Piecewise-Polynomial Outer Approximations of Obstacles for Optimization-Based Collision Avoidance

James Guthrie

***Abstract*— We propose a novel condition for ensuring obstacle avoidance in optimization-based, 2D motion planning. This condition relies on representing the obstacle with two functions which provide lower and upper bounds of one coordinate as a parametric function of the other coordinate. In general these functions are not smooth and therefore problematic for standard nonlinear optimization solvers. We formulate a method based on sum-of-squares optimization for finding outer approximations using twice continuously differentiable piecewise-polynomial functions. We allow for the obstacle and its Minkowski sum with a disk to be bounded to accommodate distance-based constraints. Our formulation is advantageous in that it can tightly approximate complex unions of convex and select non-convex shapes with only two closed-form expressions. The associated condition for obstacle avoidance only introduces a single constraint to a trajectory generation problem reducing computational complexity. For convex obstacles, we compare the performance of our approximate representation to an alternative method that uses exact representations. On a Dubin's car path planning problem we achieve faster solve times while incurring only slight reductions in performance.**

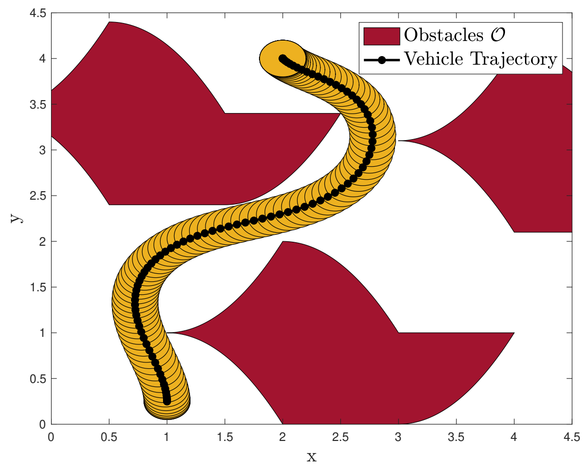

Fig. 1. Dubin's Car Navigating Obstacles With 0.2m Clearance

I. INTRODUCTION

Obstacle avoidance is a crucial component of motion planning algorithms for many robotic and autonomous systems. For mobile robots which do not move aggressively, it is often acceptable to neglect or approximate the dynamics. Sampling-based methods such as RRT or Hybrid A* are widely used in these situations [1], [2]. A shortcoming of these approaches is that they are generally limited to specific objectives such as finding the shortest-path. Additionally they often rely on representing control actions as a finite set of options. While achieving kinematic feasibility, the resulting trajectories may be difficult to track as they do not account for complex dynamic constraints [3].

Recent research has focused on using optimization-based methods for real-time motion planning that explicitly account for complex vehicle dynamics [4]–[6]. This is important for cases where the system dynamics cannot be neglected or the objective is not representable within a sampling-based planner. The majority of these methods involve solving a nonlinear optimization problem. Advances in both computing power and the availability of customized solvers has made this realizable [7]–[9].

In general, the nonlinear optimization solvers commonly used in motion planning require all constraints be closed-form expressions that are twice continuously differentiable. This is problematic for obstacle avoidance as closed-form expressions generally do not exist except for those involving circles. As such, one common approach is to approximate an object with (possibly multiple) circles and use the known closed-form expressions as constraints [6], [10]. However for tight maneuvering such as parking a car or navigating a narrow passage, these crude approximations can become unacceptable as they artificially render the problem infeasible.

A recent body of research has focused on how to incorporate more refined obstacle representations within optimization-based motion planning algorithms [3], [11]. These methods focus on cases in which objects are convex. In this case, the distance between two objects can be computed numerically using convex optimization methods. A direct embedding of optimization problem as a constraint within a larger nonlinear optimization problem would yield a bi-level problem which is computationally burdensome and not widely supported by solvers. Instead, in some instances one can directly encode the algebraic conditions that the global optimum is known to satisfy. Importantly, this can lead to a smooth reformulation of the collision avoidance constraint making it amenable to nonlinear optimization solvers [3], [12].

One downside of embedding optimization problems within optimization problems is that computational complexity often grows quickly leading to long runtimes. In [3], each constraint representing a face of an obstacle to be avoided requires introducing a new variable at each time index in the trajectory. For objects with complex descriptions this can

[1]James Guthrie is with the Johns Hopkins University, Baltimore, Maryland, USA. jguthri6@jhu.edu

quickly lead to slow solve times.

Instead of explicitly modeling the obstacle as a semi-algebraic set, one may instead seek to outer approximate this set with an implicit or parametric representation. In [13] the authors use sum-of-squares (SOS) optimization [14] to find polynomial representations of point clouds and related semi-algebraic sets. This allows complex geometry to be outer approximated by a single, implicit equation in $\mathbb{R}^3$. Tight approximations are sought by minimizing the volume of the unit sublevel set subject to set containment conditions. With the exception of ellipsoids, there does not exist expressions for the volume of the unit sublevel set of implicit functions. The authors propose various surrogate objectives for volume minimization. Related works have pursued similar ideas for forming inner or outer approximations of non-convex sets arising in various domains [15]–[17].

In this work, we propose using SOS optimization to find closed-form representations of obstacles for ensuring collision avoidance in optimization-based, 2D motion planning. Instead of seeking an implicit representation, we develop a new parametric form in which the obstacle is represented by two piecewise-polynomial functions that represent the obstacle's minimum and maximum y coordinates at a given x coordinate. This simple form allows us to directly minimize the resulting area. By bounding the Minkowski sum of the obstacle with a disk of given radius we obtain a single closed-form constraint that ensures a minimum separation between a point mass and the obstacle. This allows us to navigate complex non-convex obstacles as shown in Figure 1. Importantly no new variables are introduced to the optimization problem. We apply our method to planning for a Dubin's car model in the presence of an obstacle consisting of multiple convex polygons. We achieve runtimes 4-10x faster than the method of [3] at the cost of slight loss of optimality.

The rest of the paper is organized as follows. Section II sets up the problem. Section III develops an SOS-based approach for finding outer approximations of the obstacle. Section IV applies the resulting method to path planning for a Dubin's car model. Section V concludes the paper and discusses future directions.

A. Notation

For $\xi \in \mathbb{R}^n, \mathbb{R}[\xi]$ is the set of polynomials in ξ with real coefficients. The subset $\sum[\xi] = \{p = p_1^2 + p_2^2 + \ldots + p_n^2 : p_1, \ldots, p_n \in \mathbb{R}[\xi]$ of $\mathbb{R}[\xi]$ is the set of SOS polynomials in ξ. Let $\mathcal{D}_r = \{(x,y) \,|\, x^2+y^2 \leq r^2\}$, the disk of radius $r \geq 0$.

II. PROBLEM SETUP

Consider an obstacle in $\mathbb{R}^2$ represented by the following semi-algebraic set:

$$\mathcal{O} = \{(x,y) \in \mathbb{R}^2 | x^l \leq x \leq x^u, h^l(x) \leq y \leq h^u(x)\} \quad (1)$$

We assume that $h^l(x), h^u(x)$ are continuous but not necessarily smooth functions of x. Specifically, we represent $h^\bullet(x)$ where $\bullet$ indicates lower (l) or upper (u) as follows:

$$h(x) = \begin{cases} h_1(x), & \text{if } t_1 \leq x < t_2 \\ h_2(x), & \text{if } t_2 \leq x < t_3 \\ \ldots \\ h_n(x), & \text{if } t_n \leq x \leq t_{n+1} \end{cases} \quad (2)$$

Here each function $h_i(x)$ is assumed to be smooth, $h_i(t_{i+1}) = h_{i+1}(t_{i+1})$ for continuity and $t_1 = x^l, t_{n+1} = x^u$ represent the domain over which the function is defined.

We can define the interior of $\mathcal{O}$ as follows:

Definition 1. A point (x_p, y_p) is in the interior of $\mathcal{O}$ if the inequalities defining $\mathcal{O}$ hold strictly

$$h^l(x_p) < y_p < h^u(x_p) \quad x^l < x_p < x^u \quad (3)$$

Remark 1. If we did not assume that $h^\bullet(x)$ was piecewise-smooth then, at a point x_p where $h^\bullet(x_p)$ is discontinuous, there would be points on the boundary misclassified by (3) as being in the interior.

The following Lemma provides a related condition for determining if a point (x_p, y_p) is in the interior of $\mathcal{O}$.

Lemma 1. *A point (x_p, y_p) is in the interior of $\mathcal{O}$ if and only if $x^l < x_p < x^u$ and $(y_p - h^u(x_p))(y_p - h^l(x_p) < 0$*

Proof. The inequality constraints on x are simply from the definition. For the inequality on y_p to hold, either we have

$$y_p > h^u(x_p),\, y_p < h^l(x_p) \quad (4)$$

or we have

$$y_p < h^u(x_p),\, y_p > h^l(x_p) \quad (5)$$

The first case involves an obvious contradiction as $h^u(x) \geq h^l(x)$. Therefore only the second case can occur which agrees with the definition given. The reverse direction of the proof is just verifying the definition of the interior satisfies the stated conditions. □

From Lemma 1 we formulate the following condition that ensures a point mass (x_p, y_p) does not lie in the interior of obstacle $\mathcal{O}$:

$$(y_p - h^u(x_p))(y_p - h^l(x_p)) \geq 0 \quad \text{if} \quad x^l < x_p < x^u \quad (6)$$

Although a valid representation, this constraint is not supported in most nonlinear optimization methods used for trajectory generation as the logical condition and functions $h^l(x), h^u(x)$ are not smooth.

To eliminate the logical condition, consider the following augmented form.

$$\bar{h}^\bullet(x) = \begin{cases} a, & \text{if } x < x^l \\ h^\bullet(x), & \text{if } x^l \leq x \leq x^u \\ b, & \text{if } x > x^u \end{cases} \quad (7)$$

The parameters a and b are arbitrary constants but are the same in both $h^u(x)$ and $h^l(x)$. Condition (6) can then be replaced with the equivalent constraint

$$(y_p - \bar{h}^u(x_p))(y_p - \bar{h}^l(x_p)) \geq 0 \quad (8)$$

as outside the interval the equation reduces to $(y_p - a)(y_p - a) \geq 0$ and $(y_p - b)(y_p - b) \geq 0$ which holds trivially.

The functions $\bar{h}^u(x), \bar{h}^l(x)$ are still non-smooth however. One approach to resolve this is to search for polynomials $p^u(x), p^l(x)$ that closely approximate the non-smooth functions. In what follows, we setup a framework for doing this using sum-of-squares optimization.

III. Outer Approximations of Non-Convex Sets

We would like to find twice continuously differentiable approximations of the functions $\bar{h}^u(x), \bar{h}^l(x)$ that can be combined with condition (8) to enforce obstacle avoidance within an optimization-based motion planner. In obstacle avoidance where safety is paramount, we generally would prefer to guarantee that our approximations are conservative - i.e. they form an outer approximations of the true set $\mathcal{O}$. To encourage tight fits we minimize the volume of the area over the interval $x^l \leq x \leq x^u$. This can be posed as an optimization problem:

$$\begin{aligned} \min \quad & \int_{x^l}^{x^u} p^u(x) - p^l(x) dx \\ \text{s.t.} \quad & p^u(x) \geq h^u(x) \quad \forall \quad x \in \mathcal{X} \\ & p^l(x) \leq h^l(x) \quad \forall \quad x \in \mathcal{X} \\ & p^u(x) = p^l(x) \quad \forall \quad x \notin \mathcal{X} \end{aligned} \tag{9}$$

where $\mathcal{X}$ is the interval:

$$\mathcal{X} = \{x : x^l \leq x \leq x^u\} \tag{10}$$

More generally, we may be interested in ensuring that (x_p, y_p) satisfies a given Euclidean distance r from $\mathcal{O}$. This is equivalent to ensuring that (x_p, y_p) is not contained in the Minkowski sum of the set $\mathcal{O}$ with a disk $\mathcal{D}_r$.

$$\mathcal{O} \bigoplus \mathcal{D}_r = \{ \begin{bmatrix} x_o \\ y_o \end{bmatrix} + \begin{bmatrix} x_d \\ y_d \end{bmatrix} | (x_o, y_o) \in \mathcal{O}, x_d^2 + y_d^2 \leq r^2 \} \tag{11}$$

Given that $\mathcal{O}$ contained values of x in the interval $x^l \leq x \leq x^u$ it is obvious that the set resulting from the Minkowski sum extends this interval to $x^l - r \leq x \leq x^u + r$. As such we must modify our objective to minimize the area over this new interval. Additionally, we introduce separate (x, y) variables for $\mathcal{O}$ and $\mathcal{D}_r$ so they can vary independently of one another. We denote these as (x_h, y_h) and (x_d, y_d) respectively. The polynomial functions then bound the points (x_p, y_p) resulting from their sum.

$$\begin{aligned} \min \quad & \int_{x^l - r}^{x^u + r} p^u(x) - p^l(x) dx \\ \text{s.t.} \quad & p^u(x_p) \geq h^u(x_h) + y_d \quad \forall \quad (x_p, x_h, x_d, y_d) \in \mathcal{W} \\ & p^l(x_p) \leq h^l(x_h) + y_d \quad \forall \quad (x_p, x_h, x_d, y_d) \in \mathcal{W} \\ & p^u(x_p) = p^l(x_p) \quad \forall \quad x_p \notin \mathcal{X}_p \end{aligned} \tag{12}$$

where $\mathcal{X}_p$ is the interval

$$\mathcal{X}_p = \{x : x^l - r \leq x \leq x^u + r\} \tag{13}$$

and $\mathcal{W}$ is the semi-algebraic set:

$$\begin{aligned} \mathcal{W} = \{ (x_p, x_h, x_d, y_d) : x_p = x_h + x_d, x^l \leq x_h \leq x^u \\ x^l - r \leq x_p \leq x^u + r, x_d^2 + y_d^2 \leq r^2 \} \end{aligned} \tag{14}$$

While the above formulation is valid, the end condition $p^u(x_p) = p^l(x_p) \forall x_p \notin \mathcal{X}_p$ forces the polynomials to be the same. If $\mathcal{O}$ is non-empty, searching over polynomials is infeasible. However, for nonlinear optimization solvers we generally only need functions that are twice continuously differentiable. Motivated by this observation, we replace our polynomials with the following piecewise-smooth function:

$$p(x) = \begin{cases} p_0(x)(= a), & \text{if } x < c_1 \\ p_1(x), & \text{if } c_1 \leq x < c_2 \\ p_2(x), & \text{if } c_2 \leq x < c_3 \\ \dots \\ p_n(x), & \text{if } c_n \leq x \leq c_{n+1} \\ p_{n+1}(x)(= b), & \text{if } x > c_{n+1} \end{cases} \tag{15}$$

Here each function $p_i(x)$ is assumed to be smooth and $p_i(c_{i+1}) = p_{i+1}(c_{i+1})$ for continuity. We choose endpoints $c_1 \leq x^l - r$ and $c_{n+1} \geq x^u + r$ that bound the set $\mathcal{X}_p$. Outside of this interval, the piecewise functions are set equal to each other so the obstacle avoidance condition (6) holds trivially. We now explicitly write the functions $h^l(x_h), h^u(x_h)$ in terms of the underlying n_h functions given in (2). Similarly we write $p^l(x_p), p^u(x_p)$ in terms of the underlying n_p non-constant functions and constant endpoints $p_0^u(x_p), p_0^l(x_p), p_{n+1}^u(x_p), p_{n+1}^l(x_p)$. The resulting optimization problem is:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n_p} \int_{c_i}^{c_{i+1}} p_i^u(x) - p_i^l(x) dx \\ \text{s.t.} \quad & p_i^u(x_p) \geq h_j^u(x_h) + y_d \quad \forall (x_p, x_h, x_d, y_d) \in \mathcal{Z}_{i,j} \\ & \qquad i = 1, \dots, n_p, \quad j = 1, \dots, n_h \\ & p_i^l(x_p) \leq h_j^u(x_h) + y_d \quad \forall (x_p, x_h, x_d, y_d) \in \mathcal{Z}_{i,j} \\ & \qquad i = 1, \dots, n_p, \quad j = 1, \dots, n_h \\ & p_0^u(x_p) = a, \quad p_0^l(x_p) = a \\ & p_{n_p+1}^u(x_p) = b, \quad p_{n_p+1}^l(x_p) = b \\ & (p_{i+1}^u(x_p) - p_i^u(x_p))|_{x_p = c_{i+1}} = 0, \quad i = 0, \dots, n_p, \\ & \frac{d^{(k)}}{dx_p^{(k)}} (p_{i+1}^u(x_p) - p_i^u(x_p))|_{x_p = c_{i+1}} = 0 \\ & \qquad i = 0, \dots, n_p, \quad k = 1, 2 \\ & (p_{i+1}^l(x_p) - p_i^l(x_p))|_{x_p = c_{i+1}} = 0, \quad i = 0, \dots, n_p, \\ & \frac{d^{(k)}}{dx_p^{(k)}} (p_{i+1}^l(x_p) - p_i^l(x_p))|_{x_p = c_{i+1}} = 0 \\ & \qquad i = 0, \dots, n_p, \quad k = 1, 2 \end{aligned} \tag{16}$$

where $\mathcal{Z}_{i,j}$ is the semi-algebraic set:

$$\begin{aligned}\mathcal{Z}_{i,j} = \{(x_p, x_h, x_d, y_d) \, : x_p = x_h + x_d \\ c_i \leq x_p \leq c_{i+1}, t_j \leq x_h \leq t_{j+1} \\ x_d^2 + y_d^2 \leq r^2\}\end{aligned} \tag{17}$$

The last four constraints are new and enforce continuity of $p^u(x_p), p^l(x_p)$ and their first and second derivatives at the knots c_i. Given fixed knot positions the integral objective and continuity constraints all become linear constraints on the polynomial coefficients.

The first two constraints involve set-containment conditions. We apply the generalized $\mathcal{S}$-procedure [14] to turn these into SOS conditions involving multiplier functions $s^{\bullet}_{i,j}(x_p, x_h, x_d, y_d)$. This yields the following semidefinite program. To save space we drop the dependence on variables in the SOS conditions.

$$\begin{aligned}\min \; & \sum_{i=1}^{n_p} \int_{c_i}^{c_{i+1}} p_i^u(x) - p_i^l(x) dx \\ \text{s.t.} \; & (p_i^u - h_j^u - y_d) - s_{i,j}^{u,1}(r^2 - x_d^2 - y_d^2) \\ & -s_{i,j}^{u,2}(x_p - x_h - x_d) + s_{i,j}^{u,3}(x_p - x_h - x_d) \\ & -s_{i,j}^{u,4}(c_{j+1} - x_p) + s_{i,j}^{u,5}(c_j - x_p) \\ & -s_{i,j}^{u,6}(t_{j+1} - x_h) + s_{i,j}^{u,7}(t_j - x_h) \in \sum \\ & i = 1, \ldots, n_p \,, \quad j = 1, \ldots, n_h \\ & (-p_i^l + h_j^l + y_d) - s_{i,j}^{l,1}(r^2 - x_d^2 - y_d^2) \\ & -s_{i,j}^{l,2}(x_p - x_h - x_d) + s_{i,j}^{l,3}(x_p - x_h - x_d) \\ & -s_{i,j}^{l,4}(c_{j+1} - x_p) + s_{i,j}^{l,5}(c_j - x_p) \\ & -s_{i,j}^{l,6}(t_{j+1} - x_h) + s_{i,j}^{l,7}(t_j - x_h) \in \sum \\ & i = 1, \ldots, n_p \,, \quad j = 1, \ldots, n_h \\ & s_{i,j}^{u,m}, s_{i,j}^{l,m} \in \sum [x_p, x_h, x_d, y_h] \\ & i = 1, \ldots, n_p \,, \quad j = 1, \ldots, n_h \,, \quad m = 1, \ldots, 7 \\ & p_0^u(x_p) = a, \quad p_0^l(x_p) = a \\ & p_{n_p+1}^u(x_p) = b, \quad p_{n_p+1}^l(x_p) = b \\ & (p_{i+1}^u(x_p) - p_i^u(x_p))|_{x_p = c_{i+1}} = 0, \quad i = 0, \ldots, n_p, \\ & \frac{d^{(k)}}{dx_p^{(k)}}(p_{i+1}^u(x_p) - p_i^u(x_p))|_{x_p = c_{i+1}} = 0 \\ & i = 0, \ldots, n_p, \quad k = 1, 2 \\ & (p_{i+1}^l(x_p) - p_i^l(x_p))|_{x_p = c_{i+1}} = 0, \quad i = 0, \ldots, n_p, \\ & \frac{d^{(k)}}{dx_p^{(k)}}(p_{i+1}^l(x_p) - p_i^l(x_p))|_{x_p = c_{i+1}} = 0 \\ & i = 0, \ldots, n_p, \quad k = 1, 2\end{aligned} \tag{18}$$

Remark 2. The SOS program appears quite large as it involves enumerating all possible combinations of bounding functions $p_i^{\bullet}(x_p)$ and set functions $h_j^{\bullet}(x_h)$. However, in many instances $p_i^{\bullet}(x_p)$ will not cover any portion of $h_j^{\bullet}(x_h)$. In these cases the semi-algebraic set $\mathcal{X}_{i,j}$ is empty and the associated SOS constraints can be removed from the problem without consequence. Additionally, the free variable x_h can be eliminated from the problem by replacing it with $x_p - x_d$ and removing constraints enforcing $x_p = x_h + x_d$. Together these simplifications reduce the problem size.

IV. Optimization-Based Collision Avoidance

The SOS optimization problem provides a tractable means for finding piecewise-polynomial functions $p^u(x), p^l(x)$ that outer bound the set of points that are, at most, a distance r from the obstacle. Because these functions are twice continuously differentiable, we can use them within a nonlinear optimization problem. Specifically we can use (8) to enforce that a trajectory maintains a given distance from an obstacle. In this section we demonstrate how this can be used to navigate around non-convex obstacles. We also provide a comparison of performance with the approach detailed in [3]. In both cases we use a test case involving path planning for a Dubin's car model.

A. Path Planning for Dubin's Car

We formulate a standard trajectory problem for controlling a Dubin's car model with states (x, y, θ) and control inputs (v, ω) representing velocity and turning rate. The continuous dynamics are given by:

$$\frac{dx}{dt} = v \cos\theta, \quad \frac{dy}{dt} = v \sin\theta, \quad \frac{d\theta}{dt} = \omega \tag{19}$$

We use direct multiple shooting as the transcription method [18] with a time step $\Delta t = 0.1s$. The dynamics are then given by $x_{k+1} = f(x_k, u_k)$ where x_k and u_k denote the state and control vectors at time index k. Additionally we have constraints on the control magnitude and rate represented by $h(u_k) \geq 0$. Specifically we have:

$$-4 \leq \omega_k \leq 4, \quad 0 \leq v_k \leq 2, \quad -1 \leq \frac{(v_{k+1} - v_k)}{\Delta t} \leq 1 \tag{20}$$

Finally, we add obstacle avoidance constraints for m obstacles defined by piecewise-polynomial functions $p^u(x), p^l(x)$ using the form (8). We specify the initial state x_s and final state x_f. In all problem instances, we assume a time horizon of 10 seconds giving $N = 100$ shooting intervals. Given our initial state, final state and time are fixed, our objective is to minimize the rate of change in the control signals.

$$\begin{aligned}\min \; & \sum_{k=0}^{N} (u_{k+1} - u_k)^T (u_{k+1} - u_k) \\ \text{s.t.} \; & x_0 = x_s, \quad x_{N+1} = x_f \\ & x_{k+1} = f(x_k, u_k), \quad h(x_k) \geq 0 \\ & (y_k - p_i^u(x_k))(y_k - p_i^l(x_k)) \geq 0, \; i = 1, \ldots, m \\ & \forall \, k = 0, \ldots, N\end{aligned} \tag{21}$$

B. *Motion Planning Around Non-Convex Sets*

We demonstrate our method on a non-convex set $\mathcal{O}$ of the form (1) with $x^l = 0, x^u = 3$ and y bounded by:

$$\begin{cases} -x^2 \leq y \leq x^2, & \text{if } 0 \leq x < 1 \\ -1 \leq y \leq -x^2 + 2x, & \text{if } 1 \leq x < 2 \\ x^2 - 4x + 3 \leq y \leq 0, & \text{if } 2 \leq x \leq 3 \end{cases} \tag{22}$$

Figure 2 shows the set in red along with a disk of radius 0.2. We solve the SOS problem (18) for piecewise-polynomial functions $p^u(x), p^l(x)$ that conservatively bound $\mathcal{O} \bigoplus \mathcal{D}_{0.2}$. For both functions, we choose an equally-spaced grid of 30 knots (plus endpoints) with each interval being 6th-order polynomials. We specify the endpoints as $[-0.24, 3.24]$. Outside this interval the upper and lower bounds are constant and equal so the distance constraint is trivially satisfied (effectively inactive). Figure 2 shows the resulting functions with knot points indicated by circles. To check the quality of our fit, we numerically evaluate the Minkowski sum and plot it in gray. The result is empirically very tight with the most conservatism occurring around the endpoints where the piecewise-polynomial function must approach a constant while maintaining continuous first and second derivatives. Solving the SOS program took 0.63s with MOSEK.

We use the resulting functions to perform path planning for the Dubin's car. Starting from an initial state of $x_s = [1, 0.25, \frac{\pi}{2}]$ we solve for a trajectory to bring us to $x_f = [2, 4, \frac{\pi}{2}]$ while maintaining a distance of 0.2m from three instances of the obstacle. We initialize the decision variables for the vehicle states (x_k, y_k, ω_k) with a crude initial path obtained from Hybrid A*. Control variables are initialized to zero. Figure 1 shows the resulting trajectory. A disk of radius 0.2m is plotted around the vehicle trajectory, confirming the distance constraint is satisfied. Solving the nonlinear program took 0.80s with IPOPT.

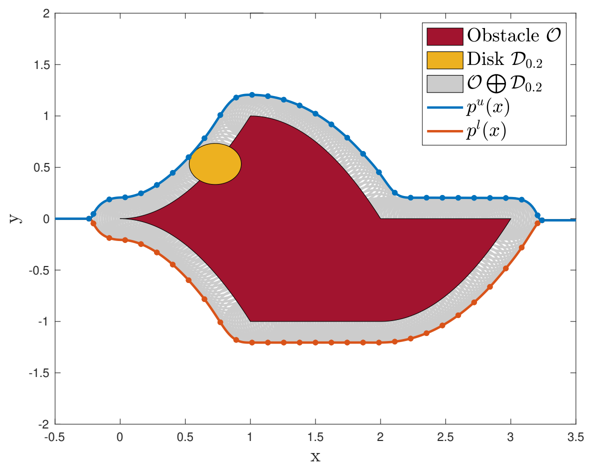

Fig. 2. Outer Approximation of Minkowski Sum of Obstacle and Disk

C. *Comparison to Exact Convex Collision Detection*

In [3] the authors present a method for exact optimization-based collision avoidance (OBCA) between a convex vehicle shape and m convex obstacles. The formulation leverages strong duality for distance computations between non-empty compact convex sets. Non-convex sets can be approximated by the union of multiple (possibly many) convex sets. The advantage of their formulation is that the distance constraints are exact compared to the conservative outer approximation presented here. Additionally, their method allows for arbitrary convex vehicle shapes while ours is presently limited to circular representations of the vehicle. The downside of their formulation is that, at each time step, new variables are introduced for each constraint used to represent an object. For complex shapes the computational complexity grows quickly. In contrast the method presented here does not introduce any new variables. A single nonlinear constraint is imposed at each time step.

Figure 3 shows a non-convex shape formed from the union of 8 convex polygons. Representing this shape in the OBCA framework requires 32 new variables at each time step in the trajectory optimization problem. We bound the Minkowski sum of this shape with a disk of radius 0.1m. The resulting piecewise-polynomial functions are shown in Figure 3. Solving the SOS problem took 1.29s with MOSEK. Empirically they are seen to be quite tight except for the end points which are chosen to be $[-1.166, 1.166]$.

We compare the performance of our piecewise-polynomial (PWP) approach to the OBCA method of [3] (OBCA) on the Dubin's path planning problem. Starting from an initial state of $x_s = [0.25, 1, 0]$ we solve for a trajectory to bring us to $x_f = [1, 2.5, 0]$ over a 10-second horizon. To test the scaleability of each method we consider test cases with 1-4 obstacles of the form shown in Figure 3. We note that the two problem formulations are identical except for the obstacle avoidance conditions. We initialize both problems identically with a crude path obtained from Hybrid A*. Dual variables in the OBCA approach are initialized to 0.1 Figure 4 shows the 4 obstacle case. In all instances the trajectories obtained by PWP and OBCA were qualitatively similar. Figure 5 provides a zoom of the trajectories. Between time indices of the transcription method (shown by circles) both trajectories violate the distance constraints. This "corner cutting" is a separate issue faced by all trajectory optimization methods which rely on discrete representations of time.

Table I lists the solve times and objective values obtained for each case. In general, the OBCA approach achieves slightly lower cost (i.e. less aggressive controls). This agrees with our general expectations as, due to the use of exact distance constraints, the OBCA problem has a slightly larger obstacle-free area to plan with. However, this comes at the cost of solver runtimes. The piecewise-polynomial approach was faster in all instances. We list the number of solver iterations performed by IPOPT. In all instances the number of iterations are relatively close, thus the speedup seen by the PWP approach can be attributed to the smaller problem

TABLE I

MOTION PLANNING WITH PIECEWISE POLYNOMIAL AND OBCA

Num. Obs.	Cost (OBCA)	Cost (PWP)	Time (s) (OBCA)	Time (s) (PWP)	Iter. (OBCA)	Iter. (PWP)
1	0.0451	0.0451	1.87	0.256	49	24
2	0.5998	0.6430	27.22	5.61	398	409
3	0.7196	0.8563	22.36	3.18	224	188
4	1.4515	0.8616	23.94	2.22	181	113

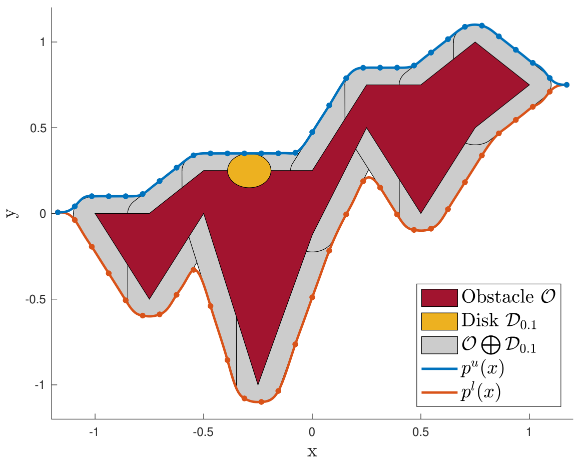

Fig. 3. Outer Approximation of Minkowski Sum of Polygon and Disk

dimension as no new variables are introduced for the obstacle avoidance constraints.

D. Implementation Details

All SOS problems were solved using MOSEK [19] in conjunction with the SOS module of YALMIP [20]. All nonlinear optimization problems were solved with IPOPT [21] with the linear solver MUMPS called via CasADi [22]. For both problem formulations reported in Table I, we expect the solve times listed can be reduced by an order of magnitude through standard approaches (e.g. compiling gradient and Hessian functions in C++ and using a tailored NLP solver such as [8]). Computations were performed on a MacBook Pro with a 2.6 GHz Intel Core i7.

V. CONCLUSIONS

In this work we developed a new approach for embedding obstacle avoidance constraints within optimization-based motion planning algorithms. This approach relied on finding tight outer approximations of semi-algebraic sets using sum-of-squares optimization. In future work we plan to perform a more rigorous comparison with competing methods across a variety of problem instances. We also plan to extend these techniques to motion planning in $\mathbb{R}^3$ and to take into account the geometry of the vehicle in addition to the obstacle.

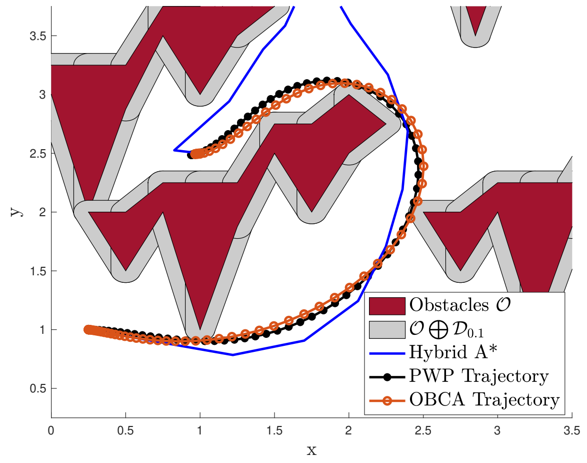

Fig. 4. Dubin's Car Navigating Four Obstacles With 0.1m Clearance

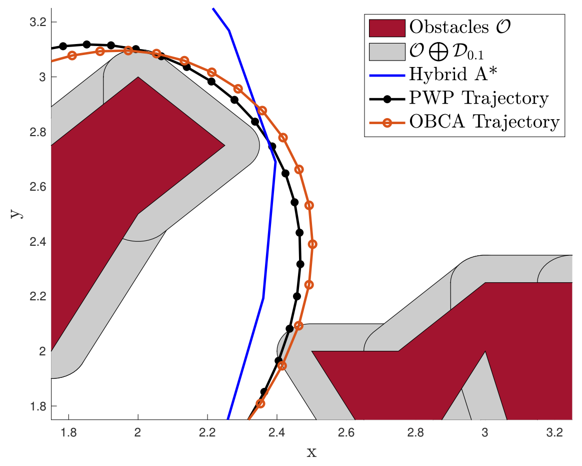

Fig. 5. Dubin's Car Navigating Four Obstacles (Zoom)

REFERENCES

[1] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," 06 2010.

[2] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *I. J. Robotic Res.*, vol. 29, pp. 485–501, 04 2010.

[3] X. Zhang, A. Liniger, and F. Borrelli, "Optimization-based collision avoidance," *IEEE Transactions on Control Systems Technology*, vol. PP, 11 2017.

[4] Z. Zhu, E. Schmerling, and M. Pavone, "A convex optimization approach to smooth trajectories for motion planning with car-like robots," *2015 54th IEEE Conference on Decision and Control (CDC)*, pp. 835–842, 2015.

[5] K. Bergman, O. Ljungqvist, and D. Axehill, "Improved path planning by tightly combining lattice-based path planning and optimal control," *IEEE Transactions on Intelligent Vehicles*, pp. 1–1, 2020.

[6] M. Brown and J. Gerdes, "Coordinating tire forces to avoid obstacles using nonlinear model predictive control," *IEEE Transactions on Intelligent Vehicles*, vol. PP, pp. 1–1, 11 2019.

[7] M. Vukov, A. Domahidi, H. J. Ferreau, M. Morari, and M. Diehl, "Auto-generated algorithms for nonlinear model predictive control on long and on short horizons," *52nd IEEE Conference on Decision and Control*, pp. 5113–5118, 2013.

[8] A. Zanelli, A. Domahidi, J. Jerez, and M. Morari, "Forces nlp: An efficient implementation of interior-point methods for multistage nonlinear nonconvex programs," *International Journal of Control*, pp. 1–26, 04 2017.

[9] R. Bonalli, A. Cauligi, A. Bylard, and M. Pavone, "Gusto: Guaranteed

sequential trajectory optimization via sequential convex programming," pp. 6741–6747, 05 2019.

[10] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for bertha — a local, continuous method," *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pp. 450–457, 2014.

[11] T. Schoels, P. Rutquist, L. Palmieri, A. Zanelli, K. O. Arras, and M. Diehl, "Ciao*: Mpc-based safe motion planning in predictable dynamic environments," 2020.

[12] M. Gerdts, R. Henrion, D. Hömberg, and C. Landry, "Path planning and collision avoidance for robots," 2012.

[13] A. Ahmadi, G. Hall, A. Makadia, and V. Sindhwani, "Geometry of 3d environments and sum of squares polynomials," 07 2017.

[14] P. Parrilo, *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.

[15] J. Guthrie and E. Mallada, "Outer approximations of minkowski operations on complex sets via sum-of-squares optimization," *Submitted to American Control Conference 2021*.

[16] D. Henrion and C. Louembet, "Convex inner approximations of nonconvex semialgebraic sets applied to fixed-order," *International Journal of Control - INT J CONTR*, vol. 85, 04 2011.

[17] F. Dabbene, D. Henrion, and C. Lagoa, "Simple approximations of semialgebraic sets and their applications to control," *Automatica*, vol. 78, 09 2015.

[18] H. Bock and K. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," pp. 242–247, 01 1984.

[19] M. ApS, *The MOSEK optimization toolbox for MATLAB manual. Version 8.1.*, 2017.

[20] J. Lofberg, "Yalmip : a toolbox for modeling and optimization in matlab," in *2004 IEEE International Conference on Robotics and Automation (IEEE Cat. No.04CH37508)*, pp. 284–289, Sep. 2004.

[21] A. Wächter and L. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, pp. 25–57, 03 2006.

[22] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, pp. 1–36, 2019.