Planning collision-free trajectories under real-time restrictions is a challenging topic in robotics. In order to reduce computational cost in the collision avoidance process, some authors have proposed different model representations. This article presents an optimal method able to generate automatically geometric models of the objects in a robotic system. For each object, two models are obtained, i.e., the minimum outer and the maximum inner models. Availability of both models allows one to face more successfully robot motion applications. The geometric modeler is focused on the generation of spherically extended polytopes. Each object to model is represented by a set of points taken from its surface. Models are generated through the application of an iterative process based on the Hough transform. When both outer and inner models have been generated, a parameter for evaluating the quality of the models is introduced. This parameter can be used by a rule-based system for increasing the complexity of the model generated and improving, therefore, the accuracy of the representation.

1. INTRODUCTION

One of the intermediate steps in the robot motion planning process is the capacity for detecting and avoiding, as fast as possible, collisions with the surrounding obstacles. Distance computation is required in many methods. Sonar sensors, laser sensors, and artificial vision systems are usually used in distance computation and therefore in collision detection.

In spite of the fact that distances can be measured or computed in real-time, the computational cost for trajectory planning, under time restrictions, used to be totally unacceptable.

The choice of a specific geometric representation for modeling a robotic system depends on the technique for distance computing to be used.
Several authors have used different types of models for representing robotic systems, for instance, robotic applications based on a sphere-based model, mobile robots and obstacles modeled by convex polygons (2D) and convex polyhedrons (3D) used for enveloping the robotic system.

One of the most popular ways for reducing time in collision avoidance consists of modeling the obstacles, mobile robots, and/or manipulators, presented in a robotic system by means of planar or spherical surfaces. Earlier examples of these techniques are the bubble model and general spherical surfaces.

In this way, a fast method for computing distances between complex objects modeled by polytopes in \( \mathbb{R}^3 \) have been introduced. The computation cost is approximately linear in the total number of vertices specifying the two polytopes.

Successive spherical approximation (SSA), based on a hierarchical external and internal representation, was introduced in ref. 16. SSA is applied to convex polyhedron. Later, it is proposed as an object representation based on a spheres’ hierarchical structure. This approximation consists of two sets of spheres. The first set is called the exterior representation and it has the property of enclosing the boundaries of the object. The second set forms the interior representation where each sphere must be contained in the object being modeled. The number of spheres used in both representations can be increased in order to improve the accuracy of the models. In this way, the representation can converge to a zero-error model. Additionally, the authors introduce a technique called the spherizer for modeling automatically a robot and its surroundings. The spherizer assumes that the object to be model is represented by a polyhedron generated by translational sweeping of a polygon.

A spherical geometric representation in robotic systems for fast distance computation is also described. The modeling technique is conceptually between the bubble model and the generalized cylinder. Later, this representation was formalized as spherically extended polytopes (s-topes). With this representation, complex objects can be described by simple spherical volumes and safety distances can be introduced without increasing complexity.

In terms of applications, authors rarely show how models have been obtained, i.e., if the model has been designed manually or through an automatic process. Even more, in these cases, the computational cost involved in order to model the robotic systems is not generally reflected in their conclusions.

In this article, we describe a method for the automatic and optimal generation of the different models needed for representing the whole robotic system. This method is referred to as a geometric modeler. Given an object, the geometric modeler is automatically able to design the minimum outer model that envelops it and the maximum inner model completely enveloped by the mentioned object.

In general, the geometric modeling of an object may be focused on, at least, two representation levels. The former is called system representation and it consists of enveloping the whole robot by means of a sole model. The latter, called element representation, is based on modeling each link of the robot separately. So, the union of all parts’ models represents the whole object. Moreover, it is possible to use models of different complexity in each representation; in this way, increasing the complexity of a model implies improvement in accuracy. This approach is referred to as the hierarchical model.

The present method is focused on generating spherically extended polytopes (s-topes) supporting hierarchical model representations.

The authors have updated this geometric modeling technique for solving the collision avoidance and path planning problems in robotic systems.

The remainder of the article is organized as follows. In section 2 we present a formal and brief description of the s-tope geometry. The geometric modeler method is divided into two sections. The generation of the minimum outer s-tope is introduced in section 3, while the generation of the maximum inner model part is shown in section 4. Section 5 presents a parameter for quantifying the accuracy of the representation. Modeling results are presented in section 6. Finally, in section 7, we present conclusions about the work.

2. SPHERICALLY EXTENDED POLYTOPES

An s-tope is the convex hull of a finite set of spheres. A sphere is denoted \( s = (c, r) \) where \( c \) is the center and \( r \) is the radius. Given the set of spheres \( S = \{s_0, s_1, \ldots, s_{n-1}\} \), the convex hull of \( S \), \( S_{0-n} \) contains an infinite set of spheres called the set of swept...
spheres which is expressed by
\[
S_{0-n} = \left\{ s : s = \sum_{i=0}^{n-1} \lambda_i s_i, \ s_i \in S, \ \lambda_i \geq 0, \ \sum_{i=0}^{n-1} \lambda_i = 1 \right\}
\] (1)

According to (1), it is possible to dispense with \( \lambda_0 \)
\[
\lambda_0 = 1 - \sum_{i=1}^{n-1} \lambda_i \] (2)

Note that the set of swept spheres does not include all possible spheres which can fit inside the s-tope, only those which are generated by Eq. (1). Graphical examples of s-topes are shown in Figure 1.

Spheres in \( S \), which define an s-tope, are called spherical-vertices. An s-tope is said to be overspecified if one or more of its spherical-vertices may be removed without changing the convex hull. An s-tope nonoverspecified is said to be a valid s-tope. The s-tope order is the minimum number of spherical-vertices.

The simplest s-tope is a single sphere. For simple s-topes (order lower than or equal to four) the spherical vertices are listed explicitly. For example, \( S_{01} \) is an s-tope with spherical-vertices \( s_0 \) and \( s_1 \). The second order s-tope is called the bi-sphere. The tri-sphere is the third order s-tope and so on.

An interesting s-tope property is stated as follows. Considering the primary sphere \( s_0 \), s-topes may be recursively defined. For instance, a tri-sphere \( S_{012} \) is formed by three bi-spheres \( s_{01}, s_{02}, \) and \( s_{12} \) (see Figure 1). A tetra-sphere \( S_{0123} \) is characterized by four tri-spheres \( S_{012}, S_{013}, S_{023}, \) and \( S_{123} \). Therefore, an n-sphere with \( n > 1 \) is formed by \( n(n-1) \)-spheres.

3. MINIMUM OUTER S-TOPE

The justification in searching the minimum outer model states that modeling an object by means of such an envelope implies that the accuracy obtained is maximal for the chosen geometry. For instance, consider that two different spheres model the same object; obviously the lower-volume sphere will provide the most accurate representation.

The information available from the object to be modeled is a set of points \( P = \{ p_0, p_1, \ldots, p_m \} \) taken from its surface. These points can be obtained by means of sonar and laser sensors, artificial vision systems, etc., requiring sometimes sensor data fusion procedures. It is also possible to get those points from the design description of the objects (CAD tools). The minimum outer model is generated with the constraint that \( P \) has to be included in it and its volume has to be the lowest.

The generation of \( S_{0-n} \) implies to find a set of spheres \( S = \{ s_0, s_1, \ldots, s_{n-1} \} \) whose 3D centers represent the model position, while the radii quantify its size. Therefore, the model generation consists of finding \( 4 \times n \) parameters meeting the above-mentioned constraint. Computational cost in the s-tope generation process is directly proportional to the s-tope order and, theoretically, the s-tope order is inversely proportional to the model accuracy. So, a good trade-off between complexity and accuracy may be reached with s-tope models.

The presented method for obtaining the minimum outer s-tope is an iterative process that is focused on two hierarchical levels. The high level consists of the downhill simplex optimization method, while the low level is based on a particular application of the Hough transform.

High level searches the position of the minimum outer s-tope; i.e., it deals with the s-topes centers. Low level, using the centers received from...
the higher, generates the needed radii, one for each center, to envelop $P$ with the minimum size. Then, the s-tope’s volume is computed and sent to high level. According to the result of this evaluation, high level will provide to the lower another set of centers. This process is repeated until success. Figure 2 shows the generation method scheme.

The downhill simplex needs an initial condition $C_0$ to start. In our application, $C_0$ has to be an initial s-tope position of searching. Assuming the influence between $C_0$ and the downhill simplex computational cost, it is important to provide an adequate initial value for $C_0$. Therefore, it is advisable to obtain $C_0$ through the application of clustering techniques to $P$.

Given the popularity of the downhill simplex optimization method, our presentation has to be mainly focused on the low level. Our exposition is presented from particular to general. So, we start in the next subsection with the bi-sphere generation process, later the tri-sphere case, and finally the n-sphere case. The sphere’s generating process, although it has been extensively studied in the framework of the mathematical theory of packing and covering, may be implemented in accordance with the presented method.

3.1. Bi-sphere

As it is a second order s-tope, i.e., $n = 2$ high level deals with a set of two centers $\{c_0, c_1\}$, and low level will have to search two radii $\{r_0, r_1\}$.

At low level, when $\{c_0, c_1\}$ is received, each point $p_i \in P$ is described by two parameters $\lambda_i$ and $d_i$. $\lambda_i$ represents the $p_i$’s position with respect to the segment formed by $c_0$ and $c_1$, while $d_i$ is the distance from $p_i$ to that segment. $\lambda_i$ is obtained as

$$p^\perp_i = c_0 + \lambda_i(c_1 - c_0)$$

and

$$(p^\perp_i - p_i) \cdot (c_1 - c_0) = 0$$

where $p^\perp_i$ is the projection of $p_i$ onto the straight line defined by the centers, $c_0c_1$.

Those points $p_i = (\lambda_i, d_i)$, located between $c_0$ and $c_1$, their $\lambda_i$’s verify $\lambda_i \in [0, 1]$; see Figure 3. They are called inward points. The set of these points is denoted IWP. In other words, IWPs are located between two planes of normal $c_0c_1$ at $c_0$ and $c_1$, respectively. The rest are called outward points OWP. They are classified into two subsets as follows:

$$\text{OWP}^{<} = \{ p_i \in P : p_i = (\lambda_i, d_i), \lambda_i \leq 0 \}$$

$$\text{OWP}^{>} = \{ p_i \in P : p_i = (\lambda_i, d_i), \lambda_i \geq 1 \}$$

(4)

After $p_i$ classification, two parameters are defined

$$\forall p_i = (\lambda_i, d_i) \in \text{OWP}^{<} \rightarrow d^{\min}_i = \max_i \{d_i\}$$

$$\forall p_i = (\lambda_i, d_i) \in \text{OWP}^{>} \rightarrow d^{\min}_i = \max_i \{d_i\}$$

(5)

where $d^{\min}_0$, $d^{\min}_1$ represent the minimum radii needed to envelop the OWP. If some subset OWP$^{<}$ is empty, then the associated parameter $d^{\min}_i$ will be set to zero.

Consequently, with $d^{\min}_0$ and $d^{\min}_1$ only the IWP may not be enveloped. But each $p_i \in$ IWP is minimally enveloped by a set of pairs $(r_0, r_1)$; see Figure 4. These pairs of radii lay on a straight line,
obtained by applying a simple proportionality law

\[
\frac{r_1 - r_0}{1} = \frac{d_i - r_0}{\lambda_i} \quad \Rightarrow \quad r_1 = \left(1 - \frac{1}{\lambda_i}\right)r_0 + \frac{d_i}{\lambda_i}
\]  

(6)

where \( r_0 \geq d_0^{\min} \), \( r_1 \geq d_1^{\min} \), and \( |r_1 - r_0| \leq l_0 \). In this article, we call \( l_{ij} = ||c_j - c_i|| \). The last constraint is required to verify the valid bi-sphere condition.\(^{21}\) Such a straight line is called the Hough straight line (HSL). HSLs have the following properties:

1. Their slopes are always negative.

\[
\forall p_i = (\lambda_i, d_i) \in IWP \\
\rightarrow \lambda_i \in ]0, 1[ \rightarrow 1 - \frac{1}{\lambda_i} < 0
\]  

(7)

2. HSL intersection with the \( r_0 = 0 \) and \( r_1 = 0 \) axis is always greater than \( o \) equal to zero. Precisely, these intersections are located at \((0, d_i/\lambda_i)\) and \((d_i/(1 - \lambda_i), 0)\).

3. Every HSL divides the \( r_0 r_1 \) plane—with \( r_0 \geq d_0^{\min} \) and \( r_1 \geq d_1^{\min} \)—into two half-planes, which are called external and internal, respectively. The external half-plane verifies

\[
\forall p_i = (\lambda_i, d_i) \in IWP \quad r_1 > \left(1 - \frac{1}{\lambda_i}\right)r_0 + \frac{d_i}{\lambda_i}
\]  

(8)

Any point \((r_0, r_1)\) at this external half-plane is representing a bi-sphere that amply envelops \( p_i \). The proof is trivial. If \( r_0 = r_0^* \) then the needed \( r_1 \) to envelop minimally \( p_i \) is given by (6), and in accordance with (8), this radius is lower than \( r_1^* \).

Analogously, the internal half-plane represents bi-spheres that do not envelop \( p_j \), because of compliance with the following expression:

\[
\forall p_j = (\lambda_i, d_i) \in IWP \quad r_1 < \left(1 - \frac{1}{\lambda_i}\right)r_0 + \frac{d_i}{\lambda_i}
\]  

(9)

Considering all the HSLs and without forgetting \( r_0 \geq d_0^{\min} \), \( r_1 \geq d_1^{\min} \), we call the minimum volume locus \((M_N \text{ VL})\) the border among all the external half-plane intersections. An example of \((M_N \text{ VL})\) is drawn by a thick stroke line in Figure 5. \((M_N \text{ VL})\) has the following properties:

1. Every point \((r_0^*, r_1^*)\) at the \((M_N \text{ VL})\) represents a bi-sphere that envelops \( P \). This is a consequence of the last property of the HSL. Therefore, the minimum outer bi-sphere at \( c_0, c_1 \) will be obtained sweeping \((M_N \text{ VL})\) between \( A \) and \( B \). The value of \( r_0 \) at \( A \), \( r_0^A \) is

\[
r_0^A = \max_i \left(\frac{d_i - \lambda_i d_i^{\min}}{1 - \lambda_i} \right) \\
p_i = (\lambda_i, d_i) \in IWP
\]  

(10)

2. Every \( p_i \in IWP \) whose HSL does not belong to the \((M_N \text{ VL})\) has no influence on the search of the minimum outer bi-sphere at \( (c_0, c_1) \). See the HSL \( r_E \) in Figure 5.
3. Every \( p_i \in IWP \) whose HSL is totally included in the area formed by the straight lines \( r_0 = d_0^{\min} \), \( r_1 = d_1^{\min} \) and the axis \( r_{0\ell} \), \( r_{1\ell} \) is already enveloped at \( \{c_0, c_1\} \) by the bi-sphere of radii \( d_0^{\min}, d_1^{\min} \).

As a consequence of the first \( M_{NVL} \) property, a search is required for finding the minimum outer bi-sphere at \( \{c_0, c_1\} \). So a parameter is needed to compute the volume of a bi-sphere. This parameter is given by the expression:

\[
A_{s_{\alpha}} = \frac{\pi}{4} (r_0^2 + r_1^2) + \frac{\alpha_{01}}{2} (r_1^2 - r_0^2) + \frac{l_{01}}{2} \cos \alpha_{01} (r_0 + r_1) \tag{11}
\]

where \( l_{01} = \|c_1 - c_0\| \) and \( \alpha_{01} = \sin((r_1 - r_0)/l_{01}) \).

\( A_{s_{\alpha}} \) represents the area, whose revolution with respect to the axis formed by the centers \( c_0, c_1 \) generates the bi-sphere \( S_{01} \).

In order to avoid an exhaustive search, the golden section optimization method \(^{26}\) is used to find the minimum bi-sphere belonging to the \( M_{NVL} \).

The interval of search in the golden section method, i.e., \( r_0 \in \{ r_0^A, r_0^B \} \), is divided into three subintervals. Then \( A_{s_{\alpha}} \) is evaluated at each interval limit. A decision process is used to either squeeze from the right or squeeze from the left and then subdivide again the chosen interval if required.

Given \( r_{0\ell} \), the associated \( r_{1\ell} \) that is located at the \( M_{NVL} \) is obtained by computing

\[
\forall \text{HSL}, \ r_0 \in \{ r_0^A, r_0^B \} : \quad r_{1\ell} = \max_i \left( \frac{1 - \frac{1}{\lambda_i}}{\lambda_i} r_0 + \frac{d_i}{\lambda_i} \right) \tag{12}
\]

Then according to (12), if the number of HSL is high, the computational cost needed to compute \( r_{1\ell} \) is significant. But, as after each search iteration, the interval of searching is reduced, not all the HSLs can belong to the \( M_{NVL} \) limited by the current \( r_{0\ell} \) interval. In this way, the first step is to sort the HSL from the greatest intersection ordinate with \( r_0 = d_0^{\min} \) to the lowest. As a consequence of the first HSL and third \( M_{NVL} \) properties, those HSLs whose intersection ordinate is lower than \( d_1^{\min} \) are rejected. The obtained set of sorted HSL is called VAL, with \( \text{VAL} \subseteq \text{IWP} \). Then, only the HSL in VAL will be considered in the search of \( M_{NVL} \).

**Proposition 1:** All the HSL in VAL whose intersection ordinate at \( r_0 = d_0^{\min} \) are lower than the intersection ordinate of the HSL that belongs to the \( M_{NVL} \) at the end of the current \( r_0 \) interval of searching have to be removed from VAL, because these HSL will never belong to the \( M_{NVL} \). See HSL \( r_D \) and \( r_E \) in Figure 5 for the initial \( r_0 \) interval case.

**Proposition 2:** All the HSL in VAL whose intersection ordinate at \( r_0 = d_0^{\min} \) are greater than the intersection ordinate of the HSL that belongs to the \( M_{NVL} \) at the beginning of the current \( r_0 \) interval of searching have to be removed from VAL because these HSL will never belong to the \( M_{NVL} \).

Taking the mentioned propositions into account, the golden section algorithm applied to the \( M_{NVL} \) search is as follows:

```plaintext
// a < c < d < b are the limits of the subintervals. f represents both A_{s_{\alpha}} and r_{1\ell}-obtaining function.

delta // Tolerance for interval width
epsilon // Tolerance for |f(b) - f(a)|
r = (5^{1/2} - 1)/2 // Golden ratio^{26}
rtwo = r^*r
a = r_0^A, b = r_0^B
VAL = Sort(IWP)  // Removing HSL verifying prop. 1
h = b - a // Accuracy
c = a + rtwo*h, d = a + r*h
while |f(b) - f(a)| > epsilon or h > delta
    if f(c) < f(d) then
        b = d, d = c, h = b - a, c = a + rtwo*h
        VAL = RemoveP1(VAL)  // Remove HSL verifying prop. 1
    else
        a = c, c = d, h = b - a, d = a + r*h
        VAL = RemoveP2(VAL)  // Remove HSL verifying prop. 2
    endif
endwhile
return min \{f(a), f(c), f(d), f(b)\}
```

The algorithm finishes with accuracy \( \pm h \). The computational cost of the golden section algorithm for searching the minimum bi-sphere at the \( M_{NVL} \) is presented in Figure 6. This cost has been obtained running the software on a Pentium® II 350 MHz. HSLs have been randomly generated.
3.2. Tri-sphere

It is the three order s-tope defined by three spherical-vertices, \( n = 3 \). Consequently, for generating a tri-sphere \( S_{012} \), it is necessary to find three spheres \( \{ s_0, s_1, s_2 \} \). In this case, the high level algorithm deals with three centers \( c_0, c_1, c_2 \), while the lower one has to find its respective radii.

Now, at low level every \( p_i \in P \) is described by \( p_i = (\lambda_{i1}, \lambda_{i2}, d_i) \). \( \lambda_{i1} \) and \( \lambda_{i2} \) indicate the \( p_i \)'s position with respect to the triangle formed by the centers. \( d_i \) is the distance from \( p_i \) to the mentioned triangle. \( \lambda_{i1}, \lambda_{i2} \) are obtained solving the following equations:

\[
p_i^\perp = c_0 + \lambda_{i1}(c_1 - c_0) + \lambda_{i2}(c_2 - c_0)
\]

\[
(p_i^\perp - p_j) \cdot (c_j - c_0) = 0 \quad j = 1, 2
\]

Points to envelop are handled in accordance with their parameters \( \lambda_{i1}, \lambda_{i2} \). In this way,

1. \( \lambda_{i1} + \lambda_{i2} < 1 \). The following cases have to be distinguished:

   a. \( \lambda_{i1}, \lambda_{i2} > 0 \). \( p_i^\perp \) is in the area of the triangle formed by the centers \( \{ c_0, c_1, c_2 \} \). These points are considered IWP and to envelop them it is necessary to take into account the three radii. So, each \( p_i \) has to be transformed into two HSLs:

\[
\forall p_i: \sum_{j=1,2} \lambda_{ij} < 1, \lambda_{ij} > 0
\]

\[
\rightarrow \begin{cases} 
    r_1 = \left( 1 - \frac{1}{\lambda_{i1}} \right) r_0 + \frac{d_i}{\lambda_{i1}} \\
    r_2 = \left( 1 - \frac{1}{\lambda_{i2}} \right) r_0 + \frac{d_i}{\lambda_{i2}} 
\end{cases}
\]

2. \( \lambda_{i1} < 0, \lambda_{i2} \geq 0 \). As can be observed in Figure 7, these points have to be enveloped by the bi-sphere \( S_{02} \); i.e., it is only necessary to consider radii \( r_0 \) and \( r_2 \). Therefore, these points are handled as it were the bi-sphere case. In this way, every \( p_i \) will be described by \( p_i = (\lambda_{i0}^{02}, d_i) \), where \( \lambda_{i0}^{02} \) is the \( p_i \)'s position with respect to the segment formed by the centers \( c_0 \) and \( c_2 \). \( d_i \) is the distance to the mentioned segment. According to \( \lambda_{i0}^{02} \), the step to follow is

\[
p_i = (\lambda_{i0}^{02}, d_i)
\]

\[
\Rightarrow \begin{cases} 
    \lambda_{i0}^{02} \leq 0 \rightarrow d_0^{\text{min}} = \max\{d_0^{\text{min}}, d_i\} \\
    \lambda_{i0}^{02} \in ]0,1[ \rightarrow r_2 = \left( 1 - \frac{1}{\lambda_{i0}^{02}} \right) r_0 + \frac{d_i}{\lambda_{i0}^{02}} \\
    \lambda_{i0}^{02} \geq 1 \rightarrow d_2^{\text{min}} = \max\{d_2^{\text{min}}, d_i\}
\end{cases}
\]

(15)

c. \( \lambda_{i1} \geq 0, \lambda_{i2} < 0 \). It is analogous to the previous one (see Figure 7), but in this case the associated bi-sphere is \( S_{01} \). So, each \( p_i \) will be described by \( p_i = (\lambda_{i1}^{01}, d_i) \). To envelop these \( p_i \) it is necessary to consider

\[
\forall p_i: \sum_{j=1,2} \lambda_{ij} < 1, \lambda_{ij} > 0
\]

\[
\rightarrow \begin{cases} 
    r_1 = \left( 1 - \frac{1}{\lambda_{i1}} \right) r_0 + \frac{d_i}{\lambda_{i1}} \\
    r_2 = \left( 1 - \frac{1}{\lambda_{i2}} \right) r_0 + \frac{d_i}{\lambda_{i2}} 
\end{cases}
\]
one of the following possibilities:

\[ p_i = (\lambda_{i1}^{10}, d_i) \]

\[
\begin{align*}
\lambda_{i1}^{10} & \leq 0 \rightarrow d_{i0}^{\text{min}} = \max\{d_{0i}^{\text{min}}, d_i\} \\
\lambda_{i1}^{10} & \in [0, 1] \rightarrow r_1 = \left(1 - \frac{1}{\lambda_{i1}^{10}}\right) r_0 + \frac{d_j}{\lambda_{i1}^{10}} \\
\lambda_{i1}^{10} & \geq 1 \rightarrow d_{i1}^{\text{min}} = \max\{d_{1i}^{\text{min}}, d_i\}
\end{align*}
\]

(16)

d. \( \lambda_{i1}, \lambda_{i2} \leq 0 \). These points can be enveloped by either the bi-sphere \( S_{0i} \) or \( S_{02} \). Therefore, as a first step, it is necessary to know which segment is closer to \( p_i \). In this way, if the closer segment is formed by the centers \( c_0 \) and \( c_i \), then \( p_i \) has to be enveloped by \( S_{0i} \), and then step (16) is followed. Otherwise, the bi-sphere to consider is \( S_{02} \), and then (15) is applied.

2. \( \lambda_{i1} + \lambda_{i2} \geq 1 \). The following situations are distinguished:

a. \( \lambda_{i1}, \lambda_{i2} \geq 0 \). According to their positions (see Figure 7), enveloping these points implies taking into account the bi-sphere \( S_{12} \). Therefore, every \( p_i \) has to be described by \( p_i = (\lambda_{i2}^{12}, d_i) \), where \( \lambda_{i2}^{12} \) indicates the \( p_i \)'s position with respect to the segment formed by the centers \( c_1 \) and \( c_2 \), \( d_i \) being the distance to that segment. Then, these points are handled

\[ p_i = (\lambda_{i2}^{12}, d_i) \]

\[
\begin{align*}
\lambda_{i2}^{12} & \leq 0 \rightarrow d_{i1}^{\text{min}} = \max\{d_{1i}^{\text{min}}, d_i\} \\
\lambda_{i2}^{12} & \in [0, 1] \rightarrow r_2 = \left(1 - \frac{1}{\lambda_{i2}^{12}}\right) r_0 + \frac{d_j}{\lambda_{i2}^{12}} \\
\lambda_{i2}^{12} & \geq 1 \rightarrow d_{i2}^{\text{min}} = \max\{d_{2i}^{\text{min}}, d_i\}
\end{align*}
\]

(17)

b. \( \lambda_{i1} < 0, \lambda_{i2} \geq 0 \). These points have to be enveloped by either bi-sphere \( S_{02} \) or \( S_{12} \). The corresponding bi-sphere is that whose segment formed by their centers is closer to \( p_i \). Therefore, if \( p_i \) is closer to the segment of \( S_{02} \), then \( p_i \) is managed as is indicated in (15); otherwise, it is handled as is indicated in (17).

c. \( \lambda_{i1} \geq 0, \lambda_{i2} < 0 \). Observing Figure 7, to envelop these points it is necessary to take into account one of the bi-spheres \( S_{0i} \) or \( S_{12} \). In order to envelop \( p_i \), the involved bi-sphere is that whose associated segment is closer to \( p_i \). In this way, if \( S_{0i} \) has to be considered, step (16) has to be applied; otherwise, \( p_i \) is handled as it is indicated in (17).

After this analysis, on the one hand, the constraints \( d_{0i}^{\text{min}}, d_{1i}^{\text{min}}, d_{2i}^{\text{min}} \) and, on the other hand, three sets of HSL relating the radii \((r_0, r_1), (r_0, r_2), \) and \((r_1, r_2)\) have been obtained. For each set of HSL and the associated constraints, a \( M_{NVL} \) is considered.

Simultaneously, the first two \( M_{NVL} \) are swept by \( r_0 \), just as has been indicated in the bi-sphere’s generating process. After each golden section iteration a combination of radii \((r_0, r_1, r_2)\) is obtained. To ensure that all the points represented by the HSL related to \((r_1, r_2)\) are enveloped, the third \( M_{NVL} \) is also taken into account. Simply, it consists of verifying if the provided \((r_1, r_2)\) are located at the intersection of all the external half-planes or at exactly the \( M_{NVL} \) formed by the mentioned HSL. If none of the two conditions is satisfied, it implies that some point is not enveloped. Therefore, \( r_2 \) will be increased until the new \((r_1, r_2)\) is at the mentioned \( M_{NVL} \).

At each moment, the radii have to verify \( r_0 \geq d_{0i}^{\text{min}}, r_1 \geq d_{1i}^{\text{min}}, r_2 \geq d_{2i}^{\text{min}} \).

The function to minimize, used in the tri-sphere case, is based on one of its characteristic. A tri-sphere \( S_{012} \) is defined by three bi-spheres \( S_{0i}, S_{02}, S_{12} \), so this function is

\[ A_{S_{012}} = A_{S_{0i}} + A_{S_{02}} + A_{S_{12}} \]

\[ + A_{c_0 c_1 c_2} \]

where \( A_{c_0 c_1 c_2} \) is the area of the triangle formed by \((c_0, c_1, c_2)\). An important property of this function is its recursive character. Notice that it is not needed to compute \( A_{c_0 c_1 c_2} \) at low level.

The tri-sphere’s low level algorithm has been implemented in C and tested on a Pentium® II 350 MHz. HSLs are randomly generated. Figure 8 shows the low level’s computational cost.

### 3.3. Tetra-sphere

As it is a four order s-tope, it is necessary to generate four spheres. Therefore, high level deals with four centers, and low level determines the corresponding radii for each center.
At low level, every \( p_i \in P \) is described by four parameters, i.e., \( p_i = (\lambda_{ij}, \lambda_{i2}, \lambda_{i3}, d_i) \), where the \( \lambda_{ij} \) holds the \( p_i \) position with respect to the tetrahedron of vertices \( (c_0, c_1, c_2, c_3) \), while \( d_i \) is the distance from \( p_i \) to the mentioned tetrahedron. \( \lambda_{ij} \) are obtained by solving the following equations:

\[
p_i^+ = c_0 + \sum_{j=1}^{3} \lambda_{ij}(c_j - c_0) \]

\[
(p_i^+ - p_j)(c_j - c_0) = 0 \quad j = 1, 2, 3 \quad (19)
\]

Every \( p_i \) with \( \lambda_{ij} + \lambda_{i2} + \lambda_{i3} \leq 1 \) and \( \lambda_{ij} \geq 0 \) are inside the tetrahedron formed by the centers. Therefore, they are already enveloped and, obviously, they have to be discarded. The rest of \( p_i \in P \), located outside the mentioned tetrahedron, has to be enveloped by one of the four tri-spheres that define a tetra-sphere. Therefore, according to their \( \lambda_{ij}, p_i \in P \) are treated as it is indicated in Table I.

When, at the right column in Table I, more than one tri-sphere appears, it means that the tri-sphere finally involved is that whose triangle formed by its centers is the closest to \( p_i \).

After considering all \( p_i \in P \), the constraints \( d_{ij}^{\text{min}} \), with \( j = 0, \ldots, 3 \) and six sets of HSL, relating the radii \( (r_0, r_1), (r_0, r_2), (r_0, r_3), (r_1, r_2), (r_1, r_3), \) and \( (r_2, r_3) \), are obtained. From each set of HSL, without forgetting the associated constraints, a \( M_N \text{VL} \) is defined. The sweep of the \( M_N \text{VL} \) is performed hierarchically such as has been indicated in the previous subsection. The tetra-sphere finally determined has to be valid.

The function to minimize is the addition of the respective functions of the four tri-spheres that define the tetra-sphere:

\[
A_{S_{0123}} = A_{S_{012}} + A_{S_{013}} + A_{S_{023}} + A_{S_{0123}} \quad (20)
\]

Anyway, at low level, as centers do not change, the following function can be used:

\[
A'_{S_{0..(n-1)}} = \sum_{i=0}^{2} \sum_{j=i+1}^{3} A_{S_{ij}} \quad (21)
\]

### 3.4. n-sphere

This section is introduced to show the generalization of the presented method for the generation of the minimum outer s-tope enveloping an object represented by a set of points \( P \).

As an n-sphere \( S_{0..(n-1)} \) is defined by \( n \) spheres \( \{s_0, s_1, \ldots, s_{n-1}\} \), generating an n-sphere implies finding \( n \) centers and \( n \) radii. Therefore, the high level must deal with the n-sphere position \( \{c_0, c_1, \ldots, c_{n-1}\} \), while the low level, using the position received from the higher, finds \( n \) radii in such a way that the n-sphere generated have to minimally envelop \( P \) at \( \{c_0, c_1, \ldots, c_{n-1}\} \).

At low level, every \( p_i \in P \) to envelop is described by a set of parameters. One of these parameters, \( d_{ij} \), is the distance from \( p_i \) to the polytope defined by the centers. In order to compute this parameter, the technique presented in ref. 15 is used.

This algorithm ends providing a subset of \( k \) centers (as maximum four) and \( k - 1 \) parameters \( \lambda_{ij} \). These centers are the closer to \( p_i \). Definitely, the

#### Table I. Tri-spheres involved in the generation of the minimum outer tetra-sphere.

<table>
<thead>
<tr>
<th>( \lambda_{ij} )</th>
<th>( \lambda_{i2} )</th>
<th>( \lambda_{i3} )</th>
<th>( \Sigma \lambda_{ij} )</th>
<th>Tri-sphere involved</th>
</tr>
</thead>
<tbody>
<tr>
<td>( &lt; 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 1 )</td>
<td>( S_{023} )</td>
</tr>
<tr>
<td>( \geq 0 )</td>
<td>( &lt; 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 1 )</td>
<td>( S_{013} )</td>
</tr>
<tr>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( &lt; 0 )</td>
<td>( \geq 1 )</td>
<td>( S_{012} )</td>
</tr>
<tr>
<td>( &lt; 0 )</td>
<td>( &lt; 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 1 )</td>
<td>( S_{013} )</td>
</tr>
<tr>
<td>( &lt; 0 )</td>
<td>( \geq 0 )</td>
<td>( &lt; 0 )</td>
<td>( \geq 1 )</td>
<td>( S_{012} )</td>
</tr>
<tr>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 1 )</td>
<td>( S_{123} )</td>
</tr>
<tr>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 1 )</td>
<td>( S_{123} )</td>
</tr>
<tr>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 1 )</td>
<td>( S_{123} )</td>
</tr>
<tr>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 0 )</td>
<td>( \geq 1 )</td>
<td>( S_{123} )</td>
</tr>
</tbody>
</table>

Figure 8. Computational cost to explore the tri-sphere’s \( M_N \text{VL} \).
algorithm finishes providing
\[
\{c'_0, \ldots, c'_k\} \quad k \leq 3 \quad \lambda_{ij} > 0 \quad \sum_{j=1}^{k} \lambda_{ij} < 1, \]
\[
p_i^* = c'_0 + \sum_{j=1}^{k} \lambda_{ij} (c'_j - c'_0) \quad d_i = \| p_i - p_i^* \| \quad (22)
\]

If \( k = 3 \), i.e., four centers have been returned, then the point is inside the polytope defined by such centers, and so the point is already enveloped and consequently discarded. If \( k = 0 \), then parameter \( d_i \) is used to determine the associated constraint \( d_{ij}^\text{min} \). Otherwise, according to the \( \lambda_{ij} \) and \( d_i \) obtained, each point will be transformed into the corresponding HSL. Notice that a \( p_i \in P \) is transformed, as a maximum, into two HSLs.

Once all \( p_i \in P \) have been treated, \( \sum_{j=1}^{n-1} j \) sets of HSLs and the constraints \( d_{ij}^\text{min} \) with \( j = 0, 1, \ldots, n-1 \) are determined. From each set of HSLs, taking into account the relative constraints, a \( M_n \) VL is defined. The sweep of the \( M_n \) VL is performed hierarchically.

By recursion, the function to minimize is the addition of the ones associated with the \( n \) \((n-1)\)-spheres that define an \( n \)-sphere:
\[
A_{S_{n-(n-1)}} = \sum_{i' = 0}^{1} \sum_{i'' = i' + 1}^{2} \cdots \sum_{j = (n-2) + 1}^{n-1} A_{S_{i'j''}} \quad (23)
\]

At low level, as centers do not change, this function can be used:
\[
A'_{S_{n-(n-1)}} = \sum_{i = 0}^{n-2} \sum_{j = i + 1}^{n-1} A_{s_{ij}} \quad (24)
\]

**4. Maximum Inner S-tope**

It is extensively known that the availability of the maximum inner model together with the minimum outer one allows a better reliability in the collision detection process. Therefore, it is completely justified to develop the maximum inner model of an object in order to face robotic applications.

The presented method for the generation of the minimum outer s-tope is readapted to deal with the maximum inner s-tope case. Anyway, the analogy between both methods is the same that exists in any optimization technique. Obviously, each object to be modeled is represented by a set of points \( P \) taken from its surface.

In this way, the generation of the maximum inner s-tope is also focused on two hierarchical levels. High level, based on the downhill simplex, handles the s-tope position (centers), but in this case, its objective is to search the maximum s-tope. The initial condition \( C_\text{in} \) needed by this optimization method can be the same as the one computed for obtaining the minimum outer s-tope. But, in this case, the high level provides a set of centers totally included in \( P \). From these centers, the low level finds the radii of the maximum inner s-tope at the mentioned position.

Now, as it was presented at the previous section, the low level procedure is going to be introduced from particular to general in the following subsections.

**4.1. Bi-sphere**

Once the centers \( \{c_0, c_1\} \) have been received from the higher, low level has to find the two radii \( \{r_0, r_1\} \) of the maximum bi-sphere that does not envelop any \( p_i \in P \). For that, every \( p_i \in P \) is described by two parameters \( (\lambda_i, d_i) \). These parameters are obtained in the same way as the minimum outer bi-sphere. According to its \( \lambda_i \), each \( p_i \) is classified into one of the sets IWP, OWP\(^{c_0}\), or OWP\(^{c_1}\). Two parameters \( d_{ij}^\text{max}, d_{ij}^\text{max} \) are defined from the outward points
\[
\forall p_i = (\lambda_i, d_i) \in \text{OWP}^{c_0} \rightarrow d_{ij}^\text{max} = \min_i \{d_i\}
\]
\[
\forall p_i = (\lambda_i, d_i) \in \text{OWP}^{c_1} \rightarrow d_{ij}^\text{max} = \min_i \{d_i\} \quad (25)
\]
where \( d_{ij}^\text{max} \) are the maximum radii that do not envelop any OWP\(^{c_i}\). If some subset OWP\(^{c_i}\) is empty, the associated constraint has no sense and so it has not to be considered.

Consequently, with just the constraints \( r_j = d_{ij}^\text{max} \) some IWP can be enveloped. To ensure that no IWP are enveloped, every \( p_i(\lambda_i, d_i) \in \text{IWP} \) is transformed into a Hough straight line as is indicated in (6), but now \( r_j \leq d_{ij}^\text{max} \)—if appropriate.

According to the third HSL property, it can be stated that a HSL associated with a given \( p_i \in \text{IWP} \) represents a set of radii \( \{r_0, r_1\} \) that minimally envelops \( p_i \). In other words, that does not maximally envelop \( p_i \). If no IWP have to be enclosed by the desired bi-sphere, all the HSLs have to be taken into account. Therefore, considering all the HSL and
constraints \( d_0^{\text{max}}, \) we call the maximum volume locus (MVL) the border formed by the intersections of all the internal half-planes defined by all HSLs. Figure 9 shows different examples of MVL drawn by thick stroke lines.

MVL has the following properties:

1. Every point \((r_0^b, r_1^b)\) at the MVL represents a bi-sphere that does not envelop \( P \). This is a consequence of the last property of the HSL. Therefore, the maximum inner bi-sphere at \((c_0, c_1)\) will be obtained sweeping MVL from \( A \) to \( B \).
2. Every \( p_i \in \text{IWP} \) whose HSL does not belong to the MVL has no influence on the search of the maximum inner bi-sphere at \((c_0, c_1)\).
3. Every \( p_i \in \text{IWP} \) whose HSL is totally excluded from the area formed by the \( r_0, r_1 \) axis and the straight lines \( r_0 = d_0^{\text{max}}, r_1 = d_1^{\text{max}} \)—if appropriate—is not enveloped at \((c_0, c_1)\) by the bi-sphere of radii \( \{d_0^{\text{max}}, d_1^{\text{max}}\} \).

Then, sweeping the MVL from \( A \) to \( B \), the maximum inner bi-sphere at \((c_0, c_1)\) is obtained.

The function to maximize is \( A_{s_{ij}} \) (11). Discarding the exhaustive sweep it is important to consider the following property.

**Lemma 1:** \( A_{s_{ij}} \) is a concave function for all \( r_0, r_1 \) when they are related by an HSL, and the valid bi-sphere condition \( |r_1 - r_0| < l_{ij} \) is verified.

The proof is trivial. Substituting \( r_1 \) for its value given by HSL, the second derivative with respect to \( r_0 \) is always strictly greater than zero.

Therefore, according to Lemma 1, the maximum of \( A_{s_{ij}} \) only can be found at the extremes \( A, B \) or at any point in the MVL where one HSL changes to another. Searching finishes with the maximum \( r_0 \) tolerance; i.e., the maximum bi-sphere in MVL is exactly found at \((c_0, c_1)\). To quickly compute the HSL changes in MVL, the following steps are performed:

1. HSLs are sorted from lower to greater intercept, i.e., the intersection with \( r_0 = 0 \) axis. The set of sorted HSLs is referred to as VAL. VAL includes all HSLs with possibilities of belonging to the MVL.
2. State the searching interval, \( A = (r_0^A, r_1^A) \) and \( B = (r_0^B, r_1^B) \). These margins depend on the existence of the constraints \( d_j^{\text{max}} \). The following situations must be distinguished:
   a. OWP \( \notin \emptyset \). It allows one to determine \( A \). Then
      \[
      r_1^A = \min \left\{ d_1^{\text{max}}, \min_{i \in \text{VAL}} \left( \frac{d_i}{\lambda_i} \right) \right\} \quad (26)
      \]
      Notice that as VAL is sorted, the minimum \((d_i/\lambda_i)\) is given by the first HSL in VAL. As \( r_1^A \) depends on \( r_0^A \), if \( r_1^A \) is the minimum HSL intercept, \( r_0^A \) is set to zero, while if \( r_1^A = d_1^{\text{max}} \) then \( r_0^A \) is obtained from the HSL with lowest intersection with \( r_1 = d_1^{\text{max}} \). So,
      \[
      r_0^A = \min_{i \in \text{VAL}} \left( \frac{d_1 - \lambda_i d_i^{\text{max}}}{1 - \lambda_i} \right) \quad (27)
      \]
      where \( r_0^A \) is always positive. \( r_0^A \) has to verify \( r_0^A \leq d_0^{\text{max}} \) when OWP \( \notin \emptyset \). If that condition is not verified the MVL is formed by a unique point \((d_0^{\text{max}}, d_1^{\text{max}})\).
   b. OWP \( \emptyset \). According to the first HSL property, those HSLs whose intercepts are lower than the
intercept of the HSLs that satisfy (27) have to be removed from VAL, because they do not belong to the $M_\chi{}$VL.

b. $OWP^{c_0} = \emptyset$. As $d_0^{max}$ has no sense, then $A = (0, \min_i \in VAL \{d_i / \lambda_i\})$.

c. $OWP^{c_1} \neq \emptyset$. It lets one obtain $B = (r_0^B, r_1^B)$. $r_0^B$ is

$$r_0^B = \min \left\{ d_0^{max}, \min_i \in VAL \left\{ \frac{d_i}{1 - \lambda_i} \right\} \right\} \quad (28)$$

On the one hand, if $r_0^B = d_0^{max}$, then $r_1^B$ is the lowest HSL intersection with $r_0 = d_0^{max}$

$$r_1^B = \min_i \in VAL \left\{ \left(1 - \frac{1}{\lambda_i} \right) d_0^{max} + \frac{d_i}{\lambda_i} \right\} \quad (29)$$

where if $OWP^{c_1} \neq \emptyset$, $r_1^B$ has to verify $r_1^B \leq d_0^{max}$; otherwise $M_\chi{}$VL is formed by a unique point $(d_0^{max}, d_1^{max})$. On the other hand, if $r_0^B$ is the minimum intersection with the $r_1 = 0$ axis, $r_0^B$ is set to zero.

Due to the first HSL property, those HSL $\in VAL$ whose intercepts are greater than the intercept that finally defines $B$ are removed from VAL, because they do not belong to the $M_\chi{}$VL.

d. $OWP^{c_2} \neq \emptyset$. As $d_0^{max}$ has no sense, then $B = (\min_i \in VAL \{d_i / (1 - \lambda_i)\}, 0)$. Moreover, the HSL whose intercepts are greater than the intercept of the HSL that gives the value to $r_0^B$ are removed from VAL, because these HSLs do not belong to the $M_\chi{}$VL.

3. Find changes of HSL in the $M_\chi{}$VL. Assuming that $r_0^{\text{new change}}$ is the abscissa of the last detected HSL change, the new $r_0^{\text{new change}}$ is obtained:

$$r_0^{\text{new change}} = \min_{i \neq 1} \left\{ \frac{d_i \lambda_i - d_1 \lambda_1}{\lambda_i - \lambda_1} \right\}$$

$$r_0^{\text{new change}} \in ]r_0^{\text{change}}, r_0^B[ \quad (30)$$

where $(\lambda_i, d_i)$ represents, at each moment, the first HSL in VAL. (30) looks for the intersections of $(\lambda_i, d_i)$ with the rest of HSL in VAL. The HSL $(\lambda_i, d_i)$ that originates $r_0^{\text{new change}}$ is converted into the first in VAL, removing the previous HSL.

After these three steps, the algorithm for exploring the $M_\chi{}$VL is stated as

```c
// f represent $A_{r_0}$ function.
VAL = Sort(OWP)
VAL = Compute(A, B, VAL) // Remove HSL, if $f$ $max$ = $max$($f(A), f(B)$) appropriate
$\text{r}_0^{\text{change}} = r_0^A$
while ($r_0^{\text{change}} < r_0^B$) and (VAL $\neq \emptyset$) then
$\text{r}_0^{\text{new change}} = \min_{i \in VAL \{r_0^B, r_0^{\text{change}}\}} \left\{ \frac{d_i \lambda_i - d_1 \lambda_1}{\lambda_i - \lambda_1} \right\}$
if ($r_0^{\text{new change}} \in [r_0^{\text{change}}, r_0^B]$) then
VAL = Set_first($\lambda_i, d_i$) // ($\lambda_i, d_i$) satisfies (30)
$\text{r}_0^{\text{change}} = r_0^{\text{new change}}$
$\text{r}_1^{\text{change}} = \left(1 - \frac{1}{\lambda_1}\right) r_0^{\text{change}} + \frac{d_1}{\lambda_1}$
$f$ $max$ = $max$($f$ $max$, $f((r_0^{\text{change}}, r_1^{\text{change}}))$
else return $f$ $max$
endif
endwhile
return $f$ $max$
```

The computational cost required to explore the $M_\chi{}$VL is shown in Figure 10. The algorithm has been implemented in C and tested on a Pentium® II 350 MHz. The set of HSLs has been randomly generated.

4.2. Tri-sphere

As it is defined by three spherical vertices $n = 3$, high level deals with three centers $(c_0, c_1, c_2)$. At each position received, low level determines the radii $(r_0, r_1, r_2)$ of the maximum inner tri-sphere that does not envelop any $p_i \in P$.

![Figure 10. Computational cost for exploring bi-sphere’s $M_\chi{}$VL.](image-url)
At low level, \( p_i \in P \) are represented by three parameters \( p_i = (\lambda_{i1}, \lambda_{i2}, d_i) \) such as is done in the minimum outer tri-sphere search. \( \lambda_{i1} \) and \( \lambda_{i2} \) state the relative location of \( p_i \) with respect to the centers. So, according to the mentioned parameters, \( p_i \in P \) are handled just in the same way as in the minimum outer case.

Anyway, there is an important difference when determining the constraints \( d_{ij}^{\text{max}} \) instead of \( d_{ij}^{\text{min}} \) with \( j = 0, 1, 2 \). In this way, all the expressions related to the mentioned constraints have to be replaced with

\[
d_{ij}^{\text{max}} = \min\{d_{ij}^{\text{max}}, d_i\} \quad j = 0, 1, 2
\]

(31) is applied to all the points included in the subsets \( \text{OWP}^{\pi_i} \). When an \( \text{OWP}^{\pi_i} \) is empty, its associated \( d_{ij}^{\text{max}} \) has no sense and so it is not used.

After considering every \( p_i \in P \), on the one hand, three sets of HSLs are obtained, related to the radii \( (r_0, r_1), (r_0, r_2) \), and \( (r_1, r_2) \). On the other hand, some of the constraints \( r_j \leq d_{ij}^{\text{max}} \) are determined. For each of the three sets of HSLs together with their constraints—if appropriate—three \( M_X \text{VL} \) are built.

First two \( M_X \text{VL} \) are simultaneously swept by \( r_0 \), just as has been indicated in the maximum inner bi-sphere’s generating process. Consequently, a combination of radii \( \{r_0, r_1, r_2\} \) is obtained. To ensure that the points transformed into the \( (r_1, r_2) \) HSL are enveloped, the third \( M_X \text{VL} \) is considered. This step consists of verifying if the provided \( (r_1, r_2) \) are located at the intersection of all the internal half-planes formed by the correspondent HSL or exactly at the mentioned \( M_X \text{VL} \). If such verification is false, then some point is being enveloped (third HSL property). Therefore, \( r_2 \) has to be decreased until the new \( (r_1, r_2) \) matches with this third \( M_X \text{VL} \) and consequently, no \( p_i \in P \) is enclosed. The radii finally obtained have to define a valid tri-sphere.

The maximum inner tri-sphere is going to be found evaluating the function 18 either at those points in the first two \( M_X \text{VL} \) where there is a change of HSL (including the extreme limits of the \( r_0 \) interval of search) or just the \( (r_0, r_1, r_2) \) where it is not needed to reduce \( r_2 \) anymore, if exits. Obviously, this \( (r_0, r_1, r_2) \) is located between the last change point where a reduction in \( r_2 \) was needed and the first change point where such a reduction has not been necessary.

Calling \( r_{0 \text{last-change}} \) the \( r_0 \) of the last change point evaluated where \( r_2 \) was reduced and \( r_{0 \text{new-change}} \) the \( r_0 \) of the first evaluated \( (r_{0}, r_{1}, r_{2}) \) where \( r_2 \) has not been reduced, the \( r_0 \) of the point where reducing \( r_2 \) will not be needed anymore is computed by

\[
r_0 = \frac{m_{12}b_{01} + b_{12} - b_{02}}{m_{02} - m_{12}m_{01}} \in ]r_{0 \text{last-change}}, r_{0 \text{new-change}}[
\]

where \( m_{ijk} \) represents the slopes of the \( (r_{i}, r_{k}) \) HSLs that belong to the associated \( M_X \text{VL} \) limited by \( r_{0 \text{last-change}} \) and \( r_{0 \text{new-change}} \), while \( b_{ijk} \) describe the intercepts.

Anyway, it has to be taken into account that all the sets of radii \( (r_0, r_1, r_2) \) have to evaluate have to verify the bi-spheres constraints \( |r_1 - r_0| < l_{01}, \ |r_2 - r_0| < l_{02}, \ |r_2 - r_1| < l_{12} \) and the constraints \( r_j \leq d_{ij}^{\text{max}} \)—if \( d_{ij}^{\text{max}} \) exist—with \( j = 0, 1, 2 \).

Figure 11 shows the computational cost required to explore the tri-sphere’s \( M_X \text{VL} \). The algorithm has been implemented in C and run on a Pentium® II 350 MHz. The set of HSLs has been randomly generated.

### 4.3. Tetra-sphere

To generate the maximum inner four-order s-tope, it is necessary to find four spheres with the constraint that no \( p_i \in P \) has to be enveloped. High level deals with the tetra-sphere centers, while low level has to determine the suitable set of radii.

At low level, every \( p_i \in P \) is represented by four parameters \( p_i = (\lambda_{i1}, \lambda_{i2}, \lambda_{i3}, d_i) \) such as is done in the minimum outer tetra-sphere search. If there is some point verifying \( \lambda_{i1} + \lambda_{i2} + \lambda_{i3} \leq 1 \) and \( \lambda_{ij} \geq 0 \),
i.e., such a point is inside the tetrahedron formed by the centers and so the low level has to exit with error.

Then, \( p_i \in P \) are handled exactly as it is indicated in Table I. After considering all \( p_i \in P \), six sets of HSLs together with the constraints \( d_{j}^{\text{max}} \), with \( j = 0, \ldots, 3 \), are obtained. From each set of HSLs, taking into account the associated \( d_{j}^{\text{max}} \)—if appropriate—a MVL is defined. The sweep of the MVL is performed hierarchically such as has been indicated in the maximum inner tri-sphere’s subsection. The tetra-sphere finally obtained has to be valid. The function to maximize is the same indicated in (20) or (21).

4.4. n-sphere

The process for generating the maximum inner s-tope is presented in this subsection. High level searches for the position of the maximum inner n-sphere’s position, while low level’s objective is to find the radii that define the maximum inner n-sphere for all sets of centers provided by higher one.

At low level, as is indicated in the minimum outer s-tope’s subsection, an algorithm\(^{15} \) is applied to every \( p_i \in P \) returning the information shown in (22). If after applying\(^{15} \) it to a \( p_i \in P \), the number of centers \( k \) returned is \( k = 3 \), then that point is inside the polytope defined by the centers and so low level exits with error. After finishing this process \( \Sigma_{j=1}^{n-1} \) sets of HSLs and some constraints \( d_{j}^{\text{max}} \), with \( j = 0, 1, \ldots, n - 1 \) are obtained. Using this information, from each set of HSLs a MVL is built. The radii of the maximum inner s-tope that does not envelop \( P \) are determined by sweeping hierarchically the MVL such as is indicated in the subsection related to the generation of the maximum inner tri-sphere.

The function to maximize is indicated in (23) or (24). Anyway, it has to be considered that every set of radii \( \{r_0, r_1, \ldots, r_{n-1}\} \) to evaluate has to verify, at each step, the constraints \( r_j \leq d_{j}^{\text{max}} \)—if they exist—with \( j = 0, 1, \ldots, n - 1 \).

5. MODELING ACCURACY

Given that the only information available from the object to model is a set of points \( P \) taken from its surface, the parameter to quantify the accuracy of the real-object representation will depend on the \( P \).

Having generated both the minimum outer and maximum inner n-sphere to model an object, the accuracy parameter \( \rho \) is the ratio between the dimensions of both models:

\[
\rho = \frac{\sum_{j=0}^{n-1} r_{j}^m}{\sum_{j=0}^{n-1} r_{j}^M}
\]

where superscripts \( m \) and \( M \) refer to the maximum inner and the minimum outer n-sphere, respectively. Accuracy is limited, i.e., \( \rho \in [0, 1] \), being maximum when it tends to one. As objects can be hierarchically modeled, \( \rho \) is also used to quantify what are acceptable bi-sphere, tri-sphere, or n-sphere models to represent them, and even more, what model best fits to our object.

Additionally, \( \rho \) may be interpreted as a degree of no-convexity of the object being modeled. If \( \rho \) is close to zero, for all n-sphere representations \( n = 0, 1, \ldots \), it implies that both maximum inner and maximum outer, obtained from the same \( P \), are appreciably different. Therefore, according to the information available, it can be concluded that the object dealt with is highly non convex.

6. MODELING RESULTS

In this section, some results of the geometric modeling are introduced. Geometric modeling tests were performed on an ABB IRB6. This manipulator is mainly formed by parallelepiped links, bi-spheres being generally the fittest s-topes to model it. Therefore, only such representation results are presented.

Upon taking some representative points of its surface, the whole manipulator was modeled. Figure 12 shows the minimum outer and maximum inner bi-spheres.
inner bi-spheres obtained, the modeling accuracy being $\rho = 0.18$. Obviously, this value invites one to model separately each link of the manipulator.

Figure 13 shows the respective minimum outer and maximum inner bi-spheres obtained separately each link. In this case, the average accuracy is $\rho = 0.77$. All the representations in Figure 13 were computed in 0.25 s (with 800 points) in a Pentium® II 350 MHz.

Using just bi-spheres, notice that this parameter can be even more increased, decomposing properly determined links again.

7. CONCLUSION

A new method for geometrical modeling of robotic systems has been presented. This method has been focused on generating both the minimum outer and the maximum inner spherically extended polytopes (s-tope) to model a given object represented by a set of characteristic points taken from its surface. The method is divided into two levels. High level is based on the downhill simplex method, while the lower consists of an application of the Hough transform. Low level’s objective is to generate the minimum and the maximum volume loci, where it is respectively located the radii of the minimum outer and the maximum inner s-tope whose spatial positions are known and, in this sense, this method has to be considered as optimal. The geometric modeler can be applied both on-line and off-line, so it can be considered as an automatic valuable tool for collision detection in robot motion planning.24

REFERENCES