HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FALTY OF COMPUTER SCIENCE AND ENGINEERING

CAPSTONE PROJECT
COMPUTER ENGINEERING

Motion Planning around Obstacles

Semester 251

Instructor: Assoc. Prof. Le Hong Trang

STT	Full name	ID	Note
1	Nguyen Tran Dang Khoa	2211635	
2	Ho Dang Khoa	2211xxxx	

Ho Chi Minh City, December 2025

Abstract

Trajectory optimization offers mature tools for motion planning in high-dimensional spaces under dynamic constraints. However, in obstacle-cluttered environments, the non-convexity of the free space typically forces roboticists to rely on sampling-based planners, which struggle with high dimensions and differential constraints. Here, we demonstrate that convex optimization can reliably solve these problems through the convex decomposition of the free space. Specifically, we decompose the collision-free configuration space into finite convex regions, organizing them into a Graph of Convex Sets (GCS). By combining Bézier curves with this graph structure, we formulate the motion planning problem as a compact mixed-integer convex program that naturally handles collision avoidance, velocity, and duration constraints. We validate GCS across a spectrum of environments with increasing complexity, ranging from simple 2D obstacles and cluttered mazes to dynamical quadrotors and high-dimensional 7-DoF manipulators. Numerical experiments demonstrate that GCS consistently outperforms widely-used sampling-based planners (e.g., PRM). Specifically, in high-dimensional tasks, GCS reduces online query time by up to an order of magnitude compared to standard PRM, while finding globally optimal trajectories that are significantly shorter (up to 40% reduction in length) than those from post-processed sampling-based planners [1]. Ultimately, this work brings the reliability and speed of convex optimization to the traditionally challenging domain of nonconvex motion planning.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In chapter 1, the overview, objectives and goals of the research's project are illustrated. The outline of the report is also presented.

1.1 Motivation and Technical Challenges

Making optimal decisions in real-time is a cornerstone of modern engineering. The necessity of tightly coupling **discrete decisions** with **continuous control** is vividly illustrated in advanced robotics:

- **Humanoid Robots (e.g., Boston Dynamics' Atlas):** Navigating complex terrain, such as a parkour course, requires the robot to simultaneously select footstep locations (discrete) and compute the trajectories for its center of mass and limbs (continuous). Currently, solutions often rely on "hand-coding" the discrete components, which severely limits the robot's autonomy in novel environments [2].
- **Industrial Manipulators (e.g., Amazon's Robin):** Sorting efficiency depends on jointly optimizing the sequence of bins (discrete) and the arm's trajectory (continuous) [3]. A mere **1% increase** in operating speed through superior optimization translates to millions of additional packages processed annually.

However, solving these coupled problems simultaneously remains an open challenge, typically necessitating manual engineering interventions or settling for suboptimal solutions.

Algorithmically, selecting a motion planning method currently forces researchers to compromise between conflicting features: dimensionality, dynamic constraints, and completeness.

- **Trajectory Optimization:** These methods excel at handling robot dynamics and high-dimensional spaces. However, when facing obstacle-cluttered environments—which render the problem inherently non-convex—they often get trapped in **local minima**, failing to identify a collision-free trajectory.[4]
- **Sampling-based Planners:** To overcome these limitations, the robotics community frequently resorts to sampling-based methods (e.g., RRT, PRM) due to their "probabilistic completeness". However, this comes at a cost: the resulting trajectories are often significantly **suboptimal**, and imposing **continuous differential constraints** on discrete samples remains notoriously difficult.[5], [6]

Mixed-Integer Convex Programming (MICP) has emerged as a promising solution that bridges this gap. It combines the strengths of both worlds: the completeness of sampling-based algorithms and the dynamic handling capabilities of trajectory optimization, while guaranteeing **global optimality** within a unified framework. Nevertheless, the adoption of MICP is severely limited by its prohibitive computational costs, often requiring minutes to solve even small-scale problems.

The objective of this research is to break this computational barrier. We focus on a crucial class of motion planning problems involving differential constraints and propose an MICP-based planner capable of reliably solving high-dimensional problems in a matter of seconds through a **Single Convex Program**.

1.2 Goals

1.3 Scope

1.4 Thesis structure

There are five chapters in this capstone project:

- Chapter 1 briefly describes the project about problem's motivation, as well as the project's objectives and scope
- Chapter 2 is dedicated to presenting the foundational knowledge about convex optimization, mixed-integer optimization, and graph theory that are essential for understanding the proposed method.
- Chapter 3 arranges the discussion of related works on the same task to deepen the understanding of existing methods and their constraints.
- Chapter 4 describes the GCS framework and the formulation of our MICP. From that, we present the experimental results and analysis of our method in various environments with different complexity levels.
- Chapter 5 summarizes whole project and the plan for future development.

Chapter 2

Theoretical Background

In Chapter 2, it presents about preliminary knowledge in this project.

2.1 Convex Analysis and Optimization

2.1.1 Convex Sets

Before discussing optimization problems, we briefly define the fundamental geometric objects used in this work. A set $\mathcal{C} \subseteq \mathbb{R}^n$ is *convex* if the line segment between any two points in the set lies entirely within the set.

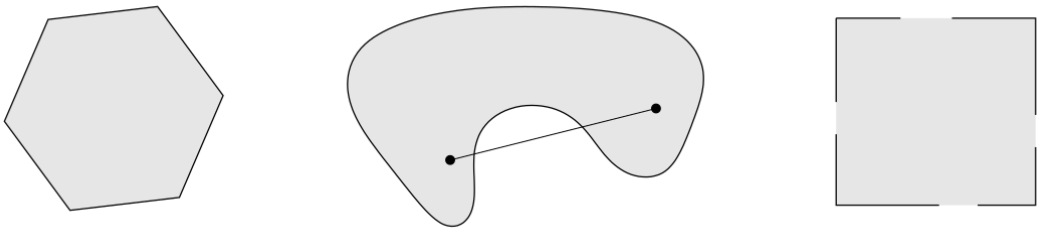

Figure 2.1: Examples of simple convex and nonconvex sets. Left: The hexagon, including its boundary, is convex. Middle: The kidney-shaped set is nonconvex, as the line segment connecting the two points lies outside the set. Right: The square contains some boundary points but not others, and is therefore not convex.[7]

Convex hulls are the smallest convex sets containing a given set of points. Formally, the convex hull of a set of points $\{x_1, x_2, \ldots, x_m\} \subseteq \mathbb{R}^n$ is defined

as:

$$\operatorname{conv}(\{x_1, x_2, \ldots, x_m\}) = \left\{ \sum_{i=1}^{m} \lambda_i x_i \mid \lambda_i \geq 0, \sum_{i=1}^{m} \lambda_i = 1 \right\}.$$

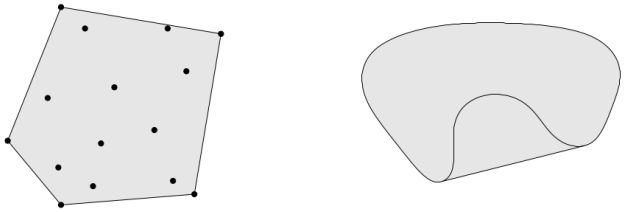

Figure 2.2: Convex hull of a set of points (shaded area).[7]

In the context of motion planning and GCS, we primarily focus on two types of convex sets:

- **Polyhedra:** Defined as the intersection of a finite number of halfspaces, $\mathcal{P} = \{x \in \mathbb{R}^n \mid Ax \leq b\}$. Bounded polyhedra are called *polytopes*.
- **Ellipsoids:** Defined as the image of a unit ball under an affine transformation, typically represented as $\mathcal{E} = \{x \in \mathbb{R}^n \mid (x-c)^\top Q^{-1}(x-c) \leq 1\}$, where $Q \succ 0$.

These sets will serve as the regions (vertices) in our motion planning framework.

2.1.2 Convex Optimization

A **Convex Program (CP)** is an optimization problem of the form

$$\text{minimize} \quad f(x) \tag{2.1a}$$

$$\text{subject to} \quad x \in \mathcal{C}, \tag{2.1b}$$

where the objective function $f : \mathbb{R}^n \to \mathbb{R}$ and the constraint set $\mathcal{C} \subset \mathbb{R}^n$ are convex. Sometimes we will refer to the set $\mathcal{C}$ as the *feasible set* of the CP, and to its elements as *feasible solutions*. The CP is said to be *feasible* if a feasible solution exists, i.e., if $\mathcal{C} \neq \emptyset$. A feasible solution x^{opt} is

optimal if $f(x^{\text{opt}}) \leq f(x)$ for all $x \in \mathcal{C}$. If an optimal solution exists, we call $f^{\text{opt}} := f(x^{\text{opt}})$ the *optimal value* of the CP.

A fundamental property of CPs is that any locally optimal solution (i.e., any point x^{opt} such that $f(x^{\text{opt}}) \leq f(x)$ for all x in a neighborhood of x^{opt}) is also an optimal solution according to the (global) definition just given [7]. A commonly satisfied assumption for the existence of an optimal solution is that the feasible set $\mathcal{C}$ is nonempty and compact.

CPs are a fundamental class of optimization problems, with applications in essentially every engineering discipline. The great majority of the CPs that we encounter in practice can be solved efficiently and reliably using interior-point methods (or other specialized algorithms such as the simplex method). However, strictly speaking, it is not always true that a CP can be solved efficiently: for example, the set of nonnegative polynomials is a convex cone (in the space of the polynomial coefficients), but checking if a polynomial is nonnegative is NP-hard. In this thesis we will be a little imprecise, and use the term "convex optimization problem" almost as a synonym of "optimization problem that is efficiently solvable."

2.1.3 Conic Optimization

A **Conic Program (KP)** is a CP with linear objective function and constraint set in conic form:

$$\text{minimize} \quad c^\top x \tag{2.2a}$$

$$\text{subject to} \quad Ax + b \in \mathcal{K}, \tag{2.2b}$$

where $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $\mathcal{K} \subseteq \mathbb{R}^m$ is a closed convex cone. Special classes of KPs are:

- **Linear Program (LP)** when $\mathcal{K}$ is the nonnegative orthant.
- **Second-Order Cone Program (SOCP)** when $\mathcal{K}$ is the Cartesian

product of second-order cones.

- **Semidefinite Program (SDP)** when $\mathcal{K}$ is (isomorphic to) the semidefinite cone.

These classes of problems have increasing modelling power: every LP is an SOCP, and every SOCP is an SDP. SDPs are efficiently solvable, and cover the vast majority of the practical uses of convex optimization.
Another important class of CPs are **Quadratic Programs (QP)**, these are CPs in the form (2.1) with quadratic objective function f and polyhedral constraint set $\mathcal{C}$. Every QP can be formulated as an SOCP. However, in many cases, active-set algorithms specialized to this class of problems can be more effective than using an SOCP solver.

2.1.4 Homogenization (Perspective Operator)

Homogenization is a technique used in convex analysis to transform a non-convex set into a convex one by introducing an additional dimension. This process is particularly useful in optimization problems where convexity is desired for efficient solution methods.
Given a set $\mathcal{S} \subseteq \mathbb{R}^n$, the homogenization of $\mathcal{S}$, denoted as $\widetilde{\mathcal{S}}$, is defined as:

$$\widetilde{\mathcal{S}} := \{(x, y) \in \mathbb{R}^{n+1} : y > 0, x \in y\mathcal{S}\}.$$

This transformation effectively "lifts" the set $\mathcal{S}$ into a higher-dimensional space, where the additional dimension y allows for the representation of convex combinations of points in $\mathcal{S}$. The homogenized set $\widetilde{\mathcal{S}}$ is convex if and only if the original set $\mathcal{S}$ is convex.

Homogenization of Sets in Conic Form

The homogenization of a convex set $\mathcal{C}$ described in conic form is computed

very easily. In fact, for $y > 0$, we observe that

$$y\mathcal{C} = \{yx : Ax + b \in \mathcal{K}\} = \{yx : A(yx) + by \in y\mathcal{K}\} = \{x' : Ax' + by \in \mathcal{K}\},$$

and this gives us

$$\widetilde{\mathcal{C}} = \{(x, y) : y > 0, Ax + by \in \mathcal{K}\}. \tag{2.3}$$

In addition, it is also easily verified that

$$\mathrm{cl}(\widetilde{\mathcal{C}}) = \{(x, y) : y \geq 0, Ax + by \in \mathcal{K}\}. \tag{2.4}$$

The expressions (2.3) and (2.4) have great practical relevance. Roughly speaking, they tell us that if we can efficiently do computations with a set $\mathcal{C}$ described in conic form, then the same is true for (the closure of) its homogenization $\widetilde{\mathcal{C}}$.

Homogenization is particularly useful in optimization problems, as it allows for the application of convex optimization techniques to problems that may initially appear non-convex. By working in the homogenized space, one can leverage the properties of convex sets and functions to find optimal solutions more efficiently.

2.2 Mixed-Integer Optimization

Mixed-integer optimization is the computational backbone of the techniques introduced in this thesis. Generally speaking, a **Mixed-Integer Program (MIP)** can be very hard to solve and the runtimes of a mixed-integer solver can grow very quickly (exponentially) with the problem size. However, this is the worst-case scenario, and a lot can be done to construct highly effective MIPs that can be solved quickly for most of (if not all) the instances that we encounter in practice. The goal of this section is twofold.

First, we introduce the essential background on mixed-integer optimization: what an MIP is, how MIPs are classified, and what algorithms can be used to solve these problems. Secondly, we introduce the notions and the tools necessary to distinguish between efficient and inefficient MIPs.
Two great sources for all the details about mixed-integer optimization are the classical book [8] and the more recent book [9].

2.2.1 Mixed-Integer Programs

An MIP is an optimization problem with continuous variables $x \in \mathbb{R}^n$ and discrete (integer) variables $y \in \mathbb{Z}^m$. Given an objective function $f : \mathbb{R}^{n+m} \to \mathbb{R}$ and a constraint set $\mathcal{S} \subseteq \mathbb{R}^{n+m}$, we can state a generic MIP as

$$\begin{aligned} \text{minimize} \quad & f(x,y) && \text{(2.5a)} \\ \text{subject to} \quad & (x,y) \in \mathcal{S}, && \text{(2.5b)} \\ & y \in \mathbb{Z}^m. && \text{(2.5c)} \end{aligned}$$

(We explicitly state the constraint $y \in \mathbb{Z}^m$ since, if not specified otherwise, variables of optimization problems will always be assumed to be real valued.) We call the set

$$\mathcal{T} := \mathcal{S} \cap \mathbb{R}^n \times \mathbb{Z}^m$$

the *feasible set* of the MIP (2.5) and its elements $(x,y) \in \mathcal{T}$ *feasible solutions*. The MIP is *feasible* if a feasible solution exists and *infeasible* otherwise. A feasible solution $(x^{\text{opt}}, y^{\text{opt}})$ is *optimal* if $f(x^{\text{opt}}, y^{\text{opt}}) \leq f(x,y)$ for all $(x,y) \in \mathcal{T}$. If an optimal solution exists, we denote with $f^{\text{opt}} := f(x^{\text{opt}}, y^{\text{opt}})$ the MIP optimal value.
If we discard constraint (2.5c) from our MIP, we obtain the following opti-

mization problem:

$$\text{minimize} \quad f(x,y) \tag{2.6a}$$

$$\text{subject to} \quad (x,y) \in \mathcal{S}. \tag{2.6b}$$

This problem is called the **relaxation** of the MIP. We denote an optimal solution of the relaxation as $(x^{\text{relax}}, y^{\text{relax}})$, and we let $f^{\text{relax}} := f(x^{\text{relax}}, y^{\text{relax}})$ be its optimal value. Observe that

$$f^{\text{relax}} \leq f^{\text{opt}},$$

since by discarding a constraint from an optimization (minimization) problem the optimal value can only decrease. As discussed below, the tightness of this inequality plays a central role in the efficiency of an MIP. Loosely speaking, an MIP is efficiently solvable if its relaxation can be solved quickly and the gap $f^{\text{opt}} - f^{\text{relax}} \geq 0$ is small.

MIPs are classified according to the properties of the function f and the set $\mathcal{S}$, which can significantly affect our ability of solving an MIP efficiently. Below we define the classes of MIPs that are most relevant for this thesis.

2.2.2 Mixed-Boolean Programs

A **Mixed-Boolean Program (MBP)** is an MIP where the discrete variables can only take binary value: $y \in \{0,1\}^m$. Equivalently, an MBP is a problem of the form of (2.5) where

$$\mathcal{S} \subset \mathbb{R}^n \times [0,1]^m.$$

All the MIPs that we will encounter in this thesis are, in fact, MBPs. However, the term MBP is unusual and, although technically our problems will be MBPs, we will still call them as MIPs.

2.2.3 Mixed-Integer Convex Programs

If the objective function f and the constraint set $\mathcal{S}$ are convex, we call problem (2.5) a **Mixed-Integer Convex Program (MICP)**. MICPs are a fundamental class of MIPs since their relaxations are convex optimization problems, which (in most of the cases) can be solved very quickly. To emphasize the convexity assumption, we call the relaxation of an MICP *convex relaxation*. Using the Branch-and-Bound (BB) algorithm, most MICPs can be reliably solved to global optimality, although the algorithm might take a long time.

In contrast to the class of MICPs, we will occasionally call **Mixed-Integer Non-Convex Program (MINCP)** an MIP where the objective function f and/or the constraint set $\mathcal{S}$ are nonconvex. Most MINCPs are intractable; the main exceptions are very small MINCPs where the feasible set $\mathcal{T}$ can be finely discretized and searched exhaustively.

2.2.4 Mixed-Integer Conic Programs

If the objective function f is linear and the constraint set $\mathcal{S}$ is a closed convex set in conic form, then problem (2.5) is called **Mixed-Integer Conic Program (MIKP)**. The subclasses of this family of problems are classified as follows:

- **Mixed-Integer Linear Program (MILP)** when the set $\mathcal{S}$ is a polyhedron,
- **Mixed-Integer Second-Order Cone Program (MISOCP)** when $\mathcal{S}$ is convex quadratic,
- **Mixed-Integer Semidefinite Program (MISDP)** when $\mathcal{S}$ is a spectrahedron.

The relaxations of these problems are named in the natural way. For example, we call *linear relaxation* the relaxation of an MILP.

We could say that MILPs play a more important role in mixed-integer optimization than LPs play in convex optimization. The first reason for this is geometric. Polyhedra are the simplest shape that can enclose finite sets of points. Therefore, as we will also see in the next chapter, they are a natural candidate to model the discrete side of an MIP. The second reason is algorithmic. The simplex algorithm for linear optimization can be integrated in the BB procedure more efficiently than the interior-point algorithm used for more general conic programs.

A **Mixed-Integer Quadratic Program (MIQP)** is an MIP of the form (2.5) with f quadratic and $\mathcal{S}$ polyhedral. These problems are representable as MISOCPs but, similar to MILPs, they can be solved more efficiently using specialized BB methods that leverage the polyhedral constraint set. Having said this, nowadays also MISOCPs and MISDPs can be solved quite effectively even with freely available solvers (see, e.g., Pajarito [10]).

2.3 Graphs of Convex Sets

2.4 Kinodynamic Trajectory Generation

This section outlines the mathematical formulation for generating kinodynamically feasible trajectories. Unlike geometric path planning, which solely considers collision-free configurations in the workspace, motion planning (kinodynamic) accounts for the system's differential constraints, such as velocity, acceleration, and jerk limits. We leverage Bézier curves as the trajectory parameterization scheme, which is particularly advantageous for the Graph of Convex Sets (GCS) framework due to its convexity properties.

2.4.1 Kinodynamic Formulation

Consider a robotic system whose state evolution is governed by a set of ordinary differential equations (ODEs):

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \quad \mathbf{x}(t) \in \mathcal{X}, \mathbf{u}(t) \in \mathcal{U} \tag{2.7}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ represents the state vector (typically comprising position, velocity, and acceleration), and $\mathbf{u}(t) \in \mathbb{R}^m$ denotes the control inputs. The goal is to compute a trajectory $\sigma(t) : [0, T] \to \mathcal{X}$ that minimizes a specific cost functional while satisfying:

1. **Boundary Conditions:** The start and goal states $\mathbf{x}(0) = \mathbf{x}_{start}$ and $\mathbf{x}(T) = \mathbf{x}_{goal}$.
2. **Safety Constraints:** $\sigma(t) \in \mathcal{X}_{free}$ for all t, ensuring collision avoidance.
3. **Dynamic Limits:** Inequality constraints on higher-order derivatives to ensure physical feasibility:

$$|\dot{\sigma}(t)| \leq v_{max}, \quad |\ddot{\sigma}(t)| \leq a_{max}, \quad |\dddot{\sigma}(t)| \leq j_{max} \tag{2.8}$$

2.4.2 Smoothness and Continuity Constraints

In the GCS framework, a full trajectory is composed of multiple piecewise Bézier segments. To ensure smooth motion across the boundaries of adjacent convex sets, continuity constraints must be enforced at the transition points. Let $r_A(t)$ defined by control points $\{a_0, \ldots, a_n\}$ and $r_B(t)$ defined by $\{b_0, \ldots, b_n\}$ be two consecutive segments.

- C^0 **Continuity (Position):** Ensures the path is connected.

$$a_n = b_0 \tag{2.9}$$

- C^1 **Continuity (Velocity):** Ensures continuous velocity, preventing infinite acceleration spikes.

$$a_n - a_{n-1} = b_1 - b_0 \tag{2.10}$$

- C^2 **Continuity (Acceleration):** Ensures continuous force/torque application, which is crucial for minimizing mechanical wear and jerk.

$$(a_n - 2a_{n-1} + a_{n-2}) = (b_2 - 2b_1 + b_0) \tag{2.11}$$

By enforcing these equality constraints at the edges of the graph, we generate a globally smooth trajectory that respects the underlying physics of the robot.

2.4.3 Background on Bézier Curves

In order to tackle the kinodynamic planning problem numerically, it is necessary to parameterize the trajectory functions through a finite number of decision variables. To this end, we employ Bézier curves. This section recalls the definition and the basic properties of this family of curves.
A Bézier curve is constructed using Bernstein polynomials. The k-th Bernstein polynomial of degree d, with $k = 0, \dots, d$, is defined as:

$$\beta_{k,d}(s) := \binom{d}{k} s^k (1-s)^{d-k}, \quad s \in [0,1] \tag{2.12}$$

Note that the Bernstein polynomials of degree d are nonnegative and, by the binomial theorem, they sum up to one. Therefore, for each fixed $s \in [0,1]$, the scalars $\{\beta_{k,d}(s)\}_{k=0}^{d}$ can be thought of as the coefficients of a convex combination. Bézier curves are obtained using these coefficients to combine

a given set of $d+1$ control points $\gamma_k \in \mathbb{R}^n$:

$$\gamma(s) := \sum_{k=0}^{d} \beta_{k,d}(s)\gamma_k \tag{2.13}$$

It is easily verified that Bézier curves enjoy the following properties, which are instrumental for the GCS framework:

- **Endpoint values:** The curve γ starts at the first control point and ends at the last control point:

  $$\gamma(0) = \gamma_0 \quad \text{and} \quad \gamma(1) = \gamma_d \tag{2.14}$$

  This property is essential for enforcing continuity constraints (C^0) when stitching multiple Bézier segments together in the graph.

- **Convex hull property:** The curve γ is entirely contained in the convex hull of its control points:

  $$\gamma(s) \in \text{conv}(\{\gamma_k\}_{k=0}^{d}), \quad \forall s \in [0,1] \tag{2.15}$$

  In the context of GCS, if a convex region X_v represents a safe set, guaranteeing collision avoidance is reduced to constraining all control points to lie within that set:

  $$\gamma_k \in X_v, \quad \forall k \in \{0,\dots,d\} \implies \gamma(s) \in X_v \tag{2.16}$$

  This allows us to enforce continuous safety constraints through a finite set of linear inclusion constraints.

- **Derivative (Hodograph):** The derivative $\dot{\gamma}$ of the curve γ is also a Bézier curve, but of degree $d-1$, with control points defined by the

difference of adjacent points:

$$\dot{\gamma}(s) = \sum_{k=0}^{d-1} \beta_{k,d-1}(s)\dot{\gamma}_k \tag{2.17}$$

where $\dot{\gamma}_k = d(\gamma_{k+1} - \gamma_k)$ for $k = 0, \ldots, d-1$. Similarly, higher-order derivatives (acceleration, jerk) can be expressed as linear combinations of the control points. Consequently, kinodynamic constraints (e.g., velocity limits $|\dot{\gamma}(s)| \leq v_{max}$) can be imposed as linear constraints on the differences between adjacent control points, preserving the convexity of the optimization problem.

- **Integral of convex function:** For a convex function $f : \mathbb{R}^n \to \mathbb{R}$ (often representing the cost function in motion planning, such as energy or path length), the integral along the curve is bounded by the convex combination of the function values at the control points:

$$\int_0^1 f(\gamma(s))ds \leq \frac{1}{d+1} \sum_{k=0}^{d} f(\gamma_k) \tag{2.18}$$

This inequality provides a convex upper bound on the integral cost, facilitating efficient optimization within the GCS mixed-integer convex programming formulation.

Chapter 3

Related works

...

3.1 Sampling-Based Methods

3.2 Discussion

In this chapter, we have seen three distinct approaches to enhance...

Chapter 4

Proposed Solution

As mentioned in Chapter 3, the enhancement of knowledge of . Chapter 4 presents the proposed solution for enhancing .

4.1 Baseline methods

4.2 Proposed solution: ...

4.3 Experiment setup

4.3.1 Implementation details

4.4 Results and analysis

Chapter 5

Conclusion

5.1 Summary

5.2 Future Work

Future research will focus on...

- ...
- ...

References

[1] R. Deits and R. Tedrake, "Motion planning around obstacles with convex optimization," *arXiv preprint arXiv:2205.04422*, 2022.

[2] Boston Dynamics, *Leaps, bounds, and backflips*, https://bostondynamics.com/blog/leaps-bounds-and-backflips/, Accessed: November 29, 2025, 2021.

[3] Amazon Robotics, *Amazon.com announces first quarter results*, https://ir.aboutamazon.com/news-release/news-release-details/2023/Amazon.com-Announces-First-Quarter-Results/default.aspx, Accessed: November 29, 2025, 2023.

[4] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," in *Fast Motions in Biomechanics and Robotics*, Springer, 2006, pp. 65–93.

[5] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University, Tech. Rep. TR 98-11, 1998.

[6] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004, ISBN: 0521833787.

[8] L. A. Wolsey, *Integer Programming*. Wiley-Interscience, 1999, ISBN: 0471283665.

[9] M. Conforti, G. Cornuéjols, and G. Zambelli, *Integer Programming*. Springer, 2014, ISBN: 9783319110073.

[10] M. Lubin, E. Yamangil, R. Bent, and J. P. Vielma, "Polyhedral approximation in mixed-integer convex optimization," *Mathematical Programming*, vol. 172, pp. 139–168, 2018.

Appendix A

...

A.1 ...

A.2 ...