Abstract—Motivated by applications in robotics and computer vision, we study problems related to spatial reasoning of a 3D environment using sublevel sets of polynomials. These include: tightly containing a cloud of points (e.g., representing an obstacle) with convex or nearly-convex basic semialgebraic sets, computation of Euclidean distances between two such sets, separation of two convex basic semialgebraic sets that overlap, and tight containment of the union of several basic semialgebraic sets with a single convex one. We use algebraic techniques from sum of squares optimization that reduce all these tasks to semidefinite programs of small size and present numerical experiments in realistic scenarios.

I. INTRODUCTION

A central problem in robotics, computer graphics, virtual and augmented reality (VR/AR), and many applications involving complex physics simulations is the accurate, real-time determination of proximity relationships between three-dimensional objects [4] situated in a cluttered environment. In robot navigation and manipulation tasks, path planners need to compute a dynamically feasible trajectory connecting an initial state to a goal configuration while avoiding obstacles in the environment. In VR/AR applications, a human immersed in a virtual world may wish to touch computer generated objects, that must respond to contacts in physically realistic ways. Likewise, when collisions are detected, 3D gaming engines and physics simulators (e.g., for molecular dynamics) need to activate appropriate directional forces on interacting entities. All of these applications require geometric notions of separation and penetration between representations of three-dimensional objects to be continuously monitored.

A rich class of computational geometry problems arises in this context, when 3D objects are outer approximated by convex bounding volumes. Due to convexity, the Euclidean distance between such bounding volumes can be computed very precisely, providing a reliable certificate of safety for the objects they enclose. This can also be effective for nonconvex objects which can be tightly covered by a finite union of convex shapes, e.g., using convex decomposition methods [10], [13]—the distance to such an object can be computed by taking the minimum of distances to its convex components. When objects overlap, quantitative measures of degree of penetration are needed in order to optimally resolve collisions, e.g., by a gradient-based trajectory optimizer. Multiple such measures have been proposed in the literature. The penetration depth [15] is a related concept: it is the minimum shrinkage of the two bodies required to reduce volume penetration down to merely surface touching.

A. Contributions and organization of the paper

In this paper, we propose to represent the geometry of a given 3D environment comprising multiple static or dynamic rigid bodies using sublevel sets of polynomials. The paper is organized as follows: In Section II, we provide an overview of the algebraic concepts of sum of squares (sos) and sum of squares-convex (sos-convex) polynomials as well as their relation to semidefinite programming and polynomial optimization. In Section III, we consider the problem of containing a cloud of 3D points with tight-fitting convex or nearly convex sublevel sets of polynomials. In particular, we propose a new volume minimization heuristic for these sublevel sets which empirically results in tighter fitting polynomials than previous proposals [12], [8]. The extent of convexity imposed on these sublevel set bounding volumes can be explicitly tuned using sum of squares optimization techniques. If convexity is imposed, we refer to them as sos-convex bodies; if it is not, we term them simply as sos-bodies. (See Section II for a more formal definition.)
The bounding volumes obtained are highly compact, and adapt to the shape of the data in more flexible ways than canned convex primitives typically used in standard bounding volume hierarchies. Their construction involves small-scale semidefinite programs (SDPs) that can fit, in an offline preprocessing phase, 3D meshes with tens of thousands of data points in a few seconds. In Section IV, we define notions related to measuring separation or penetration of two of these polynomial sublevel sets and show how they can be efficiently computed using sum of squares optimization. These include Euclidean distance and growth distance [15] computations. In Section V, we study the problem of containing (potentially nonconvex) polynomial sublevel sets (as opposed to points as in Section III) within one convex polynomial sublevel set. We end in Section VI with some future directions.

B. Preview of some experiments

Figure 1 gives a preview of some of the methods developed in this paper using as an example a 3D chair point cloud. On the left, we enclose the chair within the 1-sublevel set of three sos-convex polynomials with increasing degree (2, 4 and 6) leading to correspondingly tighter fits. The middle plot presents the 1-sublevel set of three degree-6 sos polynomials with increasing nonconvexity showing how tighter representations can be obtained by relaxing convexity. The right plot shows the 2, 1 and 0.75 sublevel sets of a single degree-6 sos polynomial; the 1-sublevel set colored green encloses the chair, while greater or lower values of the level set define grown and shrunk versions of the object. The computation of Euclidean distances, and sublevel-based measures of separation and penetration between such bodies is a tiny convex optimization problem that can be solved in a matter of milliseconds.

II. SUM OF SQUARES AND SOS-CONVEXITY

In this section, we briefly review the notions of sum of squares polynomials, sum of squares-convexity, and polynomial optimization which will all be central to the geometric problems we discuss later. We refer the reader to the recent monograph [7] for a more detailed overview of the subject. Throughout, we will denote the set of $n \times n$ symmetric matrices by $S_{n}^{n}$ and the set of degree-2$d$ polynomials with real coefficients by $\mathbb{R}_{2d}[x]$. We say that a polynomial $p(x_{1},\ldots,x_{n}) \in \mathbb{R}_{2d}[x]$ is nonnegative if $p(x_{1},\ldots,x_{n}) \geq 0$, $\forall x \in \mathbb{R}^{n}$. In many applications (including polynomial optimization that we will cover later), one would like to constrain certain coefficients of a polynomial so as to make it nonnegative. Unfortunately, even testing whether a given polynomial (of degree 2$d$ $\geq$ 4) is nonnegative is NP-hard. As a consequence, we would like to replace the intractable condition that $p$ be nonnegative by a sufficient condition for it that is more tractable. One such condition is for the polynomial to have a sum of squares decomposition. We say that a polynomial $p$ is a sum of squares (sos) if there exist polynomials $g_{i}$ such that $p = \sum_{i} g_{i}^{2}$. From this definition, it is clear that any sos polynomial is nonnegative, though not all nonnegative polynomials are sos; see, e.g., [18],[9] for some counterexamples. Furthermore, requiring that a polynomial $p$ be sos is a computationally tractable condition as a consequence of the following characterization: A polynomial $p$ of degree 2$d$ is sos if and only if there exists a positive semidefinite matrix $Q$ such that $p(x) = z(x)^{T}Qz(x)$, where $z(x)$ is the vector of all monomials of degree up to $d$ [16]. The matrix $Q$ is sometimes called the Gram matrix of the sos decomposition and is of size $(d+n \times d+n)$. (Throughout the paper, we let $N := (n + d)$.)

The task of finding a positive semidefinite matrix $Q$ that makes the coefficients of $p$ all equal to the coefficients of $z(x)^{T}Qz(x)$ is a semidefinite programming problem, which can be solved in polynomial time to arbitrary accuracy [20].

The concept of sum of squares can also be used to define a sufficient condition for convexity of polynomials known as sos-convexity. We say that a polynomial $p$ is sos-convex if the polynomial $y^{T} \nabla^{2}p(x)y$ in 2$n$ variables $x$ and $y$ is a sum of squares. Here, $\nabla^{2}p(x)$ denotes the Hessian of $p$, which is a symmetric matrix with polynomial entries. For a polynomial of degree 2$d$ in $n$ variables, one can check that the dimension of the Gram matrix associated to the sos-convexity condition is $N := n \cdot (n + d - 1)$. It follows from the second order characterization of convexity that any sos-convex polynomial is convex, as $y^{T} \nabla^{2}p(x)y$ being sos implies that $\nabla^{2}p(x) \succeq 0$, $\forall x$. The converse however is not true, though convex but not sos-convex polynomials are hard to find in practice; see [2]. Through its link to sum of squares, it is easy to see that testing whether a given polynomial is sos-convex is a semidefinite program. By contrast, testing whether a polynomial of degree 2$d$ $\geq$ 4 is convex is NP-hard [1].

A polynomial optimization problem is a problem of the form

$$\min_{x \in K} p(x),$$

where the objective $p$ is a (multivariate) polynomial and the feasible set $K$ is a basic semialgebraic set; i.e., a set defined by polynomial inequalities:

$$K := \{x \mid g_{i}(x) \geq 0, i = 1,\ldots,m\}.$$

It is straightforward to see that problem (1) can be equivalently formulated as that of finding the largest constant $\gamma$ such that $p(x) - \gamma \geq 0, \forall x \in K$. Under mild conditions (specifically, under the assumption that $K$ is Archimedean [9]), the condition $p(x) - \gamma \geq 0, \forall x \in K$ is equivalent to the existence of sos polynomials $\sigma_{i}(x)$ such that $p(x) - \gamma = \sigma_{0}(x) + \sum_{i=1}^{m} \sigma_{i}(x)g_{i}(x)$ [3]. Hence, problem (1) becomes

$$\max_{\gamma}$$

s.t. $p(x) - \gamma = \sigma_{0} + \sum_{i=1}^{m} \sigma_{i}(x)g_{i}(x),$

$$\sigma_{i} \sos, i = 0,\ldots,m.$$
bound on the optimal value of (1). As the degrees of \( \sigma_i \) increase, these lower bounds are guaranteed to converge to the true optimal value of (1). (Note that we are making no convexity assumptions about the polynomial optimization problem and yet solving it globally through a sequence of semidefinite programs.)

III. 3D POINT CLOUD CONTAINMENT

Throughout this section, we are interested in finding a body of minimum volume, parametrized as the 1-sublevel set of a polynomial of degree 2\( d \), which encloses a set of given points \( \{x_1, \ldots, x_m\} \) in \( \mathbb{R}^n \).

A. Convex sublevel sets

We focus first on finding a convex bounding volume. Convexity is a common constraint in the bounding volume literature and it makes certain tasks (e.g., distance computation among the different bodies) simpler. In order to make a set of the form \( \{ x \in \mathbb{R}^d \mid p(x) \leq 1 \} \) convex, we will require the polynomial \( p \) to be convex. (Note that this is a sufficient but not necessary condition.) Furthermore, to have a tractable formulation, we will replace the convexity condition with an sos-convexity condition as described previously. Even after these relaxations, the problem of minimizing the volume of our sublevel sets remains a difficult one. The remainder of this section discusses several heuristics for this task.

1) The Hessian-based approach: In [12], Magnani et al. propose the following heuristic to minimize the volume of the 1-sublevel set of an sos-convex polynomial

\[
\min_{p \in \mathbb{R}^{2^d}[x], H \in S^{N \times N}} - \log \det(H)
\]

s.t.

\[
p \text{ sos,} \\
y^T \nabla^2 p(x)y = w(x,y)^T H w(x,y), \ H \succeq 0, \ (3)
\]

\[
p(x_i) \leq 1, i = 1, \ldots, m,
\]

where \( w(x,y) \) is a vector of monomials in \( x \) and \( y \), of degree 1 in \( y \) and \( d - 1 \) in \( x \). This problem outputs a polynomial \( p \) whose 1-sublevel set corresponds to the bounding volume that we are interested in. A few remarks on this formulation are in order:

- The last constraint simply ensures that all the data points are within the 1-sublevel set of \( p \) as required.
- The second constraint imposes that \( p \) be sos-convex. The matrix \( H \) is the Gram matrix associated with the sos condition on \( y^T \nabla^2 p(x)y \).
- The first constraint requires that the polynomial \( p \) we are looking for be sos. This is a necessary condition for boundedness of (3) when \( p \) is parametrized with affine terms. To see this, note that for any given positive semidefinite matrix \( Q \), one can always pick the coefficients of the affine terms in such a way that the constraint \( p(x_i) \leq 1 \) for \( i = 1, \ldots, m \) be trivially satisfied. Likewise one can pick the remaining coefficients of \( p \) in such a way that the sos-convexity condition be satisfied. The restriction to sos polynomials, however, can be done without loss of generality. Indeed, suppose that the minimum volume sublevel set was given by \( \{ x \mid p(x) \leq 1 \} \) where \( p \) is an sos-convex polynomial. As \( p \) is convex and nonaffine, \( \exists \gamma \geq 0 \) such that \( p(x) + \gamma \geq 0 \) for all \( x \). Define now \( q(x) := \frac{p(x)+\gamma}{1+\gamma} \). We have that \( \{ x \mid p(x) \leq 1 \} = \{ x \mid q(x) \leq 1 \} \), but here, \( q \) is sos as it is sos-convex and nonnegative [5, Lemma 8].

The objective function of the above formulation is motivated in part by the degree \( 2d = 2 \) case. Indeed, when \( 2d = 2 \), the sublevel sets of convex polynomials are ellipsoids of the form \( \{ x \mid x^T P x \leq 1 \} \) and their volume is given by \( \frac{4}{3} \pi \cdot \sqrt{\det(P^{-1})} \). Hence, by minimizing \( -\log \det(P) \), we would exactly minimize volume. Furthermore, the matrix \( P \) associated with the form \( x^T P x \) is none other than the Hessian of the form (up to a multiplicative constant).

A related minimum volume heuristic that we will also experiment with corresponds to the following problem:

\[
\min_{p \in \mathbb{R}^{2^d}[x], H \in S^{N \times N}, V \in S^{N \times N}} \text{trace}(V)
\]

s.t.

\[
p \text{ sos,} \\
y^T \nabla^2 p(x)y = w(x,y)^T H w(x,y), H \succeq 0, \\
p(x_i) \leq 1, i = 1, \ldots, m,
\]

\[
[V \ I \ I \ H] \succeq 0.
\]

The last constraint can be rewritten using the Schur complement as \( V \succeq H^{-1} \). As a consequence, this trace formulation minimizes the sum of the inverse of the eigenvalues of \( H \) whereas the log det formulation described in (3) minimizes the product of the inverse of the eigenvalues.

2) Our approach: We propose here yet another heuristic for obtaining a tight-fitting convex body containing points in \( \mathbb{R}^n \). Empirically, we validate that it tends to consistently return convex bodies of smaller volume than the ones obtained with the methods described above. It also generates a relatively smaller convex optimization problem. Our formulation is as follows:

\[
\min_{p \in \mathbb{R}^{2^d}[x], P \in S^{N \times N}} -\log \det(P)
\]

s.t.

\[
p(x) = z(x)^T P z(x), P \succeq 0, \\
p \text{ sos-convex,} \\
p(x_i) \leq 1, i = 1, \ldots, m.
\]

One can also obtain a trace formulation of this problem by replacing the log det objective by a trace one as it was done
to go from (3) to (4):
\[
\begin{align*}
\min_{p \in \mathbb{R}^{2d}[x], P \in S^{N \times N}, V \in S^{N \times N}} & \quad \text{trace}(V) \\
\text{s.t.} & \\
& p(x) = z(x)^T P z(x), P \succeq 0, \\
& p \text{ sos-convex}, \\
& p(x_i) \leq 1, i = 1, \ldots, m, \\
& \begin{bmatrix} V & I \\ I & P \end{bmatrix} \succeq 0.
\end{align*}
\]

Note that the main difference between (3) and (5) lies in the Gram matrix chosen for the objective function. In (3), the Gram matrix comes from the sos-convexity constraint, whereas in (5), the Gram matrix is generated by the sos constraint.

In the case where the polynomial is quadratic and convex, we saw that the formulation (3) is exact as it finds the minimum volume ellipsoid containing the points. It so happens that the formulation given in (5) is also exact in the quadratic case, and, in fact, both formulations return the same optimal ellipsoid. As a consequence, the formulation given in (5) can also be viewed as a natural extension of the quadratic case.

To provide more intuition as to why this formulation performs well, we interpret the 1-sublevel set
\[ S := \{ x \mid p(x) \leq 1 \} \]
as the projection of some set whose volume is being minimized. More precisely, as \( p(x) = z(x)^T P z(x) \), then the set \( S \) can be written as
\[ S = \{ x \in \mathbb{R}^n \mid \exists v \in \mathbb{R}^{N-n} \text{ s.t. } (x, v) \in T \} \]

where
\[ T = \{ (x, v) \in \mathbb{R}^N \mid (x, v)^T P(x, v) \leq 1, z(x) = (x, v) \} \]

Note that the function \( z \) takes as input \( x \), i.e., the first \( n \) components of \( (x, v) \), and maps it to the monomial vector \( z(x) \). In other words, \( z \) defines a set of polynomial equalities in \( (x, v) \) (of the type, e.g., \( v_1 = x_1^2 \)). In this way, it is easy to see that \( T \) is not an ellipsoid itself though it is contained in the ellipsoid \( \{ (x, v) \mid (x, v)^T P(x, v) \leq 1 \} \). When we minimize \( \log \det P \), we are in fact minimizing the volume of this latter ellipsoid. Hence, we are indirectly minimizing the volume of \( T \). As \( S \) is a projection of \( T \), this is a heuristic for minimizing the volume of \( S \).

B. Relaxing convexity

Though containing a set of points with a convex sublevel set has its advantages, it is sometimes necessary to have a tighter fit than the one provided by a convex body, particularly if the object of interest is highly nonconvex. One way of handling such scenarios is via convex decomposition methods [10], [13], which would enable us to represent the object as a tight union of sos-convex bodies. Alternatively, one can aim for problem formulations where convexity of the sublevel sets is not imposed. In the remainder of this subsection, we first review a recent approach from the literature to do this and then present our own approach which allows for controlling the level of nonconvexity of the sublevel set.

1) The inverse moment approach: In very recent work [8], Lasserre and Pauwels propose an approach for containing a cloud of points with sublevel sets of polynomials (with no convexity constraint). Given a set of data points \( x_1, \ldots, x_m \in \mathbb{R}^n \), it is observed in that paper that the sublevel sets of the degree \( 2d \) sos polynomial
\[ p_{\mu,d}(x) := z(x)^T M_d(\mu(x_1, \ldots, x_m))^{-1} z(x), \]
tend to take the shape of the data accurately. Here, \( z(x) \) is the vector of all monomials of degree up to \( d \) and \( M_d(\mu(x_1, \ldots, x_m)) \) is the moment matrix of degree \( d \) associated with the empirical measure \( \mu := \frac{1}{m} \sum_{x=1}^{m} \delta_{x} \) defined over the data. This is an \( (n+d) \times (n+d) \) symmetric positive semidefinite matrix which can be cheaply constructed from the data \( x_1, \ldots, x_m \in \mathbb{R}^n \) (see [8] for details). One very nice feature of this method is that to construct the polynomial \( p_{\mu,d} \) in (7) one only needs to invert a matrix (as opposed to solving a semidefinite program as our approach would require) after a single pass over the point cloud. The approach however does not a priori provide a particular sublevel set of \( p_{\mu,d} \) that is guaranteed to contain all data points. Hence, once \( p_{\mu,d} \) is constructed, one could slowly increase the value of a scalar \( \gamma \) and check whether the \( \gamma \)-sublevel set of \( p_{\mu,d} \) contains all points.

2) Our approach and controlling convexity: An advantage of our proposed formulation (5) is that one can easily drop the sos-convexity assumption in the constraints and thereby obtain a sublevel set which is not necessarily convex. The problem then becomes:
\[
\begin{align*}
\min_{p \in \mathbb{R}^{2d}[x], P \in S^{N \times N}} & \quad -\log \det(P) \\
\text{s.t.} & \\
& p = z(x)^T P z(x), P \succeq 0 \\
& p(x_i) \leq 1, i = 1, \ldots, m.
\end{align*}
\]

This is not an option for formulation (3) as the Gram matrix associated to the sos-convexity constraint intervenes in the objective.

Note that in neither this formulation nor the inverse moment approach of Lasserre and Pauwels, does the optimizer have control over the shape of the sublevel sets produced, which may be convex or far from convex. For some applications, it is useful to control in some way the degree of convexity of the sublevel sets obtained by introducing a parameter which when increased or decreased would make the sets more or less convex. This is what our following proposed optimization problem does via the parameter \( c \),
which corresponds in some sense to a measure of convexity:
\[
\min_{p \in \mathbb{R}^{2d \times 1}, P \in \mathbb{S}^{N \times N}} - \log \det(P)
\]
s.t.
\[
p = z(x)^T P z(x), P \succeq 0
\]
\[
p(x) - c \left( \sum_i x_i^2 \right)^d \text{sos-convex.}
\]
\[
p(x_i) \leq 1, i = 1, \ldots, m.
\]
Note that when \(c = 0\), the problem we are solving corresponds exactly to (8) and the sublevel set obtained is convex. When \(c < 0\), we allow for nonconvexity of the sublevel sets. As we increase \(c\) however, we obtain sublevel sets which get progressively more and more convex.

C. Bounding volume numerical experiments

Figure 1 (left) shows the 1-sublevel sets of sos-convex bodies with degrees 2, 4 and 6. A degree 6 polynomial gives a much tighter fit than an ellipsoid (degree 2). In the middle figure, we freeze the degree to be 6 and reduce the convexity parameter \(c\) in the relaxed convexity formulation of equation (9); the 1-sublevel sets of the resulting sos polynomials with \(c = 0, -10, -100\) are shown. It can be seen that the sublevel sets gradually bend to better adapt to the shape of the object. The right figure shows the 2, 1, 0.75 sublevel sets of degree 6 polynomial with \(c = -10\): the shape is retained as the body is expanded or contracted.

In Table 1, we provide a comparison of various bounding volumes on Princeton Shape Benchmark datasets [19]. It can be seen that sos-convex bodies with higher degree polynomials provide much tighter fits than spheres or axis-aligned bounding boxes (AABB) in general. The proposed minimum volume heuristic of our formulation in (5) or (6) works better than that proposed in [12] (see (3) or (4)). In both formulations, typically, the log-determinant criteria outperforms the trace criteria. The convex hull is the tightest possible convex body. However, for smooth objects like the vase, the number of extreme vertices describing the convex hull can be a substantial fraction of the original number of points in the point cloud. (The number of vertices of the convex hull is written in parentheses in the Convex-Hull row of the table.) When convexity is relaxed, a degree 6 sos polynomial compactly described by just 84 coefficients gives a tighter fit than the convex hull. For the same degree, solutions to our formulation (9) with a negative value of \(c\) outperform the inverse moment construction of [8].

The bounding volume construction time is shown in Figure 3 for sos-convex chair models. In comparison to the volume heuristics of [12], our heuristic runs significantly faster as soon as degree exceeds 6 since our formulation leads to smaller SDPs. Our implementation uses YALMIP [11]\(^2\) with the splitting conic solver (SCS) [14]\(^3\) as its backend SDP solver (run for 2500 iterations). Note that the inverse moment approach of [8] is the fastest as it does not involve any optimization and makes just one pass over the point cloud. However, this approach is not guaranteed to return a convex body, and for nonconvex bodies, tighter fitting polynomials can be estimated using log-determinant or trace optimization as described in (9).

---

3. https://github.com/cvxgrp/scs

---

IV. MEASURES OF SEPARATION AND PENETRATION

A. Euclidean Distance

In this section, we are interested in computing the Euclidean distance between two sets

\[
S_1 = \{ x \in \mathbb{R}^n | g_1(x) \leq 1, \ldots, g_m \leq 1 \},
\]

and

\[
S_2 = \{ x \in \mathbb{R}^n | h_1(x) \leq 1, \ldots, h_r \leq 1 \},
\]

where \(g_1, \ldots, g_m\) and \(h_1, \ldots, h_r\) are all sos-convex. This problem can formally be written as follows

\[
\min_{x \in S_1, y \in S_2} ||x - y||^2_2.
\]

This is a constrained convex optimization problem which can be solved using generic techniques such as interior point methods. It also turns out that viewed as a polynomial optimization problem, this problem can be solved exactly using semidefinite programming. This is a corollary of the following more general result due to Lasserre [6] to which our problem conforms.

**Theorem 4.1 (6):** Consider the polynomial optimization problem

\[
\min_{x} p_0(x)
\]

s.t. \(p_1(x) \leq 0, \ldots, p_s(x) \leq 0\)

where \(p_0, \ldots, p_s\) are all sos-convex. Then the optimal value of this problem is the same as the optimal value of the following SDP:

\[
\max_{\gamma \in \mathbb{R}, \lambda \in \mathbb{R}^r, \sigma_0 \in \mathbb{R}^{2d \times 1}} \gamma
\]

\[
p_0(x) - \gamma = \sigma_0(x) + \sum_{i=1}^{s} \lambda_i p_i(x),
\]

\[
\sigma_0(x) \text{sos.}
\]
In this situation, the first level of the sos hierarchy described at the end of Section II is exact. Even in the case where the sets $S_1$ and $S_2$ are not convex (which would mean in particular that their defining polynomials $g_i$ and $h_i$ are not all convex), we can obtain increasingly accurate lower bounds converging to the exact Euclidean distance between the sets. This is done by applying higher levels of the sos hierarchy. The points achieving the minimum distance between degree 6 sos-convex minimum volume (log-det) bodies enclosing human and chair 3D point clouds are shown below.

In this situation, the first level of the sos hierarchy described at the end of Section II is exact. Even in the case where the sets $S_1$ and $S_2$ are not convex (which would mean in particular that their defining polynomials $g_i$ and $h_i$ are not all convex), we can obtain increasingly accurate lower bounds converging to the exact Euclidean distance between the sets. This is done by applying higher levels of the sos hierarchy. The points achieving the minimum distance between degree 6 sos-convex minimum volume (log-det) bodies enclosing human and chair 3D point clouds are shown below.

B. Penetration measures for overlapping bodies

As another application of sos-convex polynomial optimization problems, we discuss a problem relevant to collision avoidance. Here, we assume that our two bodies $S_1$, $S_2$ are of the form $S_1 := \{x \mid p_1(x) \leq 1\}$ and $S_2 := \{x \mid p_2(x) \leq 1\}$ where $p_1, p_2$ are sos-convex. As shown in Figure 1 (right), by varying the sublevel value, we can grow or shrink the sos representation of an object. The following convex optimization problem, with optimal value denoted by $d(p_1\|p_2)$, provides a measure of separation or penetration between the two bodies:

\[
d(p_1\|p_2) = \min p_1(x) \\
\text{s.t. } p_2(x) \leq 1.
\]  

(11)

Note that the measure is asymmetric, i.e., $d(p_1\|p_2) \neq d(p_2\|p_1)$. It is clear that

\[
p_2(x) \leq 1 \Rightarrow p_1(x) \geq d(p_1\|p_2).
\]

In other words, the sets $\{x \mid p_2(x) \leq 1\}$ and $\{x \mid p_1(x) \leq d(p_1\|p_2)\}$ do not overlap. As a consequence, the optimal value of (11) gives us a measure of how much we need to shrink the level set defined by $p_1$ to eventually move out of contact of the set $S_2$ assuming that the “seed point”, i.e., the minimum of $p_1$, is outside $S_2$. It is clear that

- if $d(p_1\|p_2) > 1$, the bounding volumes are separated.
- if $d(p_1\|p_2) = 1$, the bounding volumes touch.
- if $d(p_1\|p_2) < 1$, the bounding volumes overlap.

These measures are closely related to the notion of growth models and growth distances [15]. Note that as a consequence of Theorem 4.1, the optimal solution $d(p_1\|p_2)$ to (11) can be computed exactly using semidefinite programming, or using a generic convex optimizer. The two leftmost

<table>
<thead>
<tr>
<th>Object → Bounding Body ↓</th>
<th>id:#vertices</th>
<th>Human 10/950889</th>
<th>Chair 101/8409</th>
<th>Hand 181/7242</th>
<th>Vase 361/14859</th>
<th>Octopus 121/5944</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convex-Hull Sphere AABB</td>
<td></td>
<td>0.29 (364)</td>
<td>0.66 (320)</td>
<td>0.36 (652)</td>
<td>0.91 (1443)</td>
<td>0.5 (414)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.74</td>
<td>3.73</td>
<td>3.84</td>
<td>3.91</td>
<td>4.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.59</td>
<td>1.0</td>
<td>0.81</td>
<td>1.73</td>
<td>1.28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.38</td>
<td>1.79</td>
<td>0.82</td>
<td>1.16</td>
<td>1.30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.97</td>
<td>1.80</td>
<td>1.40</td>
<td>1.2</td>
<td>1.76</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.57</td>
<td>13.55</td>
<td>0.69</td>
<td>1.13</td>
<td>1.04</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.44</td>
<td>1.19</td>
<td>0.53</td>
<td>1.05</td>
<td>0.86</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.57</td>
<td>1.25</td>
<td>0.92</td>
<td>1.09</td>
<td>1.02</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.57</td>
<td>1.27</td>
<td>0.58</td>
<td>1.09</td>
<td>0.93</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.41</td>
<td>1.02</td>
<td>0.45</td>
<td>0.99</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.45</td>
<td>1.21</td>
<td>0.48</td>
<td>1.03</td>
<td>0.79</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.42</td>
<td>0.39</td>
<td>0.39</td>
<td>1.05</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.42</td>
<td>0.74</td>
<td>0.46</td>
<td>1.10</td>
<td>0.54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.21</td>
<td>0.21</td>
<td>0.26</td>
<td>0.82</td>
<td>0.28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.22</td>
<td>0.30</td>
<td>0.29</td>
<td>0.85</td>
<td>0.37</td>
</tr>
</tbody>
</table>

**TABLE I**: Comparison of various bounding volume techniques

The distance query time, reported in the table below, ranges from around 80 milliseconds to 340 milliseconds seconds as the degree is increased from 2 to 8, using MATLAB’s fmincon active-set solver. We believe that the execution time can be improved by an order of magnitude with more efficient polynomial representations, warm starts for repeated queries and reduced convergence tolerance for lower-precision results.

<table>
<thead>
<tr>
<th>degree</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>time (secs)</td>
<td>0.08</td>
<td>0.083</td>
<td>0.13</td>
<td>0.34</td>
</tr>
</tbody>
</table>
subfigures of Figure 4 show a chair and a human bounded by 1-sublevel sets of degree 6 sos-convex polynomials (in green). In both cases, we compute $d(p_1||p_2)$ and $d(p_2||p_1)$ and plot the corresponding minimizers. In the first subfigure, the level set of the chair needs to grow in order to touch the human and vice-versa, certifying separation. In the second subfigure, we translate the chair across the volume occupied by the human so that they overlap. In this case, the level sets need to contract. In the third subfigure, we plot the optimal value of the problem in (11) as the chair is translated from left to right, showing how the growth distances dip upon penetration and rise upon separation. The final subfigure shows the time taken to solve (11) when warm started from the previous solution. The time taken is of the order of 150 milliseconds without warm starts to 10 milliseconds with warm starts.

C. Separation and penetration under rigid body motion
Suppose $\{x \mid p(x) \leq 1\}$ is a minimum-volume sos-convex body enclosing a rigid 3D object. If the object is rotated by $R \in SO(3)$ and translated by $t \in \mathbb{R}^3$, then the polynomial $p'(x) = p(R^T x - R^T t)$ encloses the transformed object. This is because, if $p(x) \leq 1$, then $p'(R x + t) \leq 1$. For continuous motion, the optimization for Euclidean distance or sublevel-based separation/penetration distances can be warm started from the previous solution. The computation of the gradient of these measures using parametric convex optimization, and exploring the potential of this idea for motion planning is left for future work.

V. Containment of Polynomial Sublevel Sets
In this section, we show how the sum of squares machinery can be used in a straightforward manner to contain polynomial sublevel sets (as opposed to point clouds) with a convex polynomial level set. More specifically, we are interested in the following problem: Given a basic semialgebraic set

$$
S := \{x \in \mathbb{R}^n \mid g_1(x) \leq 1, \ldots, g_m(x) \leq 1\},
$$

find a convex polynomial $p$ of degree $2d$ such that

$$
S \subseteq \{x \in \mathbb{R}^n \mid p(x) \leq 1\}.
$$

Moreover, we typically want the unit sublevel set of $p$ to have small volume. Note that if we could address this question, then we could also handle a scenario where the unit sublevel set of $p$ is required to contain the union of several basic semialgebraic sets (simply by containing each set separately).

For the 3D geometric problems under our consideration, we have two applications of this task in mind:

- **Convexification:** In some scenarios, one may have a nonconvex outer approximation of an obstacle (e.g., obtained by the computationally inexpensive inverse moment approach of Lasserre and Pauwels as described in Section III-B) and be interested in containing it with a convex set. This would e.g. make the problem of computing distances among obstacles more tractable; cf. Section IV.

- **Grouping multiple obstacles:** For various navigational tasks involving autonomous agents, one may want to have a mapping of the obstacles in the environment in varying levels of resolution. A relevant problem here is therefore to group obstacles. In our setting, this would lead to the problem of containing several polynomial sublevel sets with one.

In order to solve the problem laid out above, we propose the following sos program:

$$
\begin{align*}
\min_{p \in \mathbb{R}_{2d}[x], \tau_i \in \mathbb{R}_{2d}[x], p \in \mathbb{S}^{N \times N}} & \quad -\log \det(P) \\
\text{s.t.} & \quad p(x) = z(x)^T P z(x), P \succeq 0, \\
& \quad p(x) \quad \text{sos-convex}, \\
& \quad 1 - p(x) - \sum_{i=1}^m \tau_i(x)(1 - g_i(x)) \quad \text{sos}, \\
& \quad \tau_i(x) \quad \text{sos}, \quad i = 1, \ldots, m.
\end{align*}
$$

It is straightforward to see that constraints (15) and (16) imply (and algebraically certify) the required set containment criterion in (13). As usual, the constraint in (14) ensures convexity of the unit sublevel set of $p$. The objective function attempts to minimize the volume of this set. A natural choice for the degree $2d$ of the polynomials $\tau_i$ is $2d = 2d - \min_i \deg(g_i)$, though better results can be obtained by increasing this parameter.

**Example.** In Figure 5, we have drawn in black three random ellipsoids and a degree-4 convex polynomial sublevel set (in
yellow) containing the ellipsoids. This degree-4 polynomial was the output of the optimization problem described above where the sos multipliers \( \tau_i(x) \) were chosen to have degree 2.

Fig. 5: Containment of 3 ellipsoids using a sublevel set of a convex degree-4 polynomial

We end by noting that the formulation proposed here is backed up theoretically by the following converse result.

**Theorem 5.1:** Suppose the set \( S \) in (12) is Archimedean\(^4\) and that \( S = \{ x \in \mathbb{R}^n | p(x) \leq 1 \} \). Then there exists an integer \( d \) and sum of squares polynomials \( \tau_1, \ldots, \tau_m \) of degree at most \( d \) such that

\[
1 - p(x) - \sum_{i=1}^{m} \tau_i(x)(1 - g_i(x))
\]

(17)

is a sum of squares.

**Proof:** The proof follows from a standard application of Putinar’s Positivstellensatz [17] and is omitted. \( \blacksquare \)

### VI. CONCLUSION

In this paper we have shown that sos polynomials offer viable bounding volume alternatives for 3D environments. The experiments on realistic 3D datasets indicate that these polynomials satisfy the core practical functions of bounding volumes (e.g., fast construction, tightness of fit, real-time proximity evaluations). We have also proposed new formulations that improve upon previous ones (e.g., in the sense that they provide tighter fitting bounding volumes and can control the extent of nonconvexity of these volumes).

Our results open multiple application areas for future work.

- Bounding volumes are ubiquitous in rendering applications, e.g., object culling and ray tracing. When considering sos polynomial bounding volumes it is not difficult to see how these ray-surface intersection operations could be framed as convex optimization problems similar to the distance calculations in sec IV. It would be interesting to explore how such techniques would perform when integrated within GPU-optimized rendering and game engine frameworks.

### ACKNOWLEDGEMENTS

We thank Erwin Coumans, Mrinal Kalakrishnan and Vincent Vanhoucke for several technically insightful discussions and guidance.

### REFERENCES


