Mobile Robot Path Planning in 2D Using Network of Equidistant Path[1]

Mladen Crnekovic, Branko Novakovic and Dubravko Majetic

Faculty of Mechanical Engineering and Naval Architecture, Automation Department, Zagreb, Croatia

This article proposes mobile robot path planning in the presence of static obstacles by arbitrary shapes. The realized path is optimal in a global sense (shortest). The path searching process is divided in three steps: network of equidistant path design, optimal path selection and simulation along the optimal path. The obtained freeways model is small, therefore the solution time is short. During the motion, the robot is absolutely safe from obstacles and never makes sudden rotations. The presented algorithm has been successfully tested on many examples in different situations and difficulties. It is able to solve even maze-type problems. Solution times are comparable with human reactions faced with the same problem. The main contribution of this paper is increasing the efficiently of equidistant path searching and introduction the widest path algorithm.

Keywords: path planning, mobile robot, equidistant path, widest pass

1. Introduction

Successful mobile robot path planning is the basic prerequisite for its autonomous work. After solving the path problem, direct task programming is possible instead of its motion programming. There are several ideas for path planning solution. Which of them will be used depends on environment information, demands on robot motion and control unit characteristics. It is necessary to make a compromise between a planned path accuracy, a free-space model size and the time for its solving.

In a mobile robot movement, the task is to arrive from the source position to the goal position and orientation, requiring that no collision occurs with any obstacle along the path. On the realized motion an additional requirement could be set: the path from the start to goal must be the shortest path, the robot must not approach any obstacle less than the setting value, etc. The solution of this problem is actual and stands in open research field.

1.1. Problem Classification

Path planning classification is possible in accordance with several aspects. From the aspect of feedback and information about environment, the path planning could be:

A1) Local path planning supposes that only information about local environment is known (from the point where the robot is temporary).

A2) Global path planning supposes that the information of global environment state is known, or that it is possible to reconstruct it from the received information. In that case it is possible to design optimal path in a global sense.

According to the robot dimensions and free ways among obstacles, there are two situations:

B1) The robot dimensions are much smaller than freeways dimensions. In this case path planning is simplified because robot orientation could be omitted. The planned path usually consists of lines.

B2) The robot dimensions are of the same order as freeways dimensions. Therefore, path planning is more complicated because robot dimensions must be taken into account.

[1] This research is supported by the Croatian Ministry of Science and Technology, No. 120008/1996.

From the aspect of an obstacle in motion, there are two situations:

C1) Obstacles are static, which means that obstacles do not change their positions, orientations and shapes during the robot movement from a start to a goal.

C2) Some obstacles are moving and some are static. If obstacle movement is not predetermined, then optimal path in global sense is not possible.

If some obstacles are moving, there are two general approaches of obstacle avoidance:

D1) Change of moving speed on nominal path (similar to car movement along a road). The method is simple and efficient in wide range of situations.

D2) Aberration from nominal path (speed change is also possible).

1.2. Solving Methods

All methods for path planning have the same idea: from complete physical space with obstacles, it is necessary to *extract a free space and build its mathematical model.* A free space is a space that is not occupied by obstacles and could be used for robot movement. In most cases, a free space model is described by a graph theory. Optimal path searching is then transformed into the search of an extreme of a cost function that is selected for optimizing.

Path searching methods differ one from the other in the way how the free space is extracted. It is possible to separate several approaches to problem solving:

I) *Configuration space method (C-space).* Path planning in a physical space, for the robot with N degrees of freedom, is transformed into a path planning in an N-dimensional configuration space. Configuration space is an enlarged space obtained by the N-dimensional discretization of a physical space. Passing in C-space, robot is shown as a point. C-space model is large, so the execution time is long. The method is used for solving situations A2, B2, C1. (Lozano-Perez 1983; Ilari et al. 1990; Zhu et al. 1991, Greenspan 1996). By using different strategies for building and searching the C-space, this method can be significantly improved.

II) *Network of equidistant path.* If obstacles are polygonal, then very often general Voronoi diagrams (GVD) are used. Designed path consists of linear and parabolic line segments. Free space design is transformed into freeways design. Mathematical model of freeways is small, therefore the execution is fast. The method is used for solving situations A2, B2, C1. (Takahashi et al. 1989; Ilari et al. 1990, Crnekovic 1990, Crnekovic 1991)

III) *Potential field method.* If all obstacles in the space and the goal point are known, then it is possible to design a potential field of a free space (or its gradient). Potential field defines the path from any point in the free space to the goal point. Execution time is acceptable, but it is difficult to obtain an optimal path in a global sense The method is used for solving situations A1, B1, C1. (Arkin 1989; Noborio et al. 1990, Kyriakopoulos et al. 1996)

IV) *Fuzzy logic approach.* The main advantage of this approach is no need to build a model of free space. From received information (obstacle distance and speed), static and dynamic degrees of the danger are determined which define the degree of collision danger. From the degree of collision danger and prepared decision tables, the behavior of the obstacle avoidance is decided. Execution is fast and the method is successful. The method is used for solving situations A1, B1, C2. (Kubota et al. 1990)

2. Concept Definition and Solving Method

Path planning problem is limited to the two-dimension Euclid space bounded by setting limits. Mobile robot is a rectangle and has three degrees of freedom: translation along x-axes, translation along y-axes and rotation around z-axes. It means that the robot dimensions are of the same order as the freeway model. Number, shape and obstacle dimensions are arbitrary. The robot start point and orientation $R_S(x_S, y_S, \varphi_S)$, and the goal point $R_G(x_G, y_G, \varphi_G)$ are set, like all obstacles, in the space. According to the state of the space, it is necessary to design a path from R_S to R_G. The requirement

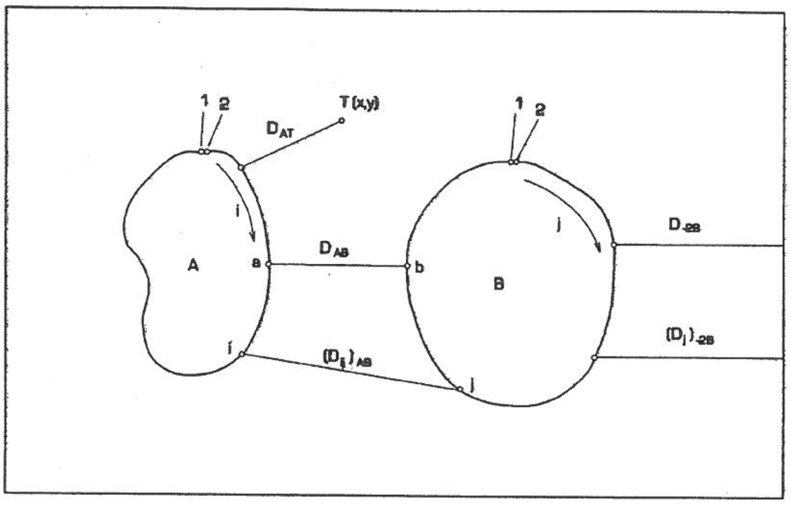

Fig. 1. Point to obstacle distance D_{AT} and obstacle to obstacle distance D_{AB}

is that no collision occurs between the robot and any obstacle by satisfying optimal criterion (the shortest path). It means that we must find a set of vectors R_i through which the robot must pass. It is supposed that the position changes and the obstacle shape changes are very small or zero during the time when the robot is passing from R_S to R_G. Also, it is supposed that there is no change of the robot dimensions and shape. Space boundaries are also obstacles.

It follows from the problem definition and environment conditions that the designed path must be optimal in a global sense. Also, the aspiration is that a method should be fast and reliable, and a free-space model small and simple. Thus, we get a practically usable algorithm with good characteristics. As this task is very complicated, the path searching process has bean divided in three steps:

Step 1. Network of equidistant path design

Step 2. Optimal path selection

Step 3. Movement simulation along the path (nominal path searching)

2.1. Obstacle Definition and Distance Calculation

We consider a bounded and finite, two-dimensional Euclid space filled by obstacles. Only inside these bounds the path searching process will be carried out. Space dimensions are 321×264 units (on the screen one unit corresponds to one pixel). It means that the space model is discrete and the least dimension length is 1, no matter which length unit.

Every obstacle **P** in this space is defined by its boundary that must be closed. It means that the end of its bound must be connected to its beginning. *Obstacle shape is arbitrary.* Obstacle bound is defined as a set of points, noted by a field **P** of every obstacle, in the form of setting points (x_i, y_i). Obstacles are marked by numbers from 1 to N_{Ob}, and space limits (walls) by numbers -1 (up), -2 (right), -3 (down) and -4 (left).

The complete algorithm for network of equidistant path design is based on two basic calculations:

- point to obstacle distance D_{AT}
- obstacle to obstacle distance D_{AB}

A distance of a point $T(x, y)$ from an obstacle **A** will be defined as the distance of the point T from a point on the obstacle **A** that is closest to the point T, Fig. 1. A distance of two obstacles **A** and **B** is a distance of the points on obstacles **A** and **B** that are closest to each other, Fig. 1. For an obstacle set points, distance to point is calculated by $O = \sqrt{N_A}$ complexity, and obstacle to obstacle distance is calculated by $O = \sqrt{N_A N_B}$ complexity. This is valid in the phase of initialization. During the network arc design, D_{AT} calculation does not depend on bound points number N_A of the obstacle **A**, and D_{AB} calculation is not necessary.

3. Network of Equidistant Path

The network of equidistant path consists of two types of elements: nodes and arcs, Fig. 2. A network node is a node that is equidistant to *three* nearest obstacles. The node is described by the set of three numbers (P_A, P_B, P_C). Numbers P_A and P_B are obstacle numbers which tell us between which obstacles the arc has been designed. Number P_C is the obstacle number that tells us which obstacle stops the arc design. A node is also the place where arcs branching takes place (if branches exist). Maximums of three arcs are possible from every node. It is valid for true nodes. In addition, there are also pseudo-nodes. Maximum number of pseudo-nodes is four. Pseudo-nodes are start and goal robot positions, and two points on the network that are closest to the start and goal positions. Pseudo-nodes have not the properties of true nodes (except if they are the same ones).

Network arcs are connections between the nodes and are equidistant among obstacles. Arcs are designed only among obstacles **A** and **B** for which it is:

$$D_{AB} \geq H_R + 2d_{\min} \qquad (1)$$

By this criterion, a condition is set so that robot of width H_R must not approach any obstacle less than the setting value $d_{\min}$.

In the network path design it is supposed that a mobile robot has no dimensions, i.e. it is a point. None of the robot dimensions are taken into account except robot width, equation (1). Arcs are designed to be *equidistant* from two nearest obstacles **A** and **B**. The path obtained in that way will, in the first approximation, permit maximum width. Because obstacle shapes are arbitrary, the method is called General Equidistant Path method (GEP).

If in a point T, a scalar function is defined as:

$$\Delta_{AB} = \Delta = |D_{AT} - D_{BT}|, \qquad (2)$$

then the equidistant path between obstacles **A** and **B** is defined by a set of points where $\Delta = 0$, and points continue side by side. Some authors define function Δ as a potential field function. In that case, the searching path is potential field minimum between obstacles **A** and **B**. Because then appears a problem of local minima, instead of using a potential field function, in this paper the function defined by equation (2) is accepted.

The network of equidistant path is defined as:

$$network = \{(x, y)|\Delta = 0\} \qquad (3)$$

As space model is in discrete form, demand $\Delta = 0$ is realized by $\Delta = \min$. From the practical point of view it is necessary to find an algorithm that will realize the equation (3). The design of every network arc starts at some node $T = \text{Node}(x, y)$. Further procedure is as follows:

Step 1. From the table of network description we shall take obstacles **A** and **B** (among which the equidistant arc will be designed), and the arc starting point - node T. It is also necessary to calculate starting index of direction $s = 1,...,8$ (see later in the network description).

Step 2. Next point from the environment of T is taken as a new point on the arc (new point T). The selected point must satisfy the request $\Delta = \min$.

Step 3. The distance of a new point T from all other obstacles, which are not **A** and **B**, has been calculated.

If the condition

$$(D_{CT} \leq D_{AT}) \vee (D_{CT} \leq D_{BT})$$

is satisfied, arc design is stopped. Instantaneous point T has the meaning of a node (new or old one). Step 1. If the condition is not satisfied, step 2.

If in step 1 it is not possible to find a new pair of obstacles **A** and **B** and new node - arc starting point, the procedure is stopped. It means that the network is finished.

At the beginning of the network design, none of the nodes are known. To find the first node arc, design begins from a point T_0 - half distance D_{AB}, because it certainly lies on the equidistant path but need not be a network node. After that, step 2 of the algorithm is carried out.

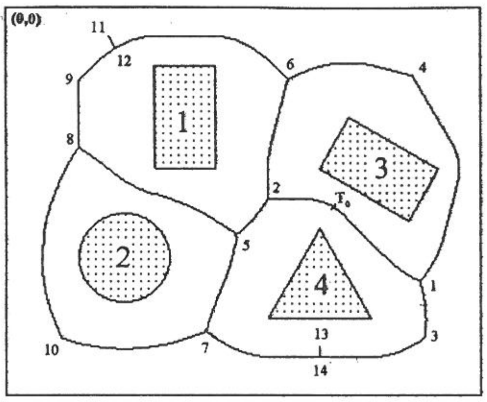

Fig. 2. Network of path

3.1. Path Network Description

Path network, i.e., freeway model has been described in two different ways: by the picture in the computer video-memory and by the network table. Network table has been built during the network design. It describes network nodes and their relations. An example of the network description is given in Table 1 and corresponds to the network in Fig. 2.

Meanings of single columns are as follows:

Node	ordinal number of the node (the last four nodes are always pseudo-nodes). If the total numbers of nodes are N_{nd} then:
node $(N_{nd}-3)$	- start node
node $(N_{nd}-2)$	- net point which is closest to the start node
node $(N_{nd}-1)$	- goal node

Node	x	y	P_A	P_B	P_C	N_1	N_2	N_3	s_1	s_2	s_3	L_{N1}	L_{N2}	L_{N3}	D_{AB}	D_{AC}	D_{BC}	D_n
1	279	186	4	3	-2	2	3	4	6	3	8	128	40	164	34	75	29	82
2	175	130	4	3	1	1	5	6	1	4	7	128	35	87	34	80	71	80
3	281	224	4	-2	-3	1	14	-1	8	4	0	40	76	0	75	52	0	78
4	274	46	3	-2	-1	1	6	-1	3	5	0	164	93	0	29	75	0	92
5	153	155	4	1	2	2	7	8	8	3	6	35	74	129	80	73	33	92
6	189	49	3	1	-1	2	4	12	4	8	6	87	93	130	71	75	40	98
7	133	219	4	-3	2	14	5	10	2	1	4	85	74	103	52	73	63	84
8	50	96	1	2	-4	5	9	10	1	7	4	129	46	144	33	100	50	100
9	50	50	1	-1	-4	12	8	-1	8	3	0	33	46	0	40	100	0	100
10	38	224	2	-3	-4	7	8	-1	2	7	0	103	144	0	63	50	0	76
11	70	20	0	0	0	12	0	0	2	0	0	9	0	0	0	0	0	0
12	74	28	0	0	0	11	9	6	6	4	1	9	3	130	0	40	40	0
13	210	230	0	0	0	14	0	0	3	0	0	7	0	0	0	0	0	0
14	210	237	0	0	0	13	3	7	7	1	5	7	76	85	0	52	52	0

Table 1. Network description from Fig. 2.

node N_{nd}	- net point which is closest to the goal node
x, y	node coordinates
P_A P_B P_C	obstacle numbers which participate in node design. P_A and P_B are obstacles among which the arc has been designed. P_C is the obstacle that stops the arc design when the node is reached. For pseudo-nodes these columns are always 0 because they are not designed by the equidistant criterion.
N_1 N_2 N_3	node numbers by which the node is connected. Number -1 means there is no connection. For pseudo-nodes the same sense has number 0. From these columns it could be found out how many arcs are coming out of the node (it is necessary to count numbers that are greater than zero). In the first column there is always a number greater than 0 and it tells us from which node the arc design has been started when the node has reached it for the first time.
s_1 s_2 s_3	index directions from the node to connected nodes mentioned in columns N_1 N_2 N_3. Where in column N_i there is no connection (0 or -1) in s_i column states 0. Index directions have the following meanings: $s = 0$ means right, $s = 3$ means down, $s = 5$ means left and so on.
L_{N1} L_{N2} L_{N3}	arcs length to connected nodes mentioned in columns N_1 N_2 N_3.
D_{AB} D_{AC} D_{BC}	obstacle distances $\mathbf{P}_A-\mathbf{P}_B$, $\mathbf{P}_A-\mathbf{P}_C$ and $\mathbf{P}_B-\mathbf{P}_C$ for the mentioned node.
D_n	node diameter.

Freeways model defined in that way consists of all elements that are necessary for path defining from the start to the goal. Also, the model is small and easily examined, with clear physical idea of the freeway model.

4. Optimal Path Selection

After the table of the network description has been done, it is necessary to define a criterion for the path selection. For this purpose we will define a cost function w that will be minimized. Because the most important are the path length and its width, the cost function could be stated as:

$$w = \alpha L(i,j) + \beta \frac{1}{D_{(AB)i,j}} \tag{4}$$

In this paper the criterion of the *shortest* path from the start to the goal has been defined ($\alpha = 1$, $\beta = 0$), so the cost function is $w = L(i,j)$. Dijkstra algorithm (Carre 1979) is one of the most efficient algorithms for a network minimum searching when $w > 0$ (for the defined cost function this is always satisfied). The time for Dijkstra algorithm is in the order $O(N_{nd}^2)$, where N_{nd} is the total number of network nodes. Because the number of network nodes is always relatively small, no heuristic search function (grass fire for example) is used.

5. Movement Simulation Along the Optimal Path

Equidistant path designed in the previous two steps is still only path's first approximation, i.e., the optimal path, defined by the network nodes. The only condition in the optimal path design, which takes into account robot dimensions, is stated by the equation (1) and requires enough width among obstacles. This condition is necessary but not sufficient for the robot movement along the optimal path without contact with any obstacle. If we want to be sure that the robot will not touch any obstacle along the path movement, simulation along the optimal path must be done. Simulation takes into account robot dimensions and real environment conditions on the optimal path. Thus, the main reason for the movement simulation is detection of a possible contact (collision) detection with obstacles during the robot motion along the optimal path, and its avoidance.

A mobile robot shape is defined by a rectangle, Fig. 4a, because this is the most frequent robot's shape. On the robot main axis, points P and Z are defined. Referring to the robot coordinate system their coordinates are: $P(L_R/4, 0)$,

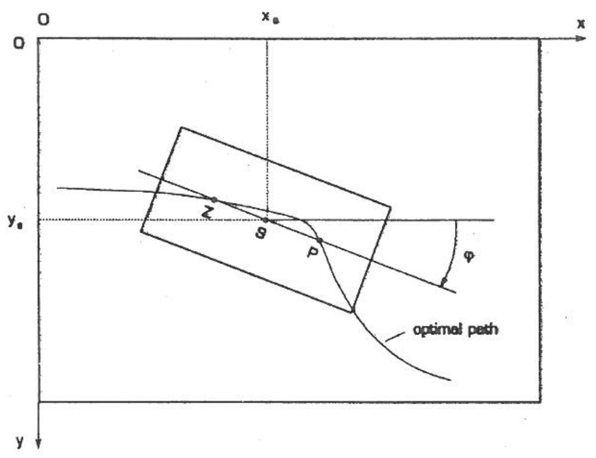

Fig. 3. Robot position and orientation on the designed path

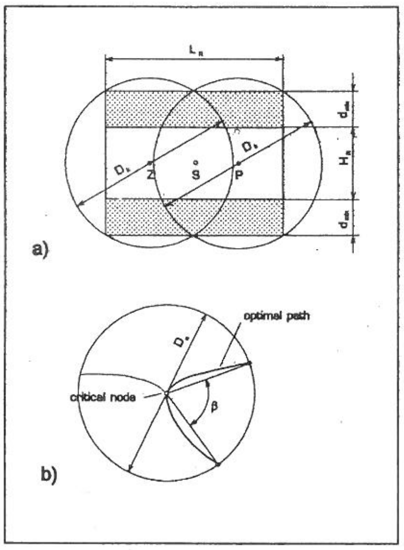

Fig. 4. Mobile robot shape (a) and conditions for movement reversion in a node (b)

$Z(-L_R/4, 0)$. The distance between points P and Z is always $L_R/2$.

During the robot motion along the optimal path, *points P and Z will always try to be on the optimal path.* It means they will define robot position S and orientation φ, Fig. 3. In that way, any sudden changing of the robot orientation is avoided where there is a great curvature on the optimal path.

5.1. Critical Movement Between Nodes and Through the Critical Nodes

Let the mobile robot move among obstacles that are distant D_{AB}. If the robot shape were a circle, then the robot with maximum diameter D_{AB} could pass among obstacles. Because the robot is a rectangle, its shape could be replaced by two circles with centers in the point P and Z, Fig. 4a. Circles also include safety distance d_{min}. New shape completely includes robot real rectangle shape. As points P and Z are always on the optimal path, any contact between the robot and obstacles A and B is not possible if their distance D_{AB} is greater than critical diameter D_k:

$$D_k = \sqrt{\left(\frac{L_R}{4}\right)^2 + \frac{1}{4}(H_R + 2d_{min})^2} \qquad (5)$$

If $D_{AB} < D_k$ then the contact between the robot and obstacle **A** or **B** (or booth) is possible. Whether it will occur depends on equidistant path curvature between the obstacles.

Robot will move through the node if at least one node of the optimal path lies among points P and Z. Let a critical node diameter, Fig. 4b, be defined by a robot diagonal:

$$D_{nk} = \sqrt{L_R^2 + H_R^2} \qquad (6)$$

then the method of the robot passing through the node depends on the relation between the node diameter and the critical node diameter.

If $D_n > D_{nk}$, then the robot passing through the node is safe, therefore collision checking during the robot motion through the node is not necessary.

If $D_n \leq D_{nk}$, the contact between the robot and obstacles is possible, and collision checking during the robot motion through the node is necessary.

Nodes are places where the robot turning aside is maximum. If the node diameter is less than critical and turning is sharp, then reverse movement planning is the best. Reversion is possible if the critical node has the third arc. The algorithm for the movement reversion could be divided in two steps:

Step 1. Point P turning from the optimal path to the auxiliary arc (the third arc of the critical node) and movement along the auxiliary arc until the point Z reaches the critical node, Fig. 5a.

Step 2. Points P and Z exchange the sense, and the robot returning from the auxiliary arc on the optimal path until the point Z reaches the critical node, Fig. 5b.

After that procedure, front and back sides of the robot will change place. Because the reverse movement condition mostly depends on the turning, i.e., path curvature, it is defined by the angle β, Fig. 4b. Depending of the D_n and β we can define three types of path nodes. They are as follows:

$$\begin{aligned}&\text{if } D_n > D_{nk} \text{ then NodeType } = 0\\ &\qquad\text{else if } \beta > \beta_k \text{ then NodeType } = 1\\ &\qquad\qquad\text{else NodeType } = 2;\end{aligned}$$

It is set $\beta_k = 100°$. The node type meaning is as follows:

NodeType $= 0$ there is no need for collision checking during the movement through the node
NodeType $= 1$ collision checking is necessary during the movement through the node
NodeType $= 2$ movement reversion and collision checking are necessary

Because of the movement reversion the path length is increased by the double amount of needed third arc length, or approximately L_R. If the collision between the robot and the obstacles is detected during the simulation of motion reversion on the optimal (equidistant) path, then the gradient of widest pass algorithm tries to solve the conflict situation. In step 1 of reversion it determines the maximum gradient of pass between obstacles **A** and **B** according to the arc *before* the node. In step 2 of reversion, it determines the maximum gradient of pass between obstacles **A** and **B** according to the arc *after* the node.

5.2. Gradient of the Widest Pass

If collision is detected during the motion along the optimal path, the conflict situation is solved by the gradient of the widest pass algorithm (GWP). The idea is to find the position between the two nearest obstacles where the path width is maximum, and to orientate the robot at that direction. Thus, we try to solve conflict situation by changing the robot orientation, but small changes in its position also occur.

In Fig. 6 an example of GWP is shown. First, it is necessary to find contact points K1 and K2 between the robot and obstacles from both sides. If there is no contact from one side, then the contact point is the point at the obstacle that is closest to the robot from that side. Then for each contact point K, adequate points M1 and M2 must be found. Each point M is the point on the obstacle from the robot opposite side, and at the same time closest to K1 or K2.

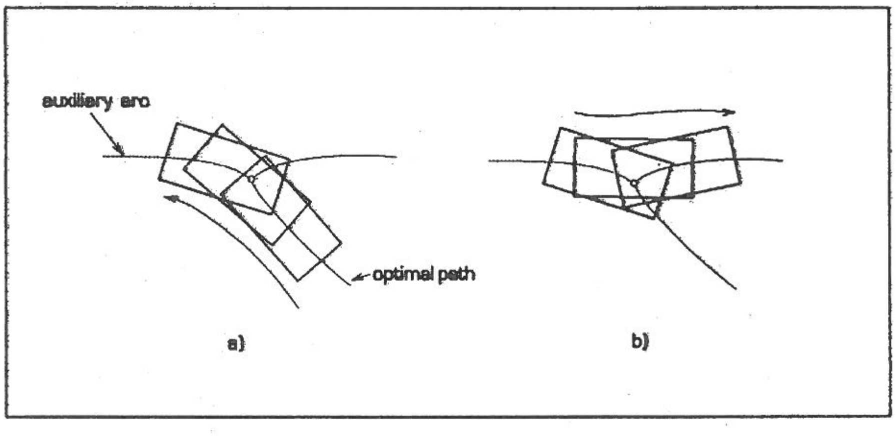

Fig. 5. Mobile robot coming to auxiliary arc (a) and return to the optimal path (b)

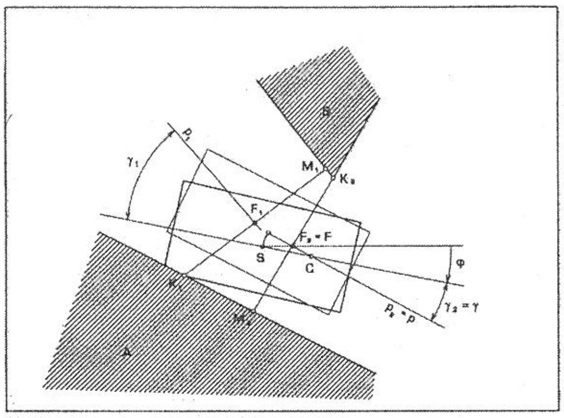

Fig. 6. Gradient of widest pass

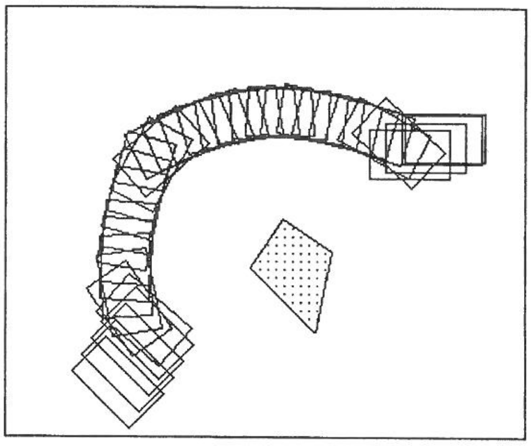

Fig. 7. Simple example with one obstacle

The half point of the length K_1M_1 is the point F_1 and through it passes a straight line p_1 that is perpendicular to the length K_1M_1. The half point of length K_2M_2 is the point F_2 and through it passes a straight line p_2 that is perpendicular to the length K_2M_2. Straight line p_1 and the main robot axis define the angle γ_1, and line p_2 defines angle γ_2.

The GWP algorithm from angles γ_1 and γ_2 selects the angle which is less. From the selected angle adequate line p and the correspondence point F are also selected. Then we could find a point C, which is the intersection of the line p and the main robot axis. The robot rotates around the point C for angle γ, i.e., until the main robot axis reaches the line p. It means that the robot aims the direction of widest pass between obstacles **A** and **B**. Using a simple analytic geometry we can calculate required values. By this new robot position and orientation, points P and Z need not be on the optimal path, but they lie close to the optimal path.

GWP is a set of rules and methods that are heuristic. It is not possible to prove that it will always give a solution. GWP could give a wrong solution if one of the obstacles is concave, because then it is possible that more than one point M exists on the concave obstacle. But, in most cases solutions of the GWP algorithm are satisfactory.

6. Examples

All algorithms for the path searching have been tested on many examples. Some of them that are the most interesting in the sense of complexion and realized paths are presented. Basic

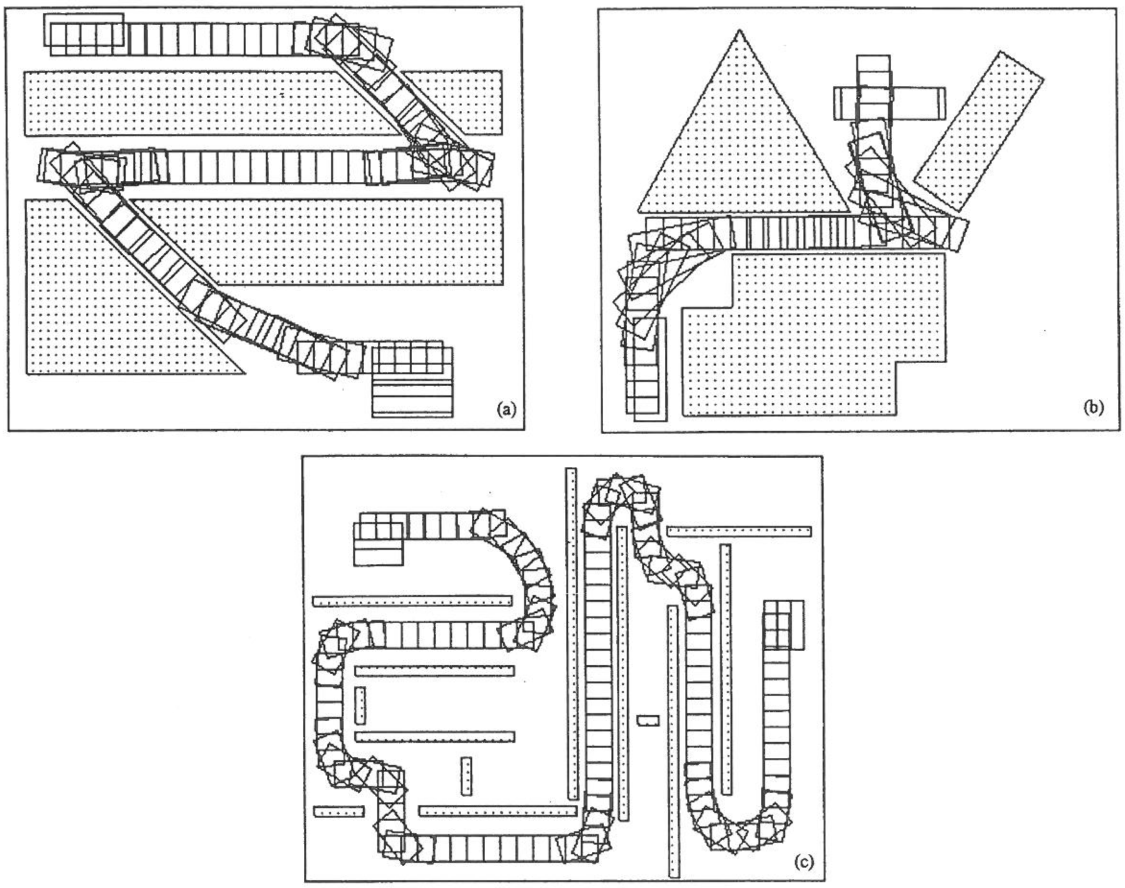

Fig. 8. Three examples from Takahashi et al. 1989

characteristics for all examples are given in the Table 2. In all examples $d_{\min} = 1$ is set.

Fig. 7 shows the simplest problem with only one obstacle (excluding bounds) and the path solution. In Fig. 8 there are three complex solutions from Takahashi et al. 1989, and in Fig. 9 there is the solution from Zhu et al. 1991. Fig. 10 shows solutions for the robot movement in the space cluttered with obstacles, and in Fig. 11 concave obstacles are presented.

The most complex situations are the maze-type problems and emphasis is placed on the ability and importance of the global optimal path planning. Figs. 8c and 12 show maze-type problems. Solution time has increased, but it does not exceed 1 second in the most complex example (for used grid size and up to 15 obstacles). If the number of obstacles is the measure for the problem complexity (although it is only partially true), then the solution time for equidistant net-

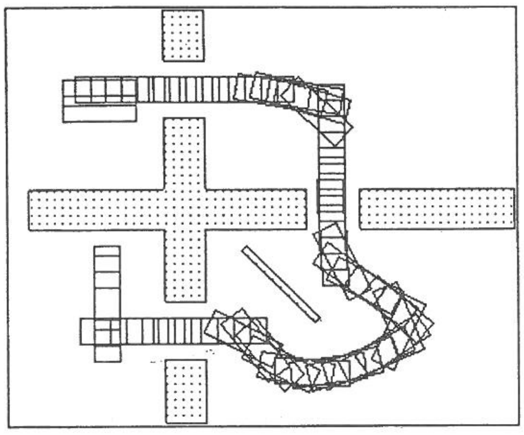

Fig. 9. Example from Zhu et al. 1991

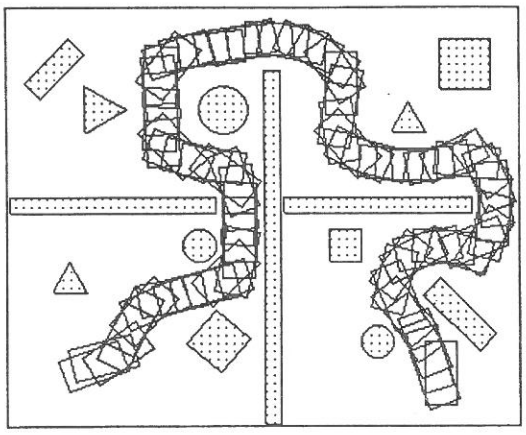

Fig. 10. Movement in the presence of many obstacles

Figure	Number of obstacles	Number of nodes	Network points	Path length
7	1	8	545	368
8a	4	14	1037	852
8b	3	12	819	373
8c	13	32	1689	1231
9	5	16	596	583
10	14	32	1293	805
11	5	16	1075	565
12	14	31	882	557

Table 2. Solutions characteristics

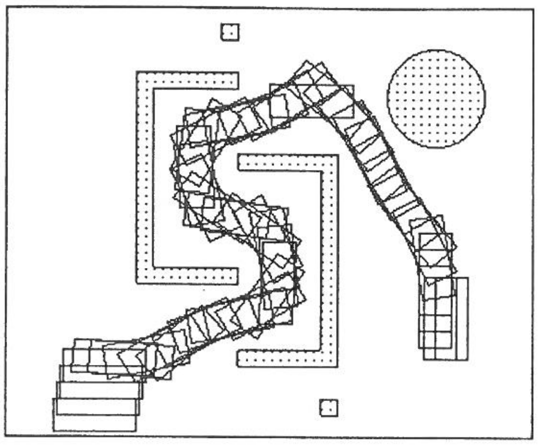
Fig. 11. Movement in the presence of concave obstacles

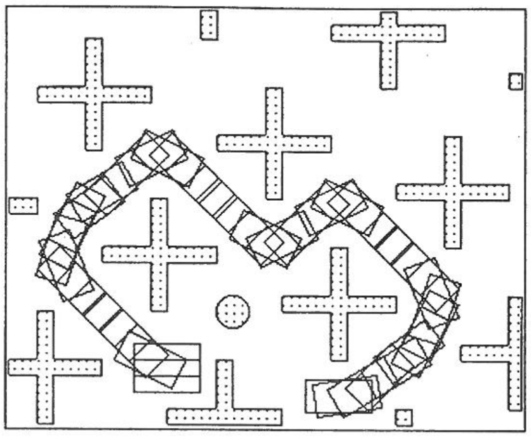
Fig. 12. Maze-type problem

work increases linearly with the number of obstacles (correlation coefficient is 0.93). While the time for the optimal path selection is small, compared by the total time, time for simulation along the optimal path could be significant. It mainly depends on calling GWP algorithm and can reach the time for the network design. As solutions are very fast, it is not possible to measure their accurate values by the PC system clock. All simulations have been done on Pentium 166 MHz and using Turbo Pascal 7.0 programming language (16 bit compiled EXE file).

For the execution control level of the mobile robot, it is necessary to design dependence $R(t)$, i.e., their three components $x(t)$, $y(t)$ and $\varphi(t)$. In Fig. 13 this dependence is shown for the example in Fig. 10. The diagram is designed on the presumption of the robot constant resultant velocity along the path.

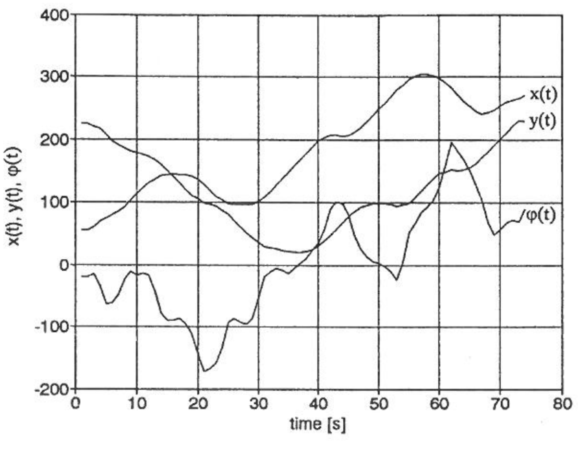

Fig. 13. Diagram $R(t)$ for the example in Fig. 10

7. Conclusion

The GEP-GWP method for the autonomous mobile robot path planning, stated in this paper permits global optimal path planning in the presence of static obstacles of arbitrary shape. Network design with generalized equidistant paths (GEP) has some advantages:

- design is very fast (proportional to number of obstacles)
- freeway model is small
- there are no limits for obstacles shape
- path problem is solved in only two dimensions
- GEP method is suitable for parallel processing

On the other hand, there are some disadvantages of the GEP method:

- if more than three obstacles are equidistant from a node, the algorithm (in present state) cannot find a solution
- path is sensitive to obstacle concavity, which results in path length and time increase
- in extreme concavity shape decomposition is necessary
- if obstacles are very close, the path between them is sensitive, i.e. it is hard to find an equidistant point
- selected path is longer than if a person would design it

The third dimension of movement (rotation) is solved only in a movement simulation along the optimal path. When collision between the robot and an obstacle is predicted, the gradient of the widest pass algorithm (GWP) tries to solve the situation. Advantages of this method are:

- there is no sudden change in the robot orientation
- robot is logically oriented (tangential on the path)
- third dimension is solved only along the optimal path

The main disadvantage of GWP algorithm is less reliability if one of the obstacles is concave. In comparison with other methods, GEP-GWP method is very fast and reliable enough. Solution times are comparable with human reaction faced with the same problem.

References

ARKIN R.C. (1989), Motor Scheme-Based Mobile Robot Navigation, *The International Journal of Robotics Research*, Vol. 8, No. 4, pp. 92–112.

CARRE B. (1979), Graphs and Networks, *Clarendon Press*, Oxford.

CRNEKOVIC M. (1990), Path Planning for Mobile Robot, *International Symposium: Automatization and Measurement Technique*, Viena, pp. 21–22.

CRNEKOVIC M. (1991), Autonomous Mobile Robot Path Planning – Problems and Methods, *International Symposium: Flexible Automation*, Strbske Pleso, pp. 30–31.

ILARI I. AND TORRAS C. (1990), 2D Path Planning: A Configuration Space Heuristic Approach, *The International Journal of Robotics Research*, Vol. 9, No. 1, pp. 75–91.

GREENSPAN M. AND BURTNYK N. (1996), Obstacle Count Independent Real-Time Collision Avoidance, *IEEE Conference on Robotics and Automation*, Minneapolis, pp. 1073–1080.

KUBOTA T. AND HASHIMOTO H. (1990), A Strategy for Collision Avoidance Among Moving Obstacles for a Mobile Robot, *11th IFAC World Congress*, Tallin, Volume 9, pp. 103–108.

KYRIAKOPOULOS K.J., KAKMBOURAS P. AND KRIKELIS N.J. (1996), Navigation of Nonholonomic Vehicles in Complex Environments with Potential Fields and Tracking, *IEEE Conference on Robotics and Automation*, Minneapolis, pp. 3389–3394.

LOZANO–PEREZ T. (1983), Spatial Planning: A Configuration Space Approach, *IEEE Transaction on Computers*, Vol. C–32, No. 2, pp. 108–120.

NOBORIO H., WAZUMI S. AND ARIMOTO S. (1990), An Implicit Approach for a Mobile Robot Running on a Force Field Without Generation of Local Minima, *11th IFAC World Congress*, Tallin, Volume 9, pp. 85–90.

TAKAHASHI O. AND SCHILLING R.J. (1989), Motion Planning in a Plane Using Generalized Voronoi Diagrams, *IEEE Transaction on Robotics and Automation*, Vol. 5, No. 2, pp. 143–150.

ZHU D. AND LATOMBE J.C. (1991), New Heuristic Algorithms for Efficient Hierarchical Path Planning, *IEEE Transaction on Robotics and Automation*, Vol. 7, No. 1, pp. 9–20.

Received: January, 1998
Revised: March, 1999
Accepted: April, 1999

Contact address:
Mladen Crnekovic, Branko Novakovic and Dubravko Majetic
Faculty of Mechanical Engineering and Naval Architecture
Automation Department
Ivana Lucica 1
10000 Zagreb
Croatia
phone: +385 1 616 8435
fax: +385 1 615 6940
e-mail: mladen.crnekovic@fsb.hr

MLADEN CRNEKOVIC was born in Croatia in 1959. He received the B.Eng. degree in mechanical engineering from the Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb (FSB), Croatia, in 1984, M.S. degree (1988) and Ph.D. degree (1992) both in control engineering, from the same institution. In 1985 he became a scientific assistant, and since 1998 has been a professor at the Control and Robotics Department. His research interests are in robotics (collision free path planning, mobile robots, programming, simulations, vision systems) and microcontrollers (system software, programming, communication). He has published 22 scientific papers and is co-author of the book Industrial Robots. He is one of the founders and the actual copresident of the Croatian Robotics Society.

BRANKO NOVAKOVIC received the M.S. and Ph.D. degrees in mechanical engineering from the University of Zagreb, Croatia, in 1974 and 1978, respectively. He worked in industry as a project designer in mechanical engineering from 1966 to 1974. He joined the Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb as an Assistant Professor (1974-1979), an Associate Professor (1979-1987), and since 1987 he has been a full Professor. His research interests include control systems, robotics, neural networks, and fuzzy control. He is author of two books: Control Systems (1985), and Control Methods in Robotics, Flexible Manufacturing Systems and Processes (1990), and co-author of the book: Artificial Neural Networks (1998). He has published more than 120 scientific papers in the fields of control systems, robotics, neural networks, and fuzzy control. He is a member of IEEE and of the New York Academy of Sciences. He is also a member of the Advisory Council for the Technology Development in the Croatian Academy of Sciences and Arts, and a member of the national society KoEREMA. His biography is listed in the 11th to the 15th Marquis Editions " Who's Who in the World" in 1993/1994 to 1997/1998, and in the 2nd to the 4th Editions " Who's Who in Science and Engineering" in 1994/1995 to 1997/1998. He received two national awards (ETAN award in 1984, and JUREMA award in 1985). He also received an international "Josip Juraj Strossmayer" award in 1991 for his second book.

DUBRAVKO MAJETIC (1962) received the B.Sc., M.Sc. and Ph.D. degrees all in Mechanical Engineering from the University of Zagreb, Faculty of Mechanical Enginnering and Naval Architecture (FSB), in 1988, 1992 and 1996 respectively. Presently he is lecturer in the Department of Control Engineering, FSB, University of Zagreb, and has taken part in several scientific and research projects. His interests include Artificial Intelligence, Neural Networks, Robotics and Automatic Control.