to conditionally generate a suitable orientation [103]. However, sampling complexity issues remain pertinent.

2.3.2 Sampling-based Methods

Early sampling-based methods used Euclidean sampling to generate grasp candidates around an object and then evaluated them using a data-driven discriminator that predicts the grasp quality. Translation component of the grasp pose can be sampled around points of a point cloud. The approaches vary in how they sample the grasp orientations at that location. The orientations can be sampled randomly [104, 105, 106, 107] or with the aid of surface normals [108, 109]. To reduce the number of grasps to be evaluated, infeasible grasps in collision or in free space without any points in the finger sweep region can be discarded based on heuristic checks. ten Pas and Platt [110] use a set of geometric constraints for generating samples on an object point cloud more efficiently. The grasp poses are sampled for each point in the point cloud, such that for each sample pose the body of the gripper is not in collision and the closing plane of the hand is parallel to a plane defined by the surface normal at that point and the principle direction of minimum curvature at the point. The frame defined by these two directions is called the Darboux frame. To classify the candidates, they

use a descriptor based on Histogram of Gradients (HoG) features and train an SVM model on a dataset of 18 objects. On single-view point clouds, they report 85% success rate on 23 test objects and 66.7% success rate on seven objects in the test set that they consider 'hard to see'. ten Pas et al. [91] improve on this approach by using a deep convolutional neural network instead of hand-crafted features for classifications. They use the same sampling strategy and feed the classifier with three views of the object from three principal directions of the Darboux frame. The proposed method, called GPD, is the first 6-DoF grasp generation method that is widely used in the community. This is because the authors provide an open-source ROS package with integration in the MoveIt! [111], a popular motion planning framework. The sampling strategy was also utilized in other works [112, 113, 114]. The main limitations of these methods are the latency for sampling candidates and the need for a large number of samples to cover the grasp distribution. Additionally, they require normal computation which adds to the computational complexity and is challenging to compute accurately for noisy point clouds and especially on thin and hollow regions of the objects like mug rims, plates and bowls. To make the candidate sampling efficient, one can model the grasp generation problem as a generative process and learn a distribution of successful grasp poses.

Generative Models

Veres et al. [115] first proposed using a deep generative model to address multi-modal grasp coverage and dense sampling. Although they use a multi-fingered hand and evaluate in simulation, they show that a Conditional Variational Autoencoder (CVAE) [116] with a convolutional backbone can more effectively learn a distribution of grasp contacts conditioned on input RGB-D images compared to other methods including feed-forward CNNs. Mousavian et al.[89] to learn a point cloud conditioned distribution of successful parallel-jaw grasps in a continuous latent space. The advantage of learning a low-dimensional latent space, if it is well-structured, is that sampling can be done very efficiently and continuously. The CVAE encoder is based on PointNet++ and the input is the partial point cloud where each point is associated with feature vector which contains grasp pose. The decoder takes in a randomly

sampled latent along with the partial point cloud. The decoder conditions the latent on the point cloud information and generates a grasp pose parameterized by three elements of translation and four elements of unit quaternion. While the VAE is successfully able to cover multiple grasp modes, VAE alone provides a large fraction of unsuccessful grasps when executed. Consequently, additional online optimization stages are used for pose refinement. Murali et al. [117] builds upon [89] with an additional stage of collision checking using a learned network. The main limitation of these methods is complexity of training, inefficiency of the architecture and lower success rates compared to dense prediction models [90]. The lack of performance may be justified by limitations of a naive VAE that have been highlighted in the generative modelling literature [118]. Several solutions like KL annealing have been proposed [119] to improve sample quality in VAEs, but have not be employed in the context of grasp synthesis.

Recently a new class of generative models called Denoising Diffusion Probabilistic Models (DDPM)[120] have gained significant popularity for learning complex distributions for high-fidelity image and point cloud generation. Score-based Generative Model (SGM) is an equivalent formulation of diffusion models based on Tweedie's formula [121], where model learns to represent the data distribution with a Fisher score function and generates sample through Markov Chain Monte Carlo techniques like Langevin dynamics. Both DDPM and SGM can be generalized with Stochastic Differential Equations (SDE)s [122] and are equivalent formulations differing in implementations. Intuitively, the generative process is a sequential process, which at each stage takes a noisy sample and de-noises it such that the log likelihood of the data increases. Diffusion models can also be seen as an extension of hierarchical VAE with Markovian relation between hierarchical latents of identical dimensions. Diffusion models are highly expressive and overcome limitations of VAE like posterior collapse and prior gap. They are also significantly simpler to train. For grasp pose generation, Urain et al. [123] use the SGM formulation and learn a scalar field that represents the energy of the data distribution. Score is represented as the gradient of the energy function. This kind of scalar cost formulation allows grasp synthesis to be included in multi-objective optimization problem. In this case, they show joint grasp and motion optimization. From

a grasp synthesis perspective, their models demonstrate refinement of randomly sampled SE(3) grasp poses to low-cost (good grasp) regions. However, the analysis only presents learning on full point clouds of a single category and the real-world tests use ground truth pose of the object. Further, Langevin-type sampling is slow and the model needs to be re-trained for downstream tasks. Diffusion models have also been utilized for dexterous grasp synthesis [124], but rely on the availability of the object model.

Overall, generative models are appealing as they can learn a data generating distribution and hopefully also the causal factors affecting the generation process. They also tend to be efficient learners in limited data settings with better generalization. Latent variable models also facilitate efficient sampling. However, they have not been investigated comprehensively. Recent dense prediction models have criticized sampling based methods for being slow. However, CVAE in [89] without post-processing can generate grasp candidates and classify them under 100ms unlike other sampling-based methods. Diffusion models are also promising for improving coverage and accuracy of grasp synthesis. Another clear limitation of existing approaches is the focus on task-relevant grasp synthesis that can aid long-horizon manipulation tasks. Very few works tackle task oriented grasping [125, 113, 126] and are detached from unconstrained grasp synthesis models. This means that we need to choose between unconstrained and task-oriented grasping models. Moreover, each has to be trained from scratch when task context changes. Given that task-specific grasps are a subset of all possible grasps, it is desirable to have a model that is amenable to both unconstrained and task-specific settings and does not require a different architecture or expensive training from scratch. Conditional generation with latent diffusion models [127] present some promise in this direction. Consequently, there is a need for exploring the potential of generative models and the design space between VAE and diffusion models for accurate, multi-modal and flexible 6-DoF grasp synthesis.

2.3.3 Datasets

Datasets are a crucial component in all the aforementioned methods. This data can be acquired from trial and error in the real-world, manual annotation or simulation. The first two

Dataset	Observ.	Labels	Grasps	Objects (Cat.)	Grasps/ Obj.	Scenes
Parallel-jaw grasps						
Dex-Net 1.0 [73]	Sim	Analytical	2.5M	13252	≈250	Single
Eppner *et al.* [5]	–	Sim	1B	21	47.8M	Single
GraspNet 1-Billion [95]	S+R	Analytical	1.1B	88	12.5M	Multi
EGAD! [128]	Sim	Analytical	233k	2331	100	Single
ACRONYM [99]	Sim	Sim	17.7M	8872 (262)	2k	Single+Multi
Multi-finger grasps						
Columbia [40]	-	Analytical	238k	7256 (161)	≈32	Single
Kappler *et al.* [129]	Sim	Manual+Sim	300k	700 (80)	≈430	Single
Veres *et al.* [115]	Sim	Sim	50k	N/A (64)	N/A	Single
Unigrasp [130]	Sim	Sim	2M+	1000	N/A	Single
DexGraspNet [131]	Sim	Sim	1.3M	5355	N/A	Single
Grasp'D-1M [132]	Sim	Sim	1M	2350	N/A	Single
MultiDex [124]	Sim	Sim	436k	58	N/A	Single

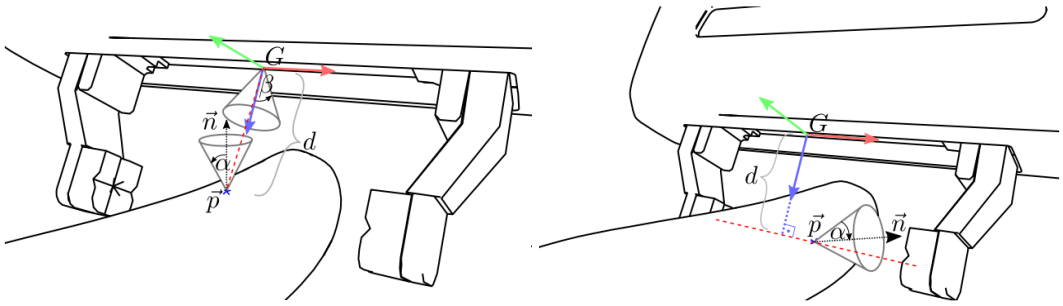
Table 2.1: Comparison of datasets for 6-DoF grasp synthesis

sources are expensive and harder to scale for effectively training general-purpose deep neural network models. Moreover, the need for high-resolution grasp coverage supports the use of simulation where grasp configurations can be sampled with high resolution. Consequently, simulation has been the most common approach for generating large scale datasets. Most commonly used datasets for 6-DoF grasping are enumerated and compared in Table 2.1.

Columbia [40] was the first large dataset of 6-DoF grasps featuring 238,737 form-closure grasp annotations on 7256 objects from Princeton shape benchmark [133]. Among these grasps, 171,158 are annotated for 3-finger Barett hand and the rest for a human hand. The grasps are sampled from a low-dimensional subspace called eigengrasps [134] and evaluated for analytical ϵ-metric and volume metric for GWS from [21]. New samples are generated as random samples in the neighbourhood of existing good samples using simulated annealing as in [134]. The grasp annotations are associated with 3D objects, instead of visual observations, requiring off-the-shelf renderer for training vision-based models.

One common strategy to sample candidates is to guide the sampling based on approach direction. Most commonly, a point on the object is sampled and the approach direction can be set to the direction of the surface normal. Alternatively, the approach vector can be more

flexibly sampled by relaxing the direction to be constrained by two cones (α and β) at the gripper frame and the query pont respectively as shown in Fig. 2.9a. A stand-off distance from the gripper surface is chosen to avoid collision and the hands rotation around the approach vector is sampled uniformly. Kappler et al. [129] use this scheme to create a dataset for 6-DoF grasps for the 3-finger Barett hand with only one pre-shape configuration. They use OpenRAVE [135] simulation to generate point clouds and surface normals from object meshes. The query point is sampled by casting rays from a bounding box and selecting the intersections. The approach direction is set to the direction of the surface normal and a total of 16 grasps are annotated per point. The grasps are labeled with analytical ϵ-metric as well as a physics-based stability metric that simulate finger contact forces and collision under perturbation. The dataset provides 300,000 grasp annotations on 700 objects of 80 categories. The physics-based metrics are validated using human feedback on images of grasp annotation, assuming humans to be optimal predictors. They establish high correlation between success predicted by the physics-based stability metric and human prediction, justifying the use of simulation for scalable grasp data generation. The dataset is limited in grasp pose density (as rotations are sampled at 45 deg intervals per point) and assumes a single hand pre-shape configuration. Additionally, sampling grasp poses with normals is not reliable on thin and hollow regions of objects like mugs and plates. Veres et al. [115] propose a similar simulation-based approach but with a sampling scheme that samples grasp at a finer resolution. They use Barett hand in V-REP [136] simulator to label 50,000 grasps on objects from 80 categories. This dataset uses a parallel finger pre-shape configuration for the Barett hand. However, since the contact computations in simulation use multi-finger contacts, we classify it with multi-finger grasping datasets in Table 2.1.The approach-based sampling scheme is used most-commonly in multi-finger grasp data generation. However, simpler methods like uniform sampling have also been used. More recent work like Shao et al. [130] scale the multi-finger grasping data to 12 different two and three fingered hands and 1000 objects. The sampling is done by uniformly sampling two and three finger contacts in a point cloud, filtered with heuristics, checking force closure condition and using inverse kinematics of the hand to generate gripper configuration. Wang et al. [137] use a five-fingered



(a) (b)

Figure 2.9: Representations of grasp sampling schemes used for dataset generation [5] (a) Approach-based sampling (b) Antipodal sampling

shadow hand and use differentiable optimization with initialization heuristics. Li et al [124] use similar approach to build MultiDex dataset that provides annotations for five different grippers with between two to five fingers. Turpin et al. [132] develop a differentiable simulation that allows gradient based optimization. Using this they generate Grasp'D-1M dataset of one million grasp annotations across three grippers with two, three and four fingers.

For parallel jaw grasping Fang et al. [98] use approach-based sampling to generate Graspnet-1billion. The dataset contains grasp annotations on cluttered scenes of 88 objects combining existing datasets and other novel objects. The grasp pose sampling is done in simulation using mesh models and each grasp is associated with an analytical score that corresponds to (1.1 - μ), where μ is the friction coefficient discretrized and sampled at intervals of size 0.1. This means that the grasp with force-closure at the lowest friction coefficient is a better one. The sampled grasps are then transferred to real world cluttered scenes. Each cluttered scene contains 10 random objects and RGB-D images are collected from 256 view points from each camera and 6-DoF poses of the objects relative to the camera in the clutter are annotated for each view. The sampled grasps are transformed and filtered for collision. This results in 1.1 billion positive and negative grasp annotations on around 95000 RGB-D images of 190 cluttered scenes. They also introduce average precision metric for quantifying grasp generation performance that is the precision of top-K grasps subject to a friction coefficient. This evaluates how much coverage grasp generation provides for force

closure grasps at various friction levels.

Another approach is to sample points on the surface of the object and search for a point with the farthest intersection on the object such that the line connecting the two lies within the two friction cones (α) as shown in Fig. 2.9b. This scheme is called anti-podal grasp sampling. This approach is most common for parallel-jaw grasping and difficult to scale for multi-finger grasps as the mapping from gripper configurations to contact locations is not bijective (one-to-one). For parallel jaw grippers, Dex-Net 1.0 [138] was the first large dataset with 2.5 million grasp annotations on around 13,000 objects. The grasp candidate sampling is accomplished by randomly sampling a point on the surface of the object followed by random sampling of the finger sweep direction. The candidate is filtered with anti-podal check which asserts that the line joining the contacts lies within the friction cones of the contacts. The grasp quality is then labeled as the probability of force closure. Similar to [129], the quality label takes into account the real world uncertainty that arise from sensor noise, imperfect calibration and actuator precision. The dataset does not contain any observations and coverage of the grasp distribution is limited by the random sampling strategy. This dataset is extended in Dex-Net 2.0 [73] by including depth images, but is restricted to planar 4-DoF setting. Morrison et al. [128] use the same approach as Dex-net 1.0 but focus on generating a dataset of grasps on adversarial object shapes using evolutionary shape synthesis.

Eppner et al. [5] study the effect of sampling scheme used to generate a dataset on grasp coverage and precision. They outline three results from their analysis on various schemes. First, heursitics used in sampling introduce high bias and suffer more from lack of grasp coverage. Second, the anti-podal scheme provides noticable more coverage with fewer samples than approach based at lower number of grasp samples (¡100,000). On the other hand, approach-based schemes reach significantly higher coverage asymptotically. Third, the probability of grasp samples is higher with less samples using anti-podal sampling, while approach-based sampling again benefits from asymptotically higher probability of success at high number of samples. Using these insights, Eppner et al. [99] build ACRONYM dataset. The dataset contains 17.7 M parallel-jaw grasp annotations on 8872 objects from 262 categories of ShapeNetSem [139] dataset. ShapeNet dataset is one of

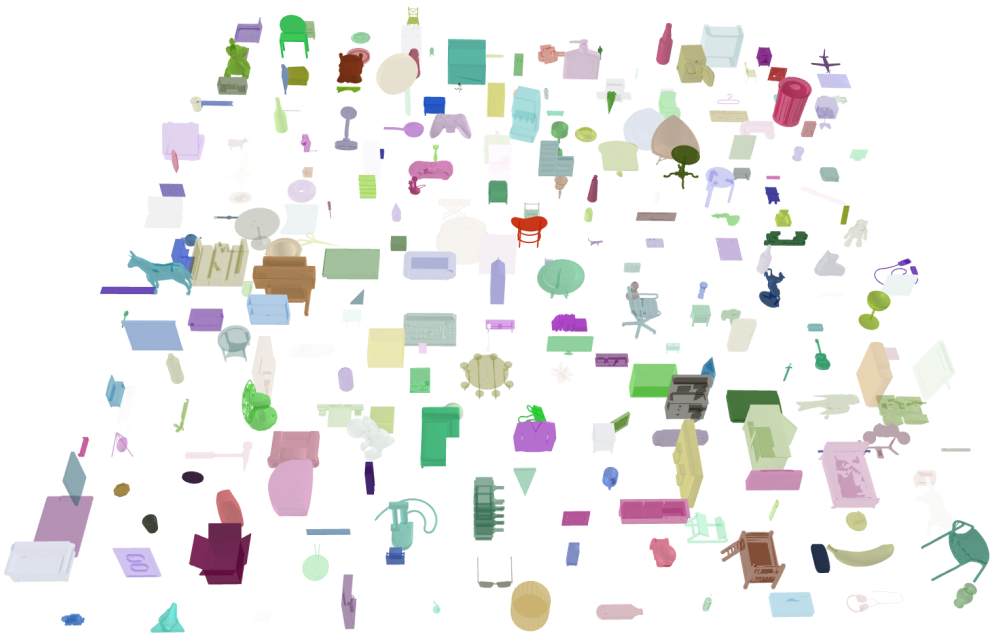


Figure 2.10: Examples of diverse object shapes from ShapeNetSem data used in ACRONYM dataset [99]

the most diverse and widely used datasets for 3D vision and learning. ShapeNetSem is a subset with physically meaningful parameters associated with the objects. Figure 2.10 shows examples of object shapes in the dataset. Grasps are sampled using the antipodal scheme and evaluated in FleX [140], a particle based physics simulation. The evaluation starts with a pre-grasp state at the grasp pose with maximum gripper width corresponding to the Franka hand. The fingers are closed using force-control up to a force threshold. Finally, the object is shaken along all axis to assert stability. Stability is a more desirable property compared to static equilibrium as asserted by force-closure tests used in other datasets. The use of FleX for generating grasping data has also been validated in a separate study by Danielzcuk et al. [141], which demonstrated 86% (highest) average precision of grasps labelled by FleX from 2625 attempts with a real robot. Each object instance in ACRONYM contains 2000 grasp annotations based on success or failure in the simulated evaluation. The authors also release a libary that supports procedural generation of cluttered scenes,

collision checks and rendering. This allows versatality to assess both object and scene-level grasping with or without cameras. This flexibility along with diversity of object categories and density of grasp annotations makes ACRONYM a unique dataset for studying vision-based grasp synthesis. Sundermeyer et al [90] show that models trained on this dataset transfer well to the real world.

2.4 Dynamic Grasping

A key limitation of all the aforementioned methods is that they are restricted to static objects. They assume that the object does not move between acquisition of the visual observations and the conclusion of grasp execution. In many settings, robots must grasp objects that are moving. An important application is human-robot handovers [142] which is relevant for terrestrial and space-borne settings. In other cases, objects may be moving due to natural dynamics of the environment and/or external forces. For instance, grasping objects on conveyor belts [143], grasping objects floating on-board ISS [144] or grasping spacecraft under relative orbital motion [145]. The literature is relatively sparse for dynamic grasping. Houshangi [146] first described a system for grasping of moving objects. Kragic et al. [147] demonstrated a method for dynamic multi-finger grasping of a known object using feature-based visual servoing and in-the-loop generation of reachable force-closure grasp using GraspIt! [38]. Several works have followed this scheme with known objects [148, 149, 150, 151], where grasp generation can be done apriori and transformed per online pose estimate. In this case, planning and control are reactive but the perception is not. For unknown objects, the perception system needs to be reactive, generating and improving grasp predictions in the loop with incremental visual observations. Broadly, we can classify dynamic grasping methods for unknown objects into two categories depending on whether grasps or object is tracked.

2.4.1 Unknown Object Grasp Tracking

These methods extend single-frame grasp generation methods discussed in Section 2.2 with temporal tracking of the grasps or semantic grasp features in subsequent images. Morrison et al. [74] use their GGCNN in a closed-loop grasp tracking system with pose-based visual servoing. Since GGCNN inference is fast (6ms), they repeatedly generate grasps for each depth frame at 30Hz. At each subsequent frame, top quality grasps are filtered around the maxima of the quality map and select the one closest to the grasp chosen at the previous frame. This system works demonstrates between 80% and 90% success rate in various settings. However, it is restricted to planar motion of the object as GGCNN is 4-DoF grasp prediction model. In such planar settings, the view of the object stays relatively similar across the image sequence and grasp pose variations are minimal, allowing consistent tracking of a single grasp. However, the view of the object can change rapidly in 6-DoF motion, occluding initially visible regions of the object and revealing others. This grasps and their predicted quality on visible surfaces may change rapidly. Moreover, the grasps predicted later in the image sequence may cover a different mode of the grasp distribution compared to the initially generated grasps. In 6-DoF dynamic grasping, these aspects must be taken into account. Rosenberger et al. [152] extends this approach to human-to-robot handovers in free space with additional object detection and hand segmentation. However, the grasp object motions in the experiments are limited by planar grasp prediction capability of GGCNN.

Yang et al. [153] address dynamic 6-DoF grasping for human-to-robot handovers by extending the 6-DoF-GraspNet model from Mousavian et al. [89]. A camera mounted statically observes the workspace containing the human, the object and the robot. It crops the region of the workspace with object heuristically using a hand tracker followed by a segmentation network to segment the object point cloud. This point cloud is used to generate grasps once. Assuming minimal motions of the object, they refine the grasp at every iteration using Metropolis Hastings sampling of the grasp poses based on inferred poses. Practically, this means that the grasps generated at the first time step are perturbed in the next frame and quality of initial grasp and perturbed grasps are evaluated on the point cloud. The grasp with higher probability is selected and the procedure is repeated at every step. They assume that

larger motion will result in grasps that are colliding or in free space. When this occurs, they re-sample the grasp on the new partial point cloud. Reimanninan motion policies [154] are used for motion generation along with RRT-Connect [155] for configuration space planning. The selection of grasp to be executed is based on a weighted score that encourages selection of the current best grasp closer to the previously selected grasp and closer to the home configuration of the robot. The system is less robust to more continuous changes in the object pose and needs to go back to home position when randomly sampled grasps around initial grasps are no more valid (non-colliding and object between fingers) i.e. under larger perturbations. Sample quality is also limited by shortcomings of the 6-DoF GraspNet model discussed in Section 2.3.2.

To relax the assumption of small motions of the object between frames, Fang et al. [98] establish feature-metric similarity between grasp poses predicted continuously in a frame sequence. The grasp features after the last layer in their AnyGrasp model are stored in a buffer at each frame. At the subsequent frame, a cosine similarity matrix is established between the features of the current and the previous frame. The similarity is supervised using a contrastive loss [156] to encourage features of similar grasps to be closer. The camera is attached on the wrist and the robot tracks the grasp pose using pose-based servoing. More recent work by Liu et al. [157] extend this temporal feature-metric association of grasp poses beyond two consecutive frames using memory-augmented refinement stage. In both cases, impressive results are shown on relatively fast moving objects. However, the scenarios are restricted to planar motions. In 6-DoF motions, establishing correspondence and reducing target pose jitter while accomodating mode shifts and view changes remain pertinent.

2.4.2 Unknown Object Pose Tracking

Instead of trying to track grasps or grasp features, these methods track objects or object features. Marturi et al. [158] track object by registering object point clouds and provide a good framework for grasp re-planning in the loop. While the object is not strictly known apriori, they scan the object from four views and pre-process a reference object point cloud. Grasps

are generated using grasp experience transfer method from Kopicki et al. [159], although they need to be pre-generated on the scanned point cloud before the object undertakes any motion. Using an object model reconstructed from multiple views can improve grasp generation, especially for grasps that are on partially visible regions, as more unique views are included. Such an approach also allows grasp generation to operate at a lower frequency. Lower frequency in turn allows for less jitter in grasp pose and quality predictions, as long as pose can be estimated reliably.

The problem of simultaneous reconstruction and tracking, closely relates to the broader research on SLAM. However, SLAM approaches most commonly assume a static world and aim to map and localize a moving camera within the scene [160, 161]. Sparse map recovery can achieve relatively fast localization of a moving camera. However, the sparse map is less useful for tasks beyond localization. To address this, dense SLAM approaches [162, 163] produce dense maps with point or voxel-based representation. However, mapping and localization are more difficult, if the objects in the environments move. Dynamic SLAM and object-centric SLAM methods relax the static world assumption by segmenting the object and registering a frame-to-model colored point cloud [164] using iterative closest points [165] estimation. These methods often use discrete point cloud representations that can be harder to optimize and may not capture high-fidelity details. Recent advancements in 3D reconstruction have enabled the recovery of high-fidelity models from images using differentiable rendering. In particular, neural fields [166] can be used to create a dense map of the scene [167, 168]. Neural fields have been used for object-centric reconstruction and tracking with RGB-D vision [169]. However, neural representations are bottlenecked by expensive sampling along pixel rays as they encode a volume implicitly. On the other hand, 3DGS [170] provides a new representation that enables high-speed forward rendering and the benefits of dynamic manipulation of point-like 3D Gaussian primitives. This representation has enabled state-of-the-art performance in dense SLAM [171, 172, 173], for large but static scenes.

A key design advantage in leveraging incremental object-centric reconstruction along with tracking is that grasp generation accumulates more information about the object over time, revealed naturally due to the relative object motion. In addition, frame-to-model reg-

istration enables global consistency that is robust for longer term grasp tracking. Recent advancements in using radiance field representation for SLAM [174] motivate the investigation of such representations for object-level 6-DoF tracking and grasp generation.

2.5 Vision-based Manipulation and Grasping in Space

Since the dawn of the space exploration, the use of visual sensors has been commonplace in both human and robotic spaceflight. While initially used for photography and scientific data [175, 176], 'engineering' cameras [1] quickly grew in use for navigation in interplanetary missions [177, 178], rendezvous, proximity operations and docking [179, 180] and surface operations [181, 182]. The use of cameras for robotic manipulation in space started with Shuttle Remote Manipulator System (SRMS) [183] aboard the space shuttle and the International Space Station (ISS). SRMS used wrist and elbow mounted cameras were used to assist astronauts in teleoperating the arm to perform inspection, berthing and deployment of shuttles, station modules and other spacecraft. Manual teleoperation with camera feedback remains pervasive aboard the ISS even through successive generations of onboard robotic arms [184]. Since ETS-VII [185], the use of intelligent robotic manipulation in orbit has shifted towards On-orbit Servicing, Assembly, and Manufacturing (OSAM) applications. ESA's robot vision tests onboard ETS-VII already tested the use of feature detection, pose estimation and 3D reconstruction for vision-based manipulation in servicing applications [186]. Although, with the exception of Orbital Express [187], no other missions have since flown with robotic manipulators.

Capture and servicing of uncooperative targets remains a key long term challenge for OSAM [188]. As robotic manipulators are suitable for controlled berthing of such uncooperative targets, tracking the targets precisely before interaction or grasping is a key pre-requisite along with tasks like visual servoing [189]. Another important problem area in OSAM is handling of diverse objects that are not pre-defined and hence need not be manufactured in strict conformance to a tool interface. Here, on-board grasp synthesis from visual observa-

[1] In space missions, intrumentation not collecting scientific data is commonly classified as engineering hardware.

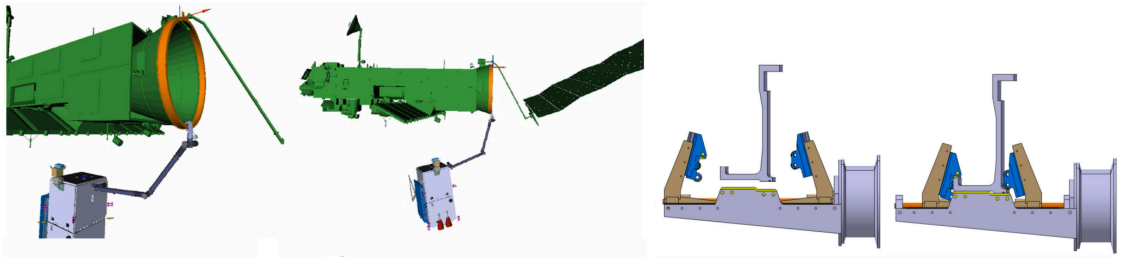


Figure 2.11: Visualization of Launch Adapter Ring capture (adapted from [193])

tion becomes a crucial problem. The latter is also critical for expanded surface operations supporting build up of larger presence on the Moon [190] and Mars [191]. Such presence also entails robotic systems working in tandem with astronauts in dynamic environments, where reasoning about moving objects becomes necessary. In both cases, whether scaling up OSAM or planetary stations, proliferation of autonomous robotic manipulation will be a necessary technology element. Versatile, modular and low-cost robotic manipulation platforms are already in advanced development [192]. Imbuing them with ability to interact with unknown objects in aforementioned environments will be critical for the future of commercial and scientific space activities.

2.5.1 Grasping in Space

Robotic grasping in space is rare occurence for the moment. Robotic arms on-board ETS-VII and Orbital Express each used a grapple mechanisms to attach to the target satellite at a pre-defined location and berth them with the host spacecraft. For uncooperative spacecraft, where no recieving mechanism is pre-installed, grasping is often considered on the Launch Adapter Ring (LAR)and in some cases on main engine cone [194] . The gripper for LAR capture are similar to a parallel-jaw grippers but designed to comply with the shape of the LAR cross-section as shown in Fig. 2.11. Since LAR is of known shape and has circular symmetry in one plane, the grasp synthesis problem reduces to localization of the ring and selection of a reachable point along the circumference with the grasp approach vector perpendicular to it. A more relevant application for grasp synthesis is the post-capture manipulation and servicing actions. However, to our best knowledge, no existing works explore this problem.

Other applications of grasping include astronaut assistance on board ISS. Robonaut [195, 196] is a humanoid robot with two dextrous five-fingered hands [197] that is designed to assist astronauts with handling cargo bay unloading and other routine tasks on the station. Martin et al. [198] demonstrate a single grasp primitive for Robonaut 1 that allows it to hold a drill securely. Bridgwater et al. [197] demonstrate dexterity of the Robonaut 2 hand to execute various dextrous grasp types, but grasping is limited to cargo transfer bags of known shape marked with fiducial markers. Ku et al. [199] studied unknown object grasping by associating convolutional features to contact points. The evaluation objects are simple in shape and only a single pre-shape configuration is used. In their following work [200] they propose a framework for dextrous manipulation for Robonaut with three examples of grasping and manipulation task learned from demonstration. In each case, the object is known and the task is pre-defined. However, there is a clear trend and interest in enabling unknown object grasping as a precursor for astronaut assistance. Another platform used to assist astronauts is the Astrobee platform [201]. Astrobee is an assistive free-flying robot that can monitor the station environment autonomously with the aid of fiducial markers. Macpherson et al. [144] explored the use of Astrobee for grasping objects floating in the station. The focus of this work was on the use of adhesive gripper and the model predictive control for dynamic grasping. A simple cylindrical object was considered and the state of the object was assumed to be known at all times.

Vision-based manipulation has also been a core building block of modern rovers like Curiosity and Perseverance [202]. In both cases, the end-effector contains an scientific instrument turret instead of a gripper that interacts selectively with the surface in a predefined manner. Using the sample collection instruments in the turret, Preseverance, with its instruments, collects surface samples for characterization. It also caches them in tubes that are left behind to be returned by a future mission as per the Mars Sample Return architecture [203]. These sample tubes need to be grasped and transferred to the lander. Grasp pose for fetching and transfer to lander can be precomputed as the sample tubes are of known shape and size, requiring only localization as studied in [204, 205]. Sample collection of more unstructured nature like rocks and other geological samples will require

reasoning about the unknown shape. Orsula et al. [206] study the use of deep reinforcement learning for grasping rocks with a parallel-jaw gripper. This is suitable for this specific task but less generalizable beyond rocks. Across all the aforementioned applications, the ability of the robot to reason about all possible grasps on any unknown object is a key capability that is needed for autonomous robotic manipulation in space.

2.5.2 Tracking of Dynamic Objects in Space

6-DoF pose estimation and tracking for uncooperative spacecraft closely relates to space-borne manipulation are important problems for proximity operations in orbit. Early works investigated a host of sensor and algorithmic choices while highlighting the need for a cost-effective and robust system [207]. Recently, the focus has shifted towards vision-based pose estimation methods due to the lower SWaP-C of vision sensors. Importantly, the progress was driven by rapidly evolving models and methods in computer vision. Early works [208, 209] utilized conventional image features that either extract parts of the structures like edges [210] and corners [211] or invariant visual descriptors like SIFT [212]. Here, the pose estimation performance relied on feature extraction and robust correspondence matching to a known 3D model. Due to the challenging visual environment, the methods lack robustness to the challenging visual environment in orbit. Subsequently, CNNs demonstrated superior 6-DoF pose estimation results on representative scenarios [213]. However, these networks require extensive pre-training which is done with synthetic data. Using synthetic data for pre-training a neural network presents the problem of performance transfer across the domain gap between the synthetic and real image formation processes. Consequently, high-quality datasets [214], image augmentations [215] and domain adaptation strategies [216, 217] have been proposed to address this issue. Simultaneously, these learned models for pose estimation were extended to state tracking assuming natural motion [218, 219]. Overall, these approaches require exhaustive and iterative cycles of dataset curation, pre-training, and ground testing for every target. Moreover, unmodeled artifacts like structural damage or material degradation remain challenging for these methods relying on the exact 3D model.

The requirement of the target model knowledge was relaxed in [220] which proposed a fast, one-shot primitive reconstruction of the target object. The shape is approximated by a fixed set of superquadrics per image. Despite being fast, the method only recovers the shape of the target and not the scale. Consequently, only a rotation is estimated per image. It also depends on a prior learned from a small dataset of objects. These attributes imply that it suffers from overfitting, viewpoint ambiguities of primitives, and a lack of fidelity. As opposed to these methods, our approach does not require pre-training and considers no apriori information about the object or its relative motion. The object is reconstructed online using a 3D Gaussian representation that simultaneously enables efficient pose tracking in the loop. Our approach applies more generally to any unknown object and can be used for other on-board applications such as robotic manipulation.

2.6 Summary and Research Questions

This chapter reviews the literature and the state of the art in robotic grasping research. The paradigm change from analytical to data-driven methods is discussed, with a focus on the evolution of vision-based grasp synthesis. The chapter also discusses the challenges and limitations of existing methods and datasets, especially in the context of 6-DoF grasp synthesis for unknown objects and going beyond static environments. Finally, perception and robotic grasping in the context of space robotics applications is reviewed. Based on the review, we identify two key problem areas which are investigated in this work.

Generative 6-DoF Grasp Synthesis for Unknown Objects

We underscored the complexity of vision-based 6-DoF grasp synthesis problem and the merits of generative modeling to learn complex data distributions. In addition, we highlighted the limitations of current grasp generation models, which need to be retrained from scratch for every task-specific grasping task. We are interested in the following research questions:

RQ-1: How can a robot reason about the distribution of successful grasps on objects from imperfect visual inputs?

- How can a conditional distribution of 6-DoF grasp poses on visual inputs be learned effectively with a generative model?
- How can task-specific grasping be enabled with minimal additional effort?
- How can such a grasp synthesis model be applied to real-world scenarios?

In Chapter 3, we focus on developing a novel generative modeling framework for object-centric grasp synthesis. In Chapter 4, we develop a system that enables object-centric grasp synthesis models to be employed in the real world.

Reconstruction and 6-DoF Tracking of Dynamic Unknown Objects

From industrial manipulation to human-robot interaction, robots need to work around non-static objects. While there has been rapid progress in closing the loop for robotic grasping, these techniques are restricted planar motions. Another limitations for existing solutions is inflexibility to large changes in viewpoints and reliance on single views. These methods are also unable to switch to better grasps as a selected grasp becomes infeasible or occluded. On the other hand, SLAM methods are limited to static world assumptions and use inflexible representations. We higlight the potential of radiance field representations for adaptive resolution 3D reconstruction and maintaining a continuous and differentiable representation of the scene. Based on these insights, we are interested in the following research questions

RQ-2: How can a robot reason about an unknown object moving in 6D space without prior knowledge of its structure or motion?

- What is the right representation for unifying dense 3D reconstruction, 6-DoF tracking and grasping in unstructured scenarios?
- How can radiance fields enable object-centric SLAM?
- How can such a method be applied for on-orbit applications?

In Chapter 5, we develop incremental dense reconstruction and 6-DoF tracking pipeline that leverages radiance field representation for object-centric SLAM. We conduct a preliminary validation of the system for tracking and 3D reconstruction of unknown spacecraft in non-natural relative motion.

Chapter 3

Generative 6-DoF Grasp Synthesis

3.1 Introduction

The challenge of robotic grasping lies not in finding the perfect solution, but in navigating a sea of possibilities. This complexity underlies the fundamental challenge of enabling robots to interact with the real world. As humans, we seamlessly navigate the interplay of geometry, material, mechanics, and semantics to interact with countless objects every day. For robots, however, this is a fundamental challenge. We find that the task of grasping objects in the real world presents three key issues. First, there is not a singular best configuration to grasp any object. Second, the set of objects one may encounter in the real world is open and unbounded. Third, the grasp executed by a robot needs to be task-relevant. As discussed in Chapter 2, these issues can be treated in an analytical or data-driven manner. The latter treatment is more effective for unstructured settings and has been bolstered by consistent progress in machine learning.

Learning 6-DoF grasp synthesis from annotated data has shown increased performance and generalization, with the availability of large and high-quality open-source datasets. In Chapter 2, the appeal of sampling-based methods was highlighted. Their ability to capture multi-modality of grasp distributions, especially by employing generative modeling, is desirable. With generative modeling, we are interested in learning the data-generating distribution of training data from which novel samples can be generated, as opposed to simply learning

a regression function. Generative modeling offers the potential to automatically discover latent (and hopefully causal) factors affecting the generation of successful grasps. They learn probabilistic models from data and can offer data-efficient generalization to new objects.

We tackle this problem of generative modeling of 6-DoF grasp synthesis with two further assumptions. First, we assume that the inputs and outputs are at an object-level or in other words, *object-centric*. On the other hand, generation may be done at a scene level, which takes all the information in the field of view. However, scene-centric grasp generation models are trained on specific types of scenes (e.g. table-top) and can not generalize to other scenes (e.g. shelf). Object-centric grasp generation is more generalizable and flexible as it does not make assumptions about the scene, but requires accurate segmentation of objects in the scene. While this has traditionally been a challenge, the progress in zero-shot instance segmentation in the wild has reached significant maturity using pre-trained foundation models [221, 222]. Motivated by these aspects, we focus on object-centric grasp generation. Secondly, we assume a parallel-jaw gripper, which is versatile and most commonly used in robotic manipulation.

With these assumptions, this chapter covers the design and implementation of a novel framework for generative modeling of 6-DoF grasp synthesis problem. Section 3.2 introduces the generative modeling problem and its application to 6-DoF grasp synthesis. Highlighting the challenges and performance gap in existing models, the *Grasp Latent Diffusion Models (GraspLDM)* framework is proposed in Section 3.3. The network architecture and other implementation details are covered in Section 3.4. Finally, experiments and results are presented in Section 3.5 along with design ablations.

3.2 Generative Modeling for Grasp Synthesis

3.2.1 Variational Autoencoders

Consider a data sample x taken from a dataset $\mathcal{D}$. In generative modeling, we aim to learn a model $p_\theta(x)$ parameterized in θ that approximates the true data distribution $p(x)$. We are then interested in sampling novel samples from this distribution. For example, consider the

problem where x is an image taken from a dataset of images of dogs and we want a model to generate new images of dogs. Since we don't have access to the true data distribution, we assume that the data is generated as a function of unobserved low-dimensional variables z , also called latent variables or simply *latents*. The distribution of the observed data and the corresponding latents is the joint distribution $p(x, z)$. To be able to learn this distribution, we want to maximize the likelihood $p_\theta(x)$ over all available data samples. The likelihood is obtained by marginalizing the latent variables as in Eq. 3.1. We can also use the chain rule of probability: $p(x, z) = p(x|z)p(z) = p(z|x)p(x)$, where $p(x|z)$, $p(z|x)$ are conditional distributions and $p(z)$, $p(x)$ are marginal distributions to rewrite it as in Eq. 3.2.

$$p_\theta(x) = \int p_\theta(x, z)dz \tag{3.1}$$

$$= \int p_\theta(x|z)p_\theta(z)dz \tag{3.2}$$

However, either integral is difficult to compute as we do not know the latent variables or how they correspond to an observed data point. To tackle this problem, we can learn an *inference model* $q_\phi(z|x)$ using amortized variational inference. Here, inference refers to the process of estimating the latent for an observed data point. Amortized refers to the method of learning a single inference model that optimizes global model with parameters θ learned across all the data samples. This is in contrast to standard variational inference for smaller datasets, where model parameters are optimized per data sample. Finally, variational refers to the fact that we are approximating a distribution and therefore the output of the model are variational parameters like mean and standard deviation. This inference model approximates the true posterior $p(z|x)$ and is referred to as the *approximate posterior*. The *generative model* $p_\theta(x|z)$ then maps any latent to the data space. With these two models, we can convert a data point to a latent and reconstruct it back.

The parameters θ and ϕ are learned by maximizing the log-likelihood over data samples

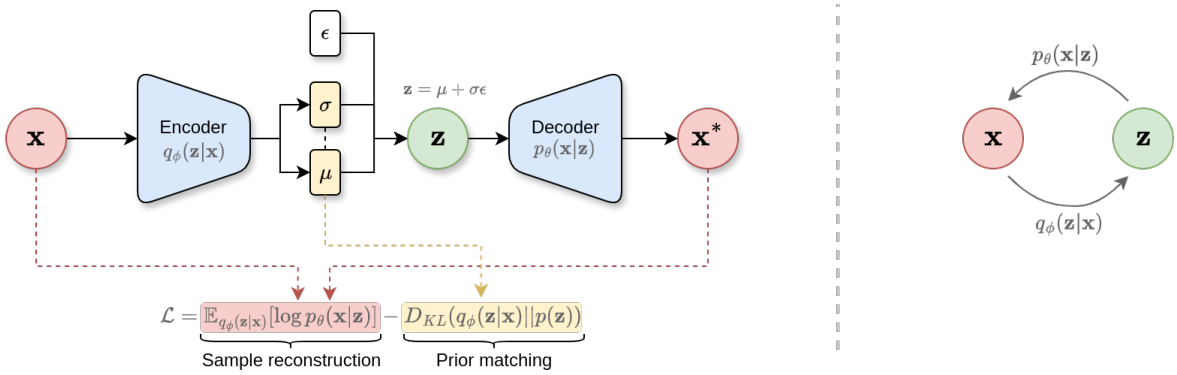
which is equivalent to maximizing the Evidence Lower Bound (ELBO) [116].

$$\begin{aligned}\log p(x) \geq \underbrace{\mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p(x,z)}{q_\phi(z|x)}\right]}_{\text{ELBO}} &= \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\right] \\ &= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] + \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p(z)}{q_\phi(z|x)}\right] \\ &= \underbrace{\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right]}_{\text{sample reconstruction}} - \underbrace{D_{KL}\left(q_\phi(z|x)||p(z)\right)}_{\text{prior matching}} \end{aligned} \tag{3.3}$$

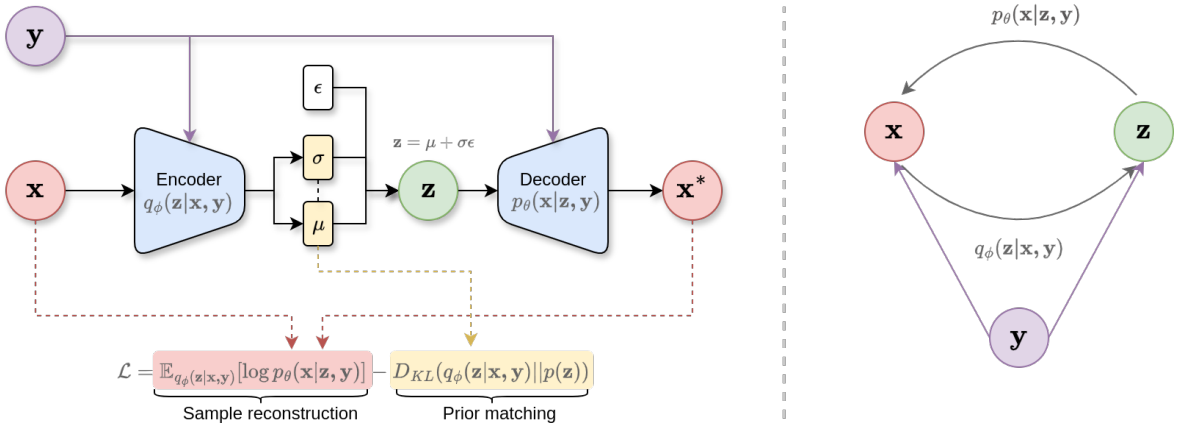
Here, $\mathbb{E}$ denotes the expectation over a distribution and D_{KL} is the Kullback-Leibler divergence between two distributions, defined as $D_{KL}(q(x)||p(x)) = \mathbb{E}_{q(x)}\left[\frac{q(x)}{p(x)}\right]$. The first term forces the model to reconstruct the original data from the latents and the second term regularizes the approximate posterior to be closer to the true prior distribution $p(z)$. Now, consider how we sample new data points from the model $p_\theta(x)$ based on Eq. 3.2: we sample a latent z from the prior $p(z)$ and then use the generative model $p_\theta(x|z)$ to generate a new data point. Here, it is convenient to assume that the prior is a simple distribution. In addition to being simple to sample from, this assumption also forces the model to learn a simple distribution in the latent space that can generalize better.

We assume that the prior $p(z) = \mathcal{N}(0, I)$ is a standard Gaussian and the true posterior ($p(z|x)$) is of a simple form like a multi-variate Gaussian with diagonal covariances. If θ and ϕ are parameters of deep neural networks with sufficient capacity, arbitrary mapping between distributions can be learned from data. This is the basic idea of a Variational Autoencoder (VAE)[116], for which a more detailed discussion is available in [223]. Fig. 3.1a shows the architecture and graphical representation of the VAE. The neural network modeling the approximate posterior is also called the encoder and the network modeling the generative distribution is also called the decoder. This derives from autoencoder models in machine learning and we will use this terminology interchangeably with the probabilistic terms. The intuitive difference between a standard autoencoder and a VAE is that the latents are not learned as points, but as soft ellipsoidal regions in the latent space.

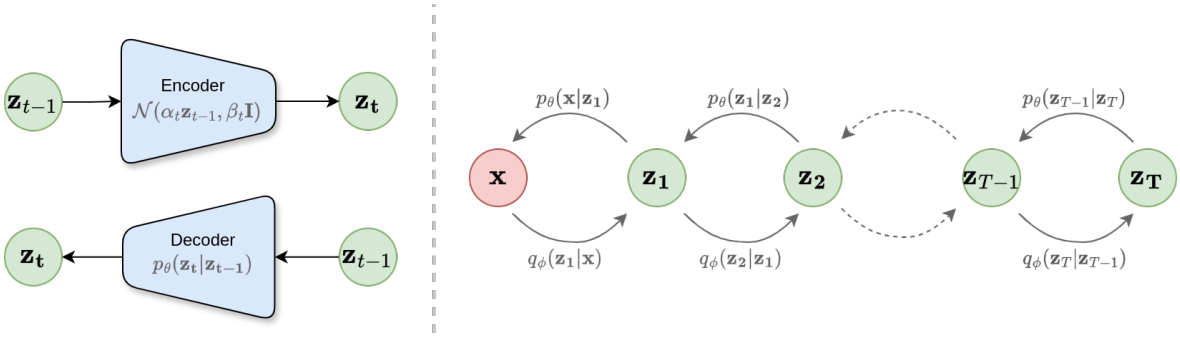
We can take this notion further to *conditional* generative models, where we are inter-



(a) Variational Autoencoder



(b) Conditional Variational Autoencoder



(c) Denoising Diffusion Probabilistic Model

Figure 3.1: Architecture and computation graphs of generative models

ested in learning conditional distributions of type $p_\theta(x|y)$. Here, y contains some meaningful semantic information on which the data point x is conditioned. In the simplest case, y can be a classification label. An intuitive example similar to earlier is a dataset that contains images of cats and dogs with corresponding binary labels 0 and 1 respectively. Then, we might be interested in generating a picture conditioned on the semantic label that lets user control the generation of whether it should be a dog or a cat. In this case, we learn a model $p_\theta(x, z|y)$ that approximates the joint distribution of data and the latent variables conditioned on y. The inference model linking the observed data to latents becomes $q_\phi(z|x, y)$, while the generative model becomes $p_\theta(x|z, y)$. This is illustrated in Fig. 3.1b. The objective, in this case, is to maximize the conditional ELBO in Eq. 3.4, which can be derived from Eq. 3.3 by considering the conditional likelihood $p(x|y) = \int p(x, z|y)dz$ and the factorization from chain rule: $p(x, z|y) = p(x|z, y)p(z|y)$. Together, this formulates the Conditional Variational Autoencoder (CVAE).

$$
\begin{aligned}
ELBO = \mathbb{E}_{q_\phi(z|x,y)}\left[\log \frac{p(x,z|y)}{q_\phi(z|x,y)}\right] &= \mathbb{E}_{q_\phi(z|x,y)}\left[\log \frac{p_\theta(x|z,y)p(z|y)}{q_\phi(z|x,y)}\right] \\
&= \mathbb{E}_{q_\phi(z|x,y)}\left[\log p_\theta(x|z,y)\right] + \mathbb{E}_{q_\phi(z|x,y)}\left[\log \frac{p(z|y)}{q_\phi(z|x,y)}\right] \\
&= \underbrace{\mathbb{E}_{q_\phi(z|x,y)}\left[\log p_\theta(x|z,y)\right]}_{\text{conditional reconstruction}} - \underbrace{D_{KL}\left(q_\phi(z|x,y)||p(z|y)\right)}_{\text{conditional prior matching}}
\end{aligned}
\tag{3.4}
$$

3.2.2 Grasp Generation

In the context of robotic grasping, we can view the problem of generating *successful grasps* as a conditional generative modeling problem. We use the term successful grasp instead of a form-closure, force-closure or stable grasp to refer more generally to grasps that have succeeded during data collection and are expected to succeed in the real world. Ideally, we want to learn stable grasps, i.e. considering the grasp and the object as a dynamical system. However, as discussed in Chapter 2, modern grasp datasets may contain grasp

annotations that are based on force-closure tests, evaluation in simulation or trial and error in the real world. In the latter two cases, the objective is to collect stable grasps. But, a manually defined post-grasp perturbation check like shaking is used to qualify stability. In these cases, it is not appropriate to use formal terminology like stability, especially if no formal guarantees can be implied.

Relating back to the CVAE formulation, the data point x becomes a grasp configuration while the conditioning variable y is a visual observation like an image or point cloud. The continuous latent variable z can be used to model the unobserved variables that generate successful grasps. We typically assume a simple prior over latent variables $p(z) = \mathcal{N}(0, 1)$. Then, a random latent z can be sampled from this simple distribution and passed through the generative model $p_\theta(x|z, y)$ to generate a successful grasp, once the model is learned. The generation can be done at a scene level, which takes all the information in the field of view, or at an object level, which takes only the local information of the concerned object. We use the latter and therefore y provides object-level observation. In general, x, y, z variables are vectors or tensors and we will use appropriate boldface symbols like $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ to denote them from here on.

Generative modeling for object-centric 6-DoF grasp synthesis was used by [89] for a parallel-jaw gripper and built upon by [117, 126, 224]. Recently, regression-based approaches [90] have shown significant performance improvement over the comparable VAE model. Other generative models [225, 226] have been proposed for multi-finger grippers. Here, we will focus on improving generative models for the parallel-jaw gripper, where the learning of conditional grasp distributions has not been explored in detail. We want to leverage well-known advantages of generative models including improved generalization and flexible conditional generation in hierarchical settings to improve over existing generative models like 6-DoF-GraspNet [89] which has been widely used, built upon, and cited in the research community. Especially, as novel generative models [120, 227] have consistently improved capabilities in other real-world generation tasks across complex domains like natural language and computer vision. In particular, a new class of generative models called Denoising Diffusion Model (DDM) have emerged with state-of-the-art performance on density

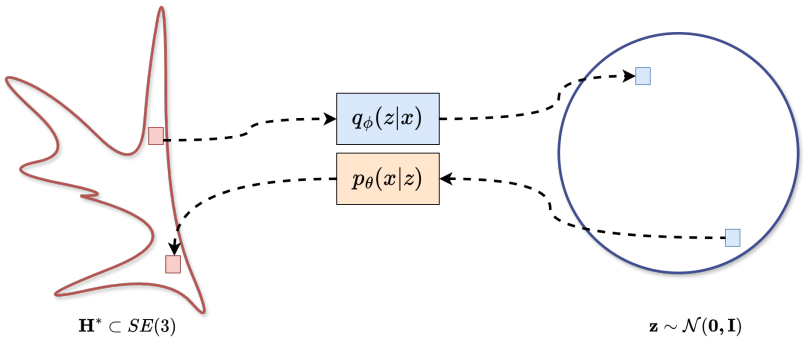


Figure 3.2: Illustration of encoding and decoding processes in a VAE. A non-trivial distribution of $SE(3)$ poses is encoded to a standard Gaussian distribution in the latent space using the approximate inference posterior $q_\phi(z|x)$. The generative model ($p_\theta(\mathbf{x}|\mathbf{z})$) can decode a random latent to a sample in the original distribution.

estimation and sample generation quality [228, 120, 122].

Diffusion models emerge naturally as we generalize the VAE graph in Fig. 3.1a (right) to multiple hierarchies of latents. So a latent at a higher level is itself generated from a lower-level latent, resulting in a hierarchical VAE. If we now constrain the latent at each stage to have the same dimensions as the input and make the inference model deterministic, we get a diffusion model as shown in Fig 3.1c. Here, the inference model or the encoder is deterministic and typically a linear Gaussian model whose parameters are scheduled such that the latent at the last step is distributed per standard Gaussian. An alternate and simpler way to interpret diffusion models is to consider the encoding process as that of gradually adding noise and learning the reverse denoising process. DDMs have multiple equivalent formulations as found in [121]. At its core, they provide a powerful framework for learning complex conditional data distributions with a simple objective. Given the successful application of DDMs to complex generation tasks for images [120] and point clouds [229], they present a good solution for improving grasp generation for robotic grasping.

3.2.3 Challenges

Let's focus on the problem of generating object-centric 6-DoF grasps of a parallel-jaw gripper using point cloud observations. We parameterize the grasp by the $SE(3)$ pose of the gripper's wrist, which can be at any location rigidly attached to the gripper. The objective is to learn representations that effectively capture the non-trivial distribution of successful object-centric 6-DoF grasps on a point cloud. Learning this representation is difficult, as the model must reason about an arbitrary set of grasp poses for each object in the $SE(3)$ space. The grasp pose distribution on any object is usually multi-modal with discontinuities. Deep generative models like VAE are a suitable candidate for such problems. They model the grasp generation problem as shown in Fig. 3.2. However, existing models like 6-DoF-GraspNet [89], which use a CVAE, exhibit poor sampling quality and require additional iterative post-processing to improve grasp samples. The model takes up to to week to train on a single GPU [90]. In addition, to employ the model for specific tasks, it needs to be re-trained from scratch with the task-specific data.

We identify three key factors that cause the aforementioned shortcomings in models like 6-DoF-GraspNet. These are the *prior gap*, *posterior gap*, and *network architecture*. The first two factors are related to the learning dynamics of the VAE, while the third factor is related to implementation choices.

1. **Prior gap** refers to gap between the assumed prior distribution and the true distribution of latents [230]. Intuitively, this means that given the latent space dimensions (which are specified in implementation), the latent representation extracted by the encoder may not truly be distributed as a simple distribution like standard Gaussian that is commonly assumed.

2. **Posterior gap** refers to the gap between the approximate posterior learned by the encoder and the true posterior. This relates to both the capacity of the encoder to model the conditional distribution and the learning dynamics emerging from Eq. 3.3& 3.4. In the latter case, the KL divergence term in ELBO objective forces the approximate posterior to match the prior. When this KL term is prioritized in optimization, the pos-

terior may quickly converge to the uninformed prior with vanishing KL term [231]. This means that the variational parameters from the encoder do not carry any information about individual samples during training. No gradient passes between the encoder and decoder through some or all latents. As a result, the decoder learns to ignore some or all of the latent dimensions to reconstruct the sample. Especially in the case of CVAEs, the decoder can become auto-regressive with respect to the conditioning variables and still maximize the likelihood or ELBO by completely ignore the latent during generation. Even for a model without complete posterior collapse, there can be an evident gap between the learned posterior and the true posterior [232]. Then, the marginal distribution $q_\phi(z) = \int q_\phi(z|x)p(x)$ to which the generative model is fit, tends to be a poor approximation of the assumed prior $p(z)$ like standard Gaussian. When generating new samples from the assumed prior $p(z)$, the samples are of poor quality. The essence of this challenge is that the two terms in the ELBO objective are in conflict with each other over how best to increase the likelihood.

3. **Network architecture** refers to the design choices made in implementing the model. In the case of 6-DoF-GraspNet, the striking elements of the design are the inputs at the encoder and the decoder. At the encoder, the object point cloud is appended with the grasp pose parameters as features, which are fed to first PointNet++ layers. The input at the decoder is similarly the latent vector appended to the point cloud as features. These design choices have two major implications. First, the raw point clouded is encoded into features twice: once in the encoder and once in the decoder. Second, appending pose parameters to the point cloud at the encoder means that for a 1000 training grasps per object, the model needs to encode the same point cloud a 1000 times. Proportionally, the gradient computation and the backpropagation processes become slower. Finally, the model design is not flexible to be used for downstream tasks without re-training.

We seek to address these issues and propose an improved framework for generative modeling of 6-DoF grasp synthesis. Contemporary literature provides several directions to

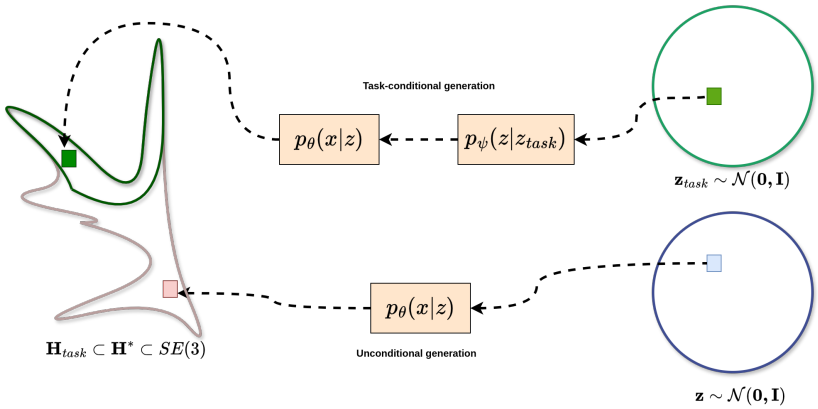


Figure 3.3: Illustration of unconditional and task-conditional generation from a prior. Assuming that task-specific grasps are a subset of all successful grasps, another generative model can hierarchically map from the prior to task-relevant distribution of latents which are then mapped to the original distribution by the unconditional model.

tackle these issues.

One possible proposition would be for VAEs to be replaced by diffusion models. As discussed earlier, these are highly effective for modeling complex conditional distributions and simultaneously are easier to train. We could simply train a diffusion model to directly generate grasps on a point cloud. However, the denoising process in a DDM is Markovian and therefore requires each denoising step to be executed in a sequence (normally 1000 steps), which increases generation latency. The effect of latency is more pronounced when the input and therefore the latent dimensionality is higher. Another alternative is to use a diffusion model as an expressive prior in the low-dimensional latent space of a VAE. This concept, called latent diffusion, was introduced in [127], applied to image generation. This architecture is primarily meant to reduce the dimensionality and speed up the sequential generation steps compared to a vanilla diffusion model. But, expressive prior in the latent space means that it can also tackle the VAE's prior gap problem.

The latent diffusion architecture brings a valuable additional benefit. Assuming that the task-relevant grasps are a subset of all successful grasps, we can use the diffusion model in the latent space as a task-specific prior. Intuitively, this would mean that the diffusion model moves the sample from a standard Gaussian distribution to the distribution of latents that

correspond to task-relevant grasps. The VAE decoder should then generate a sample that is task-relevant. This is illustrated in Fig. 3.3. As the diffusion model works in a regularized low-dimensional latent space, only latents need to be supervised using denoising loss, making it faster to train. This means that the encoder and decoder are only trained once in a task-unconditional manner. However, the diffusion can be replaced in a plug-and-play manner with another diffusion model (say for another task) without re-training the rest of the networks. This can provide flexibility that is not available in existing grasp synthesis models.

As for training dynamics of the VAE, there is careful balance to be struck between KL term and The posterior collapse problem can be tackled by annealing the weight on the KL divergence term [119]. To moderate the tug-of-war between the two terms of ELBO, the KL term can be weighted down in the ELBO objective to prioritize sample reconstruction and even promote structured latent space [233]. To address training efficiency, we can use a modular point cloud encoder that is shared between encoder and decoder. Using these insights, we propose a novel framework for generative modeling of 6-DoF grasp synthesis.

3.3 GraspLDM: Grasp Latent Diffusion Models

Grasp Latent Diffusion Models (GraspLDM) is a framework to build generative models that learn the distribution of successful grasp poses of a parallel-jaw gripper on object point clouds using latent diffusion. GraspLDM is designed to generate diverse and high-quality grasps that can be used for real-world parallel-jaw grasping. In GraspLDM architecture, a diffusion model is trained efficiently inside the low-dimensional latent space of a VAE as shown in Fig. 3.4. This diffusion model acts as a complex prior that bridges the prior gap. Using diffusion in the latent space improves the quality of sampled grasps in unconditional grasp generation. GraspLDM architecture also enables two other aspects of model flexibility that are not available in existing generative models for grasp synthesis using VAE [89] and diffusion [123] alone. First, task-specific conditional guidance can be provided solely in the latent space without re-training the VAE encoder and decoder. Second, multiple diffusion models can be trained efficiently and plugged inside this low-dimensional latent space. In

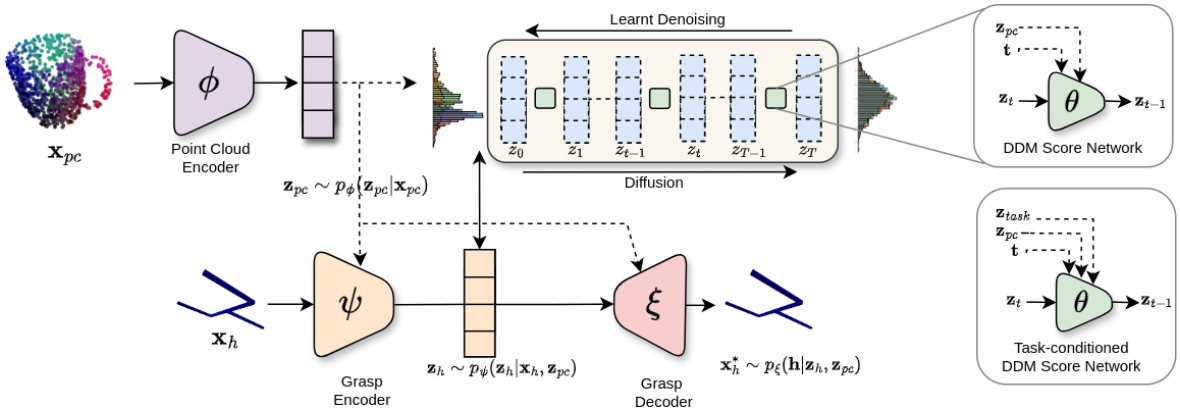


Figure 3.4: Grasp Latent Diffusion Model (GraspLDM) is composed of a point cloud encoder, a grasp encoder (ψ), a grasp decoder (ξ), and a latent diffusion module using a score network (θ). The point cloud encoder encodes a point cloud into a shape latent ($\mathbf{z}_{pc}$). At test time, the grasp encoder is not required and we sample the grasp latent $\mathbf{z}_h$ directly from the prior distribution. This latent goes through reverse diffusion before decoding. For task conditional generation, we modify the diffusion score network to accept task context $\mathbf{z}_{task}$.

the next section, we cover this design in detail. This means that task-specific adaptation occurs efficiently with representations learned in a task-unconditional setting.

3.3.1 Conditional Variational Autoencoder

Our conditional VAE consists of a point cloud encoder, a grasp pose encoder, and a grasp pose decoder with parameters ϕ, ψ, and ξ respectively, as shown in Fig. 3.4. The point cloud encoder operates on unordered 3D point sets of size $n \in \mathbb{N}^+$ to provide a fixed size latent ($\mathbf{z}_{pc} \in \mathbb{R}^m$) of size $m \in \mathbb{N}^+$ called the shape latent. The shape latent is used to condition the grasp pose encoder and the decoder. The grasp pose encoder approximates the conditional posterior $p_\psi(\mathbf{z}_h|H, \mathbf{z}_{pc})$ and the grasp pose decoder models the likelihood $p_\xi(H|\mathbf{z}_h, \mathbf{z_{pc}})$. Here, $\mathbf{z}_h$ is the point cloud conditioned grasp latent at the VAE bottleneck. Finally, the model is trained by jointly optimizing the parameters (ψ, θ, ξ) of the encoders and decoders to maximize a modified ELBO similar to Eq. 3.4.

$$\mathcal{L}_{ELBO}(\phi, \psi, \xi) = \mathbb{E}_{p(H,\mathbf{x}_{pc}),q_\phi(\mathbf{z}_{pc}|\mathbf{x}_{pc}),q_\psi(\mathbf{z}_h|H,\mathbf{z}_{pc})}\big[\log p_\xi(H^*|\mathbf{z}_h, \mathbf{z}_{pc}) - \lambda D_{KL}\left(q_\psi(\mathbf{z}_h|H, \mathbf{z}_{pc}) \,||\, \mathcal{N}(\mathbf{0}, \mathbf{I})\right)\big] \tag{3.5}$$

In Eq. 3.5, the first term is the likelihood of the decoder reconstructing the grasp pose. The second term measures the divergence between the approximate posterior distribution $q_\psi(\mathbf{z}_h|H, \mathbf{x}_{pc})$ of the grasp encoder and the conditional prior, $p(\mathbf{z_h}|\mathbf{z_{pc}}) = \mathcal{N}(0, 1)$. During training, the model should become better at learning a regularized latent space while also getting better at reconstructing the samples. In practice, the two terms contradict each other during training. The first term prioritizes keeping as much unique information in variational parameters per sample as possible to reconstruct accurately. On the other hand, the second term prioritizes the regularization of the posterior towards the prior forcing the variational parameters to be closer to zero-mean and unit variance. We can balance the two effects using λ which is the weighting hyperparameter that controls the strength of KL regularization in the latent space. A constant $\lambda = 1$ provides the strict ELBO objective. However, optimizing strict ELBO frequently results in the KL divergence term (D_{KL}) becoming vanishingly small early in the training. This is called posterior collapse and happens at the cost of reconstruction accuracy. The model gets stuck in the local minima from which point it is difficult to recover reconstruction accuracy back. Especially in a conditional VAE like ours, posterior collapse results in an auto-regressive decoder model. In this case, the decoder ignores the latent completely and becomes a predictor that uses only conditioning information to predict the output. To avoid this, we use linear annealing on the λ parameter based on [119].

3.3.2 Latent Diffusion

In principle, sampling from VAE's prior and decoding it should be sufficient to generate successful grasps. However, decoding a latent drawn from the prior often results in lower sample quality because of the prior gap problem highlighted in Section 3.2.2. To alleviate this issue, we propose to use a Denoising Diffusion Model (DDM) in the latent space. We justify the choice of latent diffusion structure from two observations. First, a simple DDPM would need to encode a point cloud at each step and use a high-dimensional intermediate representation, increasing computation for each denoising step. Alternatively, separating the point cloud encoder from the diffusion requires separate expensive training of a point cloud auto-encoder. Further, such a model needs to be re-trained for newer task conditionings.

A DDM consists of two processes: a forward diffusion process and a reverse de-noising process. In this work, we use the standard discrete-time DDPM [120], where the forward process is a linear Markov chain of deterministic Gaussian kernels in Eq. 3.6a. The noise added at each time-step $t \in [1, T]$ is pre-defined by a variance schedule β_t. We use linearly increasing β_t such that the distribution $q(\mathbf{z}_T)$ converges to the standard normal distribution at the final forward time step T. In the forward diffusion process, a noisy sample at any time step can then be obtained by Eq. 3.6b.

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) \;=\; \mathcal{N}(\sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t\mathbf{I}) \tag{3.6a}$$

$$\mathbf{z}_t \;=\; \sqrt{\bar{\alpha}_t}\,\mathbf{z}_0 \;+\; (1-\bar{\alpha}_t)\boldsymbol{\epsilon}\;; \tag{3.6b}$$

$$\bar{\alpha}_t \;=\; \prod_{t=1}^{T}(1-\beta_t) \quad ; \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Intuitively, the forward diffusion process adds noise to input data until the data distribution transitions to a simple prior distribution. Consequently, recovering the data distribution from the prior requires a time reversal of the forward process from $t = T$ to $t = 0$. This reverse diffusion process uses a learned score network (θ) which de-noises a sample, in our case the latent $\mathbf{z}_h$, from t to $t-1$. For notational simplicity, we use $\mathbf{z}_t$ to refer to conditional grasp latent $\mathbf{z}_h$ at time t. Using the reverse diffusion kernel in Eq. 3.7, the distribution of the de-noised sample at time-step $t-1$ can be modeled by the mean $\boldsymbol{\mu}_\theta(\mathbf{z}_t, t)$ and a known variance σ_t^2. More simply, we can re-parameterize the mean and instead learn a network to directly predict the noise $\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t)$ to be removed at each step [120]. The de-noising score model $\boldsymbol{\epsilon}_\theta$ predicts the noise to be removed conditioned on the current time-step, while the parameter weights (θ) are shared across all the time steps. Then, the sampling can be done using Eq. 3.8, where σ_t can be simplified to use variance similar to the forward process i.e. β_t.

$$p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t) \;=\; \mathcal{N}(\boldsymbol{\mu}_\theta(\mathbf{z}_t, t), \sigma_t^2\mathbf{I}) \tag{3.7}$$

$$\mathbf{z}_{t-1} = \frac{1}{\sqrt{(1-\beta_t)}}\big(\underbrace{\mathbf{z}_t - \frac{\beta_t}{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t)}_{\boldsymbol{\mu}_\theta} + \sigma_t\eta\big)\;;\; \eta \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{3.8}$$

DDMs with this formulation can be trained to maximize an ELBO-like objective, of the following form [120]:

$$\mathcal{L}_D(\theta) = \mathbb{E}_{p(\mathbf{z}_0),\boldsymbol{\epsilon}\sim\mathcal{N}_{(\mathbf{0},\mathbf{I})}} ||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t)||^2$$

Adapting it to our latent diffusion architecture, this objective $\mathcal{L}_D(\theta)$ can be written as::

$$\mathcal{L}_D(\theta) = \mathbb{E}_{t\sim\mathcal{U}(1,T),\boldsymbol{\epsilon}_\theta\sim\mathcal{N}(\mathbf{0},\mathbf{I}),p(\mathbf{x}_{pc},H),q_\psi(\mathbf{z}_h|H,\mathbf{z}_{pc})} ||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{z}_{h,t}, \mathbf{z}_{pc}, t)||^2 \tag{3.9}$$

3.3.3 Training and Generation

The training is done in two stages. In the first stage, we fit the parameters of encoders and decoders by maximizing the VAE ELBO in Eq. 3.5. In the second stage, we fit the parameters of the score model to minimize DDM loss in Eq. 3.9. While it is possible to train all the networks in a single stage, it is easier and faster to train the diffusion model once the latent space of the VAE is frozen. In single-stage training, the constantly changing posterior distribution at the encoder means that the optimization of the weights of the denoising score network is wasteful until VAE training has converged sufficiently. Separation of the point cloud encoder also allows us to process the point cloud only once per batch of grasp samples as opposed to 6-DoF-GraspNet [89] where the encoder processes point-cloud grasp pair together for each grasp sample. This reduces the computational cost of training our models. The training and generation algorithms for GraspLDM are summarized in Algorithms 1 and 2 respectively.

Algorithm 1 GraspLDM Training

Require: Point cloud encoder ϕ, grasp encoder ψ, grasp decoder ξ, denoising score network θ, dataset $\mathcal{D}$, number of diffusion steps T, variance schedule $\{\beta_t\}_{t=1}^{T}$

Stage 1: Train VAE
Initialize VAE parameters ϕ, ψ, ξ
while not converged **do**
  Sample $(\mathbf{x}_{pc},\, H) \sim \mathcal{D}$
  $\mathbf{z}_{pc} \leftarrow q_\phi(\mathbf{z_{pc}}|\mathbf{x}_{pc})$ ▷ Encode point cloud
  $\mathbf{z}_h \leftarrow q_\psi(\mathbf{z}_h|H, \mathbf{z_{pc}})$ ▷ Encode grasp
  $H^* \leftarrow p_\xi(H|\mathbf{z}_h, \mathbf{z_{pc}})$ ▷ Decode grasp
  Compute $\mathcal{L}_{ELBO}(\phi, \psi, \xi)$
  Compute gradients $\nabla_{\phi,\psi,\xi}\mathcal{L}_{ELBO}$
  Update ϕ, ψ, ξ
end while
Stage 2: Train Diffusion Model
Fix VAE parameters ϕ, ψ, ξ
Initialize diffusion parameters θ
while not converged **do**
  $\mathbf{z_{pc}} \leftarrow q_\phi(\mathbf{z_{pc}}|\mathbf{x}_{pc})$ ▷ Get shape latent
  $\mathbf{z}_h \leftarrow q_\psi(\mathbf{z}_h|H, \mathbf{z_{pc}})$ ▷ Get grasp latent
  Sample $t \sim$ Uniform$(\{1, ..., T\})$
  $\epsilon \sim N(0, I)$
  $\bar{\alpha}_t \;=\; \prod_{t=1}^{T} (1 - \beta_t)$
  $\mathbf{z}_t \;=\; \sqrt{\bar{\alpha}_t}\, \mathbf{z}_0 \;+\; (1 - \bar{\alpha}_t)\boldsymbol{\epsilon}$ ▷ Forward diffusion
  Compute $\mathcal{L}_D(\theta)$
  Compute gradients $\nabla_\theta \mathcal{L}_D$
  Update θ
end while

Algorithm 2 GraspLDM Generation

Require: Trained models ϕ, ξ, θ, point cloud $\mathbf{x}_{pc}$, number of diffusion steps T, variance schedule $\{\beta_t\}_{t=1}^{T}$

$\mathbf{z}_{pc} \leftarrow q_\phi(\mathbf{z}_{pc}|\mathbf{x}_{pc})$ ▷ Encode point cloud
$\mathbf{z}_T \sim N(0, I)$ ▷ Sample latent from prior
for $t = T$ to 1 **do** ▷ Denoise latent
  $\mathbf{z}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}(\mathbf{z}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{z}_t, \mathbf{z}_{pc}, t)) + \sigma_t \eta$
end for
$H \leftarrow p_\xi(H|\mathbf{z}_0, \mathbf{z}_{pc})$ ▷ Decode final latent
return Grasp pose H

3.4 GraspLDM: Implementation

3.4.1 Grasp Pose Representation

We use the representation of the grasp pose ($\mathbf{x}_h$) given in Eq. 3.10, where $\mathbf{t}$ is the translation vector while $\mathbf{a}$ is the Modified Rodrigues Parameters (MRP) [234] for orientation. Eq. 3.10 also relates MRP with quaternions where q_w is the quaternion scalar component. In contrast to regressing pose matrices or quaternion vectors that have one or more dependent parameters, the three parameters of MRPs can be regressed independently.

$$\mathbf{x}_h = \Big[\mathbf{t}, \mathbf{a}\Big]^T; \quad \mathbf{a} = \frac{4}{1+q_w} \Big[q_x, q_y, q_z\Big]^T \tag{3.10}$$

3.4.2 Network Architecture

Point Cloud Encoder

The point cloud encoder is required to extract meaningful features efficiently from a set of 3D points. To accomplish this, the architecture must tackle three properties of point sets. First, as the input data is a set of unordered 3-vectors, the architecture must be permutation invariant in this set. Second, as the neighboring points reveal local structures, the architecture must capture interactions between points in the local neighborhood. Third, semantic information in the point cloud is invariant to rigid transformations, therefore the architecture must extract global features that are invariant under transformations. PointNet [96] addressed permutation-invariance and transformation-invariance using MLPs and max-pooling layers. PointNet++ [93] improved the local feature extraction by augmenting PointNet with sampling and grouping layers. In the sampling layer, a pre-defined number of centroids are sampled using farthest point sampling, defining centers of local neighborhoods. In the grouping layer, the neighbors within a pre-defined radius are grouped forming local point groups. Subsequently, a PointNet style MLP extracts features for these points in the local groups. For tasks like point cloud segmentation which require labeling each point, PointNet++ avails a

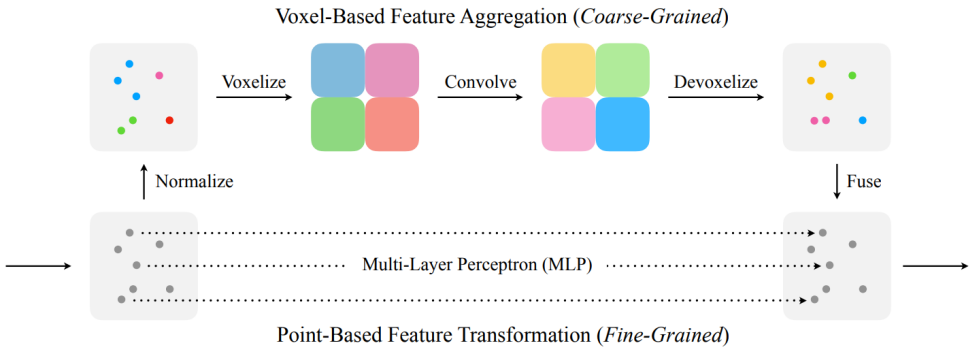
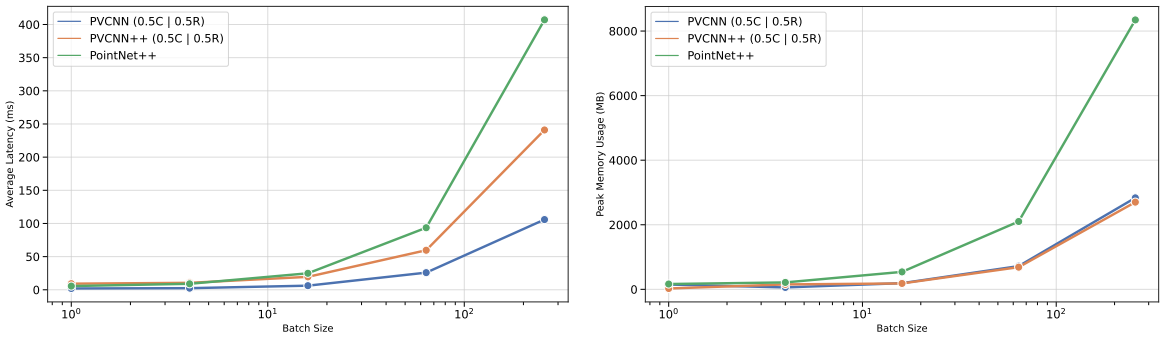


Figure 3.5: Hybrid point-voxel convolution (PVConv) operation for efficient processing of point clouds [235]. The low-resolution voxel-based branch extracts neighborhood information while the high-resolution point-based branch extracts fine-grained features.

feature propagation layer that propagates features back to individual points. In previous works on grasping [89, 98, 90, 117, 107, 126, 112], point cloud encoding networks are most commonly built with PointNet++ [93] architecture. While hierarchical feature extraction is highly effective, the sampling and grouping layers in PointNet++ incur latency costs for random memory access as the neighbors are not stored contiguously in memory and require nearest neighbor search. Another approach to encoding 3D data in grasping is using voxel-based representation [101, 75, 103] which implicitly encodes neighborhood information and can be operated on efficiently using 3D convolution lowering random memory access cost. However, memory requirements for voxel-based architectures scale cubically with resolution and are not memory-efficient.

The memory and random access trade-offs between point and voxel-based architectures can be bridged with a hybrid point-voxel convolution (PVConv) layer [235]. As shown in Fig. 3.5, PVConv takes in a point cloud and splits the processing into two branches. The point-processing branch extracts high-resolution per-point features in PointNet style without the need for grouping. The voxel-processing branch extracts features at low spatial resolution using efficient 3D convolutions without random memory access latency. The two branches are fused by deconvolution of the voxel features and assigning them to the points encapsulated by each voxel. The network composed from from PVConv layers, called Point Voxel Convolutional Neural Network (PVCNN) is 3x more memory-efficient and 5x faster



(a) (b)

Figure 3.6: Comparison of (a) latency and (b) peak memory usage for forward pass (N=1024) of comparable PVCNN [235] and PointNet [93] models. For performance comparison on point cloud learning benchmarks, see [235].

than PointNet++ with comparable performance on point cloud learning benchmarks [235]. To select a suitable point cloud encoder for GraspLDM, we compare the performance of PointNet++, PVCNN, and PVCNN++ architecture on forward pass latency and peak memory usage. The PVCNN++ architecture extends the PVCNN with feature propagation layers from PointNet++ for point-wise feature refinement. For comparison, we first use existing benchmarks to find equivalent network configurations that provides comparable performance. Then, we construct the encoder network with each as required for GraspLDM. Then, we measure the forward pass latency and peak memory usage for encoding point clouds of size 1024 points at various batch sizes. For each batch size, the measurements are averaged across 100 runs. The results are shown in Fig. 3.6a and peak memory usage in Fig. 3.6b.

The memory and latency trends affect both the generation and training speed of the grasp sampler. We observe that the forward pass of a single cloud of 1024 points with PVCNN at 2.6ms is 3x faster than PointNet++ and 3.5x faster than PVCNN++. At larger batch sizes, which is important for training speed, PVCNN is 2x faster than PVCNN++ and 4x faster than PointNet++. On the other hand, the memory consumption of the PVCNN encoder is generally comparable with PVCNN++ and 3x lower than PointNet++. Therefore, PVCNN is chosen for the point cloud encoder in GraspLDM as it is more memory-efficient

and faster than PointNet++. As we are interested in encoding the point cloud to a global shape latent, the feature propagation layers in PVCNN++ which propagate features back to the points for segmentation-type tasks are not necessary, especially due to their latency cost.

The nominal configuration of PVCNN consists of two PVConv blocks and two MLP blocks. The two PVConv blocks output 64 and 128 channel features using 16^3 and 32^3 voxel grid respectively. The two MLP blocks output 1024 and 2048 dimensional features. PVConv uses 3D convolution as linear operation, GroupNorm [236] for normalization and Swish [237] function as non-linear activation. MLPs use 1D Convolution as the linear operation, BatchNorm [238] for normalization, and Rectified Linear Unit (ReLU) as activation function. The output of the last MLP block is transformed to the shape latent of the desired size using a linear layer without activation. We will typically scale this nominal configuration up or down based on capacity requirements. In this case we will refer to the scaled configuration as PVCNN (0.5C | 0.5R), as in Fig. 3.6, which represents the factor by which the nominal dimensions of channels/features[1] as well as the resolution of the nominal voxel grid are scaled. We use PVCNN (0.5C | 0.5R) for the point cloud encoder when learning on smaller category sets and PVCNN (0.75C | 0.75R) for larger category sets. The input to the point cloud encoder is fixed at 1024 points, to balance memory usage with point sampling density. The output of the point cloud encoder is a latent ($\mathbf{z}_{pc}$) of size 128.

Grasp Encoder and Decoder

The grasp pose encoder and decoder are 1D convolutional neural networks with residual blocks [240] that avoid gradient vanishing at higher depths. The network nominally contains four blocks with 32, 64, 128, and 256 output feature channels respectively. Condition-

[1]The nomenclature of channels and feature dimensions can be ambiguous outside of image-based CNNs and are typically based on PyTorch [239] layer implementations. In 1D convolutional networks, we use (B, C, N) description, where N is the number of points and C is the number of channels per point. A 1D convolution layer expands or reduces the C dimension at output. In MLPs, the tensor dimensions are specified as (B, N, D), where D is the feature vector size. A Linear Layer expands or reduces the D dimension at the output. In the text, we will consider a feature tensor to be of shape (B, C, D), where C is feature channels and D is feature vector size.

ing of the grasp data based on shape latent is done using Feature-wise Linear Modulation (FiLM) [241], which uniquely scales and shifts the feature maps at every residual block based on the conditioning features. The decoder network is the same as the encoder but reversed, whose output is the reconstruction ($\mathbf{x}_h^*$) of the input. The decoder network is also conditioned on the shape latent.

Latent Noise Prediction Network

The noise prediction network for diffusion is a residual network similar to the grasp encoder and is conditioned on both the diffusion time step (t) and the shape latent ($\mathbf{z}_{pc}$). The network nominally contains four blocks with 32, 64, 128, and 256 output feature channels respectively. Since time step (t) is a scalar value, it is first projected to a higher dimensional feature using sinusoidal positional embedding [242]. This embedding is then transformed to the same shape as the shape latent ($\mathbf{z}_{pc}$) and added together to make the conditioning signal. This conditioning is again applied at each residual block using FiLM. The output of the noise prediction network is of the same size as the input grasp latent.

3.4.3 Dataset and Augmentations

We use ACRONYM [99] for our training dataset as justified in Section 2.3.3. ACRONYM contains 17.7 million grasp pose annotations on 8872 objects from 262 categories of ShapeNetSem [243] dataset. Each object contains 2000 grasp pose annotations (visualized in Fig 3.7). The grasps are sampled using an antipodal grasp sampling scheme [5] and labeled as success/failure based on attempts in a physics-based simulation [140]. As the ShapeNetSem dataset contains physically representative parameters for objects, simulation labels tend to be more reliable.

In the experiments, we use two subsets of ACRONYM data, named 1C and 63C, where C denotes the number of ShapeNetSem categories from the ACRONYM dataset. The 1C set is composed of objects from a single category i.e. 110 Mugs in the train set and 50 held-out Mugs in the test set. The 63C set contains 1100 objects from 63 categories in the train

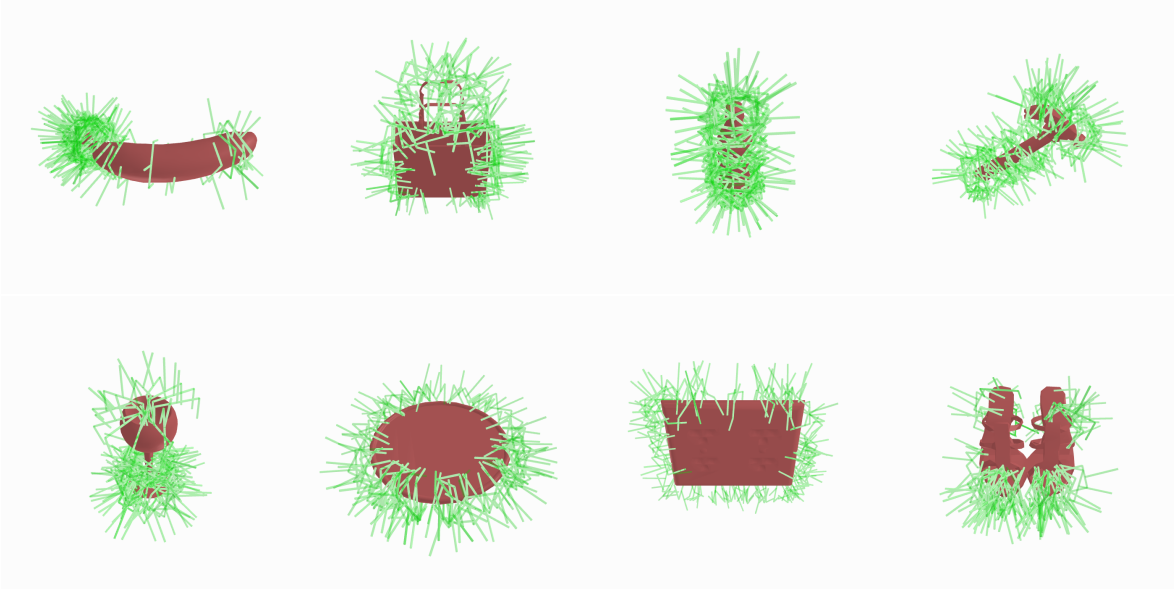


Figure 3.7: Examples of objects and their grasp annotations (random 100 samples) in ACRONYM dataset.

set and 400 held-out objects in the test set. We restrict the maximum number of categories for experiments due to available computational resources. We also show that the 63C set is a representative subset to evaluate the performance of the model and its transfer to the real world. The distribution of objects in the 63C category set is shown in Fig. 3.8, where the train-test splits are taken from [90]. For observations, we use full object point clouds and partial point clouds. Full object point clouds are sampled from the mesh surface. On the other hand, partial point clouds are rendered from a virtual depth camera at a distance between 30cm and 1m. In both cases, point clouds are randomly rotated, noised, and regularized to 1024 points which is the input to our point cloud encoder. Random rotation is sampled by selecting a random unit vector (axis) and a random magnitude (angle). Point jitter randomly perturbs the 3d coordinate of the points with a standard deviation of 1cm in all directions around the original to emulate jitter artifacts of a noisy sensor stream. Point dropout randomly reduces the availability of unique points by removing up to 40% of the input points. Since the input point cloud has to be of a fixed shape, the removed points are replaced by duplicating existing points. To avoid relying on an explicit canonical frame, the

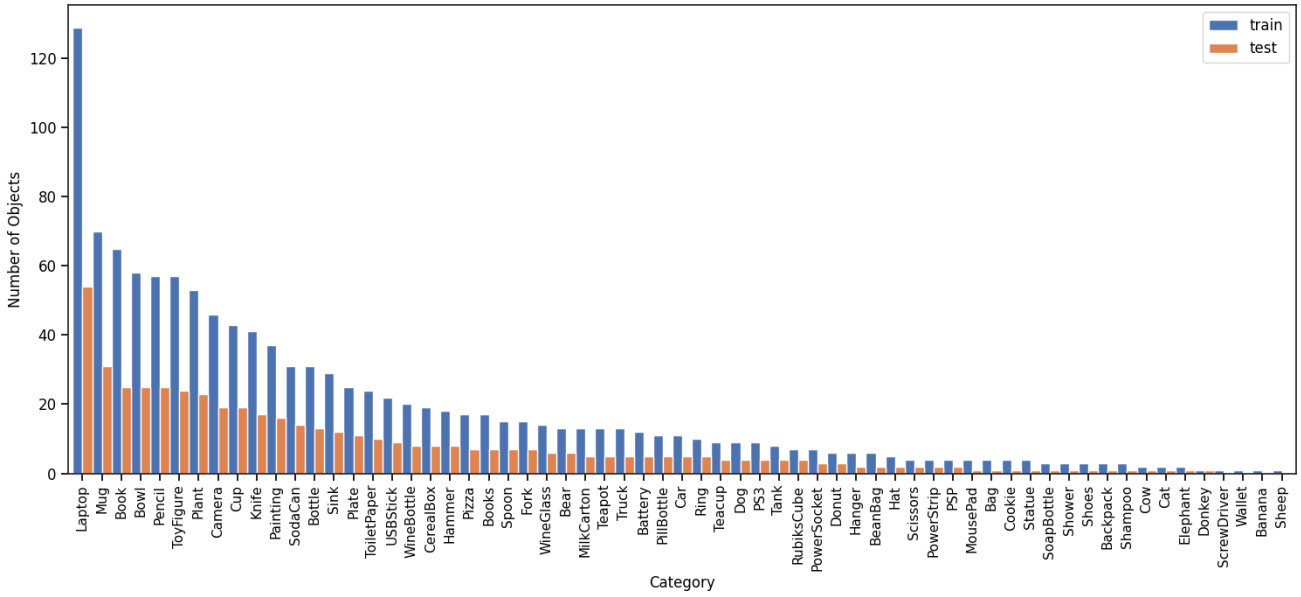
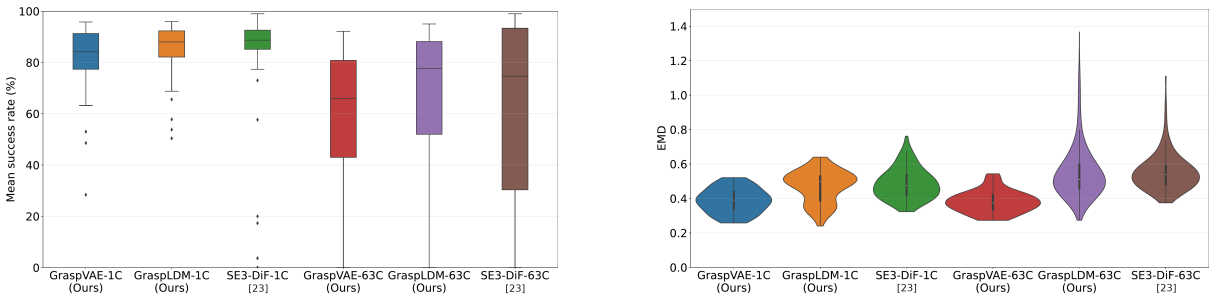


Figure 3.8: Distribution of training and test object instances in the 63C category set.

point cloud and grasp poses are both expressed in a frame with the origin at the centroid of the input point cloud.

3.4.4 Hyperparameters

We linearly anneal the ELBO modifying hyperparameter λ annealed linearly from $1e-7$ to 0.1 for 50% of the training steps and then held constant until the end of training. This ensures that the KL term does not vanish in the early stages of the training while also enforcing lower regularization to prioritize grasp pose reconstruction during the later stages. For the first term of the ELBO (Eq. 3.5) loss that measures reconstruction, we use L2 loss between inputs and outputs of the 6-parameter grasp pose. We use a multi-step schedule on the learning rate and scale down the learning by 0.1 every 1/3rd of the total training steps, going from 1e-3 to 1e-5 to avoid large optimization steps in later stages of the training. For the diffusion model, the noise variance (β_t) for forward diffusion is set to a linear schedule with $\beta_0 = 5e-5$ and $\beta_T = 1e-3$. The denoising diffusion formulation closely follows the original DDPM formulation with 1000 time steps. We track the exponential moving average of the model weights and use the model with averaged weights for evaluation. The training is done



(a) (b)

Figure 3.9: Grasp generation performance and scaling on full object point clouds ($N = 1024$). (a) The mean *success rate* in a simulation of 300 generated grasps poses per object. (b) SE(3) EMD between ground-truth grasp pose distribution and 100 sampled grasp poses (lower is better). SE3-DiF-1C and SE3-DiF-63C are the SE(3) Grasp Diffusion models from [123].

for a total of 180,000 steps (63C) and 100,000 steps (1C) on a system with an Intel Xeon Skylake CPU and two Nvidia V100 SXM2 GPUs with 16GB VRAM each.

3.5 Experiments

Evaluation

We evaluate *success rate* of the grasp poses generated by the models in a large-scale parallel simulation in Isaac Gym [244]. We evaluate the performance scaling by testing on 1C and 63C sets. In the simulation, we use only a Franka-Emika two-fingered gripper without the robotic arm and ignore gravity to avoid the effect of external factors on the evaluations. For each grasp pose, the simulation executes three steps to report a grasp success or a failure. (1) The gripper is spawned at the given grasp pose relative to the object with the fingers open. (2) The fingers are closed at the grasp pose. (3) The gripper lifts the object 1m in the +Z direction. Finally, a grasp is reported successful if the object remains attached to the gripper. Note that the evaluation is conservative as any penetration of the gripper in the first step or adverse contact in the second stage will result in a grasp failure. On the other hand, we observe that for many large objects, ground truth grasp poses in the ACRONYM dataset provide a very low or zero success rate. Therefore, we filter out the objects in each

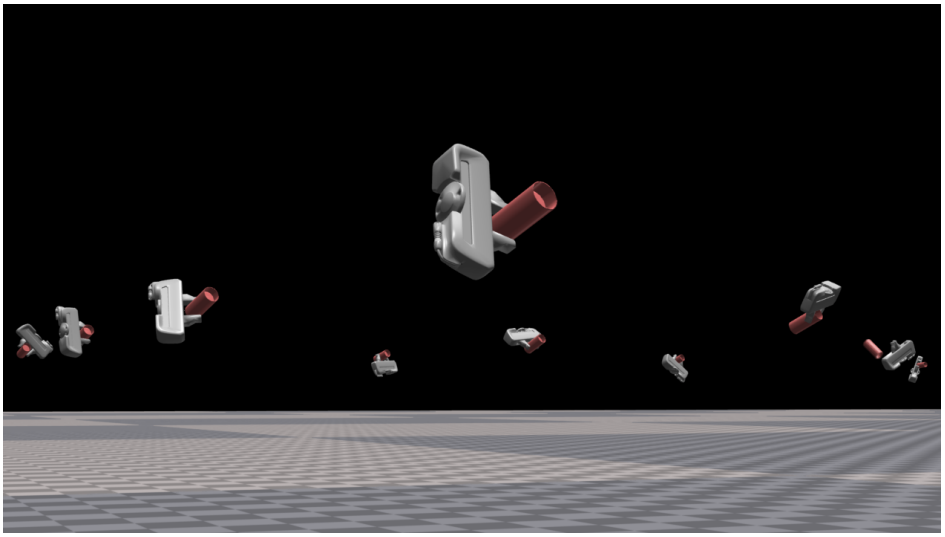


Figure 3.10: Multi-object grasping environments in Isaac Gym for success rate evaluation.

set for which the ground truth grasps fail more than 75% of the time.

While the success rate evaluates the quality of grasp generation, a model with a high success rate may generate grasps only around a small region of the object, instead of covering all the graspable regions around the object. Consequently, we also use SE(3) *Earth Mover's Distance (EMD)* metric [123] to evaluate how well the trained models learn the object-centric grasp pose distribution. The SE(3) EMD metric evaluates the empirical distance between two distributions of SE(3) poses. To compute this, we sample 100 grasps per object from a model and sample 100 random grasps from the ground truth datasets. To further validate the flexibility of GraspLDM, we provide additional results on sampling speed and task-conditional generation.

3.5.1 6-DoF Grasp Generation

Here, we evaluate the ability of GraspLDM models to learn the complex distribution of successful grasp poses on full object point clouds. For each object, we take 3 randomly sampled and augmented point clouds and generate 100 grasps on each. We aggregate the simulation results from all 300 grasp attempts into a *mean grasp success rate* per object reported

in Fig. 3.9a. To assess the benefits of adding the diffusion model, the success rate results of GraspLDM models are compared with the corresponding GraspVAE models, which is the base VAE model with the same weights. For the baseline, we use the SE(3) grasp diffusion model [123] (SE3-DiF-63C). To understand the scaling behavior, we present results from each network on 1C and 63C sets. Further, Fig. 3.9b reports the EMD metric that evaluates the ability to cover the distribution of grasps in the data.

For success rate, we outline two observations in Fig. 3.9a. First, the LDM models consistently improve upon the base VAE models, demonstrating that the sample quality improves by using a diffusion model in VAE's latent space. Second, the success rate performance of GraspLDM models scales better to a larger object set (63C). Comparing the median of the success rate on the test set in Fig. 3.9a, GraspLDM-1C improves the median success rate to 88% compared to 84% of GraspVAE-1C. On the larger 63C set, the GraspLDM-63C model improves the median success rate to 78% compared to 66% from GraspVAE-63C. GraspLDM models also improve the inter-quartile range in each case. For the comparison against the baseline, SE3-DiF-1C model reports a median success rate of 89% which is comparable to 88% the GraspLDM-1C model. On the larger 63C set, we observe that the GraspLDM-63C model registers 78% median success rate and distinctly higher inter-quartile range compared to SE3-DiF-63C. Therefore, GraspLDM models scale more favorably to larger object sets compared to the state-of-the-art approaches. During the tests, we also observed that the models available from [123] do not hold out a test split and are trained on all objects. Therefore, the comparison to SE3-DiF models [123] is conservative, which emphasizes the performance of GraspLDM on withheld objects.

Overall, the results validate our hypothesis that a diffusion model in the latent space can bridge the prior gap in the latent space to provide higher-quality grasp pose samples. Fig. 3.11 visualizes this effect where the latent de-noising moves latents corresponding to bad grasps towards good grasp regions during the reverse diffusion process. In terms of grasp failures, we observe a major portion of them in large objects whose graspable regions are far from the center of mass. When lifted in simulation, the reaction torque snaps the object out of the gripper fingers and therefore is reported as a failure.

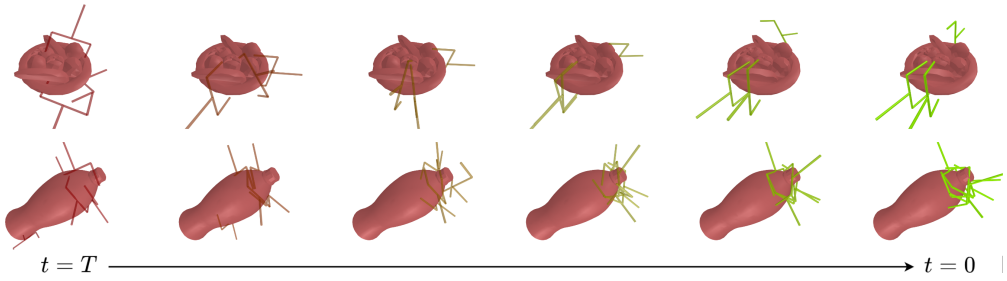


Figure 3.11: Visualization of latent space de-noising in GraspLDM. Reverse diffusion de-noises these latents from $t = T$ to $t = 0$, gradually moving the bad grasps towards regions of good grasps.

Fig. 3.9b shows that our models also register low EMD for most test objects, demonstrating good coverage of grasp modes in the ground truth data. The EMD performance of GraspVAE and GraspLDM models are comparable or marginally better than the state-of-the-art grasp diffusion models from [123]. However, the results show that while GraspLDM models effectively learn the grasp distribution, they have a slightly worse EMD compared to the GraspVAE models. We attribute this to two factors. First, reverse diffusion moves poor grasp poses (e.g. colliding or free-space grasps) towards a smaller number of high-density regions that tend to provide a higher success rate. Second, we follow the computation of SE(3) EMD from [123] by using cosine distance for rotation and metric Euclidean distance for translation. As a result, the EMD metric is more sensitive to the similarity in rotation despite translation having a potentially larger effect on the success of grasp execution.

3.5.2 Conditional Grasp Generation

For many manipulation tasks, the desired grasps are subject to a task context. Here, we demonstrate the flexibility of our architecture to provide this task context as conditional guidance in the latent space representing unconditional grasps. We do this by training a task-conditional diffusion model post hoc. For the proof-of-concept, we consider simple region-semantic labels ”top”, ”body” and ”bottom” as our conditioning signals. We use full point clouds of objects as inputs and pre-train a VAE without any task labels. We then include task labels only during the training of the diffusion model in the second stage. We associate

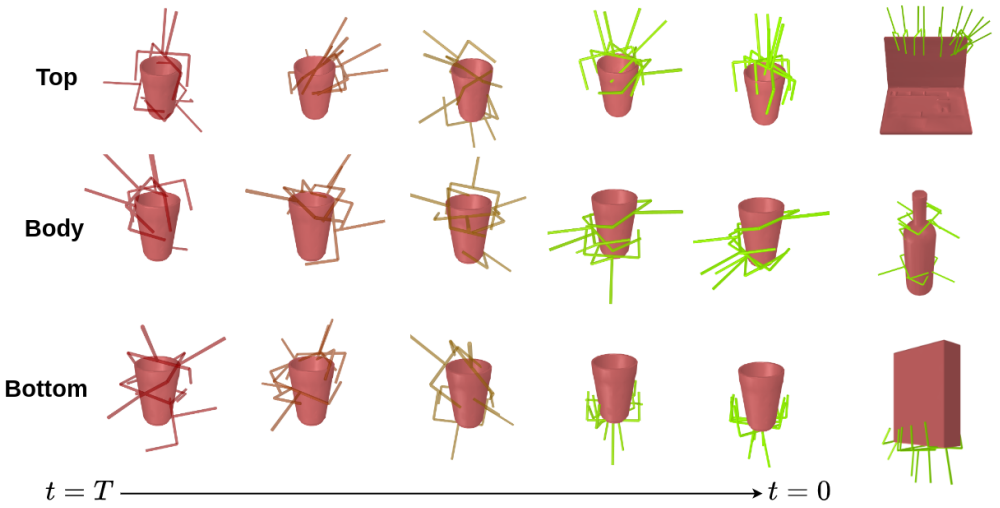


Figure 3.12: Visualization of task-conditional de-noising in the latent space for a region-semantic class label (top, body, or bottom)

a ground-truth class label of ”top”, ”body” or ”bottom” to each grasp based on whether the origin of the gripper's root is above, along, or below the unrotated object point cloud. We supply these labels to the diffusion model as an additional conditioning signal. The label and the point cloud latent are projected to a higher-dimensional embedding of the same shape. Subsequently, they are added together and used to condition feature maps at each residual block using FiLM module. We use a reduced subset of test objects for which these labels are meaningful (e.g. bottles) and remove those for which these labels are not relevant (e.g. plates). The final test set contains 200 unseen objects. We take pre-trained encoders and decoders from the GraspVAE-63C model and train the task-conditioned models (Task-GraspLDM) post-hoc, in less than 2 hours on a single NVIDIA V100 GPU.

At test time, we generate 50 grasps for each mode (top, bottom, and body) of an object. The input object point cloud is augmented with random rotations. To check whether the generated grasp complies with the input label, we reverse the rotation transformation and check the location of the gripper's root, as earlier. We report the precision between the labels of generated grasps and the labels supplied as conditioning. In this setting, the Task-GraspLDM model provides a average precision of 0.703 over all objects. This precision here is the ratio of the number of grasps generated in the correct region to the total number

Table 3.1: Reverse diffusion sampling speed-up and performance of GraspLDM-63C. Standard DDPM (1000 steps) is compared with a fast sampler- DDIM (100 steps) for the number of grasps generated (N_G) without re-training.

Sampling	$N_G = 100$ (s)	$N_G = 1000$ (s)	Median success rate
DDPM	7.39 ± 0.055	11.80 ± 0.290	$\mathbf{0.792}$
DDIM	$\mathbf{0.754 \pm 0.019}$	$\mathbf{1.149 \pm 0.009}$	0.756
VAE	$\mathbf{0.020 \pm 0.004}$	$\mathbf{0.0254 \pm 0.009}$	0.660

of grasps generated. We observe that the reverse diffusion process seamlessly moves a latent from a prior distribution to the desired task-conditional distribution, as visualized in Fig. 3.12. The tests show that GraspLDM allows effective injection of task conditioning post-hoc. Currently, the models are limited by the cases where there are no ground-truth grasps for a given label. For instance, if there are no 'bottom' grasp annotations for a laptop, the model generates grasps unconditionally all over the object. This can be addressed in future work along with more complex task contexts. Furthermore, this can be extended to more complex conditioning inputs like points, poses, and heuristics.

3.5.3 Reverse Diffusion Sampling

A notable disadvantage of using DDPM-based grasp generation is the time to execute large number reverse sampling steps in a sequential manner. In DDPM, a large T is required to ensure that the Gaussian conditional distributions in Eq. 3.7 are a good approximation [228]. A Denoising Diffusion Implicit Model (DDIM) [245] assumes a non-Markovian forward process to speed up the sample generation and requires a lower number of sampling steps. Further, it can be used as a drop-in sampler to use with a score network trained on the DDPM objective in Eq. 3.9, without any re-training. To assess the performance and sampling speed trade-off, we take the GraspLDM-63C model and compare DDIM sampler with 100 steps with the naive DDPM sampler. Table 3.1 compares the execution time of the reverse diffusion loop in DDIM with the baseline 1000-step DDPM. Using DDIM sampling as a drop-in replacement, the sampling time drops to 0.75s for 100 grasps and 1.1s for 1000 grasps. This is 10x faster than DDPM with a small loss in success rate (3.6%). The experiment was

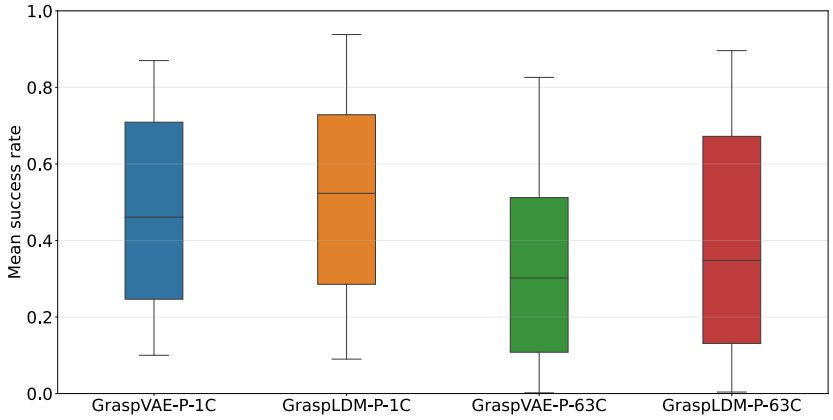
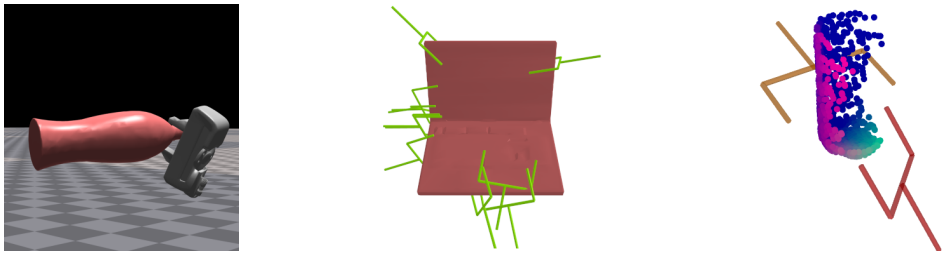


Figure 3.13: Grasp generation performance of GraspLDM models on partial point clouds of 1C and 63C object sets.

conducted on a system with NVIDIA RTX 3080Ti GPU and Intel i7-12800H CPU. Given the rapid progress in creating fast reverse diffusion samplers, our pipeline provides the flexibility to trade off speed against performance for such diffusion samplers. In contrast, SE3-GraspDiF [123] models cannot leverage new samplers without re-training.

3.5.4 Single-view Point Clouds

For many real-world use cases, only single-view point clouds are available for grasp generation. Therefore, we also evaluate the GraspLDM framework's ability to learn the distribution of grasps on noisy partial point clouds. For evaluation, we use the simulation environment as before, with the addition of a depth camera. The cameras are spawned randomly between 30cm and 1m from the object. We use the partial point cloud thus obtained to sample 25 grasps and execute them without filtering. For each object, we repeat these for 20 random camera poses. The success rate is reported in Fig. 3.13 for GraspVAE-P-1C, GraspLDM-P-1C, GraspVAE-P-63C, and GraspLDM-P-63C models, where 'P' implies that the model was trained on partial point clouds. In both cases, GraspLDM models improve the performance of the base VAE model. We found that for GraspLDM-P-63C models, higher capacity is required to compress a meaningful latent representation for 63C partial point clouds. We increase the grasp latent ($\mathbf{z}_h$) size to 16.



(a) (b) (c)

Figure 3.14: Special scenarios affecting GraspLDM performance : (a) Failure on large objects from adverse torque. (b) Failure of region-semantic conditional generation for ”bottom” grasps. (c) GraspLDM-P-63C generates colliding (orange) and free-space grasps (red).

Note that we do not use preferential camera viewpoints, grasp filtering, or a support surface in the background. As a result, this evaluation provides a conservative metric for 6-DoF grasp-generation performance. This can be considered as the approximate lower bound of the grasp generation performance for real-world use cases. A large number of failures result from grasps that collide with parts of the objects not visible in the partial point clouds as shown in Fig. 3.14c. Depending on the viewpoint, this occlusion confuses the model to see incomplete surfaces as edges where the object could be grasped. For larger objects where grasps are concentrated in a small region of the object, these regions may not be available in a partial point cloud.

These issues highlight the need for discriminating bad grasps from good grasps for real-world tests. This can be done using a classification network on top of GraspLDM (as in real-world tests in section 4.5) or preferably by adding a classification layer after the decoder. The latter problem of classifying grasps using a decoder of the current architecture, such that it can also be used for task-specific diffusion models, requires investigation and is out of the scope of the current work.

3.5.5 Ablation Study

Here, we ablate our design and hyperparameter choices to understand the impact of each component on the model performance. In each case, we take the baseline GraspVAE-63C and GraspLDM-63C model configuration and retrain it with a single variation. The evaluation follows the same protocol as in Section 3.5 except for total training steps that are reduced to 150,000.

KL Annealing

We evaluate the impact of KL weight (λ) in the ELBO objective (Eq. 3.5) and linear annealing on grasp synthesis performance. Five different KL weights from 0.01 to 1.0 are tested, each with constant and annealed schedules. In the case of annealed schedule, the weight is linearly increased from $1e-7$ to the specified value for 50% of the training steps and then held constant. In this case the value denotes the final KL weight.

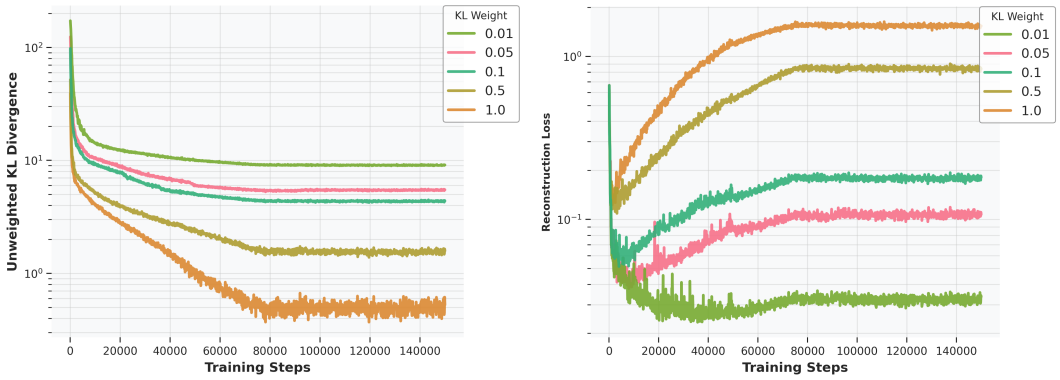
The results in Table 3.2 firstly confirm that the KL weight is crucial for the performance of the GraspVAE and GraspLDM models. For a true ELBO objective (λ = 1.0), the model fails to learn a meaningful generative model because of over-regularization of the latent space. As the KL weight is reduced, the GraspVAE model performance increases first and then decreases, with the maximum grasp success rate at λ = 0.1. Fig. 3.15 shows the evolution of the KL divergence and reconstruction loss during training of the base GraspVAE models. The trade-off between KL divergence and reconstruction loss is evident in the training plots. For higher λ, the KL divergence dominates the loss at the cost of reconstruction accuracy. Intuitively, this means that the approximate posterior is heavily regularized and each sampled latent from the posterior distribution carries less information about the data sample to allow for good reconstruction. Towards, lower λ, lower regularization results in each sampled latent carrying more distinct information about the training sample, allowing more accurate reconstruction. On the other hand, better reconstruction accuracy does not directly imply better test time performance, as seen in evaluations for λ = 0.01 and 0.05. For $\lambda = 0.1$, the model learns a generative model that generates meaningful samples from a regularized

Table 3.2: Ablation Study: Impact of KL weight and annealing on grasp synthesis performance of GraspLDM models. * denotes default configuration.

KL Weight Schedule	Weight (λ)	Median Success Rate (%)	Δ
GraspVAE-63C			
Annealed*	0.01	52.7	-17.5
	0.05	63.1	-7.1
	0.1*	**70.2**	**–**
	0.5	18.4	-51.8
	1.0	1.2	-69.0
Constant	0.01	53.9	-15.4
	0.05	65.0	-4.3
	0.1	**69.3**	**–**
	0.5	18.7	-50.6
	1.0	0.8	-68.5
GraspLDM-63C			
Annealed*	0.01	69.7	-9.5
	0.05	77.9	-1.3
	0.1*	**79.1**	**–**
	0.5	11.7	-67.5
	1.0	0.6	-78.6
Constant	0.01	70.0	-8.6
	0.05	77.1	-1.5
	0.1	**78.6**	**–**
	0.5	10.9	-67.3
	1.0	0.4	-78.2

latent space that provides test-time generalization.

A notable suprise is that we do not observe evidence of complete posterior collapse in GraspVAE-63C models, even for constant schedule of high KL weights. We expected that constant and higher KL weights would lead to posterior collapse, resulting in an autoregressive decoder as discussed in Section 3.2.3. Such an autoregressive decoder would exhibit a rapid drop in both the KL divergence and reconstruction loss. This is not observed during training. Since KL annealing is a practice to alleviate posterior collapse, then it is not entirely unexpected that there is no significant performance difference between the constant and annealed schedules for the same KL weight. It is not entirely clear why we do not observe posterior collapse at higher KL weights. There may still be a higher posterior gap in the models trained with higher KL weights, which would explain the performance drop and the reconstruction error variations. This analysis needs further investigation in future work.



(a) (b)

Figure 3.15: GraspVAE-63C: KL Divergence and Reconstruction Loss during training with different KL weights.

Another interesting observation is that the GraspLDM-63C models are less sensitive to lower KL weights, i.e. less regularized latent spaces, compared to GraspVAE-63C models. This is intuitive, as the diffusion model can fit a more complicated prior in the latent space, given sufficient capacity. As long as the decoder can meaningfully reconstruct the data from latent, the prior distribution of latent can be fit with the diffusion model. This is why the performance difference between GraspLDM models at λ = 0.01, 0.05, and 0.1 is not significant like GraspVAE-63C models. This might also indicate that our baseline diffusion model may be over-parameterized for the task, as it effectively fits more complicated priors at lower regularization. As a consequence, a more efficient diffusion model could be used to achieve similar performance at λ = 0.1.

FiLM Conditioning

GraspLDM uses a shape latent i.e. a learned global object descriptor to condition the encoding, decoding and denoising processes. The objective of the conditioning is to use this information to change a subset of parameters of the network that allows it to introduce dependency of output features on the shape latent. In FiLM, we use a small sub-network to take the shape latent as input and predict scale and bias for the features at conditioning layers. The features at those layers are then transformed using the predicted scale and

Table 3.3: Impact of FiLM conditioning on grasp synthesis performance. * denotes default configuration.

Conditioning Type	Median Success Rate (%)	Δ
GraspVAE-63C		
FiLM*	**70.2**	**–**
Linear Projection + SiLU + Addition	65.2	-5.0
GraspLDM-63C		
FiLM*	**79.1**	**–**
Linear Projection + SiLU + Addition	70.8	-8.3

shift. We compare this with a simple method of biasing the encoder/decoder features at conditioning layers i.e. without scaling. In the latter, we take the shape latent, project it to the same dimension as the feature map using linear layers, followed by the SiLU activation function and adding it to the feature map. This can be seen as per-neuron linear biasing. The layers at which features are conditioned are kept unchanged from FiLM configuration. Table 3.3 shows the performance comparison of GraspLDM models with and without FiLM. The results clearly show the merit of FiLM conditioning in improving the grasp synthesis performance in both GraspVAE and GraspLDM models.

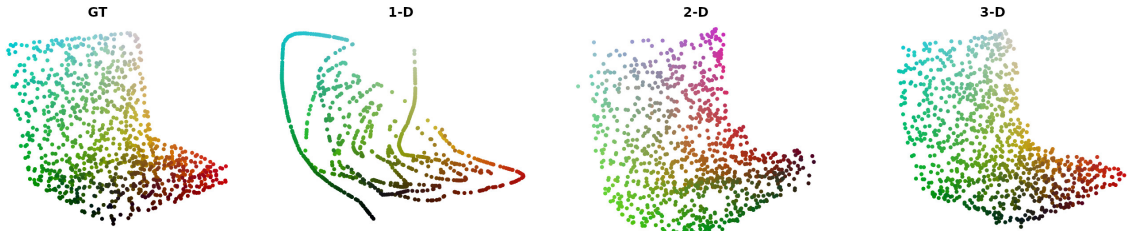
One can also use other methods like hypernetworks [246] and simple concatenation. The hypernetwork-based conditioning is usually used for small networks like an MLP, where small number of parameters are easy to predict. For larger networks like GraspLDM, the implementation is challenging. This was not attempted in this work. We were also unable to test the concatenation-based conditioning of the default 3-channel shape latent consistently in the baseline configuration of GraspLDM models. In a naive implementation, one could flatten the shape latent and concatenate each of the PVCNN features with it. However, this would significantly increase the number of parameters at conditioned layers and change the topology of the bottlenecks in residual sub-networks. This can be addressed in the future work.

Table 3.4: Impact of shape latent size on grasp synthesis performance. * denotes default configuration.

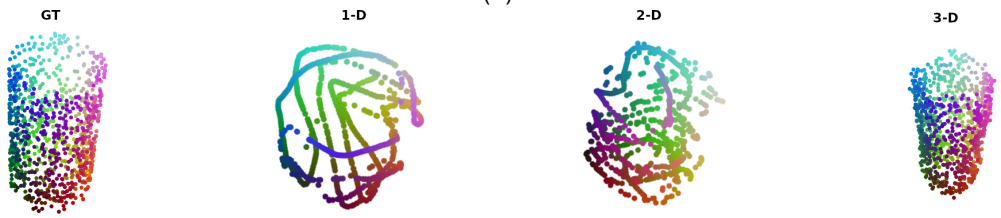
Latent size	Latent channels	Parametric capacity (#)	Median Success Rate (%)	Δ
GraspVAE-63C				
64*	**3***	**192***	**70.2**	**–**
64	1	64	29.3	-40.9
128	2	256	58.5	-11.7
256	1	256	50.6	-19.6
GraspLDM-63C				
64*	**3***	**192***	**79.1**	**–**
64	1	64	35.6	-43.5
128	2	256	65.0	-14.1
256	1	256	53.9	-25.2

Shape Latent Size and Structure

In the GraspLDM framework, we use a modular point cloud encoder to encode the observed point cloud, which provides better training efficiency. But, the information is encoded (and compressed) to a much smaller feature vector or tensor representing the object shape. This descriptor plays a crucial role in how the point cloud information is conveyed to the grasp encoder, decoder, and the diffusion model. Since the point cloud encoder is trained jointly with the VAE on the ELBO objective, it is difficult to interpret how this shape latent stores local and global information. However, we can comment on general trade-off involved in encoding. A smaller latent size allows for more efficient operations using the encoded latent, especially in GraspLDM where network layers are repeatedly conditioned on this latent using FiLM. However, the lack of capacity limits both the information that can be stored in descriptor and the separation of distinct samples in the latent space, especially for large datasets with diverse samples. The latter implies that under input and feature noise, the model may easily confuse between different kinds of objects and generate poor grasp samples. On the other hand, a larger latent size can store more information about the object shape, but it can also lead to overfitting and increased computation cost in conditioning at each layer. To balance the two effects, we are interested in understanding the impact of latent size on the model performance. We evaluate the performance of GraspLDM models with latent sizes of 8, 16,



(a)



(b)

Figure 3.16: Point cloud reconstruction from latents with different number of channels for (a) Laptop (b) Cup. The reconstructions are obtained by training a decoder (encoder configuration reversed) to minimize 3D EMD loss.

and 32. In addition, we also vary the number of channels of the latent. The results are shown in Table 3.4.

The results show that removing the latent channels in the default configuration of GraspVAE-63C and GraspLDM-63C, such that the latent is a 1D tensor, significantly reduces the grasp success rate. The natural interpretation would be that the latent with 3 channels has higher parametric capacity and is therefore more expressive. However, we observe that simply increasing the parametric capacity is not sufficient. The structure of the latent is also important. This is evident in the results, where the 3-channel latent with the length of 64, significantly outperforms the 2-channel and 1-channel latent shapes despite having lower parametric capacity. This points to inductive bias of the point cloud encoder (PVCNN architecture) towards 3-channel data structure similar to a point cloud. Such 3-channel latent structure is also utilized in [229], where they call it latent points.

To further understand if the PVCNN architecture behaves differently under different number of channels, we train a point cloud autoencoder that encodes a point cloud to a latent and tries to recover the original point cloud from it. The encoder is the same as the point

Table 3.5: Impact of conditional grasp latent size on grasp synthesis performance. * denotes default configuration.

GraspVAE-63C		
Latent size	**Median Success Rate (%)**	**Δ**
4*	70.2	-1.2
8	70.4	-1.0
16	**71.4**	**–**
32	69.7	-1.7
GraspLDM-63C		
4*	**79.1**	**–**
8	73.1	-6.0
16	74.5	-4.6
32	71.4	-7.7

cloud encoder in GraspLDM models. The decoder is simply a mirrored version of the encoder. The autoencoder is trained to minimize the 3D EMD loss. The reconstruction from the decoder for different channel configurations is shown in Fig. 3.16. It shows an interesting phenomenon where point cloud reconstructed from the 1-channel latent resembles warped line wrapped approximately around the original shape. Similarly, the 2-channel latent results in a point cloud that resembles warped plane and 3-channel latent most accurately represents the original shape. As such results are dependent on the decoder design, it might be possible to design a decoder that can accurately reconstruct the point cloud from a 1-channel latent. However, given the correlation with grasp success results in Table 3.4, it is likely that the 3-channel latent structure stores object geometry most effectively for our architecture and is most suitable for the task.

Conditional Grasp Latent Dimensions

Similar to the shape latent, the conditional grasp latent size is equally important for performance. Recollect that this parameterizes the latent space. The goal of VAE training is to learn a continuous latent space with potentially disentangled latent variables. Smaller latent size forces regularization and the training samples are fit to latents that are closer in the latent space. This is desirable for learning a continuous latent representations. However, if the latent space is low-dimensional, it can lead to poor separation of diverse samples (e.g.

different categories) in the latent space resulting in poor sample quality. While a larger latent size can store more information about the training samples, the samples may be spread too far apart in the latent space, leading to overfitting posterior. To this extent, we test four different latent sizes for the conditional grasp latent in GraspVAE-63C and GraspLDM-63C models. The results are shown in Table 3.5.

The surprising observation here is that both GraspVAE-63C and GraspLDM-63C models are less sensitive to the change in latent size. The performance of grasp latent of size 4 is relatively close to the latents of size 8, 16 and 32. This might have to do with lower regularization in the latent space induced by lowering the KL weight to 0.1. On the other hand, it might also indicate that all these latent sizes are equally suboptimal choices for learning a continuous latent space for this task. A more detailed investigation on this is left to future work.

3.6 Discussion

GraspLDM demonstrates that latent diffusion provides a powerful mechanism to improve generative performance, flexibility to downstream tasks, and sample quality. Through various analyses, we have addressed challenges facing generative models for grasp synthesis. However, there are still limitations and areas for improvement. The GraspLDM models can generate unlimited grasps around relevant regions of the object. However, some of these can be poor in quality, such as those that collide with the object or are unstable. This is especially the case when dealing with single-view and partial point-clouds. In practical applications, a separate classifier needs to be trained to discriminate the generated grasps and select the best one. One problem with external classifier is that it needs to be trained separately and hinders GraspLDM models to be truly modular for task-specific generation. In task-specific settings, the classifier needs to discriminate grasps that are stable and suitable for the task at the same time. Some notion of grasp quality should be bootstrapped inside the GraspLDM architecture in the future that supports its modular nature.

The choice of using a VAE brings with it the benefits of a regularized latent space but

also non-trivial challenges in training. We have tackled several challenges in this chapter with the use of latent diffusion prior, modular point cloud encoding, FiLM conditioning and KL weighting. However, efficiency of the GraspLDM models can be further improved. One such direction is by exploring the role of capacity of the encoder and the decoder. Currently, both networks are nearly similar in structure and capacity. While over-parameterization of the encoder can help reduce the posterior gap, over-parameterized decoders can lead to higher overfitting [232]. Similarly, the performance of the latent diffusion model at lower KL weights in Table 3.2 hinted at possible over-parameterization of the diffusion model. Finally, reverse diffusion sampling speed-up demonstrated with DDIM sampler still takes around 1s. Recent advances on efficient samplers for diffusion models can be leveraged to make the GraspLDM models faster. Together, these potential improvements can make GraspLDM more efficient and performant for 6-DoF grasp synthesis.

It is important to note that our models are trained on only 63 out of 180 categories in the ACRONYM dataset due to compute constraints. We believe that the performance and generalization can be improved further by incorporating the entire dataset in the training. A unique advantage of the GraspLDM is its ability to enable flexible plug-and-play approach to task-specific grasping that inherits from a general grasping model. The validity of this idea is validated in a preliminary experiment for class-conditioned generation in Section 3.5.2. However, the analysis needs to be extended to more complex task conditioning inputs like points, poses, and embeddings. Finally, the GraspLDM models are object-centric and scene-agnostic. This introduces the requirement for external collision checking, which can be a significant bottleneck in cluttered scenarios. To show the ability of GraspLDM to generate grasps in cluttered scenes, they can be trained on synthetic cluttered scenes. However, the ideal solution would be to incorporate efficient collision checking in a manner that allows GraspLDM to preserve its scene-agnostic and general purpose synthesis capability.

3.7 Summary

Learning generalizable representations for grasp generation is fundamental to robotic grasping. In this chapter, we proposed GraspLDM- a novel generative framework for object-centric 6-DoF grasp synthesis using latent diffusion. Our architecture allows de-noising diffusion models to be used as expressive priors in the latent space of VAEs. GraspLDM enables efficient learning of the complex multi-modal distributions of object-centric 6-DoF grasp poses on point clouds. We conduct large-scale simulation tests to show that GraspLDM outperforms baseline methods and provides a 78% median success rate on our test set of 400 objects from 63 categories that were not seen during the training. GraspLDM shows good grasp generation performance while scaling favorably to large object sets compared to existing methods using generative modeling. We also show concrete solutions to challenges faced by existing generative models and provide insights into the design choices that impact the performance of GraspLDM. These proposals are validated by an extensive ablation study. The GraspLDM framework is also validated on single-view point clouds generated in simulation to understand the lower-bound performance in arbitrary 6-DoF grasping scenarios. Finally, we demonstrate the flexibility of our architecture to train efficient task-specific models and to use speed-up reverse diffusion for downstream applications.

Contributions

1. **We introduce a new generative modeling framework for 6-DoF grasp synthesis using latent diffusion.** To the best of our knowledge, no other work has applied latent diffusion for 6-DoF grasp synthesis for scalable real-world parallel-jaw grasping.

2. **We show that a diffusion model in the latent space improves the grasp sample quality of a standard VAE model.** In simulation tests, our latent diffusion models improve generation performance. Further, they transfer to the real world to provide more stable grasps from single-view point clouds.

3. **We demonstrate that our architecture enables the injection of task-specific con-**

ditioning in generation with limited additional training effort. The separation of VAE and diffusion model introduces flexibility that allows rapid training of task-specific models in the latent space.

4. **We make the models and code available open-source for the community to use and extend.** The resources are available at `https://github.com/kuldeepbrd1/GraspLDM`.

Chapter 4

Object-centric System for Real-world Grasping

4.1 Introduction

The simulation environment offers many benefits, primarily by enabling controlled analyses. However, the usefulness of a robotic system, model, or algorithm can only be validated in the real world. Real-world manipulation systems are prone to inaccurate kinematic models, which limits the precision of finger placement on objects. In robotic grasping specifically, the complexity of contact dynamics, combined with stochastic noise from sensors and actuators, creates significant disparities between simulation and reality. Vision-based sensors introduce additional complexity due to their sensitivity to photometric changes and calibration errors. For depth sensors using stereo disparity or LiDAR, depth uncertainty increases on reflective surfaces, around object edges, and in regions with low texture.

The challenge of transferring performance from simulation to the real world has been a persistent issue in robotic grasping. Analytical grasp synthesis approaches, which rely on form or force-closure analysis with idealized hand-object contact models, often prove fragile in practice [31]. These approaches may fail to generalize to objects of certain sizes [247] or underperform in the real world, particularly when compared to techniques like imitation [32].

In the modern data-driven paradigm, the models need to be learned on a large amount of data that is impractical to generate in the real world. Then, the primary challenge lies in training models with simulation-generated data that can effectively generalize to various real-world settings.

In the previous chapter, we introduced GraspLDM - a powerful generative model for 6-DoF grasp synthesis. GraspLDM models learn a distribution of successful grasps conditioned on an object point cloud in the latent space. From this latent space any number of grasps can be sampled. In this chapter, we introduce the system that enables GraspLDM to be applied in real-world scenarios. First we outline the system architecture in Section 4.2 and describe the individual components in detail in Section 4.3. The software and hardware architecture are covered in Section 4.4. Finally, the validation of the system with GraspLDM models in real-world robotic setups is presented in Section 4.5.

4.2 Grasp-O: Object-centric Grasping System

The objective of this system is to utilize grasp samplers like GraspLDM and facilitate grasp execution with minimal human intervention. The primary application is to enable evaluation and validation of object-centric grasping methods in a consistent manner. To accomplish this, it must allow for straightforward integration and benchmarking of other grasp samplers. The secondary objective is to maintain modularity, allowing adaptation for downstream applications of object-centric grasping beyond pick-and-place tasks. Our proposed pipeline, called $Grasp - O$, is shown in Fig. 4.1. The execution begins with a calibrated camera observing a robot's workspace, capturing an RGB-D frame of the scene. The system then requires selection of the target object to be grasped. In autonomous manipulation routines, this selection would typically be made by a high-level task planner providing a semantic prompt. For our implementation, we use operator input as a proxy for such a task planner, assuming the target object is within the camera's field of view. The operator can indicate the object of interest through various inputs: bounding box specification, region selection, or point identification. This input is processed by a segmenter based on segment-anything [221, 248],

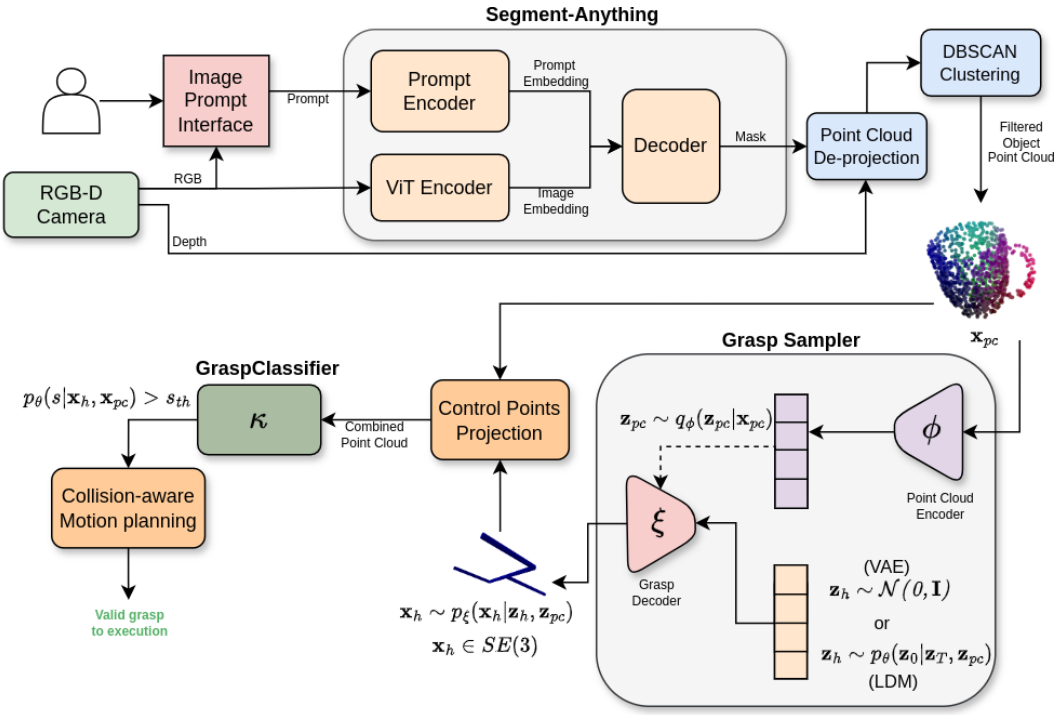


Figure 4.1: Overview of grasp execution in the real world using Grasp-O

which generates a segmentation mask for the target object. The system uses this mask to extract the object's point cloud from the aligned depth image. A reference frame is then attached to the geometric center of the observed points, and point coordinates are transformed to this frame. The object point cloud is processed by the grasp sampler, which generates N grasp poses conditioned on the point cloud in the same reference frame. A classifier then assigns a success probability to each grasp pose, ranking them in a decreasing order. Grasp poses with success probabilities above a threshold (s_{th}) are sequentially evaluated for feasibility using a motion planner. This planner incorporates joint limits, path constraints, available environment model, and dynamically assigned collision objects. The first feasible grasp is executed by the robot, after which the system returns control to the operator and awaits the next request. The following section details the system components along with hardware and software architecture. While the system is designed to be modular and extensible to multiple prompts, scenarios, and configurations, this chapter focuses specifically on the end goal of evaluating grasp synthesis models like GraspLDM in the real-world.

4.3 Pipeline

4.3.1 Classifying 6-DoF Grasp Poses

While grasp samplers like GraspLDM and GraspVAE can generate numerous diverse grasp poses, some poses may be suboptimal. These models learn from limited positive-only samples of successful grasps. The loss of precision or noise in learned representations may generate problematic grasps: gripper poses that collide with the object, grasp in free space away from the object, or target regions not visible in the point cloud. Given these challenges, classifying and ranking grasp quality is essential for real-world grasping. The need to discriminate and rank grasp poses is well-established in the literature, appearing as grasp success prediction, evaluation, and classification. When a full object model is available, analytical force closure conditions with an ϵ-metric threshold can be used [21]. However, for a single RGB-D frame, grasp classification must rely on noisy and unstructured depth information. Consequently, data-driven binary classification has become prevalent. Previous works have approached this challenge through binary classification [91, 129, 249, 102, 89] or continuous metric learning [112, 64]. For direct grasp pose regression, the prediction of grasp quality can be bootstrapped within the same network [90, 95, 98].

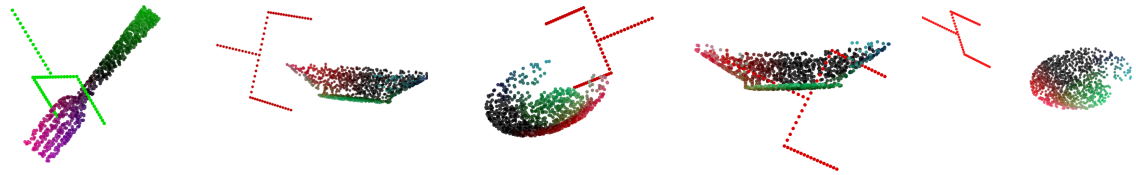
In our pipeline, we introduce GraspClassifier, an efficient grasp pose classifier that works effectively with GraspLDM models. The GraspClassifier learns to assign a success probability $p(s|\mathbf{x}_{pc}, H)$ for a pair of object point cloud and grasp pose. While previous approaches [91, 129, 249] have utilized CNNs with various image representations, we leverage the object point cloud which is already used during grasp synthesis. The quality of a grasp pose depends on the spatial relationship between object and the gripper at a query pose. Such spatial relationships in 3D are more explicitly captured in 3D representations like point clouds than in images. This makes it simpler to predict collision, free-space, or contact-poor grasp poses. Importantly, we can adapt the PVCNN architecture described in Section 3.4.2 with minimal modifications to build the classifier.

We represent gripper poses using a set of points, similar to [89]. However, rather than sampling points from a gripper mesh, we employ a set of 76 control points to minimize

computational overhead. These control points are defined in the gripper frame and sized for a nominal finger sweep width of 82mm, corresponding to the Franka hand. Recall that grasp poses are sampled relative to a reference frame at the observed point cloud's centroid. The gripper control points are transformed to this object frame and combined with the object point cloud into a composite point cloud ($\mathbf{x}_c = \mathbf{x}_{pc} \cup \mathbf{x}_g$), as shown in Figure 4.2.

Network Architecture

We use PVCNN backbone for the GraspClassifier. To distinguish between object points and gripper control points, a binary feature label is assigned to each point. PVCNN, like other point-set architectures, maintains spatial and semantic information branches along the network. The spatial branch tracks point coordinates while the semantic branch manages relevant features. Features at each layer depend on point coordinates and corresponding point features from the previous layer. The binary labeling adds an inductive bias, focusing the learning on relationships between these point subsets, guided by classification loss. The input is a N_cx3 tensor, where $N_c = 1024$(object) $+ 76$(gripper), is the total number of points. This is processed by a PVCNN $(0.5C \,|\, 0.5R)$ network with two PVConv blocks and two MLPs, where the architecture definition follows Section 3.4. This network provides a feature tensor of size 1024xN_c, which is then processed by the classifier module that first converts this feature tensor to 1D tensor of size N_c using 1-D convolution layer and then converts it to a single logit using an MLP with hidden layer of size 128. The logit is then transformed through a sigmoid function to produce class probabilities between 0 and 1. Each network forward pass predicts success probability for a single grasp pose using the corresponding composite point cloud, emphasizing the need for only adding a minimal yet representative set of gripper control points.



(a) (b) (c) (d) (e)

Figure 4.2: Illustration of positive and negative grasp samples used in training. (a) Point cloud representation of object grasp pair used as input to the classifier. (b) Negative grasp sample from ACRONYM dataset annotations that fail in simulation (c) Negative grasp sample with finger sweep in an occluded region with no points (d) Negative grasp colliding with the object (e) Negative grasp in the free space around the object

Training

The model is trained to minimize the binary cross-entropy loss:

$$L_{BCE}(s, \hat{s}) = -\frac{1}{N} \sum^{N} (s \log(\hat{s}) + (1 - s) \log(1 - \hat{s})) \tag{4.1}$$

where s is the ground truth success label and $\hat{s}$ is the predicted success probability and N is the number of samples. To train the grasp classifier well, we require positive and negative samples of grasps. Since the ACRONYM dataset provides success annotations for each grasp sample, we use it as the base training dataset. However, the grasp samples in the base dataset are sampled such that mid point of the gripper is close to the surface. We find that this limits the diversity of negative samples for the classifier. As a result, training the classifier with base dataset alone does not classify colliding and free-space grasps very well. To solve this, we synthetically generate more negative samples by perturbing existing grasps. First, we apply a random rotation and translation. Then, we check the minimum distance between the gripper and the object points. If the minimum distance is below the median intra-point distance threshold, the grasp is assumed to be colliding with the object as shown in Figure 4.2d. If a grasp is not colliding with the object, we check the sweep area between the two finger tips has any object points. If there are no object points in this sweep area, the grasp is either in the free space or in an occluded region of the object as shown in

Table 4.1: GraspClassifier performance metrics on validation set

Metric	Value
Accuracy	0.87
Precision	0.86
Recall	0.97
F1 Score	0.92

Figure 4.2c and 4.2e.

The control points used to represent the gripper are pre-generated from wrist, finger and finger sweep dimensions of the Franka hand. Specifically, we take the four links defining the root, wrist and the fingers and subsample a total of 76 points to represent the gripper. We train the classifier network on RGB-D renders of objects from the 63 category set in ACRONYM [99]. To avoid the data loading latency of online rendering in the training loop, we use a set of 8000 renders on the objects of 63 categories, similar to the training of GraspLDM-P-63C and GraspVAE-P-63C. Combining positive and negative grasps, each render contains 2000 grasp annotations in the set. The training is done two NVIDIA V100 16GB GPUs with a batch size of 192 composite point clouds. To further optimize data-loading latency for each render, we compose a batch with 36 random grasp samples for 6 random objects. As opposed to loading one random pair of grasp pose and point cloud, this allows to reduce per worker latency of fetching the training samples that includes point cloud de-projection, grasp pose augmentations for negative samples and CPU-to-GPU data transfers. In the batch, we provide approximately 50% negative samples per object by aggregating negative samples from the dataset as well as from the augmentations discussed above. For the model to be able to generalize to unseen sensor noise and object poses, we apply random rotations, point jitter and point dropout to the input. The learning-rate is scheduled for a multi-step decay, scaling down by a factor of 0.1 at every quarter of the total training steps from $1e-3$ to $1e-6$. The model is trained for 200,000 steps taking 19 hours on two Nvidia V100 GPUs. The validation metrics for the model are provided in Table 4.1.

Inference

During experiments, the grasp classifier takes batches of grasp poses generated by GraspLDM model and infers the probability of success for each. We sort the grasp poses in decreasing order of success probability above a threshold of 0.5. This ranking specifies the order in which grasps are attempted.

4.3.2 Object Instance Segmentation

Our models for grasp generation and classification are trained for 6-DoF grasps on individual objects. While it is possible to train GraspLDM and GraspClassifier models with full scene point clouds similar to [90], this approach has limitations for general purpose grasping as discussed in Chapters 2 and 3. There are two additional challenges arise with pipeline like ours. First, the graspable objects typically occupy only a small area of the observed region/frame, meaning that the input contains predominantly points from the scene (e.g., table or background) that are not directly useful. Second, the memory requirement per forward pass of the model increases with the number of input points. Simultaneously, the performance of grasp generation degrades with lower point density. This creates a trade-off between memory and point density that often becomes specific to the scenes it is being trained on. For instance, for many table-top scenes, the camera is assumed at a fixed height and angle or each within a narrow range. In contrast, object-centric models are efficient and flexible when the object can be segmented from the scene.

To segment an object, a segmentation model must handle an arbitrary number of instances of unknown objects. Early works relied on 2D [250, 251] or 3D [96, 93] semantic segmentation networks. These approaches typically provide instance-level masks for defined categories. However, they do not generalize to applications like unknown object grasping where the robot may encounter unseen objects from out-of-distribution categories. For robotic grasping, Xie et al. [252] proposed UOISNet, which uses a two-stage approach using depth for instance and mask initialization while using RGB to refine the masks. The models are trained on synthetic datasets of cluttered table-top scenes and demonstrate transfer to

(a) (b) (c)

Figure 4.3: Cluttered scene segmentation using SAM [221]. (a) Automatic mask generation produces fine-grained masks for semantically meaningful regions of the objects, including highlighted text on a sheet of paper. (b) Granularity of segmentation masks can be adjusted by modifying filtering and inference hyperparameters. (c) Prompt-conditional segmentation of specific objects in the image (coffee mug) by providing a point coordinate that belongs to the object of interest.

real-world scenarios including robotic grasping. UOIS-Net has been utilized for real-world grasping tests in other previous works [89, 90]. However, the network can produce noisy masks, especially outside of table-top settings. More recently, work on foundation models pre-trained on large-scale data has demonstrated unprecedented performance and in-the-wild generalization across vision tasks [253, 254, 255, 256].

In particular, segment-anything model (SAM) [221] demonstrates zero-shot capabilities across promptable and automatic instance segmentation tasks that are comparable to or better than state-of-the-art models dedicated for each task. SAM provides high-quality granular masks in images with extensive control over the mask generation process, as shown in Figures 4.3a and 4.3b. More importantly, segment-anything is designed to provide ambiguity-aware masks in a promptable setting, where a prompt specifies what to segment in the image. The prompts can be points, bounding boxes, regions, or natural language input. Figure 4.3c shows a predicted mask for a user prompt in the form of a point in the image where the user wants the mask. The SAM model architecture consists of a pre-trained image encoder that computes an image embedding using a Vision Transformer (ViT). This represents the largest portion of the model computation and only needs to run once per image. A modular prompt encoder embeds the prompt, and a lightweight mask decoder network combines the two embeddings to predict prompt-conditioned segmentation masks. With notable follow-up work on making SAM faster and memory-efficient [257], grounding in

language instructions [248], and applicable to efficient video segmentation [222], it emerges as the most suitable choice for object instance segmentation for general-purpose robotic grasping.

While multiple interactive prompt interfaces such as natural language, bounding boxes, and masks are possible, for evaluation, we primarily use the model with point prompts for simplicity. In this mode, the model takes in 2D point(s) in the image as inputs and provides an instance mask as shown in Fig. 4.3c. This enables users to easily indicate which object to grasp in an image. For implementation, we developed a simple user interface that allows the operator to select one or more point(s) on the object of interest in the image. The image is then processed by the image encoder, and the selected point coordinates are provided to the prompt encoder. While SAM provides valid high-quality masks, it may generate a mask on a distinct part of the object instead of the whole object. This ambiguity (e.g., between bottle cap versus bottle) can be resolved by selecting multiple points on the object for the prompt. Once a predicted mask is obtained, we apply it to the aligned depth image from the depth stream of the RGB-D camera and de-project it back to the camera frame using depth camera intrinsics.

At this stage, due to sensor noise and inaccuracies, the point cloud often exhibits point bleeding over object edges that can extend up to tens of centimeters. To address this, we employ DBSCAN [258] clustering method. DBSCAN tries to find areas in the data that satisfy a minimum point density. The minimum density threshold is specified by the user in terms of a radius (r_{nb}) around a point and the number of neighbourhood points (N^*_{nb}) in it. For each point, if there are more than N^*_{nb} points inside r_{nb}, the point is considered to lie within the object. The points that are inside the radius of one or more other points but do not have the minimum number of neighbors, they are considered border points. While those not inside any neighborhood regions are considered outliers. We use this on segmented point cloud to filter noisy outliers and obtain a clean object point cloud that can be directly fed into GraspLDM models.

4.3.3 Motion Planning

Motion planning in robotic manipulation refers to open-loop computation of joint-space trajectories from an initial joint configuration to a target end-effector pose under some set of constraints. These constraints can be in joint-space or Cartesian-space and can often be non-convex functions of joint angles. Motion planning for robotic arms on ground typically focuses on kinematic constraints. At reasonable payload mass and low-enough speeds, all kinematically feasible trajectories are also considered dynamically feasible for typical fully actuated arms with six and seven degrees of freedom. Broadly, the solutions to the motion planning problems under kinematic constraints can be classified into *optimization-based* or *sampling based* methods.

In *optimization-based* method, the problem is conveyed more naturally in the form of trajectory optimization subject to a relevant cost function with equality and inequality constraints. Most of these methods initialize a simple and potentially infeasible path (e.g. a straight line from start to goal state) that may violate constraints. The trajectory is then optimized to minimize the cost while satisfying the constraints. CHOMP [259] uses covariant gradient descent with signed distance field representation of the environment. STOMP [260] uses stochastic sampling of trajectories to deal with costs that may be non-differentiable. TrajOpt [261] uses sequential quadratic program with an efficient and convex formulation of collision checking. Due to non-convex nature of the problem, optimization-based solvers can get stuck in local minima and are not guaranteed to find a feasible solution even if one exists.

Sampling-based methods sample the configuration space and try to connect a path from initial to goal state through the sampled states. Given enough sampling attempts, these methods are guaranteed to find a solution if one exists. Probabilistic roadmaps [262] sample points in the free-space areas of robot's configuration space and connect them into a dense graph that can be used to find the shortest path. Rapidly exploring Random Trees (RRT) [263] uses incremental and directed sampling of points in the configuration space using evolving tree structures. RRTConnect [155] builds on RRT each at the start and goal states, with each tree growing towards using a greedy heuristic, connected when nearest

nodes of the two trees fall within a threshold distance. RRT* [264] extends RRT to guarantee asymptotic convergence to an optimal path but is slower than other methods. Sampling-based methods also employ post-processing for smoothing and shortening of paths generated by connecting all discrete samples in the extended tree branch. In general, sampling-based methods are slower to generate optimal trajectories with post-processing steps. On the other hand, the computational demand of optimization-based methods also depend on discretization resolution of the path. Newer works like GPMP2 [265] propose continuous Gaussian process parameterization of the trajectory and factor-graph based probabilistic inference to reduce computation time. Motion planning runtime is also being further optimized using parallelization on dedicated hardware for both types of methods [266, 267].

For robotic grasping, we primarily want to use motion planning to find a feasible path to one of the good grasp poses. A simple solution is to iterate over a ranked set of grasp poses and check feasibility for each end-effector pose in decreasing order. An alternative is to use an integrated grasp and motion planner that can reason about the set of grasp poses and the robot's kinematics simultaneously. In this way, the grasp pose set constrains the path planning and vice-versa. Assuming that a decoupled grasp sampler may provide grasp that is infeasible from the current robot state, joint grasp and motion planning can improve efficiency. Early works [268, 269, 270] integrated RRT-type methods with online grasp planning based on analytic metrics or heuristics. Subsequently, optimization-based methods like CHOMP were used along with the goal set constraints [271]. OMG-Planner [137] addressed the issue of goal switching during optimization, where costs for all possible goals are computed at each optimization iteration and the one with the lowest cost is selected. Instead of explicitly formulating the cost function, it may also be parameterized as a neural network and learnt from task data. Using neural networks to represent scalar fields implicitly allows gradient-based optimization. NGDF [272] learns a continuous distance function using a neural field, whose level-set is the set of successful grasp poses. The grasp distance field can be queried at current pose to find a scalar distance to the nearest successful grasp configuration, conditioned on signed distance field of the object. Minimizing this distance function at each step takes the robot to a successful grasp pose. GraspDiffusion [123] formulates joint

grasp and motion planning problem as reverse diffusion process using Langevin dynamics. They encode grasp quality or cost with a learned energy model conditioned on signed distance field of the object, and use the energy gradient to guide the robot to a successful grasp. While advancement of joint motion and grasp planning is promising, all of these methods rely on known object model or aggregated point clouds covering the entire object. The testing is also limited to a small number of objects and categories.

For single-view grasping in the real-world, adding collision constraints for partially observable objects is non-trivial. Consider a high-quality grasp that is available on a partially observable region of an object. Due to depth sensor noise, the points at the edges bleed over observable edges and the collision volume of segmented point cloud extends beyond the depth of the actual object. In this case, such an extended collision volume will lead to good grasp candidates being rejected because a grasp pose results in the finger colliding with the collision volume. The primary use case for our grasping system is to test and validate object-centric grasp samplers like GraspLDM, in which case joint grasp planning introduces another layer performance dependency. To avoid this, we use a simple strategy of dividing the motion to the grasp in two segments. The first segment takes the gripper from an initial pose to a pre-grasp pose. We assume that the direction perpendicular to the finger sweep is the safest approach direction for a parallel jaw gripper and compute the pre-grasp pose at a safe distance away from the target grasp pose along this axis. To plan this segment, we use an axis-aligned bounding box around the segmented point cloud as a collision volume, so it does not collide with the object while moving to the pre-grasp pose. The second segment is a straight line path from the pre-grasp pose to the grasp pose. For this segment, we remove the collision volume from the planning scene. This two-segment planning strategy is simple but effective, especially for validating 6-DoF grasp synthesis models in real-world scenarios. In terms of implementation, we design modular interface with initial support for MoveIt! [111] and Open Motion Planning Library [273], which avail a broad set of planners including RRT-based methods as well CHOMP and STOMP. Unless specified otherwise, we use RRTConnect for motion planning by default. Methods like OMG-Planner [137] and Goal-set CHOMP [271] can be readily integrated in the modular

system as they rely on a separate grasp planner.

4.4 Hardware and Software Architecture

The implemented system architecture decomposes robotic grasping into three primary subsystems: vision processing, grasp execution manager, and robot planning and control, each encapsulated within separate containerized environments. The message passing between these subsystems is implemented in ROS. Primarily, we use action servers to enable request-response type communication with asynchronous goal specification and execution status monitoring.

One of the key implementation challenges when combining vision, planning and control along with hardware elements is the difficulty of managing a broad range of dependencies and configurations in a single environment. Typically, systems with monolithic designs are light-weight but less amenable to supporting other hardware and algorithms in the future. We develop the system with the speed of integration and validation in research settings as a priority. Therefore, we take a modular approach by separating the two hardware-dependent components (vision and control) into individual container environments of their own with necessary device control drivers and libraries. In this way, when a previously unsupported sensors or manipulator needs to be used, their drivers and dependencies can be either be integrated in the same container environment or swapped with a new container with minimal changes to the wider system. Separating subsystems into individual containers enables distributed execution across multiple machines, addressing resource constraints and real-time control requirements. With modular architecture, we implement the system to support a broad range of components across multiple RGB-D sensors, robotic manipulators, motion planners, and grasp sampling methods as detailed in Table 4.2. The components marked as tested are used in the experiments in Section 4.5.

The vision subsystem consists of two primary components: a camera driver/ROS wrapper that interfaces with the hardware, and a camera interface that abstracts the camera-specific implementations. The camera interface either exposes a standardized API or a

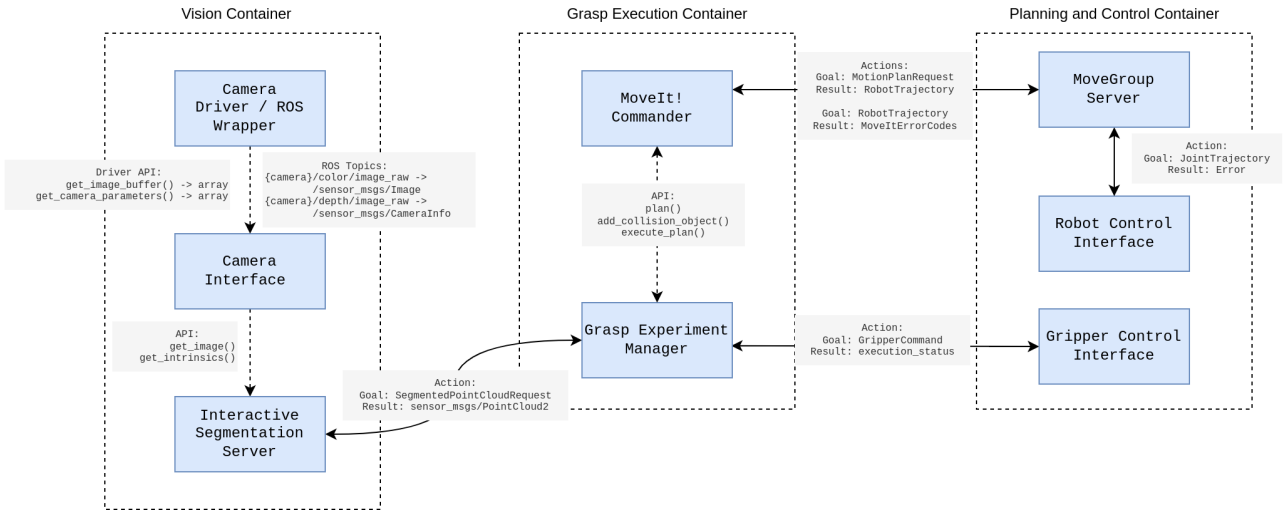


Figure 4.4: Software architecture for object-centric grasping system

set of ROS topics for image acquisition and camera parameter retrieval. This abstraction layer communicates with an interactive segmentation server through a ROS action interface. When the grasp experiment manager requests a segmented point cloud, the latest image is acquired from the camera interface and sent to the segmentation server. The segmentation server spawns a Graphical User Interface (GUI) with the RGB image that allows user to select arbitrary number points on the object of interest. The selected points are used as a prompt and a segmented point cloud is obtained using the aligned depth image as described in Section 4.3.2. The segmented point cloud is then sent back to the grasp experiment manager as response.

The grasp experiment manager is the central coordinator that orchestrates the flow of grasping experiments. It is responsible for interfacing with the vision subsystem to obtain segmented object point clouds and with planning and control subsystem to execute grasps. Internally, the grasp experiment manager spawns a grasp pipeline consisting of the grasp sampler and grasp classifier. The manager follows a sequential workflow. It begins by requesting a segmented point cloud from the vision server. This point cloud data is then processed by the grasp sampler. The resulting samples are ranked by the classifier. Finally, the manager queries the motion planner to determine a feasible plan. The manager then

Components	Tested	Supported
Cameras		
RealSense D435/D435i	✓	✓
RealSense Cameras with D400/D450 modules	-	✓
Robot Arms		
UR 10 / UR 10e	✓	✓
UR 3/5/16/20	-	✓
Franka Research 3	✓	✓
Franka Production 3 / Panda	-	✓
Grippers		
Robotiq 3-Finger Gripper	✓	✓
Franka Hand	✓	✓
Planners (via Moveit! + OMPL)		
RRTConnect [155]	✓	✓
RRT* [264]	-	✓
CHOMP [259]	-	✓
STOMP [260]	-	✓
TrajOpt [261]	-	✓
Instance Segmentors		
SAM [221]	✓	✓
UOIS [252]	-	✓
Grasp Samplers		
GraspLDM	✓	✓
GraspVAE	✓	✓
6-DoF-GraspNet [89]	✓	✓
Contact-GraspNet [90]	✓	✓

Table 4.2: Hardware and software compatibility matrix for object-centric grasping system

authorizes the motion planner to pass the planned trajectory to the robot control subsystem for execution. Once a grasp pose is reached, the grasp experiment manager commands the gripper to close and proceeds with the post-grasp sequence thereafter.

Motion planning capabilities are exposed through a unified interface to MoveIt! which avails planners from Open Motion Planning Library. This allows the grasping system to support a broad range of motion planners as shown in Table 4.2. Moveit! primarily uses ROS-Control interfaces to communicate with the hardware through robot-specific hardware abstract layer which exposes the features of the underlying drivers using ROS interfaces. In terms of hardware, the current implementation has been tested on a UR10e robot with

(a) (b)

Figure 4.5: Objects used for real-world grasping tests in (a) Setup 1 and (b) Setup 2.

Robotiq 3-finger gripper as well as a Franka Research 3 robot with Franka hand. In both cases, we use Intel Realsense D435 RGB-D camera over USB and robot hardware over ethernet. Due to general nature of interfaces, the system readily supports similar hardware configurations as shown in Table 4.2.

4.5 Real-world Experiments

Our primary goal is to evaluate the performance of GraspLDM in real-world 6-DoF grasping scenarios and complement the large-scale simulated evaluations conducted in chapter 3. In the process, we want to evaluate two aspects. First, we want to evaluate the performance transfer of GraspLDM trained purely from simulation data and in a fully supervised manner to the real world. Second, we want to evaluate whether the proposed grasping system is usable and generalizable to different real-world setups. Together, validation of these two aspects provide a good foundation for building grasping systems for the real world. We do the first by evaluating the success rate metric, as is common in the related works [90, 89, 95].

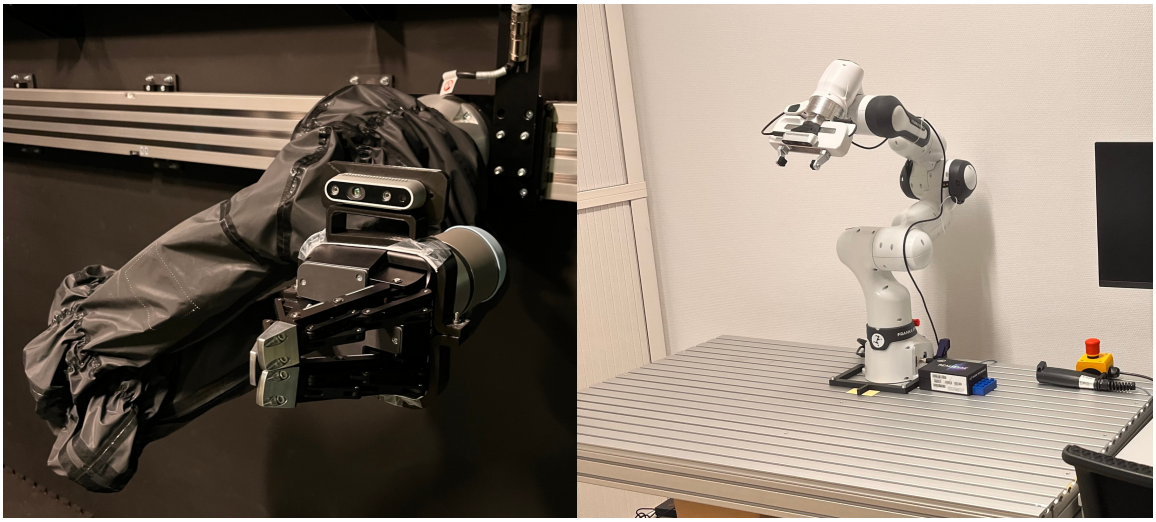
Test Protocol

We collect 16 random objects with diverse physical and geometric properties as shown in Fig. 4.5. We place a single object on a table in a random pose and the goal is to drop the object in a predefined bucket. The robot first acquires a single-view RGB-D image from a predefined relative pose to the table. This pre-defined pose is arbitrarily set 80cm away from the center of the table, such that it looks at the center of the table from an inclination between 30-45 deg. Each grasp sampler is then run on the image to generate 100 grasp poses. Ranked grasps above a success probability threshold are then sorted in decreasing order of success probability. The grasp experiment manager sequentially goes through the sorted grasp and checks feasibility using the motion planner, until a feasible grasp is found. We execute the first feasible grasp without user intervention. The robot first executes a Cartesian line path to the pre-grasp pose and then from pre-grasp to the grasp pose. At grasp pose, the grasp execution completes when the gripper fingers are closed. To determine whether a grasp is successful, the robot lifts the object 20 cm vertically and then goes to a pre-defined drop pose [1]. A grasp is successful if the object remains secure in the gripper until the pre-defined drop. To compute a meaningful success rate, we conduct five trials per object in random relative poses, to test the diversity of 6-DoF grasp pose generation in addition to the accuracy. Across methods, we approximately reproduce the object poses in five trials to ensure fairer comparison. As the motion from initial pose to pre-grasp pose is constrained to a linear Cartesian motion, the success rate relies also on diversity of grasps generation. This is representative of real-world tasks, where grasping must be done under arbitrary workspace constraints.

Baselines

For tests, we use the GraspLDM-P-63C and GraspVAE-P-63C models introduced in Section 3.5.4 without any further fine-tuning. We compare our performance against two established baselines widely used in the community. First, as a direct comparison with a gener-

[1] In Test setup 2, 30 deg shaking was added along X and Y axes after lift and before going to drop pose.



(a) (b)

Figure 4.6: Real-world experiment setups. (a) **Setup 1:** UR10e arm with Robotiq 3F operated in pinch mode and a Realsense D435 on the wrist. (b) **Setup 2:** Franka Research 3 arm with parallel jaw Franka hand and a Realsense D435 mounted on the wrist.

ative model, we use 6-DoF-GraspNet [89]. Second, we compare the effectiveness of our models against the state-of-the-art Contact-GraspNet [90], which is a feed-forward model built on PointNet++. For a fair evaluation of grasp synthesis models, we only use the generator and classifier from 6-DoF-GraspNet and do not include the iterative refinement stage. Similarly, we do not perform iterative refinement on top of Contact-GraspNet predictions. All other parts of the pipeline like segmentation remain the same in every case. We excluded SE3-DiF diffusion models [123] from real-world tests because the original work utilized full object point clouds for both training and testing. A fair and reliable comparison with these models was not possible as we observed significant performance degradation when training SE3-DiF models on single-view point clouds from 63 categories with the default hyperparameters.

4.5.1 Test Setups

We assess the sim-to-real transfer on two hardware setups. The first setup uses a wall-mounted 6-DoF UR-10e arm, a Robotiq-3F gripper, and an Intel Realsense D435 RGB-D

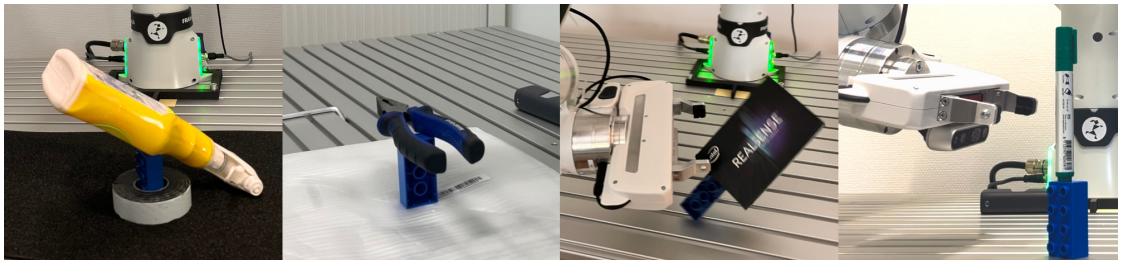


Figure 4.7: Examples of object placement on support during real-world testing. Such object placements allow grasps from more approach directions to be feasible while also demanding higher grasp pose accuracy for success.

camera. The second setup uses a table-mounted 7-DoF Franka Research 3 arm with Franka hand. In the former case, the 3-finger gripper is operated in a two-finger ”pinch” mode with the gripper sweep length restricted to 85mm to emulate the gripper used for generating the training data. In both cases, the camera is mounted at the tool flange of the arm in an eye-in-hand configuration. Due to limitations of access to hardware, we conducted the comparison with 6-DoF GraspNet on the first setup, while the comparison with Contact-GraspNet was conducted later on the second setup. In each case, we use a test set of 16 randomly selected objects of diverse physical and geometric properties as shown in Fig. 4.5. Each grasping trial is conducted by placing a single object in a random pose on a table. To ensure that the trials allow execution of grasps from more approach directions than just top-down, we lift the objects on a small support as shown in Fig. 4.7. This also makes grasping success more difficult compared to table-top placement where in-plane sliding and normal contact force supports imprecise grasp attempts. At the start of the trial, the robot observes the object from an arbitrarily fixed pose such that the camera boresight is between 30° and 60° to the table plane and the object is between 0.4m and 0.8m from the camera. We conduct five random pose trials per object to evaluate the success rate across a total of 80 grasp attempts without retries.

Table 4.3: Real-world 6-DoF grasping success rate comparison on 16 evaluation objects in five random poses

Setup	Method	Success Rate
Setup 1: UR10e + Robotiq-3F	**GraspLDM+GraspClassifier** *(ours)*	$\mathbf{80\%}$
	GraspVAE+GraspClassifier *(ours)*	76.25%
	6-DoF-GraspNet+Classifier [89]	37.5%
Setup 2: Franka FR3 + Franka hand	**GraspLDM+GraspClassifier** *(ours)*	$\mathbf{78.75\%}$
	GraspVAE+GraspClassifier *(ours)*	67.5%
	Contact-GraspNet [90]	76.25%

Results

Real-world performance comparison of GraspLDM-P-63C and GraspVAE-P-63C with the baselines is presented in Table 4.3. Additionally, Fig. 4.8 visualizes intermediate data from segmentation and grasp generation and Fig. 4.9 shows pictures of real-world grasp executions for a few objects. Detailed per object log of attempts is provided in Appendix A. The results demonstrate that GraspLDM provides a superior grasp success rate in the real world while being entirely trained with simulation data. Compared to the generative model baseline [89], GraspLDM demonstrates a significantly higher success rate. On the other hand, GraspLDM shows a comparable success rate with that of Contact-GraspNet [90], which is a feed-forward predictive model. Despite the success rate being marginally higher for GraspLDM, the difference concerns 2/80 grasp attempts. Due to several factors influencing real-world tests of 6-DoF grasps, we consider the two models to have comparable success rates. We also observe that the GraspLDM-P-63C model provides around 4.75% and 11.25% higher success rate over its base VAE model in the two test setups while using the same classifier. We believe that the relative variation in the performance of GraspVAE between the two test setups is mainly related to the non-trivial effects of viewpoint and noise on generation. We observe that for favorable viewpoints both GraspVAE and GraspLDM have similar output, while in some other cases, GraspLDM performs noticeably better than GraspVAE. This effect is more pronounced in the second setup than in the first.

We notice that high amounts of failure for the 6-DoF-GraspNet models result from the

Table 4.4: Inference latency for segmentation, grasp generation and classification models in the pipeline

Model	**Batch size**	**Inference Time (s)**	**Peak Memory Usage (MB)**
Segment-Anything (ViT-B)	1	0.35 ± 0.016	2789
GraspVAE-P-63C	100	0.02 ± 0.006	156
GraspLDM-P-63C (DDIM)	100	0.75 ± 0.019	160
GraspLDM-P-63C (DDPM)	100	7.39 ± 0.060	160
GraspClassifier	100	0.02 ± 0.002	1084

grasps colliding with the object and a lack of grasp generation diversity. In the latter case, it does not produce enough grasps in kinematically feasible regions of the object in a given pose. We also observe a higher rate of planning failures for this baseline until a successful grasp is attempted, which is connected to grasp diversity. On the other hand, ContactGraspNet outperforms and produces superior grasps for objects closer to the categories represented in the ACRONYM dataset. This is especially the case for objects with trivial edges/surfaces like boxes. While for the objects with curvature and those with shapes unlike ACRONYM categories, it underperforms compared to GraspLDM. Overall, we demonstrate that GraspLDM models outperform existing generative models and provide comparable performance against larger feed-forward models.

4.6 Discussion

In this chapter we demonstrated how GraspLDM can be integrated into a system for real-world grasping. We showed that both the 6-DoF grasp synthesis models and the system are generalizable across different hardware setups. In terms of computational efficiency, the latency and memory usage of the pipeline models are shown in Table 4.4. The GraspVAE-P-63C and GraspLDM-P-63C models both have high memory efficiency. Although, GraspLDM-P-63C models have high generation latency due to sequential reverse diffusion process. Using DDIM sampler alleviates this issue to some extent, but further

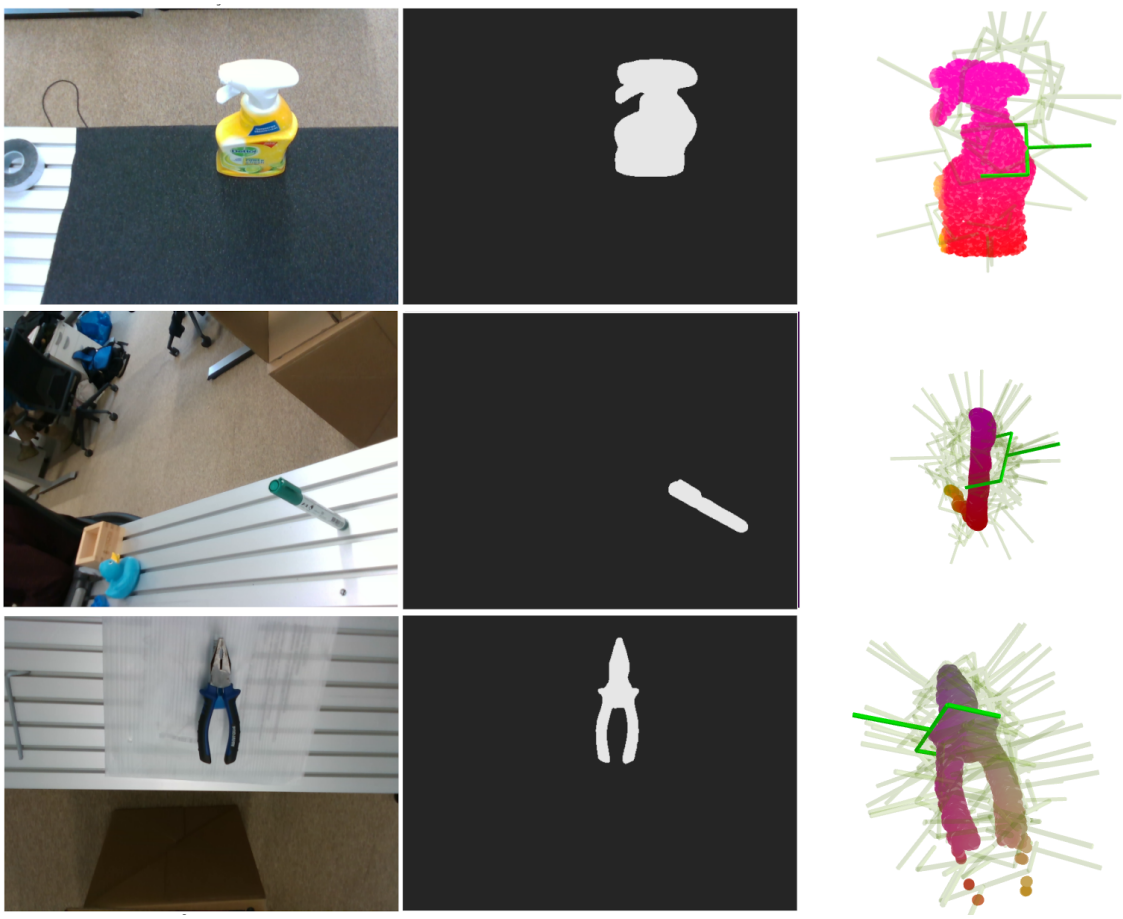


Figure 4.8: Examples of images, segmentation masks, grasp generation and selection in real world tests.

improvements are needed to make diffusion sampling more efficient. GraspClassifier has around five times more peak memory usage compared to the samplers. This is because unlike GraspLDM where the point cloud is encoded only once for any number of grasps, the classifier takes composite point cloud per grasp. As a result, classifying 100 grasps requires 100 distinct forward passes through the network. The segmentation model has a very high memory usage and moderate inference latency. This is expected as it uses a high-capacity vision transformer model. We do not consider this restrictive, as concurrent efforts in the community have already demonstrated approximately 3x speed up and 3x memory usage

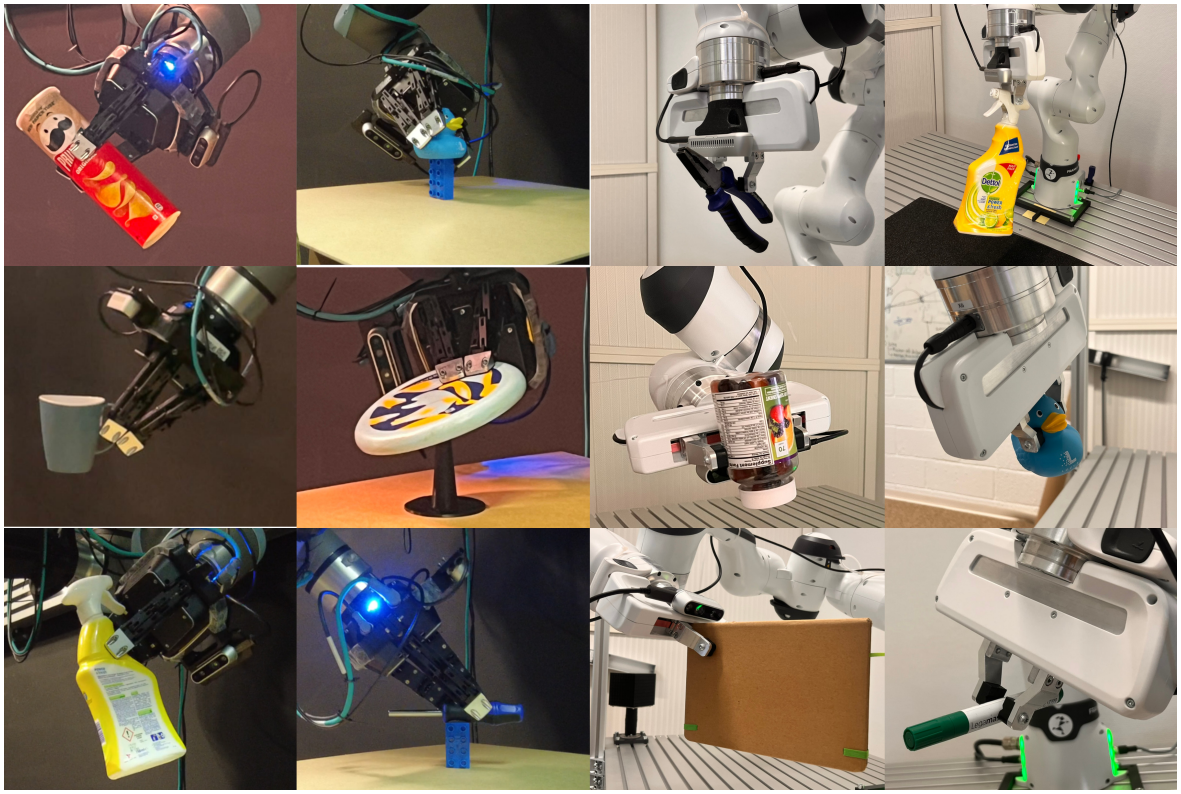


Figure 4.9: Real-world grasp execution examples on the two setups.

reduction for the same model [2]. Overall, the complete stack is currently deployable on a single machine with a GPU memory of 4GB or more. With further optimizations using quantization and kernel compilation, our system has the potential to run on mobile devices with limited resources.

We also added GraspClassifier as a separate module in the pipeline. This is necessitated by GraspLDM models not being able to provide a notion of grasp quality. While metrics are presented for the GraspClassifier model, analyses on its performance are not covered in this chapter. We observe that GraspClassifier assigns high confidence to colliding grasp poses in some cases, despite being trained on negative examples. A detailed analysis of the GraspClassifier model is required to improved grasp quality estimation and reduce overconfidence in the real world. However, what we really desire is to learn some notion of grasp

[2]Segment-Anything-Fast: `https://pytorch.org/blog/accelerating-generative-ai/`

quality bootstrapped with the GraspLDM architecture in the future. This will help alleviate the memory consumption issues highlighted above as well as task-conditioned grasping in the real world.

It is important to note that our models are competitive in the real world against state-of-the-art generative and regression-based models, despite being trained on only 63 out of 180 categories in the ACRONYM dataset due to compute constraints. We believe that the performance can be improved further by incorporating the entire dataset in the training. Motion planning for each possible grasp pose sequentially is a crucial limitation of the current system. This introduces larger latency per object pick especially when motion planning is highly constrained and top-k good grasps with highest confidence are not feasible. Advances in parallel motion planning can be integrated to mitigate this issue. Finally, the tests are restricted to performance validation of object-centric grasp samplers like GraspLDM. The system is not tested in cluttered scenarios or for task-specific grasping, which can be addressed in the future work.

4.7 Summary

In this chapter, we addressed one of the fundamental challenges in robotic manipulation: the reliable transfer of grasp synthesis from simulation to real-world scenarios. Through extensive real-world testing across two distinct robotic setups, we demonstrated that our system with GraspLDM successfully bridges this gap, achieving approximately 80% grasp success rate on unknown objects despite being trained purely on simulation data. This performance, comparable to state-of-the-art feed-forward models and significantly better than existing generative approaches, validates both the robustness of GraspLDM's learned grasp distributions and the effectiveness of our modular system architecture. The system's successful deployment on different hardware configurations - from a wall-mounted UR-10e to a table-mounted Franka FR3 - demonstrates its adaptability to varied real-world settings. We validated GraspLDM's ability to maintain consistent performance across diverse object geometries and surface properties suggests. This shows that the model has learned ro-

bust, generalizable representations from the simulation data used to train it. The superior grasp diversity and reduced planning failures compared to baseline methods indicate that our approach effectively captures the full range of viable grasp configurations, a critical requirement for practical robotic manipulation tasks. We make our implementation available open-source for the community to build novel solutions to the robotic grasping problem using object-centric grasp synthesis.

Contributions

1. **We introduce a modular system for object-centric 6-DoF grasping in the real world based on generative grasp synthesis.** The system complements GraspLDM and other generative models for grasp synthesis by providing a complete pipeline for grasping in real-world scenarios from a single RGB-D view of an object.

2. **We validate the modularity of our system on two real-world robotic setups.** We show that using the proposed system is generalizable and can be effectively transferred to different robotic setups.

3. **We validate the effectiveness of GraspLDM models in real-world grasping scenarios.** We show that GraspLDM models provide around 80% success rate across two different hardware setups in a test of 80 grasp attempts on 16 unknown objects in varying poses.

4. **We make the models and code available open-source for the community to use and extend.** The resources are available at `https://github.com/kuldeepbrd1/object-centric-grasping`.

Chapter 5

6-DoF Tracking and Reconstruction of Unknown Objects

5.1 Introduction

Previous chapters address the problem of 6-DoF grasp generation and execution for unknown objects. During validation of the solutions, the object was assumed to be static in the world for the entire duration from image acquisition to grasp execution. However, this assumption limits many real-world use cases in which the static nature of the object cannot be guaranteed. In space applications, this naturally arises in orbital manipulation scenarios like servicing, assembly and debris capture, when the target object may not be rigidly attached to the manipulating platform. More generally, applications involving robot-to-robot and robot-to-human object handovers, especially in the context of long-horizon task-relevant grasping requires the robot to reason about a dynamic target object. In all these use cases, a truly general system may not assume any prior knowledge of the object's structure or motion. This is the primary motivation for the work described in this chapter. The secondary motivation is to use the additional views of the object revealed to the sensor due to relative motion. In Chapter 3, we established the lower bound and upper bound of performance of GraspLDM models in single-view and full point clouds respectively. As more

coverage of the object is available, we can improve grasp generation and execution stepping towards upper bound performance with full point cloud.

In Section 2.4, we reviewed the two methods of tackling dynamic object grasping and outlined the benefits of object-level 3D reconstruction and pose tracking. We also highlighted emerging research in radiance fields and its potential for this problem. To this extent, this chapter addresses the problem of incremental 3D reconstruction and 6-DoF relative pose tracking of an unknown object in unconstrained motion. Our approach assumes no prior knowledge about the object's structure or motion. The goal is to realize a perception system that may facilitate unstructured dynamic grasping. The scope of this work is limited to the analysis of 3D reconstruction and tracking aspects of dynamic grasping. Challenges in 6-DoF dynamic grasping go beyond perception and need treatment of reachability, collisions and changing grasp pose targets in planning and control. These aspects are out of the scope of this work. We focus on RGB-D data assuming stereo or or another depth sensor provides depth measurement to disambiguate the scale of the scenario under observation. The principles and the architecture however is generally applicable RGB scenarios albeit with scale ambiguity and scale drift issues.

The chapter is organized as follows. Section 5.2 introduces the problem of structure and motion recovery from visual observations from a modern lens of radiance fields and analysis by synthesis. Section 5.3 introduces the object-centric representation of 3D Gaussians and the differentiable rendering process that models image formation. Utilizing this representation, Section 5.4 describes the pipeline for incremental 3D reconstruction and tracking. Section 5.5 presents the experiment setup for preliminary validation of the pipeline. Finally, Section 5.6 concludes the chapter with a discussion on the results, the limitations and future directions of the work.

5.2 Structure and Motion from Visual Observations

Extraction of object-level structure and relative motion from visual observations concerns two fundamental problems in computer vision- 3D reconstruction and 6-DoF pose track-

ing. The two problems are generally dealt with separately, while the solutions often use different internal representations and assumptions. For instance, the state-of-the-art methods for scene reconstruction in terrestrial applications use Neural Radiance Field (NeRF) representations [166]. NeRF-based methods usually assume a static scene and the availability of relative camera poses. On the other hand, 6-DoF pose estimation and tracking for unknown objects is addressed with large pre-trained neural networks. Many state-of-the-art approaches require a 3D model of the object as a template [274]. Furthermore, a downstream robotic task like grasping may use another neural network that takes RGB-D images as inputs to infer grasps. All these elements require separate datasets, isolated pre-training(s), and parameter tuning.

Grasping unknown objects in dynamic environments requires the robot to reason about the object's motion and structure in real-time. Ideally, we want to accomplish this with a representations that has commonality with other tasks like grasp synthesis, which in our case is done using point clouds. Generalizable perception systems with unified representations are crucial for both terrestrial and space-borne robotic systems that operate autonomously in dynamic and unstructured environments. Therefore, we are interested in structure and motion recovery methods that offer: (1) efficient object-level representations, (2) generalization to an open set of objects, and (3) seamless integration to 6-DoF grasp synthesis models.

5.2.1 Radiance Fields

The choice of representation of 3D scenes remains a fundamental consideration in 3D computer vision and robotics. Traditional approaches rely on discrete representations such as point clouds, voxel grids, and meshes, each with inherent limitations in resolution, memory efficiency, or reconstruction quality. Recently, the research has moved towards continuous field representations of scenes which offer unprecedented fidelity and the flexibility of joint optimization using analysis-by-synthesis [275, 276].

A field describes a scalar quantity defined over a spatial domain. Scalar fields $f : \mathbb{R}^n \to \mathbb{R}$ map spatial coordinates to scalar values, while vector fields $F : \mathbb{R}^n \to \mathbb{R}^m$ associate vectors with each spatial location. Neural fields are fields where the continuous mapping from

coordinates to scene properties is parameterized as a deep neural networks. By leveraging neural networks' ability to approximate arbitrary functions, neural fields enable rich representations of geometry, appearance, and other scene attributes. Building on this foundation, NeRFs have transformed 3D scene representation and novel view synthesis by encoding both geometry and appearance inside a continuous volumetric field. At its core, NeRFs employ a multilayer perceptron with parameters Θ that maps 3D position $\mathbf{x} = (x, y, z)$ and viewing direction vector $\mathbf{d}$ to density σ and color c:

$$(\sigma, c) \leftarrow F_\Theta(\gamma(\mathbf{x}), \gamma(\mathbf{d})) \tag{5.1}$$

where $\gamma(\cdot)$ denotes positional encoding that takes the spectral bias of neural networks in low-dimensional spaces [277]. Scene reconstruction occurs through differentiable volume rendering. For each pixel origin $\mathbf{o}$, a camera ray $r(t) = \mathbf{o} + t\mathbf{d}$ is cast, sampling points to compute the rendered color $\hat{C}(r)$:

$$\hat{C}(r) = \sum_{k=1}^{K} T_k(1 - \exp(-\sigma_k(t_{k+1} - t_k)))c_k \tag{5.2}$$

where $T_k = \exp(-\sum_{k'<k} \sigma_{k'}(t_{k'+1} - t_{k'}))$ represents accumulated transmittance. This differentiable formulation enables end-to-end optimization of scene representation from 2D observations. NeRFs demonstrate impressive reconstruction quality and have seen rapid adoption in robotics from navigation [167] to manipulation [278]. Neural fields can also be augmented with multi-modal and semantic information for holistic scene representation and rendering.

Representing scenes implicitly with NeRF has the disadvantage that it requires expensive sampling of space to render an image from the representation. Inevitably, training and rendering tends to be slow for applications like robotics. Recently, Kerbl et al. [170] introduced 3D Gaussian Splatting (3DGS) for efficient high-quality rendering of 3D scenes. 3DGS is volumetric radiance field in that it is continuous like NeRFs while using explict 3D Gaussian primitives to parameterize the field. It moves away from neural representations and facilitates high fidelity reconstruction with faster training and rendering times. Importantly,

the centers of these 3D Gaussians form a point cloud, which means that object reconstruction obtained by 3DGS could be fed directly to GraspLDM. This motivates the use of 3DGS as a scene or object representation for our problem here. The 3DGS formulation is covered in detail in Section 5.3.

5.2.2 Analysis by Synthesis

Localization of camera with respect to a scene, or equivalently an object with respect to camera in object-centric coordinates, is typically done through correspondence matching between 2D image features and 3D scene geometry or through direct pose regression from images. However, differentiable scene representation like NeRF and 3DGS now facilitate an elegant alternative through analysis-by-synthesis. Analysis-by-synthesis in radiance fields involves inverting the image formation process by iteratively refining pose parameters to align rendered and observed images [276]. Radiance Fields make this approach particularly powerful by providing a differentiable scene representation that maps 3D coordinates $\mathbf{x}$ and viewing directions $\mathbf{d}$ to density and color values through a neural network F_Θ. Given an observed image I and initial pose estimate T_0, the framework renders synthetic views through volume rendering and minimizes the discrepancy between rendered and observed images:

$$\hat{T} = \arg\min_{T \in SE(3)} \mathcal{L}(I, R(T|\Theta)) \tag{5.3}$$

where $R(T|\Theta)$ represents the rendering operation. This optimization can be performed efficiently through gradient descent since the entire pipeline from pose parameters to rendered pixels is differentiable:

$$T_{i+1} = T_i - \alpha \nabla_T \mathcal{L}(I, R(T_i|\Theta)) \tag{5.4}$$

This enables end-to-end pose refinement without requiring explicit 3D models or predefined feature correspondences. For tracking purposes, analysis-by-synthesis offers a flexible framework to obtain frame-to-model association of an acquired image to an evolving

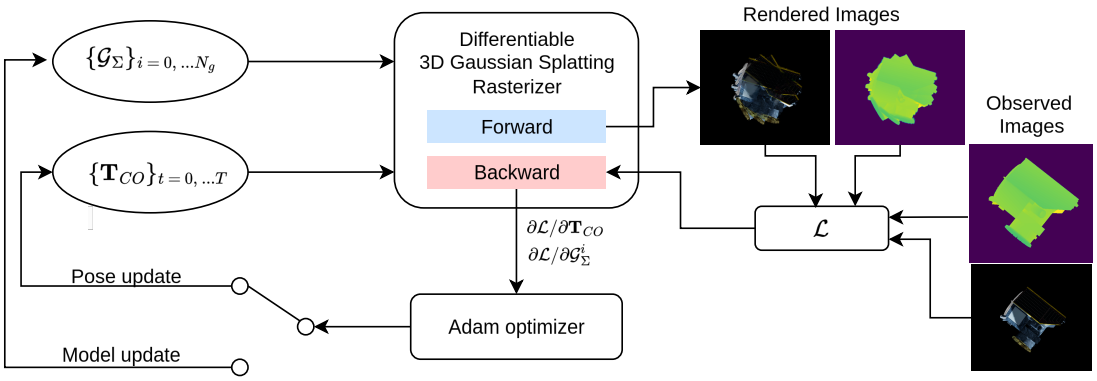


Figure 5.1: A simple incremental reconstruction and tracking methodology using differentiable rendering of 3D Gaussians. The camera pose or the object Gaussians in object frame are refined directly using first-order gradient-based optimization by propagating the gradients backward through the rendering process.

reconstruction. Along with novel-view synthesis capabilities of radiance fields, this approach is particularly well-suited for incremental tracking and reconstruction of dynamic objects.

5.3 Object-centric 3D Gaussian Splatting

5.3.1 Object Representation

We represent the appearance and geometry of the target object as a set of $N \in \mathbb{N}^+$ 3D Gaussians ellipsoids as primitives defined in the object frame, which is rigidly attached to the object. A d-dimensional Gaussian ($G_{\boldsymbol{\mu},\boldsymbol{\Sigma}}$) is parameterized in terms of a center ($\boldsymbol{\mu}^i \in \mathbb{R}^d$) and a covariance ($\boldsymbol{\Sigma}^i \in \mathbb{S}^d_+$) and evaluated as:

$$G^d_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\mathbf{x}) = \frac{1}{\sqrt{|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \tag{5.5}$$

For each 3D Gaussian G^i ($i = 0, 1, ... N$), the center (μ^i) and the covariance ($\boldsymbol{\Sigma}^\mathbf{i}$) dictate its geometry. In addition, we associate an opacity (o^i) and a color ($\mathbf{c}^i$) parameters that dictate its appearance. Intuitively, a 3D Gaussian can be understood as a translucent and colored ellipsoidal blob. The transparency of this blob decays continuously per Eq. 5.5 as a point of interest moves away from the center. As we will see later, the color parameter can be

view-dependent.

In the context of on-the-fly 3D reconstruction from images, four key attributes are essential for the underlying representation: **(1) Adaptiveness:** The ability to adjust spatial resolution based on relative camera-object motion; **(2) Efficiency:** Optimal memory usage for storage and processing speed in image formation; **(3) Rendering fidelity:** The capacity to produce artifact-free images with high SNR of traversed and novel views ; and **(4) Differentiability:** Enabling convenient optimization of object/scene parameters based on image observations. 3D Gaussians offer a favorable combination of these properties. Similar representations include point clouds and surfels [279], which explicitly represent irregular 3D point sets. These representations are adaptive and allow the addition, removal, and modification of geometry dynamically at runtime. This is advantageous in 3D reconstruction scenarios where the relative motion profile is unknown a priori. However, 3D Gaussians fill a volume continuously as opposed to point clouds that are discontinuous. Surfels, while similar, require normal vector information, which is challenging to compute accurately in real-world scenarios with noisy observations. Alternative structured grid representations such as voxels are constrained by fixed resolution, high memory requirements, and computationally expensive sampling. Meshes, while efficient for rendering, are not easily obtainable from noisy sensor observations. Furthermore, discrete topology changes (alterations in the connectivity of vertices, edges, and faces) are not differentiable.

5.3.2 Differentiable rendering of 3D Gaussians

Rendering concerns the process of evaluating the pixel intensity or color value, given camera and scene parameters. This can be written in a simple form:

$$\mathbf{I} = \mathcal{R}(\boldsymbol{\theta}) \tag{5.6}$$

where, $\mathcal{R} : \mathcal{S} \rightarrow \mathcal{V}$ maps scene parameters $\boldsymbol{\theta} \in \mathcal{S}$ to pixel color value $c_p \in \mathcal{V}$. In above expression, $\boldsymbol{\theta}$ encapsulates various scene parameters including structure, material, texture, illumination as well as camera intrinsics and extrinsics. The problem of forward

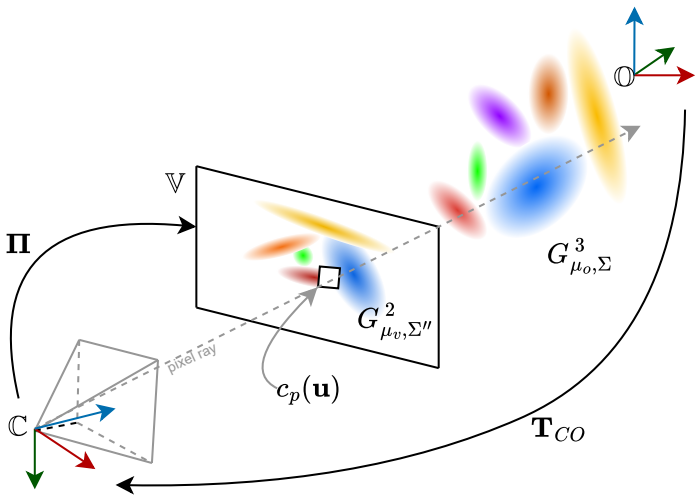


Figure 5.2: Illustration of projecting 3D Gaussians $G^{2}_{\boldsymbol{\mu_o}\,\boldsymbol{\Sigma_o}}$ to 2D Gaussian splats $\mathcal{G}_{\boldsymbol{\mu_v}\,\boldsymbol{\Sigma_v}}$.

mapping from scene parameters to image pixels is most commonly dealt with in computer graphics problems. In computer vision, the inverse problem is more common with emphasis on geometry recovery and pose estimation from image(s). If the mapping $\mathcal{R}$ is differentiable, the two processes can be connected by optimization. This enables any change in an image to be propagated back to scene parameters. In other words, scene parameters can be fit to a sequence of images as:

$$\theta \; = \arg\min_{\theta} \sum_{j} \mathcal{L}(I_j) \tag{5.7}$$

Forward Map

Splatting [280] is the forward process of projecting 3D volumetric primitives on the 2D image plane. Closely following [281], the splatting process for 3D Gaussian ellipsoids is summarized here. The process is illustrated in Fig. 5.2.

Consider the object space ($\mathbb{O}$), the camera space ($\mathbb{C}$) and the image space ($\mathbb{V}$). A point defined in the object space is viewed in camera space using a view transformation $\mathbf{T}_{CO} \in SE(3)$, with rotation component $\mathbf{R}_{CO} \in SO(3)$ and translation component $\mathbf{t}_{CO} \in \mathbb{R}^3$.

Although the focus is kept on rigid transformations the process is generally valid for any affine transformation. The Gaussian viewed from the camera frame can is obtained by substituting $\mathbf{x}_o = \mathbf{T}_{CO}^{-1}\mathbf{x}_c$ in Eq. 5.5 which results in another Gaussian with mean μ_c and covariance Σ_c given as.

$$G_{\boldsymbol{\mu_o}\boldsymbol{\Sigma_o}}(\mathbf{T}_{CO}^{-1}\mathbf{x}_c) \;=\; \frac{1}{|\mathbf{R}_{CO}^{T}|} G_{\boldsymbol{\mu_c},\boldsymbol{\Sigma_c}}(\mathbf{x}_c) \tag{5.8}$$

$$\boldsymbol{\mu_c} \;=\; \mathbf{T}_{CO}\,\boldsymbol{\mu_o}\;;\; \boldsymbol{\Sigma_c} \;=\; \mathbf{R}_{CO}\,\boldsymbol{\Sigma_o}\,\mathbf{R}_{CO}^{T} \tag{5.9}$$

For a perspective camera, a point in the camera space ($\mathbf{x}_c \in \mathbb{C}$) is transformed to the image space ($\mathbf{u} \in \mathbb{V}$) using the perspective projection ($\mathbf{\Pi}$), which is not affine. Zwicker et al. [281] linearize the transformation around a camera point ($\mathbf{x}_c^*$) using a first-order Taylor expansion as in Eq. 5.10, where $\mathbf{J}_{\Pi}^*$ is the jacobian of the projective transform, evaluated at $\mathbf{x}_c^*$ and $f_u,\, f_v$ are the focal lengths.

$$\mathbf{\Pi}^*(\mathbf{x}_c) = \mathbf{\Pi}(\mathbf{x}_c^*) \;+\; \mathbf{J}_{\Pi}^*\,(\mathbf{x}_c - \mathbf{x}_c^*) \tag{5.10}$$

$$\mathbf{J}_{\Pi}^* = \begin{bmatrix} f_u/z_c^* & 0 & -f_u \cdot x_c^*/z_c^{*2} \\ 0 & f_v/z_c^* & -f_v \cdot y_c^*/z_c^{*2} \end{bmatrix}$$

Using this approximation the image space projection of 3D Gaussian is a 2D Gaussian given as:

$$G_{\boldsymbol{\mu_c},\boldsymbol{\Sigma_c}}\,(\mathbf{\Pi}^{-1}(\mathbf{u})) \;=\; \frac{1}{|\mathbf{J}_{\Pi}^{*\,-1}|}\, G_{\boldsymbol{\mu_v},\boldsymbol{\Sigma_v}}(\mathbf{u}) \tag{5.11}$$

$$\boldsymbol{\mu_v} = \mathbf{\Pi}\mathbf{x}_c\;;\; \boldsymbol{\Sigma_v} \;=\; \mathbf{J}_{\Pi}^*\,\boldsymbol{\Sigma_c}\,\mathbf{J}_{\Pi}^{*\,T} = \mathbf{J}_{\Pi}^*\,\mathbf{R}_{CO}\,\boldsymbol{\Sigma_o}\,\mathbf{R}_{CO}^{T}\,\mathbf{J}_{\Pi}^{*\,T}$$

where, $\mathbf{J}_{\Pi}^*$ is the jacobian of the projective transform linearized at camera space coordinates $\mathbf{x}_c^*$ and $\mathbf{u}$ are the image space coordinates.

The way each 2D projection- a splat, contributes to a pixel in the image is dependent on the order in which they appear along a ray cast from that pixel. For each pixel, the $K \in \mathbb{N}^+$ overlapping Gaussians are first sorted from front to back. The sorting is done by

first grouping the 2D Gaussians into 16x16 tiles, by checking if the axis-aligned bounding box around its 3σ bounds. Then, within each tile the Gaussians are sorted by depth. Once sorted, the intensity or color ($\mathbf{c}_u$) of each 2D pixel location ($\mathbf{u} \in \mathbb{R}^2$) in the image is obtained by alpha-blending the contributions of the K Gaussians as

$$\mathbf{c}_p(\mathbf{u}) \;=\; \sum_{k=0}^{k \leq K} \mathbf{c}^k \alpha^k \prod_{j=0}^{k-1} (1 \;-\; \alpha^j) \tag{5.12}$$

$$\alpha^k = o^k G^k_{\mathbf{\Sigma}_v}(\mathbf{u})$$

where, α^k is the contribution of the k^{th} Gaussian to the pixel intensity. This contribution is computed by decaying the assigned opacity (o^k) by the 2D Gaussian ($\mathcal{G}_{\mathbf{\Sigma}_v}$). Eq. 5.12 follows the approximations to the volume rendering equations as outlined in [281].

As the 3D location of each Gaussian is known, we can similarly render a depth image using per-point z distance in the camera space:

$$d_p(\mathbf{u}) \;=\; \sum_{k=0}^{k \leq K} z^k \alpha^k \prod_{j=0}^{k-1} (1 \;-\; \alpha^j) \tag{5.13}$$

It is important to draw a parallel with NeRFs [166], which also uses differentiable volume rendering for scene reconstruction. Their forward map integrates the contribution of samples with density (σ^k), color (c^k) and transmittance (T^k) at δ^k intervals as:

$$c(\mathbf{u}) = \sum_{k=1}^{N} T^k (1 - \exp(-\sigma^k \delta^k)) \mathbf{c}^k, \quad \text{where} \quad T^k = \exp\left(- \sum_{j=1}^{i-1} \sigma^j \delta^j \right) \tag{5.14}$$

Eqs. 5.12 and 5.14 are equivalent and can be written as:

$$c(\mathbf{u}) = \sum_{k=1}^{N} T^k \, \alpha^k \, c^k \tag{5.15}$$

where $\alpha_k = (1 - \exp(-\sigma^k \delta^k))$ for NeRFs and $T^k = \prod_{j=0}^{k-1} (1 \;-\; \alpha^j)$ for 3D Gaussians.

A crucial difference is that NeRFs implicitly represent free space inside a volume, re-

quiring an expensive sampling process along a ray. On the other hand, Gaussian locations are explicitly known and the forward rendering process for Gaussians automatically ignores free-space regions in 3D along the viewing direction. As a result, rendering 3D Gaussians is considerably faster than a NeRF.

Backward Gradient Propagation

The forward map is entirely differentiable and for speed, it is explicitly implemented. Assuming that the intrinsics of the perspective cameras are known, the optimization parameters are the centers ($\boldsymbol{\mu}^i$), variances ($\boldsymbol{\Sigma}^i$), colors ($\mathbf{c}^i$), opacities (o^i) of the 3D Gaussians and the camera pose ($\mathbf{T}_{CO}$). Since the forward map is differentiable, the gradients of the photometric loss with respect to these parameters can be computed by backpropagation automatically. However, this process can be expensive for a large number of Gaussians. As it is straightforward to compute analytical expressions for gradients from the forward map, they can be implemented explicitly to reduce latency of auto-differentiation. We use the CUDA implementation from `gsplat` library [282] [1].

5.3.3 Initialization, Addition, and Removal of 3D Gaussians

We use the first depth image to initialize the 3D Gaussians (G_Σ) in the object space. The centers ($\boldsymbol{\mu}^i$) are centered around the point cloud obtained by de-projecting the depth image and the variances are initialized with 0.001. The color is initialized from the corresponding color image pixel value and the opacity is set to 0.5. For the camera-relative pose, if the ground-truth pose is available, it is provided only in the first frame to set the object reference frame. Otherwise, it can be arbitrarily initialized relative to the point cloud obtained from the first depth image.

As new regions of the object are revealed, we add more points to represent the previously unseen regions. Instead of having a fixed amount of points fit the new observations, this is advantageous for two reasons. First, the optimization is more convenient as a smaller

[1] See Appendix D in [282] for gradient derivations

number and size of steps are required to fit the new points. Second, the fidelity of the local geometry can be maintained. We use the difference between the observed depth image and the rendered depth image to add these points at every new frame. A point is added if the difference between the values in the rendered depth image and the observed depth image is larger than 10% of the difference between maximum and minimum depth. To project the points to the object frame, we need to use the latest available transformation $\mathbf{T}_{CO}$ between the object and the camera frame. As we add more points, the number of optimization parameters grows proportionally. This slows down the optimization steps. More importantly, the points are added with high density at each frame and may be redundant to the object representation. As a result, we prune points for which opacity drops below 0.6 during reconstruction. Together, this process of addition and removal of points balances the need for fidelity and ease of optimization against the computational complexity.

5.4 Incremental 3D Reconstruction and Tracking

Using the object-centric 3D Gaussian representation, we propose a simple pipeline for incremental 3D reconstruction and tracking as shown in Fig. 5.1. Intuitively, this method stores an evolving internal model of the object and camera motion to render an expected image. Then, the photometric distance or dissimilarity between this image and the observed image represents the gap between the true and the internal model. The object and camera parameters are then refined using gradient-based optimization. The input to our pipeline is a sequential set of RGB-D images. We assume the target spacecraft is sufficiently resolved in the image and its RGB appearance is view-independent. In the following, we describe each component of our method in detail.

5.4.1 3D Reconstruction by Frame-to-model Association

Assuming no constraints on the relative motion between the camera and the target object, our goal is to incrementally incorporate new information revealed in an image sequence to reconstruct and track a specific object in the scene. Consider the rendering process from

the previous section, encapsulated as:

$$\mathbf{I}_r = \mathcal{R}(\boldsymbol{\theta}, \mathbf{T}_{CO}) \tag{5.16}$$

where, $\mathcal{R}$ is the renderer that projects a set of 3D Gaussians representing the object to a rasterized image ($\mathbf{I}_r$). $\boldsymbol{\theta}$ are the object parameters obtained by concatenating the per-point Gaussian parameters: $\boldsymbol{\theta}^i = [\boldsymbol{\mu}^i, \boldsymbol{\Sigma}^i, \mathbf{c}^i, o^i]$ for $i = 0, 1, 2, ..., N$. Since the renderer is fully differentiable, we can compute and propagate gradients from image pixels to the object and camera parameters. We use Adam [283] to optimize the Gaussian and pose parameters.

To reconstruct the object by optimizing the object parameters $\boldsymbol{\theta}$, we require multi-view information to avoid overfitting the representation on restricted regions of the object. On the other hand, to optimize the pose, we only require the information from the current frame to establish the frame-to-model association. Consequently, we separate optimization into two stages that are executed in alternate frames of the incoming stream. During reconstruction, we hold camera parameters constant while optimizing the object parameters. The optimization fits the object parameters to a weighted combination of photometric loss for the color image and L1 loss for the depth image:

$$\begin{aligned} L_{recon} = (1 - \lambda)(L_1)_{color} + \lambda(L_{SSIM})_{color} \\ + \beta(L1)_{depth} \end{aligned} \tag{5.17}$$

where, L_1 is the averaged pixel-wise L1 loss between the observed image and the image with a pose $\mathbf{T}_{CO}^n$ for the n^{th} image. L_{SSIM} is the structural similarity index loss based on [284]. Unlike scene-reconstruction applications of Gaussian splatting, it is computationally impractical for our application to optimize the object representation using all the images available. Therefore, we manage a fixed window of W keyframes to optimize the object representation. Like SLAM pipelines, we aim to select keyframes that are most meaningful for multi-view constraints. To accomplish this we use up to W_{max} views with farthest angular distance between them and always include the current and the previous view.

5.4.2 Frame-to-model Pose Estimation and Tracking

For Tracking, the ground truth pose of the first frame is used to establish a canonical frame, if available. Otherwise, it is initialized with a random rotation and centered at the centroid of the partial point cloud de-projected from the first depth image. For the sake of optimization, we reduce the $SE(3)$ pose to a vector $\mathbf{x}_{CO} = [\mathbf{t}_{CO}, \mathbf{q}_{CO}]^T$, where $\mathbf{q}_{CO} = [q_w, q_x, q_y, q_z]$ is the unit quaternion representing relative orientation. The pose is optimized using:

$$L_{track} = (L_1)_{color} \; + \; \beta(L_1)_{depth}$$

The implementation is done using PyTorch using the 3DGS CUDA rasterizer implementation [170] to obtain the gradients of the loss with respect to the object parameters- $\partial\mathcal{L}/\partial\boldsymbol{\theta}$. The gradients of the loss with respect to the pose parameters $\mathbf{x_{CO}}$ are obtained via auto-differentiation. We project the Gaussians to the camera frame outside the CUDA rasterizer which provides the gradient flow from the analytical gradients $\partial\mathcal{L}/\partial\boldsymbol{\theta}$. At each new frame, we initialize the pose estimate by linearly propagating the pose as in Eq. 5.19, where t is the time index.

$$\mathbf{t}^{t+1} = \mathbf{t}^t + (\mathbf{t}^t - \mathbf{t}^{t-1}) \tag{5.18}$$

$$\mathbf{q}^{t+1} = \mathbf{q}^t \otimes (\mathbf{q}^t \otimes \bar{\mathbf{q}}^{t-1}) \tag{5.19}$$

where, For the tracking iteration, this pose is used as the initial guess and refined through optimization. On the other hand, this extrapolated pose stays fixed for the reconstruction iteration. The operations $\otimes$ and $\bar{\mathbf{q}}$ are quaternion multiplication and inverse respectively. We use 120 optimization steps for reconstruction and 80 steps for tracking.

5.5 Experiments

Our method directly optimizes the object representation and camera pose online. Consequently, no pre-training or large-scale data curation is necessary and it can be tested directly

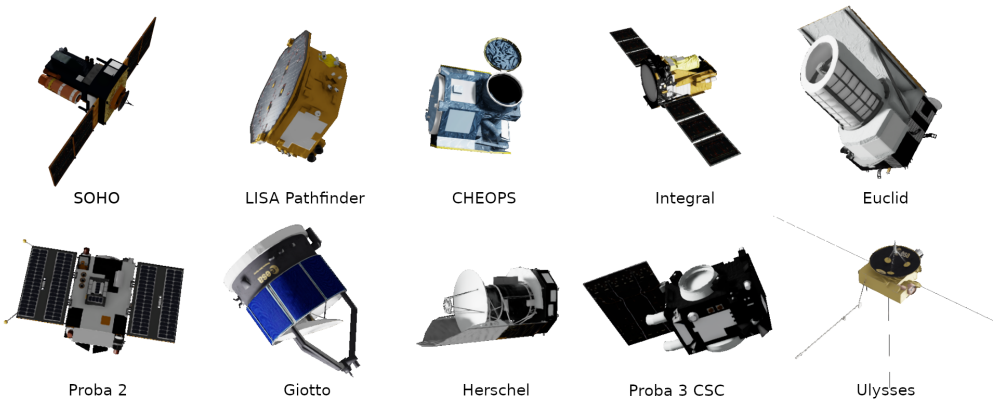


Figure 5.3

Figure 5.4: ESA science fleet spacecraft 3D models used to generate the dataset.

on sequential RGB-D data of any unknown object.

5.5.1 Datasets: Simplified Spacecraft Reconstruction and Tracking Dataset

We test the performance of our method on a custom dataset of image sequences of 10 space objects from ESA's science fleet[2] as shown in Fig. 5.4. For consistency, we simulate the same relative trajectory for all the spacecraft models and the perspective camera model. The simulated relative motion is a spherical spiral as shown in Fig. 5.4 around a sphere of radius 16m. The trajectory allows for spacecraft to incrementally reveal new views and to have varying photometric shifts between time steps along the trajectory. We note that there are no limitations on the nature of this trajectory, as long as the spacecraft is sufficiently resolved in the image and the photometric shifts between two subsequent frames are not large and abrupt. We simplify the data generation process by scaling down the spacecraft models appropriately to fit the FOV, so we can test in under the same trajectory. Further, we simplify the spacecraft's appearance by reducing the material specularity. Since we ignore the view-based appearance model in 3DGS, our method in its current form cannot deal with flares and sharp specular reflections. We use Nvidia Omniverse Isaac Sim with the ray-tracing render engine to generate a sequence of 1000 synthetic images along the trajectory

[2] `https://scifleet.esa.int/`

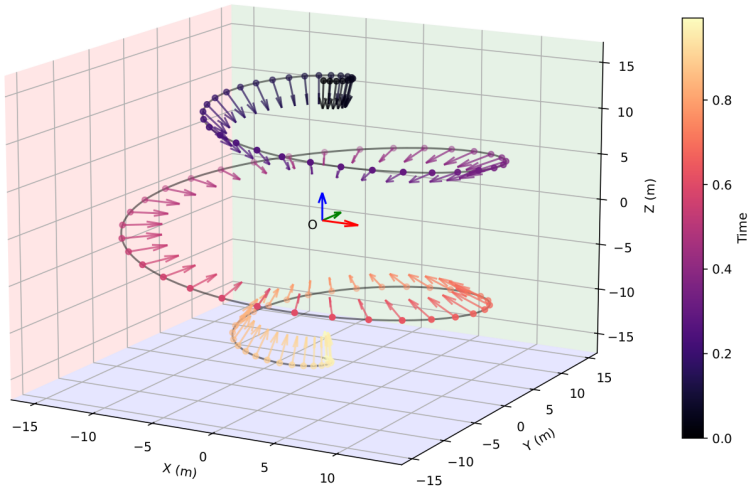


Figure 5.5: Dataset Generation: Camera trajectory relative to the target spacecraft, whose body frame is denoted by the axes at O. The arrows show the boresight direction at each point.

for each object. Through the replicator module, we generate an RGB image, an aligned depth image, a segmentation image, and the relative ground-truth pose for each time step of the trajectory. During data-loading, we add random noise in the color and depth images with a standard deviation of 0.2 (intensity) and 0.025m respectively. Furthermore, we add edge bleeding artifacts in the depth image to emulate a stereo depth output.

5.5.2 Evaluation

We evaluate the performance of our approach by running optimization for reconstruction and tracking in alternate frames. The reconstruction performance is measured by the bi-directional chamfer distance between the ground-truth surface point clouds and the point cloud obtained from the online reconstruction. The chamfer distance metric quantifies the similarity between two point clouds by averaging the pair-wise distances of the closest

points. This is given by:

$$\text{CD}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{2n_1}\sum_{i=1}^{n_1}|x_1^i - x_2^{i*}| + \frac{1}{2n_2}\sum_{j=1}^{n_2}|x_2^j - x_1^{j*}| \tag{5.20}$$

where, x^{i*} and x^{j*} are the nearest neighbor points in the other point cloud. We compute chamfer distance over 20000 points by uniformly sampling from each point cloud.

The tracking error is simply computed as the translation error and rotation error to the annotated ground truth pose:

$$\begin{aligned} t_{err} &= ||\mathbf{t}_{CO} - \mathbf{t}_{CO}^*||_2 \\ \theta_{err} &= 2\,\arccos(\mathbf{q_{CO}} \otimes \bar{\mathbf{q}}_{\mathbf{CO}}^*)_w \end{aligned} \tag{5.21}$$

where, $\mathbf{t}_{CO}^*$ and $\mathbf{t}_{CO}^*$ are ground truth position and orientation.

5.6 Discussion

The qualitative results of 3D reconstruction and pose tracking are shown in Fig. 5.6. While the reconstruction and tracking performance are coupled in our method, it is helpful to analyze their evolution over the trajectory separately at first.Table 5.1 shows the variation of the bi-directional chamfer distance between the reconstructed point cloud and the reference point cloud for each object. As newer views of the target object are revealed, the proposed method generally leads to consistent improvement in the accuracy and completeness of the reconstructed 3D model. The convergence of the reconstruction error is sensitive to the pose tracking error. When new 3D Gaussians are added, the last estimated relative pose is used to project them from the camera frame to the object frame. Consequently, larger pose tracking errors can lead to diverging reconstruction errors. This can be seen in the case of SOHO, Proba 2, and Ulysses models where the chamfer distance at the last step is not the lowest. In the case of Ulysses, the reconstruction errors present a clear sign of early tracking drift leading to sharp divergence. Another notable observation is the much lower

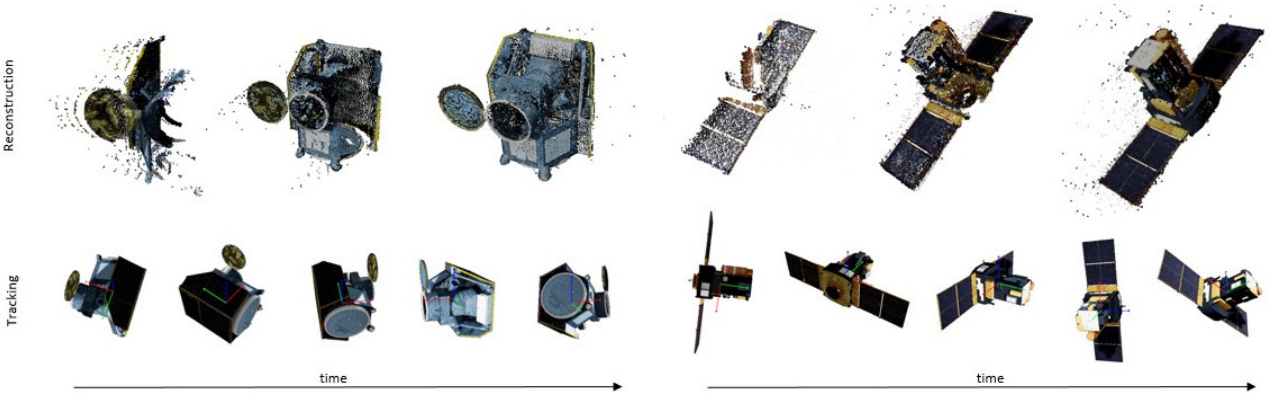


Figure 5.6: Qualitative results of incremental 3D reconstruction and pose tracking of CHEOPS (left) and SOHO (right) spacecraft.

improvement in error between the 500^{th} and the 1000^{th} step. Depending on the spacecraft geometry, most regions of the spacecraft can be revealed by step 500. This effect is dominated by stray points in the reconstruction far away from the local surfaces of the object, resulting from the noise in the depth images. This effect can be observed in the initial point cloud shown in Fig. 5.7 as well as the final reconstructions shown in Fig. 5.6.

The tracking results for each of the spacecraft in terms of translation and rotation error are shown in Fig. 5.7. For clarity, we separate the tracking error for the first 500 steps from that of the entire 1000-step sequence. We outline three direct observations. First, our method allows accurate tracking of diverse target objects over short durations. Second, over 500 steps, the tracking error for 6/10 spacecraft models is maintained under the tight bounds of 0.5 m in translation and 10 degrees in rotation. Finally, over the entire 1000 steps, the tracking error for 6/10 objects diverges uncontrollably, while that of 4/10 objects remains within a reasonable error bound. Together with the reconstruction results, these observations provide the preliminary validation of our incremental approach using 3D Gaussian representations to track unknown objects. On a system with 12th Gen Intel Core i7-12800H CPU and Nvidia RTX 3080 Ti Laptop GPU, it takes approximately 1.4s and 1.1s to optimize the reconstruction (120 steps) and pose (80 steps) respectively.

The experiments demonstrate that the proposed approach and its application to unknown spacecraft tracking are effective. However, the current implementation is limited in robust-

Table 5.1: Evaluation of 3D reconstruction performance using bi-directional chamfer distance

Spacecraft	CD@0	CD@250	CD@500	CD@1000
CHEOPS	1.18±0.02	0.61±0.02	0.54±0.02	**0.54±0.02**
SOHO	1.96±0.00	1.53±0.03	**1.04±0.14**	1.27±0.16
Giotto	0.62±0.01	0.05±0.01	0.02±0.00	**0.01±0.00**
Herschel	0.99±0.01	0.34±0.01	0.26±0.01	**0.22±0.00**
LISA Pathfinder	1.04±0.01	0.57±0.01	0.50±0.01	**0.49±0.01**
Euclid	0.61±0.01	0.19±0.01	0.17±0.00	**0.16±0.01**
Proba 2	0.78±0.01	0.45±0.01	**0.38±0.01**	0.39±0.01
Proba 3 CSC	1.43±0.01	1.11±0.02	0.94±0.12	**0.83±0.08**
Ulysses	**9.94±0.01**	31.97±1.82	34.60±1.31	33.80±1.57
Integral	2.35±0.01	1.96±0.01	1.53±0.16	**1.23±0.27**

ness and practicality by several factors that can be addressed in future works. The foremost limitation of the approach is the tracking drift over a longer duration. The key factor behind this is the lack of a global pose optimization scheme. Since the pose optimization only considers a single frame, any tracking error cascades negatively. Remember that we use the latest pose available in the previous iteration for optimizing the object parameters and adding newer points. While the repeated reconstruction compensates the tracking error to a certain extent, it diverges beyond a certain margin of error. We observe two main sources that cause this error: (1) Visual ambiguity in regions with low-intensity values and (2) view-dependent changes in appearance. The first source is evident from the tracking results of Integral, Proba-2, and Proba-3-CSC spacecraft which have large regions of surfaces that are dark and do not provide enough contribution to the loss. The tracking drift starts rising noticeably when large parts of the spacecraft observable in the image have close to zero intensity. The second factor is observed in the case of Ulysses where the simulated images contain sharp view-dependent artifacts on the long booms of the spacecraft from data generation.

The improvement of the loss function to deal with low-intensity regions can alleviate abrupt tracking errors seen in the experiments. On the other hand, better keyframe man-

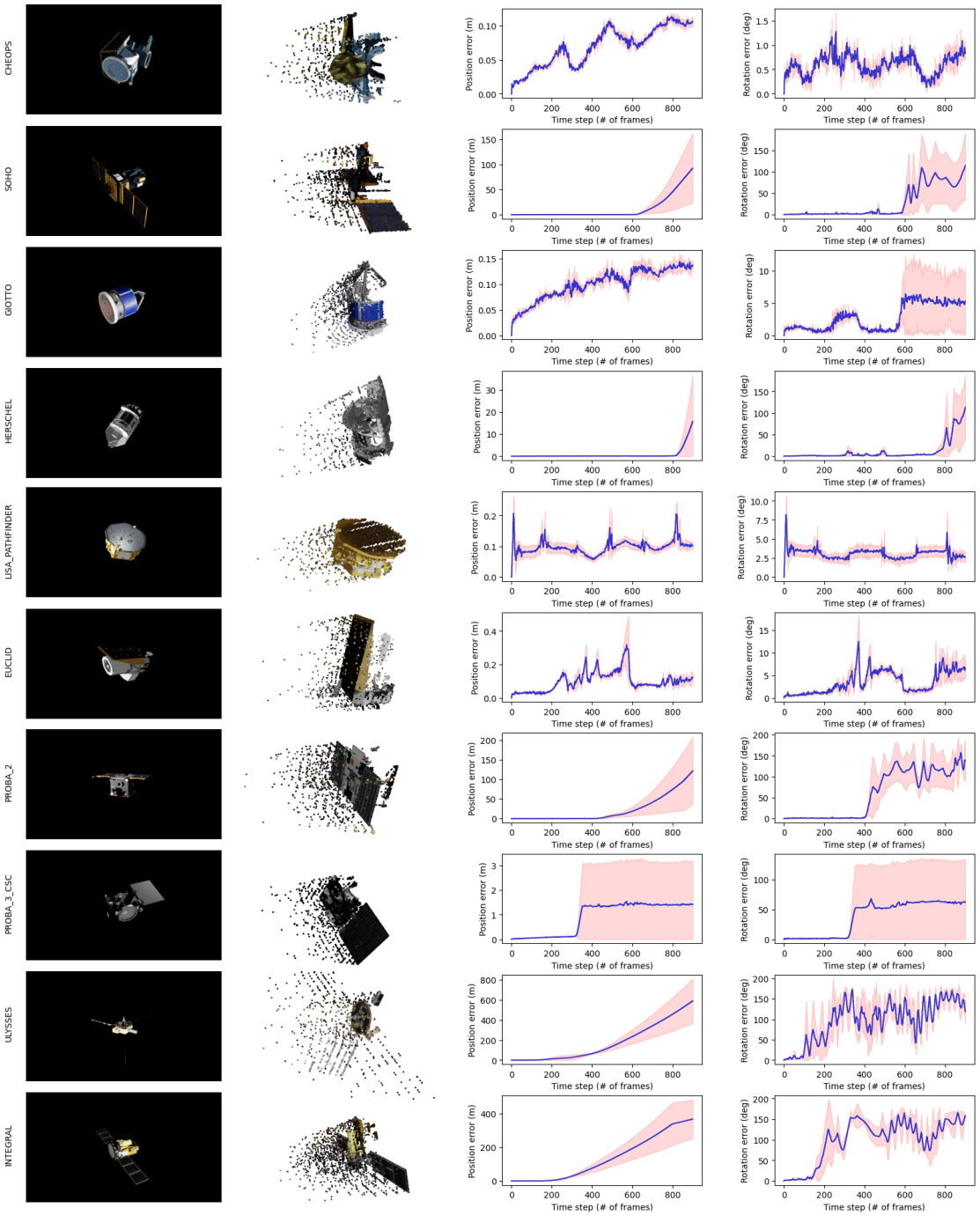


Figure 5.7: Pose tracking errors over 1000 frames for 10 synthetic spacecraft models of diverse geometry and textures. A sample input image and the point cloud of the first RGB-D frame for each spacecraft are shown on the left.

agement and pose graph optimization can solve the long-term tracking drift resulting from accumulating errors. Further, naive extrapolation of the relative pose used directly to reconstruct the object and add more Gaussians limits the magnitude of pose shift that can be handled between two consecutive images. From a design perspective, the requirement of depth images can be relaxed by initializing points with a high variance around the existing points based on the color image only. The requirement of a segmentation mask at each frame may also be relaxed by only taking the segmentation mask in the first image and then using the rendered image to mask the future frames. An important factor to include in future work is the efficient representation of specularity, which may be done using spherical harmonics coefficients for the Gaussians. The computation time can be improved by using a second-order optimizer. Finally, the analysis can be extended to a comprehensive range of relative trajectories including active maneuvers to assess the generalizability of our approach.

5.7 Summary

In this chapter, we proposed a novel approach for incremental reconstruction and pose tracking of unknown dynamic objects using 3D Gaussian representation. It assumes no prior knowledge of the object or its motion and requires no pre-training or neural elements. Relying on a differentiable rendering framework that is fast, it is more suitable for online applications than other contemporary methods. The effectiveness of the method is validated on the task of tracking an unknown dynamic spacecraft. On a custom dataset of ten spacecraft with diverse geometry and appearance, the tracking accuracy over shorter duration is less than 0.5m in translation and 10 deg in rotation. The aspects of longer duration tracking drift are discussed along with recommendations for improvements, paving the way forward for generalizable object-centric perception in space robotics.

Contributions

1. **Development of an efficient object-centric framework for incremental reconstruction and tracking of unknown objects using 3DGS.** The original 3D Gaussian Splatting formulation [170] addresses offline scene reconstruction from posed images. Our work relaxes the need for a static world frame or camera poses and proposes an object-centric framework suitable for online deployment.

2. **Investigation of high-fidelity reconstruction and 6-DoF pose tracking of dynamic unknown spacecraft**. To the best of our knowledge, we provide the first approach to tackle this problem without assuming prior knowledge about the target and its motion.

Chapter 6

Conclusions

This thesis addresses fundamental challenges in robotic manipulation from an object-centric perspective. Three interconnected contributions in this direction targeting generative modeling, real-world deployment, and dynamic object perception are explored. The work maintains a coherent vision of enabling general-purpose robotic manipulation, while maintaining generality for both terrestrial and space applications.

6.1 Generative Modeling for 6-DoF Grasp Synthesis

We introduced GraspLDM, a novel generative framework that learns generalizable representations to model the conditional distributions of 6-DoF grasps for visual observations. GraspLDM enables efficient sampling of a complex conditional distribution of successful grasp poses in a low-dimensional latent space. By combining variational autoencoders with latent diffusion, GraspLDM outperforms existing generative approaches while remaining competitive with state-of-the-art feed-forward models. The framework's flexible architecture enables efficient injection of task-specific conditioning in the latent space without retraining the base model.

Future directions for extending GraspLDM are:

- Integration of grasp quality prediction directly within the GraspLDM architecture to elim-

inate separate classifiers while maintaining task-conditional flexibility.

- Development of more sophisticated task specification mechanisms for complex manipulation sequences.
- Extending training and evaluation to larger-scale datasets and more gripper morphologies.

6.2 Real-world Object-centric Grasping

Our second contribution addressed the critical challenge of bridging simulation-to-reality gaps in robotic grasping through a complete system architecture. The modular design enables effective deployment of grasp synthesis models in unstructured scenarios, achieving approximately 80% success rates with simulation-trained models. The system's adaptability across different hardware configurations while maintaining consistent performance demonstrates its practical utility. Future work in this direction could explore:

- Extension to cluttered/constrained environments and multi-object manipulation scenarios.
- Improve grasp motion planning with parallel execution of multiple grasp hypothesis and more complex task constraints.
- Evaluation with space-relevant scenarios with representative illumination settings.

6.3 Perception of Dynamic Unknown Objects

The final contribution takes a step forward in extending manipulation capabilities to dynamic scenarios through an efficient object-centric framework for incremental 3D reconstruction and tracking. Using 3D Gaussian Splatting, our approach demonstrates promising results in tracking objects with diverse geometry and appearance without prior knowledge of structure or motion - a crucial capability for space applications like satellite servicing. Key areas for future development include:

- Integration of global pose optimization for improved long-term tracking stability.
- Robustness to challenging visual conditions including specular reflections and extreme lighting.
- Integration with real-time grasp synthesis and grasp replanning for dynamic manipulation.

References

[1] Heinrich A Ernst. “MH-1, a computer-operated mechanical hand”. In: *Proceedings of the May 1-3, 1962, spring joint computer conference*. 1962, pp. 39–51 (cit. on p. 3).

[2] John McCarthy et al. “A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955”. In: *AI magazine* 27.4 (2006), pp. 12–12 (cit. on p. 3).

[3] Frank Rosenblatt. *The perceptron, a perceiving and recognizing automaton*. Cornell Aeronautical Laboratory, 1957 (cit. on p. 3).

[4] J Kenneth Salisbury and B Roth. “Kinematic and force analysis of articulated mechanical hands”. In: (1983) (cit. on pp. 3, 11).

[5] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. “A billion ways to grasp: An evaluation of grasp sampling schemes on a dense, physics-based grasp data set”. In: *The International Symposium of Robotics Research*. Springer. 2019, pp. 890–905 (cit. on pp. 4, 36, 38, 39, 74).

[6] Domenico Prattichizzo and Jeffrey C Trinkle. “Grasping”. In: *Springer handbook of robotics* (2016), pp. 955–988 (cit. on pp. 12, 15–18).

[7] Antonio Bicchi and Vijay Kumar. “Robotic grasping and contact: A review”. In: *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*. Vol. 1. IEEE. 2000, pp. 348–353 (cit. on pp. 12, 18).

[8] Jeannette Bohg et al. “Data-driven grasp synthesis—a survey”. In: *IEEE Transactions on robotics* 30.2 (2013), pp. 289–309 (cit. on pp. 12, 19).

[9] Rhys Newbury et al. “Deep learning approaches to grasp synthesis: A review”. In: *arXiv preprint arXiv:2207.02556* (2022) (cit. on pp. 12, 13, 24).

[10] Beatriz León, Antonio Morales, and Joaquín Sancho-Bru. *From robot to human grasping simulation*. Vol. 19. Springer, 2014 (cit. on p. 14).

[11] Franz Reuleaux. *The kinematics of machinery: outlines of a theory of machines*. Vol. 1875 (Original German). Dover, 1963 (cit. on p. 14).

[12] Antonio Bicchi. “On the closure properties of robotic grasping”. In: *The International Journal of Robotics Research* 14.4 (1995), pp. 319–334 (cit. on pp. 14–16).

[13] K Lakshminarayana. “Mechanics of form closure”. In: *ASME report* (1978) (cit. on p. 15).

[14] Máximo A Roa and Raúl Suárez. “Grasp quality measures: review and performance”. In: *Autonomous robots* 38 (2015), pp. 65–88 (cit. on p. 16).

[15] Zexiang Li and S Shankar Sastry. “Task-oriented optimal grasping by multifingered robot hands”. In: *IEEE Journal on Robotics and Automation* 4.1 (1988), pp. 32–44 (cit. on p. 16).

[16] Young C Park and Gregory P Starr. “Grasp synthesis of polygonal objects using a three-fingered robot hand”. In: *The International journal of robotics research* 11.3 (1992), pp. 163–184 (cit. on p. 16).

[17] Byoung-Ho Kim et al. “Optimal grasping based on non-dimensionalized performance indices”. In: *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the Next Millennium (Cat. No. 01CH37180)*. Vol. 2. IEEE. 2001, pp. 949–956 (cit. on p. 16).

[18] Jean Ponce et al. “On computing four-finger equilibrium and force-closure grasps of polyhedral objects”. In: *The International Journal of Robotics Research* 16.1 (1997), pp. 11–35 (cit. on p. 16).

[19] Yun-Hui Liu. “Computing n-finger form-closure grasps on polygonal objects”. In: *The International journal of robotics research* 19.2 (2000), pp. 149–158 (cit. on p. 16).

[20] Van-Duc Nguyen. “Constructing force-closure grasps”. In: *The International Journal of Robotics Research* 7.3 (1988), pp. 3–16 (cit. on p. 16).

[21] Carlo Ferrari, John F Canny, et al. “Planning optimal grasps.” In: *ICRA*. Vol. 3. 4. 1992, p. 6 (cit. on pp. 16, 17, 22, 36, 100).

[22] Andrew T Miller and Peter K Allen. “Examples of 3D grasp quality computations”. In: *Proceedings 1999 IEEE international conference on robotics and automation (Cat. No. 99CH36288C)*. Vol. 2. IEEE. 1999, pp. 1240–1246 (cit. on pp. 16, 17).

[23] Charles A Klein and Bruce E Blaho. “Dexterity measures for the design and control of kinematically redundant manipulators”. In: *The international journal of robotics research* 6.2 (1987), pp. 72–83 (cit. on p. 17).

[24] Tsuneo Yoshikawa. “Manipulability of robotic mechanisms”. In: *The international journal of Robotics Research* 4.2 (1985), pp. 3–9 (cit. on p. 17).

[25] Jeffrey Kerr and Bernard Roth. “Analysis of multifingered hands”. In: *The International Journal of Robotics Research* 4.4 (1986), pp. 3–17 (cit. on p. 17).

[26] F-T Cheng and David E Orin. “Efficient algorithm for optimal force distribution-the compact-dual lp method”. In: *IEEE Transactions on Robotics and Automation* 6.2 (1990), pp. 178–187 (cit. on p. 17).

[27] Yun-Hui Liu. “Qualitative test and force optimization of 3-D frictional form-closure grasps using linear programming”. In: *IEEE Transactions on Robotics and Automation* 15.1 (1999), pp. 163–173 (cit. on p. 17).

[28] Martin Buss, Hideki Hashimoto, and John B Moore. “Dextrous hand grasping force optimization”. In: *IEEE transactions on robotics and automation* 12.3 (1996), pp. 406–418 (cit. on p. 17).

[29] Li Han, Jeffrey C Trinkle, and Zexiang X Li. “Grasp analysis as linear matrix inequality problems”. In: *IEEE Transactions on Robotics and Automation* 16.6 (2000), pp. 663–674 (cit. on p. 17).

[30] Ch Borst, Max Fischer, and Gerd Hirzinger. “A fast and robust grasp planner for arbitrary 3D objects”. In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*. Vol. 3. IEEE. 1999, pp. 1890–1896 (cit. on p. 17).

[31] Rosen Diankov. “Automated construction of robotic manipulation programs”. PhD thesis. Carnegie Mellon University, USA, 2010 (cit. on pp. 17, 19, 97).

[32] Ravi Balasubramanian et al. “Physical human interactive guidance: Identifying grasping principles from human-planned grasps”. In: *IEEE Transactions on Robotics* 28.4 (2012), pp. 899–910 (cit. on pp. 18, 19, 97).

[33] Tomás Lozano-Pérez et al. “Handey: A robot system that recognizes, plans, and manipulates”. In: *Proceedings. 1987 IEEE international conference on robotics and automation*. Vol. 4. IEEE. 1987, pp. 843–849 (cit. on p. 18).

[34] Sharon A Stansfield. “Robotic grasping of unknown objects: A knowledge-based approach”. In: *The International journal of robotics research* 10.4 (1991), pp. 314–326 (cit. on p. 18).

[35] Garfield B Dunn and Jakub Segen. “Automatic discovery of robotic grasp configurations”. In: *Proceedings. 1988 IEEE International Conference on Robotics and Automation*. IEEE. 1988, pp. 396–401 (cit. on p. 18).

[36] Ming Tan. “CSL: A cost-sensitive learning system for sensing and grasping objects”. In: *Proceedings., IEEE International Conference on Robotics and Automation*. IEEE. 1990, pp. 858–863 (cit. on p. 18).

[37] Ishay Kamon, Tamar Flash, and Shimon Edelman. “Learning to grasp using visual information”. In: *Proceedings of IEEE International Conference on Robotics and Automation*. Vol. 3. IEEE. 1996, pp. 2470–2476 (cit. on p. 18).

[38] Andrew T Miller and Peter K Allen. “Graspit! a versatile simulator for robotic grasping”. In: *IEEE Robotics & Automation Magazine* 11.4 (2004), pp. 110–122 (cit. on pp. 18, 41).

[39] Christoph Borst, Max Fischer, and Gerd Hirzinger. “Grasping the dice by dicing the grasp”. In: *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*. Vol. 4. IEEE. 2003, pp. 3692–3697 (cit. on p. 19).

[40] Corey Goldfeder et al. “The columbia grasp database”. In: *2009 IEEE international conference on robotics and automation*. IEEE. 2009, pp. 1710–1716 (cit. on pp. 19, 36).

[41] Kai Huebner and Danica Kragic. “Selection of robot pre-grasps using box-based shape approximation”. In: *2008 IEEE/RSJ international conference on intelligent robots and systems*. IEEE. 2008, pp. 1765–1770 (cit. on pp. 19, 20).

[42] Markus Przybylski, Tamim Asfour, and Rüdiger Dillmann. “Planning grasps for robotic hands using a novel object representation based on the medial axis transform”. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2011, pp. 1781–1788 (cit. on p. 19).

[43] Staffan Ekvall and Danica Kragic. “Interactive grasp learning based on human demonstration”. In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA’04. 2004*. Vol. 4. IEEE. 2004, pp. 3519–3524 (cit. on p. 19).

[44] Charles de Granville, Joshua Southerland, and Andrew H Fagg. “Learning grasp affordances through human demonstration”. In: *Proceedings of the International Conference on Development and Learning (ICDL’06)*. 2006 (cit. on p. 19).

[45] Oliver B Kroemer et al. “Combining active learning and reactive control for robot grasping”. In: *Robotics and Autonomous systems* 58.9 (2010), pp. 1105–1116 (cit. on p. 19).

[46] Renaud Detry et al. “Learning grasp affordance densities”. In: *Paladyn* 2 (2011), pp. 1–17 (cit. on p. 19).

[47] Paul J Best. “A method for registration of 3-D shapes”. In: *IEEE Trans Pattern Anal Mach Vision* 14 (1992), pp. 239–256 (cit. on p. 19).

[48] Chavdar Papazov and Darius Burschka. “An efficient ransac for 3d object recognition in noisy and occluded scenes”. In: *Asian conference on computer vision*. Springer. 2010, pp. 135–148 (cit. on p. 19).

[49] Alvaro Collet, Manuel Martinez, and Siddhartha S Srinivasa. “The moped framework: Object recognition and pose estimation for manipulation”. In: *The international journal of robotics research* 30.10 (2011), pp. 1284–1306 (cit. on p. 19).

[50] Renaud Detry, Nicolas Pugeault, and Justus H Piater. “A probabilistic framework for 3D visual object representation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.10 (2009), pp. 1790–1803 (cit. on p. 19).

[51] Sahar El-Khoury and Anis Sahbani. “Handling objects by their handles”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. POST_TALK. 2008 (cit. on p. 20).

[52] Ashutosh Saxena, Justin Driemeyer, and Andrew Y Ng. “Robotic grasping of novel objects using vision”. In: *The International Journal of Robotics Research* 27.2 (2008), pp. 157–173 (cit. on p. 20).

[53] Luis Montesano et al. “Learning object affordances: from sensory–motor coordination to imitation”. In: *IEEE Transactions on Robotics* 24.1 (2008), pp. 15–26 (cit. on p. 20).

[54] Jeannette Bohg and Danica Kragic. “Learning grasping points with shape context”. In: *Robotics and Autonomous Systems* 58.4 (2010), pp. 362–377 (cit. on p. 20).

[55] Renaud Detry et al. “Generalizing grasps across partly similar objects”. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 3791–3797 (cit. on p. 20).

[56] Alexander Herzog et al. “Template-based learning of grasp selection”. In: *2012 IEEE international conference on robotics and automation*. IEEE. 2012, pp. 2379–2384 (cit. on p. 20).

[57] Renaud Detry et al. “Learning a dictionary of prototypical grasp-predicting parts from grasping experience”. In: *2013 IEEE International Conference on Robotics and Automation*. IEEE. 2013, pp. 601–608 (cit. on p. 20).

[58] Oliver Kroemer et al. “A kernel-based approach to direct action perception”. In: *2012 IEEE international Conference on Robotics and Automation*. IEEE. 2012, pp. 2605–2610 (cit. on p. 20).

[59] Hao Dang and Peter K Allen. “Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task”. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 1311–1317 (cit. on p. 20).

[60] Ulrich Hillenbrand and Maximo A Roa. “Transferring functional grasps through contact warping and local replanning”. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 2963–2970 (cit. on p. 20).

[61] Lucas Manuelli et al. “kpam: Keypoint affordances for category-level robotic manipulation”. In: *The International Symposium of Robotics Research*. Springer. 2019, pp. 132–157 (cit. on p. 20).

[62] Zengyi Qin et al. “Keto: Learning keypoint representations for tool manipulation”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 7278–7285 (cit. on p. 20).

[63] Peter R Florence, Lucas Manuelli, and Russ Tedrake. “Dense object nets: Learning dense visual object descriptors by and for robotic manipulation”. In: *arXiv preprint arXiv:1806.08756* (2018) (cit. on p. 20).

[64] Bowen Wen et al. “Catgrasp: Learning category-level task-relevant grasping in clutter from simulation”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 6401–6408 (cit. on pp. 20, 100).

[65] Dirk Kraft et al. “Birth of the object: Detection of objectness and extraction of object shape through object–action complexes”. In: *International Journal of Humanoid Robotics* 5.02 (2008), pp. 247–265 (cit. on p. 20).

[66] Kaijen Hsiao et al. “Contact-reactive grasping of objects with partial shape information”. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2010, pp. 1228–1235 (cit. on p. 20).

[67] Mila Popović et al. “Grasping unknown objects using an early cognitive vision system for general scene understanding”. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2011, pp. 987–994 (cit. on p. 20).

[68] Claire Dune et al. “Active rough shape estimation of unknown objects”. In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2008, pp. 3622–3627 (cit. on p. 21).

[69] Gary M Bone, Andrew Lambert, and Mark Edwards. “Automated modeling and robotic grasping of unknown three-dimensional objects”. In: *2008 IEEE International Conference on Robotics and Automation*. IEEE. 2008, pp. 292–298 (cit. on p. 21).

[70] Jeannette Bohg et al. “Mind the gap-robotic grasping under incomplete observation”. In: *2011 IEEE international conference on robotics and automation*. IEEE. 2011, pp. 686–693 (cit. on p. 21).

[71] Vincenzo Lippiello et al. “Visual grasp planning for unknown objects using a multifingered robotic hand”. In: *IEEE/ASME Transactions on Mechatronics* 18.3 (2012), pp. 1050–1059 (cit. on p. 21).

[72] Tony Z Zhao et al. “Learning fine-grained bimanual manipulation with low-cost hardware”. In: *arXiv preprint arXiv:2304.13705* (2023) (cit. on p. 21).

[73] Jeffrey Mahler et al. “Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics”. In: *arXiv preprint arXiv:1703.09312* (2017) (cit. on pp. 22, 24, 36, 39).

[74] Douglas Morrison, Peter Corke, and Jürgen Leitner. “Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach”. In: *arXiv preprint arXiv:1804.05172* (2018) (cit. on pp. 22, 23, 42).

[75] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. “Efficient grasping from rgbd images: Learning using a new rectangle representation”. In: *2011 IEEE International conference on robotics and automation*. IEEE. 2011, pp. 3304–3311 (cit. on pp. 22, 71).

[76] Ian Lenz, Honglak Lee, and Ashutosh Saxena. “Deep learning for detecting robotic grasps”. In: *The International Journal of Robotics Research* 34.4-5 (2015), pp. 705–724 (cit. on pp. 22, 24).

[77] Lerrel Pinto and Abhinav Gupta. “Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours”. In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2016, pp. 3406–3413 (cit. on p. 22).

[78] Sulabh Kumra and Christopher Kanan. “Robotic grasp detection using deep convolutional neural networks”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 769–776 (cit. on pp. 23, 24).

[79] Joseph Redmon and Anelia Angelova. “Real-time grasp detection using convolutional neural networks”. In: *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2015, pp. 1316–1322 (cit. on p. 23).

[80] Jacob Varley et al. “Shape completion enabled robotic grasping”. In: *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2017, pp. 2442–2447 (cit. on p. 23).

[81] Umar Asif, Jianbin Tang, and Stefan Harrer. “Densely supervised grasp detector (DSGD)”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 8085–8093 (cit. on p. 23).

[82] Sergey Levine et al. “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection”. In: *The International journal of robotics research* 37.4-5 (2018), pp. 421–436 (cit. on p. 23).

[83] Douglas Morrison et al. “Cartman: The low-cost Cartesian manipulator that won the amazon robotics challenge”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 7757–7764 (cit. on p. 23).

[84] Andy Zeng et al. “Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching”. In: *The International Journal of Robotics Research* 41.7 (2022), pp. 690–705 (cit. on p. 23).

[85] Amaury Depierre, Emmanuel Dellandréa, and Liming Chen. “Jacquard: A large scale dataset for robotic grasp detection”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 3511–3516 (cit. on p. 24).

[86] Hu Cao et al. “Lightweight convolutional neural network with Gaussian-based grasping representation for robotic grasping detection”. In: *arXiv preprint arXiv:2101.10226* (2021) (cit. on p. 24).

[87] Yuzhe Qin et al. “S4g: Amodal single-view single-shot se (3) grasp detection in cluttered scenes”. In: *Conference on robot learning*. PMLR. 2020, pp. 53–65 (cit. on pp. 24, 29).

[88] Yiheng Xie et al. “Neural fields in visual computing and beyond”. In: *Computer Graphics Forum*. Vol. 41. 2. Wiley Online Library. 2022, pp. 641–676 (cit. on pp. 26, 31).

[89] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. “6-dof graspnet: Variational grasp generation for object manipulation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 2901–2910 (cit. on pp. 28, 33–35, 42, 59, 61, 64, 68, 71, 100, 105, 112, 113, 115, 117, 181).

[90] Martin Sundermeyer et al. “Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 13438–13444 (cit. on pp. 28, 30, 34, 41, 59, 61, 71, 75, 100, 104, 105, 112, 113, 115, 117, 182).

[91] Andreas Ten Pas et al. “Grasp pose detection in point clouds”. In: *The International Journal of Robotics Research* 36.13-14 (2017), pp. 1455–1473 (cit. on pp. 28, 33, 100).

[92] Philipp Schmidt et al. “Grasping of unknown objects using deep convolutional neural networks based on depth images”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 6831–6838 (cit. on p. 29).

[93] Charles R Qi et al. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *arXiv preprint arXiv:1706.02413* (2017) (cit. on pp. 29, 30, 70–72, 104).

[94] Binglei Zhao et al. “Regnet: Region-based grasp network for end-to-end grasp detection in point clouds”. In: *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2021, pp. 13474–13480 (cit. on p. 29).

[95] Hao-Shu Fang et al. “Graspnet-1billion: A large-scale benchmark for general object grasping”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11444–11453 (cit. on pp. 29, 36, 100, 113).

[96] Charles R Qi et al. “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”. In: *arXiv preprint arXiv:1612.00593* (2016) (cit. on pp. 29, 70, 104).

[97] Minghao Gou et al. “Rgb matters: Learning 7-dof grasp poses on monocular rgbd images”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 13459–13466 (cit. on p. 29).

[98] Hao-Shu Fang et al. “AnyGrasp: Robust and Efficient Grasp Perception in Spatial and Temporal Domains”. In: *arXiv preprint arXiv:2212.08333* (2022) (cit. on pp. 30, 38, 43, 71, 100).

[99] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. “Acronym: A large-scale grasp dataset based on simulation”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 6222–6227 (cit. on pp. 30, 36, 39, 40, 74, 103).

[100] Zhenyu Jiang et al. “Synergies between affordance and geometry: 6-dof grasp detection via implicit representations”. In: *arXiv preprint arXiv:2104.01542* (2021) (cit. on p. 31).

[101] Michel Breyer et al. “Volumetric grasping network: Real-time 6 dof grasp detection in clutter”. In: *Conference on Robot Learning*. PMLR. 2021, pp. 1602–1611 (cit. on pp. 30, 71).

[102] Mark Van der Merwe et al. “Learning continuous 3d reconstructions for geometrically aware grasping”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 11516–11522 (cit. on pp. 31, 100).

[103] Pinhao Song, Pengteng Li, and Renaud Detry. “Implicit Grasp Diffusion: Bridging the Gap between Dense Prediction and Sampling-based Grasping”. In: *8th Annual Conference on Robot Learning*. 2024 (cit. on pp. 32, 71).

[104] Xibai Lou, Yang Yang, and Changhyun Choi. “Learning to generate 6-dof grasp poses with reachability awareness”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 1532–1538 (cit. on p. 32).

[105] Xibai Lou, Yang Yang, and Changhyun Choi. “Collision-aware target-driven object grasping in constrained environments”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 6364–6370 (cit. on p. 32).

[106] Mia Kokic, Danica Kragic, and Jeannette Bohg. “Learning task-oriented grasping from human activity datasets”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 3352–3359 (cit. on p. 32).

[107] Jens Lundell, Francesco Verdoja, and Ville Kyrki. “Ddgc: Generative deep dexterous grasping in clutter”. In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 6899–6906 (cit. on pp. 32, 71).

[108] Marc A Riedlinger et al. “Model-free grasp learning framework based on physical simulation”. In: *ISR 2020; 52th International Symposium on Robotics*. VDE. 2020, pp. 1–8 (cit. on p. 32).

[109] Marcus Gualtieri et al. “High precision grasp pose detection in dense clutter”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 598–605 (cit. on p. 32).

[110] Andreas Ten Pas and Robert Platt. “Using geometry to detect grasp poses in 3d point clouds”. In: *Robotics Research: Volume 1* (2018), pp. 307–324 (cit. on p. 32).

[111] David Coleman et al. “Reducing the barrier to entry of complex robotic software: a moveit! case study”. In: *arXiv preprint arXiv:1404.3785* (2014) (cit. on pp. 33, 109).

[112] Hongzhuo Liang et al. “Pointnetgpd: Detecting grasp configurations from point sets”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 3629–3635 (cit. on pp. 33, 71, 100).

[113] Adithyavairavan Murali et al. “Same object, different grasps: Data and semantic knowledge for task-oriented grasping”. In: *Conference on robot learning*. PMLR. 2021, pp. 1540–1557 (cit. on pp. 33, 35).

[114] Matt Corsaro, Stefanie Tellex, and George Konidaris. “Learning to detect multi-modal grasps for dexterous grasping in dense clutter”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 4647–4653 (cit. on p. 33).

[115] Matthew Veres, Medhat Moussa, and Graham W Taylor. “An integrated simulator and dataset that combines grasping and vision for deep learning”. In: *arXiv preprint arXiv:1702.02103* (2017) (cit. on pp. 33, 36, 37).

[116] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013) (cit. on pp. 33, 56).

[117] Adithyavairavan Murali et al. “6-dof grasping for target-driven object manipulation in clutter”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 6232–6238 (cit. on pp. 34, 59, 71).

[118] Shengjia Zhao, Jiaming Song, and Stefano Ermon. “Infovae: Information maximizing variational autoencoders”. In: *arXiv preprint arXiv:1706.02262* (2017) (cit. on p. 34).

[119] Hao Fu et al. “Cyclical annealing schedule: A simple approach to mitigating kl vanishing”. In: *arXiv preprint arXiv:1903.10145* (2019) (cit. on pp. 34, 64, 66).

[120] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6840–6851 (cit. on pp. 34, 59, 60, 67, 68).

[121] Calvin Luo. “Understanding diffusion models: A unified perspective”. In: *arXiv preprint arXiv:2208.11970* (2022) (cit. on pp. 34, 60).

[122] Yang Song et al. “Score-based generative modeling through stochastic differential equations”. In: *arXiv preprint arXiv:2011.13456* (2020) (cit. on pp. 34, 60).

[123] Julen Urain et al. “SE (3)-DiffusionFields: Learning cost functions for joint grasp and motion optimization through diffusion”. In: *arXiv preprint arXiv:2209.03855* (2022) (cit. on pp. 34, 64, 77–80, 83, 108, 115).

[124] Puhao Li et al. “Gendexgrasp: Generalizable dexterous grasping”. In: *arXiv preprint arXiv:2210.00722* (2022) (cit. on pp. 35, 36, 38).

[125] Renaud Detry, Jeremie Papon, and Larry Matthies. “Task-oriented grasping with semantic and geometric scene understanding”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 3266–3273 (cit. on p. 35).

[126] Jens Lundell et al. “Constrained generative sampling of 6-dof grasps”. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023, pp. 2940–2946 (cit. on pp. 35, 59, 71).

[127] Robin Rombach et al. “High-resolution image synthesis with latent diffusion models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10684–10695 (cit. on pp. 35, 63).

[128] Douglas Morrison, Peter Corke, and Jürgen Leitner. “Egad! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation”. In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4368–4375 (cit. on pp. 36, 39).

[129] Daniel Kappler, Jeannette Bohg, and Stefan Schaal. “Leveraging big data for grasp planning”. In: *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2015, pp. 4304–4311 (cit. on pp. 36, 37, 39, 100).

[130] Lin Shao et al. “Unigrasp: Learning a unified model to grasp with multifingered robotic hands”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 2286–2293 (cit. on pp. 36, 37).

[131] Ruicheng Wang et al. “Dexgraspnet: A large-scale robotic dexterous grasp dataset for general objects based on simulation”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 11359–11366 (cit. on p. 36).

[132] Dylan Turpin et al. “Fast-Grasp’D: Dexterous Multi-finger Grasp Generation Through Differentiable Simulation”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 8082–8089 (cit. on pp. 36, 38).

[133] Philip Shilane et al. “The princeton shape benchmark”. In: *Proceedings Shape Modeling Applications, 2004.* IEEE. 2004, pp. 167–178 (cit. on p. 36).

[134] Matei Ciocarlie, Corey Goldfeder, and Peter Allen. “Dimensionality reduction for hand-independent dexterous robotic grasping”. In: *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2007, pp. 3270–3275 (cit. on p. 36).

[135] Rosen Diankov and James Kuffner. “Openrave: A planning architecture for autonomous robotics”. In: *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34* 79 (2008) (cit. on p. 37).

[136] Eric Rohmer, Surya PN Singh, and Marc Freese. “V-REP: A versatile and scalable robot simulation framework”. In: *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE. 2013, pp. 1321–1326 (cit. on p. 37).

[137] Lirui Wang, Yu Xiang, and Dieter Fox. “Manipulation trajectory optimization with online grasp synthesis and selection”. In: *arXiv preprint arXiv:1911.10280* (2019) (cit. on pp. 37, 108, 109).

[138] Jeffrey Mahler et al. “Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards”. In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2016, pp. 1957–1964 (cit. on p. 39).

[139] Manolis Savva, Angel X Chang, and Pat Hanrahan. “Semantically-enriched 3D models for common-sense knowledge”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2015, pp. 24–31 (cit. on p. 39).

[140] Miles Macklin et al. “Unified particle physics for real-time applications”. In: *ACM Transactions on Graphics (TOG)* 33.4 (2014), pp. 1–12 (cit. on pp. 40, 74).

[141] Michael Danielczuk et al. “Reach: Reducing false negatives in robot grasp planning with a robust efficient area contact hypothesis model”. In: *The International Symposium of Robotics Research*. Springer. 2019, pp. 757–772 (cit. on p. 40).

[142] Valerio Ortenzi et al. “Object handovers: a review for robotics”. In: *IEEE Transactions on Robotics* 37.6 (2021), pp. 1855–1873 (cit. on p. 41).

[143] Fahad Islam et al. “Provably constant-time planning and replanning for real-time grasping objects off a conveyor belt”. In: *The International journal of robotics research* 40.12-14 (2021), pp. 1370–1384 (cit. on p. 41).

[144] Roshena MacPherson et al. “Trajectory optimization for dynamic grasping in space using adhesive grippers”. In: *Field and Service Robotics: Results of the 11th International Conference*. Springer. 2018, pp. 49–64 (cit. on pp. 41, 47).

[145] Roberto Lampariello. “Motion planning for the on-orbit grasping of a non-cooperative target satellite with collision avoidance”. In: *i-SAIRAS 2010* (2010) (cit. on p. 41).

[146] Nasser Houshangi. “Control of a robotic manipulator to grasp a moving target using vision”. In: *Proceedings., IEEE International Conference on Robotics and Automation*. IEEE. 1990, pp. 604–609 (cit. on p. 41).

[147] Danica Kragic, Andrew T Miller, and Peter K Allen. “Real-time tracking meets online grasp planning”. In: *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*. Vol. 3. IEEE. 2001, pp. 2460–2465 (cit. on p. 41).

[148] Seungsu Kim, Ashwini Shukla, and Aude Billard. “Catching objects in flight”. In: *IEEE Transactions on Robotics* 30.5 (2014), pp. 1049–1065 (cit. on p. 41).

[149] Arjun Menon, Benjamin Cohen, and Maxim Likhachev. “Motion planning for smooth pickup of moving objects”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 453–460 (cit. on p. 41).

[150] Iretiayo Akinola et al. “Dynamic grasping with reachability and motion awareness”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 9422–9429 (cit. on p. 41).

[151] Mederic Fourmy et al. “Visually Guided Model Predictive Robot Control via 6D Object Pose Localization and Tracking”. In: *arXiv preprint arXiv:2311.05344* (2023) (cit. on p. 41).

[152] Patrick Rosenberger et al. “Object-independent human-to-robot handovers using real time robotic vision”. In: *IEEE Robotics and Automation Letters* 6.1 (2020), pp. 17–23 (cit. on p. 42).

[153] Wei Yang et al. “Reactive human-to-robot handovers of arbitrary objects”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 3118–3124 (cit. on p. 42).

[154] Nathan D Ratliff et al. “Riemannian motion policies”. In: *arXiv preprint arXiv:1801.02854* (2018) (cit. on p. 43).

[155] James J Kuffner and Steven M LaValle. “RRT-connect: An efficient approach to single-query path planning”. In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. Vol. 2. IEEE. 2000, pp. 995–1001 (cit. on pp. 43, 107, 112).

[156] Prannay Khosla et al. “Supervised contrastive learning”. In: *Advances in neural information processing systems* 33 (2020), pp. 18661–18673 (cit. on p. 43).

[157] Jirong Liu et al. “Target-referenced reactive grasping for dynamic objects”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 8824–8833 (cit. on p. 43).

[158] Naresh Marturi et al. “Dynamic grasp and trajectory planning for moving objects”. In: *Autonomous Robots* 43 (2019), pp. 1241–1256 (cit. on p. 43).

[159] Marek Kopicki et al. “One-shot learning and generation of dexterous grasps for novel objects”. In: *The International Journal of Robotics Research* 35.8 (2016), pp. 959–976 (cit. on p. 44).

[160] Jakob Engel, Vladlen Koltun, and Daniel Cremers. “Direct sparse odometry”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.3 (2017), pp. 611–625 (cit. on p. 44).

[161] Christian Forster et al. “SVO: Semidirect visual odometry for monocular and multicamera systems”. In: *IEEE Transactions on Robotics* 33.2 (2016), pp. 249–265 (cit. on p. 44).

[162] Zachary Teed and Jia Deng. “Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras”. In: *Advances in neural information processing systems* 34 (2021), pp. 16558–16569 (cit. on p. 44).

[163] Shahram Izadi et al. “Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera”. In: *Proceedings of the 24th annual ACM symposium on User interface software and technology*. 2011, pp. 559–568 (cit. on p. 44).

[164] Lu Ma et al. “Simultaneous localization, mapping, and manipulation for unsupervised object discovery”. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 1344–1351 (cit. on p. 44).

[165] Paul J Besl and Neil D McKay. “Method for registration of 3-D shapes”. In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. Spie. 1992, pp. 586–606 (cit. on p. 44).

[166] Ben Mildenhall et al. “Nerf: Representing scenes as neural radiance fields for view synthesis”. In: *Communications of the ACM* 65.1 (2021), pp. 99–106 (cit. on pp. 44, 125, 132).

[167] Edgar Sucar et al. “iMAP: Implicit mapping and positioning in real-time”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 6229–6238 (cit. on pp. 44, 126).

[168] Mohammad Mahdi Johari, Camilla Carta, and François Fleuret. “Eslam: Efficient dense slam system based on hybrid representation of signed distance fields”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 17408–17419 (cit. on p. 44).

[169] Bowen Wen et al. “BundleSDF: Neural 6-DoF Tracking and 3D Reconstruction of Unknown Objects”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 606–617 (cit. on p. 44).

[170] Bernhard Kerbl et al. “3D Gaussian Splatting for Real-Time Radiance Field Rendering”. In: *ACM Transactions on Graphics* 42.4 (2023) (cit. on pp. 44, 126, 136, 144).

[171] Huajian Huang et al. “Photo-slam: Real-time simultaneous localization and photorealistic mapping for monocular, stereo, and rgb-d cameras”. In: *arXiv preprint arXiv:2311.16728* (2023) (cit. on p. 44).

[172] Hidenobu Matsuki et al. “Gaussian splatting slam”. In: *arXiv preprint arXiv:2312.06741* (2023) (cit. on p. 44).

[173] Nikhil Keetha et al. “Splatam: Splat, track & map 3d gaussians for dense rgb-d slam”. In: *arXiv preprint arXiv:2312.02126* (2023) (cit. on p. 44).

[174] Muhammad Zubair Irshad et al. “Neural Fields in Robotics: A Survey”. In: *arXiv preprint arXiv:2410.20220* (2024) (cit. on p. 45).

[175] Robert B Leighton. “The photographs from Mariner IV”. In: *Scientific American* 214.4 (1966), pp. 54–71 (cit. on p. 45).

[176] MC Malin et al. “Design and development of the Mars Observer Camera”. In: *International Journal of Imaging Systems and Technology* 3.2 (1991), pp. 76–91 (cit. on p. 45).

[177] S Synnott et al. “Interplanetary optical navigation-Voyager Uranus encounter”. In: *Astrodynamics Conference*. 1986, p. 2113 (cit. on p. 45).

[178] Shyam Bhaskaran et al. “Navigation of the deep space 1 spacecraft at Borrelly”. In: *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*. 2002, p. 4815 (cit. on p. 45).

[179] Richard T Howard et al. “The video guidance sensor-a flight proven technology”. In: *Guidance and Control*. 1999 (cit. on p. 45).

[180] Richard T Howard et al. “Advanced video guidance sensor (AVGS) development testing”. In: *Spaceborne Sensors*. Vol. 5418. SPIE. 2004, pp. 50–60 (cit. on p. 45).

[181] Allan Read Eisenman et al. “Mars exploration rover engineering cameras”. In: *Sensors, Systems, and Next-Generation Satellites V*. Vol. 4540. SPIE. 2001, pp. 288–297 (cit. on p. 45).

[182] Mark Maimone, Yang Cheng, and Larry Matthies. “Two years of visual odometry on the mars exploration rovers”. In: *Journal of Field Robotics* 24.3 (2007), pp. 169–186 (cit. on p. 45).

[183] Glenn Jorgensen and Elizabeth Bains. “SRMS history, evolution and lessons learned”. In: *AIAA SPACE 2011 Conference & Exposition*. 2011, p. 7277 (cit. on p. 45).

[184] Mai Lee Chang and Jessica J Marquez. *Human-automation allocations for current robotic space operations: space station remote manipulator system*. Tech. rep. 2018 (cit. on p. 45).

[185] Mitsushige Oda, Kouichi Kibe, and Fumio Yamagata. “ETS-VII, space robot in-orbit experiment satellite”. In: *Proceedings of IEEE international conference on robotics and automation*. Vol. 1. IEEE. 1996, pp. 739–744 (cit. on p. 45).

[186] G Visentin and F Didot. “Testing space robotics on the Japanese ETS-VII satellite”. In: *ESA bulletin* 99 (1999), pp. 61–65 (cit. on p. 45).

[187] Andrew Ogilvie et al. “Autonomous robotic operations for on-orbit satellite servicing”. In: *Sensors and Systems for Space Applications II*. Vol. 6958. SPIE. 2008, pp. 50–61 (cit. on p. 45).

[188] Dale Arney et al. “In-space Servicing, Assembly, and Manufacturing (ISAM) State of Play-2023 Edition”. In: *Consortium for Space Mobility and ISAM Capabilities (COSMIC) Kickoff*. 2023 (cit. on p. 45).

[189] Lina María Amaya-Mejía et al. “Visual Servoing for Robotic On-Orbit Servicing: A Survey”. In: *2024 International Conference on Space Robotics (iSpaRo)*. IEEE. 2024, pp. 178–185 (cit. on p. 45).

[190] Steve Creech, John Guidi, and Darcy Elburn. “Artemis: An Overview of NASA’s Activities to Return Humans to the Moon”. In: *2022 IEEE Aerospace Conference (AERO)*. 2022, pp. 1–7. DOI: 10.1109/AERO53065.2022.9843277 (cit. on p. 46).

[191] Marguerite Syvertson et al. *Expanding the Horizons of Mars Science*. Technical Report. Prepared for NASA Science Mission Directorate’s Mars Exploration Program (MEP). Program Directors: Eric Ianson (Director), Tiffany Morgan (Deputy Director), Joe Parrish (Manager). National Aeronautics and Space Administration (NASA), 2024 (cit. on p. 46).

[192] Vinayak Vadlamani et al. “STAARK-Affordable and Mission-Agnostic Robotic Arm”. In: *Proceedings of ASTRA*. European Space Agency. 2023 (cit. on p. 46).

[193] Steffen Jaekel et al. “Design and operational elements of the robotic subsystem for the e. deorbit debris removal mission”. In: *Frontiers in Robotics and AI* 5 (2018), p. 100 (cit. on p. 46).

[194] Kosuke Fujii et al. “Comparative Analysis of Robotic Gripping Solutions for Cooperative and Non-cooperative Targets”. In: *2024 IEEE Aerospace Conference*. IEEE. 2024, pp. 1–14 (cit. on p. 46).

[195] Robert O Ambrose et al. “Robonaut: NASA’s space humanoid”. In: *IEEE Intelligent Systems and Their Applications* 15.4 (2000), pp. 57–63 (cit. on p. 47).

[196] Myron A Diftler et al. “Robonaut 2-the first humanoid robot in space”. In: *2011 IEEE international conference on robotics and automation*. IEEE. 2011, pp. 2178–2183 (cit. on p. 47).

[197] Lyndon B Bridgwater et al. “The robonaut 2 hand-designed to do work with tools”. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 3425–3430 (cit. on p. 47).

[198] Toby B Martin et al. “Tactile gloves for autonomous grasping with the NASA/DARPA Robonaut”. In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA’04. 2004*. Vol. 2. IEEE. 2004, pp. 1713–1718 (cit. on p. 47).

[199] Li Yang Ku, Erik G Learned-Miller, and Roderic A Grupen. *Associating grasping with convolutional neural network features*. 2016 (cit. on p. 47).

[200] Li Yang Ku et al. “A Framework for Dexterous Manipulation”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 4131–4138 (cit. on p. 47).

[201] Maria Bualat et al. “Astrobee: Developing a free-flying robot for the international space station”. In: *AIAA SPACE 2015 conference and exposition*. 2015, p. 4643 (cit. on p. 47).

[202] Yang Cheng, Mark W Maimone, and Larry Matthies. “Visual odometry on the Mars exploration rovers-a tool to ensure accurate driving and science imaging”. In: *IEEE Robotics & Automation Magazine* 13.2 (2006), pp. 54–62 (cit. on p. 47).

[203] Brian K Muirhead et al. “Mars sample return campaign concept status”. In: *Acta Astronautica* 176 (2020), pp. 131–138 (cit. on p. 47).

[204] Israel Raul Tinini Alvarez et al. “SAMPLE-TUBE POSE ESTIMATION BASED ON TWO-STAGE APPROACH FOR FETCHING ON MARS”. In: (2022) (cit. on p. 47).

[205] Raul Castilla-Arquillo et al. “Hardware-accelerated mars sample localization via deep transfer learning from photorealistic simulations”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 12555–12561 (cit. on p. 47).

[206] Andrej Orsula et al. “Learning to Grasp on the Moon from 3D Octree Observations with Deep Reinforcement Learning”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 4112–4119 (cit. on p. 48).

[207] Roberto Opromolla et al. “A review of cooperative and uncooperative spacecraft pose determination techniques for close-proximity operations”. In: *Progress in Aerospace Sciences* 93 (2017), pp. 53–72 (cit. on p. 48).

[208] Bo J Naasz et al. “The HST SM4 relative navigation sensor system: overview and preliminary testing results from the flight robotics lab”. In: *The Journal of the Astronautical Sciences* 57.1-2 (2009), pp. 457–483 (cit. on p. 48).

[209] Simone D’Amico, Mathias Benn, and John L Jørgensen. “Pose estimation of an uncooperative spacecraft from actual space imagery”. In: *International Journal of Space Science and Engineering 5* 2.2 (2014), pp. 171–189 (cit. on p. 48).

[210] John Canny. “A computational approach to edge detection”. In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), pp. 679–698 (cit. on p. 48).

[211] Chris Harris, Mike Stephens, et al. “A combined corner and edge detector”. In: *Alvey vision conference*. Vol. 15. 50. Citeseer. 1988, pp. 10–5244 (cit. on p. 48).

[212] David G Lowe. “Object recognition from local scale-invariant features”. In: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee. 1999, pp. 1150–1157 (cit. on p. 48).

[213] Sumant Sharma and Simone D’Amico. “Neural network-based pose estimation for noncooperative spacecraft rendezvous”. In: *IEEE Transactions on Aerospace and Electronic Systems* 56.6 (2020), pp. 4638–4658 (cit. on p. 48).

[214] Tae Ha Park et al. “SPEED+: Next-generation dataset for spacecraft pose estimation across domain gap”. In: *2022 IEEE Aerospace Conference (AERO)*. IEEE. 2022, pp. 1–15 (cit. on p. 48).

[215] Kevin Black et al. “Real-time, flight-ready, non-cooperative spacecraft pose estimation using monocular imagery”. In: *arXiv preprint arXiv:2101.09553* (2021) (cit. on p. 48).

[216] Juan Ignacio Bravo Pérez-Villar, Álvaro García-Martín, and Jesús Bescós. “Spacecraft Pose Estimation Based on Unsupervised Domain Adaptation and on a 3D-Guided Loss Combination”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 37–52 (cit. on p. 48).

[217] Tae Ha Park and Simone D’Amico. “Online Supervised Training of Spaceborne Vision during Proximity Operations using Adaptive Kalman Filtering”. In: *arXiv preprint arXiv:2309.11645* (2023) (cit. on p. 48).

[218] Lorenzo Pasqualetto Cassinis et al. “Evaluation of tightly-and loosely-coupled approaches in CNN-based pose estimation systems for uncooperative spacecraft”. In: *Acta Astronautica* 182 (2021), pp. 189–202 (cit. on p. 48).

[219] Tae Ha Park and Simone D’Amico. “Adaptive Neural-Network-Based Unscented Kalman Filter for Robust Pose Tracking of Noncooperative Spacecraft”. In: *Journal of Guidance, Control, and Dynamics* 46.9 (2023), pp. 1671–1688 (cit. on p. 48).

[220] Tae Ha Park and Simone D’Amico. “Rapid Abstraction of Spacecraft 3D Structure from Single 2D Image”. In: *AIAA SCITECH 2024 Forum*. 2024, p. 2768 (cit. on p. 49).

[221] Alexander Kirillov et al. “Segment anything”. In: *arXiv preprint arXiv:2304.02643* (2023) (cit. on pp. 54, 98, 105, 112).

[222] Nikhila Ravi et al. “Sam 2: Segment anything in images and videos”. In: *arXiv preprint arXiv:2408.00714* (2024) (cit. on pp. 54, 106).

[223] Diederik P Kingma, Max Welling, et al. “An introduction to variational autoencoders”. In: *Foundations and Trends® in Machine Learning* 12.4 (2019), pp. 307–392 (cit. on p. 56).

[224] Zehang Weng et al. “GoNet: An Approach-Constrained Generative Grasp Sampling Network”. In: *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*. IEEE. 2023, pp. 1–7 (cit. on p. 59).

[225] Chaozheng Wu et al. “Grasp proposal networks: An end-to-end solution for visual learning of robotic grasps”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 13174–13184 (cit. on p. 59).

[226] Jens Lundell et al. “Multi-fingan: Generative coarse-to-fine sampling of multi-finger grasps”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 4495–4501 (cit. on p. 59).

[227] Yang Song and Stefano Ermon. “Generative modeling by estimating gradients of the data distribution”. In: *Advances in neural information processing systems* 32 (2019) (cit. on p. 59).

[228] Jascha Sohl-Dickstein et al. “Deep unsupervised learning using nonequilibrium thermodynamics”. In: *International Conference on Machine Learning*. PMLR. 2015, pp. 2256–2265 (cit. on pp. 60, 82).

[229] Xiaohui Zeng et al. “LION: Latent Point Diffusion Models for 3D Shape Generation”. In: *arXiv preprint arXiv:2210.06978* (2022) (cit. on pp. 60, 90).

[230] Jakub Tomczak and Max Welling. “VAE with a VampPrior”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2018, pp. 1214–1223 (cit. on p. 61).

[231] Samuel R Bowman et al. “Generating sentences from a continuous space”. In: *arXiv preprint arXiv:1511.06349* (2015) (cit. on p. 62).

[232] Chris Cremer, Xuechen Li, and David Duvenaud. “Inference suboptimality in variational autoencoders”. In: *International conference on machine learning*. PMLR. 2018, pp. 1078–1086 (cit. on pp. 62, 93).

[233] Irina Higgins et al. “beta-vae: Learning basic visual concepts with a constrained variational framework”. In: *International conference on learning representations*. 2017 (cit. on p. 64).

[234] John L Crassidis and F Landis Markley. “Attitude estimation using modified Rodrigues parameters”. In: *Flight Mechanics/Estimation Theory Symposium 1996*. 1996 (cit. on p. 70).

[235] Zhijian Liu et al. “Point-voxel cnn for efficient 3d deep learning”. In: *Advances in Neural Information Processing Systems* 32 (2019) (cit. on pp. 71, 72).

[236] Yuxin Wu and Kaiming He. “Group normalization”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 3–19 (cit. on p. 73).

[237] Prajit Ramachandran, Barret Zoph, and Quoc V Le. “Searching for activation functions”. In: *arXiv preprint arXiv:1710.05941* (2017) (cit. on p. 73).

[238] Sergey Ioffe. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015) (cit. on p. 73).

[239] Adam Paszke et al. “Pytorch: An imperative style, high-performance deep learning library”. In: *Advances in neural information processing systems* 32 (2019) (cit. on p. 73).

[240] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on p. 73).

[241] Ethan Perez et al. “Film: Visual reasoning with a general conditioning layer”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018 (cit. on p. 74).

[242] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017) (cit. on p. 74).

[243] Manolis Savva, Angel X. Chang, and Pat Hanrahan. “Semantically-Enriched 3D Models for Common-sense Knowledge”. In: *CVPR 2015 Workshop on Functionality, Physics, Intentionality and Causality* (2015) (cit. on p. 74).

[244] Viktor Makoviychuk et al. “Isaac gym: High performance gpu-based physics simulation for robot learning”. In: *arXiv preprint arXiv:2108.10470* (2021) (cit. on p. 77).

[245] Jiaming Song, Chenlin Meng, and Stefano Ermon. “Denoising diffusion implicit models”. In: *arXiv preprint arXiv:2010.02502* (2020) (cit. on p. 82).

[246] David Ha, Andrew Dai, and Quoc V Le. “Hypernetworks”. In: *arXiv preprint arXiv:1609.09106* (2016) (cit. on p. 88).

[247] Jonathan Weisz and Peter K Allen. “Pose error robust grasping from contact wrench space metrics”. In: *2012 IEEE international conference on robotics and automation*. IEEE. 2012, pp. 557–562 (cit. on p. 97).

[248] Tianhe Ren et al. “Grounded sam: Assembling open-world models for diverse visual tasks”. In: *arXiv preprint arXiv:2401.14159* (2024) (cit. on pp. 98, 106).

[249] Xinchen Yan et al. “Learning 6-dof grasping interaction via deep geometry-aware 3d representations”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 3766–3773 (cit. on p. 100).

[250] Kaiming He et al. “Mask r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969 (cit. on p. 104).

[251] Yi Li et al. “Fully convolutional instance-aware semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2359–2367 (cit. on p. 104).

[252] Christopher Xie et al. “Unseen object instance segmentation for robotic environments”. In: *IEEE Transactions on Robotics* 37.5 (2021), pp. 1343–1359 (cit. on pp. 104, 112).

[253] Maxime Oquab et al. “Dinov2: Learning robust visual features without supervision”. In: *arXiv preprint arXiv:2304.07193* (2023) (cit. on p. 105).

[254] Danny Driess et al. “Palm-e: An embodied multimodal language model”. In: *arXiv preprint arXiv:2303.03378* (2023) (cit. on p. 105).

[255] Xi Chen et al. “Pali: A jointly-scaled multilingual language-image model”. In: *arXiv preprint arXiv:2209.06794* (2022) (cit. on p. 105).

[256] Haotian Liu et al. “Visual instruction tuning”. In: *Advances in neural information processing systems* 36 (2024) (cit. on p. 105).

[257] Chaoning Zhang et al. “Faster segment anything: Towards lightweight sam for mobile applications”. In: *arXiv preprint arXiv:2306.14289* (2023) (cit. on p. 105).

[258] Martin Ester et al. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: *kdd*. Vol. 96. 34. 1996, pp. 226–231 (cit. on p. 106).

[259] Nathan Ratliff et al. “CHOMP: Gradient optimization techniques for efficient motion planning”. In: *2009 IEEE international conference on robotics and automation*. IEEE. 2009, pp. 489–494 (cit. on pp. 107, 112).

[260] Mrinal Kalakrishnan et al. “STOMP: Stochastic trajectory optimization for motion planning”. In: *2011 IEEE international conference on robotics and automation*. IEEE. 2011, pp. 4569–4574 (cit. on pp. 107, 112).

[261] John Schulman et al. “Motion planning with sequential convex optimization and convex collision checking”. In: *The International Journal of Robotics Research* 33.9 (2014), pp. 1251–1270 (cit. on pp. 107, 112).

[262] Lydia E Kavraki et al. “Probabilistic roadmaps for path planning in high-dimensional configuration spaces”. In: *IEEE transactions on Robotics and Automation* 12.4 (1996), pp. 566–580 (cit. on p. 107).

[263] Steven LaValle. “Rapidly-exploring random trees: A new tool for path planning”. In: *Research Report 9811* (1998) (cit. on p. 107).

[264] Sertac Karaman and Emilio Frazzoli. “Sampling-based algorithms for optimal motion planning”. In: *The international journal of robotics research* 30.7 (2011), pp. 846–894 (cit. on pp. 108, 112).

[265] Mustafa Mukadam et al. “Continuous-time Gaussian process motion planning via probabilistic inference”. In: *The International Journal of Robotics Research* 37.11 (2018), pp. 1319–1340 (cit. on p. 108).

[266] Sean Murray et al. “Robot motion planning on a chip.” In: *Robotics: Science and Systems*. Vol. 6. 2016 (cit. on p. 108).

[267] Balakumar Sundaralingam et al. “CuRobo: Parallelized collision-free minimum-jerk robot motion generation”. In: *arXiv preprint arXiv:2310.17274* (2023) (cit. on p. 108).

[268] Nikolaus Vahrenkamp et al. “Integrated grasp and motion planning”. In: *2010 IEEE International Conference on Robotics and Automation*. IEEE. 2010, pp. 2883–2888 (cit. on p. 108).

[269] Joan Fontanals et al. “Integrated grasp and motion planning using independent contact regions”. In: *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2014, pp. 887–893 (cit. on p. 108).

[270] Joshua A Haustein, Kaiyu Hang, and Danica Kragic. “Integrating motion and hierarchical fingertip grasp planning”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 3439–3446 (cit. on p. 108).

[271] Anca D Dragan, Nathan D Ratliff, and Siddhartha S Srinivasa. “Manipulation planning with goal sets using constrained trajectory optimization”. In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 4582–4588 (cit. on pp. 108, 109).

[272] Thomas Weng et al. “Neural grasp distance fields for robot manipulation”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 1814–1821 (cit. on p. 108).

[273] Ioan A Sucan, Mark Moll, and Lydia E Kavraki. “The open motion planning library”. In: *IEEE Robotics & Automation Magazine* 19.4 (2012), pp. 72–82 (cit. on p. 109).

[274] Yann Labbé et al. “Megapose: 6d pose estimation of novel objects via render & compare”. In: *arXiv preprint arXiv:2212.06870* (2022) (cit. on p. 125).

[275] Xu Chen et al. “Category level object pose estimation via neural analysis-by-synthesis”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16*. Springer. 2020, pp. 139–156 (cit. on p. 125).

[276] Lin Yen-Chen et al. “inerf: Inverting neural radiance fields for pose estimation”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 1323–1330 (cit. on pp. 125, 127).

[277] Matthew Tancik et al. “Fourier features let networks learn high frequency functions in low dimensional domains”. In: *Advances in neural information processing systems* 33 (2020), pp. 7537–7547 (cit. on p. 126).

[278] Anthony Simeonov et al. “Neural descriptor fields: Se (3)-equivariant object representations for manipulation”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 6394–6400 (cit. on p. 126).

[279] Hanspeter Pfister et al. “Surfels: Surface elements as rendering primitives”. In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. 2000, pp. 335–342 (cit. on p. 129).

[280] Lee Westover. “Interactive volume rendering”. In: *Proceedings of the 1989 Chapel Hill workshop on Volume visualization*. 1989, pp. 9–16 (cit. on p. 130).

[281] Matthias Zwicker et al. “EWA splatting”. In: *IEEE Transactions on Visualization and Computer Graphics* 8.3 (2002), pp. 223–238 (cit. on pp. 130–132).

[282] Vickie Ye et al. “gsplat: An open-source library for Gaussian splatting”. In: *arXiv preprint arXiv:2409.06765* (2024) (cit. on p. 133).

[283] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014) (cit. on p. 135).

[284] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612 (cit. on p. 135).

Appendix A

Real-world Experiment Logs

A.1 Test Setup 1

This test setup uses a UR10e robot arm with a Robotiq 3F gripper and a RealSense D435i camera. The comparison of GraspVAE and GraspLDM is conducted with 6-DoF-GraspNet [89].

Object	6-DoF-GraspNet	GraspVAE-P-63C	GraspLDM-P-63C
Cleaning Spray	✓ × ✓ ✓ ×	✓ ✓ ✓ ✓ ✓	✓ ✓ ✓ ✓ ✓
Realsense Box	× ✓ × ✓ ✓	✓ × ✓ ✓ ×	✓ ✓ × ✓ ✓
Blue Cup	× ✓ × × ×	✓ ✓ × ✓ ×	✓ × × ✓ ✓
Rubber Ducky	× × ✓ × ✓	× ✓ ✓ ✓ ✓	× ✓ ✓ ✓ ✓
Pliers	× × × ✓ ✓	× ✓ ✓ ✓ ✓	× ✓ × ✓ ✓
Screwdriver	✓ × × × ×	✓ × × ✓ ✓	✓ ✓ ✓ ✓ ✓
Green Bowl	× × ✓ × ×	✓ ✓ ✓ ✓ ✓	✓ ✓ ✓ ✓ ✓
Red Marker	× × × ✓ ×	✓ ✓ ✓ × ✓	✓ ✓ ✓ ✓ ✓
Pringles Can	✓ × × × ×	✓ ✓ ✓ ✓ ×	✓ ✓ ✓ × ×
Black 3D Printed Cuboid	× × ✓ × ✓	× ✓ ✓ × ✓	✓ × ✓ ✓ ✓
Aluminum Profile	✓ × ✓ ✓ ✓	✓ ✓ ✓ ✓ ✓	✓ ✓ × × ✓
Sunscreen Bottle	× × × ✓ ✓	✓ ✓ ✓ ✓ ×	✓ ✓ ✓ × ✓
Frisbee	× × × ✓ ✓	✓ ✓ ✓ ✓ ✓	✓ ✓ ✓ ✓ ✓
Stapler	× ✓ × × ×	✓ × ✓ ✓ ×	✓ × ✓ ✓ ✓
Lint Roller	× × × ✓ ✓	✓ ✓ ✓ ✓ ×	✓ ✓ ✓ ✓ ✓
3D Printed Probe	✓ × × × ✓	× ✓ × × ✓	✓ ✓ × × ✓
Success Rate	37.5%	76.25%	80.83%

A.2 Test Setup 2

This test setup uses a Franka Research 3 robot arm with the Franka hand and a RealSense D435i camera. The comparison of GraspVAE and GraspLDM is conducted with Contact-GraspNet [90].

Object	**Contact-GraspNet**	**GraspVAE-P-63C**	**GraspLDM-P-63C**
Cleaning Spray	✓ × ✓ ✓ ✓	✓ × ✓ ✓ ×	✓ ✓ ✓ ✓ ×
Realsense Box	✓ ✓ ✓ × ✓	✓ ✓ × ✓ ×	✓ ✓ ✓ × ×
Blue Cup	✓ ✓ ✓ ✓ ✓	✓ ✓ × ✓ ×	✓ ✓ ✓ ✓ ×
Sanitizer	× × ✓ ✓ ✓	× ✓ ✓ ✓ ✓	✓ ✓ ✓ ✓ ✓
Screwdriver	✓ ✓ ✓ ✓ ×	✓ ✓ × ✓ ✓	✓ ✓ ✓ ✓ ✓
Pliers	✓ ✓ ✓ ✓ ×	✓ ✓ × ✓ ✓	✓ ✓ ✓ ✓ ×
Board Cleaner Bottle	✓ ✓ × ✓ ✓	✓ × ✓ ✓ ×	✓ × ✓ ✓ ✓
IROS tray	✓ ✓ ✓ ✓ ✓	× × ✓ × ✓	× × × ✓ ✓
Mint Box	✓ ✓ ✓ ✓ ✓	✓ ✓ ✓ ✓ ✓	✓ ✓ ✓ ✓ ✓
Gummy Bottle	× ✓ × × ✓	× ✓ ✓ ✓ ✓	✓ ✓ ✓ ✓ ✓
Staper	✓ × ✓ × ✓	✓ ✓ × ✓ ✓	✓ ✓ × ✓ ✓
DJI Osmo Holder	✓ ✓ × ✓ ×	✓ ✓ ✓ × ✓	✓ ✓ ✓ ✓ ✓
Heavy Pliers	× × × ✓ ✓	× ✓ ✓ × ✓	✓ ✓ ✓ × ✓
Rubber Ducky	× ✓ ✓ ✓ ✓	✓ × ✓ ✓ ×	✓ × ✓ ✓ ×
Marker	✓ × ✓ ✓ ✓	× ✓ × × ✓	✓ × ✓ × ✓
Book	✓ ✓ ✓ ✓ ✓	× ✓ × ✓ ✓	✓ ✓ × ✓ ×
Success Rate	76.25%	67.5%	78.75%