

# Fast Iterative Region Inflation for Computing Large 2-D/3-D Convex Regions of Obstacle-Free Space

Qianhao Wang<sup>†</sup>, Zhepei Wang<sup>†</sup>, Chao Xu, and Fei Gao\*

## I. INTRODUCTION

Benefiting from the compact representation and convexity, convex polytopes are inherently suitable for characterizing passable regions applied in efficiency-focused trajectory planning. Evidently, the fundamental prerequisite for harnessing this advantage is to compute large convex polytopes of the free regions from complex maps that contain massive data.

Despite extensive research efforts dedicating to this topic, striking a balance between the quality of the convex polytope and the efficiency of the generation process remains a formidable challenge. Typically, either existing approaches excel at generating high-quality convex free regions but require substantial computational budget [1], [2], or they demand a cheap computational overhead yet yield rough even conservative results [3], [4]. High-quality convex free polytope refers to be as large as possible in this paper.

In addition to quality and efficiency, we introduce the concept of *managibility* as another crucial consideration in generating convex regions. Managibility refers to the ability of the generated convex polytope to accurately encompass a specified set of points. For example, there are situations where it is desirable for the path used to guide the polytope generation to lie entirely within the generated region [4], or where the generated region should contain the robot itself [5], [6], particularly in trajectory planning scenarios that take into account the shape of the robot, i.e., whole-body planning (Han et al., 2021; Wang et al., 2022). However, many existing algorithms prioritize the quality of the region without adequately considering or ensuring managibility [1], [3], [7], [8].

Based on the above analysis, we conclude that an ideal algorithm for computing free convex polytope should possess both managibility and computational efficiency, while generating the free region with high quality. To satisfy these requirements, we propose an efficient algorithm, called **Fast Iterative Region Inflation (FIRI)**. As depicted in the Fig. 1 and Algorithm 1, FIRI takes obstacles, a seed that is required to be included, and an initial ellipsoid as inputs. It iteratively proceeds with two modules: (1) Restrictive Inflation: inflating the ellipsoid and utilizing its contact planes tangent to the obstacles to generate a set of halfspaces, which separate a convex polytope from the obstacles; (2) calculating the Maximum Volume Inscribed Ellipsoid (MVIE) of the convex

polytope as the ellipsoid to be inflated in the next iteration. MVIE serves as a lower bound of the volume of the convex polytope, which monotonically expands during the iterative update, leading to a growing obstacle-free region that ensures the high-quality result of FIRI. Although a similar idea of monotonic updating the lower bound was pioneered in Deits' work IRIS [1], the lack of managibility and its daunting computational overhead in complex environments [1], [3], limit its real-time applications. In contrast, FIRI ensures managibility by restricting the halfspaces that composes the polytope to necessarily exclude obstacles yet contain the seed. It is even more significant that, in terms of efficiency, we design methods specifically for the two optimization-based modules involved in FIRI, leveraging their geometric properties, which results in a remarkable computational efficiency improvement, achieving a speedup of orders of magnitude compared to IRIS.

For Restrictive Inflation, we convert the halfspace computation into a strictly convex quadratic programming (QP). In addition, considering the low-dimensional yet multi-constrained nature of this QP, we generalize Seidel's method [9], which is originally designed for solving linear programming with linear complexity, to handle with strictly convex QP problems. Compared with several well-known solvers [10]–[12] for QP, this method achieves the capability of obtaining analytical solutions within a significantly shorter time.

For solving MVIE, known as one of the most challenging extremal ellipsoid problems [13], is highly demanded in a number of applications [14], [15]. Most existing methods adopt semidefinite programming (SDP) formulations [1], [16] and employ various interior point methods for solving [13], [17]–[19]. They face challenges in achieving satisfactory computational efficiency for the low-dimensional but massive constraint scenario addressed in this paper, due to the need to handle with large-scale systems of equations in each iteration. To ease the computation overhead without sacrificing the solution quality, we present an equivalent formulation for MVIE in the form of second-order conic programming (SOCP). This brings about a noticeable boost in computational efficiency.

Moreover, in 2-D scenarios, to attain ultimate computational efficiency, we propose an analytical method for MVIE. The proposed method exhibits a time complexity linear in the number of the hyperplanes of the input convex polytope, which is given for the first time to the best of our knowledge. As the dual problem of MVIE, the linear-time exact algorithm for 2-D Minimum Volume Enclosing Ellipse (MVEE) [20], [21] has long been established. However, the corresponding approach for MVIE remains absent for several decades, due to its failure to satisfy basic regularity [20] and the lack of

<sup>†</sup> Equal contribution.

\* Corresponding author.

All authors are with the State Key Laboratory of Industrial Control Technology, Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, China and Huzhou Institute, Zhejiang University, Huzhou 313000, China. {qhwangaa, wangzhepei, fgaoaa}@zju.edu.cn

analytical solutions for basis computation. In this paper, we address above challenges by designing a novel randomized framework with a meticulous problem redefinition and a bottom-up strategy motivated by the subalgorithm for distance calculation in GJK [22]. As a result, the proposed specialized 2-D method achieves a substantial efficiency gain compared to other state-of-the-art methods [1], [19], surpassing them by several orders of magnitude.

Based on the above methods for Restrictive Inflation and MVIE, the proposed algorithm FIRI realizes, for the first time, managibility, high quality, and high efficiency simultaneously in convex polytope generation. We conduct comprehensive comparisons between FIRI and various convex polytope generation algorithms [1], [3], [4]. The findings provide compelling evidence that our algorithm outperforms other approaches across the three aforementioned requirements. Moreover, we extensively perform experiments involved both simulations and real-world applications to showcase the applicability of FIRI. High-performance implementation of FIRI will be open-sourced for the reference of the community.

In summary, the contributions are:

- 1) Restrictive Inflation is designed to ensure the managibility of the generated convex polytope. Based on its characteristic of few variables but rich constraints, an efficient and numerically stable solver is designed.
- 2) A novel method that formulates the MVIE problem into SOCP formulation is proposed, which avoids directly confronting the positive definite constraints and improves the computational efficiency.
- 3) Especially for 2-D MVIE, a linear-time exact algorithm is introduced for the first time, filling a gap that existed for several decades [20], [21] and further enabling ultra-fast computational performance.
- 4) Building upon the above methods, a reliable convex polytope generation algorithm FIRI is proposed. Extensive experiments verify its superior comprehensive performance in terms of quality, efficiency, and managibility.

## II. RELATED WORK

### A. Generating Convex Free Polyhedra

Deits et al. [1] propose Iterative Region Inflation by Semidefinite Programming (IRIS) with the objective of generating the largest possible free convex polyhedra. IRIS involves inflating an ellipsoid, using it as a seed, to obtain a convex polyhedra formed by the intersection of contact planes. Then in the next iteration, the maximum volume inscribed ellipsoid (MVIE) of the obtained convex polyhedra is selected as the new seed for inflation. However, this method requires solving semidefinite programming problems in each iteration to obtain the MVIE, which significantly hampers the computational efficiency. Furthermore, it does not guarantee that the generated convex hull always include the initial seed point, thereby lacking managibility. Although IRIS is able to terminate iterations when the generated polytope does not contain the seed point, its essence does not lie in maximize the volume while simultaneously constraining the seed to be inside. In contrast, FIRI approaches this problem from the perspective of constrained

optimization, incorporating Restrictive Inflation that considers more general scenarios. To address the efficiency and local managibility deficiencies of IRIS, Liu et al. [4] propose the Regional Inflation by Line Search algorithm (RILS) which takes line segment as seed input. RILS first generate a maximal ellipsoid that contains the segment yet excludes all obstacles and then inflate the ellipsoid to form a convex polyhedra with contact planes from the obstacles. The RILS algorithm demonstrates high computational speed. Nonetheless, since the ellipsoid always has the input line segment as major axis, it tends to produce conservative convex hull. Specifically for voxel maps, Gao et al. [2] develop Parallel Convex Cluster Inflation algorithm. The algorithm starts with a seed voxel that is unoccupied and then incrementally grows in layers along the coordinate axes while checking the visibility of the new voxels to the existing set of voxels, which ensures the convexity of the voxel set throughout the growth process. Although it utilizes parallel computing resources to accelerate the computation, this algorithm can only achieve near real-time performance when the map resolution is not fine.

On the other hand, Savin et al. [8] utilize the concept of space inversion. They apply a spherical polar mapping transformation to all the obstacle points with respect to the input seed point, which flips the feasible region around the seed point to the outside. Then, they computed the convex hull of the inverted obstacle point set and then transform it back to the original space to obtain the final convex hull. The limitation of this method lies in the strong nonlinearity of the spherical polar mapping, which means that any volume loss in the inverted space can result in significant gaps between the final convex polyhedra and the obstacles, leading to an overly conservative result. To address this limitation, Zhong et al. [3] propose using sphere flipping mapping as an alternative approach. Due to the properties of the hidden point remove operator inherent [23] in this mapping, this approach obtains the visible star-convex region around the seed point. They further devise a heuristic method to partition the star-convex region into the final convex polyhedra. However, the approach involves Quickhull [24] whose performance degrades when confronted with the points that are approximately distributed near the sphere after the mapping. In addition, heuristic partitioning can also lead to conservative result.

Conclusively, existing methods consistently suffer from at least one of the following issues: inefficiency, conservatism, and lack of managibility, thereby limiting their practicality. In this paper, we propose FIRI for computation of convex feasible region with high efficiency, strong managibility, and high quality.

### B. Maximum Volume Inscribed Ellipsoid (MVIE)

MVIE is also known as inner Löwner-John ellipsoid [25]. Nesterov et al. [18] utilize an interior-point algorithm with a specialized rescaling method on each iteration to achieve a polynomial time solution of MVIE, which is much faster than using ellipsoid algorithm [26]. Khachiyan et al. [27] transform the problem into a sequence of subproblems with only linear constraints, constructed by using the barrier method. This

approach requires fewer computations compared to Nesterov's method [18]. Then Anstreicher et al. [28] make improvements to both methods [18], [27] and demonstrate that computing an approximate analytic center of the polytope beforehand can reduce the complexity effectively. Zhang et al. [17] also provide a modification of Khachiyan's approach [27] by replacing the inefficient primal barrier function method with a primal-dual interior-point method to solve the subproblems. Additionally, instead of dealing with a number of subproblems, they propose a novel primal-dual interior-point method free of matrix variables to solve MVIE directly. Through nonlinear transformations, they eliminate the positive-definite constraint on the coefficient matrix of the ellipsoid during the iteration process and provide a proof demonstrating that these nonlinear transformations preserve the uniqueness of the solution. Building upon similar idea of eliminating matrix variables, Nemirovskii [29] reformulate MVIE as a saddle-point problem based on the Lagrangian duality. Then they adopt a path-following method to obtain approximate saddle points according to the self-concordance theory proposed in Nesterov's work [18]. However, these aforementioned interior-point methods face challenges when dealing with the scenario where the number of constraints is significantly larger than the dimension of the space. In these scenarios which are the focus of this paper, the large-scale system of linear equations that these methods need to solve at each iteration prevents them from computing MVIE within an acceptable time. To address the issue of inefficiency, Lin et al. [30] employ the fast proximal gradient method [31] to introduced a first-order optimization-based approach for MVIE. Although this approach significantly improves computational efficiency, it does not provide an exact solution since it approximates the nondifferentiable indicator function by employing a differentiable one-sided Huber function to deal with the positive-definite constraint in MVIE.

MVIE is the most challenging problem among the extremal ellipsoid problems [13]. One of these problems, known as the Minimum Volume Enclosing Ellipsoid (MVEE), can be transformed into MVIE with a linear reduction, which is irreversible. In addition to interior-point methods, analytical solutions of MVEE can be obtained using randomized methods in low-dimensional cases [20], [32]. Inspired by MVEE, in this paper we propose a randomized algorithm specifically for the 2-D case to efficiently obtain the analytical solutions for the more difficult MVIE problem, which is given for the first time to the best of our knowledge.

### III. FAST ITERATIVE REGION INFLATION

#### A. Problem Formulation

Consider that a convex robot is navigating in an obstacle-rich  $n$ -dimensional environment, where  $n = 2, 3$ . The geometrical shape of the robot is given by the  $\mathcal{V}$ -representation [33] of a convex polytope, i.e., convex combination of a finite number of points

$$\mathcal{Q} = \text{conv}\{v_1, \dots, v_s\}, \quad (1)$$

in which  $v_1, \dots, v_s \in \mathbb{R}^n$  are allowed to be redundant, which means that they are not required to only contain the extreme

points of  $\mathcal{Q}$ . The obstacle region  $\mathcal{O}$  is, albeit nonconvex, assumed to be the union of convex obstacles  $\mathcal{O} = \cup_{i=1}^R \mathcal{O}_i$ , of which the  $i$ -th one is determined by  $s_i$  points

$$\mathcal{O}_i = \text{conv}\{u_{i,1}, \dots, u_{i,s_i}\}. \quad (2)$$

We require no collision between the robot  $\mathcal{Q}$  and the obstacle region  $\mathcal{O}$ , thus implying  $\mathcal{Q} \cap \mathcal{O} = \emptyset$ .

Our problem is to compute an obstacle-free convex polytope  $\mathcal{P}$  which is required to contain the robot  $\mathcal{Q}$  while excluding all obstacles  $\mathcal{O}_i$ . Besides,  $\mathcal{P}$  should have the largest possible volume within a prescribed region of interest. For convenience, we define that the boundaries of the prescribed region are considered as obstacles and are encoded into  $\mathcal{O}$ , which makes the volume of obstacle-free space surrounding  $\mathcal{Q}$  bounded, as shown in Fig. 1. Concluding above requirements yields the optimization

$$\max \text{vol}(\mathcal{P}), \text{ s.t. } \mathcal{Q} \subseteq \mathcal{P}, \mathcal{O} \subseteq \text{coint}(\mathcal{P}), \quad (3)$$

where  $\text{vol}(\mathcal{P})$  denotes the volume of the convex polytope  $\mathcal{P}$  and  $\text{coint}(\cdot)$  denotes the complement of the interior of a set. Note that the solution set of (3) will never be empty since the robot  $\mathcal{Q}$  itself is already a feasible solution.

---

#### Algorithm 1: FIRI

---

**Input:**  $\mathcal{Q}, \mathcal{O}, \mathcal{E}_0, \rho$   
**Output:**  $\mathcal{P}$

```

1 begin
2    $k \leftarrow 0$ 
3   repeat
4      $k \leftarrow k + 1$ 
5     /* Restrictive Inflation starts */
6      $\mathcal{R} \leftarrow \{1, \dots, R\}$ 
7      $\mathcal{E} \leftarrow \mathcal{E}_{k-1}, \bar{\mathcal{P}} \leftarrow \mathbb{R}^n$ 
8      $\bar{\mathcal{Q}} \leftarrow D_{\mathcal{E}}^{-1} A_{\mathcal{E}}^T (\mathcal{Q} - b_{\mathcal{E}})$ 
9     foreach  $i \in \mathcal{R}$  do
10       $\bar{\mathcal{O}}_i \leftarrow D_{\mathcal{E}}^{-1} A_{\mathcal{E}}^T (\mathcal{O}_i - b_{\mathcal{E}})$ 
11       $a_i \leftarrow \arg \max_{a \in \mathbb{R}^n} a^T a,$ 
        s.t.  $\bar{\mathcal{Q}} \subseteq \mathcal{H}(a), \bar{\mathcal{O}}_i \subseteq \text{coint}(\mathcal{H}(a))$ 
12    end
13    repeat
14       $j \leftarrow \arg \min_{i \in \mathcal{R}} a_i^T a_i$ 
15       $\bar{\mathcal{P}} \leftarrow \bar{\mathcal{P}} \cap \mathcal{H}(a_j)$ 
16       $\mathcal{R} \leftarrow \mathcal{R} \setminus \{i \in \mathcal{R} \mid \bar{\mathcal{O}}_i \subseteq \text{coint}(\mathcal{H}(a_j))\}$ 
17    until  $\mathcal{R} = \emptyset$ 
18     $\mathcal{P}_k \leftarrow A_{\mathcal{E}} D_{\mathcal{E}} \bar{\mathcal{P}} + b_{\mathcal{E}}$ 
19    /* Restrictive Inflation ends */
20    /* MVIE starts */
21     $\mathcal{E}_k \leftarrow \mathcal{E}^*(\mathcal{P}_k)$ 
22    /* MVIE ends */
23  until  $\text{vol}(\mathcal{E}_k) \leq (1 + \rho) \text{vol}(\mathcal{E}_{k-1})$ 
24  return  $\mathcal{P}_k$ 
25 end
```

---

#### B. Proposed Algorithm

For the optimization (3), dealing with constraints on non-convex sets directly is intractable [34] and no known polynomial complexity algorithm has been given so far for the case of

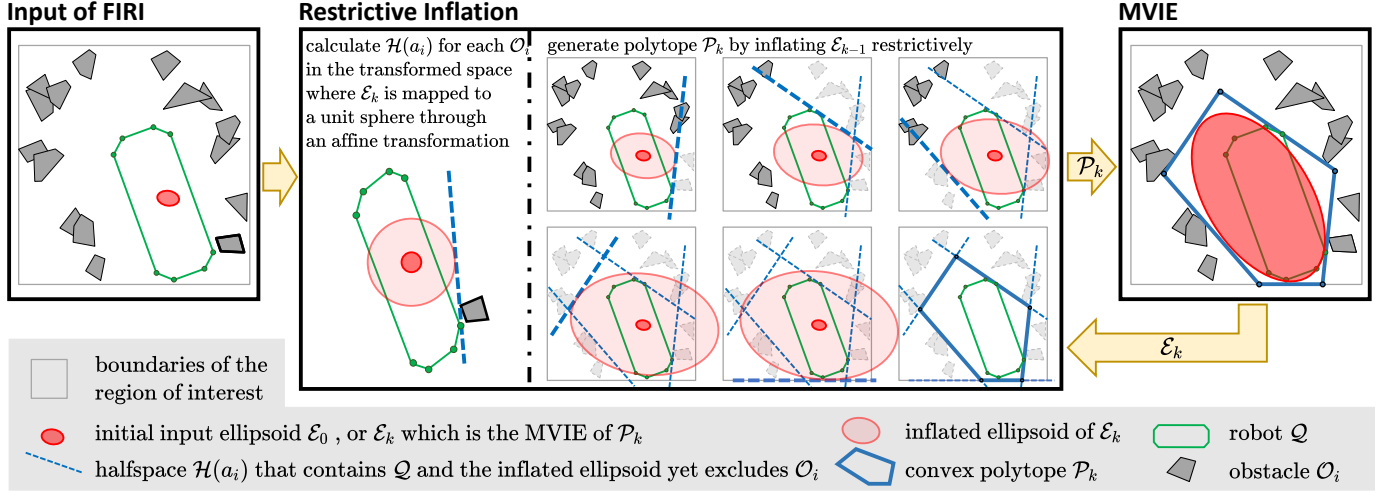


Fig. 1. Illustration of the computation process of FIRI, corresponding to the Restrictive Inflation and MVIE calculation steps depicted in Algorithm. 1.

$n > 3$ . Thus we only seek a high-quality solution to (3) instead of the global optimal solution. Additionally, we optimize a reasonable lower bound of  $\text{vol}(\mathcal{P})$ , which is straightforward to evaluate, to maximize the original objective function  $\text{vol}(\mathcal{P})$  whose computation is NP-hard [17]. As adopted in [1], we also choose the volume of the Maximum Volume Inscribed Ellipsoid (MVIE) of  $\mathcal{P}$  for this lower bound. An ellipsoid is defined by

$$\mathcal{E} = \{p \mid p = A_{\mathcal{E}} D_{\mathcal{E}} x + b_{\mathcal{E}}, \|x\| = 1\}, \quad (4)$$

where  $A_{\mathcal{E}} \in \mathbb{R}^{n \times n}$  is orthonormal,  $D_{\mathcal{E}} \in \mathbb{R}^{n \times n}$  is diagonal and non-negative, and  $b_{\mathcal{E}} \in \mathbb{R}^n$ . The diagonal elements of  $D_{\mathcal{E}}$  correspond to the lengths of the semi-axes of  $\mathcal{E}$ .

The proposed novel algorithm Fast Iterative Region Inflation (FIRI), as shown in Algorithm 1. FIRI takes the robot  $\mathcal{Q}$ , the obstacles  $\mathcal{O}$ , an initial ellipsoid  $\mathcal{E}_0$ , and a precision parameter  $\rho \in (0, 1)$  as inputs. Any ellipsoid that is strictly contained in  $\mathcal{Q}$  can be used to initialize the algorithm. FIRI iteratively executes two modules Restrictive Inflation and solving MVIE within its outer loop (line 4-21). The former takes an ellipsoid as input and expands the ellipsoid to obtain a new convex hull, and the latter takes the convex hull as input and computes its MVIE.

In Restrictive Inflation (line 4-18), For each obstacle  $\mathcal{O}_i$ , we maximally inflate the input ellipsoid  $\mathcal{E}$  under the constraint that there exists a halfspace which does not contain the obstacle yet includes both the inflated ellipsoid and robot  $\mathcal{Q}$ . Then we use these halfspaces to form a new convex polytope as the output of Restrictive Inflation. Specifically,  $\mathcal{Q}$  and  $\mathcal{O}_i$  are transformed (line 8 and 10) by the inverse affine map determined by the ellipsoid  $\mathcal{E}$  generated in the previous iteration. Since  $\mathcal{Q}$  and  $\mathcal{O}_i$  are both in  $\mathcal{V}$ -representation, their images after transformation are still the convex combinations of the images of their vertices, i.e.,

$$\bar{\mathcal{Q}} = \text{conv}\{D_{\mathcal{E}}^{-1} A_{\mathcal{E}}^T (v_j - b_{\mathcal{E}}), 1 \leq j \leq s\}, \quad (5)$$

$$\bar{\mathcal{O}}_i = \text{conv}\{D_{\mathcal{E}}^{-1} A_{\mathcal{E}}^T (u_{i,j} - b_{\mathcal{E}}), 1 \leq j \leq s_i\}. \quad (6)$$

The affine transformation can be massively parallelized for all vertices. It is important that in the transformed space the

ellipsoid  $\mathcal{E}$  is transformed to a unit ball  $\mathcal{B}$  as

$$\mathcal{B} = \{p \mid \|p\| = 1\}, \quad (7)$$

whose collision check is far cheaper to handle than the ellipsoid  $\mathcal{E}$ . Then the maximum inflation ratio of the unit ball  $\mathcal{B}$  for each transformed obstacle  $\bar{\mathcal{O}}_i$  is computed in the first inner loop (line 10-11). Compared to the methods that take the managibility to include the robot [1], [4], the proposed Restrictive Inflation compute a halfspace for each obstacle  $\bar{\mathcal{O}}_i$  such that it contains  $\bar{\mathcal{Q}}$  and the corresponding inflated ball  $\mathcal{B}_i$  yet excludes  $\bar{\mathcal{O}}_i$  while maximizing the  $\mathcal{B}_i$  which is defined by

$$\mathcal{B}_i = \{p \mid \|p\| = a_i^T a_i\}, \quad (8)$$

where  $a_i$  is the only contact point of the halfspace with the inflated ball  $\mathcal{B}_i$ . The halfspace  $\mathcal{H}(a_i)$  is defined as

$$\mathcal{H}(a_i) := \{x \in \mathbb{R}^n \mid a_i^T x \leq a_i^T a_i\}. \quad (9)$$

Then Restrictive Inflation can be written as an optimization as shown in line 11. The second inner loop (line 14-16) iteratively finds the closest allowable halfspace, adds it into  $\bar{\mathcal{P}}$ , and removes unused obstacles. If all obstacles in  $\mathcal{I}$  have been processed, a feasible polytope in  $\mathcal{H}$ -representation [33] can be formed in the transformed space. Finally, a new polytope  $\mathcal{P}_k$  is generated by recovering  $\bar{\mathcal{P}}$  to the original space (line 18). Thus, the output of FIRI is an obstacle-free convex polytope in  $\mathcal{H}$ -representation, i.e., intersection of  $m$  halfspaces

$$\mathcal{P} = \{x \in \mathbb{R}^n \mid A_{\mathcal{P}} x \leq b_{\mathcal{P}}\}, \quad (10)$$

where  $A_{\mathcal{P}} \in \mathbb{R}^{m \times n}$  and  $b_{\mathcal{P}} \in \mathbb{R}^m$ .

For computing the MVIE of a convex polytope (line 21), as long as a bounded  $\mathcal{P}$  has nonempty interior, its unique MVIE [35] is determined by

$$\mathcal{E}^*(\mathcal{P}) = \max_{A_{\mathcal{E}}, D_{\mathcal{E}}, b_{\mathcal{E}}} \text{vol}(\mathcal{E}), \text{ s.t. } \mathcal{E} \subseteq \mathcal{P}. \quad (11)$$

FIRI continuously optimizes the lower bound of the objective function of the original problem (3). During the iterative computation, this algorithm always ensures the feasibility of the solution, which means the generated convex polytope  $\mathcal{P}_k$  maintains satisfying the constraints of the original problem

as  $k$  increases. Moreover, it maintains the monotonicity of the volume of ellipsoid  $\mathcal{E}_k$  as the lower bound of the volume of the convex polytope  $\mathcal{P}_k$ .

For the feasibility, each halfspace solved in Restrictive Inflation (line 11) satisfies the original constraints, thus the new convex polytope  $\mathcal{P}_k$  formed by the intersection of these halfspaces is guaranteed to be feasible, given by

$$\mathcal{Q} \subseteq \mathcal{P}_k, \mathcal{O} \subseteq \text{coint}(\mathcal{P}_k). \quad (12)$$

Additionally, as defined in Sec III-A, the prescribed region of interest is bounded and the boundaries of this region are encoded into  $\mathcal{O}$ , thus the feasibility also indicates that the generated convex polytope  $\mathcal{P}_k$  is closed.

For the monotonicity, in the  $k$ -th iteration, we inflate the unit ball  $\mathcal{B}$  for each transformed obstacle  $\bar{\mathcal{O}}_i$  in Restrictive Inflation, which always ensures the contact point satisfies  $\|a_i\| \geq 1$ . Thus, in the transformed space, the MVIE of  $\bar{\mathcal{P}}$  acquired from the second inner loop (line 14-16) can never be smaller than the unit ball  $\mathcal{B}$  as

$$\text{vol}(\mathcal{E}^*(\bar{\mathcal{P}})) \geq \text{vol}(\mathcal{B}). \quad (13)$$

Since the relative size relationship between the MVIE of  $\bar{\mathcal{P}}$  and the unit ball  $\mathcal{B}$  is invariant to affine maps, and  $\mathcal{B}$  is obtained by applying the inverse affine map to the input ellipsoid  $\mathcal{E}_{k-1}$ , in the original space (line 18 and 21) we have

$$\text{vol}(\mathcal{E}_k) \geq \text{vol}(\mathcal{E}_{k-1}). \quad (14)$$

which gives the monotonicity.

A sequence  $\{\mathcal{P}_1, \mathcal{E}_1, \dots, \mathcal{P}_k, \mathcal{E}_k, \dots\}$  is generated by repeating the iteration. Since  $\text{vol}(\mathcal{E}_k)$  is non-decreasing and bounded, the ellipsoid volume will converge to a finite value. Consequently, FIRI terminates when this lower bound for  $\text{vol}(\mathcal{P}_k)$  cannot be sufficiently improved (line 23).

Algorithm 1 describes a skeleton of FIRI. In conclusion, Restrictive Inflation brings managibility to FIRI, and monotonically iterative updates allows FIRI to continuously optimize the lower bound of (3) to obtain a high quality output. It is worth emphasizing that the efficiency of FIRI relies strongly on the performance of solving two optimization programming (line 11 and 21). In subsequent sections, we propose novel sub-algorithms that are practically efficient and reliable, exploiting the geometrical structure of the involved programming, which greatly benefits the computational efficiency of FIRI.

#### IV. SOLVING RESTRICTIVE HALFSPACE COMPUTATION VIA SDQP

Combining the definition of the halfspace  $\mathcal{H}(a_i)$  in (9), we formulate the halfspace calculation in Restrictive Inflation defined in line 11 of Algorithm 1 into

$$\max_{a_i \in \mathbb{R}^n} a_i^T a_i, \quad (15a)$$

$$\text{s.t. } v^T a_i \leq a_i^T a_i, \forall v \in \bar{\mathcal{Q}}, \quad (15b)$$

$$u^T a_i \geq a_i^T a_i, \forall u \in \bar{\mathcal{O}}_i. \quad (15c)$$

Although such a inflated ball  $\mathcal{B}_i$  maximization is nonconvex, we can obtain its equivalent convex QP formulation through its polar duality. Algorithm 1 keeps  $a_i^T a_i > 0$ , which ensures that

the origin always lies within the interior of the the halfspace  $\mathcal{H}(a_i)$ . Thus we directly optimize the polar dual vector of  $a_i$  by substituting  $a_i = b/(b^T b)$  and obtain an equivalent QP

$$\min_{b \in \mathbb{R}^n} b^T b, \quad (16a)$$

$$\text{s.t. } v^T b \leq 1, \forall v \in \bar{\mathcal{Q}}, \quad (16b)$$

$$u^T b \geq 1, \forall u \in \bar{\mathcal{O}}_i. \quad (16c)$$

Note that (16) is a  $L_2$ -norm minimization problem after inverse affine map in Algorithm 1 (line 8 and line 10), whereas it takes the general form of a strictly convex QP before map, and its positive definite matrix of the quadratic objective function is determined by the ellipsoid matrix of current iteration. In other words, the first inner loop of Algorithm 1 implicitly transforms a strictly convex QP into an equivalent form of  $L_2$ -norm minimization.

To efficiently perform the restrictive halfspace computation, we propose a randomized method, **SDQP**, for strictly convex QPs with small dimension but massive constraints, which computes exact solutions. This method generalizes Seidel's algorithm from Linear Programming (LP) to strictly convex QP, and enjoys linear expected complexity about the constraint number. Then we provide in detail the flow of SDQP: 1) converting a strictly convex QP into  $L_2$ -norm minimization, 2) solving the minimization analytically in linear complexity.

We consider the following strictly convex QP,

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} x^T M_Q x + c_Q^T x, \text{ s.t. } A_Q x \leq b_Q, \quad (17)$$

where  $M_Q \in \mathbb{R}^{n \times n}$  is positive definite,  $c_Q \in \mathbb{R}^n$ ,  $A_Q \in \mathbb{R}^{d \times n}$ , and  $b_Q \in \mathbb{R}^d$ .  $d$  denotes the number of constraints, much larger than  $n$ . Thus the Cholesky factorization of  $M_Q$  exists and is unique, which we denote as  $M_Q = L_Q L_Q^T$ . We introduce a new variable  $y \in \mathbb{R}^n$  that satisfies

$$y = L_Q^T x + L_Q^{-1} c_Q. \quad (18)$$

Then the original QP (17) is equivalent to a  $L_2$ -norm minimization below,

$$\min_{y \in \mathbb{R}^n} \frac{1}{2} y^T y, \text{ s.t. } E y \leq f, \quad (19)$$

where  $E = A_Q L_Q^{-T}$  and  $f = A_Q (L_Q L_Q^T)^{-1} c_Q + b_Q$ . It is apparent that solving the minimization (19) can directly yield the solution to the associated QP (17), and this transformation is implicitly executed in Algorithm 1. Hereafter, we present a randomized algorithm with linear complexity for the low-dimensional  $L_2$ -norm minimization (19) in Algorithm 2.

As the input of Algorithm 2, we define  $\mathcal{H}_E$  as the set of the hyperplanes corresponding to the constraints of the  $L_2$ -norm minimization (19). The central idea of Algorithm 2 is to start with  $y = \mathbf{0}$  which is the solution of an unconstrained  $L_2$ -norm minimization (line 2), and then gradually check the constraint  $h \in \mathcal{H}_E$  in random order (line 7), denoting  $I$  as the set of the constraints that have already been checked, as illustrated in Fig. 2(a). We check whether the solution under the constraints  $I$  violates the new constraint  $h$ . If it is not violated as shown in Fig. 2(b), next constraint will be checked. If it is violated as shown in Fig. 2(c),  $h$  must be active for

**Algorithm 2: LowDimMinNorm**


---

**Input:**  $\mathcal{H}_E$   
**Output:**  $y$

```

1 begin
2    $y \leftarrow 0$ 
3   if  $\dim(\mathcal{H}_E) == 1$  then
4      $y \leftarrow \text{OneDimMinNorm}(\mathcal{H}_E)$ 
5   end
6    $\mathcal{I} \leftarrow \{\}$ 
7   foreach  $h \in \mathcal{H}_E$  in a random order do
8     if  $y \notin h$  then
9        $\{M, v, \mathcal{H}_{E'}\} \leftarrow \text{HouseholderProj}(\mathcal{I}, h)$ 
10       $y' \leftarrow \text{LowDimMinNorm}(\mathcal{H}_{E'})$ 
11       $y \leftarrow My' + v$ 
12    end
13     $\mathcal{I} \leftarrow \mathcal{I} \cup \{h\}$ 
14  end
15  return  $y$ 
16 end

```

---

the  $L_2$ -norm minimization under constraints  $\mathcal{I} \cup \{h\}$ . That is, the solution must be on the constraint plane of  $h$ , thus minimization of this iteration can be written as

$$\min_{y \in \mathbb{R}^n} \frac{1}{2} y^T y, \quad (20a)$$

$$\text{s.t. } E_{\mathcal{I}} y \leq f_{\mathcal{I}}, \quad (20b)$$

$$E_h y = f_h, \quad (20c)$$

where  $E_{\mathcal{I}}$  and  $f_{\mathcal{I}}$  represent the coefficients corresponding to the hyperplanes in  $\mathcal{I}$ ,  $E_h$  and  $f_h$  represent the coefficients of  $h$ . Using the geometric structure of this problem, we transform the minimization into a subproblem of  $n - 1$  dimensional  $L_2$ -norm minimization with the same form as (19) as shown in Fig. 2(d), which will then be presented in detail. This allows for a recursive call of Algorithm 2 (line 10). When  $n = 1$ , the problem (19) is equivalent to a trivial problem of computing the smallest absolute value in a interval, and the feasibility determination of the problem and solution can be calculated directly. We assume that the subproblem can be successfully solved, and thus the solution of this iteration can be calculated (line 11), then the next constraint can be checked.

An essential process of Algorithm 2 is the transformation of (20) into a  $n - 1$  dimensional subproblem with the same form as problem (19) in line 9. The efficiency and numerical stability of this process have a great impact on the efficiency and stability of Algorithm 2. Since  $L_2$ -norm is invariant under the orthogonal transformation, we establish a new  $n - 1$  dimensional Cartesian coordinate system on the constraint plane (20c). As illustrated in Fig. 2(c), we take the point of  $L_2$ -norm minimization on the constraint plane as the origin of the coordinate system,

$$v = \frac{f_h E_h^T}{E_h E_h^T}. \quad (21)$$

It is obvious that any point  $y$  on the constraint plane satisfies

$$\frac{1}{2} y^T y = \frac{1}{2} (y - v)^T (y - v) + \frac{1}{2} v^T v. \quad (22)$$

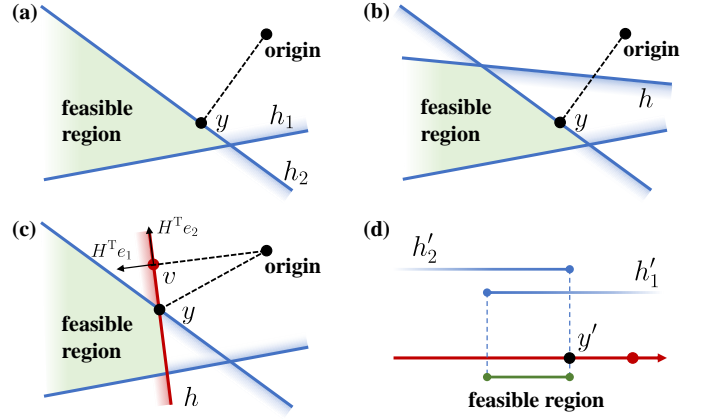


Fig. 2. Illustration of Algorithm 2 in 2-D. (a) indicates that both the inequality constraints  $h_1$  and  $h_2$  have been checked, which means  $h_1, h_2 \in \mathcal{I}$ . And  $y$  is the solution of 2-D  $L_2$ -norm minimization under  $\mathcal{I}$ . (b) illustrates the case where  $y$  does not violate the newly added constraint  $h$ . (c) shows the case where the original solution under  $\mathcal{I}$  violates the new constraint  $h$ , thus we need to find a new solution  $y$  in the constraint plane corresponding to  $h$ , implying equation constraint. The vector  $v$  is normal to the constraint plane, and  $H^T e_1, H^T e_2$  are a set of orthogonal basis of the 2-D space, where  $H^T e_2 \perp v$ . In (d), we establish a new coordinate for the 1-D subspace in the constraint plane with  $v$  as the origin and  $H^T e_2$  as the orthogonal basis, transform the checked constraints to this coordinate as  $h'_1, h'_2$ , and finally transforms (c) into a 1-D  $L_2$ -norm minimization with only inequality constraints.

which means that, with the constraint (20c), the  $L_2$ -norm minimization of  $y$  is equivalent to the  $L_2$ -norm minimization of the  $(y - v)$  which can be transformed into an  $n - 1$  dimensional vector in the newly established coordinate system.

Then we establish the new coordinate system by Householder reflection.  $v$  is a normal vector of the constraint plane proportional to the geometric scale of the problem, which has intuitive numerical stability. Thus we use the normal vector  $v$  to compute the Householder reflection. Taking the index of the element of  $v$  with the largest absolute value as

$$j = \operatorname{argmax}_{k \in \{1, \dots, n\}} \|v_k\|. \quad (23)$$

Then the obtuse reflection vector  $u$  that transforms  $v$  to be parallel to  $e_j$  is

$$u = v + \operatorname{sgn}(v_j) \|v\| e_j, \quad (24)$$

whose corresponding Householder matrix is

$$H = \mathbf{I}_n - \frac{2uu^T}{u^T u}, \quad (25)$$

which is an orthogonal matrix. The use of the obtuse reflection vector  $u$  corresponding to the dimension of the largest absolute value element of  $v$  prevents the Householder transformation from being ill-conditioned due to a small reflection angle. This procedure is implicitly equivalent to a single-step operation of the Householder QR factorization. Easy to verify

$$Hv = -\operatorname{sgn}(v_j) \|v\| e_j, \quad (26)$$

which indicates that the set of all col vectors of the orthogonal  $H^T$  except the  $j$ -th one form an orthonormal basis for the orthogonal complement of  $v$ , as shown in Fig. 2(c). This process of calculating orthogonal basis has higher numerical stability than the Gram-Schmidt orthogonalization which may



fall victim to the catastrophic cancelation problem [36]. Since  $v$  is normal to constraint plane, we adopt this orthonormal basis to define the new coordinate system of the  $n - 1$  dimensional subspace on the constraint plane, and introduce the corresponding coordinates  $y' \in \mathbb{R}^{n-1}$ . We denote  $M \in \mathbb{R}^{n \times (n-1)}$  as the matrix obtained by removing the  $j$ -th column of  $H^T$ . Then the point  $y$  in the origin coordinate system corresponding to  $y'$  is

$$y = My' + v. \quad (27)$$

According to the decomposition (22) and (27), minimizing the  $L_2$ -norm of  $y$  on the constraint plane is equivalent to minimizing

$$\frac{1}{2} \|My'\|^2 = \frac{1}{2} y'^T M^T M y' = \frac{1}{2} y'^T y', \quad (28)$$

Eventually, for (20) with a equality constraint, we construct its corresponding  $n - 1$  dimensional  $L_2$ -norm minimization with only linear inequality constraints on the constraint plane as

$$\min_{y' \in \mathbb{R}^{n-1}} \frac{1}{2} y'^T y', \quad (29a)$$

$$\text{s.t. } E_{\mathcal{I}} M y' \leq f_{\mathcal{I}} - E_{\mathcal{I}} v. \quad (29b)$$

As Fig. 2(d) shows, the linear inequality constraints on  $y'$  can be obtained by substituting (27) into  $E_{\mathcal{I}} y \leq f_{\mathcal{I}}$ , and the set of their corresponding hyperplanes is denoted as  $\mathcal{H}_E'$  (line 9-10). Obviously, (29) has the same structure as (19) and has a lower dimension, thus it can be solved by recursively calling Algorithm 2 (line 10).

Finally, we complete the extension of Seidel's algorithm [9] to strictly positive definite QP with inequality constraints. There is a probability to trigger a recursive calls of the  $n - 1$  dimensional subproblem in each iteration. The idea of this randomized algorithm is that when the dimension  $n$  of the problem is small but the constraint scale  $d$  is huge, the probability of a newly added constraint becoming a active constraint is tiny, hence the probability of recursive calls is tiny as well. The expected complexity of this algorithm is  $O(n!d)$ , so that in the common dimensions  $n \in \{2, 3\}$ , the expected complexity only increases linearly with the constraint scale [9]. In addition, since the algorithm preprocess the order of constraints by a random permutation, its actual performance does not depend on the input order and almost always achieves expected linear complexity. With the help of this randomized algorithm, the computational complexity of handling all obstacles in the constrained inflation of Algorithm 1 grows linearly only with the total number of vertices of the obstacles.

## V. SOLVING SOCP-REFORMULATION OF MVIE VIA AFFINE SCALING ALGORITHM

In this section, we propose a practically efficient algorithm for MVIE in a low dimension but with a large number of constraint. By directly optimizing the Cholesky factorization of the ellipsoid matrix, we reformulate the MVIE problem into a Second-Order Conic Programming (SOCP) form. Then we provide the solution of using Affine Scaling algorithm for solving SOCP, achieving efficient solving of MVIE.

As the coefficients of the ellipsoid  $\mathcal{E}$  defined in (4), the diagonal elements of  $D_{\mathcal{E}}$  are the lengths of the semi-axes of  $\mathcal{E}$ . Thus the objective  $\text{vol}(\mathcal{E})$  of MVIE (11) is proportional to the determinant  $\det(D_{\mathcal{E}})$ . The semi-infinite constraint  $\mathcal{E} \subseteq \mathcal{P}$  is equivalent to  $[[A_{\mathcal{P}} A_{\mathcal{E}} D_{\mathcal{E}}]^2 \mathbf{1}]^{\frac{1}{2}} \leq b_{\mathcal{P}} - A_{\mathcal{P}} b_{\mathcal{E}}$  [16] which is a second-order cone (SOC) constraint, where  $[\cdot]$  implies entry-wise operations. Thus, problem (11) can be written as

$$\max_{A_{\mathcal{E}}, D_{\mathcal{E}}, b_{\mathcal{E}}} \det(D_{\mathcal{E}}), \quad (30a)$$

$$\text{s.t. } [[A_{\mathcal{P}} A_{\mathcal{E}} D_{\mathcal{E}}]^2 \mathbf{1}]^{\frac{1}{2}} \leq b_{\mathcal{P}} - A_{\mathcal{P}} b_{\mathcal{E}}, \quad (30b)$$

$$D_{\mathcal{E}} = \text{diag}(d_{\mathcal{E}}), \quad d_{\mathcal{E}} \in \mathbb{R}_{\geq 0}^n, \quad (30c)$$

$$A_{\mathcal{E}}^T A_{\mathcal{E}} = I_n, \quad (30d)$$

where  $\text{diag}(\cdot)$  indicates either constructing a diagonal matrix or taking all diagonal entries of a square matrix, and  $A_{\mathcal{P}}$ ,  $b_{\mathcal{P}}$  are the coefficients of the halfspace constituting the convex polyhedra  $\mathcal{P}$ . However, this program still imposes orthonormality constraints on  $A_{\mathcal{E}}$ . Additionally, if we use  $A_{\mathcal{E}} D_{\mathcal{E}}$  as the decision variable, (30) become an SDP form which is often used to solve MVIE [1], [16], [19].

Noting that  $A_{\mathcal{E}} D_{\mathcal{E}}^2 A_{\mathcal{E}}^T$  is always positive definite for the optimal solution of the non-degenerate problem (30). Thus the Cholesky factorization  $A_{\mathcal{E}} D_{\mathcal{E}}^2 A_{\mathcal{E}}^T = L_{\mathcal{E}} L_{\mathcal{E}}^T$  is unique [37] for this solution, where  $L_{\mathcal{E}}$  is a lower triangular matrix with positive diagonal entries. If we treat  $L_{\mathcal{E}}$  as decision variables, since  $A_{\mathcal{E}}$  is orthonormal, the objective (30a) and constraints (30b) can be written as

$$\det(D_{\mathcal{E}}) = \sqrt{\det(A_{\mathcal{E}} D_{\mathcal{E}}^2 A_{\mathcal{E}}^T)} = \det(L_{\mathcal{E}}). \quad (31)$$

Consequently, the orthonormality constraint (30d) on  $A_{\mathcal{E}}$  is avoided, and (30) is equivalent to

$$\max_{L_{\mathcal{E}}, b_{\mathcal{E}}} \det(L_{\mathcal{E}}), \quad (32a)$$

$$\text{s.t. } [[A_{\mathcal{P}} L_{\mathcal{E}}]^2 \mathbf{1}]^{\frac{1}{2}} \leq b_{\mathcal{P}} - A_{\mathcal{P}} b_{\mathcal{E}}, \quad (32b)$$

$$L_{\mathcal{E}} \text{ is lower triangular}, \quad (32c)$$

Now the objective  $\det(L_{\mathcal{E}})$  is simply the product of the diagonal entries. Denoting the hypograph of geometric mean by

$$\mathcal{K}_{1/n} = \left\{ (x, t) \in \mathbb{R}_{\geq 0}^n \times \mathbb{R} \mid (x_1 \cdots x_n)^{\frac{1}{n}} \geq t \right\}, \quad (33)$$

thus maximizing the volume of the ellipsoid  $\mathcal{E}$  is equivalent to maximizing  $t$  with the constraint  $(\text{diag}(L_{\mathcal{E}}), t) \in \mathcal{K}_{1/n}$ . As for the constraints (30b), we denote the SOC as  $\mathcal{K}_n$  and describe the Cartesian product of  $m$  SOC as  $\mathcal{K}_n^m$ ,

$$\mathcal{K}_n = \left\{ (t, x) \in \mathbb{R} \times \mathbb{R}^{1 \times n-1} \mid t \geq \|x\| \right\}, \quad (34)$$

$$\mathcal{K}_n^m = \mathcal{K}_n \times \cdots \times \mathcal{K}_n \subseteq \mathbb{R}^{m \times n}. \quad (35)$$

The MVIE problem now becomes

$$\max_{t, L_{\mathcal{E}}, b_{\mathcal{E}}} t, \quad (36a)$$

$$\text{s.t. } (\text{diag}(L_{\mathcal{E}}), t) \in \mathcal{K}_{1/n}, \quad (36b)$$

$$(b_{\mathcal{P}} - A_{\mathcal{P}} b_{\mathcal{E}}, A_{\mathcal{P}} L_{\mathcal{E}}) \in \mathcal{K}_{n+1}^m, \quad (36c)$$

$$L_{\mathcal{E}} \text{ is lower triangular}. \quad (36d)$$

$\mathcal{K}_{1/n}$  can also be represented by SOC using additional  $O(n)$  variables and cones of  $\mathcal{K}_3$  [16].

Thus, we reformulate MVIE into a pure SOCP from (36). For brevity, we denote the it as

$$\min_x c_{\mathcal{K}}^T x, \quad (37a)$$

$$\text{s.t. } (c_i^T x + d_i, x^T A_i) \in \mathcal{K}_{n_i}, \quad 1 \leq i \leq \bar{m}, \quad (37b)$$

where  $x \in \mathbb{R}^{\bar{n}}$ ,  $c_{\mathcal{K}} \in \mathbb{R}^{\bar{n}}$ ,  $c_i \in \mathbb{R}^{\bar{n}}$ ,  $d_i \in \mathbb{R}$  and  $A_i$  are all constant. The new decision variable  $x$  consists of all lower triangular elements of  $L_{\mathcal{E}}$ ,  $b_{\mathcal{E}}$ ,  $t$  of (36), and the elements added in order to deal with constraint (36b) in form of hypograph of geometric mean. We have  $\bar{n} = O(n^2)$  because an  $n$  dimensional ellipsoid already has  $n(n+3)/2$  variables. For convenience, we set  $t$  to be the  $\bar{n}$ -th element of  $x$ , then we have

$$c_{\mathcal{K}} = (0, 0, \dots, 0, -1)^T \in \mathbb{R}^{\bar{n}}. \quad (38)$$

$\bar{m} = O(m+n)$  indicates the amount of the SOC constraints. For the constraints in (37b) corresponding to the origin constraints of (36b) which are represented by additional cones of  $\mathcal{K}_3$ ,  $A_i \in \mathbb{R}^{\bar{n} \times 2}$  and  $n_i = 3$ . For the constraints in (37b) corresponding to the origin constraints of (36c),  $A_i \in \mathbb{R}^{\bar{n} \times n}$  and  $n_i = n+1$ .

Then we generalize the Affine Scaling (AS) [38], [39], an interior point method, for efficiently solving the SOCP (37). At each iteration, AS uses the logarithmic barrier function of the constraints to compute a strictly feasible region and calculates the optimal step within this region. For (37), its logarithmic barrier function  $\phi(x)$  is defined as

$$\phi(x) = - \sum_{i=1}^{\bar{m}} \log(f_i(x)), \quad (39)$$

$$f_i(x) = (c_i^T x + d_i)^2 - x^T A_i A_i^T x. \quad (40)$$

Then the Hessian of  $\phi(x)$  is given by

$$H_{\phi}(x) = \sum_{i=1}^{\bar{m}} \frac{1}{f_i(x)^2} \nabla f_i(x) \nabla f_i(x)^T - \sum_{i=1}^{\bar{m}} \frac{1}{f_i(x)} \nabla^2 f_i(x), \quad (41)$$

where the gradient and Hessian of  $f_i(x)$  are

$$\nabla f_i(x) = 2(c_i^T x + d_i) c_i - 2A_i A_i^T x, \quad (42)$$

$$\nabla^2 f_i(x) = 2(c_i c_i^T - A_i A_i^T). \quad (43)$$

For brevity, we denote  $\mathcal{K}$  as the feasible region of (37), and the interior of the feasible region as  $\mathcal{K}^\circ$ . Now we obtain the region of AS at  $k$ -th iteration with the solution  $x_k \in \mathcal{K}^\circ$  as

$$\{x \mid (x - x_k)^T H_{\phi}(x_k)(x - x_k) \leq 1\}, \quad (44)$$

which is an ellipsoidal region as  $H_{\phi}(x_k)$  is positive definite. Thus the update step of (37) can be given by

$$x_{k+1} = x_k - \tau \frac{H_{\phi}^{-1} c_{\mathcal{K}}}{\sqrt{c_{\mathcal{K}}^T H_{\phi}^{-1} c_{\mathcal{K}}}}, \quad (45)$$

where  $\tau \in (0, 1]$  is the step size.

## VI. SOLVING 2-D MVIE ANALYTICALLY AND RAPIDLY VIA RANDOMIZED ALGORITHM

For 2-D MVIE, we reformulate it into an enumeration of many subproblems which can be solved analytically. Instead of violent enumeration, we implement a randomized algorithm to achieve an efficient lazy enumeration based on a bottom-up strategy. This lazy randomized algorithm is presented in Sec. VI-A and the analytic solution of the subproblems is detailed in Sec. VI-B.

### A. Randomized Maximal Inscribed Ellipse Algorithm via Bottom-Up Strategy

In contrast to the numerical solution presented in Sec. V, we propose a novel approach for 2-D MVIE of the convex polygon  $\mathcal{P}$ , based on the subproblem

$$\bar{\mathcal{E}}(\mathcal{P}^N) = \max_{\mathcal{E}} \text{aera}(\mathcal{E}), \quad (46a)$$

$$\text{s.t. } \mathcal{E} \subseteq \bar{\mathcal{P}}^N, \quad (46b)$$

$$\mathcal{E} \cap h^\circ \neq \emptyset, \quad \forall h \in \bar{\mathcal{P}}^N, \quad (46c)$$

where  $\bar{\mathcal{P}}^N$  is a subset of  $\mathcal{P}$  that is the set of the finite halfspaces whose intersection forms the input polygon,  $i$  indicates the amount of elements of  $\bar{\mathcal{P}}^N$ , and the boundary of a halfspace  $h$  is denoted as  $h^\circ$  which is a hyperplane. Since an ellipse has only five degrees of freedom, we require  $N \leq 5$ . We define  $\bar{\mathcal{E}}$  as infinity when the space consisting of the elements of  $\bar{\mathcal{P}}^N$  is not closed. When  $\bar{\mathcal{P}}^N$  is closed and has no redundant element, (46) has an analytical solution, and its result  $\bar{\mathcal{E}}$  is the Maximal Ellipse tangent to  $N$  edges of the  $N$ -gon formed by  $\bar{\mathcal{P}}^N$ , denoted as MENN, which is detailed in Sec. VI-B. But if there are redundant elements in a closed  $\bar{\mathcal{P}}^N$ , we define its  $\bar{\mathcal{E}}$  as a point whose aera is 0. Then based on (46), the MVIE of  $\mathcal{P}$  defined in (11) is equivalent to

$$\mathcal{E}^* = \max_{\bar{\mathcal{P}}^N \subseteq \mathcal{P}} \text{aera}(\bar{\mathcal{E}}), \quad \text{s.t. } \bar{\mathcal{E}} \subseteq \mathcal{P}, \quad (47)$$

which can be solved through enumeration. Inspired by the subalgorithm of GJK [22], we organize the enumeration in an orderly and recursive way by using a bottom-up strategy, about which we illustrate a detailed example in Fig. 3.

As demonstrated in the orange box of Fig. 3, the MVIE computation of  $\bar{\mathcal{P}}_1^4$  that forms a convex quadrilateral are performed. If the MVIE of its any subset  $\bar{\mathcal{P}}_j^3$ ,  $1 \leq j \leq 4$  satisfies the constraint of being included by  $\bar{\mathcal{P}}_1^4$ , we can simply select the largest one among the MVIEs of its subsets  $\bar{\mathcal{P}}_j^3$  which satisfy this constraint, as the MVIE of  $\bar{\mathcal{P}}_1^4$ . That is, there is no need to explicitly compute the MENN of  $\bar{\mathcal{P}}_1^4$  by (46), which is shown on rightmost side of the orange box. Because the MENN of  $\bar{\mathcal{P}}_1^4$  is one of the inscribed ellipses of the triangle formed by one of its subset  $\bar{\mathcal{P}}_3^3$ . When the largest one of these inscribed ellipses, which is the MVIE of  $\bar{\mathcal{P}}_3^3$ , is contained by  $\bar{\mathcal{P}}_1^4$ , it indicates that the MENN of  $\bar{\mathcal{P}}_1^4$  is less than or equal to the MVIE of one of its subset  $\bar{\mathcal{P}}_3^3$  as shown in Fig. 3. Similarly, this lazy approach can be applied to the MVIE computation of  $\bar{\mathcal{P}}^5$  in the grey box of Fig. 3.

In order to implement above lazy thought, which does not require an exhaustive enumeration, we adopt the MSW



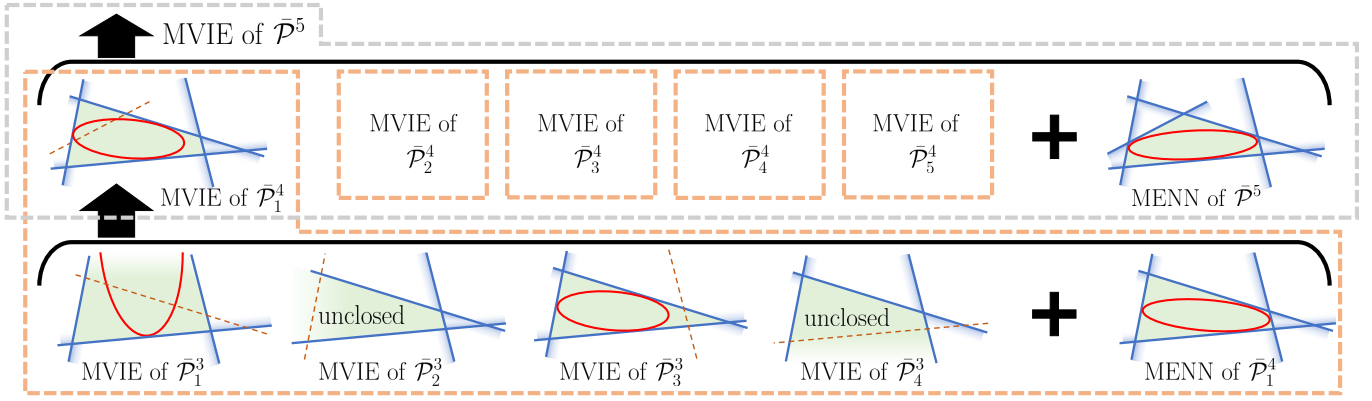


Fig. 3. Illustration of bottom-up strategy. Although the calculation process is bottom-up, we describe it from top to bottom for clarity of presentation. In the figure, our goal is to compute the MVIE of a closed set  $\bar{\mathcal{P}}^5$  containing 5 elements. As shown in the gray box, first we calculate the MVIE for each of its subsets  $\bar{\mathcal{P}}^4_j (1 \leq j \leq 5)$  containing 4 elements, and calculate the MENN of  $\bar{\mathcal{P}}^5$ . Then among these ellipses, we select the largest one that satisfies the constraint of being included by  $\bar{\mathcal{P}}^5$  as the MVIE of  $\bar{\mathcal{P}}^5$ . Except for the MENN of  $\bar{\mathcal{P}}^5$  which can be directly calculated by (46), the MVIE of  $\bar{\mathcal{P}}^4_j$  requires to be obtained in a similar way as used for the MVIE of  $\bar{\mathcal{P}}^5$ . As shown in the orange box, a specific procedure for calculating the MVIE of  $\bar{\mathcal{P}}^4_1$  is given, where its subsets are denoted by  $\bar{\mathcal{P}}^3_k (1 \leq k \leq 4)$ . Similarly, we compute the MVIE for each subsets  $\bar{\mathcal{P}}^3_k$ , and the MENN of  $\bar{\mathcal{P}}^4_1$ . Then among the 5 results, we select the largest one that satisfies the constraint of being included by  $\bar{\mathcal{P}}^4_1$  as the MVIE of  $\bar{\mathcal{P}}^4_1$ . If we extend the content of the diagram downwards and upwards: 1) Since at least 3 halfspaces are required to form a closed space in 2-D, the MVIE of  $\bar{\mathcal{P}}^3_k$  can be directly computed by (46), without the need of splitting and enumeration in the grey or orange box. 2) For a set  $\mathcal{P}$  containing more than 5 elements, its MVIE can be obtained by calculating the MVIE of all its subsets  $\bar{\mathcal{P}}^5$  and selecting the largest ellipse contained within  $\mathcal{P}$ .

algorithm [20], an randomized algorithm that gives an expected time bound linear in the number of constraints for solving LP-type problem [40]. However, with many unclosed cases as shown in Fig. 3, MVIE fails to be directly applied to MSW. We therefore reformulate MVIE into a LP-type problem that satisfies the axioms of MSW. Then we present the reformulation and its implementation into the MSW algorithm.

The new formulation is an optimization problem specified by pairs  $(\mathcal{P}, w)$ ,  $w : 2^{\mathcal{P}} \rightarrow \mathcal{W}$  is a value function. We give three important definitions from MSW: for  $\mathcal{H} \subseteq \mathcal{P}$ ,  $h \in \mathcal{P}$ ,

- $w(\mathcal{H})$  indicates the *value* of  $\mathcal{H}$ ,
- halfspace  $h$  is *violated* by  $\mathcal{H}$ , if  $w(\mathcal{H}) < w(\mathcal{H} \cup \{h\})$ ,
- the *basis* of  $\mathcal{H}$  is the minimal subset of  $\mathcal{H}$  with the same value of  $\mathcal{H}$ ,

based on which, two primitive operations is defined: *violation test* and *basis computation*. The value domain  $\mathcal{W}$  is defined in lexicographical ordering set  $(\mathcal{W}, <)$ . We denote  $<$  as the lexicographical ordering on  $\mathbb{R}^d$ , that is, for  $\forall x, y \in \mathbb{R}^d$ ,  $x < y$ , if  $x_1 < y_1$ , or  $x_1 = y_1$  and  $x_2 < y_2$ , or so on. We extend the ordering to  $\mathbb{R}^n \cup \{-\infty\}$  by the convention that  $-\infty < x$ ,  $\forall x \in \mathbb{R}^d$ , which means a unique minimum value. Then the definition of value function  $w$  is given by

$$w(\mathcal{H}) = \begin{cases} -\infty & \text{if } \mathcal{H} = \emptyset \\ (-\infty, \bar{w}_a(\mathcal{H})) & \text{if } \mathcal{H} \text{ is unclosed,} \\ (0, \bar{w}_s(\mathcal{H})^{-1}) & \text{if } \mathcal{H} \text{ is closed} \end{cases} \quad (48)$$

where  $\bar{w}_a(\mathcal{H})$  is the *angle value* of  $\mathcal{H}$  when the  $\mathcal{H}$  is unclosed, and  $\bar{w}_s(\mathcal{H})$  is the size of the MVIE of  $\mathcal{H}$  when closed. Here the objective of reformulated problem is to compute the basis of  $\mathcal{P}$ , and in the process, the MVIE of  $\mathcal{P}$  will be determined. Then we give the definition of angle value of  $\mathcal{H}$ . We define a new set, denoted as  $\mathcal{H}_v$ , which consists of the vectors corresponding

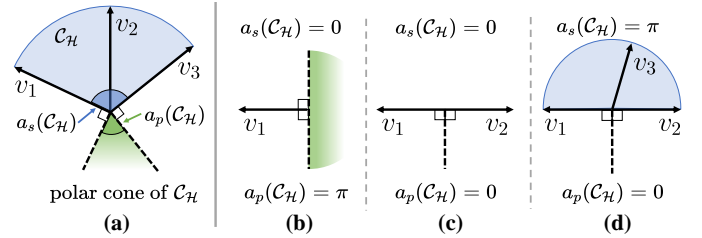


Fig. 4. Illustration of the definition of angle value of  $\mathcal{H}$ . The blue circular sector represents  $\mathcal{C}_\mathcal{H}$ , the green region represents the polar cone of  $\mathcal{C}_\mathcal{H}$ . (b)-(d) illustrate the angle value in the special case where  $v_1$  and  $v_2$  are parallel but in opposite directions.

to the direction of each halfspace in  $\mathcal{H}$  and give

$$\mathcal{C}_\mathcal{H} = \left\{ v \mid \|v\| \leq 1, v = \sum_{v_i \in \mathcal{H}_v} k_i v_i, k_i \in \mathbb{R}_{\geq 0} \right\}, \quad (49)$$

$$\bar{w}_a(\mathcal{H}) = a_s(\mathcal{C}_\mathcal{H}) - a_p(\mathcal{C}_\mathcal{H}), \quad (50)$$

where  $\mathcal{C}_\mathcal{H}$  is a circular sector,  $a_s(\mathcal{C}_\mathcal{H})$  denotes its central angle and  $a_p(\mathcal{C}_\mathcal{H})$  denotes the angle of its polar cone as shown in Fig. 4. Such a particular definition is intended to deal with the corner cases that arise due to the parallel elements in  $\mathcal{H}$ .

The maximum cardinality of any basis is denoted as combinatorial dimension  $\delta$ , and based on the value function (48) we have  $\delta = 5$ . According to the lemma of MSW, the reformulated problem can be easily transformed into a basis-regular LP-type problem by adding an element to the end of the lexicographical ordering of  $w(\mathcal{H})$  when  $\mathcal{H} \neq \emptyset$ . The element is given by

$$\min\{\delta, |\mathcal{H}|\}, \quad (51)$$

where  $|\mathcal{H}|$  denotes the number of the elements of  $\mathcal{H}$ . Now the problem is adapted to the MSW algorithm, and thus can be solved with finite primitive operations, whose expected number is linear in  $|\mathcal{H}|$ .

**Algorithm 3: MaxEllipse**


---

**Input:**  $\mathcal{H}, \mathcal{X}, v_{\mathcal{X}}$   
**Output:**  $\mathcal{B}, v_{\mathcal{B}}$

```

1 begin
2   UpdateValue( $\mathcal{X}, v_{\mathcal{X}}$ )
3   if  $|\mathcal{X}| \geq \delta$  then
4     return  $\mathcal{X}, v_{\mathcal{X}}$ 
5   end
6    $\mathcal{S} \leftarrow \{\}$ 
7    $\mathcal{B} \leftarrow \mathcal{X}$ 
8    $v_{\mathcal{B}} \leftarrow v_{\mathcal{X}}$ 
9   foreach  $h \in \mathcal{H}$  in a random order do
10    if ViolateTest( $h, v_{\mathcal{B}}, \mathcal{X}$ ) then
11       $\mathcal{B}, v_{\mathcal{B}} \leftarrow \text{MaxEllipse}(\mathcal{S}, \mathcal{X} \cup \{h\}, v_{\mathcal{X}})$ 
12    end
13     $\mathcal{S} \leftarrow \mathcal{S} \cup \{h\}$ 
14  end
15  return  $\mathcal{B}, v_{\mathcal{B}}$ 
16 end

```

---

**Algorithm 4: UpdateValue**


---

**Input:**  $\mathcal{X}, v_{\mathcal{X}}$

```

1 begin
2   if CheckCloesd( $\mathcal{X}$ ) then
3      $\text{MENN}(|\mathcal{X}|, \mathcal{X}, v_{\mathcal{X}}) \triangleright$  detailed in Sec.VI-B
4   else
5      $\text{ConeAngle}(\mathcal{X}, v_{\mathcal{X}})$ 
6   end
7 end

```

---

Then we discuss the primitive operations involved in the above problem by case. For the case when  $\mathcal{H}$  is unclosed, both primitives are trivial. For the closed case, the violation test can be implemented by evaluating (30b), but the basis computation is challenging. The basis here are actually a subset of the edges tangent to the MVIE of  $\mathcal{H}$ . In the original MSW algorithm, the basis of  $\mathcal{H}$  will be directly computed by (46-47), which involves huge amounts of redundant calculations. Thus, we here utilize the bottom-up strategy aforementioned to rebuild the recursive algorithm of MSW, utilizing the property that the basis computation of each subset of  $\mathcal{H}$  is part of basis computation of  $\mathcal{H}$ . Finally, a novel randomized algorithm we proposed for 2-D MVIE are presented in Algorithm 3 and 4.

Algorithm 3 takes as input the set  $\mathcal{H}$  to be checked, the set  $\mathcal{X}$  known to belong to the basis of  $\mathcal{H} \cup \mathcal{X}$ , and  $v_{\mathcal{X}}$  that will update at the beginning of the algorithm (line 2), which is detailed in Algorithm 4. The output of Algorithm 3 is the basis of  $\mathcal{H} \cup \mathcal{X}$  and its value. For the initial call to this algorithm, we set  $\mathcal{H} \leftarrow \mathcal{P}$  and  $\mathcal{X} \leftarrow \{\}$ . Within the loop (line 9-13), each elements  $h$  of  $\mathcal{H}$  is checked sequentially in a randomized order, testing whether  $h$  is violated by  $\mathcal{B}$ . If violated, we recursively call the algorithm (line 11). Based on the enumeration principle in Fig. 3, the number of the elements of  $\mathcal{X}$  in the recursive algorithm cannot exceed the combination dimension  $\delta$  (line 3). In ViolateTest function (line 10), in

addition to the evaluation involved  $h$  and  $v_{\mathcal{B}}$  based on (48-51), in the case where  $h$  is violated by  $\mathcal{B}$ , if  $\mathcal{X} \cup \{h\}$  forms a convex  $N$ -gon but  $N < |\mathcal{X} \cup \{h\}|$ , then false is returned. The reason is that we perform checks based on the bottom-up strategy, the non-redundant closed subset of the closed set  $\mathcal{X} \cup \{h\}$  that contains redundant elements will definitely be checked in the algorithm. Therefore, there is no need to check on the redundant closed set. Moreover, this ensures that the input  $\mathcal{X}$  of Algorithm 4 is free of redundant elements in the closed case. Overall, the thought of bottom-up is embedded in the violation test and recursive calls. For example, when we recursively call Algorithm 3 with the input that meets  $|\mathcal{X}| = 4$ , it means that the basis of each its subset containing 3 edges has already been checked and violated by the fourth constraint. Thus, in Algorithm 4, if the input  $\mathcal{X}$  is closed, we can directly compute its MENN as its MVIE to update  $v_{\mathcal{X}}$ . Otherwise we will compute the angle value of  $\mathcal{X}$  to update  $v_{\mathcal{X}}$ .

By introducing the bottom-up strategy and reformulating MVIE into an LP-type problem, we avoid the need for violent enumeration and redundant calculations in solving (46-47). Eventually, we achieve linear-time complexity for solving 2-D MVIE. Another factor that affects the efficiency of the algorithm is the MENN subproblem (46) solving, for which we present an efficient analytic solution in Sec. VI-B.

*B. Maximal Ellipse tangent to  $N$  edges of the  $N$ -gon*

As demanded in Algorithm 4, in this section we present the analytic computation of the maximal inscribed ellipse that is tangent to all edges of arbitrary convex  $N$ -gon, with  $N \in \{3, 4, 5\}$ , which we denote as MENN. For arbitrary non-degenerate triangle [41], convex quadrilateral [42] or convex pentagon [35], there exists and only exists unique such ellipse. Since the inputs from Algorithm 4 are non-degenerate and convex, in the following we default to the existence and uniqueness of the MENN.

Referring to the definition in (4), in the 2-D case we define the point  $p$  on the boundary of the ellipse  $\mathcal{E}$  to satisfy

$$(p - b_{\mathcal{E}})^T (A_{\mathcal{E}} D_{\mathcal{E}}^2 A_{\mathcal{E}}^T)^{-1} (p - b_{\mathcal{E}}) = 1, \quad (52)$$

We denote the *homogeneous coordinate* [43] of point  $p$  as

$$\hat{p} = (p_0, p_1, 1)^T, \quad (53)$$

where  $*_i$  represents the  $i$ -th element of the vector  $*$ . Since  $A_{\mathcal{E}} D_{\mathcal{E}}^2 A_{\mathcal{E}}^T$  is positive definite symmetric, (52) is a second-degree polynomial equation in two variables  $p_0$  and  $p_1$ . We transform (52) into a homogeneous quadratic form, then the ellipse  $\mathcal{E}$  can be described in homogeneous coordinate as

$$\mathbb{P} = \{\hat{p} \mid \hat{p}^T M_P \hat{p} = 0\}, \quad (54)$$

where the coefficient matrix  $M_P \in \mathbb{R}^{3 \times 3}$  is symmetric.

For the MENN problem in this section, it is intractable to solve the problem by (54) which demand the points of tangency. Thus we adopt to leverage the information of the tangents directly, based on the polarity of points and lines with respect to the ellipse  $\mathcal{E}$  [44]. First, we require an algebraic characterization of a line. Specifically, in the projective plane, given a point with a homogeneous coordinate  $\hat{p}$ , the line

passing through the point  $\hat{p}$  can be denote in the form of *line coordinate* [43] as  $l \in \mathbb{R}^3$  that satisfies

$$\hat{p}^T l = 0, \quad (55)$$

based on which, the calculation of the line coordinate of a line can be performed by the Grassmanian expansion of the homogeneous coordinates of two different points on the line [45]. Then, according to the polarity of ellipse [44], when  $\hat{p}$  fulfils (54), which means  $\hat{p}$  is on the ellipse  $\mathcal{E}$ , the line coordinate  $l$  of the line tangent to  $\mathcal{E}$  at the point  $\hat{p}$  is given as

$$l = M_P \hat{p}. \quad (56)$$

Since the ellipse is not degenerate,  $M_P$  is invertible. Based on (54, 56), similarly to the representation by a set of the points in (54), the ellipse  $\mathcal{E}$  now can be described by the set of all its tangents in line coordinate as

$$\mathbb{L} = \{l \mid l^T M_L l = 0\}, \quad (57)$$

$$M_L \propto M_P^{-1}. \quad (58)$$

where the coefficient matrix  $M_L \in \mathbb{R}^{3 \times 3}$  is symmetric.

Now we can calculate the MENN by utilizing the information of the tangents directly, abandoning the necessity of calculating the points of tangency. Specifically, we first compute the line coordinates of edges of the polygon, which actually are the tangents of the ellipse, through the vertices of the polygon by (55). Then we calculate the coefficient matrix  $M_L$  via the tangents in the line coordinate by (57), based on which the coefficient matrix  $M_P$  in the homogeneous coordinate can be obtained by (58). Finally the ellipse in the desired form of (4) can be calculated by (52) and (54). In the following, we present the implementation details.

1) *MENN of a convex pentagon*: For (57), the symmetric  $M_L$  has 6 variables. Bringing the line coordinates of the five edges of the convex pentagon into (57) respectively, we can obtain a six-element homogeneous system of linear equations consisting of five equations. Since there exists only one unique ellipse tangent to all edges of the convex pentagon [35], the homogeneous system always has a non-trivial solution. Then based on the solution, we can obtain the MENN progressively through the process aforementioned.

2) *MENN of a convex quadrilateral*: In contact with convex pentagon, a convex quadrilateral only provide four tangents. Thus there are a unique one-parameter set of inscribed ellipses that are tangent to all edges of the quadrilateral, whose parameter can be taken to be a prescribed point of contact on any single edge of the quadrilateral [35] as shown in Fig. 5. Additionally, only one of them has the largest area [42], which is the MENN of the quadrilateral.

For simplicity of calculation, as shown in Fig. 5, we translate the quadrilateral so that one of its vertices coincides with the origin and one of the edges connected to that vertex coincides with the  $x$ -axis, which will not change the shape of the quadrilateral. Then we denote the line coordinate of the coinciding edge as  $l_\lambda$ . Inspired by the work of Hayes [46], we introduce new constraint by using the point of tangency on  $l_\lambda$ , and denote the point as  $\hat{p}_\lambda$ . Then we have

$$\hat{p}_\lambda = (\lambda, 0, 1)^T, \quad l_\lambda = [0, 1, 0]^T, \quad (59)$$

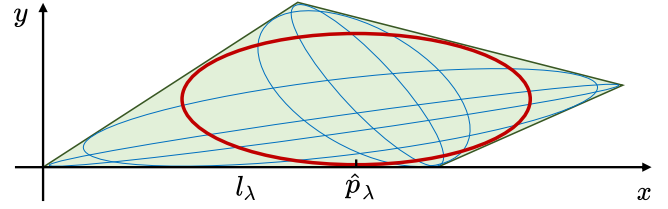


Fig. 5. Illustration of the calculation of the MENN of a convex quadrilateral. The blue and red ellipses represent the ellipses obtained by sampling different points  $\hat{p}_\lambda$  on the coinciding edge  $l_\lambda$ .

where  $\lambda$  is a variable indicating the position of the point. Combining (56) and (58), their polarity relationship can be written as

$$M_L l_\lambda \propto \hat{p}_\lambda, \quad (60)$$

which is an independent new linear constraint on variables of  $M_L$ . By incorporating this constraint with the equations get by substituting the four tangents into (57), we acquire a system of equations similar to Sec. VI-B1. Via the previously described process, we obtain  $M_L$  and  $M_P$  sequentially which here are in terms of  $\lambda$ . Based on  $M_P$ , the area  $A_\mathcal{E}$  of ellipse  $\mathcal{E}$  [46] can be calculated by

$$A_\mathcal{E}(\lambda) = \frac{\det(M_P)}{\det(\tilde{M}_P)^{\frac{3}{2}}} \pi, \quad (61)$$

where  $\tilde{M}_P \in \mathbb{R}^{2 \times 2}$  is the submatrix obtained by removing the leftmost column and the top row of  $M_P$ . Finally, the optimal  $\lambda$  corresponding to the MENN of the convex quadrilateral can be computed by calculating the zeros of the first order derivative of  $A_\mathcal{E}(\lambda)$ . This calculation only involves solving a quadratic equation w.r.t.  $\lambda$ , which can be solved quickly and analytically.

3) *MENN of a triangle*: For a triangle, the Steiner inellipse which is tangent to the three edges of the triangle at their midpoints, has the largest area [41], which is the MENN of the triangle. We denote the 2-D Cartesian coordinates of the vertices of the triangle as  $v^* = (v_0^*, v_1^*)^T$ ,  $*$  = {i, ii, iii}. The center  $v^o$  and two conjugate diameters  $f_1, f_2 \in \mathbb{R}^2$  of the Steiner inellipse can be written as

$$v^o = \frac{1}{3}(v^i + v^{ii} + v^{iii}), \quad (62)$$

$$f_1 = \frac{1}{2}(v^o - v^{iii}), \quad f_2 = \frac{1}{2\sqrt{3}}(v^i - v^{ii}). \quad (63)$$

Then the parameters required in (4) can be calculated by

$$b_\mathcal{E} = v^o, \quad A_\mathcal{E} D_\mathcal{E}^2 A_\mathcal{E}^T = (f_1 \ f_2) (f_1 \ f_2)^T \quad (64)$$

## VII. EVALUATION AND BENCHMARK

In order to comprehensively prove the superiority of the algorithms proposed in this paper, we design comparison experiments for solving MVIE and for generating convex free polytope. We will demonstrate the efficiency of the proposed methods in the benchmark of solving MVIE, which greatly benefits the computational efficiency of the proposed algorithm for convex free space generation, FIRI. Then we will show the outstanding advantages of FIRI in terms of computational efficiency, quality and managibility by comparing FIRI with other state of the art convex polytope generation algorithms

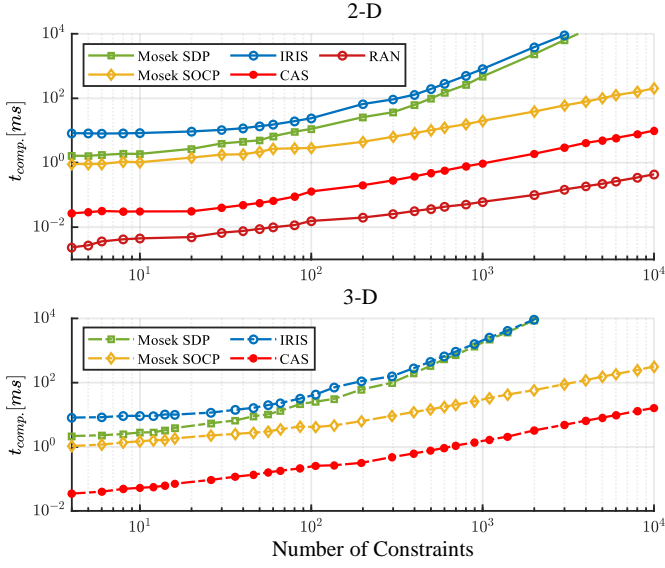


Fig. 6. Computation time of different methods for MVIE under different constraint numbers.

TABLE I  
COMPARISON OF PRECISION  $\psi_{\mathcal{E}}$  BETWEEN DIFFERENT METHODS FOR SOLVING MVIE

Scenario	Precisions $\psi_{\mathcal{E}}$				
	IRIS [1]	Mosek [19] SDP	Mosek [19] SOCP	CAS (Ours)	RAN (Ours)
2-D	8.56e-8	6.47e-9	4.87e-12	1.59e-8	<b>4.41e-16</b>
3-D	1.54e-8	1.56e-8	4.05e-12	2.04e-8	/

extensively. All benchmarks are conducted on an Intel(R) Core(TM) i7-12700KF CPU @ 3.61GHz under Linux, and are implemented in C++11 without any explicit hardware acceleration.

#### A. Comparison of Solving MIVE

For MVIE (11), we propose two algorithms, namely the SOCP-reformulation algorithm and the randomized algorithm specialized for 2-D case, which are presented in Sec. V and VI, respectively. Since the randomized algorithm yields an analytical solution, here we abbreviate it as **RAN**. Then to distinguish from the methods to be compared, we denote the proposed SOCP-reformulation algorithm which is solved by affine scaling method as **CAS**. We compare the proposed algorithms with three methods: 1) The optimization method based on SDP formulation of MVIE in IRIS [1]. 2) The example<sup>1</sup> of the cutting-edge solver Mosek [19] to compute the Lowner-John inner ellipsoidal approximations of a polytope. 3) A strategy of solving the SOCP form (37) of the MVIE by using Mosek. As the example in Mosek formulate MVIE into a mixed conic quadratic and semidefinite problem, we denote it as Mosek SDP below and denote another strategy using Mosek as Mosek SOCP. We use the default parameters for IRIS and Mosek.

<sup>1</sup><https://docs.mosek.com/latest/cxxfusion/examples-list.html#doc-example-file-lownerjohn-ellipsoid-cc>

We compare the computation time and average precision of each method to calculate the maximum ellipsoid in closed convex polytopes consisting of different numbers of halfspaces. The precision is defined as

$$\psi_{\mathcal{E}} = \mathcal{L}_{\psi} \left( \left[ [A_{\mathcal{P}} A_{\mathcal{E}}^* D_{\mathcal{E}}^*]^2 \mathbf{1} \right]^{\frac{1}{2}} + A_{\mathcal{P}} b_{\mathcal{E}}^* - b_{\mathcal{P}} \right), \quad (65)$$

where  $A_{\mathcal{E}}^*$ ,  $D_{\mathcal{E}}^*$  and  $b_{\mathcal{E}}^*$  are the coefficients of the maximum inscribed ellipsoid solved by each method,  $A_{\mathcal{P}}$  and  $b_{\mathcal{P}}$  are the parameters that define the halfspaces of the input polytope  $\mathcal{P}$  (10), and  $\mathcal{L}_{\psi}(\cdot)$  denotes a function that takes the absolute value of the largest element of the input vector. The corresponding performance are summarized in Fig. 6 and Tab. I. By comparing the results of Mosek SDP and Mosek SOCP, we can get this conclusion that transforming the commonly used SDP formulation to the SOCP formulation leads to a significant improvement in computational efficiency without sacrificing accuracy. Additionally, in the low-dimension massive-constraint case faced in this paper, the use of the affine scaling method avoids the requirement of solving a large-scale system of linear equations at each iteration, compared to the interior point method used in Mosek. Thus as shown the results demonstrate, CAS is capable of solving MVIE in less time while maintaining a comparable precision. Moreover, for the 2-D case, our proposed linear-time complexity algorithm RAN further enhances the computational efficiency by orders of magnitude and compute analytical results.

#### B. Comparison of Generating Convex Free Polytope

Based on the description and analysis of several state-of-the-art algorithms for convex free polytope generation in Sec. II, we benchmark the proposed algorithm FIRI with IRIS [1], Galaxy [3] and RILS [4]. Since Gao's method [2] relies on modeling the environment as a grid map and can only be barely real-time with low map resolution, it is not compared here. Additionally, Galaxy can be regarded as an enhancement of Savin's method [8], both of which are based on space inversion, hence we directly employ Galaxy. For IRIS and our proposed method FIRI, we set the same parameter  $\rho = 0.02$ , that is, the stopping condition of them is that the volume of the MVIE of the convex hull obtained in this iteration grows less than 2% from the last iteration. For Galaxy and RILS, we used the default parameters.

To begin with, we compare the capability of these methods to adapt to different types of seed and obstacle inputs. As the summarized results in Tab. II shows, FIRI demonstrates the

TABLE II  
COMPARISON OF ADAPTABILITY OF DIFFERENT METHODS FOR GENERATING CONVEX FREE POLYTOPE

Method	Seed Type			Obstacles Type	
	Point	Line	Polytope	Point	Polytope
<b>FIRI</b>	✓	✓	✓	✓	✓
IRIS [1]	✓	✗	✗	✓	✓
Galaxy [3]	✓	✗	✗	✓	✗
RILS [4]	✓	✓	✗	✓	✗

TABLE III  
SUCCESS RATE OF DIFFERENT METHODS FOR GENERATING CONVEX POLYTOPES CONTAINING SEED ACROSS DIFFERENT SCENARIOS AND SEED TYPES

Scenario		Success Rate [%]											
		Point Seed				Line Seed				Polytope Seed			
		<b>FIRI</b>	IRIS [1]	Galaxy [3]	RILS [4]	<b>FIRI</b>	IRIS [1]	Galaxy [3]	RILS [4]	<b>FIRI</b>	IRIS [1]	Galaxy [3]	RILS [4]
2-D	Sparse	<b>100</b>	98.4	100	100	<b>100</b>	74.7	81.6	100	<b>100</b>	88.9	95.8	97.0
	Medium	<b>100</b>	97.0	100	100	<b>100</b>	53.9	64.2	100	<b>100</b>	79.7	91.1	95.3
	Dense	<b>100</b>	96.6	100	100	<b>100</b>	48.5	39.1	100	<b>100</b>	69.7	73.9	90.6
3-D	Sparse	<b>100</b>	99.1	100	100	<b>100</b>	96.5	79.1	100	<b>100</b>	96.2	85.1	98.0
	Medium	<b>100</b>	97.8	100	100	<b>100</b>	78.2	61.3	100	<b>100</b>	70.6	78.0	88.3
	Dense	<b>100</b>	96.2	100	100	<b>100</b>	59.6	47.0	100	<b>100</b>	37.3	45.1	70.6

highest level of adaptability to various inputs. Subsequently, based on the reported adaptability, we conduct extensive benchmarks to validate the superior performance of FIRI in terms of managibility, efficiency, and quality.

1) *managibility*: When convex polytope is applied in robot navigation, managibility is crucial. For instance, there are situations where we require the convex hull to contain the line segment of the path generated by the frontend [47], or during whole body planning, we demand the convex hull to encompass the robot [48]. Thus we compare the managibility of various methods by using point, line, and convex polytope as the seed input. And the obstacles input are characterized by points.

We conduct benchmark in a complex environment of  $50 \times 50$   $m$  size (with a height of  $10m$  for the 3-D case), where we generate random obstacles by using Perlin noise [49]. For each test, we randomly generate a collision-free seed in the environment as input. The boundary of each convex hull generation algorithm is constrained to be a square with side length  $6$   $m$  centered on the seed's center and parallel to the coordinate axes. And the obstacle input is the points within the boundary of the square in the map. When the seed input is a convex polytope, we set it as a rectangular for the 3-D case, or a rectangle for the 2-D case. We generate the experimental environments with different obstacle densities.

To be fair, we perform several adjustments in different seed cases based on Tab. II. When the seed is a line, for Galaxy and IRIS, which can only use one point as seed input, we use both the endpoints and midpoints of line as seed inputs for them to compute 3 convex polyhedra. We then select the convex hull with the highest degree of line containment as the result. When the seed is a convex polytope, FIRI can directly take the polyhedra as input. IRIS and Galaxy take the each vertices and center point of the seed as inputs to compute multiple corresponding convex hulls. For RILS, which is adapted to use a line as input, we compute multiple convex hulls using the seed's edges and diagonals as inputs. For each of the above three methods, we choose the convex hull that maximizes the inclusion of the robot as the respective result.

We calculate the success rate of each method in generating convex polytope that completely contains the seed input, as shown in Tab. III. Due to greedy approach of IRIS in seeking the largest possible volume of convex hull, it performs poorly in terms of managibility, to the extent that it cannot guarantee

to contain the seed even when the seed is a point. While RILS can ensure to contains the line seed, it is not sufficient to guarantee that the generated convex hull contain the seed when the seed is represented as a convex hull. As for Galaxy, its heuristic cut method does not ensure that the seed is included either, when the seed is a line or a convex polytope. In contrast, benefiting from Restrictive Inflation, FIRI ensures managibility for all three types of seed inputs across various obstacle densities.

2) *Efficiency and Quality*: We construct experiments to compare the efficiency and quality in the aforementioned random environments. The obstacle input and boundary settings remain the same as in Sec. VII-B1. We record the computation time for each algorithm when seeded with a point as input. If IRIS generates a convex polytope during the iteration process that does not include the seed, we force IRIS to terminate prematurely and return the polytope of the previous iteration that includes the seed as the result. Since both RILS and FIRI guarantee managibility for line seeds, we document their respective time overhead when the seed input is a line as well. In addition, FIRI maintains its managibility when only a single iteration is performed, thus we also record the results of a single iteration of FIRI, denoted as **FIRI(SI)**.

To provide a fair and intuitive representation of the size of convex polytopes generated by each algorithm, we take FIRI, which aims to maximize the convex hull volume, as the baseline. Specifically, we report the ratio of the volume of the convex hull obtained by each algorithm to the volume of the one obtained by FIRI, as demonstrated in Fig. 7. The efficiency results are presented in Tab. IV, where, for clarity, we distinguish between iterative and non-iterative methods using dashed lines in the case of point seed. In addition, since the number of input obstacles is the primary factor influencing the computational efficiency, we showcase the the number of input obstacles in different density environments in Tab. V to provide readers with an insight into the computational efficiency of each algorithm. As the results illustrate, both IRIS and FIRI iteratively compute larger convex polytopes by continuously inflating the MVIE, resulting in similar sizes. However, in IRIS, the SDP-based MVIE solving method consumes significant computation time [1], whereas FIRI achieves significant efficiency improvements in MVIE calculations as shown in Sec. VII-A, leading to remarkably more computationally efficient compared to IRIS.



TABLE IV  
COMPARISON OF COMPUTATION TIME BETWEEN DIFFERENT METHODS FOR GENERATING CONVEX FREE POLYTOPE

Seed Type	Scenario	Method	Computation Time [ms]											
			Sparse				Medium				Dense			
			avg	std	min	max	avg	std	min	max	avg	std	min	max
Point	2-D	<b>FIRI</b>	<b>0.038</b>	<b>0.013</b>	<b>0.013</b>	<b>0.069</b>	<b>0.120</b>	<b>0.035</b>	<b>0.032</b>	<b>0.189</b>	<b>0.273</b>	<b>0.068</b>	<b>0.127</b>	<b>0.451</b>
		IRIS [1]	34.444	2.978	24.997	40.125	37.730	2.982	28.001	46.620	39.671	5.111	20.848	48.155
		Galaxy [3]	0.069	0.013	0.042	0.099	0.123	0.023	0.077	0.168	0.233	0.038	0.143	0.327
		RILS [4]	0.011	0.004	0.005	0.021	0.037	0.010	0.016	0.061	0.082	<b>0.017</b>	0.027	0.122
		FIRI(SI)	<b>0.008</b>	<b>0.003</b>	<b>0.002</b>	<b>0.017</b>	<b>0.032</b>	<b>0.010</b>	<b>0.014</b>	<b>0.052</b>	<b>0.066</b>	0.021	<b>0.016</b>	<b>0.103</b>
	3-D	<b>FIRI</b>	<b>0.143</b>	<b>0.035</b>	<b>0.069</b>	<b>0.237</b>	<b>0.660</b>	<b>0.224</b>	<b>0.259</b>	<b>1.259</b>	<b>2.116</b>	<b>0.560</b>	<b>1.109</b>	<b>3.535</b>
		IRIS [1]	34.638	7.083	16.966	76.988	55.724	15.327	11.173	103.176	87.897	20.823	14.826	167.441
		Galaxy [3]	0.144	0.054	0.054	0.328	1.337	0.467	0.445	2.476	5.916	1.079	4.321	9.185
		RILS [4]	0.044	0.017	0.020	0.113	0.346	0.165	0.120	0.658	1.649	0.431	0.920	2.817
		FIRI(SI)	<b>0.020</b>	<b>0.007</b>	<b>0.008</b>	<b>0.041</b>	<b>0.149</b>	<b>0.058</b>	<b>0.046</b>	<b>0.341</b>	<b>0.544</b>	<b>0.173</b>	<b>0.310</b>	<b>1.049</b>
Line	2-D	<b>FIRI</b>	0.038	0.014	0.010	0.068	0.120	0.039	0.047	0.204	0.263	0.086	0.040	0.440
		RILS [4]	0.014	0.004	0.007	0.028	0.041	0.012	0.016	0.073	0.085	0.028	<b>0.030</b>	0.142
		FIRI(SI)	<b>0.008</b>	<b>0.003</b>	<b>0.002</b>	<b>0.018</b>	<b>0.029</b>	<b>0.010</b>	<b>0.011</b>	<b>0.050</b>	<b>0.079</b>	<b>0.017</b>	0.045	<b>0.115</b>
	3-D	<b>FIRI</b>	0.163	0.037	0.090	0.239	0.678	0.208	0.272	1.110	2.977	0.930	1.709	4.651
		RILS [4]	0.061	0.020	0.027	0.114	0.417	0.134	0.148	0.721	2.069	0.532	1.099	3.214
		FIRI(SI)	<b>0.025</b>	<b>0.007</b>	<b>0.012</b>	<b>0.039</b>	<b>0.148</b>	<b>0.049</b>	<b>0.053</b>	<b>0.249</b>	<b>0.696</b>	<b>0.216</b>	<b>0.349</b>	<b>1.084</b>

TABLE V  
INPUT OBSTACLES NUMBER OF DIFFERENT SCENARIOS

Scenario		Input Obstacle Number			
		avg	std	min	max
2-D	Sparse	246.7	111.0	44	525
	Medium	1157.6	277.8	553	1683
	Dense	3007.5	515.2	1443	4048
3-D	Sparse	453.6	119.6	303	670
	Medium	2677.8	613.6	1524	3929
	Dense	12659.0	764.9	11032	14536

Moreover, FIRI achieves a computational time that is within three times the time of non-iterative RILS which even does not require MVIE computations. In contrast, the other three non-iterative methods yield smaller convex polytopes, but achieve better computational efficiency than IRIS in Tab. IV. Galaxy, however, is even less efficient than iterative FIRI due to the fact that the Quickhull [24] algorithm involved degrades in the scene to which Galaxy corresponds, which is particularly noticeable in the 3D case. As for FIRI(SI), similar to RILS, it directly updates the convex hull and can generate polytope of similar size to RILS, but with higher computational efficiency. Moreover, benefiting from the managibility brought about by the Restrictive Inflation, if the objective is to obtain a free convex polytope that encompasses the seed as fast as possible, without pursuing maximum volume, we believe that FIRI(SI) is a suitable choice.

In order to demonstrate the significance of convex hull quality and its impact on trajectory planning, we conduct an experiment on trajectory planning based on different convex hull generation methods in the random environment as shown in Fig. 8(a). We first generate a collision-free path connecting

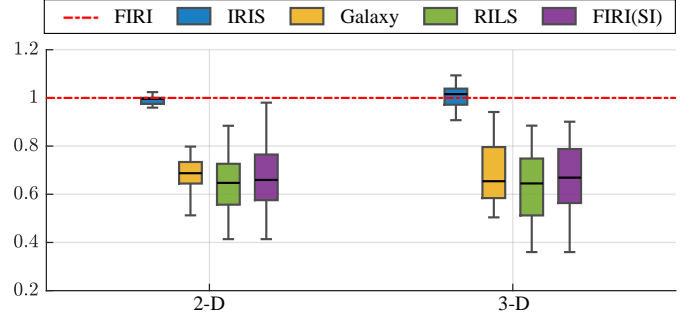
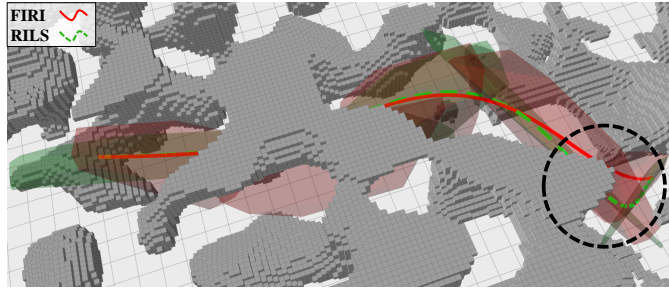


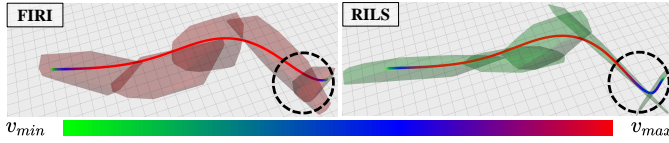
Fig. 7. Comparison of the sizes of the free convex polytopes generated by different methods. The dashed line with a value of 1 indicates the result obtained by IRIS, which we take as the baseline.

the start and end points using RRT\* [50]. The obtained path can be viewed as a set of connected line segments, based on which we generate convex hulls subsequently. As indicated in Tab. III, only FIRI and RILS exhibit managibility over line seed. Thus we build corridors for comparison based on these two methods, which correspond to iterative and non-iterative strategies of whether or not to pursue the maximum volume, respectively. The process of generating a safe flight corridor (SFC) is as follows: Sequentially traverse each line segment on the path. If the line segment is already contained within the previously generated polytopes, proceed to the next segment. Otherwise, we use this line as a seed to generate a new convex polytope. Due to managibility, the generated set of convex polytopes must have an intersection between two neighboring pairs, and this set of convex hull is the result SFC. Based on the generated SFC, we adopt GPOPS-II [51] to obtain the optimal trajectory constrained within the corridor. This collocation-based method transcribe the trajectory optimization problem into a constrained Nonlinear Programming via Gauss pseudospectral method, which is then solved by

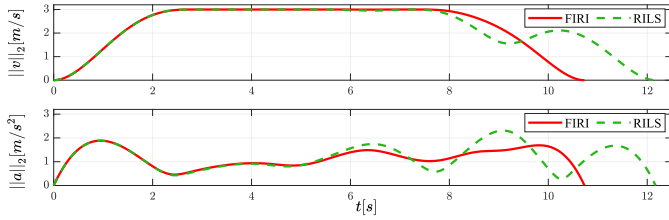




(a) The corridors generated by FIRI and RILS in the random forest and the time optimal trajectories constrained in each corridor.



(b) The speed profiles of the trajectories, colored by the speed magnitude.



(c) The velocity and acceleration magnitude for the trajectories constrained in the corridors generated by FIRI and RILS respectively.

Fig. 8. The comparison between the SFC established based on FIRI and RILS in a complex environment and the optimal trajectories constrained within the generated SFC.

the well-established NLP solver SNOPT [52]. In GPOPS-II, each trajectory phase is confined within one polytope, and we set the feasibility constraints for velocity and acceleration as  $3m/s$  and  $6m/s^2$ , and set the time weight as 20. As depicted in Fig. 8(b), RILS generates narrow convex polytopes in the area marked by the black dashed circles, resulting in limited maneuvering space for trajectory optimization. Consequently, the trajectory constrained in the SFC generated by RILS exhibits a conservative behavior in the marked area. In contrast, FIRI, due to its pursuit of maximizing convex hulls, is capable of generating larger corridors, providing greater spatial flexibility in trajectory planning.

## VIII. ACKNOWLEDGMENT

The linear-time complexity algorithm for 2-D MVIE is initially designed by Zhepei Wang and finalized by Qianhao Wang.

## REFERENCES

- [1] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Algorithmic Foundations of Robotics XI*. Springer, 2015, pp. 109–124.
- [2] F. Gao, L. Wang, B. Zhou, X. Zhou, J. Pan, and S. Shen, "Teach-Repeat-Replan: A complete and robust system for aggressive flight in complex environments," *IEEE Transactions on Robotics*, vol. 36, no. 5, pp. 1526–1545, 2020.
- [3] X. Zhong, Y. Wu, D. Wang, Q. Wang, C. Xu, and F. Gao, "Generating large convex polytopes directly on point clouds," *arXiv preprint arXiv:2010.08744*, 2020.
- [4] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments," *IEEE Robotics and Automation Letters*, pp. 1688–1695, 2017.
- [5] Z. Han, Z. Wang, N. Pan, Y. Lin, C. Xu, and F. Gao, "Fast-racing: An open-source strong baseline for se(3) planning in autonomous drone racing," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8631–8638, 2021.
- [6] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3259–3278, 2022.
- [7] A. Sarmientoy, R. Murrieta-Cidz, and S. Hutchinson, "A sample-based convex cover for rapidly finding an object in a 3-d environment," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE, 2005, pp. 3486–3491.
- [8] S. Savin, "An algorithm for generating convex obstacle-free regions based on stereographic projection," in *2017 International Siberian Conference on Control and Communications (SIBCON)*. IEEE, 2017, pp. 1–6.
- [9] R. Seidel, "Small-dimensional linear programming and convex hulls made easy," *Discrete & Computational Geometry*, vol. 6, no. 3, pp. 423–434, 1991.
- [10] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOases: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, pp. 327–363, 2014.
- [11] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "Osqp: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, pp. 1–36, 2020.
- [12] G. Frison and M. Diehl, "Hpipm: a high-performance quadratic programming framework for model predictive control," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6563–6569, 2020.
- [13] L. G. Khachiyan and M. J. Todd, "On the complexity of approximating the maximal inscribed ellipsoid for a polytope," *Mathematical Programming*, vol. 61, no. 1, pp. 137–159, 1993.
- [14] L. Lovász, *An algorithmic theory of numbers, graphs and convexity*. SIAM, 1986.
- [15] H. W. Lenstra Jr, "Integer programming with a fixed number of variables," *Mathematics of operations research*, vol. 8, no. 4, pp. 538–548, 1983.
- [16] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. SIAM, 2001.
- [17] Y. Zhang and L. Gao, "On numerical solution of the maximum volume ellipsoid problem," *SIAM Journal on Optimization*, vol. 14, no. 1, pp. 53–76, 2003.
- [18] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, 1994.
- [19] MOSEK Aps, "MOSEK Optimizer API for C," 2020. [Online]. Available: <https://www.mosek.com>
- [20] J. Matoušek, M. Sharir, and E. Welzl, "A subexponential bound for linear programming," in *Proceedings of the eighth annual symposium on Computational geometry*, 1992, pp. 1–8.
- [21] B. Gärtner and S. Schönherr, "Exact primitives for smallest enclosing ellipses," in *Proceedings of the thirteenth annual symposium on Computational geometry*, 1997, pp. 430–432.
- [22] M. Montanari, N. Petrinic, and E. Barbieri, "Improving the gjk algorithm for faster and more reliable distance queries between convex objects," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 3, pp. 1–17, 2017.
- [23] S. Katz, A. Tal, and R. Basri, "Direct visibility of point sets," in *ACM SIGGRAPH 2007 papers*, 2007, pp. 24–es.
- [24] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, pp. 469–483, 1996.
- [25] F. John, "Extremum problems with inequalities as subsidiary conditions," *Traces and emergence of nonlinear programming*, pp. 197–215, 2014.
- [26] S. P. Tarasov, "The method of inscribed ellipsoids," in *Soviet Mathematics-Doklady*, vol. 37, no. 1, 1988, pp. 226–230.
- [27] L. G. Khachiyan and M. J. Todd, "On the complexity of approximating the maximal inscribed ellipsoid for a polytope," Cornell University Operations Research and Industrial Engineering, Tech. Rep., 1990.
- [28] K. M. Anstreicher, "Improved complexity for maximum volume inscribed ellipsoids," *SIAM Journal on Optimization*, vol. 13, no. 2, pp. 309–320, 2002.

- [29] A. Nemirovski, "On self-concordant convex-concave functions," *Optimization Methods and Software*, vol. 11, no. 1-4, pp. 303–384, 1999.
- [30] C.-H. Lin, R. Wu, W.-K. Ma, C.-Y. Chi, and Y. Wang, "Maximum volume inscribed ellipsoid: A new simplex-structured matrix factorization framework via facet enumeration and convex optimization," *SIAM Journal on Imaging Sciences*, vol. 11, no. 2, pp. 1651–1679, 2018.
- [31] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.
- [32] E. Welzl, "Smallest enclosing disks (balls and ellipsoids)," in *New Results and New Trends in Computer Science: Graz, Austria, June 20–21, 1991 Proceedings*. Springer, 2005, pp. 359–370.
- [33] C. D. Toth, J. O'Rourke, and J. E. Goodman, *Handbook of Discrete and Computational Geometry*. CRC Press, 2017.
- [34] J.-S. Chang and C.-K. Yap, "A polynomial solution for the potato-peeling problem," *Discrete & Computational Geometry*, vol. 1, no. 2, pp. 155–182, 1986.
- [35] M. Agarwal, J. Clifford, and M. Lachance, "Duality and inscribed ellipses," *Computational Methods and Function Theory*, vol. 15, pp. 635–644, 2015.
- [36] G. H. Golub and F. V. Loan, *Matrix Computations*. The Johns Hopkins University Press, 2013.
- [37] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 2012.
- [38] J. Lagarias and R. Vanderbei, "Ti dikin's convergence result for the affine scaling algorithm," *Contemp. Math*, vol. 114, p. 109, 1990.
- [39] T. Tsuchiya and R. D. C. Monteiro, "Superlinear convergence of the affine scaling algorithm," *Mathematical Programming*, vol. 75, no. 1, pp. 77–110, 1996.
- [40] M. Sharir and E. Welzl, "A combinatorial bound for linear programming and related problems," in *STACS 92: 9th Annual Symposium on Theoretical Aspects of Computer Science Cachan, France, February 13–15, 1992 Proceedings 9*. Springer, 1992, pp. 567–579.
- [41] D. Minda and S. Phelps, "Triangles, ellipses, and cubic polynomials," *The American Mathematical Monthly*, vol. 115, no. 8, pp. 679–689, 2008.
- [42] A. Horwitz, "Ellipses of maximal area and of minimal eccentricity inscribed in a convex quadrilateral," *Australian Journal of Mathematical Analysis and Applications*, vol. 2, no. 1, p. 12, 2005.
- [43] A. C. Jones, *An introduction to algebraical geometry*. Clarendon Press, 1912.
- [44] J. Richter-Gebert and J. Richter-Gebert, "Conics and their duals," *Perspectives on projective geometry: A guided tour through real and complex geometry*, pp. 145–166, 2011.
- [45] F. Klein, C. A. T. Noble, and E. R. T. Hedrick, *Elementary Mathematics from an Advanced Standpoint-Geometry: Transl. from the Third German Ed. by ER Hedrick and CA Noble*. Dover, 1939.
- [46] M. J. D. Hayes, Z. A. Copeland, P. J. Zsombor-Murray, and A. Gfrerrer, "Largest area ellipse inscribing an arbitrary convex quadrangle," in *Advances in Mechanism and Machine Science: Proceedings of the 15th IFToMM World Congress on Mechanism and Machine Science 15*. Springer, 2019, pp. 239–248.
- [47] J. Ji, N. Pan, C. Xu, and F. Gao, "Elastic tracker: A spatio-temporal trajectory planner for flexible aerial tracking," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 47–53.
- [48] Z. Han, Y. Wu, T. Li, L. Zhang, L. Pei, L. Xu, C. Li, C. Ma, C. Xu, S. Shen *et al.*, "An efficient spatial-temporal trajectory planner for autonomous vehicles in unstructured environments," *IEEE Transactions on Intelligent Transportation Systems*, 2023.
- [49] J. C. Hart, "Perlin noise pixel shaders," in *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware*, 2001, pp. 87–94.
- [50] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.
- [51] M. A. Patterson and A. V. Rao, "Gpops-ii: A matlab software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming," *ACM Transactions on Mathematical Software (TOMS)*, vol. 41, no. 1, pp. 1–37, 2014.
- [52] P. E. Gill, W. Murray, and M. A. Saunders, "Snopt: An sqp algorithm for large-scale constrained optimization," *SIAM review*, vol. 47, no. 1, pp. 99–131, 2005.