Spherical-Object Representation and Fast Distance Computation for Robotic Applications

J. Tornero

Departamento de Ingeniería de Sistemas, Computadores y Automática (DISCA)
Universidad Politécnica de Valencia
P.O.B. 22012, E-46071 Valencia, SPAIN.

Abstract

A three-dimensional object representation technique for generating spherical-geometries and a fast procedure for computing distances using this geometry is presented. An object is approximated by an infinite number of spheres. The shortest distance between two objects is obtained by finding the two spheres, one from each object, that are closest. Exceptional numerical results have been obtained, for example, the maximum time for computing self-collision for a standard PUMA robot-arm is equal to 2.30 milliseconds with an error in distance of less than 1cm. This makes the new technique an invaluable tool for computing distances and therefore permits collision-detection in real-time. This technique has been applied to a complex robotic system consisting of two PUMA robots, each mounted on a three-degree of freedom platform used at the CIRSSE to study robotic assembly of structures in space.

1 Introduction

The fast computation of distances is an important part of many robotic applications. Distances between the elements of a robot manipulator, between elements of cooperating robot arms and between the robotic elements and objects in the environment may all need to be computed to guarantee collision free payload transportation and manipulation. Fast distance computation is an asset to trajectory planning and manipulation planning and is essential for the on-time implementation of collision detection and obstacle avoidance strategies. In the context of vehicular navigation in both 2-D and 3-D environments, fast distance computations are also important. Potential fields [1] and other similar functions find use in trajectory planning and can similarly benefit from fast distance computation. Several optimal-control techniques are based on a good description of distances between objects. Although these applications are not explicitly addressed in this paper, it is clear that the approach presented here is appropriate whenever distance computations have to be fast.

Since many elements of robotic systems are bounded by plane surfaces, polyhedra have been used to develop procedures for computing distances [2]. However, the large number of comparisons required for such elements is highly time consuming. Thus, some authors have recast the problem into a standard linear programming form. Sometimes, norm 1 and norm ∞ rather than Euclidean distances are used to save computational effort [3]. Objects of revolution have been used to obtain faster algorithms. However, this objects have the disadvantage of combining curvilinear side surfaces with plane end surfaces. Replacing the end planes with hemispheres is one way to speed up the algorithm [4].

This paper presents a modeling technique which is conceptually between the bubble model of O'Rourke and Badler [5] and the generalized cylinders of Agin [6]. The model is derived from the basic idea that an object can be approximated by an infinite number of spheres. Hence it retains the advantages of the bubble model and avoids the complexity of the generalized cylinder. Using spheres, the shortest distance between two objects can be obtained by finding the two spheres, one from each object, that are closest. Given these spheres, the computation of the shortest distance is trivial. Further, the intrinsic symmetry of the spheres eliminates the need to consider orientation.

The thrust of this paper is to first describe an object
representation technique based on spherical-volumes and second to develop a fast procedure for computing distances between these volumes. The good numerical results obtained validate thoroughly the theory as well as makes this technique a valuable tool for distance computation and collision detection in real-time. An existing robot system has been used to show how this technique may be used for complex systems.

2 Object Representation

The modeling technique is based on the idea that any real object can be approximated by an infinite number of spheres. For a particular object many different sets of spheres can be generated, depending on the degree of accuracy required. In general accuracy is directly related to the complexity of the process for obtaining the set of spheres. Later on, we will introduce the idea of degree-of-freedom as a first measurement of this complexity.

The way of generating a set of spheres for a given object is provided by introducing the concept of dynamic-spheres. A dynamic-sphere is defined as a sphere whose center \( P(x,y,z) \) can move in a three-dimensional space and whose radius is a function of the position at each moment \( R(x,y,z) \).

The object model corresponds to the volume swept by the dynamic-sphere when moving in a bounded subspace. This subspace can be defined by

1. A set of inequality constraints, e.g., \( x_0 \leq x \leq x_1 \).
2. A set of functional constraints in the form \( f_i(x,y,z) = 0 \).

The number of functional constraints will determine the degrees of freedom for the center of the dynamic-sphere. For example, a two-degree of freedom geometry is obtained when one functional constraint is introduced (e.g., \( f_1(x,y,z) = 0 \)). In parametric representation, that is

\[
P = P(\lambda), \quad R = R(\lambda)
\]

The complexity of the volume swept by the dynamic-sphere depends on the functions \( P(\lambda_1) \) and \( R(\lambda_2) \), as well as the range of the \( \lambda \)'s. To make the problem as standard as possible, \( \lambda_1 \) will be fixed, \( \lambda_2 \in [0,1] \), \( \forall i \).

The simplest class of objects for the one-degree of freedom geometry is obtained by considering linear functions. We define a spherical-cone as the volume swept by a dynamic-sphere whose center and radius obey linear functions in the parameter \( \lambda \), that is

\[
P = P(\lambda_1, \lambda_2), \quad R = R(\lambda_1, \lambda_2)
\]

Particular cases of this geometry for \( \lambda \) varying within a given range, such as spherical-cylinders, etc., can be seen in Figure 1.

For a spherical-cone \( \text{i} \), functions \( P(\lambda_1) \) and \( R(\lambda_1) \), can be written in terms of the radius and center of the end-spheres at the two extremes of the volume, \((P_{i0}, R_{i0})\) and \((P_{i1}, R_{i1})\), in the following way,

\[
\begin{align*}
P_i &= P_{i0} + \lambda_i (P_{i1} - P_{i0}) \quad & R_i &= R_{i0} + \lambda_i (R_{i1} - R_{i0}) \\
\lambda_i &\in [0,1]
\end{align*}
\]

The degree and the angle of convergence of a spherical-cone can be defined respectively by

\[
\eta_i = \frac{(R_{i1} - R_{i0})}{|P_{i1} - P_{i0}|} \quad \alpha_i = \arcsin \eta_i
\]

For values of degree of convergence equal or greater than 1, \( \eta \geq 1 \), the volumes degenerate into a simple sphere equal to the largest end-sphere.

For the two-degree of freedom case, the simplest geometry which can be represented is a dynamic-sphere moving tangentially between two planes, which is called as a spherical-plane. In a similar way as a spherical-cone is bounded by two end-spheres, a spherical-plane is bounded by four end-spherical-cones which constitute the sides of the plane. A volume \( \text{i} \) of this kind can be described by two spherical-cones with a common end-sphere, \([P_{i0}, R_{i0}],[P_{i1}, R_{i1}]\) and \([P_{i0}, R_{i0}],[P_{i2}, R_{i2}]\), which constitute two continuous sides. The volume is then generated from a dynamic-sphere which follows the following equations

\[
\begin{align*}
P_i &= P_{i0} + \lambda_{i1}(P_{i1} - P_{i0}) + \lambda_{i2}(P_{i2} - P_{i0}) \\
R_i &= R_{i0} + \lambda_{i1}(R_{i1} - R_{i0}) + \lambda_{i2}(R_{i2} - R_{i0}) \\
\lambda_{i1}, \lambda_{i2} &\in [0,1]
\end{align*}
\]
According to Equations 5, the center of the dynamic-sphere moves inside a parallelogram, which can be easily cut-off by lines, defined as inequality constraints, in the form of linear combination of parameters λ’s,

\[ a_1 \cdot \lambda_1 + a_2 \cdot \lambda_2 \leq 1 \]  

(6)

A more general, but still simple, two degree of freedom structure can be obtained from two opposite end-spherical-cones of the plane, \([P_{10}, R_{10}], (P_{11}, R_{11})\) and \([P_{12}, R_{12}], (P_{13}, R_{13})\). In this case, the geometrical locus of the center of the dynamic-sphere is not forced to be a parallelogram. However, several relations between the two spherical-cones must be stated in order belong to the same spherical-plane.

Several kinds of spherical-planes can be obtained with this two-degree of freedom geometry as depicted in figure 2. Spherical planes are bounded at least by three spherical-edges which are in the class of spherical-cones or one-degree of freedom geometries. These spherical-edges intersect in spherical-vertices which corresponds to the zero-degree of freedom geometries.

Dynamic-spheres with three degrees of freedom are not considered in this paper, given that the most outstanding advantage of this object representation technique comes from the possibility of generating three-dimensional models with less than three degrees of freedom. However, the basic idea can be easily extended to dynamic-spheres moving in three or even more degrees of freedom. Extra degrees of freedom can be introduced in order to increase the accuracy of the representation or to consider aspects such as the time-factor for describing movements of objects in the space.

3 Procedure for Fast Distance Computation

The problem of finding the shortest distance between two bodies whose volumes have been generated by two dynamic-spheres, \((P_1, R_1)\) and \((P_2, R_2)\), can be stated as a Min-Max problem as follows,

\[ \min_{\lambda, \lambda'} \{ f[I \lambda _1 - P_1 + P_2] \} \]

where \( f(x) = x \) if \( x \geq 0 \)

\[ = 0 \] otherwise

The problem can be expressed in terms of finding two spheres, each belonging to a distinct geometric structure, with the shortest distance between them.

3.1 Distance Between Volumes Generated from Dynamic Spheres with One-degree of Freedom

The kind of functions selected in the definition of the movement of centers \(P_i(\lambda_i)\) and evolution of the radius \(R_i(\lambda_i)\) affects directly the complexity of the Min-Max problem. Even for the simplest objects generated from spheres with one degree of freedom and linear functions the use of Euclidean distances means that, the set of resulting equations involve square roots.

This set of non-linear equations for computing the distance between two spherical-cones as shown in Figure 3 can be simplified resulting in two sets of vector equations which solve the problem in two steps.

1. Compute the direction of the shortest distance.

\[ n_\lambda \cdot (P_{11} - P_{10}) = R_{11} - R_{10} \]

\[ n_\lambda \cdot (P_{21} - P_{20}) = -(R_{21} - R_{20}) \]  

(8)

where the unknown \(n_\lambda\) is the unitary vector in the direction of the shortest distance between volumes and which is perpendicular to both surfaces.

2. Compute the spheres involved.

\[ P_{10} + \lambda_1 \cdot (P_{11} - P_{10}) = P_{20} + \lambda_2 \cdot (P_{21} - P_{20}) + d \cdot n_\lambda \]  

(9)
where \( \lambda_i, \lambda_j \) are the parameter values which define the spheres involved.

The distance is then obtained as follows.

\[
\text{Distance} = d - R_i(\lambda_i) - R_j(\lambda_j)
\]  
(10)

The unitary vector \( \vec{n}_s \) perpendicular to the surfaces can be computed by using a pseudo-polar coordinate system representation, as shown in Figure 4, and based on the three following vectors,

\[
\vec{v}_i = \left( \frac{P_{i1} - P_{i0}}{P_{i1} - P_{i0}} \right) \vec{v}_j = \left( \frac{P_{j1} - P_{j0}}{P_{j1} - P_{j0}} \right)
\]  
(11)

\[
\vec{n}_s = \vec{v}_i \times \vec{v}_j
\]

Any point in space, or any vector centered at the coordinate origin can be described in terms of a linear combination of these three unitary vectors. Thus

\[
\vec{n}_s = g_{v_i} \cdot \vec{v}_i + g_{v_j} \cdot \vec{v}_j + g_{n_s} \cdot \vec{n}_s
\]  
(12)

where

\[
\begin{align*}
g_{v_i} &= \rho \cdot \cos \theta \cdot \frac{\sin(\alpha - \phi)}{\sin a} \\
g_{v_j} &= \rho \cdot \cos \theta \cdot \frac{\sin \phi}{\sin a} \\
g_{n_s} &= \rho \cdot \sin \theta \\
a &= \arccos(\vec{v}_i \cdot \vec{v}_j)
\end{align*}
\]

For the coordinate system introduced and using the degrees of convergence, the inner-products in Equations 8 can be rewritten as

\[
\begin{align*}
g_{v_i} + g_{v_j} \cdot \cos a &= \eta_i \\
g_{v_i} \cdot \cos a + g_{v_j} &= -\eta_j
\end{align*}
\]  
(13)

By solving these equations, we get

\[
\tan \phi = \frac{\eta_j + \eta_i \cdot \cos a}{\eta_i \cdot \sin a}, \quad \cos \theta = \frac{\eta_i}{\cos \phi}, \quad \rho = 1
\]  
(14)

In the special case of two cylinders, the value of vector \( \vec{n}_s \) is obtained directly as \( \vec{n}_s = \vec{n}_c \).

When the vector \( \vec{n}_s \) has been computed, the values of the parameters \( \lambda \)'s can be obtained by solving the set of scalar linear equations or directly from the vector equation 9. Multiply this equation by \( \vec{n}_s = \vec{n}_s \times \vec{v}_i \) and \( \vec{n}_s = \vec{n}_s \times \vec{v}_j \) respectively, to obtain \( \lambda_i \) and \( \lambda_j \).

\[
\lambda_i = \frac{(P_{j0} - P_{i0}) \cdot \vec{n}_s}{(P_{j1} - P_{i0}) \cdot \vec{n}_s}
\]

\[
\lambda_j = -\frac{(P_{j0} - P_{i0}) \cdot \vec{n}_s}{(P_{j1} - P_{j0}) \cdot \vec{n}_s}
\]  
(15)

When at least one of the \( \lambda \)'s is out of range (e.g. \( \lambda_i \notin [0,1] \)), the distance between the corresponding end-sphere (\( P_{i*} = P_i(0) \) or \( P_{i*} = P_i(1) \)) and the other volume is computed. The problem is identical to computing the shortest distance between a sphere and a spherical-cone.

This problem is solved in two steps,

1. Compute the point in the axis nearest to the center of the sphere.

\[
\lambda_j^* = (P_{i*} - P_{j0}) \cdot (P_{j1} - P_{j0}) \left| P_{j1} - P_{j0} \right|
\]  
(16)

2. Update the value of \( \lambda_j \), to obey the angular condition represented by Equation 8.

\[
\lambda_j = \lambda_j^* + \frac{P_{i*} - P_j(\lambda_j^*)}{\left| P_{j1} - P_{j0} \right|} \cdot \tan \alpha_j
\]  
(17)

where \( \alpha_j = \arcsin \eta_j \).

### 3.2 Distance Between Volumes Generated from Dynamic Spheres with Two-degrees of Freedom

Distances between spherical-planes (two-degree of freedom) will involve determining the distances between spherical-vertices and spherical-planes as well as between spherical-edges. All possibilities are included by these two types of comparisons.

Spherical-edges are one-degree of freedom geometries, so they have been considered previously. However, given that an exhaustive distance computation between all the spherical-edges will be considered highly inefficient, several rules will have to be considered to reduce the number of elements potentially involved and therefore the number of comparisons.
The distance between a spherical-vertex or simply a sphere and a spherical-plane is computed by a process similar to the one for computing the distance between a point and a plane. That is, first by projecting the point onto the plane and then by measuring the distance between the point and the projected-point.

In our case, a sphere projected onto a spherical-plane produces a "projected-sphere" inside the spherical-plane in such a way that the distance between the two volumes is obtained by measuring the distance between these two spheres.

The process for computing the distance between spherical-vertices of volume \( i \) and the spherical-plane \( j \) is solved in two stages:

1. Compute the perpendicular to the surface of spherical-plane \( j \): The unitary vector perpendicular to the surface, \( \bar{n}_{jx} \), must obey the following equations
   \[
   \frac{\bar{n}_{jx} \cdot \bar{v}_1}{\bar{n}_{jx} \cdot \bar{v}_2} = \eta_1 = \frac{\bar{n}_{jx} \cdot \bar{v}_1}{\bar{n}_{jx} \cdot \bar{v}_2} \quad (18)
   \]
   where
   \[
   \eta_1 = \frac{(R_{j1} - R_{j0})}{|P_j - P_{j0}|} \quad \eta_2 = \frac{(R_{j2} - R_{j0})}{|P_j - P_{j0}|} \quad (19)
   \]
   \[
   \bar{v}_1 = \frac{(P_{j1} - P_{j0})}{|P_{j1} - P_{j0}|}, \quad \bar{v}_2 = \frac{(P_{j2} - P_{j0})}{|P_{j2} - P_{j0}|} \quad (20)
   \]
   A pseudo-spherical coordinates system based on vectors \( \bar{v}_1, \bar{v}_2 \) and \( \bar{n}_{jx} = \bar{v}_1 \times \bar{v}_2 \) is introduced to compute the vector \( \bar{n}_{jx} \), according to the Equation 12. In this case the values of the basic variables are
   \[
   \phi_j = \arctan\left(\frac{\eta_2 - \eta_1 \cdot \cos a}{\eta_1 \cdot \sin a}\right), \quad \theta_j = \arccos\left(\frac{\eta_1}{\cos \phi_j}\right), \quad \rho = 1 
   \]
   where \( a = \arccos(\bar{v}_1 \cdot \bar{v}_2) \)

2. Project each spherical-vertex of volume \( i \) onto volume \( j \) and compute the distances:
   (a) Each spherical-vertex of volume \( i \), \( \{(P_{vi}, R_{vi}), \quad k = 1, K, \quad K \geq 3\} \), is associated with a projected-sphere in the volume \( j \), \( (P_{pvi}, R_{pvi}) \). The center of the projected sphere is computed as follows
   \[
   P_{pvi} = P_{vi} + d_{ik} \cdot \bar{n}_{ij} \quad (22)
   \]
   where the distance between centers is computed by
   \[
   d_{ik} = \frac{(P_{j0} - P_{pvi}) \cdot \bar{n}_{je}}{\bar{n}_{je} \cdot \bar{n}_{js}} \quad (23)
   \]
   When the center, \( P_{pvi} \), has been computed the radius, \( R_{pvi} \), and the parameters \( \lambda 's \) for the projected-sphere need to be obtained. To compute \( \lambda_1 \) and \( \lambda_2 \), multiply the first Equation 5 by \( \bar{n}_{je} = \bar{n}_{je} \times \bar{v}_2 \) and \( \bar{n}_{je} = \bar{n}_{je} \times \bar{v}_1 \) respectively, to give
   \[
   \lambda_1 = \left(\frac{P_{pvi} - P_{j0}}{P_{j1} - P_{j0}} \right) \cdot \bar{n}_{je} \quad (24)
   \]
   \[
   \lambda_2 = \left(\frac{P_{pvi} - P_{j0}}{P_{j2} - P_{j0}} \right) \cdot \bar{n}_{je} \quad (25)
   \]
   (b) The distance between the spherical-vertex and the spherical-plane, supposed it unbounded, is given by
   \[
   D_{ik} = d_{ik} - R_i(\lambda_1, \lambda_2) - R_j(\lambda_1, \lambda_2) \quad \forall \lambda 's \in \mathbb{R}
   \]
   If the spherical-vertex corresponding to the minimum distance, \( D_{ik}^* = \min\{D_{ik}\} \), generates a projected-sphere inside the spherical-plane, \( (\lambda 's \in [0, 1]) \), then this minimum distance is actually the distance between the two objects. Otherwise, the values of the \( \lambda 's \) of the projected-sphere for all the spherical-vertex will determine up to four spherical-edge to spherical-edge distance computations.

4 Numerical Results and Example

A set of algorithms has been implemented in order to verify the theory and to determine the simplicity and efficiency of the technique presented in this paper. The spherical-geometries generated from dynamic-spheres have been used to represent various robotic systems and their environments.

The distance computation algorithms have been tested on a standard SUN SPARCstation 1, a RISC-based workstation, using sets of one hundred randomly generated and positioned volumes. The minimum (Min.), maximum (Max.), and average (Ave.) times in milliseconds for computing distance between all the combinations of objects developed in the theory have been recorded in the following table.
Objects Compared | Min. | Max. | Ave. |
--- | --- | --- | --- |
sphere to sphere | 0.05 | 0.05 | 0.05 |
sphere to sph-cylinder | 0.14 | 0.14 | 0.14 |
sphere to sph-cone | 0.25 | 0.26 | 0.26 |
sphere to sph-plane | 0.65 | 0.91 | 0.87 |
sph-cylinder to sph-cylinder | 0.20 | 0.23 | 0.23 |
sph-cylinder to sph-cone | 0.65 | 1.57 | 1.05 |
sph-cone to sph-plane* | 0.72 | 3.86 | 2.82 |
sph-plane* to sph-plane* | 2.81 | 9.09 | 7.91 |

(*) For these computations, planes with 4 vertices have been considered.

A robotic testbed platform, used at the CIRSSE to study robotic assembly of structures in space, has been modeled in order to thoroughly test the entire theory. The setup consists of a PUMA 560 and a PUMA 600 each mounted on a three-degree of freedom platform.

A kinematic model based on homogeneous transformation matrices using a modified form of the Denavit-Hartenberg parameters [7] is used in order to determine the position of the robot links.

Each link of the robotic system has at least one associated volume as can be seen in Figure 5. In particular each individual PUMA robot-arm is modeled by the following spherical-volumes: A sphere and a spherical-cylinder for Link 0, a spherical-cylinder for Link 1, spherical-planes for links 2 and 3, this latter one including the wrist, and a sphere as an approximation to the gripper. A more precise representation of the latter objects could be constructed using several smaller volumes.

When considering self-collision for the PUMA robot arm, the set of volumes described above can be simplified to give even a better approximation. In terms of distance computation, links 2 and 3 are represented better by a combination of spherical-cones and spherical cylinders. This is due to the fact that certain links can never collide and in addition, certain surfaces will never be involved in the computation of shortest distances between elements in the same robot-arm.

Only six distances need to be checked for self-collision: one sphere to sphere, three sphere to spherical-cylinder, one sphere to spherical-cone, and one spherical-cylinder to spherical-cone. Taking values from the table of computation times, the minimum, maximum, and average times in milliseconds are 1.37, 2.30, and 1.78 respectively. The maximum error in distances with respect to the exact shape of links 0 to 3 plus the sphere at the end is less than one centimeter.

Collisions with the floor are determined immediately from the z-coordinates of the sphere bounding the gripper and one end-sphere of the spherical-cone for Link 3.

For the overall robotic system, using the volumes shown in Figure 5, the number of comparisons required are 1 sphere to sphere, 14 sphere to sphere-cylinder, 11 sphere to spherical-plane, 20 spherical-cylinder to spherical-cylinder, 25 spherical-cylinder to spherical-plane and 12 spherical-plane to spherical-plane. This gives minimum, maximum and average times in milliseconds of 66, 220 and 170 respectively.

When restricting the volumes to zero or one degree of freedom, using spherical-cones for links 2 and 3 plus wrists, as well as replacing platforms with hemispheres, the computation time is reduced significantly. In this case the number of comparisons are as follows, 6 sphere to sphere, 14 sphere to sphere-cylinder, 14 sphere to sphere-cone, 10 spherical-cylinder to spherical-cylinder, 15 spherical-cylinder to spherical-plane and 4 spherical-cone to spherical-cone. This gives minimum, maximum and average times in milliseconds of 20, 38 and 28 respectively. Details of these numerical results are presented and discussed in a CIRSSE report [8].

## 5 Conclusions and Future Work

A new three-dimensional object representation technique for generating spherical-volumes has been presented based on the idea that any real object can be approximated by an infinite number of spheres. These spheres are easily described by introducing the concept of dynamic-spheres. Although only linear functions have been used for describing the dynamic-spheres, the set of volumes generated is quite extensive and has been proved to be enough to model a robotic system with reasonable accuracy. The concept is easily extended to include other kinds of functions.

A very fast procedure for computing distances between spherical-volumes has been developed with good numerical results. The procedure has been shown to be simple and efficient when dealing with robotic systems.

Further work will focus on using other kinds of functions for the dynamic-spheres, with the intent of: - increasing the variety of volumes generated, e.g., in two-degree of freedom a dynamic-sphere could follow general surfaces in order to give a better approximation of complex objects; - considering basic movements such as spheres rotating around an axis to describe the movement of a robot-arm in terms of the volumes swept by its links.
Acknowledgement
Josep Tornero gratefully acknowledges the support of this research through a grant from the Spanish Government (MEC-DGICYT). Gregory J. Hamlin and Robert B. Kelley appreciate the support in part by Grant NAGW-1333 from NASA to the Center for Intelligent Robotic Systems for Space Exploration.

References