Proceedings of the 26th Conference
on Decision and Control
Los Angeles, CA • December 1987

ROBOT PATH PLANNING WITH DISTANCE-SAFETY CRITERION

Suk-Hwan Suh and Kang. G. Shin

Robot Systems Division
Center for Robotics and Integrated Manufacturing
The University of Michigan
Ann Arbor, MI 48109

Abstract

Euclidean distance is the most popular criterion for robot path planning. However, the shortest path (SP) is dangerous in some cases because such a path drives the robot too close to obstacles. When safety is the main concern, a center-line path (CLP) providing the maximum clearance from obstacles would be preferable over the SP, although the length of a CLP could be considerably longer than that of a SP. Since the SP and CLP are two extremes with respect to the distance and safety criteria, respectively, it would be useful in practice to strike a compromise between the two criteria.

The purpose of this paper is to develop a method for determining an optimal path with a weighted distance-safety criterion. The method is composed of three parts: (i) construction of a region map by dividing the workspace, (ii) inter-region optimization to determine the entry and departure points of the path in each region, and (iii) intra-region optimization for determining the (optimal) path segment within each region. The region map is generated by using an approximate Voronoi diagram, and the inter-(intra-) region optimization is achieved by using the variational dynamic programming. Although it is developed for 2D problems, our method can be easily extended to a class of 3D problems. Numerical examples are also presented to demonstrate the method.

1 Introduction

Robot path planning is concerned with the determination of collision-free paths for a robot by connecting the starting and destination points in the workspace cluttered with obstacles. An *optimal path* is the one that optimizes a desired performance criterion. The usual performance criterion adopted in the literature is Euclidean distance, i.e., the minimum-distance path or shortest path (SP) problem.

The SP problem has been studied extensively by many researchers from a variety of points of view. In 2D, the SP is determined by using the visibility graph or V-graph [8]. Use of the visibility graph is based on the property that the SP passes through some of the vertices of polygon obstacles. However, this is not always valid for 3D problems since the SP around polyhedral obstacles do not pass through just a sequence of vertices, but may pass through a sequence of edges as well. Thus, the SP in 3D is determined by finding the points of contact with those edges that the SP passes [1,9,10]. In both 2D and 3D, the SP problem has essentially been solved; the main remaining issue is to develop efficient algorithms for computing the SP.

Since the SP minimizes the Euclidean distance to travel, it certainly is an attractive path for many cases. The SP, however, requires the robot to be too close to the obstacles (actually, it touches the obstacles) and hence possesses high potential for collision with obstacles. Thus, it may not even be desirable for certain cases. One may, of course, argue that the obstacles can be artificially enlarged to "some" extent to provide a leeway between the robot and obstacles. However, it is usually very difficult to determine the degree of enlargement of the obstacles during path planning because of its dependence on the utilization of the workspace as well as the uncertainty in robot dynamics during path execution.

The *safety* of a robot path can be quantified by the distance or clearance between the path and obstacles. Naturally, the larger its clearance, the safer the path will be. In contrast with the SP problem, if the safety of a path is the only concern, one would choose a path which provides the maximum clearance from obstacles. This path, which we will henceforth term a *center-line path* (CLP), would travel along a Voronoi diagram. A simple example showing both the SP and CLP is given in Fig. 1. A CLP could be considerably longer than a SP, indicating that the CLP is not desirable at all if the robot's traversal distance is a major consideration in path planning. A larger clearance (thus safer) from the obstacles usually results in a longer path than a SP. The Voronoi diagram method [7,11,13,17] may be used for the *safest path planning*.

The safety of a robot path has not been fully considered in almost all known path planning approaches except for the one in [8]. The path safety was obtained in [8] by first growing each obstacle by a specified amount (margin of safety) and then applying the shortest path algorithm. There are two problems in this approach. First, as pointed out in [9], the only feasible path could have been eliminated by growing obstacles. Second, the collision risk is represented by a simple binary variable; 1 within the safety margin and 0 beyond the safety margin. This is not realistic because of the uncertainty in robot dynamics during path execution, as mentioned earlier. In fact, the safety (or collision-risk) of a path varies with its clearance from the obstacles.

Since both the SP and CLP are two interesting and useful extremes with respect to the distance and safety criteria, respectively, one may wish, for practical reasons, to strike a compromise between these two paths. In other words, an optimal path that minimizes a weighted distance-safety criterion needs to be derived. Clearly, neither of the visibility graph and the Voronoi diagram alone can determine the optimal path. A new solution method must be called for. The main purpose of this paper is to develop a solution approach to the robot path planning problem with a weighted distance-safety criterion. The approach consists of three major steps. The first step is concerned with the determination of a region map. The workspace is divided into a set of regions using an approximate Voronoi diagram. The second step, called *inter-region optimization*, is to determine the entry and departure points of the optimal path in each region. The third step is concerned with the determination of the optimal path segment within each region and is called *intra-region optimization*. Closed-form solutions are obtained for the third step and a dynamic programming approach is developed for the second step.

Throughout the paper, a robot is assumed to be a disk in the 2D case or a sphere in the 3D case, thus allowing the robot to become a point by enlarging obstacles as much as the radius of the disk or sphere.

2 Problem Statement

The problem of finding a minimum-cost path (MCP) for a robot is of practical importance to various robot-based automation systems. The MCP problem can be loosely stated as follows: Find a collision-free path to steer the robot to a destination from a starting position, while minimizing some form of cost such as distance and traversal time.

In a room filled with obstacles, the robot is to move from a starting point to a destination point, denoted by S and E, respectively. Let $x(\omega)$, $\omega_0 \le \omega \le \omega_f$, be a path parameterized[2] with the curvilinear distance, and $\mathcal{O} = \{p \mid p \in \mathcal{O}_i,\ i = 1, \dots, n_{\mathcal{O}}\}$ be the space occupied by $n_{\mathcal{O}}$ obstacles. The MCP problem is then to minimize the cost functional

[1] This work was supported in part by the NASA Johnson Space Center under the grant No. NCC-9-16, the NSF grant No. ECS-8409938, and the Industrial Affiliates Program, Robot Systems Division, Center for Robotics and Integrated Manufacturing (CRIM), The University of Michigan.

[2] as discussed in [12].

$$C = \int_{\omega_0}^{\omega_f} L(x(\omega), \omega, \mathcal{O}) \; d\omega, \tag{1}$$

subject to the following constraints:

$$x(\omega_0) = S, \;\; x(\omega_f) = E, \tag{2}$$

$$\{x(\omega)\} \cap \mathcal{O} = \phi, \;\; \forall\, \omega \in [\omega_0, \omega_f]. \tag{3}$$

Note that the MCP problem is characterized by the form of the cost integrand, $L(x(\omega), \omega, \mathcal{O})$. For example, the MCP problem becomes a SP problem if $L(x(\omega), \omega, \mathcal{O}) = 1$, while it becomes a safest path problem if $L(x(\omega), \omega, \mathcal{O}) = -\|x(\omega) - \mathcal{O}\| \equiv -\min_{p \in \mathcal{O}} \|x(\omega) - p\|$. In the latter case, $\|\cdot\|$ represents Euclidean distance and the CLP will become the optimal path.

When both distance and safety are used as a criterion for the MCP problem – the main problem to be dealt with in this paper – the cost integrand may be defined as follows:

$$L(x(\omega), \omega, \mathcal{O}) = W - (1 - W) \|x(\omega) - \mathcal{O}\| / (\omega_f - \omega_0). \tag{4}$$

Note that minimization of this cost integrand is equivalent to minimizing distance and maximizing the *average* clearance of the path with a weighting factor W, $0 \leq W \leq 1$. Averaging the clearance is necessary to avoid double counting the part of the cost contributed by the path length.

The evaluation of $\|x(\omega) - \mathcal{O}\|$, usually requires a significant amount of computation to an extent that finding of the optimal path x is intractable even by numerical methods. One way to side step such a problem is to use the CLP as a reference path for measuring the safety. For any region of a collision-free space, the CLP can always be considered as the safest path. Consequently, an alternative way to handle the safety of a path is to use the deviation from the CLP, namely *center-line deviation*, $\|x(\omega) - CLP\|$. Hence a safer path is meant to have less center-line deviation.

Since the CLP passes through regions of varying clearance, a unit center-line deviation at a region would have a different degree of safety from that at another region. For example, consider the two points P_1 and P_2 in Fig. 2, both of which deviate from the CLP by the same amount. Obviously, P_1 in Region 1 has higher collision-risk than P_2 in Region 2, since the clearance of Region 1 is smaller than that of Region 2. Thus, it is necessary to *scale* the center-line deviation on the basis of the *criticality* of a unit center-line deviation in each region when the center-line deviation is used as the safety metric for a path. The scaling constant $\beta(\omega)$ should be inversely proportional to the clearance between obstacles and the CLP.

Considering all the aspects mentioned above, the global MCP (GMCP) problem with a weighted distance-safety criterion can now be stated as: Find a path $\{x(\omega),\ CLP_{\omega_0} \leq \omega \leq CLP_{\omega_f}\}$ that minimizes the cost

$$C = \int_{CLP_{\omega_0}}^{CLP_{\omega_f}} [W + (1 - W)\, \gamma(\omega) \|x(\omega) - CLP\|]\, d\omega, \tag{5}$$

where $CLP_{\omega_0} \leq \omega \leq CLP_{\omega_f}$ is the parameter describing the curvilinear distance along the CLP and $\gamma(\omega) = \beta(\omega)/(CLP_{\omega_f} - CLP_{\omega_0})$, subject to constraints (2) and (3).

In a given workspace, there could be several approximate[3] CLPs connecting the starting and destination points. Let an *open-space* be a space in which at least one collision-free path between the starting and destination points exists. Then, there exists a CLP for every open-space. Thus, the GMCP problem is decomposed into a set of local MCP (LMCP) problems, one for each open-space. Let CLP^i be the CLP for open-space i. The LMCP problem for open-space i can then be stated as: Find a path $x^i(\omega)$, $CLP_{\omega_0} \leq \omega \leq CLP_{\omega_f}$, minimizing the cost

$$C^i = \int_{CLP^i_{\omega_0}}^{CLP^i_{\omega_f}} [W + (1 - W)\, \gamma(\omega) \|x^i(\omega) - CLP^i\|]\, d\omega, \tag{6}$$

subject to constraints (2) and (3).

Since the GMCP problem can be solved by solving the LMCP problem Q times, where Q is the number of open-spaces, it is sufficient to consider the LMCP problem only. In the following section, a detailed solution approach to the LMCP problem is presented.

3 The Solution Approach

Our solution approach to the LMCP problem in 2D is composed of three parts: a) construction of a region map, b) inter-region optimization for finding the entry and departure points of the path at each region, and c) intra-region optimization for determining the optimal path within each region. In what follows, each part of the solution will be discussed in detail. An extension of the solution to a class of 3D problems will also be discussed later in this section.

Our approach is based on a modified version of the dynamic programming (DP), which is briefly described below. Suppose that an open-space is divided into a discrete grid (Fig. 3). As the grid becomes finer, the choice of a curve to connect two adjacent points on the grid matters less as long as the curve is smooth. Thus, one can choose any smooth curve that eases the computation of the cost[4] of connecting two adjacent grid points. A curve connecting two adjacent grid points is defined as a *path primitive*. It is easy to see that if a straight line segment is used as the path primitive, then the cost of going from one point on the grid to the next can readily be computed. Once all incremental costs are computed, the minimum cost path can be constructed backward from the destination point to the starting point. However, such a DP approach will not be attractive, because the accuracy of the DP solution often requires the grid to be too fine to be computationally traçtable. Thus, one way to overcome the computational problem without sacrificing the accuracy of the solution is to use a path primitive that minimizes the cost functional in a relatively coarse grid. To reduce the computational amount, the grid size should be enlarged to a maximum extent that the use of the path primitive can be justified. In such a case, the selection of a path primitive itself will become an optimization problem.

In what follows, we shall develop a computationally efficient solution to the LMCP problem on the basis of the latter form of dynamic programming. The solution begins with the construction of a discrete grid, called a *region map*, which is followed by the determination of entry and departure points of the optimal path in each region. Finally, an optimal path primitive will be determined to connect the entry and departure points within each region.

3.1 Region Map

Consider an open-space bounded by the edge of polygon obstacles. A set of *barricading-line segments* or simply *barricades* can be placed such that all collision-free paths must pass through them. The barricades may be formed in such a way that a straight-line segment connecting the bisection points of the two successive barricades is a part of the CLP. By connecting a pair of vertices of two neighboring polygons not to cross each other, the barricades and CLP can be determined as shown in Fig. 4(a). If the CLP intersects obstacles, or one wishes to obtain a more accurate CLP, additional barricades can be introduced by connecting various pairs of vertices. Some of these barricades may produce infeasible or poor CLP's as shown in Fig. 4(b), while others may produce more accurate CLP's as shown in Fig. 4(c). The most accurate CLP can be found by an impractical, exhaustive checking of all possible pairs of vertices. However, such an exhaustive checking is unnecessary for our problem, since it needs only an approximate CLP.[5] An example region map will be given later in Fig. 13. It is important to note that a region is always either a triangle or a quadrilateral.

Since all collision-free paths pass through the barricades, the LMCP problem now becomes that of finding the optimal points on each of the barricading lines and an optimal inter-barricade path primitive. Thus, the problem of finding the LMCP is divided into two parts: (i) inter-region optimization for finding two optimal points on two adjacent barricades, and (ii) intra-region optimization to determine an optimal path primitive connecting the two points.

3.2 Inter-Region Optimization

Suppose an open-space is characterized by N center-line segments and $N - 1$ barricades (Fig. 5). To apply the dynamic programming to the

[3] since the starting and destination points may not be on the center-line paths.

[4] This is an incremental cost in the dynamic programming.

[5] using an approximate Voronoi diagram that is similar to the one used in [4,6,13]. An exact Voronoi diagram is made up of straight-line and curve segments, and can be obtained by the *generalized* Voronoi diagram [7].

inter-region optimization, the barricading lines must be divided into a finite number of equally spaced points or *gates*. Let $G_i(j)$ and $M > 1$ be respectively the j^{th} gate and the number of gates on barricade i. Then the location of $G_i(j)$ is represented as:

$$G_i(j) = G_i(1) + (j-1)\eta_i\, \vec{\imath}, \tag{7}$$

where $\vec{\imath}$ is a unit vector from the one end, $G_i(1)$, to the other end, $G_i(M)$, of barricade i, and $\eta_i = \mathcal{D}_i/(M-1)$ and $\mathcal{D}_i$ is the length of barricade i. For notational convenience, the starting and destination points are treated as barricades with $M = 1$. These are not real but *pseudo* barricades. Thus, there are a total of $N+1$ barricades, where $G_1(1)$ and $G_{N+1}(1)$ represent the starting and destination points.

Let $I = \{g_1, \ldots, g_{N+1}\}$ be the set of gates that a path passes through, where g_i is the gate number of barricade i. Then, the inter-region optimization is to find a gate set such that the corresponding path has the minimum cost. The cost for the gate set I is given by

$$C(I) = \sum_{i=1}^{N} C_i(G_i(j), G_{i+1}(k)), \tag{8}$$

where $j, k \in I$, and $C_i(G_i(j), G_{i+1}(k))$ is the cost of the optimal path primitive, x_i^*, connecting the gates $G_i(j)$ and $G_{i+1}(k)$. With the cost functional (6),

$$C_i(G_i(j), G_{i+1}(k)) = \int_{G_i(j)}^{G_{i+1}(k)} [\, W + (1-W)\,\gamma(\omega)\, \|x_i^* - C_i\|\,]\ d\omega \tag{9}$$

where C_i is the i^{th} center-line segment. Finding the optimal path primitive, x^*, is the intra-region optimization which will be discussed in the next subsection.

The inter-region optimization procedure starts at barricade N. (Note that barricade $N+1$ is the destination point.) At barricade N, the cost of gate j, denoted by $C_N(j)$, can be computed once $x_N^*(j)$ is determined using $G_N(j)$ and $G_{N+1}(1)$ as the starting and destination points, respectively, i.e., $C_N(j) = C_N(G_N(j), G_{N+1}(1))$, and the pointer[6] at gate j on barricade N, $d_N(j) = 1$, $\forall\, j = 1, \ldots, M$.

The cost at gate j on barricade $N-1$ becomes

$$C_{N-1}(j) = \min_k [\, C_{N-1}(G_{N-1}(j), G_N(k)) + C_N(k)\,], \quad k = 1, \ldots, M, \tag{10}$$

and the pointer $d_{N-1}(j)$ will be the k that minimizes the magnitude of $[\cdot]$ in Eq. (10). Recursive equations for barricade i can be obtained by replacing N and $N-1$ in the above with $i+1$ and i, respectively.

The DP algorithm for the inter-region optimization is summarized as follows:

```
• Step 1: Initialization
    1. Read in the center-line and barricade data.
    2. Form gates: $G_i(j),\ i = 1, \ldots, N+1;\ \ j = 1, \ldots, M$.
    3. Set $I := N$.
    4. For $j = 1, \ldots, M$,[7] set $d_I(j) := 1$, and compute $C_I^*(j) = C_I^*(G_I(j), E)$.
• Step 2: Termination Check
  Set $I := I - 1$.
  If $I = 0$, stop. Otherwise go to Step 3.
• Step 3: Continuation
  For $j = 1, \ldots, M$,
    1. Set $C_I^*(j) := \infty$.
    2. For $k = 1, \ldots, M$,
       Compute $X = C_I^*(G_I(j), G_{I+1}(k)) + C_{I+1}^*(j)$
       If $X < C_I^*(j)$, then $C_I^*(j) := X$ and $d_I(j) := k$.
  Go to Step 2.
```

3.3 Intra-Region Optimization

Intra-region optimization is to find an optimal path primitive connecting two adjacent gates, each located on a barricading line. A region is defined as the *convex hull* of the two barricades. Then, the region is either a triangle or a quadrilateral with two edges of barricading lines. Let L and U denote the non-barricading edges of the region. One of L and U will be a point if the region is a triangle; otherwise L and U represent the upper and lower (with respect to the center-line segment of the region) obstacle-boundary lines.

Consider a plane using a rectangular coordinate system with origin at one end of the center-line segment and the horizontal axis coinciding with the center-line segment that connects the bisection points of the two adjacent barricades. Let the two gates $G_i(j)$ and $G_{i+1}(k)$ in this coordinate system be represented as (ω_0, x_0) and (ω_f, x_f), respectively, and equations of the boundary lines and the center-line segment be

$$L(\omega) = a_L\omega + b_L, \quad \omega_{L_L} \le \omega \le \omega_{L_U}, \tag{11}$$

$$U(\omega) = a_U\omega + b_U, \quad \omega_{U_L} \le \omega \le \omega_{U_U}, \tag{12}$$

$$C(\omega) = 0, \quad 0 \le \omega \le \omega_{C_U}, \tag{13}$$

where a_i, b_i for $i = L, U$ are coefficients of boundary line equations, $\omega_{L_U} - \omega_{L_L}$ and $\omega_{U_U} - \omega_{U_L}$ are the projected curvilinear lengths of L and U on the horizontal axis, respectively, and ω_{C_U} in Eq. (13) is the length of the center-line segment C of this region.

Define a path x as a function of ω such that the coordinate of a point $x(\omega)$ is represented by $(\omega, x(\omega))$. Consider the following case which corresponds to *Case A* in Fig. 8(a)

$$\omega_0 \ge 0, \quad \omega_f \le \omega_{C_U}. \tag{14}$$

Since the horizontal axis of the coordinate system coincides with the center-line segment, the center-line deviation of the point $x(\omega)$ becomes

$$\|\, x(\omega) - C \,\| = |\, x(\omega)\,|, \quad \forall\, \omega \in [\omega_0, \omega_f]. \tag{15}$$

The length of a path $\{x(\omega),\ \omega_0 \le \omega \le \omega_f\}$ can be computed as $\int_{\omega_0}^{\omega_f} \sqrt{1 + \dot{x}(\omega)^2}\ d\omega$, where $\dot{x} = dx/d\omega$. The scaling constant for the region is defined as a reciprocal of the clearance, which may be obtained by computing the distance between the midpoint of C and the upper and lower boundary lines:[8]

$$\overline{\beta} = 1/[\ \|(\tfrac{1}{2}\,\omega_{C_U}, 0) - U\| + \|(\tfrac{1}{2}\,\omega_{C_U}, 0) - L\|\ ]. \tag{16}$$

To find the optimal path primitive $\{x^*(\omega),\ \omega_0 \le \omega \le \omega_f\}$, it is necessary to solve the following problem called the *Path Primitive Problem.*

Path Primitive Problem : Find x^* minimizing the following functional

$$J = \int_{\omega_0}^{\omega_f} [\, W(1 + \dot{x}^2(\omega)) + (1-W)\,\gamma\, x^2(\omega)\,]\ d\omega, \tag{17}$$

where $\gamma = \overline{\beta}/(\omega_f - \omega_0)$, subject to constraints:

$$x(\omega_0) = x_0, \quad x(\omega_f) = x_f, \tag{18}$$

$$a_L\omega + b_L \le x(\omega) \le a_U\omega + b_U, \quad \forall\, \omega \in [\omega_0, \omega_f]. \tag{19}$$

It is worth noting that the quadratic form in (17) is used for mathematical tractability (e.g., the LQ problem [5]), and can be employed to the problem of finding the path primitive, x^*.

The Weierstrass-Edermann corner conditions [5] assure that there is no corner in the optimal path of this problem as long as $W \ne 0$ (see Appendix A for a detailed description). The solution to the Path Primitive Problem can be obtained by solving the Euler equation, and the necessary condition for the optimality is given as:

$$\begin{aligned} 0 &= \frac{\partial g}{\partial x}(x^*(\omega), \dot{x}^*(\omega), \omega) - \frac{d}{d\omega}\left[\frac{\partial g}{\partial \dot{x}}(x^*(\omega), \dot{x}^*(\omega), \omega)\right] \\ &= 2(1-W)\,\gamma\, x^*(\omega) - \frac{d}{d\omega}[2W\dot{x}^*(\omega)], \end{aligned}$$

[6] for the next gate.

[7] If $N = 1$, let $M = 1$.

[8] Notice that we are using the average clearance for a region and that $\beta(\omega) = \overline{\beta}$, $\forall\, \omega \in [\omega_0, \omega_f]$.

or

$$W\ddot{x}^*(\omega) - (1 - W)\gamma\, x^*(\omega) = 0. \tag{20}$$

Since Eq. (20) is linear in x^* with constant coefficients, it can be readily solved. The optimal path within a region, $x^*(\omega)$, is obtained as follows.

Case 1 : $W = 0$: The Euler equation (20) becomes $x^*(\omega) = 0$, indicating that the optimal path is the same as the center-line segment. With the boundary conditions (18), the optimal path becomes:

$$x^*(\omega) = \begin{cases} \omega_0 & \text{if } \omega = \omega_0 \\ 0 & \text{if } \omega_0 < \omega < \omega_f \\ \omega_f & \text{if } \omega = \omega_f \end{cases} \tag{21}$$

Notice that the optimal path in this case is straight-line segments connecting *four points* in sequence; $(\omega_0, x_0), (\omega_0, 0), (\omega_f, 0)$, and (ω_f, x_f). Therefore, there are two corner points if $x_0 \neq x_f$.

Case 2 : $W = 1$: The Euler equation (20) becomes $\ddot{x}^*(\omega) = 0$, i.e.,

$$x^*(\omega) = c_1\omega + c_2, \quad \omega_0 \leq \omega \leq \omega_f, \tag{22}$$

where c_1, c_2 are determined by the two boundary conditions (18). Note that the optimal path in this case is a straight line segment connecting the entry and departure gates.

Case 3 : $0 < W < 1$:

$$x^*(\omega) = c_3 e^{\sqrt{k}\omega} + c_4 e^{-\sqrt{k}\omega}, \quad \omega_0 \leq \omega \leq \omega_f, \tag{23}$$

where c_3 and c_4 are the integration constants determined by the two boundary conditions (18), and $k = (1 - W)\gamma / W$.

Optimal paths with various values of W for two sets of entry and departure gates are plotted in Figs. 6(a)-(b). Fig. 8 shows that as the value of W increases (decreases), the path tends to be closer to a straight line (center line). Also, all the paths traverse inside the convex hull defined by the four points, $(\omega_0, x_0), (\omega_0, 0), (\omega_f, 0), (\omega_f, x_f)$.

It is worth mentioning that the necessary condition (20) is derived as if we were dealing with an unconstrained optimization problem, i.e., ignoring the constraint (19). This is valid under a certain condition as stated below.

Theorem 1: The constraint (19) is redundant, $\forall W \in [0, 1]$, if the entry and departure gates, $G_i(j), G_{i+1}(j)$, are within the collision-free space.

Proof: Consider the upper-boundary line U. Suppose $x^*(\omega)$ sways above the line U. Then there exist two distinct points, one to break-out, $B_1 = (\omega_1, x_1)$, and the other to break-in, $B_2 = (\omega_2, x_2)$, to the line, since the path must be continuous and $x_0, x_f \leq U(\omega)$, $\forall\, \omega \in [\omega_0, \omega_f]$ (see Fig. 7). For the interval $[\omega_1, \omega_2]$, clearly the straight line path connecting B_1 and B_2 is superior to the x^* in the sense of both distance and center-line deviation. Thus, an optimal path cannot traverse over the upper boundary line. The same argument holds for the lower boundary line L. Hence, constraint (19) is met automatically. □

Corollary 1: If a collision-free space is a convex set F, the obstacle-avoidance constraint is redundant $\forall\ W \in [0, 1]$ iff $G_i(j),\ G_{i+1}(j) \in F$.

Proof: The optimal path traverses inside the convex hull defined by the four points $\forall W \in [0, 1]$. Since the center-line segment traverses inside the collision-free sapce, the four points, (ω_0, x_0), $(\omega_0, 0)$, $(\omega_f, 0)$, (ω_f, x_f), iff $G_i(j),\ G_{i+1}(j) \in F$. Thus, the convex hull $\in F$, and $\{x^*(\omega) |\ \omega_0 \leq \omega \leq \omega_f\} \in F$. □

The obstacle avoidance constraints can be ignored when x^* is determined by the intra-region optimization since every region is either a triangle or a quadrilateral, i.e., a convex set.

The use of the x^* given in Eqs. (21)-(23) can be generalized by the following theorem.

Theorem 2 : Even if the condition (14) is not met, the x^* given in Eqs. (21)-(23) is still an optimal path.

Proof : Let *Case A* : $\omega_0 \geq 0,\ \omega_f \leq \omega_{C_U}$, *Case B* : $\omega_0 \geq 0,\ \omega_f \geq \omega_{C_U}$, *Case C* : $\omega_0 \leq 0,\ \omega_f \leq \omega_{C_U}$, and *Case D* : $\omega_0 \leq 0,\ \omega_f \geq \omega_{C_U}$.

Consider *Case B* (see Fig. 8(b)). To derive the solution for *Case B*, the functional (17) is represented by:

$$J = \int_{\omega_0}^{\omega_f} [\, W(1 + \dot{x}^2(\omega)) + (1 - W)\gamma\, \|(\omega, x(\omega)) - C\|^2\,]\, d\omega. \tag{24}$$

Note that the center-line deviation in (24) is represented in terms of the distance between the point coordinate of $(\omega, x(\omega))$ and the center-line segment C. Also, the representation inside the norm is the same as $x^2(\omega)$ in (17) when $\omega_0 \geq 0,\ \omega_f \leq \omega_{C_U}$. The center-line deviation is

$$\begin{aligned} \int_{\omega_0}^{\omega_f} \|(\omega, x(\omega)) - C\|^2 &= \int_{\omega_0}^{\omega_{C_U}} \|(\omega, x(\omega)) - C\|^2\, d\omega \\ &+ \int_{\omega_{C_U}}^{\omega_f} \|(\omega, x(\omega)) - C\|^2\, d\omega \\ &= \int_{\omega_0}^{\omega_{C_U}} x^2(\omega)\, d\omega + \int_{\omega_{C_U}}^{\omega_f} [x^2(\omega) + (\omega - \omega_{C_U})^2]\, d\omega \\ &= \int_{\omega_0}^{\omega_f} x^2(\omega)\, d\omega + \int_{\omega_{C_U}}^{\omega_f} (\omega - \omega_{C_U})^2\, d\omega. \end{aligned} \tag{25}$$

Since for given ω_{C_U} and ω_f the second term in (25) is a constant, indicating that the $x(\omega)^*$ given in Eqs. (21)-(24) is also the solution for *Case A*. Similiarily, the same conclusion can be drawn for Cases C and D. □

As a result of Theorem 2, $\{\, x^*(\omega),\ \omega_0 \leq \omega \leq \omega_f \,\}$ given in Eqs. (21)-(24) can be used as an optimal path primitive for any pair of gates.

Since the path equation is given in a local coordinate frame whose origin is at one end of the center-line segment, a coordinate transform may be necessary to convert the path in the local coordinate frame into the one in the world coordinate frame. For this purpose, the following transform equation can be used.

$$\begin{aligned} \begin{bmatrix} {}^W\omega \\ {}^Wx \end{bmatrix} &= \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \omega \\ x \end{bmatrix} + \begin{bmatrix} \omega^S \\ x^S \end{bmatrix} \\ &= R(\theta) \begin{bmatrix} \omega \\ x \end{bmatrix} + \begin{bmatrix} \omega^S \\ x^S \end{bmatrix}, \end{aligned} \tag{26}$$

where $R(\theta)$ is the rotation matrix corresponding to the local coordinate frame relative to the world coordinate system, and θ is the rotational angle of the local coordinate frame defined as the counter-clockwise angle between C and the horizontal axis of the world coordinate frame, and $(\omega^S,\ x^S)$ represents the translation of C from the origin of the world frame.

3.4 3D Extension

We have presented above a method to determine an optimal path in 2D. In 2D, the safest path is represented as a straight line, i.e., center-line path, and the collision-free space is divided into regions by the barricades and obstacle boundaries. These barricades are placed in such a way that all the collision-free paths must pass through all of the barricades. In case of 3D, however, the CLP is not a line but a plane in general.

Consider a room filled with polyhedral objects. Suppose an *articulated cylinder* is placed as shown in Fig. 9. Then we may consider the axial symmetry of the cylinder as a CLP and its cross-sectional diameter as the clearance. The barricade may be the cross-sectional circle where the CLP is bent. The articulated cylinder may be thought of as an open-space. Note, however, that the approximation of the collision-free space by the articulated cylinder wastes the collision-free space.

With the above analogy, our DP algorithm can be directly applied to 3D problems. The differences are: a) the gates are generated by discretizing the barricading plane (see Fig. 10), and b) the intra-region optimization to determine an inter-gate path primitive is performed with the functional

$$J = \int_{\omega_0}^{\omega_f} [\, W(1 + \dot{x}(\omega)^2 + \dot{y}(\omega)^2) + (1 - W)\gamma\,(x(\omega)^2 + y(\omega)^2)\,]\, d\omega, \tag{27}$$

where $\gamma = \overline{\beta}/(\omega_f - \omega_0)$, and $\overline{\beta}$ is the reciprocal of the average diameter of each cylinder. Details of the solution to this problem are omitted since it is very similar to the 2D case.

4 Numerical Examples

Several path planning examples are presented to demonstrate the use of the path planning algorithm discussed thus far. The algorithm was coded in FORTRAN and run on a VAX-11/780[9] under the UNIX[10]4.2 operating system.

The first example in Fig. 11 is to show the behavior of optimal paths with respect to an obstacle while varying the weighting factor, W. The

[9] VAX is a trademark of the Digital Equipment Corporation.

[10] UNIX is a trademark of AT&T Bell Laboratories.

example is simulated with two barricading lines and 20 gates along each barricade. Fig. 11 illustrates that the optimal path tends to traverse away from (close to) the obstacles as the value of W decreases (increases). The paths with W close to either 0 or 1 have visible corners, but they are relatively smooth with mid-range values of W. Note that there do not exist ridges around the gates where the two path primitives are joined regardless of W value, since such a path cannot be optimal.

The second example in Fig. 12 is to show that the open-space where the GMCP is found can be changed with the value of W. This example is simulated with 15 gates on each barricade. As shown in Fig. 13, there are 4 open-spaces. Figs. 14(a)-(c) show the plots of LMCP's for three different values of W. As illustrated in these figures, when the safety factor is large, e.g., $W = 0.25$, the GMCP tends to traverse the most spacious open-space (open-space 1), while the GMCP traverses in any region to minimize the traveling distance as the weighting factor gets large, e.g., $W = 0.9$.

As the safety factor gets large, the LMCP tends not to deviate from the safest path (i.e., CLP) to minimize the safety cost, especially when the LMCP traverses through a narrow-channel region (see the LMCP II and III in Fig. 14(a)). As a result, the LMCP bends itself several times and yields visual corners. As mentioned before, these visual corners disappear in the mid-range values of the weighing factor (see Fig. 14(b)). On the other hand, when the distance factor gets large, the LMCP is made up of straight-line segments to minimize the distance cost (See Fig. 14(c)). Note that a bend around the beginning part of the LMCP II is due to the discretization of the barricades.

5 Concluding Remarks

In this paper we presented a new method for optimal robot path planning with a weighted distance-safety criterion. Path planning with this criterion has practical importance but calls for a new solution approach. Neither of the visibility graph and Voronoi diagram alone can determine an optimal path. Our solution method is composed of three parts: a) construction of a region map, b) inter-region optimization, and c) intra-region optimization. It was also shown that the method can be readily extended to a class of 3D problems.

The key idea of the method is to coarsen the grid resolution of the dynamic programming by decomposing the optimization procedure into two parts. The decomposition turns out to reduce the computational requirements significantly. Furthermore, since a closed form solution for the intra-region optimization is derived, the computational amount is quite low. The total CPU time per open-space for the examples shown in Section 4 is approximately 10 seconds for the first example and 20 seconds for the second example. Note that the CPU time is in general a non-decreasing function of the number of gates on each barricade, the number of barricades in each open-space, and the number of open-spaces in the workspace. Since all these factors are related to the region map used, developing a structured method for constructing an efficient region map is important. An efficient region map should minimize the number of barricades while satisfying the required accuracy of the center-line path. This is a matter of our future inquiry.

References

[1] C. Bajaj, "An Efficient Parallel Solution for Euclidean Shortest Path in Three Dimensions," *Proc. IEEE Int'l. Conf. Robotics Automat.*, pp. 1897-1900, San Francisco, April 1986.

[2] C. O. Dulaing and C. K. Yap, "The Voronoi Method for Motion Planning: I. The Case of a Disk," *J. Algorithm*, vol. 6, pp. 104-111, 1985.

[3] E. G. Gilbert and D. W. Johnson, "Distance Functions and Their Application to Robot Path Planning in the Presence of Obstacles," *IEEE J. Robot. Automat.*, vol. RA-1, no. 1, pp. 21-30, March 1985.

[4] K. Kant and S. W. Zucker, "Toward Efficient Trajectory Planning: The Path-Velocity Decomposition," *Int'l. J. Robot. Res.*, vol. 5, no. 3, pp. 72-89, Fall 1986.

[5] D. E. Kirk, *Optimal Control Theory: An Introduction*, Englewood Cliffs: Prentice Hall, Inc., 1970.

[6] D. G. Kirkpatrick, "Efficient Computation of Continuous Skeletons," *Proc. 20th IEEE Symp. Foundations Comp. Science*, San Juan, pp. 18-27, 1979.

[7] D. T. Lee and R. L. Drysdale, "Generalization of Voronoi Diagrams in the Plane," *SIAM J. Comput.*, vol. 10, no. 1, pp. 73-87, February 1981.

[8] T. Lozano-Perez and M. A. Wesley, "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles," *Comm. ACM*, vol. 22, no. 10, pp. 560-570, October 1979.

[9] J. Mitchell and C. Papadimitriou, "Planning Shortest Paths," Technical Report, Dept. Operations Research, Stanford University.

[10] C. H. Papadimitriou, "An Algorithm for Shortest-Path Motion in Three Dimensions," Technical Report, Dept. Operations Research, Stanford University.

[11] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, New York: Springer-Verlag, 1985.

[12] K. G. Shin and N. D. McKay, "Selection of Near-Minimum Time Geometric Paths for Robotic Manipulators," *IEEE Trans. Automat. Contr.*, vol. AC-31, no. 6, pp. 501-511, June 1986.

[13] K. G. Shin and R. D. Throne, "Robot Path Planning Using Geodesics and Straight Line Segments with Voronoi Diagram," CRIM Technical Report RSD-TR-27-86, Center for Robotics and Integrated Manufacturing, The University of Michigan, December 1986.

[14] S. H. Suh, "Development of an Algorithm for a Minimum-Time Trajectory Planning Problem Under Practical Considerations," Ph.D Dissertation, The Ohio State University, 1986.

[15] S. H. Suh and A. B. Bishop, "Tube Concept and Its Application to the Obstacle-Avoiding Minimum-Time Trajectory Planning Problem," publication pending.

[16] S. H. Suh and K. G. Shin, "Tube Parameters for the Planning of the Optimal Robot Trajectories," in preparation.

[17] C. K. Yap, "An O(n log n) Algorithm for the Voronoi Diagram of a Set of Simple Curve Segments (Preliminary Version)," Technical Report 161, Dept. Computer Science, New York University, May 1985.

A Appendix A

Applying the Weierstrass-Erdmann corner conditions [5] to the functional (17) of the *Path Primitive Problem*:

$$\frac{\partial g}{\partial \dot{x}} \; (x^*(\omega_1, \dot{x}^*(\omega_1^-), \omega_1) = \frac{\partial g}{\partial \dot{x}} \; (x^*(\omega_1, \dot{x}^*(\omega_1^+), \omega_1), \tag{28}$$

and

$$g(x^*(\omega_1, \dot{x}^*(\omega_1^-), \omega_1) - \left[\frac{\partial g}{\partial \dot{x}} \; (x^*(\omega_1), \dot{x}^*(\omega_1^-), \omega_1) \right] \dot{x}^*(\omega_1^-)$$
$$= g(x^*(\omega_1, \dot{x}^*(\omega_1^+), \omega_1) - \left[\frac{\partial g}{\partial \dot{x}} \; (x^*(\omega_1), \dot{x}^*(\omega_1^+), \omega_1) \right] \dot{x}^*(\omega_1^+), \tag{29}$$

where $g(\cdot)$ is the integrand in Eq. (17). From Eqs. (28)-(29), we get

$$2W\dot{x}^*(\omega_1^-) = 2W\dot{x}^*(\omega_1^+), \tag{30}$$

$$\begin{aligned} &W[1 + \dot{x}^{*2}(\omega_1^-)] + (1 - W)\,\gamma\, x^{*2}(\omega_1) - [2W\dot{x}^*(\omega_1^-)]\dot{x}^*(\omega_1^-) \\ &= W[1 + \dot{x}^{*2}(\omega_1^+)] + (1 - W)\gamma x^{*2}(\omega_1) - [2W\dot{x}^*(\omega_1^+)]\dot{x}^*(\omega_1^+). \end{aligned} \tag{31}$$

From Eq. (30), if $W \neq 0$, $\dot{x}^*(\omega_1^-) = \dot{x}^*(\omega_1^+)$, i.e., no corners exist. However, if $W = 0$, there may be some corners: as shown in Eq. (22), there exist two corners unless $x_0 = x_f$.

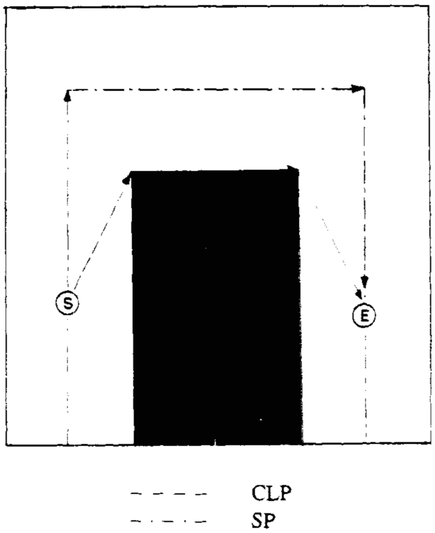


Fig. 1: CLP and SP

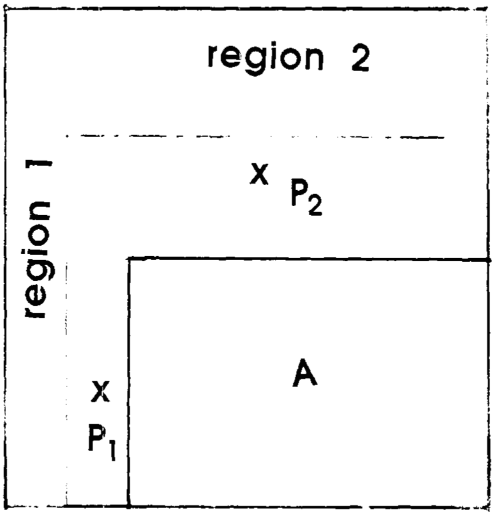


Fig. 2: Two points

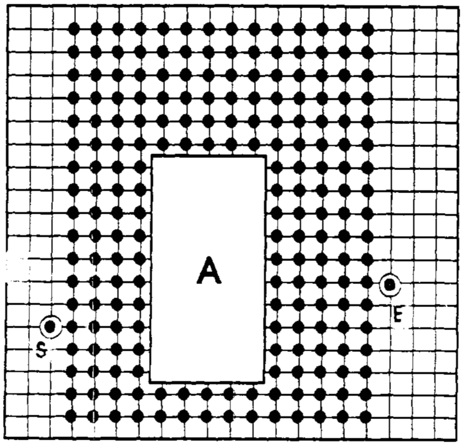


Fig. 3: Grid for a simple dynamic programming

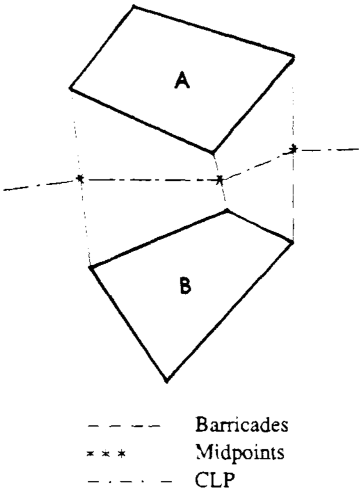


Fig. 4(a): CLP and barricades

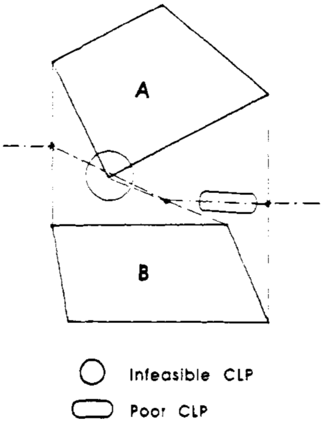


Fig. 4(b): Infeasible and poor CLP

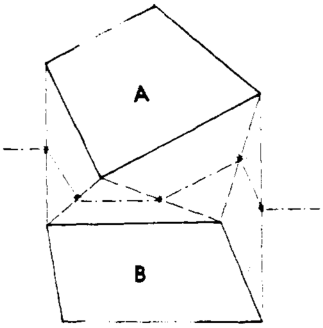


Fig. 4(c): Modified CLP

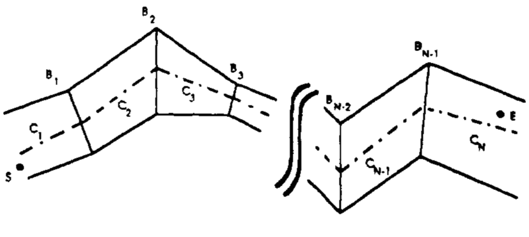


Fig. 5: Collision-free space with N center-line segments

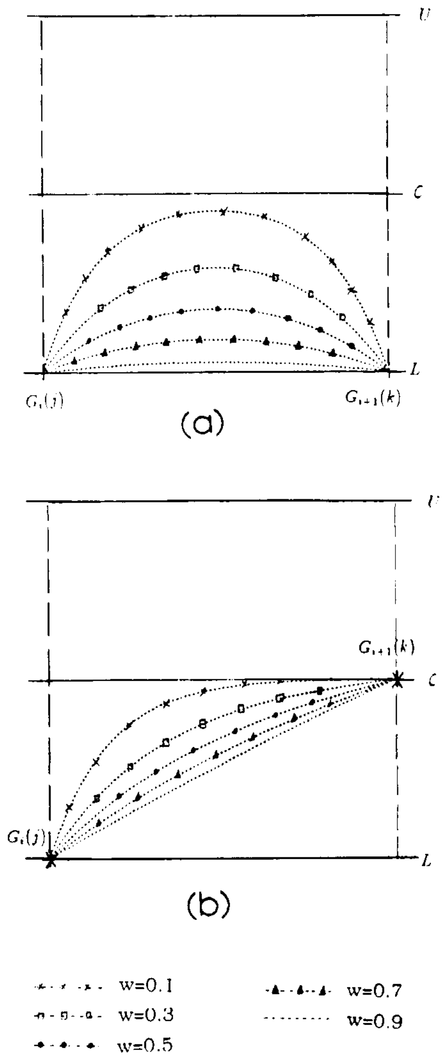


Fig. 6: Optimal paths

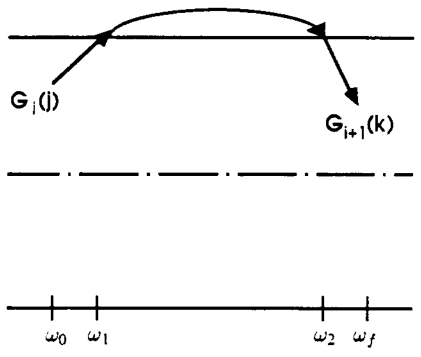


Fig. 7: Break-in and break-out points

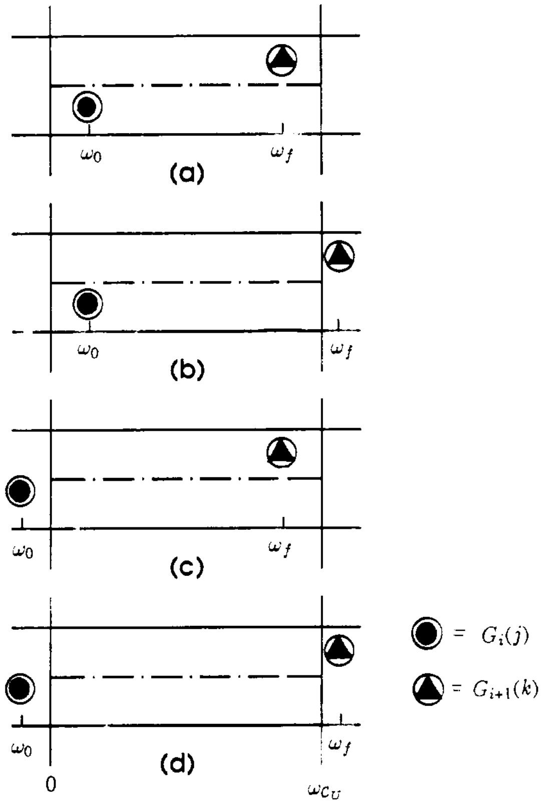


Fig. 8: Four possible cases

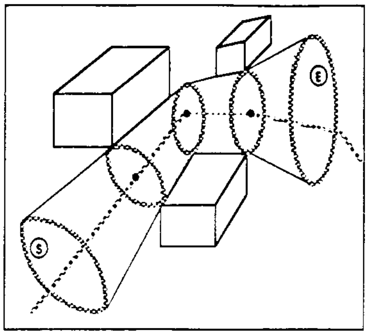


Fig. 9: A blended articulated cylinder

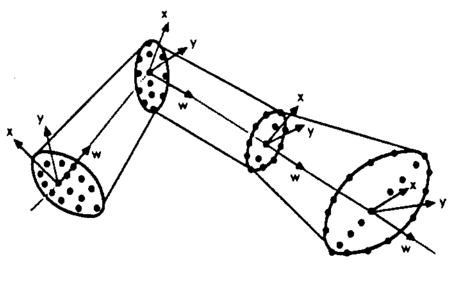


Fig. 10: Barricading circles and axis nomenclature

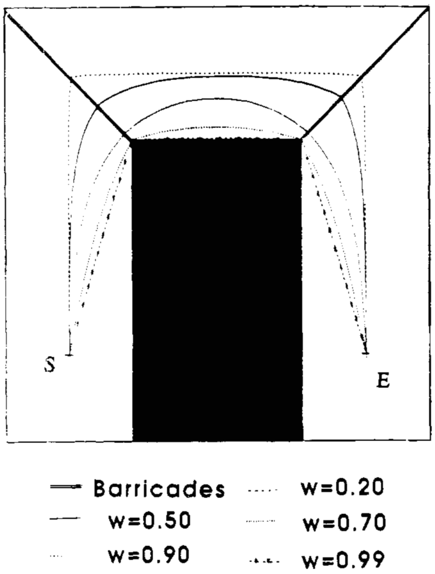


Fig. 11: Example 1

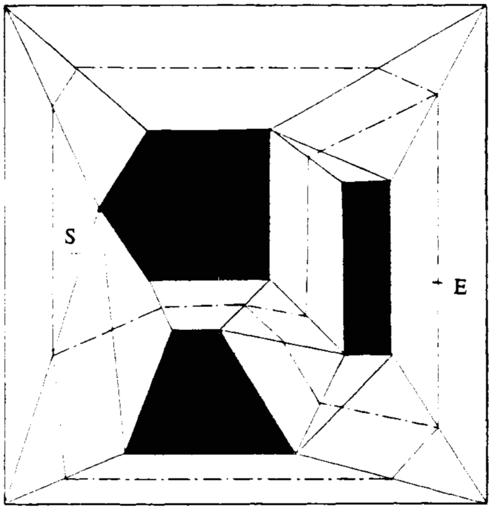


Fig. 12: Region map for Example 1

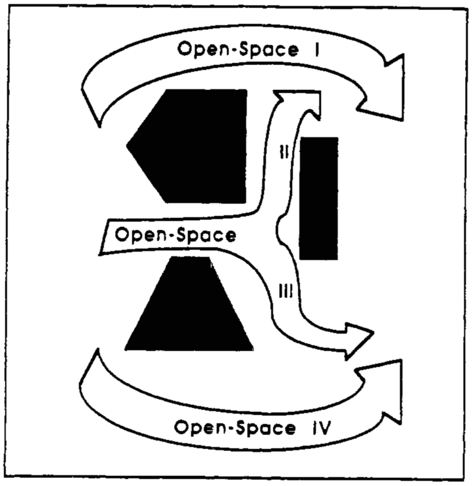
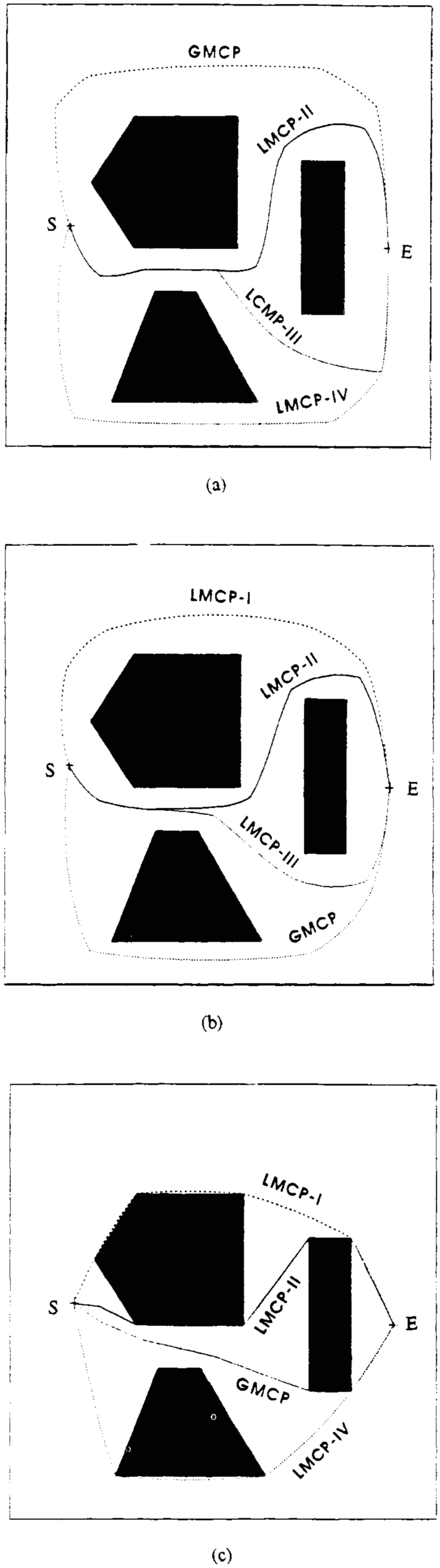


Fig. 13: Open-spaces for Example 2



(a)

(b)

(c)

Fig. 14: GMCP and LMCPs (a)W = 0.25 (b)W = 0.50 (c) W = 0.90