2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Robust Collision Avoidance of Quadric and Polygonal Surfaces Moving in Planar Environments

Kashish Dhal[1], Abhishek Kashyap[1], Animesh Chakravarthy[1]

Abstract— **The determination of robot trajectories that can achieve reactive collision avoidance with moving objects is of great interest. In cluttered environments with tight spaces, it becomes important to consider the shapes of the objects in computing these trajectories. The literature largely models the shapes of the objects as circles, and this can make the avoidance maneuvers very conservative, especially when the objects are elongated and/or non-convex. In this paper, we model the shapes of the objects using combinations of quadric surfaces or polygons, and employ a collision cone approach to achieve reactive collision avoidance, in the presence of measurement noise. The collision avoidance design employs a dual-loop control architecture where the inner loop uses a dynamic inversion-based method and the outer loop uses Linear Matrix Inequalities (LMIs). Simulation results demonstrating the collision avoidance laws for dynamic, heterogeneous quadric and polygonal surfaces are presented.**

I. Introduction

Achieving reactive collision avoidance is an important component of the trajectory generation problem for autonomous vehicles. When the robot and obstacles are operating in close proximity and in cluttered environments, their individual shapes can play an important role in the determination of avoidance trajectories. A common practice is to use circular approximations for the objects, and then compute the avoidance conditions for these circles. However, the circular approximations become overly conservative for objects that are elongated and or non-convex. For example, consider the object shown in red in Fig 1(a). Approximating this with a single circle is highly over-conservative because this reduces the area of the free space in which the robot trajectories can lie. Approximating this more tightly with multiple smaller circles (See Fig 1(b),(c)) increases the available free space, however it leads to increased computational expense to store a model of the shape. If we consider the engagement between two such non-convex shapes, and we model the two objects using m and n circles, respectively, then the computational complexity of predicting collision between the two objects is $O(mn)$, which is quite expensive for large m and n.

For elongated objects, ellipses have been used to serve as better approximators for the shapes [1]-[2]. However for complex shapes, even elliptical approximations can become over conservative. In such cases, the shapes can be approximated with polygons as in [3]. For non-convex shapes, one can also take recourse to non-convex bounding approximations involving a combination of quadric surfaces such as an ellipse and a hyperbola. Such quadric surfaces are also used to model objects in CAD/CAM and industrial manufacturing [4], as well as complex object shapes in animations [5].

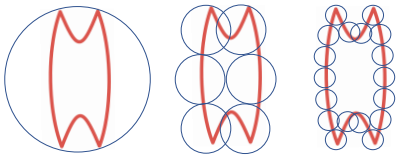
This paper employs a collision cone based approach to determine collision avoidance laws for moving objects whose shapes are modeled by combinations of quadric surfaces or polygons. The collision cone approach, originally introduced in [6], has some similarities with the velocity obstacle approach [7] in that both approaches determine the set of velocities of the robots that will place them on a collision course with one or more obstacles. The collision cone approach of [6] has been used to develop analytical expressions of collision cones for a large class of object shapes [8]. Such analytical expressions can lead to computational savings, especially in multi-obstacle environments, and also serve as a basis for designing collision avoidance laws. The collision cone approach has been extensively employed in the literature (See for example, [9],[10],[11],[12],[13],[14],[15]).

When both agents are objects of different shapes, a common recourse is to perform a Minkowski sum operation to reduce one of the objects to a point, while enlarging the other object. This operation may be computationally expensive. In [16], the authors computed the collision cone between moving quadrics without taking recourse to computing the Minkowski sum, and developed acceleration laws for collision avoidance, while assuming that the robots have perfect state information. In the current paper, we consider obstacles of a larger class of shapes and then use polygons or a combination of quadrics to approximate the shapes. The collision cone approach, used in the paper for calculating the obstacle avoidance trajectory, can be viewed as a general purpose algorithm, that can be applied to any of the above shape approximations. Given an image or a point cloud of the obstacle, users have flexibility to approximate the shape of the obstacle as an ellipse using Khachiyan's Algorithm [17] or as a polygon [18] or as a non-convex quadric, based on the accuracy needed and the computational limitations, and the collision cone algorithm works with either approximation.

In this paper, we consider scenarios where the robots have noisy, imperfect state information. We first perform an analytical quantification of the effect of these noisy states on the computation of the collision cone. We then employ a two-loop feedback architecture, with the objective to attenuate the effects of noise and thereby achieve robust collision avoidance. The inner loop provides a baseline acceleration component (developed using a dynamic inversion approach

This work was supported by a grant from the National Science Foundation IIS-1851817.

[1]Kashish Dhal, Abhishek Kashyap and Animesh Chakravarthy are with the Department of Mechanical and Aerospace Engineering, University of Texas at Arlington, TX, USA kashish.dhal@mavs.uta.edu, abhishek.kashyap@mavs.uta.edu, animesh.chakravarthy@uta.edu



(a) (b) (c)

Fig. 1: Approximating a non-convex shape using circles

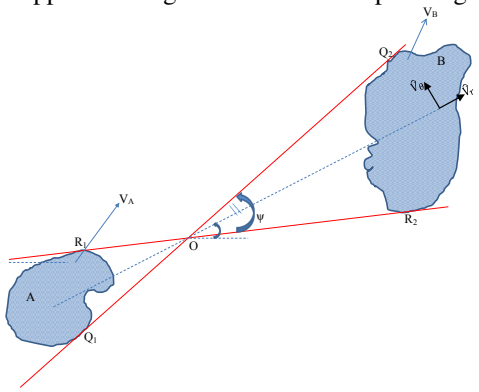


Fig. 2: Engagement between arbitrarily shaped objects

which does not account for the effects of noise), while the outer loop provides correctional components to this baseline acceleration. These correctional terms are developed using Linear Matrix Inequalities (LMIs) which account for the effects of measurement noise. This two-loop architecture achieves robust collision avoidance of moving obstacles with heterogeneous shapes.

II. Computation of the Collision Cone

A. Background on the Collision Cone

Refer Fig 2, which shows two arbitrarily shaped objects A and B moving with velocities V_A and V_B, respectively. The lines Q_1Q_2 and R_1R_2 form a sector with the property that this represents the smallest sector that completely contains A and B such that A and B lie on opposite sides of the point of intersection O. Let $\hat{V}_r$ and $\hat{V}_\theta$ represent the relative velocity components of the angular bisector of this sector, as shown. As demonstrated in [6], A and B are on a collision course if their relative velocities belong to a specific set. This set is encapsulated in a quantity y defined as follows:

$$y = \frac{\hat{V}_\theta^2}{(\hat{V}_\theta^2 + \hat{V}_r^2)} - \sin^2\left(\frac{\psi}{2}\right) \tag{1}$$

The collision cone is defined as the region in the $(\hat{V}_\theta, \hat{V}_r)$ space for which $y < 0$, $\hat{V}_r < 0$ is satisfied. Any relative velocity vector satisfying this condition lies inside the collision cone. A challenge in computing the collision cone for arbitrarily shaped objects is in the computation of the sector enclosing the objects A and B (shown in Fig 2), and determination of the angle ψ. Note that as A and B move, the angle ψ changes with time. In [16], the authors presented a method to compute ψ for objects that can be modeled by quadric surfaces. This method is computationally inexpensive thereby making it suitable for real-time implementation, and is briefly discussed in the subsection below.

Algorithm 1 Determination of angle ψ

// Given two ellipses represented by matrices M_1 and M_2

Compute $C_1 = M_1^{-1}$, $C_2 = M_2^{-1}$

// Determine the degenerate conics

$[U, \mathbf{t}] = eig(C_2^{-1}C_1)$ ▷ $U = [\mathbf{u_1}\ \mathbf{u_2}\ \mathbf{u_3}]$, $t = [t_1,\ t_2,\ t_3]^T$

// Do Projective transform of degenerate conic by U

$L_i = U^T(C_1 - t_i C_2)U$ for $i = 1, 2$

// Find one of the intersection point of these projected degenerate conics

$u = \sqrt{(-L_2(3,3))/L_2(1,1)}$, $v = \sqrt{(-L_1(3,3))/L_1(2,2)}$

// Find all tangent lines in projected coordinates

$$s = \begin{bmatrix} u & v & 1 \\ -u & -v & 1 \\ -u & v & 1 \\ u & -v & 1 \end{bmatrix}$$

// Project solution back to homogeneous coordinates

$s = sU^T$

// Define the centers of M_1 and M_2

$p_1 = C_1[0\ 0\ 1]^T$, $p_2 = C_2[0\ 0\ 1]^T$

// Determine the inner tangents by using the property that two centers of the ellipses lie on opposite sides of the inner tangents. Defining a boolean vector

$bool = (sp_1) \odot (sp_2) < 0$ ▷ $\odot$ is Hadamard product

// Eliminating the indices of s marked as false in $bool$

$s = s(bool)$

Check if p_1 is located below tangent 1 and above tangent 2. If not, change sign of the normal vector of tangent 2

if $(s(1,:).p_1)(s(2,:).p_1) > 0$, $s(2,:) = -s(2,:)$ **end if**

// Determine the angle ψ

$$\psi = \arccos\left(\frac{s(1,1:2).s(2,1:2)}{\|s(1,1:2)\|\,\|s(2,1:2)\|}\right)$$

B. Collision cone between two ellipses

A general ellipse can be represented as follows:

$$aX^2 + bXY + cY^2 + dX + eY + f = 0 \tag{2}$$

$$\Rightarrow \underbrace{\begin{bmatrix} X & Y & 1 \end{bmatrix}}_{\mathbf{x}^T} \underbrace{\begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}}_{\mathbf{x}} = 0 \tag{3}$$

Eqn (3) can be compactly written as $\mathbf{x}^T M \mathbf{x} = 0$. Note that $\mathbf{x}$ represents the homogeneous coordinates of a point (x, y). Now, for any given point $\mathbf{p}$ on the ellipse, $\mathbf{l} = M\mathbf{p}$ represents the homogeneous coordinates of the tangent to the ellipse at that point, and the equation of this tangent line is given by $\mathbf{l}^T\mathbf{x} = 0$. The dual of this ellipse is defined as the set of lines tangent to M, and can be expressed by the equation $\mathbf{l}^T M^{-1}\mathbf{l} = 0$ [19]. Thus, any line satisfying this equation belongs to the tangent set of M. To find the equations of the common tangents to two ellipses, the intersections between their corresponding duals are required. We use Algorithm 1 to determine the inner common tangents and the angle ψ between them, and then use this in (1) to compute the collision cone.

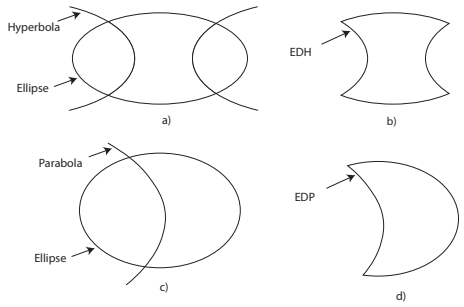


Fig. 3: *EDP* and *EDH*

C. Collision cone between two polygons

Given a set of points obtained from the point cloud of a LiDAR, the Graham scan [18] can be used to obtain the smallest convex hull that approximates the point cloud by a polygon with the smallest number of sides in $\mathcal{O}(n \log n)$.

Assume there are m and n vertices corresponding to the convex hulls of the two objects. Thus mn lines can be drawn between the vertices of the two polygons. The common inner tangents to the two polygons are identified as the tangent lines that also serve as separating hyperplanes between the two objects, and these are used to compute ψ which is used in (1) to compute the collision cone. Note that we use the phrase "tangent" with a slight abuse of notation in the sense that these lines may not actually be tangents when the object has a non-smooth surface as in the case of a polygon.

D. Collision cone between shapes formed by the combination of two general quadrics

1) Equation for quadrics made by multiple overlapping quadrics: Consider an intersecting ellipse and hyperbola, as shown in Fig 3(a). Then, we define an Ellipse Delimited by a Hyperbola (EDH) as follows:

$$\{\mathbf{x} : \mathbf{x}^T \mathbf{M_e} \mathbf{x} \leq 0 \bigcap \mathbf{x}^T \mathbf{M_h} \mathbf{x} \leq 0\} \tag{4}$$

where, M_h and M_e represent matrices that correspond to a hyperbola and an ellipse, respectively. Note that the structure of M_h and M_e is similar to that in (3). An example of an EDH is shown in Fig 3(b). In the special case when the focal points of the ellipse and hyperbola coincide, the ensuing shape is a confocal quadric which was described in our previous paper [16]. Similarly, we can construct a quadric which is a combination of an ellipse and a parabola shown in Fig 3(c). Doing so, we obtain an ellipse delimited by a parabola (EDP) shown in Fig 3(d). These quadrics can be used to approximate the objects more tightly.

2) Computation of angle ψ for these surfaces: Ref Fig 4, which shows two general quadrics A and B. To find the common tangents between them, we follow the steps given below. 1) First, draw tangents from the elliptical portion of A to the elliptical portion of B. These are t_1t_1' and t_2t_2' in Fig 4. 2) Check if these lines satisfy the equations of the surfaces representing A and B. If yes, then they can be considered as valid common tangents; if not, we proceed to the next step. In Fig 4, t_2t_2' is valid while t_1t_1' is not. 3) Find the intersection

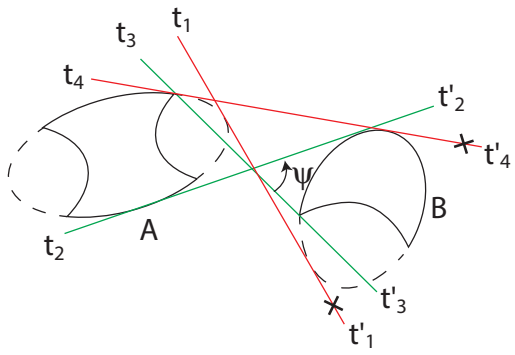


Fig. 4: Computing ψ for a pair of general quadrics

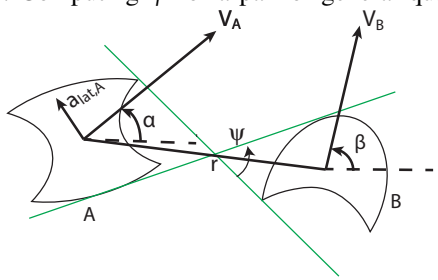


Fig. 5: Engagement Geometry between general quadrics

of the ellipse and hyperbola using Algorithm 1 where C_1 and C_2 are the ellipse and hyperbola matrices, respectively. These intersection points are the corner points of that general quadric. 4) From each corner point of a quadric, draw two tangents to the elliptical portion of the other quadric, shown as line t_4t_4' in Fig 4. Check if these lines satisfy the equation of that quadric and form separating hyperplanes. If so, they can be considered as valid common tangents. Here t_4t_4' does not satisfy the criteria. If we have not obtained two valid tangents yet, we proceed to the next step. 5) We take the corner points of each quadric and draw two tangents between them as described in the case of the polygons. Here, t_3t_3' is a valid tangent joining the corner points of the two shapes. 6) If we have three valid tangent lines, then we select the two which gives the maximum ψ.

E. Computational Complexity for calculating ψ for different shape approximations

Ellipse-Ellipse: As we can see from Algorithm 1, all the operations are constant time operations, therefore the worst time complexity is $\mathcal{O}(1)$. Polygon-Polygon: Let m and n be the number of vertices of the two polygons, where $m > n$. The time complexity is $\mathcal{O}(m^2 n)$, since there are mn tangent lines and each of them have to be checked to find those which are also separating hyperplanes. General Quadric-Polygon: Here the tangents are drawn using the combination of Algorithm 1 and polygon-polygon case. Since the time complexity of Algorithm 1 is $\mathcal{O}(1)$ and the number of corner points is upper bounded by 4, therefore the total complexity becomes $\mathcal{O}(1 \cdot n^2 \cdot 4)$, n the number of polygon vertices, simplified to $\mathcal{O}(n^2)$. General Quadric-General Quadric: The algorithm is similar to the one used in the above case. Since each general quadric has maximum four corner points, therefore the time complexity becomes $\mathcal{O}(1 \cdot 4^2 \cdot 4)$ simplified to $\mathcal{O}(1)$.

Shape Engagement	Complexity
Ellipse-Ellipse	$\mathcal{O}(1)$
General Quadric-General Quadric	$\mathcal{O}(1)$
General Quadric-Polygon	$\mathcal{O}(n^2)$
Polygon-Polygon	$\mathcal{O}(m^2n), m > n$

I. Complexity Analysis

III. Collision Avoidance Acceleration Computation by Dynamic Inversion

In this section, we derive analytical expressions for the acceleration laws required for collision avoidance. Consider the engagement geometry in Fig 5, where A and B are two general quadrics, moving with speeds V_A and V_B, respectively, and heading angles α and β, respectively. The distance between the centers of A and B is represented by r, and the angle made by the line joining these centers is represented by θ. The control input of A is its lateral acceleration $a_{lat,A}$, which acts normal to the velocity vector of A. The kinematics governing the engagement geometry are characterized by the following equations:

$$\begin{bmatrix} \dot{r} \\ \dot{\theta} \\ \dot{V}_\theta \\ \dot{V}_r \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} V_r \\ V_\theta/r \\ -V_\theta V_r/r \\ V_\theta^2/r \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -\cos(\alpha-\theta) \\ \sin(\alpha-\theta) \\ 1/V_A \end{bmatrix} a_{lat,A} \quad (5)$$

The inner common tangents are shown in green in Fig 5 and ψ is the angle between these tangents. The quantity θ_b represents the angle which the angular bisector of the sector (formed by the inner common tangents), makes with the x-axis. In general, the angular bisector of the sector will be distinct from the line joining the centers of A and B, and thereby have different relative velocity components. The quantities V_r, V_θ are related to $\hat{V}_r$, $\hat{V}_\theta$ (bisector quantities) by a rotation matrix by an angle $(\theta-\theta_b)$. In the special case when $\theta = \theta_b$ we have $V_r = \hat{V}_r$ and $V_\theta = \hat{V}_\theta$.

Thus we can transform the function y (defined in (1)), so as to write it in terms of the kinematic states:

$$y = \frac{\begin{bmatrix} V_\theta^2\cos^2(\theta-\theta_b) + V_r^2\sin^2(\theta-\theta_b) \\ +2V_rV_\theta\cos(\theta-\theta_b)\sin(\theta-\theta_b) \end{bmatrix}}{V_r^2+V_\theta^2} - \sin^2\left(\frac{\psi}{2}\right) \quad (6)$$

When $y < 0, \hat{V}_r < 0$, this means that A is on a collision course with B. It needs to apply a suitable lateral acceleration $a_{lat,A}$ to drive y to a reference value $w \geq 0$, and this will be equivalent to steering its velocity vector out of the collision cone. We employ dynamic inversion to determine the baseline lateral acceleration to drive y to the reference w. Differentiating (6), we obtain the dynamic evolution of y:

$$\dot{y} = \frac{\partial y}{\partial \theta_b}\dot{\theta}_b + \frac{\partial y}{\partial \theta}\dot{\theta} + \frac{\partial y}{\partial V_\theta}\dot{V}_\theta + \frac{\partial y}{\partial V_r}\dot{V}_r + \frac{\partial y}{\partial \psi}\dot{\psi} \quad (7)$$

Define an error quantity $z(t) = y(t) - w$. Taking w as a constant $\forall t$, we seek to determine $a_{lat,A}$ which will ensure the error $z(t)$ follows the dynamics $\dot{z} = -Kz$ where $K > 0$ is a constant. This in turn causes the quantity y to follow the dynamics $\dot{y} = -K(y-w)$.

Note that all the partial derivatives of y can be computed analytically from (6). While the state kinematic equations are given in (5), we however do not have analytical expressions of $\dot{\theta}_b$ and $\dot{\psi}$ and these have to be synthesized numerically. Substituting partial derivatives of y and state derivatives from (5) in (7), we eventually get the expression for $a_{lat,A}$ as:

$$a_{lat,A} = -(V_r^2+V_\theta^2)\frac{N_1+N_2}{D_1D_2} \quad (8)$$

where, N_1, N_2, D_1 are as follows:

$$\begin{aligned} N_1 &= (V_r^2+V_\theta^2)(2K(w-y)+\dot{\psi}\sin(\psi)) \\ D_1 &= 2V_rV_\theta\cos(2(\theta-\theta_b)) + (V_r^2-V_\theta^2)\sin(2(\theta-\theta_b)) \\ N_2 &= 2\dot{\theta}_bD_1, D_2 = 2\left(V_r\cos(\alpha-\theta)+V_\theta\sin(\alpha-\theta)\right) \end{aligned}$$

IV. Robust Controller Design using LMIs

The acceleration law (8) derived by the dynamic inversion method requires accurate knowledge of the true values of the states. However, as is well known, measurement sensors possess noise which corrupts the state measurements thereby causing the controller to behave improperly. In this section, we quantify the effect of noise on the dynamics of the collision cone parameter y. We then design an LMI-based controller to attenuate the effects of noise, and thereby achieve robust collision avoidance performance.

A. Effects of Noise on Collision Cone Parameter Dynamics

In the absence of noise, the dynamic inversion law ensures that y follows the dynamics $\dot{y} = -K(y-w)$, or equivalently, $\dot{z} = -Kz$. Now, assume the presence of noise so that the measurements of the relative engagement states r, θ, V_r, V_θ, θ_b and ψ are each corrupted by additive noise terms Δr, $\Delta\theta$, ΔV_r, ΔV_θ, $\Delta\theta_b$ and $\Delta\psi$, respectively. We accent the erroneous states by the symbol $\sim$. Therefore, $\tilde{r} = r + \Delta r$, $\tilde{\theta} = \theta + \Delta\theta$, and similarly for the other states. This noise, as a consequence, leads to an error in the value of the collision cone parameter y. Let this error be represented by Δy, so that the noisy value is $\tilde{y} = y + \Delta y$. Additionally, the dynamic inversion law includes the derivative terms $\dot{\theta}_b$ and $\dot{\psi}$, which are also corrupted to $\dot{\tilde{\theta}}_b$ and $\dot{\tilde{\psi}}$, respectively. Substituting these terms in (8), we see that the applied acceleration is

$$\begin{aligned} \tilde{a}_{lat,A} &= -(\tilde{V}_r^2+\tilde{V}_\theta^2)\frac{\tilde{N}_1+\tilde{N}_2}{\tilde{D}_1\tilde{D}_2} \quad (9) \\ \tilde{N}_1 &= (\tilde{V}_r^2+\tilde{V}_\theta^2)(2K(w-\tilde{y})+\dot{\tilde{\psi}}\sin(\tilde{\psi})) \\ \tilde{D}_1 &= 2\tilde{V}_r\tilde{V}_\theta\cos(2(\tilde{\theta}-\tilde{\theta}_b)) + (\tilde{V}_r^2-\tilde{V}_\theta^2)\sin(2(\tilde{\theta}-\tilde{\theta}_b)) \\ \tilde{N}_2 &= 2\dot{\tilde{\theta}}_b\tilde{D}_1, \tilde{D}_2 = 2\left(\tilde{V}_r\cos(\alpha-\tilde{\theta})+\tilde{V}_\theta\sin(\alpha-\tilde{\theta})\right) \end{aligned}$$

Substituting (9) in (7), after performing algebraic manipulations, we see that the error z now follows the dynamics:

$$\dot{z} = -Kaz + v \quad (10)$$

where, a and v are

$$a = \frac{D_1D_2\left(\tilde{V}_r^2+\tilde{V}_\theta^2\right)^2}{\tilde{D}_1\tilde{D}_2\left(V_r^2+V_\theta^2\right)^2} \quad (11)$$

$$v = a\underbrace{\left(\frac{\tilde{N}_2}{2\left(\tilde{V}_r^2+\tilde{V}_\theta^2\right)}+\dot{\tilde{\psi}}\frac{\sin\psi}{2}-\Delta y\right)}_{T}-\underbrace{\left(\frac{\dot{\theta}_b D_1}{\left(V_r^2+V_\theta^2\right)}+\dot{\psi}\frac{\sin\psi}{2}\right)}_{U} \tag{12}$$

As evident from (10), a and v represent multiplicative and additive noise terms respectively. They impact the dynamics of the collision cone error function z and are functions of the states and their erroneous measurements. Note that in the absence of noise (when $\tilde{V}_r = V_r$, $\tilde{V}_\theta = V_\theta$ and so on), these terms reduce to $a = 1$, $v = 0$ and (10) reduces to $\dot{z} = -Kz$.

B. Bounds on the Multiplicative and Additive Noise Terms

In this subsection, we determine bounds on the multiplicative and additive noise terms a and v, respectively, by determining the mean and variance of these quantities, in terms of the mean and variance of the noisy state measurements. We use a linearized approximation for this purpose, with a standard assumption that the noise is Gaussian with zero mean and known variance. We first consider the multiplicative term a in (10), and get $\mathrm{E}[a] = 1$. To determine the variance of a, we define a quantity $G = \left(V_r^2+V_\theta^2\right)^2/(D_1 D_2)$, and note from (11), that $a = \tilde{G}/G$. Then the variance of a can be written in terms of the partial derivatives of G as:

$$\begin{aligned}\mathrm{Var}[a] &= \frac{1}{G^2}\left(\left(\frac{\partial G}{\partial V_r}\right)^2\mathrm{Var}[\Delta V_r]+\left(\frac{\partial G}{\partial V_\theta}\right)^2\mathrm{Var}[\Delta V_\theta]\right.\\ &+ \left.\left(\frac{\partial G}{\partial \theta}\right)^2\mathrm{Var}[\Delta\theta]+\left(\frac{\partial G}{\partial \theta_b}\right)^2\mathrm{Var}[\Delta\theta_b]\right)\end{aligned} \tag{13}$$

We next consider the additive term v in (12). Using a linearized approach, we get $\mathrm{E}[v] = 0$, and variance as:

$$\mathrm{Var}[v] = \mathrm{Var}[aT - U] = \mathrm{Var}[aT] \tag{14}$$

where, T and U are as defined in (12). The second equation above follows from the fact that U does not contain any noise terms and therefore does not contribute to the calculation of variance. Using a Taylor series approximation on a and T, ignoring the higher order terms, and assuming the variance of the individual terms to be independent, we get the following:

$$\begin{aligned}\mathrm{Var}[v] &= \left(\frac{\partial T}{\partial V_r}+T_0\frac{\partial a}{\partial V_r}\right)^2\mathrm{Var}[\triangle V_r]+\left(\frac{\partial T}{\partial \psi}\right)^2\mathrm{Var}[\triangle\psi]\\ &+ \left(\frac{\partial T}{\partial V_\theta}+T_0\frac{\partial a}{\partial V_\theta}\right)^2\mathrm{Var}[\triangle V_\theta]+\left(\frac{\partial T}{\partial \dot{\theta}_b}\right)^2\mathrm{Var}[\triangle\dot{\theta}_b]\\ &+ \left(\frac{\partial T}{\partial \theta}+T_0\frac{\partial a}{\partial \theta}\right)^2\mathrm{Var}[\triangle\theta]+\left(\frac{\partial T}{\partial \dot{\psi}}\right)^2\mathrm{Var}[\triangle\dot{\psi}]\\ &+ \left(\frac{\partial T}{\partial \theta_b}+T_0\frac{\partial a}{\partial \theta_b}\right)^2\mathrm{Var}[\triangle\theta_b]\end{aligned} \tag{15}$$

where a_0 and T_0 denote the true values as given below

$$a_0 = 1, \quad T_0 = \left(\frac{N_2}{2\left(V_r^2+V_\theta^2\right)}+\dot{\psi}\frac{\sin\psi}{2}\right) \tag{16}$$

Fig 6 shows the time plots of the additive and multiplicative noise terms, as well as the calculated 3σ bounds for each of these terms, as determined from (13) and (14),

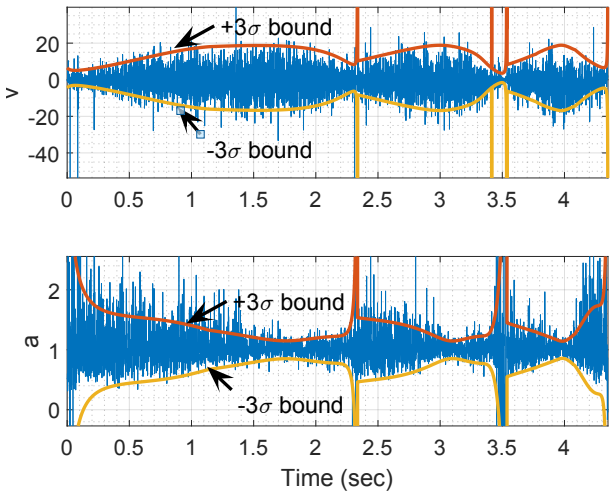


Fig. 6: (a) Additive (b) Multiplicative Noise, along with their 3σ bounds

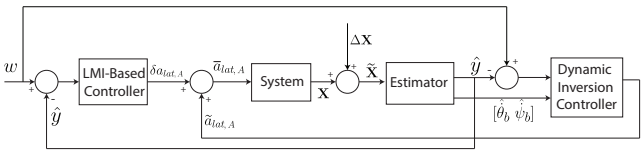


Fig. 7: Two-Loop Control Architecture

respectively, for a scenario (described later in the simulations section). These 3σ bounds help us to specify the bounds on the parameters of the robust controller discussed next.

C. Implementation of LMI-based Controller

To attenuate the adverse effects of the a, v terms in (10), an additional input $\Delta a_{lat,A}$ is added to controller (9) as:

$$\bar{a}_{lat,A} = \tilde{a}_{lat,A} + \Delta a_{lat,A} \tag{17}$$

We now demonstrate use of an LMI-based approach for designing the $\Delta a_{lat,A}$ term. Substitute (17) in (10) to get:

$$\dot{z} = -Kaz + b\Delta a_{lat,A} + v \tag{18}$$

where the quantity b is:

$$b = -D_1 D_2/\left(2\left(V_r^2+V_\theta^2\right)^2\right) \tag{19}$$

We note that the values of both a and b are unknown. We can however use the analysis of the previous subsection to guide us in inferring bounds on a and b. Let a lie within the bounds $[-\nu, \nu]$, and let the bound on the absolute value of b be λ. We assume that the sign of b is known. Rewriting (18) in discrete form, we get

$$z(k+1) = Az(k) + B\Delta a_{lat,A}(k) + Dv(k) \tag{20}$$

In the above equation, the quantities A, B and D lie within bounds $[A_1, A_2]$, $[B_1, B_2]$, and $[D_1, D_2]$, respectively, where these bounds are defined in terms of ν, λ and the discretization time step $\triangle t$. The values of A, B and D thus lie within

a polytope Ω, which is defined as follows:

$$\Omega = \{[A_1, B_1, D_1], [A_2, B_2, D_2]\}$$
$$= \left\{ \left[e^{K\nu\triangle t}, 1/(K\nu)(e^{K\nu\triangle t}-1)\lambda, 1/(K\nu)(e^{K\nu\triangle t}-1) \right], \right.$$
$$\left. \left[e^{-K\nu\triangle t}, 1/(-K\nu)(e^{-K\nu\triangle t}-1)\lambda, 1/(-K\nu)(e^{-K\nu\triangle t}-1) \right] \right\}$$

A robust input-constrained MPC controller is implemented using LMIs [20] to generate a feedback control law of the form $\Delta a_{lat,A}(k) = K(k)z(k)$, that satisfies several objectives. It should minimize the tracking error z, while also ensuring that the influence of the disturbance v on z is kept below a pre-defined threshold. Also, the magnitude of $\Delta a_{lat,A}$ should be such that the total acceleration (which includes the components generated from dynamic inversion and the LMIs) remains within the actuator saturation limits. These objectives are mathematically formulated as follows:
1) Let J_∞ represent a cost function defined as:

$$J_\infty(k) = \sum_{i=0}^{\infty} \left(\hat{Q} z^2(k+i) + \hat{R} \Delta a_{lat}^2(k+i) \right) \tag{21}$$

where, $\hat{Q}$ and $\hat{R}$ are weights on z and $\Delta a_{lat,A}$ respectively. Determine $\Delta a_{lat,A}$, to minimize the upper bound Γ on $J_\infty(k)$.
2) Let T_z^v represent the transfer function from v to z. Determine $\Delta a_{lat,A}$ so as to ensure that the effect of v on z is upper bounded in the sense of the H_∞ norm of T_z^v, that is, $\|T_z^v\|_\infty < \mu$. Satisfying this objective will attenuate the influence of the term v in (20).
3) The input $\Delta a_{lat,A}(k)$ is constrained between the bounds $u_{min}(k)$ and $u_{max}(k)$. These limits are re-calculated such that for every simulation time step, the total acceleration $\bar{a}_{lat,A}$ lies within the actuator saturation limits $|\bar{a}_{lat,A}| \leq a_{lat,A_{max}}$

The above objectives are to be satisfied for the entire polytope Ω. To meet the above objectives, the MPC control gain K has the following structure:

$$K(k) = Y(k)Q^{-1}(k) \tag{22}$$

where, Y and Q are obtained by solving the following LMIs at each time step k :

$$\min_{\Gamma,Q,Y} \Gamma \tag{23}$$

$$\text{such that } \begin{bmatrix} 1 & z(k) \\ z(k) & Q \end{bmatrix} \geq 0 \tag{24}$$

$$\begin{bmatrix} Q & QA_i^T + Y^T B_i^T & Q\hat{Q}^{1/2} & Y^T \hat{R}^{1/2} \\ A_iQ + B_iY & Q & 0 & 0 \\ \hat{Q}^{1/2}Q & 0 & \Gamma I & 0 \\ \hat{R}^{1/2}Y & 0 & 0 & \Gamma I \end{bmatrix} \geq 0, i = 1,2 \tag{25}$$

$$\begin{bmatrix} Q & 0 & QA_i^T + Y^T B_i^T & Q \\ 0 & \Gamma\mu & \Gamma D_i^T & 0 \\ A_iQ + B_iY & \Gamma D_i & Q & 0 \\ Q & 0 & 0 & \Gamma I \end{bmatrix} \geq 0, i = 1,2 \tag{26}$$

$$\begin{bmatrix} u_{max}(k)Q - z(k)Y & 0 \\ 0 & z(k)Y - u_{min}(k)Q \end{bmatrix} \geq 0 \tag{27}$$

Note that the above equations (24)-(27) actually constitute a system of 6 LMIs, since (25)-(26) need to be satisfied at every corner point of the polytope Ω. A formal proof which demonstrates that obtaining a feasible solution to the above LMIs is equivalent to satisfying the three objectives mentioned above, can be found in [20].

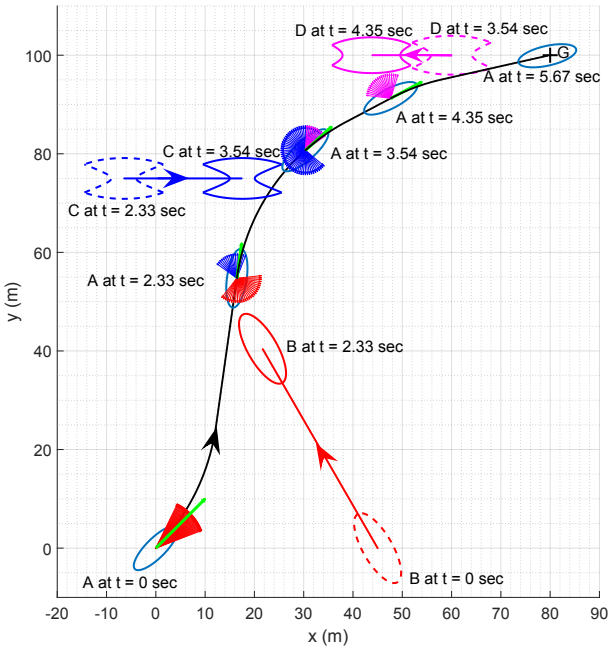


Fig. 8: Trajectory of the Agent and the obstacles

From (24), it is seen that the MPC requires the true value of the error $z(k)$ in order to calculate $\Delta a_{lat,A}$. Since only a noisy value of $z(k)$ is available, a Kalman Filter is used to estimate $z(k)$, and this is used in the LMIs. Also, measurement noise has compounding effects on the numerical derivative terms $\dot{\psi}$ and $\dot{\theta}_b$, which appear in the dynamic inversion acceleration law (8). This can make it difficult to find a feasible solution for the LMIs. Therefore the numerical derivatives $\dot{\psi}$ and $\dot{\theta}_b$ are also obtained through Kalman Filters. The overall control architecture used for collision avoidance thus combines computations from dynamic inversion-based control, LMI-based control and Kalman Filters. A block diagram of the overall two-loop architecture is provided in Fig. 7.

V. Simulation Results

Simulation Scenario I: An elliptical object A, initially at the origin, navigates through an environment with a series of fast-moving obstacles of varying quadric-shapes: an elliptical obstacle B, a non-convex confocal quadric obstacle C and finally a shape-changing confocal quadric obstacle D. The state measurements have additive Gaussian noise as follows:

$$\Delta r(t) = 20\% r(t)m(t), \Delta\theta(t) = 3^\circ m(t), \Delta V_r(t) = 40\% V_r(t)m(t),$$
$$\Delta V_\theta(t) = 40\% V_\theta(t)m(t), \Delta\psi(t) = 1^\circ m(t), \Delta\theta_b(t) = 1^\circ m(t)$$

where $m(t)$ is a Gaussian random variable of zero mean and standard deviation $1/3$. In the first phase, obstacle B starts from $(45,0)$ with a heading angle of 120° as seen in Fig. 8. A starts at the same time with an initial heading

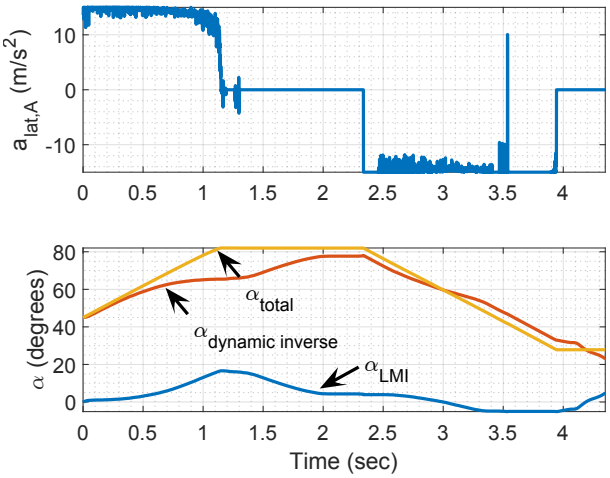


Fig. 9: a) Total Commanded Acceleration, $\bar{a}_{lat,A}$ b) Heading Angle

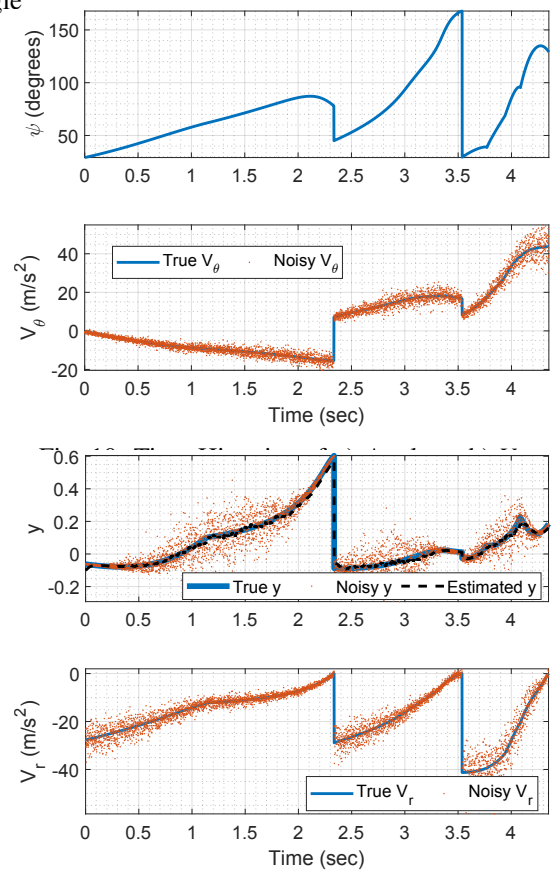


Fig. 11: Time Histories of a) Collision Cone y b) V_r

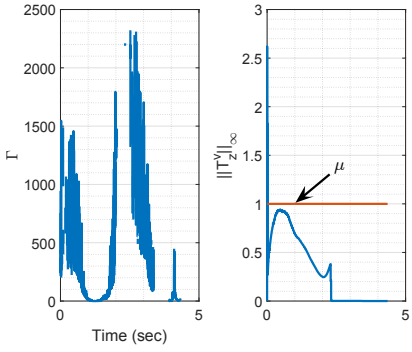


Fig. 12: Time Histories of a) Gamma b) $\|T_z^v\|_\infty$

angle of 45°. Fig. 8 provides a visualisation of the collision cone (marked by red arrows). The initial heading of A lies inside the collision cone, meaning A is on a collision course with B. Fig. 11 shows the collision cone parameters - true, noisy and estimated values of the collision cone function y, as well as the true and noisy relative velocity components V_r, $\tilde{V}_r$, V_θ and $\tilde{V}_\theta$. It is seen that initially the true y and V_r are both negative, indicating a collision course and furthermore, the noise in the states is quite substantial which impacts the acceleration obtained from the dynamic inversion algorithm. However, the LMI-generated $\Delta a_{Lat,A}$ provides a continuous additional acceleration that steers A away from collision. After $2.33\,s$, A comes out of the collision cone to B (See Fig. 8), both true V_r and y are positive (See Fig. 11) and the angle ψ starts to decrease (See Fig. 10). In the second phase, A faces a confocal quadric C which starts from $(-7,75)$ with a heading angle of 0. From Fig 11, it is evident that A is on a collision course with C, as $y<0$, $V_r<0$. Fig. 8 shows that the heading angle of A lies inside the collision cone to C. Again, accelerations from the dynamic inversion-based controller and the LMI-based controller act in tandem to steer the velocity vector of A out of the collision cone to C. Finally in the third phase, A encounters a shape-changing confocal quadric D, for which the angle ψ changes not only because of A coming closer to D but also due to the changing shape of D. However, the total commanded acceleration is able to steer A out of the collision cone, by $4.35\,s$. The simulation ends when A reaches it's goal at $(80,100)$. The complete time history of $\bar{a}_{lat,A}$ and the contribution of the heading angle changes generated by the dynamic inversion-based and the LMI-based commands are seen in Fig. 9. We note that the time profiles of the noise shown in Fig 6 is for the above simulation scenario.

The performance of the LMI-based controller is shown in Fig 12, which depicts the time history of the upper bound Γ on the LMI cost function in (21). It is seen that the LMI plays an active role in ensuring collision avoidance, except during the time intervals $[2,2.3]\,s$ and $[3.5,4]\,s$ during which Γ is undefined. This is because the velocity vector of A has been steered out of the collision cone to B by 2 sec, and during $[2,2.3]\,s$, A is not on a collision course, due to which the contribution from both the dynamic inversion and the LMI based controllers are zero. During $[3.5,4]\,s$, the dynamic inversion controller is fully saturated and thus the LMI-based controller does not contribute a correction. The LMI-based controller is also able to effectively reject the behaviour of the additive noise v as can be seen from Fig 12(b). This figure shows that the H_∞ norm of the transfer function T_s^v is less than the bound μ except at the first two time instants.

To further verify the effectiveness of the two loop Robust Controller, a Monte Carlo test is performed with 1000 cases with the noise as given above and collision avoidance is tested using the four controllers - Dynamic Inversion (DI), DI with the Kalman Filter, the two loop Robust Dynamic Inversion (RDI) Controller and Robust Dynamic Inversion (RDI) with Kalman filter. The results are shown in Fig 13. It can be seen that the RDI with Filter outperforms the other

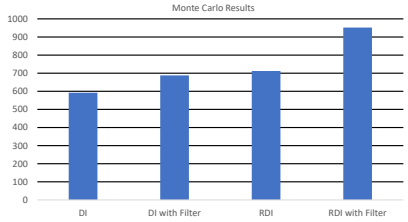


Fig. 13: Monte Carlo Results showing the effectiveness of the controllers

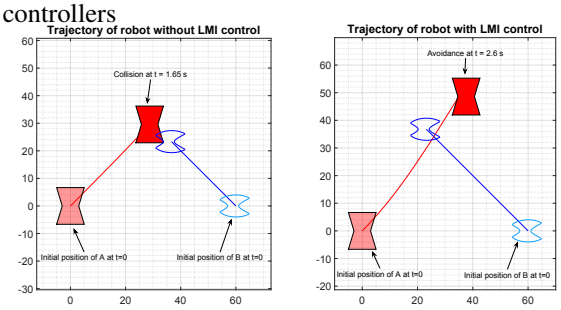


Fig. 14: Trajectory plots for Scenario II

controllers with 952 successes.

Simulation Scenario II: In this scenario, A is a polygon and B is a EDH. A has speed $25m/s$ with initial heading angle 45° and B has speed $20m/s$ with a constant heading angle of 145°. The inner tangents and ψ for the following engagement are computed using the steps described in the General Quadric-Polygon part of subsection II-E. The noise in the state measurements are kept the same as that described in Scenario I. Fig 14 shows the trajectory of the agent and the obstacle with and without the robust LMI controller. It can be seen that without the LMI controller, the agent doesn't deviate much from it's initial heading, as it is unable to calculate the required latax to avoid the obstacle and hence collision occurs. In the second plot it can be seen that with the LMI controller, a compensatory latax is generated that ensures collision avoidance.

VI. CONCLUSIONS

We present a two-loop feedback architecture that can achieve collision avoidance between moving, heterogeneous quadric surfaces (which are not necessarily the same shape), in the presence of measurement noise. The inner loop is designed using a dynamic inversion approach, while the outer loop is designed using an LMI-based approach, where the LMIs account for imperfections in the measurements. The design of both these loops rely on the analytical foundations provided by the collision cone approach. Simulations are presented to demonstrate the efficacy of the avoidance laws.

REFERENCES

[1] H. Kumar, S. Paternain, and A. Ribeiro, "Navigation of a quadratic potential with ellipsoidal obstacles," in *2019 IEEE CDC*.

[2] K. Nishimoto, R. Funada, T. Ibuki, and M. Sampei, "Collision avoidance for elliptical agents with control barrier function utilizing supporting lines," in *2022 American Control Conference (ACC)*, 2022.

[3] M. Missura and M. Bennewitz, "Predictive collision avoidance for the dynamic window approach," in *2019 ICRA*.

[4] Y.-K. Choi, W. Wang, B. Mourrain, C. Tu, X. Jia, and F. Sun, "Continuous collision detection for composite quadric models," *Graphical models*, vol. 76, no. 5, pp. 566–579, 2014.

[5] Q. Chen, G. Luo, Y. Tong, X. Jin, and Z. Deng, "Shape-constrained flying insects animation," *Computer Animation and Virtual Worlds*, vol. 30, no. 3-4, p. e1902, 2019.

[6] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment: A collision cone approach," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 1998.

[7] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, 1998.

[8] A. Chakravarthy and D. Ghose, "Collision cones for quadric surfaces," *IEEE Transactions on Robotics*, vol. 27, 2011.

[9] A. Haraldsen, M. S. Wiig, and K. Y. Pettersen, "Reactive collision avoidance for underactuated surface vehicles using the collision cone concept," in *2021 IEEE CCTA*, 2021.

[10] ——, "Reactive collision avoidance for nonholonomic vehicles in dynamic environments with obstacles of arbitrary shape," *IFAC-PapersOnLine*, vol. 54, 2021.

[11] E. Lalish and K. A. Morgansen, "Distributed reactive collision avoidance," *Autonomous Robots*, vol. 32, 2012.

[12] B. L. Boardman, T. L. Hedrick, D. H. Theriault, N. W. Fuller, M. Betke, and K. A. Morgansen, "Collision avoidance in biological systems using collision cones," in *2013 American Control Conference*.

[13] P. Karmokar, K. Dhal, W. J. Beksi, and A. Chakravarthy, "Vision-based guidance for tracking dynamic objects," in *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1106–1115.

[14] K. Dhal, P. Karmokar, A. Chakravarthy, and W. J. Beksi, "Vision-based guidance for tracking multiple dynamic objects," *Journal of Intelligent & Robotic Systems*, vol. 105, no. 3, p. 66, 2022.

[15] W. Zuo, K. Dhal, A. Keow, A. Chakravarthy, and Z. Chen, "Model-based control of a robotic fish to enable 3d maneuvering through a moving orifice," *IEEE Robotics and Automation Letters*, vol. 5, 2020.

[16] K. Dhal, A. Kashyap, and A. Chakravarthy, "Collision avoidance and rendezvous of quadric surfaces moving in planar environments," in *Proceedings of 2021 IEEE Control and Decision Conference*.

[17] M. J. Todd and E. A. Yıldırım, "On khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids," *Discrete Applied Mathematics*, vol. 155, no. 13, pp. 1731–1744, 2007.

[18] R. L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set," *Info. Proc. Lett.*, vol. 1, pp. 132–133, 1972.

[19] J. Richter-Gebert, "Conics and their duals," in *Perspectives on Projective Geometry*. Springer, 2011, pp. 145–166.

[20] M. V. Kothare, V. Balakrishnan, and M. Morari, "Robust constrained model predictive control using linear matrix inequalities," *Automatica*, vol. 32, 1996.