GCS Path Planning: Formulation and Decomposition

Antigravity

November 18, 2025

1 Introduction

This document details the mathematical formulation for the Graph of Convex Sets (GCS) path planning algorithm and the convex decomposition method used to generate the free space regions.

2 Convex Decomposition

The goal is to decompose the free space $\mathcal{F} = \mathcal{W} \setminus \bigcup_{j=1}^{M} \mathcal{O}_j$, where $\mathcal{W}$ is the rectangular domain and $\mathcal{O}_j$ are convex obstacles, into a set of convex polygons $\{R_1, \ldots, R_N\}$.

2.1 Hole Integration

Since the free space is multiply-connected (contains holes), we first transform it into a simply-connected polygon P.

1. Let P_0 be the boundary of $\mathcal{W}$ (ordered Counter-Clockwise).
2. For each obstacle $\mathcal{O}_j$ (ordered Clockwise):
   - Find a "bridge" connecting a vertex $v \in \mathcal{O}_j$ to a visible vertex $u \in P_{current}$.
   - Insert the sequence of $\mathcal{O}_j$ vertices into $P_{current}$ at u, effectively merging the hole into the boundary.
3. The result is a single polygon P (possibly with self-touching edges) that represents $\mathcal{F}$.

2.2 Partitioning

We decompose P into convex regions using a heuristic approach based on resolving reflex vertices (vertices with internal angle $> 180°$).

Algorithm: Convex Decomposition

1. Initialize $Regions \leftarrow \emptyset$.
2. While P is not convex:
   (a) Find a diagonal $d = (v_i, v_k)$ such that:
      - The polygon formed by $v_i, \ldots, v_k$ is convex.
      - The diagonal d does not intersect any other edges of P.
      - No other vertices of P lie inside the formed polygon.

   (b) Let $C = \{v_i, \ldots, v_k\}$.

   (c) $Regions \leftarrow Regions \cup \{C\}$.

   (d) $P \leftarrow P \setminus C$ (Update boundary of P).
3. $Regions \leftarrow Regions \cup \{P\}$.

3 Trajectory Optimization (MICP)

We solve a Mixed-Integer Convex Program (MICP) to find a path through the graph of convex sets and optimize the trajectory within them.

3.1 Sets and Indices

- V: Set of convex regions (polygons), indexed by i.
- E: Set of edges (i, j) representing adjacency between region i and j.
- K: Degree of the Bézier curve (we use $K = 2$ for quadratic).
- d: Dimension ($d = 2$).

3.2 Variables

- $y_i \in \{0, 1\}$: Binary variable, equal to 1 if region i is visited.
- $z_{ij} \in \{0, 1\}$: Binary variable, equal to 1 if the transition from region i to j is active.
- $x_{i,k} \in \mathbb{R}^2$: Control point k ($k \in \{0, \ldots, K\}$) for the Bézier curve in region i.
- $t_{i,k} \in \mathbb{R}^2$: Slack variables for path length (velocity) minimization.
- $a_i \in \mathbb{R}^2$: Slack variables for acceleration (smoothness) minimization.

3.3 Optimization Problem

The objective is to minimize a weighted sum of path length and acceleration (smoothness).

$$\min \sum_{i \in V} \left(\sum_{k=1}^{K} \|x_{i,k} - x_{i,k-1}\|_1 + \lambda_{smooth} \|x_{i,2} - 2x_{i,1} + x_{i,0}\|_1 \right)$$

Subject to:

1. Flow Conservation

$$\begin{aligned}
&\sum_{j:(s,j) \in E} z_{sj} - \sum_{j:(j,s) \in E} z_{js} = 1 \quad \text{(Start Node } s\text{)} \\
&\sum_{j:(g,j) \in E} z_{gj} - \sum_{j:(j,g) \in E} z_{jg} = -1 \quad \text{(Goal Node } g\text{)} \\
&\sum_{j:(i,j) \in E} z_{ij} - \sum_{j:(j,i) \in E} z_{ji} = 0 \quad \forall i \in V \setminus \{s, g\} \\
&y_i \geq \sum_j z_{ij}, \quad y_i \geq \sum_j z_{ji}
\end{aligned}$$

2. Containment

$$\begin{aligned}
&A_i x_{i,k} \leq b_i + M(1 - y_i) \quad \forall i \in V, k \in \{0, \ldots, K\} \\
&-My_i \leq x_{i,k} \leq My_i \quad \text{(Force } x_{i,k} = 0 \text{ if } y_i = 0\text{)}
\end{aligned}$$

3. Continuity (C^0)

$$\|x_{i,K} - x_{j,0}\|_\infty \leq M(1 - z_{ij}) \quad \forall (i, j) \in E$$

4. Heading Consistency (C^1)

To ensure smooth transitions, we enforce continuity of the velocity vector. For quadratic Bézier curves, the tangent is proportional to the difference between control points.

$$\|(x_{i,K} - x_{i,K-1}) - (x_{j,1} - x_{j,0})\|_\infty \leq M(1 - z_{ij}) \quad \forall (i,j) \in E$$

5. Boundary Conditions

$$x_{s,0} = p_{\text{start}}, \quad x_{g,K} = p_{\text{goal}}$$

Note: The L1 norm objective and infinity norm constraints are used to keep the problem linear (MILP), which is compatible with the HiGHS solver. M is a sufficiently large constant ("Big-M"). λ_{smooth} is a regularization weight to encourage smoother paths.