Proceedings of the 2000 IEEE
International Conference on Robotics & Automation
San Francisco, CA • April 2000

Real-Time Generation of Collision-Free Paths for a Mobile Sphere

Enrique J. Bernabeu Josep Tornero
Departmento de Ingeniería de Sistemas y Automática
Universidad Politécnica de Valencia
Valencia, POB 22012, SPAIN

Abstract

A novel and fast technique for solving the basic motion-planning problem is introduced in this paper. Obstacle avoidance is based on the computation of minimum translational distances that allows to obtain collision-free intermediate configurations. When path planning is constrained to two degrees of freedom, Hough Transform is applied to such configurations. After that, several collision-free paths are generating by joining different sets of selected intermediate configurations. These selections are obtained by applying the technique called Minimum Volume Locus. Complexity of this path planner is linear with the number of obstacles. Object representation is based on spherically extended polytopes.

1 Introduction

In this paper, path-planning problem is addressed: given a mobile sphere, a start and goal configurations and a set of obstacles, find a (or more) collision-free path joining the extreme configurations.

Obstacles are modeled by spherically extended polytopes (s-tope) [7]. A s-tope is the convex hull containing an infinite set of swept spheres. Therefore, volume swept by the mobile sphere is also modeled by a s-tope.

Minimum translational distances are computed between the obstacles and the s-tope modeling the straight-line path joining the start and goal configurations. The most common measure of translational distance is given in [4]. Geometrically, this measure corresponds to the shortest translation, in all possible directions, such that both objects evolve to a contact state.

Applying the proposed translational distances, intermediate collision-free configurations are obtained. When an obstacle is being collided, the associated intermediate configuration states a new mobile position where such an obstacle is properly avoided.

The approach based on searching over not collision-free paths has been previously used by [9]. These path-planning methods are best suited for industrial applications because of initial paths that connect both extreme configurations are known: sometimes, the path is supplied by an operator; sometimes, a previous collision-free path that is no longer so because of minor spatial modifications of the obstacles.

When path planning is constrained to two degrees of freedom, it is presented an algorithm whose complexity is linear with the number of obstacles. The technique consists of transforming each intermediate configuration into a straight line by applying the Hough transform [8]. Even more, this planner is able to generate more than one collision-free path by applying only one intersection test per obstacle.

Using the mentioned straight lines, a Minimum Volume Locus (MVL) is build. MVL is introduced in [1,2,3] for obtaining the minimum outer s-tope to model a given object. Nevertheless, MVL's properties has been approached in this paper in order to determine what intermediate configurations are the strictly needed for generating a collision-free path passing through such configurations.

The remainder of the paper is organized as follows. In Section 2, Minimum Volume Locus technique is described. Computation of minimum translational distances is shown in Section 3. Path planning in three-dimensional space is presented in Section 4, while path-planning method in bi-dimensional space is introduced in Section 5. Finally, conclusions are listed in Section 6.

2 Spherically Extended Polytopes and Minimum Volume Locus

The aim of this section is mainly concerned with the description of the Minimum Volume Locus based on the generation of spherically extended polytopes [1,2,3].

A spherically extended polytope (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 its radius. Given the set of spheres $S=\{s_0,s_1,\ldots,s_n\}$, the convex hull of such a set, S_S, contains an infinite set of swept spheres expressed by

$$S_S = \left\{ s : s = s_0 + \sum_{i=1}^{n} \lambda_i (s_i - s_0), s_i \in S, \lambda_i \geq 0, \sum_{i=1}^{n} \lambda_i \leq 1 \right\} \quad (1)$$

The convex hull of spheres does not include all possible spheres which can fit inside the s-tope, only those that are generated by (1). Spheres in S are called spherical vertices. S-tope order is the number of spherical vertices. Graphical examples of s-topes are shown along the paper.

As a polytope with four or more points is a polyhedral object with triangular facets, tetra-spheres (or greater-order s-topes) are objects composed of tri-sphere facets [7].

An automatic s-tope generator has been developed based on an iterative process that is focused on two hierarchical levels [1,2,3]. S-topes are generated with the constraint of enveloping a set of points P taken from the surface of the object to model. The function to minimize is the s-tope volume. High level searches the optimal centers position of

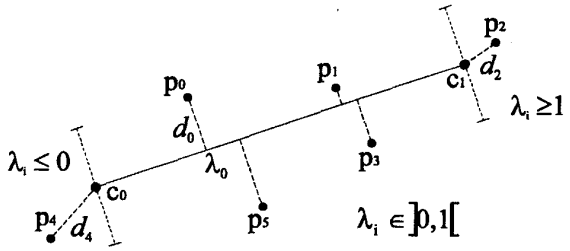

Figure 1. Relationship to p_i distribution and λ_i

the spherical vertices. Low level, using the centers received from the high one, finds a set of suitable radii, defining a s-tope that minimally envelops P. The low level is based on the application of the Hough transform.

MVL contains the spherical-vertex radii of the minimum s-tope for modeling a given object [1,2,3]. Now, this technique is used for determining what obstacles have to be considered for generating a collision-free path. In order to explain the concept of MVL, a simple generation-model based on bi-spheres as low level description is considered.

The two-level hierarchical structure works as follows: at low level the information available are the centers $\{c_0,c_1\}$ and a set of points P (object representation). In order to find the suitable radii $\{r_0,r_1\}$, each $p_i \in P$ is described by two parameters λ_i, d_i. λ_i represents the p_i's position respect the segment formed by c_0 and c_1, while d_i is the distance from p_i to such a segment. λ_i is obtained by solving the following equations:

$$\begin{aligned} p_i^{\perp} &= c_0 + \lambda_i (c_1 - c_0) \\ (p_i^{\perp} - p_i)\cdot(c_1 - c_0) &= 0 \end{aligned} \qquad (2)$$

where $p_i^{\perp}$ is the projection of p_i onto the straight line defined by the centers. According to $p_i=(\lambda_i,d_i)$ with $\lambda_i \notin]0,1[$ (see figure 1), radii have to be constrained to $r_0 \geq d_0^{min}$ and $r_1 \geq d_1^{min}$. These constraints are obtained as follows,

$$\begin{aligned} \forall p_i = (\lambda_i,d_i) : \lambda_i \leq 0 \;&\rightarrow\; d_0^{min} = \max_i \{d_i\} \\ \forall p_i = (\lambda_i,d_i) : \lambda_i \geq 1 \;&\rightarrow\; d_1^{min} = \max_i \{d_i\} \end{aligned} \qquad (3)$$

The pairs of radii needed to envelop minimally the rest of $p_i \in P$, with $\lambda_i \in]0,1[$, are locate at a straight line (see figure 2). This straight line is obtained as follows:

$$\frac{r_1 - r_0}{1} = \frac{d_i - r_0}{\lambda_i} \rightarrow r_1 = \left(1 - \frac{1}{\lambda_i}\right) r_0 + \frac{d_i}{\lambda_i} \;;\quad r_0, r_1 \geq 0 \qquad (4)$$

This straight line is called Hough straight line (HSL). HSL has the following properties:

1) Slope is always negative.

2) HSL intersections with the r_0=0 and r_1=0 axis are always greater or equal to zero.

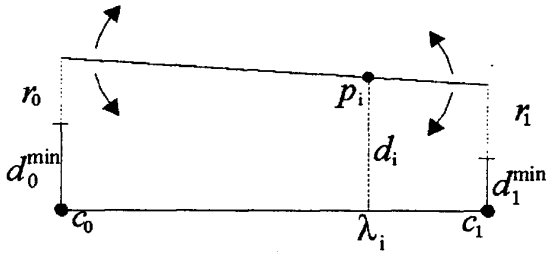

Figure 2. (r_0,r_1) enveloping p_i

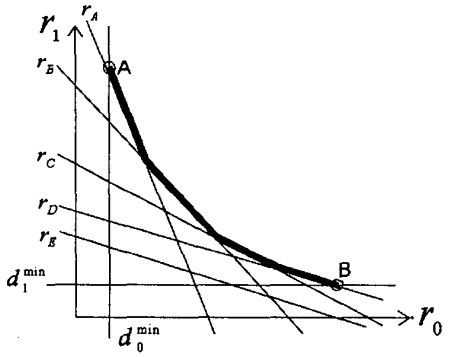

Figure 3. Minimum Volume Locus (thick stroke line)

3) HSL divides the r_0r_1-plane (parameter space) into two half-planes which are respectively called external and internal half-planes. The external half-plane verifies

$$\forall p_i = (\lambda_i,d_i) : \lambda_i \in]0,1[,\; r_1 > \left(1 - \frac{1}{\lambda_i}\right) r_0 + \frac{d_i}{\lambda_i} \qquad (5)$$

Any point (r_0^j,r_1^j) at this half-plane represents a bi-sphere enclosing p_i. The proof is trivial and it is deduced from comparing expression (4) and (5) at (r_0^j,r_1^j). Analogously, the internal half-plane represents bi-spheres too small to envelop p_i.

Considering all the HSL and the constraints d_0^{min}, d_1^{min}, we call Minimum Volume Locus the border defined by all the external half-plane intersections, see figure 3.

MVL verifies the following properties:

1) Any point (r_0^j,r_1^j) located at the MVL represents a bi-sphere at $\{c_0,c_1\}$ enveloping every $p_i \in P$. This is a direct consequence of property 3 of HSL.

2) Every $p_i \in P$, whose associated HSL does not belong to MVL, is irrelevant and has no influence on the search of the minimum bi-sphere. See r_E in figure 3.

3) When a HSL is totally included in the area bounded by $r_0 = d_0^{min}$, $r_1 = d_1^{min}$ and axis r_0=0, r_1=0, its associated $p_i \in P$ is already enveloped by the bi-sphere with radii $r_0 = d_0^{min}, r_1 = d_1^{min}$.

4) The minimum bi-sphere located at $\{c_0,c_1\}$ is found sweeping the MVL.

3 Intersection Tests. Translational Distances

The proposed method for distance computation between s-topes is based on the application of the well-known GJK algorithm [6] that computes the distance between two polytopes as the separation between the origin point O and their Minkowski difference.

In this sense, the Minkowski difference s-tope of two s-topes S_A, S_B, defined respectively by the sets of spheres A and B, is given in [7],

$$S = S_A - S_B = \{s=(c,r): c = c_A - c_B, r = r_A + r_B, \\ (c_A,r_A)\in A, (c_B,r_B)\in B\} \quad (6)$$

As GJK algorithm deals with polytopes, only the set of the centers of the spherical vertices is provided. GJK returns

$$S' = \{c'_0, c'_1, ...c'_l\};\ l \le 3;\ O^{\perp} = c'_0 + \sum_{j=1}^{l} \lambda_{Oj}(c'_j - c'_0) \\ \lambda_{Oj} \in [0,1];\ \sum_{j=1}^{l} \lambda_{Oj} \le 1\ ;\ d_O = \| O^{\perp} \| \quad (7)$$

where S' is a subset of as maximum four centers (spheres). $O^{\perp}$ is the projection of the origin point onto the structure defined by the centers and d_O is the resulting distance. Minimum translational distance (MTD) between two s-topes is then computed according to l.

a) If l=0, i.e. $S' = \{c'_0\}$, then

$$MTD(O, S_A - S_B) = d_O - r'_0\ ; \qquad \hat{v}_{MTD} = \frac{O^{\perp}}{d_O} \quad (8)$$

where r'_0 is the radius of the sphere whose centers is c'_0. $\hat{v}_{MTD}$ states the translational vector of the MTD. Sign of MTD always codifies the relationship between s-topes

$$sign(MTD) = \begin{cases} 1, & MTD > 0 \rightarrow \text{Separation} \\ 0, & MTD = 0 \rightarrow \text{Contact} \\ -1, & MTD < 0 \rightarrow \text{Penetration} \end{cases} \quad (9)$$

b) If l=1,2 then

$$MTD(O, S_A - S_B) = d_O - \left(r'_0 + \sum_{j=1}^{l} \lambda_{Oj}(r'_j - r'_0) \right) \quad (10)$$

r'_j, j=0,1,2 are respectively the radii of the spheres whose centers has been returned in S'. $\hat{v}_{MTD}$ is obtained as (8).

c) If l=3, i.e. $S' = \{c'_0, c'_1, c'_2, c'_3\}$ implies that the O is inside the structure defined by the centers and then a collision is presented [6]. Consequently, MTD is computed as

$$MTD(O, S_A - S_B) = -min\{MTD^+(O, S'_{012}), \\ MTD^+(O, S'_{013}), MTD^+(O, S'_{023}), MTD^+(O, S'_{123})\} \quad (11)$$

$MTD^+(O, S'_{ijk})$, with S'_{ijk} being an extern facet of S_A-S_B, is obtained applying b) but considering the spheres $\{s'_i, s'_j, s'_k\}$ and adding the radii expression instead of subtracting it. $\hat{v}_{MTD}$ is obtained as (8) but its sign is changed because MTD has been computed in the opposite sense.

MTD algorithm finishes returning as maximum two parameters λ_{Oj}.

This technique presents important advantages. Compared with [7], the computational cost required is improved more than 90%. With respect [6], our technique inherits all its advantages including the ending of the algorithm in a few iterations. Condition lost in [5]. Moreover, except when l=3, computational cost for obtaining penetration or separation distances is exactly the same.

When radii of the spherical vertices are zero (s-topes are then polytopes), MTD computation coincides with [6].

4 Moving a Sphere in a 3-D Space

As a consequence of the previously proposed method for computing minimum translational distances, it is introduced an algorithm for generating a collision-free path for a free-flying sphere based on penetration distances.

Algorithm requires start s_s and goal s_g configurations, an initial not collision-free path represented by the bi-sphere defined by the extreme-configuration spheres and all the obstacles modeled by means of s-topes.

Assuming that S_{sg} is the bi-sphere representing a given not-collision-free path, movement of the mobile sphere is parameterized as follows

$$S_{sg} = \{s=(c,r): c = c_s + \lambda(c_g - c_s),\ \lambda \in [0,1]\} \quad (12)$$

where c_s, c_g are respectively the sphere centers of the start and goal configurations. λ states the infinite intermediate position between start and goal configurations. Note that (12) verifies s-tope definition given in (1).

A collision-free path is obtained by means of applying intersection tests between the bi-sphere S_{sg} and the obstacles. After detecting a collision between S_{sg} and a given obstacle (MTD is negative), an intermediate collision-free configuration s_I=(c_I,r) is determined as follows (see figure 4)

$$c_I = c_s + \lambda(c_g - c_s) - \delta MTD\, \hat{v}_{MTD} \quad (13)$$

where MTD is the translational penetration distance, $\hat{v}_{MTD}$ is the corresponding vector, $\delta \ge 1$ states a safety threshold and λ represents the sphere in S_{sg} whose translation is minimal to avoid the involved obstacle and it is the result of the addition of λ_{Oj} parameters returned by MTD algorithm. Aditionally, $\hat{v}_{MTD}$ is constrained to be different from vector c_g-c_s, When the associated intermediate position belongs to S_{sg} then obstacle is not properly avoided.

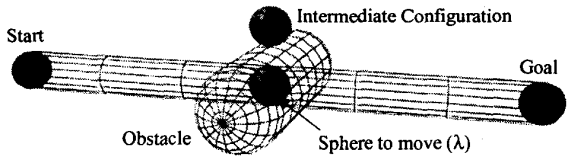

Figure 4. Obstacle Avoidance

Consequently, each obstacle is represented by three parameters: λ, MTD and $\hat{v}_{MTD}$. Obstacles are sorted according to their parameter $\lambda \in [0,1]$.

After determining a new intermediate configuration, two news paths are considered. One connecting the start position with the intermediate configuration, and the other joining the intermediate and the goal configurations. Process is then repeated recursively until no collision occurs. The recursive planner leads to the following algorithm.

```
function GetPath (s_s, s_g:configuration; λ[]:obstacles) {
  π ← {φ}
  Interference_test(s_s, s_g, λ)
  n ← number_of_collided_obstacles (sorted by their λ)
  if (collision) {
    for i ← 1,...,n   {
      if (i=1)    π ← GetPath(s_s, s_1^1 ,λ∈[0,λ^1])
      if (1<i<n) π ← Conc(π,GetPath( s_1^{i-1} , s_1^i ,λ∈[λ^{i-1},λ^i]))
      if (i=n)    π ← Conc(π,GetPath( s_1^{i-1} ,s_g,λ∈[λ^{i-1},1]))
    }
    return π
  }
  else return S_sg
}
```

GetPath generates a collision-free path between the given configurations. Obstacles to be considered are those whose λ belong to the provided λ-interval. λ^i is the corresponding parameter associated to the i^{th} collided obstacle (sorted). *Interference_test* computes MTD between the provided path and obstacles. *Conc* is just a concatenate procedure.

Recursivity in the path-planning algorithm provides smooth paths. This is also a consequence of s-tope-based world description (not sharp-cornered).

Algorithm always finds a collision-free path if distance among obstacles is greater than the diameter of the mobile sphere. If two obstacles are not enough separated, they are joined and considered as a sole obstacle (s-tope). This situation is easily detected when parameter λ of a collided obstacle verifies $\lambda \notin [0,1]$.

Figure 5 shows two paths generated by this algorithm in different environments. Paths have been computed in 570 μsec. and 490 μsec. respectively on a Pentium® II 350 MHz. The number of intermediate configurations generated (rendered in figure 5) has been 26 and 20 respectively.

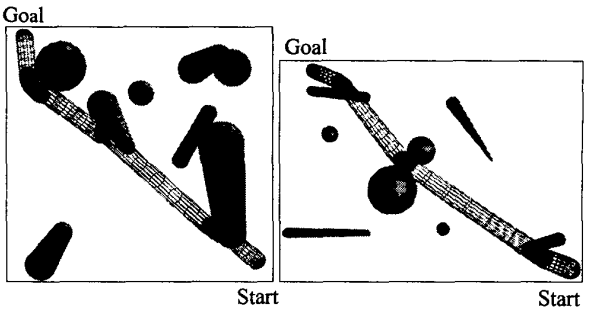

Figure 5. Two path planning examples for a free-flying sphere

5 Moving a Sphere in Bi-Dimensional Space

In spite of the fact that the method is defined in a three-dimensional space, a considerably improvement is obtained when movement is constraint to two degrees of freedom. In this case, computational cost of the path planner is linear with the number of obstacles.

In addition, and on the one hand, computational cost reduction comes from the MTD computation. GJK algorithm will never return four centers, i.e. l=3. Moreover, when l=2, as O is in the same plane that the spheres' centers, MTD is reduce to the computation of the minimum MTD^+ from O to each one of the corresponding external bi-spheres.

Given a s-tope obstacle S_{ob} defined by a set of spheres $\{s_0,s_1,...,s_n\}$ and a not collision-free path S_{sg}. Minkowski difference s-tope S_{sg}-S_{ob} is defined by $\{s_s\text{-}s_i, s_g\text{-}s_i\}$ with i=0,1,...n. When MTD is then computed, vectors of the centers of the involved bi-spheres of S_{sg}-S_{ob} are either $c_{sg}=c_g-c_s$ or $c_{ij}=\pm(c_j-c_i)$ with $i\neq j$ and $i,j=0,1,...n$.

As $\hat{v}_{MTD}$ is normal to the structure defined by the centers [6] and is constrained to be different from c_{sg}, MTD has to be computed from O to the closest external bi-spheres, whose centers hold the vector c_{sg}, in S_{sg}-S_{ob},

On the other hand, reduction in computational cost is obtained since it is only needed to apply one intersection test (MTD computation) per obstacle to generate a collision-free path. This is achieved by transforming each obstacle into a HSL (using the parameters λ and |MTD|). When all obstacles have been transformed into HSL, it is determined by applying the MVL properties, the minimum number of intermediate configurations needed for generating a collision-free path between the start and goal positions.

Figure 6 shows a path-planning example (dotted line), where parameters of the obstacles λ, MTD and $\hat{v}_{MTD}$ are also depicted (for more detail, see obstacle S_8 in figure 6). In order to avoid collision with an obstacle, it is needed to translate the mobile sphere to the configuration given by (13). Nevertheless, the mobile sphere does not have to be translated beyond the position given by (13), when dealing with collision-free obstacles (parameters represented by dashed lines). In this way, obstacle S_7 is collided because its maximum translation constraint has been violated.

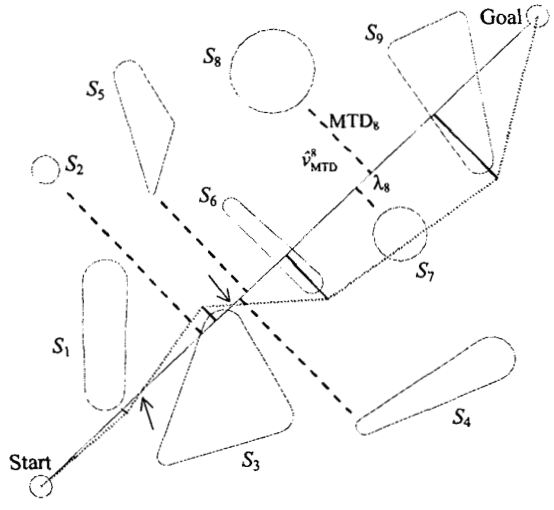

Figure 6. Depicting parameters λ, MTD *and* $\hat{v}_{MTD}$

Each obstacle is then transformed into a HSL as follows

$$r_1 = \left(1 - \frac{1}{\lambda}\right) r_0 + \frac{|MTD|}{\lambda} \quad \text{with } \lambda \in]0,1[\qquad (14)$$

Obstacles are classified into two types, collided ones, represented by MTD$^{<0}$, and collision-free ones, MTD$^{\geq 0}$. Next, c_{sg} axis is divided into sections, verifying that all $\hat{v}_{MTD}$ of the MTD$^{<0}$ obstacles have the same sign. Note that three sections (limited by arrows) are considered in figure 6.

Considering all the obstacles located at a given section, those MTD$^{\geq 0}$ obstacles, whose $\hat{v}_{MTD}$ sign is different from the $\hat{v}_{MTD}$ sign of the MTD$^{<0}$ ones, are rejected (obstacles S_5 and S_8 in figure 6). Using the HSL obtained from the MTD$^{<0}$ obstacles a MVL is defined. According to the first MVL and third HSL properties, it is stated that:

1) Those MTD$^{\geq 0}$ obstacles, whose associated HSL invade the intersection of all the external half-planes limited by the mentioned MVL, are rejected.

2) Intermediate configuration of the MTD$^{<0}$ obstacles that are defining the MVL are selected because they are strictly needed to generate the collision-free path.

The set of not considered obstacles is called Θ. If there are no MTD$^{\geq 0}$ obstacles in Θ, the process finishes, otherwise obstacles in Θ are selected or rejected according to the third MVL's property. Starting with the MTD$^{\geq 0}$ obstacles and following with the collided ones, next steps are applied: let C_i=(λ_i,MTD$_i$, $\hat{v}^i_{MTD}$) be an obstacle in Θ and let C_p=(λ_p,MTD$_p$, $\hat{v}^p_{MTD}$) and C_t=(λ_t,MTD$_t$, $\hat{v}^t_{MTD}$) be its immediately previous and following selected obstacles, then:

a) C_i is a MTD$^{\geq 0}$ obstacle. C_i is selected (removed from Θ) if only if its associated HSL is fully included in the area delimited by $r_0 = d_0^{min}$=MTD$_p$, $r_1 = d_1^{min}$=MTD$_t$ and axis r_0=0, r_1=0. Otherwise, C_i is rejected. Formally, if

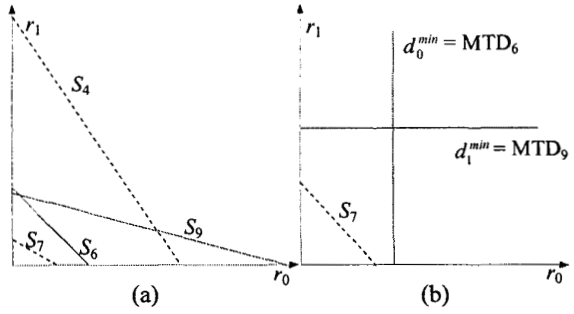

Figure 7. MVL applied to select intermediate configurations

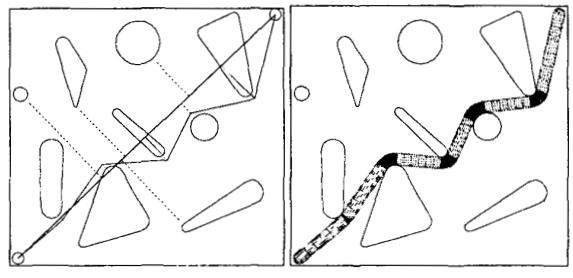

Figure 8. Intermediate configurations for a collision-free path

$$|MTD_t| > \left(1 - \frac{1}{\bar{\lambda}_i}\right)|MTD_p| + \frac{MTD_i}{\bar{\lambda}_i};\ \bar{\lambda}_i = \frac{\lambda_i - \lambda_p}{\lambda_t - \lambda_p} \qquad (15)$$

b) C_i is a MTD$^{<0}$ obstacle. C_i is selected (removed from Θ) if only if its associated HSL invades the area delimited by $r_0 = d_0^{min}$=MTD$_p$ and $r_1 = d_1^{min}$=MTD$_t$. Otherwise, C_i is rejected. Formally, if

$$|MTD_t| < \left(1 - \frac{1}{\bar{\lambda}_i}\right)|MTD_p| + \frac{|MTD_i|}{\bar{\lambda}_i};\ \bar{\lambda}_i = \frac{\lambda_i - \lambda_p}{\lambda_t - \lambda_p} \qquad (16)$$

A collision-free path is obtained passing through the intermediate configurations of the selected obstacles.

Figure 7 shows the selection/rejection process of obstacles of the third section in figure 6. Obstacles belonging to this section are S_4, S_5, S_6, S_7, S_8, S_9. S_5 and S_8 are rejected because their $\hat{v}_{MTD}$ are different from MTD$^{<0}$ obstacles ones. After transforming obstacles into HSL, a MVL is defined by the MTD$^{<0}$ obstacles (see S_6 and S_9 in figure 7a). According to step 1) obstacle S_4 is rejected, while S_6 and S_9 are selected by 2). Finally, obstacle S_7 is selected by step a).

Consequently, the generated collision-free path presents as intermediate positions those configurations associated with obstacles S_1, S_3, S_6, S_7 and S_9 (see figure 8a). Computational cost required for obtaining the above-mentioned intermediate configurations has been 88 μsec. in a Pentium® II 350 MHz. Final path is generated by considering the curvature radii of the involved obstacles (see figure 8b).

5.1 Alternative Paths

As it is easily observed (see figures 6 or 8a), all $\hat{v}_{MTD}$ hold the same direction but different sign (representing translation to left or right). Moreover, when an obstacle is collided, O is inside the corresponding Minkowski difference, so MTD is computed as the minimum of the MTD^+ from O to a set of bi-spheres whose centers are c_s, c_g.

Both ideas let us obtain alternative paths keeping the same computational cost in the intersection-test process, because each one of the computed MTD^+ states an alternative way to avoid the involved obstacle. Figure 9 shows all the different sets of parameters λ, MTD and $\hat{v}_{MTD}$ determined when dealing with collided obstacles.

Considering the sets of intermediate configurations with the same sign in $\hat{v}_{MTD}$, two alternative new collision-free paths are generated (shown in figure 9). Intermediate configurations of both paths have been obtained in 137 μsec. on a Pentium® II 350 MHz.

Observing right path in figure 9, it is deduced that obstacle S_7 is not optimally avoided. This problem is easily solved by computing d_O "crossing the obstacle" and its MTD as

$$\mathrm{MTD}(O, S_{sg} - S_{ob}) = -\left(d_O + r'_0 + \sum_{j=1}^{J} \lambda_{Oj}(r'_j - r'_0) \right) \qquad (17)$$

When (17) is applied to S_7, a new collision-free path is generated avoiding properly S_7 (right side). More alternative paths are generated applying (17) to $MTD^{\geq 0}$ obstacles.

6 Conclusions

Firstly, in this paper has been presented a technique for computing the minimum translational distance between spherically extended polytopes. Applying this distance between a given not collision-free path and an obstacle, an intermediate configuration is obtained. This process is repeated recursively until obtaining a collision-free path in three-dimensional space.

Important and new advantages have been introduced when movement is constrained to two degrees of freedom. After having obtained all the intermediate configurations, Hough transform is applied to such configurations. Then, it is applied a technique called Minimum Volume Locus (MVL) for selecting the minimum number of intermediate configurations (obstacles) needed to determine a collision-free path passing through such selected configurations.

As the proposed distance technique allows to obtain more than one intermediate configurations (different ways for avoiding an obstacle), alternative collision-free paths are generated by selecting appropriated configurations using the properties of the Minimum Volume Locus again.

Additionally, as MTD computation is equal to GJK algorithm when radii of the spherical vertices are zero

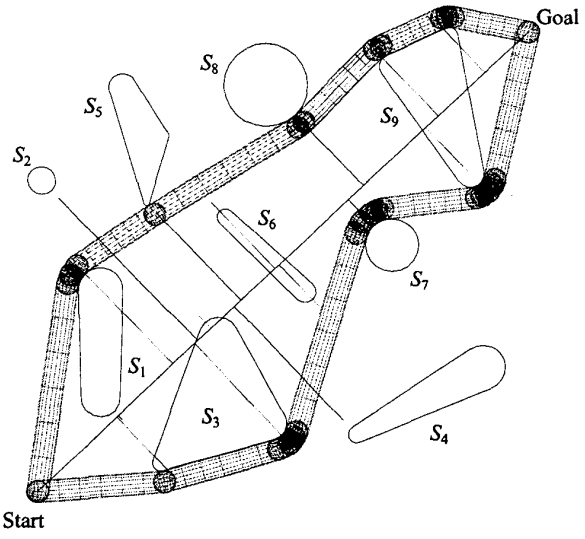

Figure 9. Alternative collision-free paths

(polytopal model), all path-planning methods presented in this paper are also applicable to move a point among such polytopal environment.

References

[1] E.J. Bernabeu, J. Tornero, "An Automatic Method for the Geometric Modeling of Robotic Systems Based on the Hough Transform". *11th Int. Conf. on CAD/CAM, Robotics & Factories of Future*,2,pp.758-763, Columbia, 1995.

[2] E.J. Bernabeu, J. Tornero, "Robotic Systems Modeling for Trajectory Planning". *IEEE-SMC Computational Engineering in Systems Applications*, pp. 670-675, Lille, France 1996.

[3] E.J. Bernabeu, J. Tornero, M. Mellado "A Generalized Method for Optimal Modeling in Robotics Applications". *2nd World Automation Congress. 6th Int. Symp. on Robotics & Manufacturing*, vol. 6, pp. 69-74, Montpellier, 1996.

[4] S. Cameron, R.K. Culley, "Determining the Minimum Translational Distance between Two Convex Polyhedra". *Int. Conf. on Robotics & Automation*. pp. 591-596, SF, 1986

[5] E.G Gilbert, C.P. Foo, "Computing the Distance between General Convex Objects in Three-Dimensional Space". *IEEE Trans on Robotics & Automation* 6, pp.53-61, 1990

[6] E.G. Gilbert, D.W. Johnson, S.S. Keerthi S. S., "A Fast Procedure for Computing the Distance between Complex Objects in Three-Dimensional Space". *IEEE Journal of Robotics & Automation*, 4, nº 2, pp. 193-203, 1988.

[7] G.J. Hamlin, R.B. Kelley, J. Tornero, "Efficient Distance Calculation using Spherically-Extended Polytope (S-tope) Model". *IEEE Int. Conf. on Robotics & Automation*, vol. 3, pp. 2502-2507, Nice, 1992.

[8] P.V.C. Hough, "Methods and Means for Recognition Complex Patterns". U.S. Patent 3,069,654, 1962

[9] Ong C.J. and Gilbert E.G., "Robot Path Planning with Penetration Growth Distance". *Journal of Robotic Systems* 15(2), pp. 57-74, 1998.