

使用凸集的时空图进行系统约束公式化和无碰撞轨迹规划

Matthew D. Osburn, Cameron K. Peterson, and John L. Salmon

Abstract—在本文中，我们创建了最佳的、无碰撞的、时间依赖的轨迹，适用于拥挤的动态环境。由于存在大量的空间和时间约束，为数值求解器找到初始猜测变得困难。凸集图（GCS）和最近开发的时空凸集图公式（ST-GCS）使我们能够在不为求解器提供初始猜测的情况下生成最优的、最小距离的无碰撞轨迹。我们还探索了一般 GCS 兼容约束的推导，并记录了一种适用于该框架的直观策略。我们证明，当环境静止时，ST-GCS 产生的结果与标准的 GCS 公式相同。然后我们展示 ST-GCS 在动态环境中运行以找到最小距离的无碰撞轨迹。

I. 介绍

在自主系统中，一个基本问题是生成一个安全的参考路径或轨迹，供反馈控制器跟踪。存在众多规划方法用于计算满足任务特定目标的几何路径 [1]。然而，对于动态或时变环境而言，这些路径通常是不够的 [2], [3]，这需要明确地建模障碍物或目标随时间的变化。在这些情况下，轨迹——定义为一个随时间参数化的路径——更为适用，因为它能够在环境变化时实现时间上的协调和优化 [4], [5]。

在运动规划中，通过将空间划分为能够进行高效基于搜索的方法的区域，环境经常表示为离散图形。算法例如 Dijkstra 算法 [6]、A* 和 D* [7] 被广泛用于计算全局最优路径，而不需要初始猜测 [8]。然而，图搜索算法的计算复杂度随着图的大小而增加 [9], [10]，并且产生的路径通常忽略物理约束，如连续性或平滑性。因此，离散解通常需要后处理以将其转换为可行的连续轨迹 [11]，这可能导致偏离全局最优解。

相比之下，连续优化方法可以直接在轨迹上施加物理约束。这些方法使用解析函数来建模障碍物，例如符号距离场 [12]、控制障碍函数 [13] 或多边形表示 [14]，并在这些约束条件下优化成本函数。然而，连续优化的一个关键限制是其对良好初始猜测的依赖。在复杂或杂乱的

环境中，糟糕的初始化可能导致次优或不可行的解决方案。为了缓解这一问题，通常使用基于采样的方法来生成可行的初始轨迹 [15], [16]。尽管如此，这些方法对移动障碍物特别敏感，因为这会引入额外的时间非线性和局部极小值 [17], [18]。

为克服这些局限性，Marcucci 等人引入了凸集图（Graphs of Convex Sets, GCS）框架 [19]–[21]。GCS 是一种轨迹优化方法，它将图搜索的离散结构与连续优化的约束处理能力相结合。该框架构建了一个凸区域的图，并将轨迹优化表述为一个结合图搜索和凸优化的程序。值得注意的是，GCS 已被应用于最短路径规划 [20], [22]、迷宫环境中的最短时间导航 [21] 以及高维配置空间中的机器人手臂运动规划 [23]–[25]。

GCS 的一大优势在于它能够找到连续的轨迹，而不需要初始猜测。它使用分支定界算法来迭代地细化候选解，直到找到全局最优轨迹。此外，GCS 提供了指标来表示候选解的质量，从而能够快速选择高质量的局部最优解 [22]。

尽管具有这些优势，但当前的 GCS 流程管道存在明显的局限性。正如在 [21] 中所解释的那样，它通过首先计算一个静态环境中的全局最优无碰撞路径来分离空间和时间规划，然后通过外部非线性优化来确定轨迹的时间安排。后者不仅牺牲了全局优化保证，而且即使存在有效解决方案，也可能无法产生有效解。此外，现有方法对动态障碍规避的处理有限，必须进一步开发和集成这种处理，才能在这个框架下可靠地应用于安全轨迹规划。

导致这种复杂性的一个特别具有挑战性的方面是，在整个图上统一公式化约束条件，如 [19] 中所述。即使是基本约束，例如在样条段之间强制连续性，也必须用高度具体的数学形式表达，而这种形式通常被简化或完全省略。与其他框架相比，这一要求增加了实施的挑战。据我们所知，

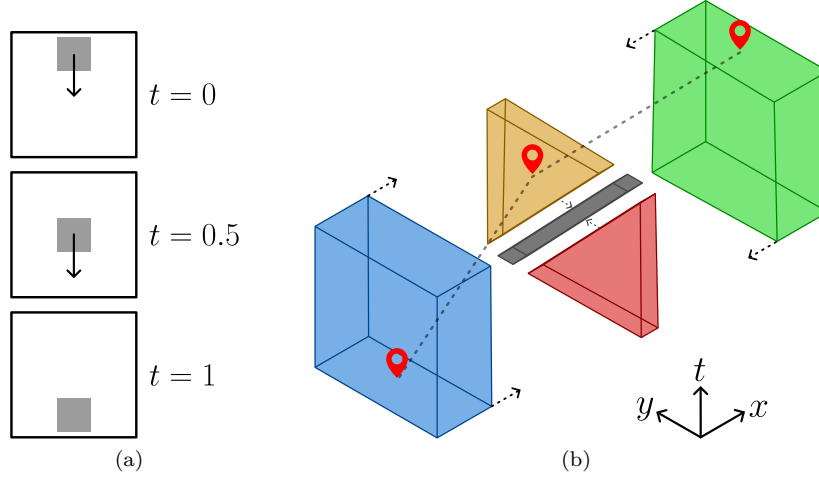


Fig. 1: 在不同的时间点展示了动态障碍物如 $\mathbb{R}^2$ (a)。空间和时间中的安全区域被分割成凸集图 (GCS)，然后通过图生成无碰撞轨迹 (b)。

以前的工作没有提供这些约束的明确、易于访问的表述，进一步阻碍了采用。本文旨在通过提出详细的方法来公式化和实施这些约束，以填补这一空白，从而促进 GCS 框架的更广泛使用。

Werner 等人在障碍物规避方面的一次显著尝试是 [26]，他们重新计算凸集的图，并在每个时间步重新生成轨迹。为了加速这个过程，他们的方法需要 GPU 加速的图形计算。然而，该方法没有将障碍物运动的任何先验知识融入规划中，并且仍然依赖于非线性优化步骤来应用路径的时间规划——因此需要良好的初始猜测。相比之下，Marcucci 等人 [21] 通过连接两个机器臂的配置空间提出联合规划。虽然这种方法对少量代理有效，但由于组合配置空间的指数增长而扩展性差，从而限制了其适用性。此外，因为在现实世界的场景中，代理通常无法控制其他物体的运动，所以这些方法不适用于通用的动态障碍规避。

迄今为止最有前景的障碍物避让 GCS（生成协调结构）公式是 Tang 等人提出的，他们最近引入了用于多机器人运动规划的凸集时空图 (ST-GCS) 框架。在该公式中，配置空间被增强了时间维度，使得可以以与 GCS 框架兼容的方式表示恒速障碍物。他们的工作仅考虑涉及在空间中初始和最终点之间的联合规划的其他代理。该方法优先考虑最短时间的解决方案，并结合了一种机制，用于“保留”时空轨迹作为后续代理

的动态障碍。

这种方法的局限性在于现有的 GCS 框架要求在图中的只有一个凸集可以与最终轨迹状态相交。因此，Tang 等人保守地选择最终集合，这可能会阻止框架找到真实的最小时间轨迹，即使它们存在。此外，他们的公式仅限于规划分段常速度轨迹，而未应用于高阶样条。在时空配置空间内确保因果关系的重要约束存在，但其重要性未被讨论或阐明。

在本文中，我们使用 ST-GCS 框架生成无碰撞的最小距离轨迹，并具有固定的结束时间，从而能够与静态设置中的标准 GCS 公式进行直接比较。最小距离成本函数使我们能够验证该框架在动态场景中的适用性，并评估其在时间变化环境中的性能和局限性。图 1 展示了 ST-GCS 公式，描述了一个移动的二维物体及时间扩展配置空间中生成的凸集合图。

首先，我们提供了一种系统的方法，并附有逐步的例子，用于将常见的约束重新写成 GCS 框架所需的形式。其次，基于 ST-GCS 框架，我们研究了在动态 2D 环境中生成最小距离、无碰撞轨迹的方法。通过假设已知的障碍物位置和恒定的障碍物速度，我们的方法无需依赖非线性求解器、初始时间猜测或产生不必要的计算复杂性，就能生成更高阶的最优样条轨迹。

本文的其余部分结构如下。第 II 节概述了凸集图和 Bézier 样条的基础数学。第 III 节详细介

绍了我们的方法，并给出了构建 GCS 兼容约束的示例。第 IV 节展示了在动态环境中使用 GCS 优化轨迹的结果。最后，第 V 节总结了本文并讨论了未来工作的方向。

II. 背景

GCS 优化框架为最终用户提供了强有力的保证，其根植于凸分析和优化理论中的高级概念。在本节中，我们将重点介绍理解 GCS 高层次的关键概念。若需更全面地了解 GCS，我们建议读者参考 [19]–[27]。

A. 凸工具

顾名思义，凸集是 GCS 框架的基础。在本小节中，我们介绍与凸集相关的关键定义和方程，但建议读者参考 [27] 的第二章和第三章，以获得对该主题的更全面的理解。

一个集合 $\mathcal{S} \subseteq \mathbb{R}^n$ 是凸的，如果它内部的任意两点间所连接的直线段完全位于 $\mathcal{S}$ 内。凸性可以通过取两个点 $a, b \in \mathcal{S}$ 并展示这两点之间的线性插值总是位于集合 $\mathcal{S}$ 内来证明。形式上，这被表达为：

$$(1 - \lambda)a + \lambda b \in \mathcal{S} \quad \forall a, b \in \mathcal{S} \quad \lambda \in [0, 1]. \quad (1)$$

我们在实践中遇到的集合通常不是凸的。集合 $\mathcal{S}$ 的凸包，记为 $\text{conv}(\mathcal{S})$ ，是包含 $\mathcal{S}$ 的最小凸集。在图 2 中展示了这一点。如果 $\mathcal{S}$ 已经是凸的，那么 $\text{conv}(\mathcal{S}) = \mathcal{S}$ 。

在凸优化中，另一个重要的概念是锥。一个集合 $\mathcal{S}$ 被称为锥（或称为非负齐次的），如果对于所有的 $x \in \mathcal{S}$ 和 $\lambda \geq 0$ ，我们有 $\lambda x \in \mathcal{S}$ （如图 3 所示）。如果一个集合既是凸的又是锥的，它就是一个凸锥。

值得注意的是，一个集合可以是一个锥体而不具有凸性。例如，考虑到 $\mathbb{R}^2$ 的第一和第三象限的并集。这个集合是一个锥体，因为 $\mathbb{R}^2$ 的第一和第三象限中的任何向量都可以通过一个非负数进行缩放而仍然在集合内，但它不是凸的。我们将在本文中仅使用凸锥体。

凸组合是我们在将约束转换为 GCS 框架时使用的另一个有用工具。一个点 $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_m \in \mathbb{R}^n$ 的凸组合 x_c 定义为

$$\vec{x}_c = \sum_{i=1}^m \vec{x}_i \lambda_i \quad \text{where} \quad \lambda_i \geq 0, \quad \sum_{i=1}^m \lambda_i = 1. \quad (2)$$

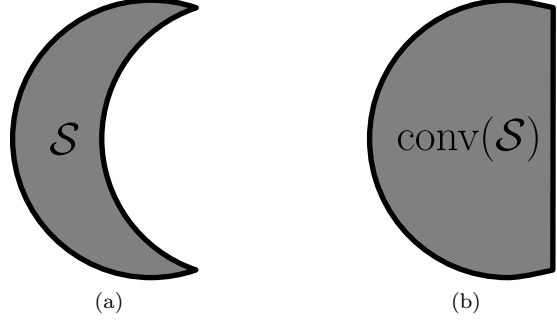


Fig. 2: 一个在 (a) 中显示的非凸集合 $\mathcal{S}$ 及其在 (b) 中显示的相关凸包 $\text{conv}(\mathcal{S})$ 。

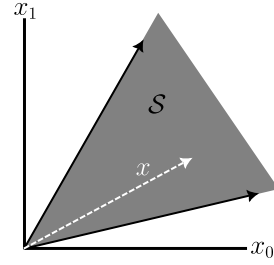


Fig. 3: 如果 $\mathcal{S}$ 是一个锥体，向量 $x \in \mathcal{S}$ 在被非负常数缩放后仍然会在 $\mathcal{S}$ 中。

，这保证了 $\vec{x}_c$ 位于 $\text{conv}(\{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_m\})$ 内。非负系数的线性组合可以通过除以系数之和（假设总和为正）规范化为凸组合。具体而言，如果

$$\lambda \vec{x}_c = \sum_{i=1}^m \vec{x}_i \lambda_i \quad \text{where} \quad \lambda_i \geq 0, \quad \lambda = \sum_{i=1}^m \lambda_i \quad (3)$$

，则

$$\vec{x}_c = \frac{1}{\lambda} \sum_{i=1}^m \vec{x}_i \lambda_i \quad \text{where} \quad \lambda_i \geq 0, \quad \lambda > 0, \quad \frac{1}{\lambda} \sum_{i=1}^m \lambda_i = 1. \quad (4)$$

。这种关系在我们推导并解释方法部分的 GCS 兼容的连续性和可微性约束时会很有帮助。

GCS 框架中使用的另一个重要工具是闭集均匀化的闭包。给定一个闭集 $\mathcal{S} \in \mathbb{R}^n$ ，该集合均

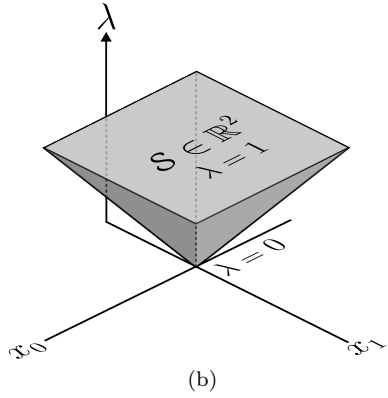
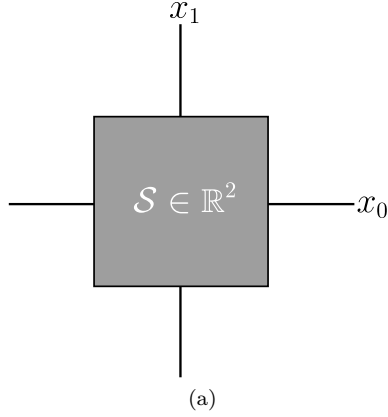


Fig. 4: 集合 $S \in \mathbb{R}^2$ (a) 以及 S (b) 同质化后的闭包。 $\text{clh}(S)$ 在 $\mathbb{R}^3$ 中形成一个锥形。

匀化的闭包，记为 $\text{clh}(S)$ ，使我们能够在 $\mathbb{R}^{n+1}$ 中构建一个锥（见图 4）。 $\text{clh}()$ 运算定义为

$$\text{clh}(S) := \{\text{concat}(\vec{x}, \lambda) \in \mathbb{R}^{n+1} : \lambda \geq 0, \vec{x} \in \lambda S\}, \quad (5)$$

其中 $\text{concat}(\vec{x}, \vec{v})$ 将向量 $\vec{x} \in \mathbb{R}^n$ 和 $\vec{v} \in \mathbb{R}^m$ 连接为 $\mathbb{R}^{n+m}$ 。通过在 $\lambda = 1$ 处取横截面，可以恢复原始集合。如果 S 是凸的，那么 $\text{clh}(S)$ 也是凸的 [19]。在 GCS 文献中，约束使用集合表示，通常写作 $\mathcal{X}$ ，并通过 $\text{clh}(\mathcal{X})$ 表示在整个图中应用。

B. 图论

我们记一个图为 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ ，其中 $\mathcal{V} = \{v_0, v_1, \dots, v_m\}$ 是顶点集， $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ 是边集。当 $\mathcal{E}$ 由有序对组成时，图是有向的。我们将使

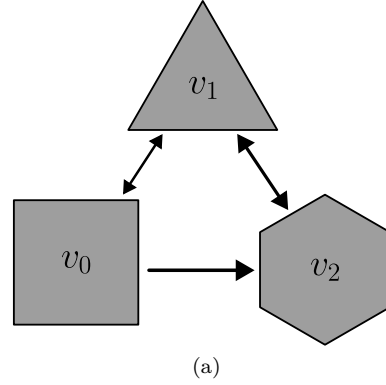


Fig. 5: 凸集的有向图。

用有序对 (v_a, v_b) 表示从 v_a 到 v_b 的有向连接。有些方程适用于图中每一条有向边。在这种情况下，我们将图中的一个普通边记为 e ，

凸集图是一个有向图，其中每个顶点都与一个凸集相关联。正式地，它被写为 $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ ，其中 $\mathcal{A} = \{A_{v_0}, A_{v_1}, \dots, A_{v_m}\}$ 是一个凸集的集合，每个集合 A_{v_i} 被分配给顶点 $v_i \in \mathcal{V}$ 。这些凸集可以存在于任意维度，并可以编码各种各样的问题结构。值得注意的是，存在一条边 $(v_a, v_b) \in \mathcal{E}$ 并不意味着对应的集合 A_{v_a} 和 A_{v_b} 在 $\mathbb{R}^n$ 中相交或重叠。

图 5 显示了凸集的一个示例图。该图是有向的，每个顶点都与 $\mathbb{R}^2$ 中的不同凸形状相关联。在该图示中，这些凸集在 $\mathbb{R}^2$ 中没有接触或重叠。

C. 凸集图优化

GCS 优化框架结合了连续优化和离散图搜索的优点。在这个框架中，为每个凸集分配一个参数化曲线。在优化过程中，未访问顶点对应的曲线会被舍弃，而访问的顶点处的曲线则在集合边界处连接，形成一个连续的轨迹。与传统图搜索不同，在传统图搜索中，边或顶点的代价是一个固定值，而在这里，代价随着曲线在各自的凸集内变化而连续变化。通过图中的路径代价最小化可以使用目标函数来表述：

$$\min_x \sum_{v \in \mathcal{V}} f_v(\vec{x}_v) + \sum_{e \in \mathcal{E}} f_e(\vec{x}_a, \vec{x}_b) \quad (6a)$$

s.t.

$$(\mathcal{V}, \mathcal{E}, \mathcal{A}) \subseteq \mathcal{G} \quad (6b)$$

$$\vec{x}_v \in \mathcal{X}_v \quad \forall v \in \mathcal{V} \quad (6c)$$

$$\vec{x}_a \in \mathcal{X}_e \quad \forall e \in \mathcal{E} \quad (6d)$$

$$\vec{x}_b \in \mathcal{X}_e \quad \forall e \in \mathcal{E}. \quad (6e)$$

这里, x 表示定义轨迹曲线段的所有连续变量。变量 $\vec{x}_v$ 是与顶点 v 相关联的设计变量。顶点成本函数 $f_v(\vec{x}_v)$ 代表顶点内连续变量的成本。边缘成本函数 $f_e(\vec{x}_a, \vec{x}_b)$ 表示从 v_a 到 v_b 的边缘传递的成本, 并耦合连接顶点的连续变量。构成最优轨迹的顶点 $\mathcal{V}$ 、边 $\mathcal{E}$ 和相关的凸集 $\mathcal{A}$ 是最小化整体成本的图 $\mathcal{G}$ 的一个子集。集合 $\mathcal{X}_v$ 和 $\mathcal{X}_e$ 定义了设计变量必须满足的顶点和边缘约束。

这种表述的一个局限性是, 它在优化过程中不提供寻找 $(\mathcal{V}, \mathcal{E}, \mathcal{A}) \subseteq \mathcal{G}$ 的方法。为了解决这个问题, 可以通过引入二进制变量来表示最终解中使用的顶点和边, 将该问题重新表述为一个混合整数规划 (MIP)。二进制变量 $y_v \in \{0, 1\}$ 表示一个顶点是否包含在解中, 而二进制变量 $y_e \in \{0, 1\}$ 表示一条边是否被遍历。

二元变量被乘入方程 6 中的成本和约束项。这样可以确保只有当遍历顶点或边时, 成本和约束才是活跃的, 而当顶点或边不在最终解中时, 约束将被去除。MIP 公式写为:

$$\begin{aligned} \min_{\vec{x}, y} \quad & \sum_{v \in \mathcal{V}} y_v f_v(\vec{x}_v) \\ & + \sum_{e \in \mathcal{E}} y_e f_e(\vec{x}_a, \vec{x}_b) \end{aligned} \quad (7a)$$

s.t.

$$y_v \in \{0, 1\} \quad (7b)$$

$$y_e \in \{0, 1\} \quad (7c)$$

$$y_v \vec{x}_v \in y_v \mathcal{X}_v \quad \forall v \in \mathcal{V} \quad (7d)$$

$$y_e \vec{x}_a \in y_e \mathcal{X}_e \quad \forall e \in \mathcal{E} \quad (7e)$$

$$y_e \vec{x}_b \in y_e \mathcal{X}_e \quad \forall e \in \mathcal{E}. \quad (7f)$$

随着图的增长, 二元变量的组合性质使得这个 MIP 变得不可直接解决。为了解决这个问题, 二元变量被放宽为连续变量 $y \in [0, 1]$, 并使用分支定界算法反复解决该问题。在这种情况下, 放宽的变量表示从轨迹的起点到终点通过图的流动。流动变量受到限制, 以确保在所有中间顶点的流动守恒。虽然问题仍然是 NP 难的, 这种放宽是紧的, 并显著提高了解决的可行性。

为处理双线性项 $y_v \vec{x}_v$ 和 $y_e \vec{x}_a, y_e \vec{x}_b$, 应用了变量替换: $\vec{z}_v = y_v \vec{x}_v$ 和 $\vec{z}_{e,v} = y_e \vec{x}_v$ 。假设 f_v

和 f_e 是凸函数, $\mathcal{X}_v$ 和 $\mathcal{X}_e$ 是凸集, 这种替换将问题转化为混合整数凸规划 (MICP), 这大大降低了每次分支定界迭代的计算时间。放松的 MICP 可以写成:

$$J(\vec{z}, y) = \sum_{v \in \mathcal{V}} y_v f_v \left(\frac{\vec{z}_v}{y_v} \right) + \sum_{e \in \mathcal{E}} y_e f_e \left(\frac{\vec{z}_a}{y_a}, \frac{\vec{z}_b}{y_b} \right) \quad (8a)$$

$$\min_{\vec{z}, y} J(\vec{z}, y) \quad (8b)$$

s.t.

$$y_v \in [0, 1] \quad (8c)$$

$$y_e \in [0, 1] \quad (8d)$$

$$\sum y_{(a,v)} + \delta_s$$

$$= \sum y_{(v,b)} + \delta_f = y_v \quad \forall v \in \mathcal{V} \quad (8e)$$

$$\vec{z}_v \in y_v \mathcal{X}_v \quad \forall v \in \mathcal{V} \quad (8f)$$

$$\vec{z}_{e,a} \in y_e \mathcal{X}_e \quad \forall e \in \mathcal{E} \quad (8g)$$

$$\vec{z}_{e,b} \in y_e \mathcal{X}_e \quad \forall e \in \mathcal{E}. \quad (8h)$$

方程 (8e) 引入了流量守恒约束。变量 $y_{(a,v)}$ 表示所有进入顶点 v 的边变量, $y_{(v,b)}$ 表示所有离开顶点 v 的边变量。如果 v 是起始顶点, 则值为 $\delta_s = 1$, 否则为 $\delta_s = 0$ 。如果 v 是终点顶点, 则值为 $\delta_f = 1$, 否则为 $\delta_f = 0$ 。在包含轨迹起始和结束的顶点中, 这些创建了图中的源和汇。在其他文献中, 变量 z 和 y 在方程 (8f)、(8g) 和 (8h) 中被连接, 并使用 $\text{clh}(\mathcal{X})$ 表达约束。为了清晰和与代码实现的一致性, 我们省略了这种表示法, 并保留了此处写的约束。对于所得集合的几何解释, 我们请读者参考图 4。

先前的 GCS 文献表明, 方程 (8) 中的 MICP 形式可以用于找到全局最优轨迹。通过一个分支定界算法迭代地解决凸松弛问题, 逐步对 y 的值添加约束, 直到找到全局最优解。我们将利用这种公式找到全局最优的无碰撞轨迹。

D. 生成凸集的图

生成凸集的方法有多种。在这项工作中, 我们使用半定规划的迭代区域膨胀 (IRIS) 算法, 如 [28] 中所述。该算法是之前 GCS 文献中生成凸集的标准方法。

IRIS 算法首先在配置空间中采样一个点。然后, 该算法通过迭代添加半空间平面来定义一个凸集, 并增加拟议凸集内一个内切椭球的体积,

直至体积达到最大。如果采样点位于障碍物内, 则会被拒绝, 并且不创建凸集。

IRIS 算法可以被配置为生成相交或仅在边界接触的凸集。一旦自由空间被充分采样, 每个集合将在图中与一个顶点关联, 图的边则根据用户定义的标准创建。在这项工作中, 我们生成在边界上接触的集合, 以避免起点和终点包含在多个集合中的情况。边界上接触的集合被连接以形成图的边。

这种方法的优点在于, 它可以在高于 $\mathbb{R}^3$ 维度的空间中生成无碰撞的凸集, 比如一个 7 自由度机器人手臂的配置空间。然而, 其主要缺点是需要对配置空间进行采样, 并且不保证完全覆盖自由空间。因此, 使用该图计算的轨迹相对于图来说是全局最优的, 但相对于整个自由空间不一定是全局最优的。

E. 贝塞尔样条

轨迹通常使用称为样条的分段多项式函数来表示。有许多类型的样条, 例如 B 样条, 最小体积样条和 Bézier 样条。之前的 GCS 论文已经使用 Bézier 样条来表示代理的轨迹, 为了一致性, 我们也使用 Bézier 样条。

贝塞尔样条在计算上对采样和修改是高效的。它们的导数可容易地作为定义轨迹的控制点的线性组合来计算, 并且本身也是低阶样条。因为轨迹经过贝塞尔曲线的第一个和最后一个控制点, 连续性和可微性约束在许多应用中直观且易于实施。

n^{th} 阶贝塞尔曲线的矢量值方程为

$$\vec{B}(s) = \sum_{k=0}^n \binom{n}{k} s^k (1-s)^{n-k} \vec{x}_k, \quad 0 \leq s \leq 1, \quad (9)$$

, 其中 $\vec{x}_k$ 是曲线的第 k 个矢量值控制点, $\binom{n}{k}$ 是二项式系数。 n^{th} 阶曲线有 $n+1$ 个控制点。单个贝塞尔曲线组合成分段函数以创建贝塞尔样条, 它可以表示为

$$\vec{P}(r) = \begin{cases} \vec{B}_0(r) & 0 \leq r < 1 \\ \vec{B}_1(r-1) & 1 \leq r < 2 \\ \vdots & \\ \vec{B}_n(r-l) & l \leq r < l+1, \end{cases} \quad (10)$$

, 其中 $r \in [0, n]$ 是沿样条插值的参数, l 表示单个贝塞尔曲线段的数量。

贝塞尔样条具有凸包性质 [29], 这保证了贝塞尔曲线永远不会离开其控制点的凸包 (参见图 6)。在这个图示中, 控制点集 $C = \{x_0, x_1, x_2, x_3\}$ 定义了一条贝塞尔曲线, 该曲线永远不会离开 $\text{conv}(C)$ 。这个性质常用于碰撞避免算法中以确保安全性。

在 GCS 框架中, 每个图的顶点都分配了一个 Bézier 曲线段。控制点被限制在其关联的闭合凸集的内部或边界上, 确保样条不能离开该集合。当 MICP 问题被反复求解时, 曲线段最终连接形成最终的样条解决方案。图中未使用的部分, 及其相关的 Bézier 曲线被丢弃, 留下穿过凸集合图的单一连续样条。

图 7a 说明了一个未解决的 GCS 轨迹问题。凸集以不同深浅的灰色显示, 中央的障碍物用黑色表示。星星标记了起始和终点位置, 单独的贝塞尔曲线用红色绘制, 小黑圆表示贝塞尔控制点。图 7b 显示了解决的轨迹, 其中贝塞尔曲线段连接形成从起点到目标的连续路径, 未使用的曲线被舍弃。

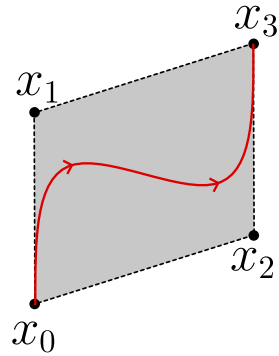


Fig. 6: 当 $s \in [0, 1]$ 时, 样条 (红色显示) 保持在其控制点的凸包内。

III. 方法

本节介绍了我们使用 GCS 框架生成最优无碰撞轨迹的方法。我们首先描述如何生成凸集以表示环境中的安全区域。然后, 我们概述了一种制定和实施 GCS 兼容约束的一般策略, 解决了顶点和边的约束。我们介绍了用于时空配置空间中的轨迹规划的特定约束, 包括因果关系和运动学约束, 并描述了障碍物在时空配置空间中的表示方法。最后, 我们以能够在二维中实现时间相关轨迹的完整优化公式作为结尾。

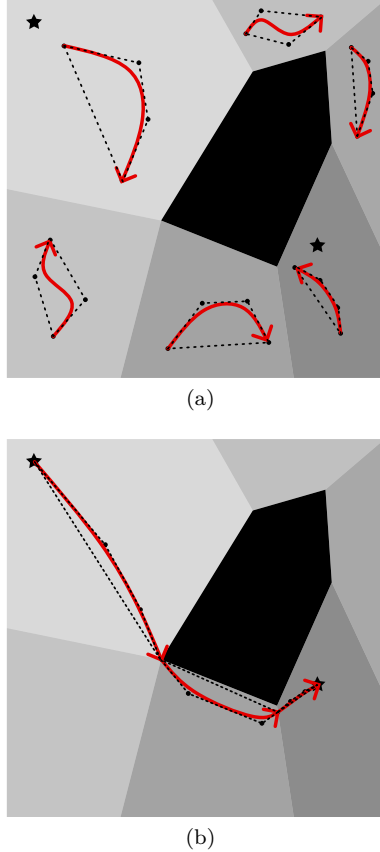


Fig. 7: GCS 中的集合关联的 Bézier 曲线 (a) 以及将 Bézier 曲线连接起来形成从开始到结束的样条的 GCS 问题的解决方案 (b)。未使用的曲线被移除。

A. GCS 约束

在 GCS 框架中，约束的公式化和实现，特别是那些跨越图中边缘的约束，是最具挑战性的方面之一。虽然之前的工作证明，连续性和可微性的约束可以在 GCS 框架内成功实现，但现有文献通常缺乏这些约束的精确或易于理解的公式化。特别是，之前的工作展示了必须实现满足方程 (8g) 和 (8h) 的约束的结果，但从未展示如何做到这一点，除了声明集合 $\mathcal{X}$ 代表约束之外。我们旨在通过提供这些约束的详细推导，以及提供一个可重复的逐步策略来在图的边缘上应用约束，以填补这一空白。

诸如样条的连续性和可微性之类的平凡约束，

在混合整数规划的凸松弛中变得不平凡（见方程 (8)）。凸松弛引发了一些问题，例如：当连接它们的边仅部分激活时（例如，当 $y_e = 0.5$ 时），相邻顶点的两个 Bézier 样条连续意味着什么？如果多个连接边同时处于半激活状态怎么办？样条段如何同时连接到多个段？

设计与 GCS 兼容的边约束需要不同的方法，因为约束必须根据边是处于激活、半激活还是未激活状态进行调整。当边是激活状态（ $y_e = 1$ ）时，约束必须被完全实施。例如，在样条曲线连续性中，这意味着相邻集合中的两曲线形成连续的分段函数。当边是半激活状态（ $0 < y_e < 1$ ）时，约束必须允许图中可能连接之间的平滑过渡。最后，当边是未激活状态（ $y_e = 0$ ）时，约束的两侧应该变为零，实际上解除约束，这样约束就不会影响设计变量或它们在其他边上的交互。设计符合这三种行为的约束同时保持连续性是一个具有挑战性的任务。

我们编写与 GCS 兼容的边约束的方法总结在算法 1 中。第一步是将约束编写成在解子图中对单个边有效的形式。然后，将方程的两边乘以代表该边是否有效的决策变量。接着，流出单个顶点的所有边的约束相加并简化，得到潜在约束的一个凸组合。然后，将双线性乘积替换为双线性变量 z 。最后，对图中的每个顶点重复该约束，但排除汇点。

Algorithm 1 为 GCS 编写边缘约束

- 1: Write constraint C for a single edge, in terms of the vertex control points x_v and the connecting control points $\vec{x}_b$.
- 2: Multiply both sides of the constraint by the edge decision variable y_e .
- 3: Sum the constraints for all edges connected to the vertex v .
- 4: Simplify the expression using the relationship in Equation (8e) assuming that $\delta_f = 0$, $y_v = \sum y_{(v,b)}$.
- 5: Substitute bilinear products with the variables $\tilde{z}_v = y_v \vec{x}_v$ and $\tilde{z}_{e,v} = y_e \vec{x}_v$.
- 6: Repeat the constraint for each vertex $v \in \mathcal{V}, v \neq \text{sink}$.

以下几个小节将逐步推导组成方程 8f、8g 和 8h 的约束条件。

B. 凸集约束

像许多其他 GCS 应用一样，我们依赖凸包性质来保证整个轨迹每个点的安全性。回想一下，图中的每个顶点都与一个凸集合相关联，在该集合内定义了一段贝塞尔曲线。因为贝塞尔曲线始终保持在其控制点的凸包内，我们可以保证曲线在安全区域内，如果我们限制控制点位于相关的凸集合内。这种约束将被写成与公式 8f 兼容的形式。

一个多边形凸集可以表示为有限多个半空间的交集。设 $\hat{n}_i$ 表示 i^{th} 平面的法向量， p_i 是位于该平面上的一点。如果 $\hat{n}_i \cdot \vec{r} \leq \hat{n}_i \cdot p_i$ ，则通用点 $\vec{r}$ 位于半空间平面上或其前。我们可以将每个面法向量作为行向量堆叠在矩阵 A 中，并将相应的点积 $\hat{n}_i \cdot p_i$ 堆叠成一个向量 $\vec{d}$ ，通过检查由 $A\vec{r} \leq \vec{d}$ 形成的线性不等式来确定点 $\vec{r}$ 是否位于凸集中。

变量 $\vec{x}_{v,i}$ 表示与顶点 v 相关联的曲线中的 i^{th} 控制点。线性不等式 $A_v \vec{x}_{v,i} \leq \vec{d}_v$ 应用于每组中的每个控制点。用双线性变量 $\vec{z}_{v,i} = y_v \vec{x}_{v,i}$ 重新表述这个不等式，我们得到方程

$$A_v \frac{\vec{z}_{v,i}}{y_v} \leq \vec{d}_v \quad (11)$$

，可以重新排列为

$$A_v \vec{z}_{v,i} \leq \vec{d}_v y_v. \quad (12)$$

C. 连续性约束

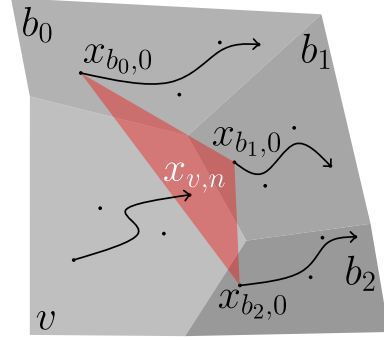
我们首先使用算法 1 推导连续性约束。我们的目标是使一条曲线的最后一个控制点在边完全激活时与连接曲线的第一个控制点匹配。此外，该约束应允许当边仅部分激活时，不同连接之间的连续转换。因为这个约束需要图中边相关的变量，所以以下约束形式属于方程 8g 和 8h 的类别。我们从为图中的单个通用边写约束开始：

$$\vec{x}_{v,n} = \vec{x}_{b,0}. \quad (13)$$

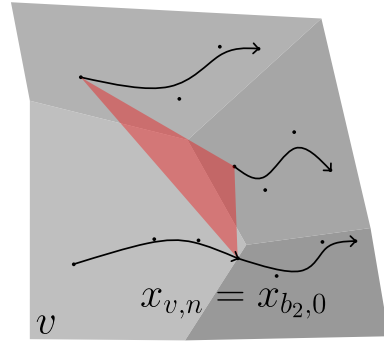
这里， $\vec{x}_{v,n}$ 表示集合 v 中当前曲线的最后一个控制点，而 $\vec{x}_{b,0}$ 表示连接集合 b 的第一个控制点。

然后我们将方程两边乘以连接集合 v 和集合 b 、 $y_{(v,b)}$ 的边决策变量，使得当边被纳入解中时，约束完全激活，否则不激活：

$$y_{(v,b)} \vec{x}_{v,n} = y_{(v,b)} \vec{x}_{b,0}. \quad (14)$$



(a)



(b)

Fig. 8: 一个顶点 v 与相邻顶点 b_i 。连续性约束确保最后的 Bézier 控制点落在连接边的第一个 Bézier 控制点的凸包内 (a)，然后连接两条边如 $y_{v,b_i} \rightarrow 1$ (b)。

接下来，我们对所有连接到顶点 v 的边求和和这些约束：

$$\sum_b y_{(v,b)} \vec{x}_{v,n} = \sum_b y_{(v,b)} \vec{x}_{b,0}. \quad (15)$$

假设来自方程 8 e 的流量守恒约束成立，左侧简化为：

$$y_v \vec{x}_{v,n} = \sum_b y_{(v,b)} \vec{x}_{b,0}. \quad (16)$$

回归到方程 (3) 和 (4)，这个约束确保一组的最后一个控制点位于连接集合的第一个控制点的凸包内，按其对应的决策变量 $y_{(v,b)}$ 加权。当应用于要求所有控制点保持在各自凸集内的约束时，它确保在 $y = 1$ 时，控制点在集合之间

的边界相遇，因为这是唯一一个同时满足两个约束的地方。这在边处于活跃状态时强制曲线段之间的连续性，并在凸松弛中实现平滑过渡。如果所有边和顶点都处于非活跃状态，约束没有效果。因为这个表述将一个顶点与其外向边关联起来，它被应用于除汇点以外的每个顶点，汇点通过图中的其他边约束隐含捕获。或者，该约束可以使用内向边表述，在这种情况下则排除源顶点。图 8 展示了这个约束。

然后我们替代双线性变量 $\vec{z}_{v,n} = y_v \vec{x}_{v,n}$ 和 $\vec{z}_{(v,b),0} = y_{(v,b)} \vec{x}_{b,0}$ ，将约束重写为：

$$\vec{z}_{v,n} = \sum_{i=0}^n \vec{z}_{(v,b),i}. \quad (17)$$

D. 可微性约束

接下来，我们推导可微性约束，以确保样条在曲线连接点处关于插值变量 r 是可微的（见方程 (10)）。与连续性约束类似，可微性约束将属于方程 8g 和 8h 的类别。

为了使两条曲线连接处的路径可导，其中一条曲线的最后两个控制点之间的向量必须与连接曲线的前两个控制点之间的向量相等。对于边 (v,b) ，该约束可以写为：

$$\vec{x}_{v,n} - \vec{x}_{v,n-1} = \vec{x}_{b,1} - \vec{x}_{b,0}. \quad (18)$$

如前所述，我们将等式的两边都乘以边决策变量 $y_{(v,b)}$ ，当边在解中时，此约束变为有效。该方程可以写为：

$$y_{(v,b)}(\vec{x}_{v,n} - \vec{x}_{v,n-1}) = y_{(v,b)}(\vec{x}_{b,1} - \vec{x}_{b,0}). \quad (19)$$

然后我们将这些约束对所有连接到顶点 v 的边求和：

$$\sum_b y_{(v,b)}(\vec{x}_{v,n} - \vec{x}_{v,n-1}) = \sum_b y_{(v,b)}(\vec{x}_{b,1} - \vec{x}_{b,0}). \quad (20)$$

假设方程 8e 中的流量守恒约束成立，左侧简化为：

$$y_v(\vec{x}_{v,n} - \vec{x}_{v,n-1}) = \sum_b y_{(v,b)}(\vec{x}_{b,1} - \vec{x}_{b,0}). \quad (21)$$

然后我们替换双线性变量 $\vec{z}_{v,0} = y_v \vec{x}_{v,n}$ 、 $\vec{z}_{v,n-1} = y_v \vec{x}_{v,n-1}$ 、 $\vec{z}_{(v,b),1} = y_{(v,b)} \vec{x}_{b,1}$ 和 $\vec{z}_{(v,b),0} = y_{(v,b)} \vec{x}_{b,0}$ ，将约束重写为：

$$\vec{z}_{v,n} - \vec{z}_{v,n-1} = \sum_b (\vec{z}_{(v,b),1} - \vec{z}_{(v,b),0}). \quad (22)$$

然而，这种表述效率低下，因为它需要为每个有向边定义两个 z 变量，导致了 $|\mathcal{E}| \times 2$ 个额外的变量来强制执行这一约束，其中 $|\mathcal{E}|$ 表示图的边的基数。为了减少变量的数量，我们引入了一个最终的替代变量 $\vec{z}_{(v,b_i),\text{diff}} = \vec{z}_{(v,b_i),1} - \vec{z}_{(v,b_i),0}$ 。这使我们能将可微性约束表达为：

$$(\vec{z}_{v,n} - \vec{z}_{v,n-1}) = \sum_{i=0}^n \vec{z}_{(v,b_i),\text{diff}}. \quad (23)$$

这种表述仅需要 $|\mathcal{E}|$ 个额外的变量来确保可微性，从而加速优化过程。

E. 时空配置约束

我们现在引入时空配置约束，这对于在时空配置空间中生成现实的轨迹至关重要。在时空配置空间中，可以构造出非因果的样条并导致无限的速度。为了防止这种情况发生，我们施加一个约束，以强制因果性并限制样条的最大速度。这个约束将以一种与方程 8f 兼容的方式书写。

样条 v_{spline} 在控制点之间的最大速度可以通过控制点的 x 、 y 和 t 组件的变化来界定，因此

$$v_{\text{spline}} \leq \frac{\|(\vec{x}_{v,i+1} - \vec{x}_{v,i})\|_{xy,2}}{(\vec{x}_{v,i+1} - \vec{x}_{v,i})_t} \leq v_{\text{max}} \quad (24)$$

$$(\vec{x}_{v,i+1} - \vec{x}_{v,i})_t \geq 0 \quad (25)$$

，其中 v_{max} 是最大允许速度。符号 $\|\cdot\|_{xy,2}$ 表示仅针对控制点的 x 和 y 组件的二范数，而 $(\cdot)_t$ 是在括号内的时间组件的值。

为了使这个约束与凸求解器兼容，我们可以将约束重新排列为：

$$\|(\vec{x}_{v,i+1} - \vec{x}_{v,i})\|_{xy,2} \leq v_{\text{max}}(\vec{x}_{v,i+1} - \vec{x}_{v,i})_t \quad (26)$$

$$(\vec{x}_{v,i+1} - \vec{x}_{v,i})_t \geq 0 \quad (27)$$

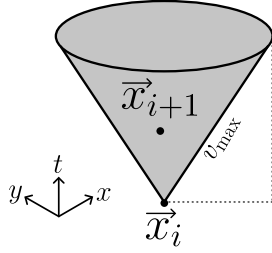


Fig. 9: 速度约束在 $\mathbb{R}^3$ 中形成一个圆锥。每个控制点必须位于前一个控制点形成的圆锥内，以确保斜率保持在最大速度以下。

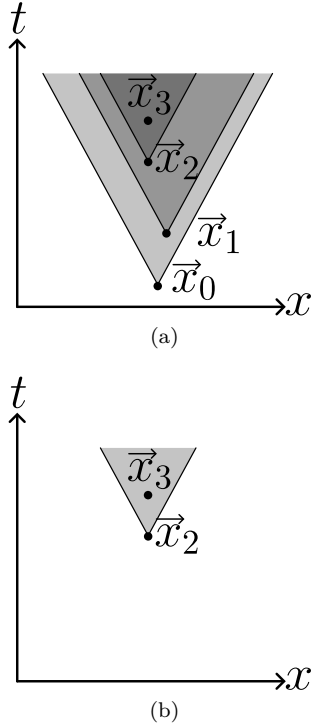


Fig. 10: 所有速度锥约束的交集仍然是凸的 (a)。最终控制点必须位于该交集内，这相当于仅在倒数第二个控制点周围放置一个速度锥 (b)。

替换双线性变量 $\bar{z}_{v,i}$ ，这变成：

$$\left\| \left(\frac{\bar{z}_{v,i+1}}{y_v} - \frac{\bar{z}_{v,i}}{y_v} \right) \right\|_{xy,2} \leq v_{\max} \left(\frac{\bar{z}_{v,i+1}}{y_v} - \frac{\bar{z}_{v,i}}{y_v} \right)_t \quad (28)$$

$$\left(\frac{\bar{z}_{v,i+1}}{y_v} - \frac{\bar{z}_{v,i}}{y_v} \right)_t \geq 0. \quad (29)$$

以下约束

$$\left(\frac{\bar{z}_{v,i+1}}{y_v} - \frac{\bar{z}_{v,i}}{y_v} \right)_t \geq \epsilon \quad (30)$$

可以替代以确保控制点保持最小时间间隔。

最后，简化 y_v 得到最终形式的约束：

$$\|(\bar{z}_{v,i+1} - \bar{z}_{v,i})\|_{xy,2} \leq v_{\max}(\bar{z}_{v,i+1} - \bar{z}_{v,i})_t \quad (31)$$

$$(\bar{z}_{v,i+1} - \bar{z}_{v,i})_t \geq y_v \epsilon. \quad (32)$$

这个约束的几何直观是它在 $\mathbb{R}^3$ 中形成了一个圆锥，如图 9 a 所示。由于凸包性质，贝塞尔曲线的瞬时斜率绝不会超过两个相邻控制点之间的斜率。因此，如果两个控制点之间的斜率小于最大速度，曲线将永远不会违反此约束。为了确保满足最大速度约束，我们必须确保连接相邻控制点位于由前一个控制点形成的圆锥内。如图 9 所示。

虽然对于任意一对相邻控制点来说，这个约束显然是凸的，但合并所有相邻对的约束后是否仍然保持凸性可能并不明显。图 10 说明了最终控制点的可行区是所有速度锥约束的交集。这些可行区的交集相当于仅取前一个控制点处的速度约束。

F. 时空障碍

在 $\mathbb{R}^2$ 中移动的障碍物可以通过将其顶点的初始位置与初始时间连接起来，然后将最终位置与最终时间连接起来，并最终连接所有顶点以形成空间时间 $\mathbb{R}^3$ 配置中的障碍物来表示。图 11 说明了这个过程。该方法假设障碍物是静止的或以恒定速度移动。更复杂的运动需要更精细的采样，并且只会在 $\mathbb{R}^3$ 中近似其真实运动。

G. 成本

实现时空 GCS 公式所需的最后一个组成部分是成本函数。选择顶点成本函数 f_v 和边成本函数 f_e 时很重要，以便在引入决策和松弛变量时

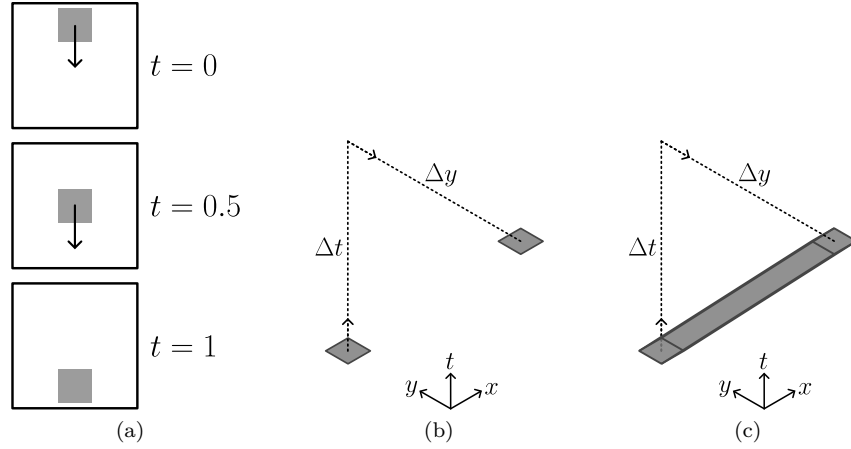


Fig. 11: 动态障碍物在不同时间点如 $\mathbb{R}^2$ (a) 所示。顶点的初始和最终位置被放置在时空配置空间中 (b)。这些顶点被连接起来以形成一个形状，显示其在 $\mathbb{R}^3$ (c) 中的轨迹。

整体目标保持凸性并简化，如方程 8 所示。在这项工作中，我们的目标是在 (x, y) 平面上最小化轨迹所行驶的距离。众所周知，Bézier 曲线的一个性质是相邻控制点之间的距离之和是曲线总长度的上界 [30]。为简单起见，我们假设所有凸集在 $\mathbb{R}^n$ 上接触，并且穿越一条边没有额外的成本，因此 $f_e(x_e) = 0$ 。因此，每个节点的成本函数可以用设计变量表示为：

$$f_v(x_v) = \sum_{i=0}^{n-1} \|x_{v,i+1} - x_{v,i}\|_{xy,2}. \quad (33)$$

将 $z_{v,i} = y_v x_{v,i}$ 代入 f_v 得：

$$\begin{aligned} f_v\left(\frac{z_v}{y_v}\right) &= \sum_{i=0}^{n-1} \left\| \frac{z_{v,i+1}}{y_v} - \frac{z_{v,i}}{y_v} \right\|_{xy,2} \\ &= \frac{1}{|y_v|} \sum_{i=0}^{n-1} \|z_{v,i+1} - z_{v,i}\|_{xy,2}. \end{aligned} \quad (34)$$

我们可以从范数中去掉 y_v ，并且由于它非负，可以去掉它周围的绝对值符号。总成本函数可以写为：

$$\begin{aligned} J(y, z) &= \sum_{v \in \mathcal{V}} y_v \frac{1}{y_v} \sum_{i=0}^{n-1} \|z_{v,i+1} - z_{v,i}\|_{xy,2} \\ J(z) &= \sum_{v \in \mathcal{V}} \sum_{i=0}^{n-1} \|z_{v,i+1} - z_{v,i}\|_{xy,2}. \end{aligned} \quad (35)$$

这是我们用来生成无碰撞轨迹的 GCS 公式中的成本函数。

H. 优化公式

我们将成本函数以及边和顶点的约束整合到由方程 8 描述的整体优化框架中。所得的公式是一个松弛的混合整数凸规划 (MICP)，表示为：

$$J(z) = \sum_{v \in \mathcal{V}} \sum_{i=0}^{n-1} \|z_{v,i+1} - z_{v,i}\|_{xy,2} \quad (36a)$$

$$\min_{z, y} J(z) \quad (36b)$$

s.t.

$$y_v \in \{0, 1\} \quad (36c)$$

$$y_e \in \{0, 1\} \quad (36d)$$

$$\begin{aligned} \sum y_{(a,v)} + \delta_s \\ = \sum y_{(v,b)} + \delta_f = y_v \quad \forall v \in \mathcal{V} \end{aligned} \quad (36e)$$

$$A_v z_{v,i} \leq b_v y_v. \quad (36f)$$

$$z_{v,n} = \sum_b \vec{z}_{(v,b),0} \quad \forall v \in \mathcal{V} \quad (36g)$$

$$z_{v,n} - z_{v,n-1} = \sum_b z_{(v,b),\text{diff}} \quad \forall v \in \mathcal{V} \quad (36h)$$

$$\begin{aligned} \|(z_{v,i+1} - z_{v,i})\|_{xy,2} \leq \\ v_{\max}(z_{v,i+1} - z_{v,i})_t \end{aligned} \quad (36i)$$

$$(\vec{z}_{v,i+1} - \vec{z}_{v,i})_t \geq y_v \epsilon. \quad (36j)$$

最终的公式包括每个顶点和边的松弛二进制变量 y 、每个顶点内的 n 控制点向量 $z_{v,i}$ ，以及表示连续性 $z_{(v,b),0}$ 和可微性 $z_{(v,b),diff}$ 的额外向量。每个这些函数和约束都是凸的，并且与一般凸求解器兼容。

IV. 结果

在本节中，我们将我们的时空 GCS 公式应用于一系列场景并评估其性能。首先，我们将在仅有静态障碍物的场景中，将我们的公式与基础版 GCS 公式进行对比验证。接着，我们证明其在包含许多动态障碍物的复杂环境中的有效性。

这些模拟和算法是在 Python 中实现的，并在具有 32GB RAM 的 AMD Ryzen 7 9700X 处理器上执行。凸优化问题使用 CVXPY 进行公式化，并使用开源求解器 CLARABELL 进行求解。

A. 标准 GCS 比较

因为标准的 GCS 公式与动态环境不兼容，我们将在静态场景中将空间-时间方法与其进行比较。我们展示了在这种情况下它会产生相同的最优轨迹。对于标准的 GCS 公式，环境被分解为二维的凸集图，而对于空间-时间公式，则是三维的凸集图。

场景参数如下。一个状态为 (x,y,t) 的代理从 $(0.5m, 0m, 0s)$ 开始，必须移动到 $(0.5m, 1m, 1s)$ 。代理的最大速度为 $2m/s$ 。存在一个静态障碍物，其顶点位于 $(0.3m, 0.2m)$ 、 $(0.6m, 0.2m)$ 、 $(0.6m, 0.4m)$ 和 $(0.3m, 0.4m)$ 。由于该障碍物略偏中心，存在一个避免障碍物的几何最小距离，该距离位于起点和终点之间，为 $1.0318m$ 。图 12 以二维和三维展示了场景。静态障碍物用黑色表示，其顶点在 $t=0s$ 和 $t=1s$ 之间垂直对齐。起点和终点分别用绿色和红色表示。

图 13 展示了由标准 GCS 公式生成的轨迹。集合及其相关的 Bèzier 曲线控制点相应地进行了颜色编码。凸集图由四个集合和八条有向边组成。总计算时间为 $0.2s$ 。行进的最优距离为 $1.0318m$ ，与几何最小值一致。

图 14 显示了通过时空方法生成的控制点的从上往下视角和等轴视角。当从上往下看时，几何路径与标准 GCS 解决方案相同。等轴视角展示了每个样条控制点随时间的垂直分布。凸集

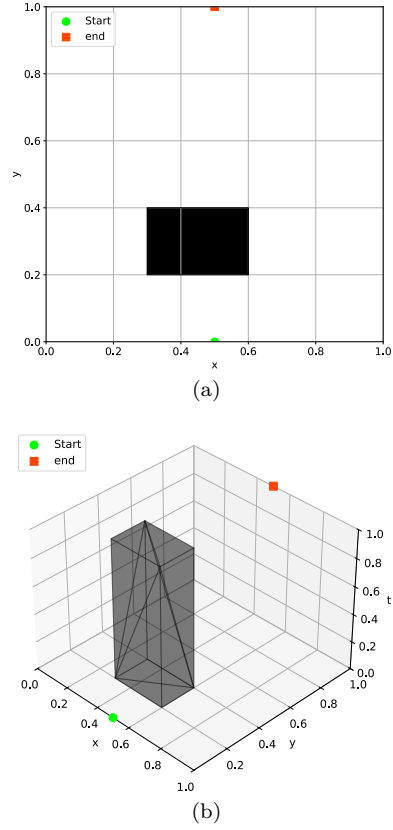


Fig. 12: 环境在二维中显示 (a) 和将在时空 GCS 公式中使用的环境的等距视图 (b)。

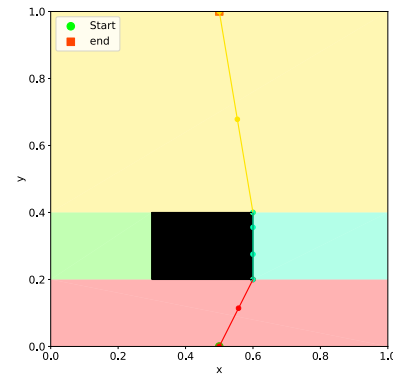


Fig. 13: 由标准 GCS 公式生成的路径。

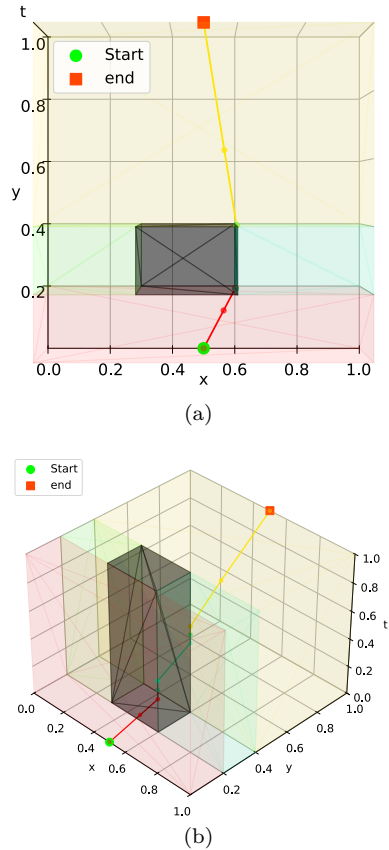


Fig. 14: 从顶部 (a) 和轴测视图 (b) 展示的时空 GCS 的结果集合和轨迹。

图由四个集合和八个有向边组成。总计算时间为 $0.29s$ 。在二维空间中行进的最佳距离也是 $1.0318m$ ，与标准 GCS 解决方案和真实几何最小值相匹配。这些结果显示，我们修改后的 GCS 公式在静态环境中产生的轨迹与原始 GCS 相同，从而验证了我们实现的正确性。

B. 动态环境

接下来，我们展示一个有单个移动障碍物的简单场景。在这种情况下，多条最优轨迹具有相同的全局最小距离。我们展示了我们的方法如何通过凸集图找到一条最优轨迹，同时避开移动障碍物。需要注意的是，标准的 GCS 方法在不引入非线性求解器来确定路径的时间和避免碰撞的情况下，无法用于解决这个场景。

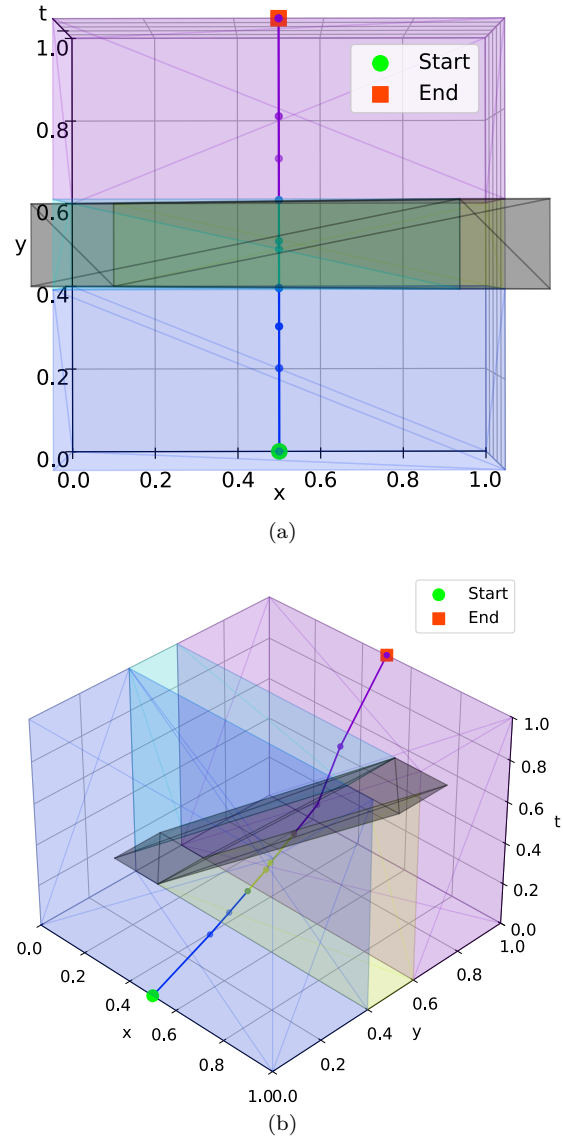


Fig. 15: 从顶部视图 (a) 和等角视图 (b) 展示的时空 GCS 的结果集合和无碰撞轨迹。

场景参数如下。一个状态为 (x, y, t) 的智能体从 $(0.5m, 0m, 0s)$ 出发, 必须移动到 $(0.5m, 1m, 1s)$ 。智能体的最大速度为 $2m/s$ 。有一个长度为 $0.2m$ 的方形动态障碍物。这个方形起初位于 $(0m, 0.5m, 0s)$, 并移动到 $(1m, 0.5m, 1s)$ 。从智能体的起始位置到终点位置, 避开障碍物的几何最小距离是 $1.0m$ 。

图 15 展示了我们的方法生成的控制点, 从俯视图和等距视图来看。从上方观察, 几何路径从开始到结束呈直线移动。等距视图显示代理在开始时快速移动, 通过轨迹的斜率证明了这点, 以越过障碍, 然后减速。凸集图由四个集合和八个有向边组成。总计算时间为 $0.52s$ 。在二维中行驶的最佳距离也是 $1.0m$, 即真正的几何最短距离。这个例子展示了我们的方法在动态障碍物存在的情况下生成无碰撞、最佳轨迹的能力, 而不依赖初始猜测。

C. 杂乱环境

前面的例子显示了我们的方法可以找到避免单个障碍物的最优轨迹。但是, 在实际场景中, 通常有更多的障碍物需要规避。下一个场景展示了我们的方法在一个拥挤的环境中如何表现, 寻找通过凸集图的最优路径, 并提出了一种改善结果的凸集生成启发式方法。我们还讨论了一些由于生成凸集的方法而出现的缺点。

场景参数如下。一个智能体从 $(0.5m, 0m, 0s)$ 出发, 必须移动到 $(0.5m, 1m, 1s)$ 。智能体的最大速度为 $3m/s$, 允许它在指定的 $1s$ 时间内行驶最多三倍的起止距离。在 $(0m, 0.2m)$ 和 $(1m, 0.8m)$ 之间随机初始化二十个障碍物, 并且具有随机的恒定速度。图 16 显示了这个动态环境的起始和结束 2D 截面图。图 17 显示了从侧面和等距视角看到的 3D 外观。找到一个躲避这个环境中所有障碍物的轨迹的初始猜测是极具挑战性的, 甚至可能是不可能的。

自由空间被分解为三维空间中的凸集图。IRIS 分割算法通过从种子点开始扩展生成每个凸集, 以创建包含种子点的最大可能的无碰撞凸集。该算法通过随机采样环境来选取种子点, 丢弃位于障碍物或现有集合中的样本, 然后扩展凸集图。图 18 a 显示了在采样了 80 个随机点并适当地丢弃样本后的凸集图的三维视图。在三维空间中, 许多集合在视觉上被障碍物或其他集合遮挡。图 18 b 显示了凸集图在 $(t = 0.80s)$ 处的截面图, 集合使用不同颜色标记, 障碍物显示为

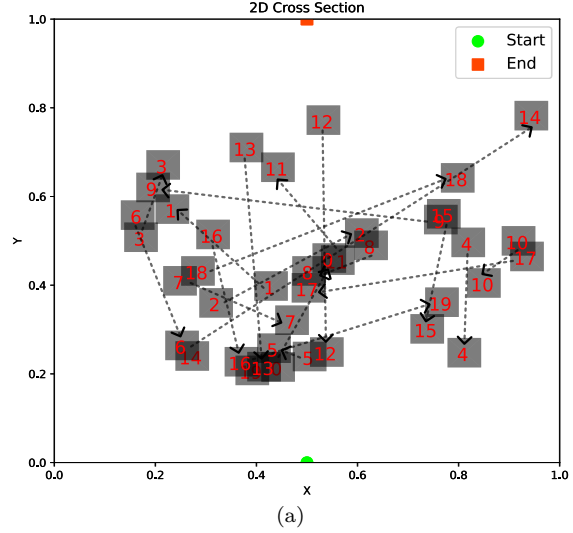


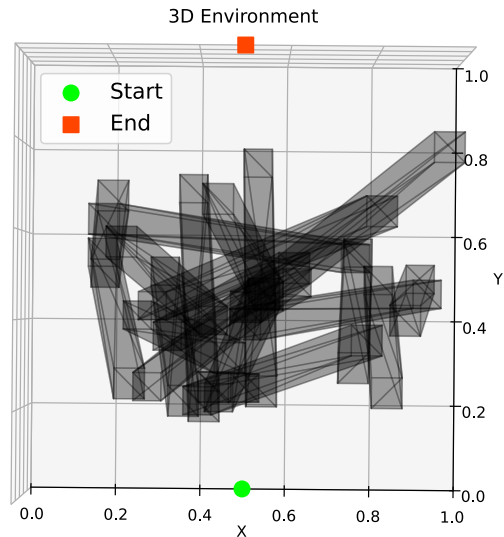
Fig. 16: 障碍物以恒定速度从随机起点移动到随机终点。箭头指示运动方向。轨迹的起点和终点分别用绿色和红色表示。

黑色。利用这一随机种子, 凸集图由十四集合和九条有向边组成。

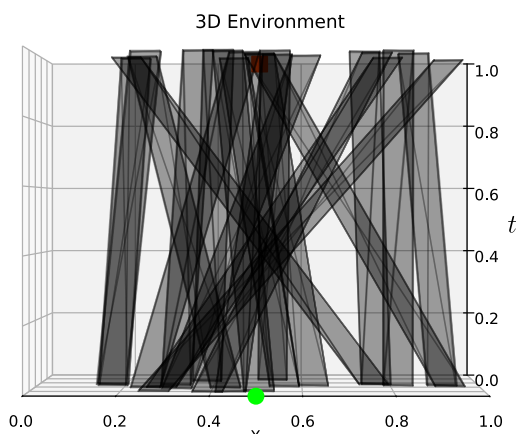
图 19 展示了由我们的方法生成的轨迹。环境中没有与任何 20 个障碍物发生碰撞。然而, 尽管该轨迹在凸集图方面是全局最优的, 集合本身并没有完全覆盖自由空间。如图 18 b 所示, 在有许多障碍的区域, 集合的密度较低。这导致轨迹在几何最小距离方面不是全局最优的。图 19 显示了从顶部和等距视图的 3D 轨迹。总计算时间为 $4.12s$ 。行驶的最优距离是 $1.28m$ 。

我们在上面的相同场景下运行实验, 同时改变 IRIS 样本的数量, 并使用不同的种子重复实验 100 次。对于每次运行, 我们记录了图中创建的集合和边的数量、最终的轨迹成本以及生成图形和优化轨迹所需的总计算时间。如预期所示, 当样本数量增加时, 集合和边的数量都增加, 从而导致较低的平均轨迹成本。

表 I 总结了使用不同数量的样本初始化 IRIS 算法时平均结果的变化情况。计算时间随着图的大小增加而增加, 而轨迹的距离则减少。随着图的扩展, 计算轨迹所需的时间大幅增长。虽然一些 GCS 论文提出了一些策略, 比如在第一次分支定界迭代后修剪图以减少计算时间, 但我们在这项工作中并没有实现这些方法。图 20 显示了



(a)

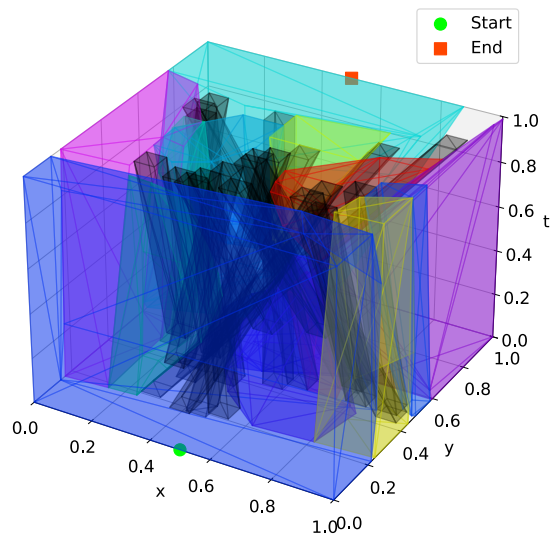


(b)

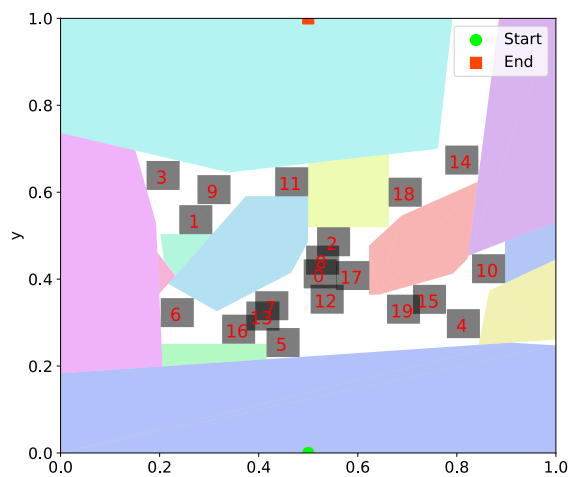
Fig. 17: 从顶部 (a) 和侧面视图 (b) 显示的杂乱环境。

在对 IRIS 算法取样 1000 个点后, 在 $t = 0.8s$ 处的一个示例横截面, 从而通过环境中心实现了更密集的集合覆盖, 并使得解更接近几何最小值。

尽管该方法成功地避免了所有障碍, 这些结果强调了凸集密度对轨迹最优性的影响。在有许多障碍的区域中, 稀疏的覆盖可能导致次优路径, 即使在构建的图中解决方案在全球范围内是最优的。增加样本数量可以提高集密度和解决方案



(a)



(b)

Fig. 18: 在采样 80 个点后凸集图的三维视图 (a) 以及在 $t = 0.80s$ 处图的截面 (b)。

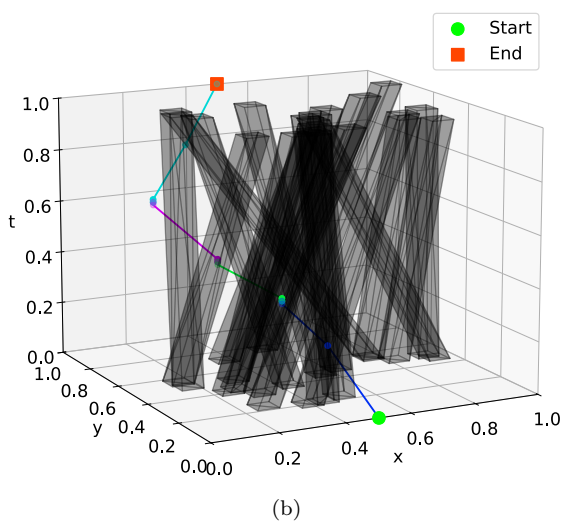
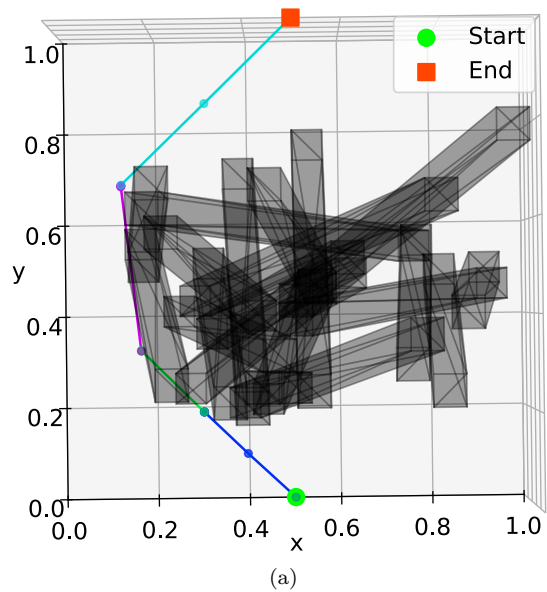


Fig. 19: 由我们的方法生成的轨迹的俯视图 (a) 和轨迹的透视图 (b)。

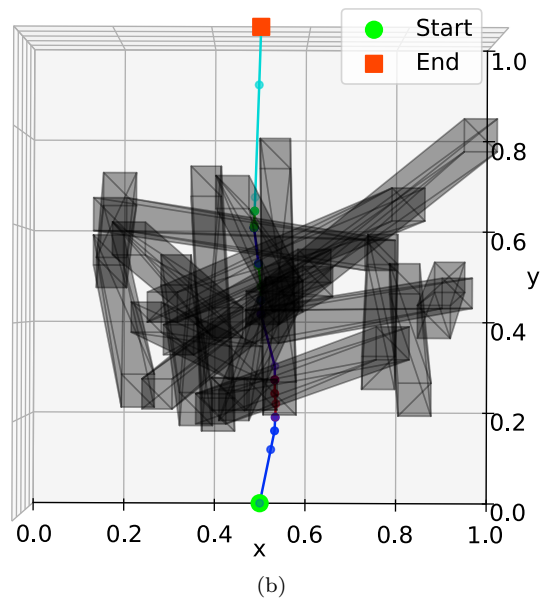
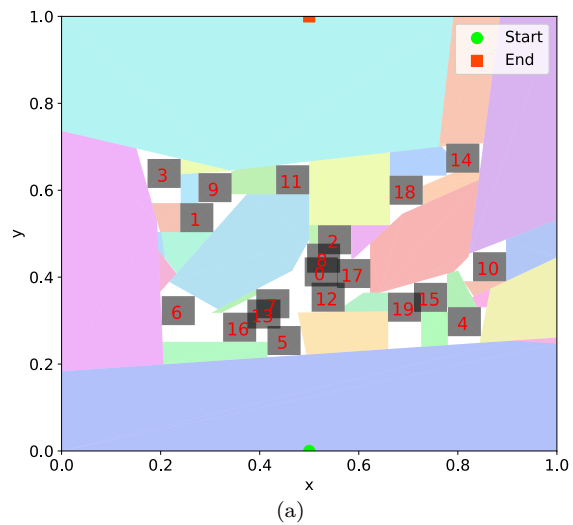


Fig. 20: 横截面 $t = 0.8s$ ，使用 1000 个样本作为 IRIS 算法的初始样本 (a)。在环境的中心，凸集被密集地排列，导致较短的轨迹 (b)。

TABLE I: IRIS 采样对 GCS 轨迹优化的平均效果

Samples	Sets	Edges	Cost (m)	Time (s)
80	15.22	22.71	1.24	4.75
100	17.05	26.45	1.23	5.28
250	27.58	52.28	1.12	11.50
500	40.16	95.27	1.05	26.80
1000	55.62	148.44	1.03	82.30

质量, 但代价是显著增加计算时间。

V. 结论

我们提出了一种用于在拥挤的动态环境中生成轨迹的 GCS 公式。此处介绍的约束开发策略扩展了 GCS 在更广泛的轨迹优化问题中的适用性。尽管凸集生成方法限制了解的质量, 但 GCS 框架提供的安全性和最优性保证在其他轨迹优化方法中很少见。此外, 时空公式使 GCS 能够有效处理动态环境。值得注意的是, 这种方法允许在不需要良好的初始空间或时间猜测的情况下生成最优轨迹。总体来说, 这项工作展示了一种实现 GCS 约束的策略, 以及如何扩展 GCS 框架以处理动态环境。

VI.

致谢

该项目由 NSF SBIR 第二阶段奖项编号 2404858 和 4D Avionic Systems, LLC 资助。我们感谢 Garth Thompson 博士的反馈和富有洞察力的建议。

REFERENCES

- [1] L. Yang, P. Li, S. Qian, H. Quan, J. Miao, M. Liu, Y. Hu, and E. Memetimin, "Path Planning Technique for Mobile Robots: A Review," *Machines*, vol. 11, no. 10, p. 980, Oct. 2023. [Online]. Available: <https://www.mdpi.com/2075-1702/11/10/980>
- [2] K. Cai, C. Wang, J. Cheng, C. W. De Silva, and M. Q. H. Meng, "Mobile Robot Path Planning in Dynamic Environments: A Survey," 2020, publisher: arXiv Version Number: 2. [Online]. Available: <https://arxiv.org/abs/2006.14195>
- [3] F. A. Raheem, S. M. Raafat, and S. M. Mahdi, "Robot Path-Planning Research Applications in Static and Dynamic Environments," in *Earth Systems Protection and Sustainability*, J. N. Furze, S. Eslamian, S. M. Raafat, and K. Swing, Eds. Cham: Springer International Publishing, 2022, pp. 291–325. [Online]. Available: https://link.springer.com/10.1007/978-3-030-85829-2_12
- [4] T. Kröger, *On-Line Trajectory Generation in Robotic Systems*, ser. Springer Tracts in Advanced Robotics, B. Siciliano, O. Khatib, and F. Groen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, vol. 58. [Online]. Available: <http://link.springer.com/10.1007/978-3-642-05175-3>
- [5] R. Smierzchalski and Z. Michalewicz, "Path Planning in Dynamic Environments," in *Innovations in Robot Mobility and Control*, J. Kacprzyk, S. Patnaik, L. C. Jain, S. G. Tzafestas, G. Resconi, and A. Konar, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, Aug. 2005, vol. 8, pp. 135–153, series Title: Studies in Computational Intelligence. [Online]. Available: http://link.springer.com/10.1007/10992388_4
- [6] M. Luo, X. Hou, and J. Yang, "Surface Optimal Path Planning Using an Extended Dijkstra Algorithm," *IEEE Access*, vol. 8, pp. 147 827–147 838, 2020. [Online]. Available: <https://ieeexplore.ieee.org/document/9165710/>
- [7] C. Zhou, B. Huang, and P. Fränti, "A review of motion planning algorithms for intelligent robots," *Journal of Intelligent Manufacturing*, vol. 33, no. 2, pp. 387–424, Feb. 2022. [Online]. Available: <https://link.springer.com/10.1007/s10845-021-01867-z>
- [8] W. Fink, V. R. Baker, A. J.-W. Brooks, M. Flammia, J. M. Dohm, and M. A. Tarbell, "Globally optimal rover traverse planning in 3D using Dijkstra's algorithm for multi-objective deployment scenarios," *Planetary and Space Science*, vol. 179, p. 104707, Dec. 2019. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0032063318302526>
- [9] A. Martelli, "On the complexity of admissible search algorithms," *Artificial Intelligence*, vol. 8, no. 1, pp. 1–13, Feb. 1977. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/0004370277900029>
- [10] R. Valenzano, N. Sturtevant, and J. Schaeffer, "Worst-Case Solution Quality Analysis When Not Re-Expanding Nodes in Best-First Search," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, Jun. 2014. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/8850>
- [11] X. Xiong, H. Min, Y. Yu, P. Wang, and State Key Laboratory of Automotive Simulation and Control, Jilin University, No. 5988, Renmin Street, Changchun, Jilin 130022, China, "Application improvement of A* algorithm in intelligent vehicle trajectory planning," *Mathematical Biosciences and Engineering*, vol. 18, no. 1, pp. 1–21, 2021. [Online]. Available: <http://aimspress.com/article/doi/10.3934/mbe.2021001>
- [12] M. N. Finean, W. Merkt, and I. Havoutis, "Predicted Composite Signed-Distance Fields for Real-Time Motion Planning in Dynamic Environments," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 31, pp. 616–624, May 2021. [Online].

- Available: <https://ojs.aaai.org/index.php/ICAPS/article/view/16010>
- [13] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control Barrier Functions: Theory and Applications," in *2019 18th European Control Conference (ECC)*. Naples, Italy: IEEE, Jun. 2019, pp. 3420–3431. [Online]. Available: <https://ieeexplore.ieee.org/document/8796030/>
 - [14] Y. Arai, T. Sago, Y. Ueyama, and M. Harada, "Avoidance Trajectory Generation for Quad-Rotors by Polygonal Bounding of Congested Obstacles," in *AIAA SCITECH 2024 Forum*. Orlando, FL: American Institute of Aeronautics and Astronautics, Jan. 2024. [Online]. Available: <https://arc.aiaa.org/doi/10.2514/6.2024-1596>
 - [15] L. Yuan, J. Zhao, W. Li, and J. Hou, "Improved Informed-RRT* Based Path Planning and Trajectory Optimization for Mobile Robots," *International Journal of Precision Engineering and Manufacturing*, vol. 24, no. 3, pp. 435–446, Mar. 2023. [Online]. Available: <https://link.springer.com/10.1007/s12541-022-00756-6>
 - [16] X. Zhu, Y. Gao, Y. Li, and B. Li, "Fast Dynamic P-RRT*-Based UAV Path Planning and Trajectory Tracking Control Under Dense Obstacles," *Actuators*, vol. 14, no. 5, p. 211, Apr. 2025. [Online]. Available: <https://www.mdpi.com/2076-0825/14/5/211>
 - [17] M. Osburn, C. K. Peterson, and J. L. Salmon, "Optimization of 4D Splines for Unmanned Aerial System Trajectories Under Kinematic, Obstacle, and Time Constraints," in *AIAA SCITECH 2025 Forum*. Orlando, FL: American Institute of Aeronautics and Astronautics, Jan. 2025. [Online]. Available: <https://arc.aiaa.org/doi/10.2514/6.2025-2230>
 - [18] Z. Zang, J. Song, Y. Lu, X. Zhang, Y. Tan, Z. Ju, H. Dong, Y. Li, and J. Gong, "A Unified Framework Integrating Trajectory Planning and Motion Optimization Based on Spatio-Temporal Safety Corridor for Multiple AGVs," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 1217–1228, Jan. 2024. [Online]. Available: <https://ieeexplore.ieee.org/document/10149534/>
 - [19] T. Marcucci, "Graphs of Convex Sets with Applications to Optimal Control and Motion Planning."
 - [20] T. Marcucci, J. Umenberger, P. Parrilo, and R. Tedrake, "Shortest Paths in Graphs of Convex Sets," *SIAM Journal on Optimization*, vol. 34, no. 1, pp. 507–532, 2024, eprint: <https://doi.org/10.1137/22M1523790>. [Online]. Available: <https://doi.org/10.1137/22M1523790>
 - [21] T. Marcucci, M. Petersen, D. v. Wrangel, and R. Tedrake, "Motion Planning around Obstacles with Convex Optimization," May 2022, arXiv:2205.04422 [cs]. [Online]. Available: <http://arxiv.org/abs/2205.04422>
 - [22] S. Morozov, T. Marcucci, A. Amice, B. P. Graesdal, R. Bosworth, P. A. Parrilo, and R. Tedrake, "Multi-Query Shortest-Path Problem in Graphs of Convex Sets," Sep. 2024, arXiv:2409.19543 [cs]. [Online]. Available: <http://arxiv.org/abs/2409.19543>
 - [23] T. Cohn, M. Petersen, M. Simchowicz, and R. Tedrake, "Non-Euclidean motion planning with graphs of geodesically convex sets," *The International Journal of Robotics Research*, p. 02783649241302419, Dec. 2024. [Online]. Available: <https://journals.sagepub.com/doi/10.1177/02783649241302419>
 - [24] S. Garg, T. Cohn, and R. Tedrake, "Planning Shorter Paths in Graphs of Convex Sets by Undistorting Parametrized Configuration Spaces," Nov. 2024, arXiv:2411.18913 [cs]. [Online]. Available: <http://arxiv.org/abs/2411.18913>
 - [25] D. Von Wrangel and R. Tedrake, "Using Graphs of Convex Sets to Guide Nonconvex Trajectory Optimization," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Abu Dhabi, United Arab Emirates: IEEE, Oct. 2024, pp. 9863–9870. [Online]. Available: <https://ieeexplore.ieee.org/document/10802426/>
 - [26] P. Werner, R. Cheng, T. Stewart, R. Tedrake, and D. Rus, "Superfast Configuration-Space Convex Set Computation on GPUs for Online Motion Planning," Apr. 2025, arXiv:2504.10783 [cs]. [Online]. Available: <http://arxiv.org/abs/2504.10783>
 - [27] S. P. Boyd and L. Vandenberghe, *Convex optimization*, version 29 ed. Cambridge New York Melbourne New Delhi Singapore: Cambridge University Press, 2023.
 - [28] R. Deits and R. Tedrake, "Computing Large Convex Regions of Obstacle-Free Space Through Semidefinite Programming," in *Algorithmic Foundations of Robotics XI*, H. L. Akin, N. M. Amato, V. Isler, and A. F. Van Der Stappen, Eds. Springer International Publishing, 2015, vol. 107, pp. 109–124, series Title: Springer Tracts in Advanced Robotics. [Online]. Available: https://link.springer.com/10.1007/978-3-319-16595-0_7
 - [29] M. Duncan, "Bézier curves I," in *Applied Geometry for Computer Graphics and CAD*. Springer, 2005, pp. 135–160.
 - [30] J. Gravesen, "Adaptive subdivision and the length and energy of Bkzier curves."