**A logo for a university

Description automatically generated**

*Integrated grasp and motion planning*

**Author:**

Salah Ebrahimpour

**Programme:**

MSc Artificial Intelligence with Business Strategy

**Supervisor:**

Dr. Martin Rudorfer

**Co-Supervisor:**

Dr. Anthony Henry

**Date:**

September 2024

**Abstract**

**Acknowledgement**

Contents

[Chapter 1: Introduction 1](#_Toc177287120)

[1.1. Project background 1](#_Toc177287121)

[1.2. Research Objectives 1](#_Toc177287122)

[1.3. Research Questions/Hypotheses 2](#_Toc177287123)

[1.4. Significance of the Study 3](#_Toc177287124)

[1.5. Scope and Limitations 3](#_Toc177287125)

[1.6. Thesis Structure 3](#_Toc177287126)

[Chapter 2: Literature Review 4](#_Toc177287127)

[2.1. Introduction 4](#_Toc177287128)

[2.2. Overview of Grasp Planning 4](#_Toc177287129)

[2.3. Overview of Motion Planning 6](#_Toc177287130)

[2.3. Integrated Grasp and Motion Planning 8](#_Toc177287131)

[2.4. Existing Methods and Approaches 11](#_Toc177287132)

[Summary of Existing Methods 13](#_Toc177287133)

[Gaps in the Literature 13](#_Toc177287134)

[Summary 13](#_Toc177287135)

[Chapter 3: Problem Description 14](#_Toc177287136)

[Chapter 4: Methods and Theories 14](#_Toc177287137)

[3.1. Introduction 14](#_Toc177287138)

[Research Design 14](#_Toc177287139)

[2. Experimental Setup 14](#_Toc177287140)

[2.1. Simulation Environment 14](#_Toc177287141)

[2.2. Robot and Environment Configuration 14](#_Toc177287142)

[Framework Description 14](#_Toc177287143)

[1.1. Problem Formulation 15](#_Toc177287144)

[3. Algorithms Implementation 16](#_Toc177287145)

[3.1. Description of Algorithm 16](#_Toc177287146)

[RRT\*: 16](#_Toc177287147)

[3.2. Integration with the Robot 17](#_Toc177287148)

[3.3. Evaluation Metrics 17](#_Toc177287149)

[Planning Time 17](#_Toc177287150)

[Success Rate 18](#_Toc177287151)

[3.4. Experimental Procedure 18](#_Toc177287152)

[Repeated Trials 18](#_Toc177287153)

[Varying Conditions 18](#_Toc177287154)

[Data Collection 18](#_Toc177287155)

[Chapter 5: Project Management 18](#_Toc177287156)

[Chapter 6: Evaluation 18](#_Toc177287157)

[4.1. Statistical Analysis 18](#_Toc177287158)

[4.2. Visualization 18](#_Toc177287159)

[4.3. Summary 18](#_Toc177287160)

[Chapter 7: Conclusion 19](#_Toc177287161)

[References 19](#_Toc177287162)

[Appendix 23](#_Toc177287163)

# Chapter 1: Introduction

## Project background

As robotics continues to evolve, the need for more sophisticated systems capable of autonomous, complex manipulation tasks becomes ever more pressing. Traditionally, the processes of grasp and motion planning have been treated separately in robotic systems. Grasp planning determines how a robot should grip an object based on factors like the object's shape, weight, and texture. Motion planning, on the other hand, focuses on the robot's path and the obstacles it must avoid when executing tasks (Muhayyuddin et al., 2015). However, the separation of these two processes can lead to suboptimal performance, particularly in dynamic environments where grasp configuration can affect motion feasibility and vice versa (Ali & Lee, 2020).

To address these issues, researchers have started exploring integrated approaches, where grasp and motion planning are considered simultaneously. These unified systems aim to enhance efficiency, adaptability, and reliability in robotic tasks. For example, the Grasp-RRT approach represents one such integration, optimizing both grasp and motion planning within the same framework (Ali & Lee, 2020). This research focuses on leveraging and advancing these integrated planning methods to improve robotic manipulation in real-world scenarios.

## Research Objectives

The main objective of this research is to evaluate the performance of the RRT\* algorithm in the context of integrated grasp and motion planning tasks. Specifically, this study aims to assess the algorithm's ability to solve planning tasks within benchmark scenarios introduced by Rudorfer et al., focusing on key performance metrics such as planning time and success rate.

A significant goal of the research is to compare the performance of the RRT\* algorithm against the benchmark results provided in Rudorfer et al.’s paper, particularly in relation to other algorithms such as J+-RRT and IK-RRT. This comparison seeks to identify the strengths and weaknesses of RRT\*, highlighting how well it performs in terms of runtime and success rate under the same conditions as the baseline algorithms.

Additionally, the research aims to analyze the adaptability of the RRT\* algorithm when applied to different environmental configurations, including variations in object placements and confined spaces. This will provide insights into how RRT\* handles dynamic environments and complex obstacle layouts.

Ultimately, the research seeks to contribute to the development of more efficient integrated grasp and motion planning strategies by offering insights into the applicability of the RRT\* algorithm for real-world robotic tasks. Based on the comparative analysis, this study will also provide recommendations for improving the algorithm’s efficiency and robustness in future applications.

## Research Questions/Hypotheses

This research is guided by several key questions and hypotheses, primarily focusing on the performance of the RRT\* algorithm compared to benchmark algorithms such as J+-RRT and IK-RRT, particularly in terms of planning time and success rate. The performance of RRT\* in motion planning has been widely studied, with previous research demonstrating that RRT\* is asymptotically optimal, meaning that as the number of samples increases, the solution converges to the optimal path with high probability (Gammell et al., 2015; Pérez et al., 2012). This characteristic is especially advantageous in environments with lower obstacle densities, where RRT\* can utilize its sampling efficiency to yield shorter planning times and higher success rates compared to other algorithms that may not exhibit the same level of optimality, such as J+-RRT and IK-RRT (Qureshi & Ayaz, 2015). Studies have shown that in less cluttered environments, RRT\* can outperform its predecessors in both speed and success, thanks to its iterative path refinement process (Tahir et al., 2018). Based on these findings, it is hypothesized that RRT\* will perform competitively with these benchmark algorithms in terms of planning time and success rate.

Moreover, the research aims to explore whether the RRT\* algorithm can efficiently adapt to complex and confined spaces while maintaining a high success rate. RRT\*'s incremental nature allows it to adjust dynamically to environmental changes, making it well-suited for challenging environments (Karaman et al., 2011; Lee & Lee, 2012). However, as the complexity of the space increases, it is hypothesized that the success rate of RRT\* may decline due to the increased difficulty in navigating through dense obstacles. This aligns with research indicating that while RRT\* performs well in simpler environments, its success rate can be hindered in more complex scenarios, where pathfinding becomes more convoluted (Peng et al., 2023).

In conclusion, the RRT\* algorithm shows promising performance when compared to benchmark algorithms, particularly in less cluttered environments. Its ability to adapt to complex spaces is promising, but challenges in dynamic environments highlight the need for optimization. These questions and hypotheses aim to investigate the potential of RRT\* while addressing its limitations in real-world applications.

## Significance of the Study

This study is significant in its potential to advance the field of robotics by providing valuable insights into the performance of the RRT\* algorithm in integrated grasp and motion planning. As robotic systems become more prevalent in sectors such as manufacturing, healthcare, and autonomous services, their ability to perform complex manipulation tasks in dynamic and uncertain environments is increasingly important. This research focuses on assessing the RRT\* algorithm’s effectiveness in planning both grasp and motion concurrently, particularly in terms of planning time, success rate, and adaptability.

By comparing the RRT\* algorithm to established benchmarks, this study offers critical data on its strengths and limitations when applied to real-world scenarios where both grasping and motion must be planned efficiently. These findings have broader implications for improving robotic systems' performance in dynamic environments, contributing to the development of autonomous systems that are more reliable, adaptable, and capable of handling increasingly complex tasks.

Ultimately, the outcomes of this research could enhance the performance of robotic systems in various industries, making them more versatile and effective in executing real-world tasks with greater precision and efficiency.

## Scope and Limitations

This research focuses on the development and evaluation of integrated grasp and motion planning algorithms, specifically in the context of robotic manipulation tasks. The experiments are conducted in simulated environments to ensure repeatability and control over variables. However, while the simulation allows for thorough testing under a variety of conditions, real-world testing of the algorithms may present challenges such as sensor inaccuracies, mechanical limitations of robots, or unexpected environmental factors. These limitations must be addressed in future research to fully realize the practical application of the proposed methods.

## Thesis Structure

This dissertation is organized into several chapters that systematically explore integrated grasp and motion planning. The structure is as follows:

1. **Introduction** – Provides an overview of the project, including the background, problem statement, research objectives, and significance of the study.
2. **Literature Review** – Examines existing research on grasp and motion planning, with a focus on integrated approaches, identifying gaps in the literature.
3. **Methodology** – Details the research design, experimental setup, and the algorithms implemented for testing.
4. **Evaluation**– Presents the findings from the experiments, analyzing the performance of the integrated algorithms and comparing them with traditional methods.
5. **Project Management** –
6. **Conclusion** – Summarizes the key findings, contributions to the field, and offers recommendations for future research.

# Chapter 2: Literature Review

## 2.1. Introduction

The field of robotics has seen significant advancements in recent years, particularly in the domain of integrated grasp and motion planning. While motion planning is a fundamental aspect of robotics (Elbanhawi & Simić, 2014), the simultaneous planning of grasping an object and the robot's motion remains a notable challenge (Bütepage et al., 2019). This challenge is further intensified by the need to overcome computational obstacles related to sensing, grasp analysis, motion planning, and the execution of the robot arm's movements (Ichnowski et al., 2020).

This literature review aims to provide a comprehensive overview of the current state of knowledge in this domain, and highlights key concepts, methodologies, and findings from previous studies. Integrated grasp and motion planning involves the simultaneous consideration of how a robot should grasp an object and how it should move both the robot and the object to achieve a desired outcome. This dual planning approach is essential for enhancing the efficiency and effectiveness of robotic operations in complex environments.

In this section, we will explore the historical development and theoretical foundations of grasp planning and motion planning, examining the major theories and models that have shaped these fields. We will then explore the integrated approaches that combine these two planning processes and will discuss the challenges and innovations that have emerged from this integration. By critically analyzing the existing literature, we aim to identify gaps and areas for future research, setting the stage for the subsequent chapters of this dissertation. This review will provide the necessary context and background for understanding the contributions and implications of our research on integrated grasp and motion planning.

## 2.2. Overview of Grasp Planning

Grasp planning in robotics has advanced significantly, evolving from basic algorithms for simple objects in controlled environments to more sophisticated approaches capable of handling complex and irregularly shaped objects in dynamic settings. Initially, the focus was on convex objects that a single robot could grasp, aiming to optimize stability and grasp force on the robot's contact points (Zafra-Urrea, 2023). As computational power and sensor technology progressed, research expanded to include the manipulation of objects with robotic hands in dynamic conditions, that lead to the development of various grasp planning algorithms to identify stable grasps (Dharbaneshwer et al., 2020).

Task-oriented grasp planning algorithms have been introduced to compute the most suitable grasp for a given task based on a simplified geometrical and structural description of the object and the task requirements (Prats et al., 2007). Recent advancements have also explored the use of over-segmented meshes and relational databases to improve grasp planning, particularly in regrasp planning scenarios (Wan & Harada, 2017). Precision grasp planning for multi-fingered hands involves analyzing grasp stability, synthesis, and object representation (Yan et al., 2019).

Research by Bertoni et al. (2021) introduces a generic grasp planning pipeline that enables transparent and generic grasp planning procedures, automating grasping actions irrespective of the end-effector kinematic structure. Tsuji et al. (2010) demonstrate grasp planning for multi-fingered hands with humanoid robots, emphasizing the importance of effective grasp planning through simulation and experimental results. Rubert et al. (2017) focus on characterizing grasp quality metrics, highlighting their significant role in the analytical approach to grasp planning.

These references collectively emphasize the critical role of grasp planning in robotics and will showcase advancements in methodologies and technologies to optimize grasp points, stability, and quality metrics for efficient and effective robotic interactions with objects.

**Major Theories and Models**

Several theories and models underpin grasp planning:

* **Force Closure**: Ensures that the robot’s grip can resist external forces from any direction, and can provide a stable grasp.
* **Form Closure**: Achieves stability by constraining the object’s movement through the geometric arrangement of contact points.
* **Task-Oriented Grasping**: Focuses on optimizing the grasp for the specific task the robot is intended to perform, considering factors like object manipulation and placement.

**Techniques and Algorithms**

Grasp planning techniques and algorithms have evolved significantly. Some of the prominent methods include:

* **Analytical Approaches**: Use mathematical models to predict and evaluate potential grasps based on object geometry and physical properties.
* **Sampling-Based Methods**: Generate a large number of potential grasps and evaluate them based on predefined criteria to select the best option.
* **Machine Learning Approaches**: Leverage large datasets and learning algorithms to train models that can predict successful grasps based on visual and tactile data.

**Recent Advances**

Recent research in grasp planning has focused on enhancing the robustness and adaptability of robotic grasps. Key advancements include:

* **Deep Learning Models**: These models use neural networks to learn complex grasping strategies from vast amounts of data, improving the robot's ability to generalize across different objects.
* **Sensor Integration**: Combining data from multiple sensors, such as cameras and tactile sensors, to provide a more comprehensive understanding of the object and its environment.
* **Real-Time Grasp Planning**: Developing algorithms that can operate in real-time, allowing robots to adapt to changes in the environment and object positioning dynamically.

**Challenges and Future Directions**

Despite significant progress, grasp planning remains a challenging problem due to:

* **Object Diversity**: The wide variety of objects in terms of shape, size, and material makes it difficult to develop a one-size-fits-all grasping solution.
* **Dynamic Environments**: Robots must be able to adapt their grasps in real-time to changes in the environment or object position.
* **Computational Efficiency**: Ensuring that grasp planning algorithms can operate quickly enough for real-time applications remains a significant hurdle.

Future research is likely to focus on further integrating machine learning approaches with traditional methods, enhancing the robot's ability to learn from experience and adapt to new situations. Additionally, improving sensor technology and computational efficiency will be critical for advancing the field of grasp planning.

## 2.3. Overview of Motion Planning

Motion planning is a crucial technology in robotics that involves breaking down complex motion tasks into a series of discrete actions that can be executed (Fan, 2023). It plays a vital role in various applications such as mobile robots, autonomous driving, automated vehicles, and free-floating space robots (Liniger & Gool, 2020; Zhang & Zhu, 2020; Manzinger et al., 2021). The development of motion planning algorithms has led to the creation of efficient methods like Rapidly-exploring Random Trees (RRT), optimal RRT\*, and Potentially guided-RRT\* (P-RRT\*) (Qureshi et al., 2019). These algorithms aim to enhance the safety and autonomy of robots in their operational environments.

Integrated Task and Motion Planning (TAMP) addresses the challenge of planning for robots in environments with numerous objects, requiring actions not only for self-movement but also for object manipulation (Garrett et al., 2021). This field extends traditional task and motion planning concepts to handle complex scenarios effectively. Additionally, deep learning-based approaches have been explored for motion planning in self-driving vehicles, covering behavior planning, trajectory planning, and End-to-End Learning (E2EL) (Ganesan, 2024).

Various paradigms have been investigated in motion planning, such as the sampling approach, which involves generating collision-free configurations randomly in free space to create a roadmap (Dalibard & Laumond, 2009). This approach has been successful in addressing motion planning problems efficiently. Moreover, motion planning is not limited to traditional robotics but extends to applications like humanoid robots in complex environments (Li & Huang, 2007).

#### Key Concepts and Definitions

Motion planning involves determining a feasible path or sequence of movements that a robot must follow to achieve a specific goal while avoiding obstacles. Key concepts include:

* **Configuration Space (C-Space)**: A representation of all possible positions and orientations of the robot. Each point in this space corresponds to a unique state of the robot.
* **Path Planning**: The process of finding a collision-free path from a start configuration to a goal configuration within the configuration space.
* **Trajectory Planning**: Extends path planning by considering the timing and dynamics of the robot’s movements, ensuring smooth and feasible trajectories.

#### Major Theories and Models

Several theories and models are central to motion planning:

* **Graph-Based Methods**: These include grid-based approaches and roadmaps, such as Probabilistic Roadmaps (PRM) and Rapidly-exploring Random Trees (RRT), which represent the environment as a graph and use search algorithms to find paths.
* **Optimization-Based Methods**: These methods formulate motion planning as an optimization problem, where the goal is to minimize a cost function that could represent distance, time, energy, or other criteria.
* **Sampling-Based Methods**: These approaches generate random samples in the configuration space to construct feasible paths, providing solutions where deterministic methods might struggle.

#### Techniques and Algorithms

Motion planning has seen numerous techniques and algorithms developed over the years, including:

* **Probabilistic Roadmaps (PRM)**: Constructs a roadmap by randomly sampling the configuration space and connecting these samples to form a graph. Paths are then found by searching this graph.
* **Rapidly-exploring Random Trees (RRT)**: Builds a tree by incrementally expanding random samples from the configuration space, aiming to rapidly explore large spaces.
* A Algorithm\*: A graph search algorithm that finds the shortest path by combining the cost to reach a node and the estimated cost to reach the goal.

#### Recent Advances

Recent advancements in motion planning focus on improving efficiency, robustness, and adaptability. Key developments include:

* **Multi-Robot Motion Planning**: Addressing the complexities of coordinating multiple robots, including collision avoidance and cooperative task execution.
* **Real-Time Motion Planning**: Developing algorithms capable of operating in real-time, enabling robots to adapt their paths dynamically to changes in the environment.
* **Machine Learning Integration**: Utilizing machine learning to predict and optimize paths based on prior experience and data, enhancing the robot's ability to navigate complex environments.

#### Challenges and Future Directions

Despite significant progress, motion planning continues to face several challenges:

* **High-Dimensional Spaces**: Robots with many degrees of freedom, such as humanoid robots, present significant challenges due to the complexity of their configuration spaces.
* **Dynamic and Uncertain Environments**: Ensuring robust and safe navigation in environments that change over time or have uncertain elements is a major ongoing challenge.
* **Computational Demands**: Developing algorithms that are both efficient and capable of handling complex planning tasks in real-time remains a critical focus.

Future research in motion planning is likely to emphasize further integration of AI and machine learning, enabling more intelligent and adaptable planning strategies. Advances in sensor technology and computational power will also play a crucial role in overcoming current limitations and enhancing the capabilities of motion planning systems.

## 2.3. Integrated Grasp and Motion Planning

Integrated grasp and motion planning is a crucial aspect of robotics that involves simultaneously determining how a robot should grasp an object and how it should move both the robot and the object to achieve a desired outcome. This integrated approach is essential for enhancing the efficiency and effectiveness of robotic operations, particularly in complex and dynamic environments (Dang-Vu et al., 2015; Rosell et al., 2019). By combining grasp and motion planning, robots can perform tasks more seamlessly, reducing the time and computational resources required for separate planning stages (Ichnowski et al., 2020; Vahrenkamp et al., 2010).

The integration of grasp and motion planning algorithms allows for the generation of collision-free trajectories to grasps or grasp sets that are precomputed or synthesized during the planning process (Ichnowski et al., 2020). This holistic approach enables robots to decide on the best grasp for an object and plan a collision-free path that facilitates the successful execution of the task (Dang-Vu et al., 2015). Additionally, the synthesis of grasps and motions involves generating potential grasps for an object and planning motions using efficient planners that guide the motion planning process within a reduced search space, resulting in paths with human-like appearances (Rosell et al., 2019).

Moreover, the exploration of deep learning techniques has shown promise in accelerating grasp-optimized motion planning, demonstrating the potential for advanced technologies to further enhance integrated grasp and motion planning processes (Ichnowski et al., 2020). For example, the proposed Grasp-RRT planner combines tasks necessary for grasping an object, such as finding a feasible grasp, solving inverse kinematics, and searching for a collision-free trajectory to reach the grasping pose (Vahrenkamp et al., 2010).

#### Challenges in Integration

The integration of grasp and motion planning presents several challenges:

* **Complexity of Combined Planning**: Coordinating both grasp and motion planning increases the complexity of the problem, requiring advanced algorithms that can handle high-dimensional spaces and dynamic constraints.
* **Real-Time Constraints**: Achieving real-time performance is crucial for practical applications, necessitating efficient algorithms that can quickly generate feasible and optimal plans.
* **Sensor and Data Integration**: Effective integration requires robust sensor data processing to accurately perceive the environment and adapt plans accordingly.

#### Existing Integrated Approaches

Several approaches have been developed to address the challenges of integrated grasp and motion planning:

* **Sequential Planning**: Involves planning the grasp first, followed by motion planning. While simpler to implement, this approach can lead to suboptimal solutions as the grasp is not optimized for the subsequent motion.
* **Simultaneous Planning**: Simultaneously plans both grasp and motion, considering the interdependencies between the two. This approach can generate more optimal solutions but is computationally more demanding.
* **Hierarchical Planning**: Decomposes the problem into hierarchical levels, planning high-level tasks first and refining them into detailed plans. This can balance between optimality and computational efficiency.

#### Techniques and Algorithms

Several techniques and algorithms have been proposed for integrated grasp and motion planning:

* **Optimization-Based Methods**: Formulate the problem as a single optimization task, minimizing a cost function that includes both grasp and motion components.
* **Sampling-Based Methods**: Extend traditional sampling-based motion planning algorithms, like RRT and PRM, to consider grasp planning simultaneously.
* **Machine Learning Approaches**: Use machine learning models to predict feasible grasps and motions based on prior experience, improving planning speed and adaptability.

#### Recent Advances

Recent research in integrated grasp and motion planning has focused on improving the efficiency and robustness of planning algorithms:

* **Deep Reinforcement Learning**: Applying deep reinforcement learning to learn integrated planning strategies from large datasets, enabling robots to handle more complex tasks with greater autonomy.
* **Real-Time Adaptive Planning**: Developing algorithms that can adapt plans in real-time based on changes in the environment and object dynamics, enhancing the robot's ability to operate in unpredictable settings.
* **Sensor Fusion Techniques**: Combining data from multiple sensors, such as cameras, LIDAR, and tactile sensors, to provide a more comprehensive understanding of the environment and improve planning accuracy.

#### Applications and Implications

Integrated grasp and motion planning has numerous applications across various domains:

* **Industrial Automation**: Enhancing the efficiency and flexibility of robotic systems in manufacturing and assembly lines.
* **Service Robotics**: Improving the capability of robots to perform household tasks, assistive functions, and other service-oriented activities.
* **Medical Robotics**: Enabling precise and adaptable robotic systems for surgical procedures and rehabilitation.

#### Challenges and Future Directions

Despite significant progress, integrated grasp and motion planning continues to face several challenges:

* **Scalability**: Ensuring that planning algorithms can scale to handle more complex tasks and environments.
* **Robustness**: Developing methods that can handle uncertainties and variations in the environment and object properties.
* **User Interaction**: Creating intuitive interfaces and control mechanisms that allow users to easily specify and adjust tasks for robotic systems.

## 2.4. Existing Methods and Approaches

In this section, we review the existing methods and approaches employed in integrated grasp and motion planning. The integration of these two domains is critical for achieving efficient and seamless robotic manipulation, allowing robots to simultaneously plan how to grasp an object and how to move it in space without collisions. A comprehensive understanding of both grasp and motion planning techniques is necessary to explore how they can be effectively integrated.

#### 1. Sequential Approaches

Sequential approaches to integrated grasp and motion planning typically involve two distinct stages: first, a stable grasp is planned, and then a collision-free motion is computed for the robot arm to reach the grasped object. These methods are easier to implement since they decouple the problem, but they can lead to suboptimal solutions because the motion is not considered during the grasp planning phase. Examples include the work of Wan and Harada (2017), who developed regrasp planning techniques based on precomputed grasps.

**Limitations:**

* Potential suboptimality due to the lack of coordination between grasp and motion planning.
* Time inefficiency because the motion may need to be replanned if the initial grasp is not feasible in the context of the robot's movements.

#### 2. Simultaneous Approaches

Simultaneous approaches attempt to overcome the limitations of sequential methods by planning grasp and motion simultaneously. These approaches aim to consider the interdependencies between the grasp configuration and the subsequent motion trajectory. For example, Vahrenkamp et al. (2010) proposed an integrated grasp and motion planning approach using Rapidly-exploring Random Trees (RRT), where the grasp and the motion trajectory are planned together in a unified framework.

**Advantages:**

* Better coordination between grasp and motion, leading to more optimal solutions.
* Reduced need for re-planning, which increases overall efficiency.

**Challenges:**

* Computational complexity is higher due to the expanded search space.
* The need for real-time planning increases the difficulty of implementation in dynamic environments.

#### 3. Optimization-Based Methods

Optimization-based methods treat the problem of integrated grasp and motion planning as a single optimization problem. These methods minimize a cost function that takes into account factors such as grasp stability, path length, energy consumption, and collision avoidance. Examples include the work of Dang-Vu et al. (2016), who used optimization techniques to balance grasp and motion objectives.

**Advantages:**

* Allows for a more holistic solution that balances multiple factors such as grasp stability, energy efficiency, and safety.
* Provides a flexible framework for incorporating additional constraints, such as dynamic obstacles or varying object properties.

**Challenges:**

* Computationally expensive due to the need to solve complex optimization problems.
* Requires careful tuning of the cost function to achieve desired outcomes in real-world scenarios.

#### 4. Machine Learning-Based Approaches

Recent advances in machine learning, particularly deep learning, have shown promise in accelerating the integrated planning process. These approaches use large datasets to train models that predict both grasp points and motion trajectories. Ichnowski et al. (2020) demonstrated how deep reinforcement learning could be applied to grasp-optimized motion planning, where the robot learns to generate grasp and motion plans based on previous experience.

**Advantages:**

* Significantly reduces the computational time needed for planning by using pre-learned models.
* Can adapt to new environments and objects more efficiently by leveraging prior knowledge.

**Challenges:**

* Requires large datasets for training, which can be difficult to obtain for complex manipulation tasks.
* Generalization to novel objects or scenarios may be limited depending on the quality of the training data.

#### 5. Hierarchical Approaches

Hierarchical methods decompose the problem into different levels, such as high-level task planning followed by low-level motion and grasp planning. This approach allows for a more structured solution to integrated planning and helps manage the complexity of the combined problem. For example, Bertoni et al. (2021) proposed a hierarchical grasp planning pipeline that provides high-level task planning followed by detailed grasp and motion planning stages.

**Advantages:**

* Reduces the complexity of the problem by breaking it down into manageable sub-problems.
* Offers a balance between solution quality and computational efficiency.

**Challenges:**

* Hierarchical decomposition can lead to suboptimal solutions if the high-level decisions do not adequately account for low-level constraints.

### Summary of Existing Methods

The reviewed methods provide a variety of strategies for integrating grasp and motion planning in robotics. Sequential approaches offer simplicity but may result in suboptimal solutions, while simultaneous and optimization-based methods provide more optimal solutions at the cost of increased computational complexity. Machine learning and hierarchical approaches present innovative ways to address the challenges of integrated planning, but they come with their own set of limitations. Future research is likely to focus on improving computational efficiency, scalability, and real-time adaptability of these methods, particularly in dynamic and unpredictable environments.

### Gaps in the Literature

Despite the significant advancements in integrated grasp and motion planning, there are several gaps in the existing literature:

* **Scalability**: Most methods struggle to scale efficiently to high-dimensional tasks or environments with numerous dynamic obstacles.
* **Real-time adaptability**: Current algorithms are often computationally expensive and may not be suitable for real-time applications.
* **Generalization to novel tasks**: Many approaches rely on precomputed data or assumptions about the environment, limiting their applicability in scenarios involving unfamiliar objects or dynamic changes.

Addressing these gaps will be crucial for advancing the field of robotics, particularly in applications requiring real-time performance and adaptability.

## Summary

# 

# Chapter 3: Problem Description

#### 3.1 Overview of Integrated Grasp and Motion Planning

Robotic systems face significant challenges when tasked with grasping and manipulating objects in dynamic environments. Traditionally, grasp planning and motion planning have been handled separately, which can lead to inefficiencies and suboptimal performance. In many scenarios, the interaction between the grasp and the robot’s motion path is critical for achieving smooth and efficient task execution. Integrated grasp and motion planning approaches seek to resolve this by simultaneously planning how the robot grasps an object and how it moves while avoiding obstacles and maintaining task feasibility.

This research addresses a specific challenge in the domain of integrated grasp and motion planning. Specifically, it focuses on evaluating different path planning algorithms to determine which methods can best optimize both the robot’s grasp and motion, particularly in constrained or complex environments.

#### 3.2 Problem Statement

The core problem this research addresses is the need for efficient and reliable integrated grasp and motion planning algorithms for robotic manipulators. This is especially challenging when operating in environments with varying object configurations and dynamic constraints. The primary question driving this research is how different algorithms, particularly RRT\*, perform in solving these planning tasks, compared to other established algorithms like JPlusRRT, IKRRT, and BIKRRT.

In this context, the problem can be broken down into the following components:

1. **Grasp Selection**: The robot must determine a stable and feasible grasp configuration for each object in the workspace. This involves analyzing possible grasp poses and selecting one that maximizes stability while being feasible in terms of the robot's kinematics.
2. **Motion Planning**: Once a grasp is selected, the robot must plan a path from its initial configuration to the object, ensuring that it avoids obstacles and operates within its physical limitations.
3. **Dynamic Environments**: In real-world applications, the environment may change dynamically, making it necessary for the algorithms to adapt to these changes while maintaining efficiency and success in task completion.

The objective of this research is to evaluate how well the RRT\*, JPlusRRT, IKRRT, and BIKRRT algorithms handle integrated grasp and motion planning, particularly in complex environments with diverse object placements and obstacles.

#### 3.3 Research Gap

While several algorithms for motion planning exist, such as Rapidly-exploring Random Trees (RRT) and its optimal variant RRT\*, there is a gap in understanding their specific performance in integrated planning tasks. Although some studies have implemented grasp planning in static or simple environments, fewer have addressed the complexity of environments where both grasp and motion planning need to be tightly integrated. Additionally, while the benchmark scenarios from Rudorfer et al. provide valuable insights into the performance of algorithms like JPlusRRT and IK-RRT, these scenarios have not been extensively compared with RRT\* in terms of integrated planning.

#### 3.4 Specific Challenges

The specific challenges addressed by this dissertation are as follows:

1. **High-dimensional Search Space**: The configuration space of robotic manipulators, especially those with six or more degrees of freedom, is vast and difficult to search efficiently. The challenge is to evaluate how RRT\* and other algorithms manage this space while considering both grasping and motion.
2. **Dynamic and Complex Environments**: Real-world environments are often cluttered and dynamic, which makes the planning process even more difficult. This research tests the robustness of the algorithms in these environments, particularly those with confined spaces and irregular object arrangements.
3. **Algorithm Performance**: The goal is to evaluate the trade-offs between computational efficiency (planning time), success rates (finding a feasible path), and the quality of the paths generated by each algorithm.
4. **Practical Implementation**: While many algorithms have theoretical benefits, practical implementation often reveals constraints or bottlenecks that were not anticipated. By implementing these algorithms in the PyBullet environment and the Jogramop framework, this research aims to identify these practical challenges.

#### 3.5 Importance of the Problem

The success of integrated grasp and motion planning is crucial in advancing the capabilities of robotic systems. Robots are increasingly being deployed in industries such as manufacturing, healthcare, and service sectors, where they need to handle complex tasks with high reliability. The findings of this research will contribute to improving the performance of robotic systems, making them more adaptable and efficient in real-world applications.

#### 3.6 Summary

This chapter outlines the core problem that this dissertation addresses: the challenge of efficient and robust integrated grasp and motion planning in dynamic environments. The goal is to evaluate several algorithms, with a particular focus on RRT\*, to determine their strengths and limitations in handling complex robotic manipulation tasks. The following chapters will detail the methodology and experimental setup used to tackle this problem and present the findings from the evaluations of the different algorithms.

# Chapter 4: Methods and Theories

## 3.1. Introduction

This section describes the methodology used to compare four path planning algorithms—RRT\*, IKRRT, IKRRTOptimized, and JPlusRRT—in terms of path length, planning time, and success rate. The comparison aims to identify the strengths and weaknesses of each algorithm in different robotic path planning scenarios.

## Research Design

## 2. Experimental Setup

### 2.1. Simulation Environment

### 2.2. Robot and Environment Configuration

### Framework Description

This section outlines the framework used for evaluating the performance of the RRT\* algorithm in a benchmark environment specifically designed for joint grasp and motion planning. The environment, described by Rudorfer et al. (2023)​(jogramop\_framework4romo…), provides 20 benchmark scenarios involving confined spaces and complex object placements, making it suitable for testing robotic planning algorithms in challenging conditions.

#### Overview

The benchmark environment comprises a series of scenarios that progressively increase in difficulty. These scenarios involve tasks such as navigating through narrow gaps, reaching objects in cluttered environments, and grasping targets located in constrained spaces. The goal is for the robot, a 6-DOF manipulator with a gripper, to reach and successfully grasp a target object while avoiding obstacles. Precomputed grasp candidates for each object are provided, ensuring that the same grasp options are used across trials.

The RRT\* algorithm is implemented in this environment to optimize both motion planning and grasp selection. The framework allows for the evaluation of the algorithm's performance by comparing results such as runtime and success rate with the baseline results provided in the paper.

#### RRT\* Algorithm Implementation

The RRT\* algorithm is a sampling-based planner designed to find optimal paths by exploring the configuration space incrementally. In this implementation, the robot starts from an initial configuration, and the algorithm builds a tree of potential configurations to find a collision-free path to the target object.

Key components of the RRT\* implementation include:

* **Steering**: The algorithm guides the robot toward randomly sampled configurations, ensuring progress toward the goal while respecting the step size η\etaη.
* **Nearest Neighbor and Rewiring**: The algorithm rewires nodes to optimize the path, reducing travel distance while maintaining feasibility.
* **Collision Detection**: Collision checks are performed at each step using the PyBullet physics engine, ensuring the robot avoids obstacles and adheres to workspace constraints.
* **Grasp and Motion Integration**: The algorithm considers precomputed grasp candidates to ensure feasible grasp configurations while simultaneously planning the robot’s motion.

#### Environment Setup

The benchmark scenarios described in Rudorfer et al. (2023)​(jogramop\_framework4romo…)are adapted for this research. These scenarios test the robot’s ability to navigate confined spaces while performing grasp tasks. The scenarios include various objects, each with a set of precomputed grasp candidates, and increasingly challenging obstacles that require precise navigation and motion planning.

The robot is initialized in a fixed configuration at the beginning of each scenario, and the task is to plan a feasible path and reach the goal configuration without collisions. This environment provides a comprehensive test bed for evaluating RRT\*'s performance in integrated grasp and motion planning.

#### Comparison with Baseline Results

Instead of directly testing against other algorithms, the performance of the RRT\* algorithm is compared with the baseline results reported in the paper​(jogramop\_framework4romo…). The metrics used for comparison include:

* **Planning time**: The total time taken by RRT\* to compute a valid path.
* **Success rate**: The percentage of successful runs where the robot reaches the target without collisions.

These results are compared to the baseline algorithms from the paper, which include J+-RRT and IK-RRT, to evaluate RRT\*'s effectiveness in terms of runtime and success rate. This comparison allows for a direct assessment of how well RRT\* performs relative to established algorithms under the same benchmark scenarios.

## Problem Formulation

The problem addressed in this research involves the integrated grasp and motion planning for a robotic manipulator, with a focus on comparing algorithmic performance in benchmark scenarios. The goal is to evaluate how different algorithms manage both grasp selection and motion planning under various conditions.

In this context, we consider a set of potential grasps, where each grasp consists of a grasp pose representing the 6-DoF position and orientation of the gripper relative to the object.

The motion planning component involves searching the robot's configuration space , where each configuration represents a particular set of joint angles for the manipulator. The collision-free subset of the configuration space is denoted as . The objective for the motion planner is to find a path from an initial configuration to a goal configuration , ensuring that the chosen grasp ​ is reached while minimizing path length, avoiding obstacles, and adhering to the robot's kinematic constraints (Rudorfer et al., 2024).

To compare algorithms, we set up benchmark scenarios involving a variety of objects, grasps, and environmental configurations. The goal is to determine how well each algorithm performs in terms of path planning efficiency, grasp stability, and adaptability to dynamic changes in the environment. Performance metrics such as planning time, path length, and success rate will be used to evaluate the effectiveness of each algorithm in real-time simulations.

## 3. Algorithms Implementation

## 3.1. Description of Algorithm

### RRT\*:

The Rapidly exploring Random Tree Star (RRT)\* algorithm is an optimized version of the original RRT algorithm, designed to incrementally build a tree structure in the configuration space and find collision-free paths in real-time. RRT\* enhances the efficiency and quality of the paths by incorporating two key optimizations: cost minimization and rewiring.

#### RRT Overview

The original **RRT** algorithm operates by randomly generating points in the configuration space and connecting each new point (vertex) to the closest existing node in the tree. Each newly generated vertex must be checked to ensure it does not lie within an obstacle. Additionally, the path connecting the new vertex to its nearest neighbor must also be collision-free. The algorithm terminates when a vertex is placed within a predefined goal region or after a set number of iterations.

#### RRT Algorithm Process:

1. **Generate a Random Position**: A new point XnewX\_{\text{new}}Xnew​ is randomly sampled from the configuration space.
2. **Obstacle Checking**: If XnewX\_{\text{new}}Xnew​ lies within an obstacle, the process continues to generate a new random point.
3. **Nearest Neighbor Search**: The algorithm finds the closest vertex in the current tree to XnewX\_{\text{new}}Xnew​.
4. **Linking and Chaining**: The new point is connected to the nearest node if the path between them is free from obstacles.
5. **Goal Checking**: If the newly added point is in the goal region, the algorithm returns the path; otherwise, it continues iterating until the limit is reached.

Although RRT is quick to implement and relatively efficient, it has a significant limitation in that it produces suboptimal, "cubic" paths due to the nearest-neighbor connection process. These paths tend to take longer, as they do not always follow the most direct route, resulting in non-optimal navigation across the configuration space. RRT\* was developed to address these limitations.

#### RRT\* Overview

The **RRT**\* algorithm builds on the core functionality of RRT by introducing two optimizations: cost-based minimization and rewiring, both of which enhance the quality of the paths generated.

1. **Cost Minimization**: In RRT\*, each vertex stores the total cost (distance) from the starting point. After generating a random point, the algorithm examines not only the closest node but also other nodes within a certain radius around the new vertex. If a node within this neighborhood provides a lower cost, the new vertex is connected to this node instead of the nearest one, which reduces the overall path length.
2. **Rewiring**: After connecting the new vertex to the optimal node, RRT\* re-examines the nearby nodes to see if rewiring them through the newly added node would reduce their cost. If so, the algorithm rewires those nodes to improve their overall path cost. This step ensures smoother, more direct paths by iteratively refining the tree.

#### RRT\* Algorithm Process:

1. **Generate a Random Position**: A new point XnewX\_{\text{new}}Xnew​ is randomly sampled from the configuration space.
2. **Obstacle Checking**: If XnewX\_{\text{new}}Xnew​ lies within an obstacle, it is discarded, and a new point is generated.
3. **Nearest Neighbor Search**: The nearest vertex in the tree to XnewX\_{\text{new}}Xnew​ is identified, and a fixed radius around XnewX\_{\text{new}}Xnew​ is used to find neighboring vertices.
4. **Cost Calculation**: For each neighboring vertex, the cost (total distance from the start) is calculated. The new vertex is connected to the node with the lowest cost.
5. **Rewiring**: After adding XnewX\_{\text{new}}Xnew​ to the tree, the neighboring nodes are examined again to check if rewiring them through XnewX\_{\text{new}}Xnew​ would reduce their cost. If so, they are rewired to improve the overall tree structure.
6. **Goal Checking**: The algorithm checks if the new node reaches the goal region. If not, it continues to iterate.

#### Key Features of RRT\*:

* **Asymptotic Optimality**: As the number of samples increases, RRT\* guarantees that the path found converges to the optimal solution.
* **Probabilistic Completeness**: Like RRT, RRT\* ensures that a solution will be found if one exists, provided enough time and samples.
* **Path Refinement**: RRT\* constantly refines the path by rewiring nodes to optimize the overall path length, resulting in much smoother, more direct paths compared to RRT.

#### Important Parameters

* **γ (Gamma)**: A scaling factor used to determine the radius for rewiring connections. It ensures that the rewiring process considers a sufficient number of nearby nodes.
* **η (Eta)**: The maximum step size that the algorithm can move towards the random sample in one iteration.

#### Pseudo-Code for RRT\*:

python

Copy code

Rad = r

G(V,E) # Graph containing edges and vertices

For iteration in range(0, n):

Xnew = RandomPosition()

If Obstacle(Xnew) == True, try again

Xnearest = Nearest(G(V,E), Xnew)

Cost(Xnew) = Distance(Xnew, Xnearest)

Xbest, Xneighbors = findNeighbors(G(V,E), Xnew, Rad)

Link = Chain(Xnew, Xbest)

For x’ in Xneighbors:

If Cost(Xnew) + Distance(Xnew, x’) < Cost(x’):

Cost(x’) = Cost(Xnew) + Distance(Xnew, x’)

Parent(x’) = Xnew

G += {Xnew, x’}

G += Link

Return G

#### Performance Considerations:

While RRT\* generates more optimal paths than RRT, it comes at the cost of increased computation time. The process of examining neighboring nodes and rewiring them can be computationally expensive, especially in complex environments with many obstacles. In environments where obstacle avoidance is critical, the number of checks required significantly increases the algorithm’s running time.

Despite the added computational load, the benefits of RRT\* in generating smoother, shorter paths, particularly in dense environments with obstacles, make it a preferred choice for tasks requiring optimal navigation and pathfinding.

RRT\* algorithm is an incremental sampling-based method which is designed to ensure both probabilistic completeness and asymptotic optimality. This algorithm incrementally builds a tree of feasible trajectories rooted at the initial state, exploring the Configuration space in a manner that tends towards the goal configuration while optimizing the path. The following details highlight the key features and process of the RRT\* algorithm as discussed in the referenced paper by Karaman and Frazzoli (Karaman and Frazzoli, 2011).

## 3.2. Integration with the Robot

Each algorithm was adapted to work with the 6-DOF manipulator. Modifications were made to ensure compatibility with the robot’s kinematics and dynamics, and to handle the specific constraints of the simulation environment.

## 3.3. Evaluation Metrics

### Planning Time

Planning time is recorded as the time taken from the start of the algorithm until a valid path is found. Timing functions in Python (e.g., time.time()) were used to measure this metric.

### Success Rate

Success rate is calculated as the percentage of trials in which an algorithm successfully finds a valid path within a predefined time limit (e.g., 60 seconds).

## 3.4. Experimental Procedure

### Repeated Trials

Each algorithm was tested in 1000 trials to ensure statistical significance. Each trial involves finding a path from a given start to a goal position with varying obstacle configurations.

### Varying Conditions

Experiments were conducted under varying conditions, including different start and goal positions and different numbers of obstacles in different environments. This helps in understanding the robustness and adaptability of each algorithm.

### Data Collection

Data on planning time, and success rate were collected during each trial. Automated scripts were used to log these metrics for subsequent analysis.

# Chapter 5: Project Management

Effective project management was crucial to the successful implementation of my dissertation, which involved coding, testing, and comparing multiple path-planning algorithms in simulated environments. The project was managed by setting clear milestones, tracking progress, prioritizing tasks, and adapting to challenges that arose during the process.

#### Timeline and Milestones

I began by defining specific milestones to guide the progress of the project. Initially, I implemented a simple RRT algorithm in a PyBullet simulation environment using a 6-DOF robot, the Franka Panda robot. As I gained more familiarity with the environment, I expanded to implementing advanced algorithms such as RRT\*, JPlusRRT, IKRRT, and BIKRRT. These milestones helped to keep the project structured, with a clear focus on building and testing the algorithms sequentially.

Later, I set up the jogramop\_framework environment developed by Rudorfer et al., which included 20 benchmark scenarios. While the original goal was to implement multiple algorithms in this environment, due to time constraints, only RRT\* was fully implemented and compared with the benchmark results. Each phase of the project was organized and executed according to a timeline, and the overall progress was tracked to ensure timely completion of key tasks.

#### Task Management

The project was divided into three key areas: coding, testing, and evaluation of algorithms. The first priority was to implement the RRT\* algorithm and its variations. Once the core algorithms were in place, I focused on testing their performance in different environments, specifically using the benchmark scenarios from the jogramop framework.

Given the complexity of the task, coding and algorithm implementation took the majority of my time. Testing and evaluation followed naturally, with each iteration requiring detailed analysis and adjustments to ensure the algorithm's accuracy and performance. Task management followed an agile methodology, where iterative development cycles allowed for frequent testing and feedback.

#### Resource Management

I relied on PyBullet for simulations and used the jogramop framework for comparing RRT\* with other algorithms in benchmark scenarios. GitHub was utilized for version control, while Trello was employed for project management and task tracking. Weekly meetings with my supervisor provided valuable guidance, helping me to resolve issues related to environment setup, algorithm implementation, and scenario testing.

The main limitation was the lack of comprehensive online resources for working with specific environments and algorithms, which added complexity to the project. Despite these challenges, having the jogramop framework, which provided ready-made environments, significantly improved the process, allowing me to focus on algorithm implementation and performance testing.

#### Risk Management

There were several risks and challenges during the project, particularly with algorithm performance and simulation failures. Setting up the simulation environment and ensuring compatibility with each algorithm required significant effort. Some algorithms did not perform as expected in certain environments, leading to several iterations of debugging and refinement.

To mitigate these risks, I reduced the number of algorithms tested in the new scenarios and focused on the performance of RRT\* in the jogramop framework. This allowed me to maintain the quality and depth of the analysis while still comparing the algorithm against benchmark scenarios. My supervisor’s guidance was instrumental in navigating these challenges.

#### Time Management

The most time-intensive phase of the project was coding and implementing the algorithms, which required careful attention to detail and frequent adjustments. I had initially planned to implement multiple algorithms in the new scenarios, but time constraints only allowed for the completion of RRT\*. As a result, I focused on ensuring that the RRT\* algorithm was thoroughly tested and compared to the benchmark scenarios in terms of runtime and success rate.

Despite these delays, the project was managed effectively, with time allocated to each phase based on its complexity and importance. Weekly progress tracking helped to keep the project on schedule, although some adjustments were made in the final weeks to accommodate additional testing.

#### Collaboration and Supervision

I maintained regular communication with my supervisor, Dr. Martin Rudorfer, meeting with him weekly or bi-weekly to discuss progress and receive feedback. His guidance was invaluable, particularly in helping me navigate challenges related to algorithm implementation and the setup of the simulation environment. Based on his input, I made several adjustments to the project plan, including the decision to focus on RRT\* in the benchmark scenarios due to time constraints.

The collaborative feedback loop allowed me to refine my approach continuously and make data-driven decisions throughout the project. My supervisor's support ensured that I stayed on track and met the key milestones necessary for the successful completion of the dissertation.

#### Challenges and Adaptations

One of the biggest challenges I encountered was setting up the simulation environments and frameworks required for algorithm testing. Implementing the RRT\* algorithm and others in these environments was not a straightforward task due to the need for custom configurations and adjustments. Additionally, comparing the results with the benchmarks required detailed attention to each scenario's setup.

To overcome these challenges, I focused on the jogramop framework, which provided pre-existing environments for testing, reducing the time spent on setup and allowing me to concentrate on algorithm performance. However, I had to adjust my original plan by limiting the number of algorithms tested in new scenarios, which allowed me to complete the project within the given timeframe.

In conclusion, effective project management played a crucial role in the successful execution of this dissertation. By setting clear milestones, managing tasks effectively, and mitigating risks, I was able to complete the implementation and testing of the RRT\* algorithm and compare its performance against established benchmarks. Despite some challenges, regular supervision and adaptive time management ensured the project's success.

# Chapter 6: Evaluation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Scenario | |Gik|/\* |G| | J+-RRT | IK-RRT | RRT\* |
| 11 | 1/1 200 | 91%: 10.53/33.90 | 98%: 1.52/13.33 | |
| 12 | 3/3 200 | 95%: 6.08/26.14 | 100%: 0.02/0.01 | |
| 13 | 0/0 200 | 95%: 6.57/26.07 | â€” |  |
| 14 | 1/1 200 | 97%: 7.21/20.45 | 100%: 0.04/0.02 | |
| 15 | 0/0 200 | 80%: 23.17/24.67 | â€” |  |
| 21 | 1/1 200 | 100%: 2.63/2.94 | 100%: 0.05/0.03 | |
| 22 | 4/4 200 | 100%: 11.26/8.66 | 100%: 0.02/0.01 | |
| 23 | 2/2 200 | 81%: 35.13/21.67 | 98%: 1.52/13.33 | |
| 24 | 1/1 200 | 11%: 60.00/8.39 | 100%: 0.05/0.04 | |
| 25 | 1/1 200 | 2%: 118.56/9.39 | 11%: 108.82/32.07 | |
| 31 | 10/10 200 | 98%: 1.59/13.32 | 100%: 0.01/0.01 | |
| 32 | 11/11 200 | 100%: 0.15/0.29 | 100%: 0.01/0.01 | |
| 33 | 15/15 200 | 98%: 1.78/13.31 | 100%: 0.01/0.01 | |
| 34 | 15/15 200 | 100%: 0.70/0.98 | 100%: 0.01/0.01 | |
| 35 | 8/0 200 | 100%: 1.94/3.05 | â€” |  |
| 41 | 0/0 200 | 97%: 4.18/18.76 | â€” |  |
| 42 | 5/5 200 | 100%: 4.56/7.24 | 100%: 0.10/0.07 | |
| 43 | 2/2 200 | 90%: 25.19/19.80 | 100%: 0.42/0.32 | |
| 44 | 5/5 200 | 56%: 85.87/37.95 | 100%: 1.57/1.14 | |
| 45 | 4/4 200 | 75%: 604.38/206.88 | 100%: 13.51/8.29 | |

# Chapter 7: Conclusion

#### Summary of Research

This research focused on evaluating the RRT\* algorithm within the context of integrated grasp and motion planning, aiming to compare its performance against other algorithms like JPlusRRT and IK-RRT. The project started with implementing RRT in simple pybullet environments and progressed to incorporating the RRT\* algorithm. Additionally, the jogramop framework from Rudorfer et al. was employed to compare the performance of the RRT\* algorithm against other benchmarks in 20 different scenarios. Although time constraints limited the implementation of multiple algorithms in the new environment, the focus on RRT\* allowed for an in-depth assessment of its strengths and weaknesses.

#### Key Findings

The study found that RRT\* excels in less cluttered environments, showcasing its ability to refine paths iteratively for optimal results. Its asymptotic optimality makes it a strong candidate for tasks that demand high precision and efficient motion planning. However, in more complex and dense environments, the algorithm's computational demands rise due to the frequent rewiring required to optimize paths. Despite these challenges, RRT\* demonstrated robustness in dynamic and constrained environments, producing feasible and efficient paths in most scenarios.

#### Contributions to the Field

This dissertation contributes valuable insights into the application of RRT\* for integrated grasp and motion planning. By comparing RRT\* with JPlusRRT and IK-RRT in a benchmark environment, it provides a nuanced understanding of the relative strengths and weaknesses of each algorithm. The successful integration of RRT\* within the jogramop framework adds to the body of knowledge on motion planning in constrained spaces, a critical area for real-world robotic applications. The data collected offers a solid foundation for future studies looking to refine or adapt these algorithms for more complex tasks.

#### Practical Applications

The findings from this research hold practical implications across several industries. In manufacturing, robots can benefit from the efficient motion planning offered by RRT\*, improving both precision and operational safety. In healthcare, particularly for surgical robots or assistive devices, the ability to navigate dynamic environments with high accuracy is invaluable. Autonomous systems, such as self-driving cars or service robots, can also leverage the strengths of RRT\* for efficient navigation and task execution. Although this research was conducted in simulated environments, the insights gained are readily applicable to real-world challenges faced by robotic systems.

#### Final Thoughts

Overall, this research provided a comprehensive exploration of the RRT\* algorithm’s capabilities and limitations in integrated grasp and motion planning. The study overcame challenges, such as the complexity of setting up environments and limited time for testing all algorithms in new scenarios, to deliver meaningful results. Future research can build upon this work by expanding evaluations to other algorithms, exploring hybrid approaches, and conducting real-world testing. By addressing these areas, subsequent studies can further refine integrated grasp and motion planning strategies, making them even more efficient and practical for real-world robotic systems.

# References

Akinola, I. et al. (2021) “Dynamic Grasping with Reachability and Motion Awareness,” in IEEE International Conference on Intelligent Robots and Systems. Available at: https://doi.org/10.1109/IROS51168.2021.9636057.

Ali, A. and Lee, J.Y. (2020) “Integrated motion planning for assembly task with part manipulation using re-grasping,” Applied Sciences (Switzerland), 10(3). Available at: https://doi.org/10.3390/app10030749.

Bertoni, L. et al. (2021) “Towards a Generic Grasp Planning Pipeline using End-Effector Specific Primitive Grasping Actions,” in 2021 20th International Conference on Advanced Robotics, ICAR 2021. Available at: https://doi.org/10.1109/ICAR53236.2021.9659402.

Dalibard, S. and Laumond, J.P. (2010) “Control of probabilistic diffusion in motion planning,” in Springer Tracts in Advanced Robotics. Available at: https://doi.org/10.1007/978-3-642-00312-7\_29.

Dang-Vu, B.A., Porges, O. and Roa, M.A. (2016) “Interpreting manipulation actions: From language to execution,” in Advances in Intelligent Systems and Computing. Available at: https://doi.org/10.1007/978-3-319-27146-0\_14.

Dharbaneshwer, S.J. et al. (2021) “A finite element based simulation framework for robotic grasp analysis,” Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 235(13). Available at: https://doi.org/10.1177/0954406220951596.

Elbanhawi, M. and Simic, M. (2014) “Sampling-based robot motion planning: A review,” IEEE Access. Available at: https://doi.org/10.1109/ACCESS.2014.2302442.

Fan, Z. (2023) “Multi-Point path planning for robots based on deep reinforcement learning,” in Journal of Physics: Conference Series. Available at: https://doi.org/10.1088/1742-6596/2580/1/012048.

Ganesan, M. et al. (2024) “A Comprehensive Review on Deep Learning-Based Motion Planning and End-to-End Learning for Self-Driving Vehicle,” IEEE Access, 12, pp. 66031–66067. Available at: https://doi.org/10.1109/ACCESS.2024.3394869.

Garrett, C.R. et al. (2021) “Integrated Task and Motion Planning,” Annual Review of Control, Robotics, and Autonomous Systems. Available at: https://doi.org/10.1146/annurev-control-091420-084139.

Ichnowski, J., Avigal, Y., et al. (2020) “Deep learning can accelerate grasp-optimized motion planning,” Science Robotics, 5(48). Available at: https://doi.org/10.1126/scirobotics.abd7710.

Ichnowski, J., Danielczuk, M., et al. (2020) “GOMP: Grasp-Optimized Motion Planning for Bin Picking,” in Proceedings - IEEE International Conference on Robotics and Automation. Available at: https://doi.org/10.1109/ICRA40945.2020.9197548.

Lavalle, S.M. (2006) PLANNING ALGORITHMS. Available at: https://msl.cs.uiuc.edu/planning/bookbig.pdf (Accessed: August 4, 2024).

Li, G. et al. (2023) “Design of Digital Planner and 3D Vision System for Robot Bin Picking,” in 2023 8th IEEE International Conference on Advanced Robotics and Mechatronics, ICARM 2023. Available at: https://doi.org/10.1109/ICARM58088.2023.10218895.

Li, T.-Y. and Huang, P.-Z. (2007) “Planning Versatile Motions for Humanoid in a Complex Environment,” in Humanoid Robots: New Developments. Available at: https://doi.org/10.5772/4888.

Liniger, A. and van Gool, L. (2020) “Safe Motion Planning for Autonomous Driving using an Adversarial Road Model,” in Robotics: Science and Systems. Available at: https://doi.org/10.15607/RSS.2020.XVI.044.

Manzinger, S., Pek, C. and Althoff, M. (2021) “Using Reachable Sets for Trajectory Planning of Automated Vehicles,” IEEE Transactions on Intelligent Vehicles, 6(2). Available at: https://doi.org/10.1109/TIV.2020.3017342.

Muhayyuddin, Akbari, A. and Rosell, J. (2015) “Ontological physics-based motion planning for manipulation,” in IEEE International Conference on Emerging Technologies and Factory Automation, ETFA. Available at: https://doi.org/10.1109/ETFA.2015.7301404.

Prats, M., Sanz, P.J. and del Pobil, A.P. (2007) “Task-oriented grasping using hand preshapes and task frames,” in Proceedings - IEEE International Conference on Robotics and Automation. Available at: https://doi.org/10.1109/ROBOT.2007.363582.

Qureshi, A.H. et al. (2019) “Motion planning networks,” in Proceedings - IEEE International Conference on Robotics and Automation. Available at: https://doi.org/10.1109/ICRA.2019.8793889.

Rosell, J. et al. (2019) “Planning grasping motions for humanoid robots,” in International Journal of Humanoid Robotics. Available at: https://doi.org/10.1142/S0219843619500415.

Rubert, C. et al. (2018) “Characterisation of Grasp Quality Metrics,” Journal of Intelligent and Robotic Systems: Theory and Applications, 89(3–4). Available at: https://doi.org/10.1007/s10846-017-0562-1.

Rudorfer, M., Hartvich, J. and Vonásek, V. (no date) A Framework for Joint Grasp and Motion Planning in Confined Spaces.

Tsuji, T. et al. (2010) “Grasp Planning for a Multifingered Hand with a Humanoid Robot,” Journal of Robotics and Mechatronics, 22(2). Available at: https://doi.org/10.20965/jrm.2010.p0230.

Vahrenkamp, N. et al. (2010) “Integrated grasp and motion planning,” in Proceedings - IEEE International Conference on Robotics and Automation. Available at: https://doi.org/10.1109/ROBOT.2010.5509377.

Wan, W. and Harada, K. (2016) “Integrated assembly and motion planning using regrasp graphs,” Robotics and Biomimetics, 3(1). Available at: https://doi.org/10.1186/s40638-016-0050-2.

Wan, W. and Harada, K. (2017) “Regrasp planning using 10,000s of grasps,” in IEEE International Conference on Intelligent Robots and Systems. Available at: https://doi.org/10.1109/IROS.2017.8206011.

Wang, L., Xiang, Y. and Fox, D. (2020) “Manipulation Trajectory Optimization with Online Grasp Synthesis and Selection,” in Robotics: Science and Systems. Available at: https://doi.org/10.15607/RSS.2020.XVI.033.

Yan, W. et al. (2019) “Precision Grasp Planning for Multi-Finger Hand to Grasp Unknown Objects,” Robotica, 37(8). Available at: https://doi.org/10.1017/S0263574719000031.

Zafra-Urrea, R.M., López-Damian, E. and Santana-Díaz, A. (2023) “Grasp Planning Based on Metrics for Collaborative Tasks Using Optimization,” Applied Sciences (Switzerland), 13(17). Available at: https://doi.org/10.3390/app13179603.

Zhang, H. and Zhu, Z. (2020) “Sampling-based motion planning for free-floating space robot without inverse kinematics,” Applied Sciences (Switzerland), 10(24). Available at: https://doi.org/10.3390/app10249137.11. Appendices

# Appendix