Fast Iterative Region Inflation for Computing Large 2D/3D Convex Regions of Obstacle-Free Space [1]

*Wang et al. 2025; Reimplementation by Varun Raghavendra, MS Robotics Student at Northeastern University.

https://github.com/varuncraghavendra/Fast-Iterative-Region-Inflation-

***Abstract*—Scalable autonomous robots and self driving cars are finally a reality and this has only been possible due to the intense research and development of various algorithms for major autonomy problems like safe motion planning, collision avoidance, sensor data filtering, localization and mapping, learning dynamics through reinforcement learning, and the list goes on. For example, in systems like self-driving cars where real cars carry real people on real roads and have to drive autonomously, safety becomes the biggest challenge and no amount of redundancy in safety is enough to ensure safe autonomy. Hence, researchers across the world have contributed to various geometrical and dynamical techniques to solve the problem of safe driving of robots in challenging environments in the absence of a global positioning system. One such algorithm is the Fast Iterative Region Inflation (FIRI in short) proposed by Wang et al. and his fellow researchers at Zhejiang University, where they propose a novel formulation for solving high quality and computationally efficient obstacle-free convex regions for cluttered 2D and 3D environments. This becomes crucial in efficiently running obstacle avoidance and whole body planning on the robot in cluttered environments like forests, or pushing the boundaries of autonomous racing cars where high speed and precision becomes crucial in pushing the boundaries of high speed navigation in a safety reinforced manner. In this project, I propose the 2D Implementation of FIRI algorithm with essential modules implemented in MATLAB, using a new optimization solver and benchmark against the work when compared to the proposed solver and results in the original paper.**

I. Introduction

The obstacle space in the real world is highly dynamic and non-convex. Identifying local regions of any environment and characterizing the obstacle-free space is a real challenge. To parametrize these obstacles in algorithms for evading obstacles, perception plays a very important role in capturing a rich map of the environment and different objects within them. Using this data, several techniques can be used to categorize regions as free space, obstacle space and the rest is assumed to be unknown space. For the robot to approach these spaces carefully, it is advised to apply different post processing methods on the image or point cloud data (PCD) collected from the perception sensors, of which occupancy grid maps (OGM's) have been used successfully in real time segmentation of free, convex and unknown regions and help in progressing the navigation. Furthermore, for efficient whole body planning and trajectory planning, geometric methods provide assurance that certain non-holonomic bounds are not crossed at any instant of time and enforcing convex hull bounds on the obstacle free regions can give a good closure to solving configuration spaces on the go. But knowing the discrete point cloud or OGM values is not enough to make sense of the local information and the fine geometries of different types of static and dynamic obstacles in the environment, and this is where some assumptions can be handy to make more sense of the map data. One of the earliest works in making such assumptions and solving efficiently was the (Iterative Regional Inflation by Semidefinite programming (IRIS) where Deits et al. [2] proposes a new method for quickly computing large polytopic and ellipsoidal regions of obstacle-free space through a series of convex optimizations. The assumptions made in this work has been a benchmark in the research community for solving the problem of obstacle avoidance in robotics. The main assumption in this work is that the configuration space consists of a bounded region in Rn which contains polyhedral obstacles. This makes the whole c-space defined as a union of discrete convex regions (we can group the filled voxels in the OGM and a union of these convex regions can be defined as the obstacle regions and free spaces) and the whole problem can be defined in a solvable geometric way. For a 2D environment, we can think of the free space as a polygon with polygonal holes. More importantly, we can construct convex linear safety constraints from non convex obstacles using these convex polytopes, which helps in problem formulation and obtaining solutions efficiently and with assurance. The setup of the convex decomposition problem in [2] is as follows :

Around an initial seed point (signifying the robot's initial position or the beginning of a front end path for the robot), we construct convex polytopes representing obstacle-free subsets of configuration space by iteratively generating separating hyperplanes that exclude non-convex obstacles. Compute a Maximum Volume Inscribed Ellipsoid (MVIE) within the current convex polytope to guide its expansion. Larger convex polytopes provide more flexibility for trajectory optimization, potentially leading to smoother trajectories.

To validate the algorithm, we can use some critical metrics and we discuss them below in brief :

Quality of the convex polytopes in each iteration is a question when it comes to how much computational resources are required to solve the volumes of these polytopes, and if we are compromising on the resources, then we need to settle for a low quality output. Manageability is hard to maintain, meaning that the robot partly resided inside these convex hulls, which could be a real issue in whole body planning, and hence complete safety cannot be assured. Computational efficiency plays a big role in running such algorithms in robots on the go! Hence it is important to use as few resources as possible and solve for the convex regions within a considerable amount of time. The faster the solution, the more freedom the robot gets to navigate autonomously. I chose to review and implement the work in [1] because it provides compelling evidence that the key metrics quality, manageability, and efficiency have been thoughtfully addressed and optimized algorithmically, yielding results that surpass many of the aforementioned research efforts in this field. The primary goal of the discussed approach is to find safe and continuous/overlapping convex hulls in C-space for vehicles with non-holonomic constraints. In [1], the authors discuss probably the best known method today to achieve the intended goals. To achieve high quality, FIRI ensures monotonic improvement in the volume of the convex region, which means the algorithm guarantees that each iteration returns an equal or bigger volume than previous iterations. FIRI, through the Restrictive Inflation (RsI) algorithm ensures that the halfspaces exclude obstacles and include the seed in each iteration, thereby enhancing the manageability bounds. For achieving computational efficiency and real time estimates of these free spaces, FIRI proposes the reformulation of the Maximum Volume Inscribed Ellipsoid (MVIE) algorithm as an optimization problem using first linear-complexity analytical algorithm for 2D environments and second order conic problem (SOCP) for 3D environments, which proves to be much faster and efficient than the semidefinite programming (SDP) proposed in the prior work. The reason for this is that SDP defines convex polytopes via linear matrix inequalities (LMI), and solving these LMI's scales poorly as the number of dimensions increases. But we should take into consideration the fact that although the SDP solver is computationally expensive, it guarantees convexity, whereas the SOCP solver might not capture strict convexity like SDP, but does bring in the balance between high-quality output and efficiency. Overall, the FIRI algorithm proposes an iterative approach to solve for the two main components, the Restrictive Inflation (RsI) polytope and Maximum Volume Inscribed Ellipsoid (MVIE)

II. Overview of 2D FIRI Algorithm

The **Fast Iterative Region Inflation (FIRI)** algorithm is based on convex optimization and geometric duality principles, designed to iteratively construct high-quality obstacle-free convex regions while ensuring strict containment of the seed set. The algorithm is governed by two alternating modules: **Restrictive Inflation (RsI)** and **Maximum Volume Inscribed Ellipsoid (MVIE)**, which are both formulated as a solvable convex optimization problem. The RsI module begins by transforming the seed $\mathcal{Q}$ and obstacles $\mathcal{O}_i$ into a

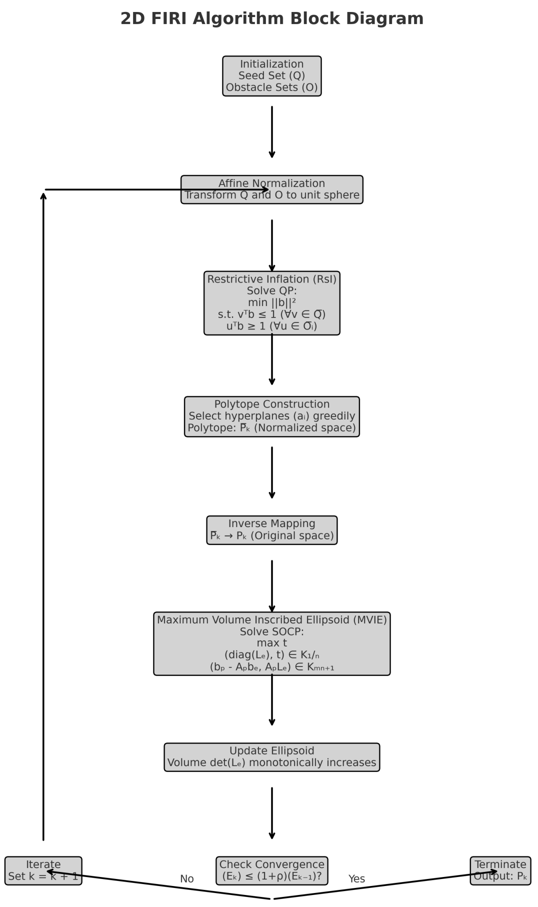

Fig. 1. Block Diagram of the 2D FIRI Algorithm illustrating iterative steps of Restrictive Inflation (RsI) and Maximum Volume Inscribed Ellipsoid (MVIE) computations.

normalized coordinate space via an affine mapping derived from the current ellipsoid parameters $(A_\mathcal{E}, D_\mathcal{E}, b_\mathcal{E})$, converting the ellipsoid into a unit sphere. Within this space, RsI solves a convex quadratic program (QP) for each obstacle to compute a hyperplane a_i that maximizes the margin between $\mathcal{Q}$ and $\mathcal{O}_i$. The non-convex original problem—maximizing $a_i^T a_i$ under quadratic constraints is reformulated into a convex minimum-norm problem via polar duality ($a_i = b/\|b\|^2$), ensuring tractability through linear constraints. The resulting hyperplanes are greedily selected to construct a polytope $\bar{\mathcal{P}}_k$ in normalized space, which is inversely mapped to the original space. The MVIE module computes the largest ellipsoid inscribed within $\mathcal{P}_k$ by solving a second-order cone program (SOCP), where L and d parameterize the ellipsoid. This avoids semidefinite programming (SDP) while preserving convexity. The monotonic improvement of the ellipsoid volume $\det(L)$ guarantees convergence, as each iteration expands $\mathcal{P}_k$ while maintaining $\mathcal{Q} \subseteq \mathcal{P}_k$. Theoretical guarantees arise from the convexity of the reformulated QP/SOCP, affine invariance of the normalization step, and geometric duality between hyperplanes and norm minimization. By decoupling obstacle exclusion (via RsI's linear constraints) and volume maximization (via MVIE's conic optimization), FIRI balances computational efficiency and geometric optimality, supporting extensions to non-Euclidean manifolds. FIRI provides theoretical guarantees through four key properties. *Feasibility* is ensured through quadratic programming (QP) constraints that strictly enforce seed containment ($\mathcal{Q} \subseteq \mathcal{P}_k$) while excluding obstacles. *Monotonicity* is guaranteed by the maximum volume inscribed ellipsoid (MVIE) serving as a lower bound, where $(\mathcal{E}_k) \geq (\mathcal{E}_{k-1})$ for all iterations k. This monotonic improvement in the ellipsoid volume induces *convergence*, as the bounded sequence $\{(\mathcal{E}_k)\}$ approaches a fixed point when $(\mathcal{E}_k) \leq (1+\rho)(\mathcal{E}_{k-1})$.

A. Problem Formulation

The biggest assumption is that the obstacles are convex (but are actually in a topological space) and that the union of these convex obstacles makes up the obstacle containing environment $\mathcal{O}_i$. Given a convex seed $\mathcal{Q} = \{v_1, \dots, v_s\} \subset^n$ ($n \in \{2,3\}$) and obstacle region $\mathcal{O} = \bigcup_{i=1}^{N} \mathcal{O}_i$ where $\mathcal{O}_i = \{u_{i,1}, \dots, u_{i,s_i}\}$, FIRI solves:

$$\max_{\mathcal{P}}(\mathcal{P}) \quad \text{s.t.} \quad \mathcal{Q} \subseteq \mathcal{P}, \quad \mathcal{O} \cap \text{int}(\mathcal{P}) = \emptyset \tag{1}$$

where $\mathcal{P}$ is a convex polytope. This problem is approached through iterative lower-bound optimization using maximum volume inscribed ellipsoid (MVIE).

The mathematical formulation of the algorithm relies heavily on two alternating optimization procedures: Restrictive Inflation (RsI) and Maximum Volume Inscribed Ellipsoid (MVIE), is explained in detail in the upcoming sections. These two optimization steps complement each other, facilitating iterative improvements until convergence is achieved.

B. Restrictive Inflation (RsI)

The restrictive halfspace computation in RsI solves the non-convex optimization (QCQP):

$$\max_{a_i \in \mathbb{R}^n} \quad a_i^\top a_i \tag{2}$$
$$\text{s.t.} \quad v^\top a_i \leq a_i^\top a_i \quad \forall v \in \bar{\mathcal{Q}} \tag{3}$$
$$u^\top a_i \geq a_i^\top a_i \quad \forall u \in \bar{\mathcal{O}}_i \tag{4}$$

where $\bar{\mathcal{Q}}$ and $\bar{\mathcal{O}}_i$ represent the normalized seed and obstacles. While Equation (2) maximizes the inflated ball radius $\|a_i\|^2$, the quadratic constraints create non-convexity. However, the condition $a_i^\top a_i > 0$ (guaranteeing origin containment in $H(a_i)$) enables convex reformulation through *polar duality*.

Convex Reformulation : Substituting $a_i = \frac{b}{b^\top b}$ into Equations (2)-(4) yields:

$$\min_{b \in \mathbb{R}^n} \|b\|^2 \quad \text{s.t.} \quad \begin{cases} v^\top b \leq 1 & \forall v \in \bar{\mathcal{Q}} \\ u^\top b \geq 1 & \forall u \in \bar{\mathcal{O}}_i \end{cases} \tag{5}$$

Derivation: 1. Substitute $a_i = \frac{b}{\|b\|^2}$ into (3):

$$v^\top \frac{b}{\|b\|^2} \leq \frac{\|b\|^2}{\|b\|^4} \implies v^\top b \leq 1$$

2. Similarly, (4) becomes:

$$u^\top \frac{b}{\|b\|^2} \geq \frac{\|b\|^2}{\|b\|^4} \implies u^\top b \geq 1$$

3. The objective $\max \|a_i\|^2 = \max \frac{1}{\|b\|^2}$ becomes $\min \|b\|^2$.

Algorithmic Advantages : Equation (5) exhibits these key properties:

- **Convexity**: Strictly convex quadratic objective with linear constraints
- **Unified Constraints**: Both seed and obstacle constraints become linear inequalities

This reformulation enables efficient solution via quadratic programming (QP), contrasting with prior IRIS methods that lack explicit seed-containment constraints. While IRIS handles only obstacle exclusion through:

$$\max_{a_i} \|a_i\| \quad \text{s.t.} \quad u^\top a_i \geq 1 \quad \text{(obstacles only)},$$

our formulation adds $v^\top b \leq 1$ to enforce $\mathcal{Q} \subseteq \mathcal{P}_k$ without compromising convexity.

The polar duality substitution maps:

- Hyperplane $H(a_i) = \{x \mid a_i^\top x \leq \|a_i\|^2\}$ to $H(b) = \{x \mid b^\top x \leq 1\}$
- Inflation magnitude $\|a_i\|$ to $\|b\|^{-1}$
- Origin containment $0 \in H(a_i)$ to $0 \in H(b)$

This preserves separation properties while converting the non-convex spherical inflation problem into a convex norm minimization in dual space.

C. Maximum Volume Inscribed Ellipsoid (MVIE)

Initially, the MVIE problem, which is inherently a determinant maximization problem constrained by semi-infinite inequalities, poses computational challenges due to orthogonality constraints on the ellipsoid shape matrix A_E. Specifically, the objective is to maximize the volume of the ellipsoid, mathematically equivalent to maximizing the determinant of the diagonal scaling matrix D_E. Formally, the MVIE problem is expressed as:

$$\max_{A_E, D_E, b_E} \quad \det(D_E), \tag{6a}$$
$$\text{s.t.} \quad \left[[A_P A_E D_E]^2 \mathbf{1} \right]^{\frac{1}{2}} \leq b_P - A_P b_E, \tag{6b}$$
$$D_E = \text{diag}(d_E),\ d_E \in \mathbb{R}^n_{\geq 0}, \tag{6c}$$
$$A_E^T A_E = I_n. \tag{6d}$$

The constraints (6b) represent a second-order cone (SOC) condition enforcing that the ellipsoid remains fully contained within the polytope $\mathcal{P}$. However, the orthogonality constraint (6d) significantly complicates the optimization, rendering it computationally intensive and numerically challenging if addressed directly through semidefinite programming (SDP). To overcome these challenges, we can reformulate the MVIE problem into a Second-Order Conic Programming (SOCP) format, and eliminating orthogonality constraints. We observe that the matrix product $A_E D_E^2 A_E^T$ is always positive definite at the optimum for non-degenerate instances, enabling a unique Cholesky decomposition $A_E D_E^2 A_E^T = L_E L_E^T$, with L_E being a lower triangular matrix with strictly positive diagonal elements. By introducing L_E as a new decision variable, the optimization transforms, and the orthogonality constraint can thus be entirely discarded. The determinant objective simplifies accordingly, as:

$$\det(D_E) = \sqrt{\det(A_E D_E^2 A_E^T)} = \det(L_E), \tag{7}$$

allowing direct optimization over the lower triangular Cholesky factor L_E. Furthermore, the original SOC constraints can be restated in terms of L_E, simplifying computational implementation. To linearize the determinant objective effectively, we can leverage the hypograph formulation of the geometric mean. Specifically, introducing the hypograph cone $K_{1/n}$, we maximize a scalar variable t subject to the constraint:

$$(\mathrm{diag}(L_E), t) \in K_{1/n} = \left\{ (x,t) \in \mathbb{R}^n_{\geq 0} \times \mathbb{R} \;\middle|\; (x_1 x_2 \dots x_n)^{\frac{1}{n}} \geq \tag{8}\right.$$

Additionally, the second-order constraints are captured concisely via the Cartesian product of standard second-order cones, denoted as K_n^m. Thus, the final SOCP reformulation of the MVIE problem becomes:

$$\max_{t, L_E, b_E} \quad t, \tag{9a}$$

$$\text{s.t.} \quad (\mathrm{diag}(L_E), t) \in K_{1/n}, \tag{9b}$$

$$(b_P - A_P b_E, A_P L_E) \in K_{n+1}^m, \tag{9c}$$

$$L_E \text{ lower triangular.} \tag{9d}$$

This formulation, expressing the MVIE optimization purely in terms of SOC constraints coupled with a linear objective function, significantly enhances computational efficiency.

III. My Implementation of the 2D FIRI Algorithm using MATLAB

A. Seed Initialization and Obstacle Generation

The initialization phase defines the seed region and randomly generates convex obstacles within a defined environment, ensuring appropriate spacing and clearances. The environment is set as a bounded 2D workspace of dimensions $50\,m \times 50\,m$. The seed is initialized as a rectangular convex polytope Q, specified by a center point $\mathbf{d}_E \in \mathbb{R}^2$, dimensions $w = 1.5\,m$ and $h = 0.75\,m$, and a rotation angle $\theta = \pi/6$. This seed is represented mathematically by an affine transformation applied to local coordinates:

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}, \tag{10}$$

The initial ellipsoid $\mathcal{E}_0$ associated with this seed is parameterized by a linear transformation matrix L_E and center $\mathbf{d}_E$:

$$L_E = R \cdot \mathrm{diag}\left(\frac{w}{2}, \frac{h}{2}\right), \quad \mathbf{d}_E = \begin{bmatrix} 25 \\ 25 \end{bmatrix}. \tag{11}$$

To simulate realistic scenarios, random convex polygonal obstacles $\mathcal{O}_i$ $(i = 1, \dots, 15)$ are generated within a bounding box surrounding the seed.

Obstacles are generated ensuring strict conditions:

1) **Minimum Spacing:** Obstacles maintain a minimum inter-obstacle distance $(0.8\,m)$, preventing overlaps.
2) **Minimum Clearance:** Each obstacle maintains at least $2.0\,m$ distance from both the seed vertices and the ellipsoid boundary.
3) **Convexity:** Generated obstacles are ensured convex using a convex hull calculation applied to randomly displaced vertices.

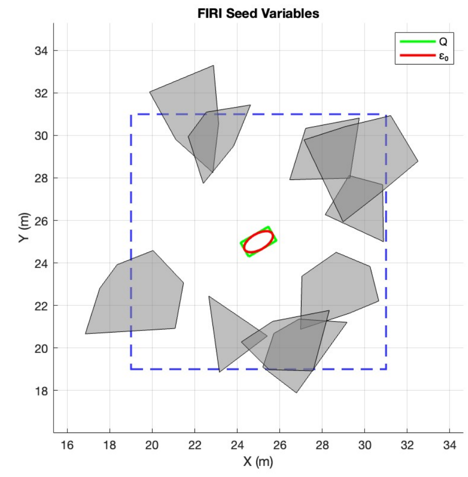

Fig. 2. Snapshot of the 50x50m (zoomed in) seed environment plotted on MATLAB. The red geometry represents the seed ellipse and the green rectangle denotes the seed polytope (ideally a little bigger than the dimension of the robot where it is assumed to originate in the environment). The blue dotted box represents that the volume growth happens within a 12x12m box, whereas in Section VII A in [1] mentions 6m box/cube.

Collision checking employs pairwise distance computations between obstacle vertices and the seed boundary points as well as points sampled along the ellipsoid boundary. The resultant configuration ensures a valid and robust initial condition for subsequent region inflation algorithms, guaranteeing a feasible and collision-free initial solution space.

B. Implementation of Restrictive Inflation (RsI) Algorithm

The Restrictive Inflation (RsI) algorithm constructs a convex polytope ensuring seed containment and obstacle exclusion. This implementation leverages CasADi's optimization framework to efficiently solve the quadratic program (QP).

Initially, the seed vertices $Q \subset \mathbb{R}^2$ are transformed into a normalized coordinate space defined by the ellipsoid parameters (L_E, d_E). Specifically, each vertex $v \in Q$ is mapped using the affine transformation:

$$\bar{Q} = L_E^{-1}(Q - d_E). \tag{12}$$

Similarly, for each obstacle $O_i \subset \mathbb{R}^2$, we compute its normalized coordinates:

$$\bar{O}_i = L_E^{-1}(O_i - d_E). \tag{13}$$

Within this normalized coordinate space, RsI solves the following convex optimization problem for each obstacle set $\bar{O}_i$:

$$\begin{aligned} \text{minimize} \quad & b^\top b && \text{(14a)} \\ \text{subject to} \quad & v^\top b \le 1, \quad \forall v \in \bar{Q}, && \text{(14b)} \\ & u^\top b \ge 1, \quad \forall u \in \bar{O}_i. && \text{(14c)} \end{aligned}$$

The optimization problem defined in (14) is solved using CasADi's high-level 'Opti' framework, designed for efficient numerical optimization. The procedure is as follows:

Linear constraints enforce seed containment,

$$v^\top b_{\text{var}} \le 1 \quad \forall v \in \bar{Q},$$

and obstacle exclusion,

$$u^\top b_{\text{var}} \ge 1 \quad \forall u \in \bar{O}_i.$$

The normalized hyperplane parameters follow as $a_i = \frac{b_{\text{opt}}}{\|b_{\text{opt}}\|^2}$ and $b_i = \frac{1}{\|b_{\text{opt}}\|^2}$. Finally, to map back to the original space one computes

$$a_i^{\text{original}} = L_E^{-\top} a_i, \qquad b_i^{\text{original}} = b_i + (a_i^{\text{original}})^\top d_E,$$

yielding the final polytope P that satisfies $Q \subseteq P$ and $O \cap P = \emptyset$.

C. Fallback Strategy

If the optimization fails (e.g., due to numerical issues or infeasibility), the algorithm defaults to conservative halfspaces defined by identity directions, ensuring minimal containment of the seed. This ensures robustness in the event of optimization failure and expressed as follows :

$$A_{\text{list}} = eye(2), \quad b_{\text{list}} = [1; 1] + A_{\text{list}}^\top d_E \qquad (15)$$

D. Maximum Volume Inscribed Ellipsoid (MVIE)

The Maximum Volume Inscribed Ellipsoid (MVIE) module in the FIRI algorithm is responsible for computing the largest-volume ellipsoid fully contained within a given convex polytope. The polytope is defined by a set of halfspace constraints $\mathcal{P} = \{x \in \mathbb{R}^2 \mid Ax \le b\}$. The ellipsoid is parametrized by a center $b_E \in \mathbb{R}^2$ and a lower-triangular Cholesky factor matrix $L_E \in \mathbb{R}^{2\times 2}$ such that the ellipsoid is expressed as:

$$\mathcal{E} = \left\{ x \in \mathbb{R}^2 \mid \left\| L_E^{-1}(x - b_E) \right\|_2 \le 1 \right\}. \qquad (16)$$

The optimization objective is to maximize the volume of this ellipsoid. Since the volume is proportional to $\det(L_E)$, the problem is reformulated to maximize a scalar variable t representing a lower bound on the geometric mean of the diagonal entries of L_E. The matrix L_E is constrained to be lower triangular with strictly positive diagonal entries to ensure a valid Cholesky decomposition.

The optimization problem is solved using CasADi's `Opti` framework, which allows symbolic expression of the decision variables and constraints. The decision variables in the problem include:

- L_E, the 2×2 lower-triangular matrix with entries L_{11}, L_{21}, and L_{22};
- b_E, the center of the ellipsoid;
- t, a scalar representing the hypograph constraint for geometric mean maximization.

The objective function is defined as the maximization of t, which is implemented equivalently as the minimization of $-t$. To enforce the lower-triangular structure, the upper triangular entry of L_E is fixed to zero, and its diagonal entries are constrained to be strictly positive.

The geometric mean condition is enforced by the inequality $L_{11}L_{22} \ge t^2$, which corresponds to the hypograph constraint for the second-order cone representation of the geometric mean.

For each halfspace constraint $A_i^\top x \le b_i$ in the polytope, a second-order cone (SOC) constraint is added to ensure that the ellipsoid remains within the polytope. These are formulated as:

$$\left\| A_i^\top L_E \right\|_2 \le b_i - A_i^\top b_E, \quad \forall i = 1, \dots, m, \qquad (17)$$

where m is the number of halfspace constraints. This ensures that every boundary of the ellipsoid lies strictly within the polytope.

To improve solver convergence, initial values for all variables are provided. The initial L_E is taken as the lower-triangular part of a given matrix, and b_E is initialized with a provided center point. The scalar t is initialized as the square root of the product of the diagonal elements of L_E.

Finally, the problem is solved using the IPOPT solver, configured for silent execution to minimize solver output. Upon successful solution, the values of L_E and b_E are extracted to represent the optimal ellipsoid. If the solver fails, the algorithm gracefully returns the initial values as a fallback. This MVIE module guarantees monotonic expansion of the ellipsoid volume and ensures convergence of the FIRI algorithm while avoiding the use of semidefinite programming (SDP).

IV. Key Outcomes and Learnings

Based on the computational time tabulation in Table IV given in [1], it is safe to say that FIRI is the fastest convex decomposition algorithm in literature today, and it also introduces a robust algorithm for ensuring seed containment in all iterations and proposes the efficient reformulation of the MVIE computation via SOCP, which enables any robot developer to adopt to a high speed planner algorithm, which can run onboard. One such efficient usage of FIRI is cited in [6], where the authors publish groundbreaking results for high speed navigation of an autonomous UAV system, breaching speeds of 20 m/s with full onboard processing in a cluttered GNSS denied forest environment, demonstrating how efficient and powerful this algorithm can be in the coming days. Since I worked only on 2D version of this algorithm, I would like to highlight some of the key outcomes and learnings I gained out of this work.

At first, I read and understood the paper [2] to see what problems were solved and what additional features were added

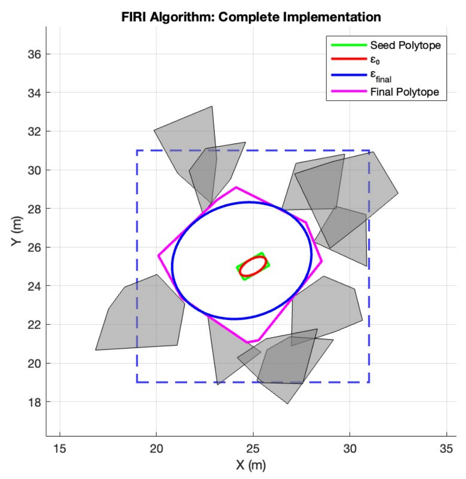

Fig. 3. Final Output of the 2D FIRI Algorithm. The blue geometry represents the final inflated ellipse in 2D space, whereas the pink region represents the largest free convex polytope available for the robot to freely navigate in the configuration space.

in [1], and tried to run the python implementation of IRIS-2D Algorithm [7] to understand the workflow better. In the first few runs itself, I noticed some loopholes in IRIS algorithm. The first loophole I observed was that of seed containment, which was missing in a few iterations of the algorithm itself, which was alarming. The region either grew outside the seed point or it encapsulated the seed in the first few iterations of free space growth, but the seed fell out gradually of the region. With this observation, I realized how important it was for the FIRI algorithm to propose the seed containment as a constraint in each iteration of the algorithm, and through this I learnt how whole body planning works in robotics.

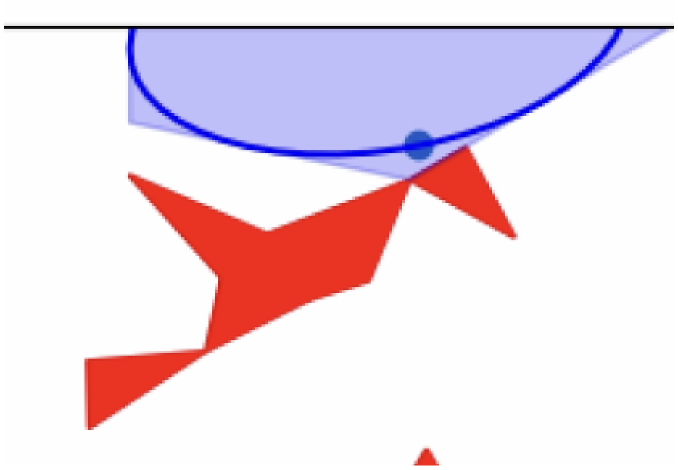

Fig. 4. Snapshot of output of IRIS algorithm from [7]. Here, it is clearly visible that seed containment (blue dot) is absent in the first few iterations, hence the hard seed containment constraints of FIRI algorithm is preferred.

I started questioning about the polytopical region and the choice of the ellipsoid as the suitable geometry for growth, and read more about how such geometries support the riemannian interfaces of robots with obstacles as the dimensions grow. Furthermore, I also realized how convex hulls are important in motion planning using spline based methods which strictly operate in continuous convex hulls. The best outcome out of this project for me was the adoption of Casadi solver, to solve conic problems with an NLP inspired SOCP approach, instead of directly using conic solvers like ECOS.

A. *Limitations*

1) Section III of [1] mentions that the algorithm is tested only in 2 and 3 dimensions, raising questions about its scalability to higher dimensions.
2) The ellipsoid volume-growth parameter ρ controls the trade-off between approximation tightness and runtime: a smaller ρ yields a tighter approximation (more iterations), while a larger ρ speeds up runtime at the potential cost of region quality. It is unclear how a single linear scaling parameter can govern polytope growth in higher dimensions.
3) Even with the greedy halfspace selection algorithm, very cluttered scenes could still produce hundreds of halfspaces, which could be computationally and geometrically complex.

B. *Future Scope of Work*

1) I still wish to look at the remainder implementations proposed in the paper such as SDMN solver for RsI, 2D Analytical algorithm for MVIE computation, 3D implementation of the FIRI algorithm, trajectory planning within the convex hulls obtained from the FIRI algorithm, whole body planning for a 2D wheeled robot and a 3D drone.
2) It would be great to use an Embedded Cone Solver instead of the Casadi approach to obtain better computational results.
3) I would like to extact more numerical results for the algorithm in the coming days.
4) Benchmark the results with the prior work results of [2] - [5] as given in the paper, for more number of obstacles in the environment to verify the scalability of the algorithm.

V. Acknowledgement

I would like to thank Prof. Michael Everett for providing the opportunity to conduct this project, clarifying some of my doubts during the project and being highly considerate about my requests to modify the benchmarking criteria for the FIRI algorithm. I would also like to thank Mr. Adarsh Salagame for timely clarification of some of my doubts with regards to the FIRI paper.

References

[1] Q. Wang, Z. Wang, C. Xu, and F. Gao, "Fast iterative region inflation for computing large 2-d/3-d convex regions of obstacle-free space," *arXiv preprint arXiv:2403.02977*, 2024.

[2] 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.

[3] 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*, vol. 2, no. 3, pp. 1688–1695, 2017.

[4] 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.

[5] P. Werner, T. Cohn, R. H. Jiang, T. Seyde, M. Simchowitz, R. Tedrake, and D. Rus, “Faster algorithms for growing collision-free convex polytopes in robot configuration space,” *arXiv preprint arXiv:2410.12649*, 2024.

[6] Y. Ren, F. Zhu, G. Lu, L. Yin, “Safety-assured high-speed navigation for MAVs,” *Science Robotics*, 2025.

[7] T. Cohn, “https://github.com/cohnt/iris-2d” *IRIS-2D*, 2022.