A Hierarchical Genetic Algorithm for Path Planning in a Static Environment with Obstacles

Chunmiao Wang, Y. C. Soh, Han Wang, Hui Wang
Division of Control & Instrumentation, School of EEE,
Nanyang Technological University,
IMRL Lab, 42 Nanyang Avenue, Singapore 639798.
P149510969@ntu.edu.sg

Abstract - In this paper, a new hierarchical genetic algorithm for path planning in a static environment with obstacles is presented. The algorithm of path planning in this paper is inspired by the Dubins' theorem regarding shortest paths of bounded curvature in the absence of obstacles. The algorithm is based on the Dubins' theorem to simplify the problem model, the genetic algorithm to search the best path, a special hierarchical structure of chromosome to denote a possible path in the environment, the special genetic operators for the each module, a penalty strategy to "punish" the infeasible chromosomes during searching. The performance results presented have shown that the approach is able to produce high quality solutions in reasonable time.

1. INTRODUCTION

Motion Planning for nonholonomic vehicles is attracted a wide interest in Robotics.

A good application is in the small league of RoboCup [1], where there are one ball and two teams with each team of five moving robots. A common dynamic path planning problem is to control one robot to kick the ball at a certain direction. This problem is still not totally solved, and some successful team managed to goal only with some basic techniques [2]. This paper does not consider totally solving this problem, but using the small team league of RoboCup as a static model since it is a very classic model.

There is a very interesting problem about how to planning an optimal path in a static environment, which is called static path planning and has a wide range of applications, by adding the following constraint on the dynamic path planning problem: *Only one robot can move, and all the other robots and the ball are static obstacles.* The robot which can move is the robot to kick the ball with a certain direction. Compared with dynamic path problem, the static path planning problem is much easier.

There are also some hidden constraints in the problem. The presence of lower bounds on the minimum turning radius involves curvature constraints on feasible trajectories that greatly affect the geometry of the problem. This paper assumes that the robot moving only with only one direction (The robot with such property may be called Dubins' car). Fig. 1 represents such a model.

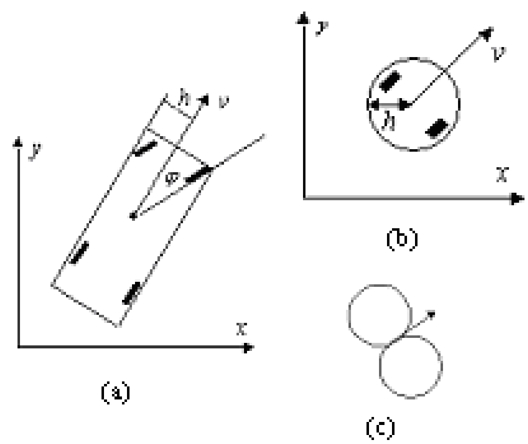

Fig. 1 (a) A generic car-like robot (b) A particular unicycle vehicle. (c) The circles with the minimum turning radius for a robot

There are several techniques which has been proposed to solve the static path planning problem: Distance Transform [3] and Lee's algorithm [4] provide ways of solving the problem by maze searching or discrete image processing, but path from these algorithms is zigzag because the maze or image provide four directions. Artificial Potential Fields (APF) algorithm tries to solve the real-time control problem by the concept of fields and forces [5], but the curve of path is far from optimal.

Dubins[6] solved the geodesic problem without reversal, while Reeds and Shepp[7] found the solution with reversals. Their research work shows that the shortest path can always be built by concatenating linear or circular segments. Similar results have been elegantly derived again by Sussmann and Tang[8] and Boissonnat, Cerezo, and Leblond[9], using Pontryagin's maximum principle. Traditionally, motion planning has been treated as a kinematic problem, i.e. determining the path that avoids obstacles without concern to robot speeds. This was first extensively addressed for articulated robots by transforming the problem into the configuration space, in which the robot reduces to a point and the obstacles map into C-space obstacles [10]. And the path planning of circle obstacles can easily be extended to path planning of polyhedral obstacles [11].

2. STATIC PATH PLANNING MODEL

For the path planning in the static environment, a model problem is as shown in Fig.2. It represents the playing field of the robot soccer.

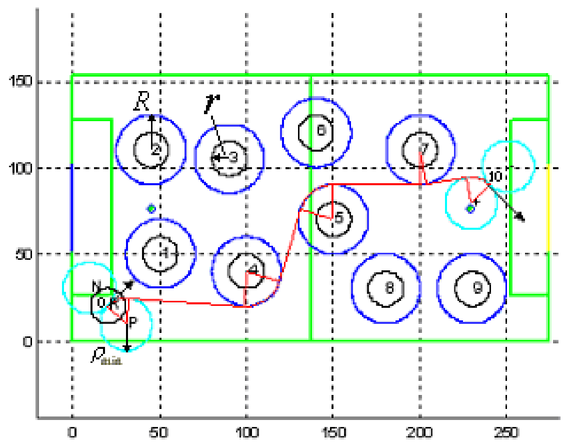

Fig. 2 The path planning model of robot soccer

For our problem, we do need to make the following assumption:

- The robot can only move in one direction as the arrow in the figure.
- The smaller circles with radius r represent the moving robot and obstacles since every obstacle is identical to the moving robot.
- The circles have the minimum turning radius $\rho_{\min}$ at the initial position and final position.

If the circle robot is transformed into a point robot, the problem after transformation is equal to the problem before transformation because the radius of each obstacle is enlarged by the radius of the moving robot. Now each obstacle circle with the radius of $R = \max(\rho_{\min}, 2r)$ and moving robot is a point robot, and the problem of path planning with obstacle avoidance is not changed.

In this model, the Dubins' theorem, which tells us that a path with the shortest length can always be built by concatenating at most five linear or circular segments, can be applied in the environment with many obstacles. We know that the optimal (or shortest) path in the environment with many obstacles must be built by concatenating a finite number of linear or circular segments.

The optimal path is somewhat like a "tight rope" in the environment. It is like a rope, with one end fixed on the same initial position of the robot, and the rope can wind a way to the target position (of course with a certain final direction) with obstacle avoidance. Just imagine that we draw the final end of the rope if the path is still loose, the length of the path will become shorter until the rope is tight as we draw the rope. So, the optimal path must be a "tight rope", and be tangential to the initial and target circles, and also tangential to any obstacle circles it passes by (a tight rope is shown in Fig.2).

2.1 Relationships between a Path and an Obstacle

Considering the relationships between the optimal path and a circle obstacle, there are only three relationships.

(1) The optimal path is not tangent to circle, that means the robot's path don't tangent (but may be intercepted) with the circle;

(2) The optimal path is tangent to circle, but the moving direction in the circle is clockwise;

(3) The optimal path is tangent to circle, but the moving direction in the circle is counter-clockwise.

The three relationships can be denoted as '0', '+1', '-1' respectively.

2.2 Genetic Algorithm (GA)

Although we know that the optimal path must be a tight path in the model mentioned above, it is still not easy to plan an optimal path. The robot must decide which way to go when it encounters an obstacle. Finding an optimal path is to search through a tree, we know the problem is an N-P harder problem. Let's consider the number of the possible (maybe not feasible) paths with obstacle avoidance.

In the initial position there are two possible directions, left or right, so are the selections in the final position. For each obstacle in the field, there are three possible ways. We should consider the enumeration given by 9 obstacles. So the number of the possible ways is $N = 2^2 \bullet 3^9 \bullet 9! \simeq 2.857 \times 10^{10}$! Of course some ways may not be feasible since the paths may intercept with the obstacles.

Genetic Algorithm [12] is implemented here since GA is well-known as a good tool for searching and optimizing methodology. And special techniques are implemented here for the problem. Here we present a novel way to solve the path planning problem in a static environment

0-7803-7282-4/02/$10.00 ©2002 IEEE

based on Dubins' theorem by hierarchical genetic algorithm.

3. PATH PLANNING BY HGA

The basic idea under hierarchical genetic algorithm is that for some complex systems which can not be easily represented by basic GA chromosome, complicated chromosomes may provide a good new way to solve the problem. There are some successful examples of HGA, one is Digital IIR Filter Design [13]. For a tight path, we should not only decide the sequence of obstacles, but also select the relationship between the obstacle and point robot. It is a bit difficult to be directly represented by binary GA. For the first part of the problem, which is about the sequence of obstacles, it is quite like the problem of Travel Salesman Problem (TSP), which can be solved by GA[14]; For the second part of the problem, which is about the selection of the relationships, it can be solved by binary GA. Combining these two techniques, we try to solve this problem by HGA. The basic steps for HGA in this paper is as follows:

1 t = 1
2 initialize population P(t)
3 compute fitness P(t) with penalty strategy
4 t = t+1
5 if termination criterion achieved go to step 12
6 select P(t) from P(t-1)
7 crossover upper module of P(t)
8 mutate upper module of P(t)
9 crossover lower module of P(t)
10 mutate lower module of P(t)
11goto 3
12 output best and stop.

3.1 Fitness Function

The static path planning in this paper is to find the shortest path with obstacle avoidance, or minimize the function $Dist(s)$, here s can be any feasible chromosome and $Dist(s)$ is the length of the path represented by chromosome s. Actually it is the sum of the length of the path which is concatenated by the segments of lines and circles. Appendix A provides the analysis for the calculation of the length.

Generally the GA is to find a chromosome with the maximum fitness, so the fitness function can be the reciprocal of the path multiply with a const, which can be defined as

$$fit(s) = \text{const}/(Dist(s)+1) \tag{1}$$

here $const$ =10000.

3.2 Penalty Strategy

In the searching process, the GA will produce a lot of infeasible solutions, which in this paper could be infeasible paths such as the paths intercepted with the obstacles. Penalty strategy is a common technique to deal with infeasible solutions by a simple idea of "punishing" the infeasible solutions with penalty.

Since we are going to find the path with the minimum length, the infeasible solution intercepted with the obstacle should be "punished" by adding some extra penalty. Appendix B provides the criterion for interception between a line and a circle with two points. Since a path is the concatenation of line segments and segments of circles, we can judge the relationship between each line segment with each circle and count the times of the interception altogether.

Here, l_i is the ith line segment in a possible path, C_j is the jth obstacle, $Intercept(l_i, C_j)$ equals '1' when the C_j is intercepted by l_i, and '0' otherwise. $\sum Intercept(l_i, C_j)$ is the times of the interception altogether for a path.

The distance function can be modified into

$$Dist_m(s) = Dist(s) + \delta \bullet \sum Intercept(l_i, C_j) \tag{2}$$

Here δ is a parameter for the penalty, and $2\pi R$ is selected. Now the fitness function should be modified to

$$fit(s) = \text{const}/(Dist_m(s)+1) \tag{3}$$

3.3 The Encoding

[the sequence of the circles]

Upper Module	0	4	5	7	1	2	3	6	8	9	10
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Lower Module	1	-1	1	-1	0	0	0	0	0	0	1

[the relationships of the path and obstacles]

Fig.3 The chromosome represents the path in the above Fig.2

Before we discuss the encoding of the string, we must know four kinds of relationships between two consecutive circles tangent with the tight path in the Appendix A. The Appendix A tells us the relationships between two consecutive circles tangent with the path, analyses the critical points and angles needed in the path, calculates the length of the path. It is the foundation of rendering the path and calculation of its length.

Fig. 3 shows a chromosome representation of the path shown in Fig.2 by HGA. This chromosome consists of the *control genes* (which is the upper module) and the *parametric genes*

0-7803-7282-4/02/$10.00 ©2002 IEEE

(which is the lower module). The activation of the *parametric gene* is governed by the value of the corresponding *control gene*. The chromosome represents the path in the Fig.3. The following is the explanation of the chromosome.

- The "0", or the first position of the upper module represents the initial position of the moving robot, and controls two selections in the lower module, namely, the left circle ("-1" in the lower module, moving direction is counterclockwise) or the right circle ("+1" in the lower module, moving direction is clockwise) to the initial moving direction.
- The "10", or the last position of the upper module represents the final position of the moving robot, and controls two selections.
- The genes from "1" to "9" in the upper module represent the corresponding obstacles denoted in the Fig.3, and each gene controls three possible selections as mentioned before: not tangent ("0" in the lower module), tangent and clockwise ("+1" in the lower module), tangent and counterclockwise ("-1" in the lower module).
- The meaning of the chromosome "**0 4 5 7 1 2 3 6 8 9 10 :: 1 –1 1 –1 0 0 0 0 0 0 1**" is a possible path in the field shown in Fig.3.

From the above explanation, we know how to express a possible path by the encoding mentioned above. But there are a lot of different chromosomes to express one path. For example, the path mentioned before can be expressed in the chromosome "**0 4 5 <u>7</u> 1 2 3 6 8 <u>9</u> 10 :: 1 –1 1 <u>–1</u> 0 0 0 0 0 <u>0</u> 1**" or in another chromosome "**0 4 5 1 2 3 6 8 <u>9</u> <u>7</u> 10 :: 1 –1 1 0 0 0 0 0 <u>0</u> <u>–1</u> 1**" since changing the sequence of the not tangent circle(s) doesn't change the meaning of the chromosome, which is a certain path. So a path or the chromosomes with the same meaning can be expressed by the concept of set. In the example above, the path can be represented as **{0 4 5 7 1 2 3 6 8 9 10 :: 1 –1 1 –1 0 0 0 0 0 0 1}**. From the experiment on this problem, we will get different chromosomes of same path. If correspond a path to a unique chromosome, the algorithm will lose some efficiency from the experiment.

3.4 Genetic operators

After the encoding of the chromosome is known, genetic operators must be selected to search the optimal chromosome according to the basic steps for HGA mentioned before.

The genetic operators are *selection* operator, *crossover* operator, and *mutation* operator. The selection operator here is *roulette wheel selection.* The crossover and mutation operators will have some changes since the structure of the upper module of the chromosome has its special features while the structure of the lower module of the chromosome is like the structure of the binary chromosome. As a matter of factor, the structure the upper module is much like the structure of chromosome to solving the problem of Travel Salesman Problem (TSP) [14], so the crossover and mutation operators which are implemented in TSP can also be used in crossover and mutation of the upper module with some variances. The crossover and mutation in the lower module is much like the crossover and mutation of the binary GA also with some variances.

This paper does not want to go much further on the variant genetic operators for the TSP. For the upper module, Partially Matched Crossover (PMX), Order Crossover (OX), or Cycle Crossover (CX) can be selected as crossover operator, and Inversion Mutation or Reciprocal Exchange Mutation can be selected as mutation operator. These genetic operators are very common in use [11]. Cycle Crossover (CX) and Inversion Mutation are selected in the experiment.

For the lower module, both crossover and mutation operators are quite commonly used. Two points crossover is selected as crossover operator. Mutation operator is very common and selects one direction from two or three way.

4. EXPERIMENT RESULTS

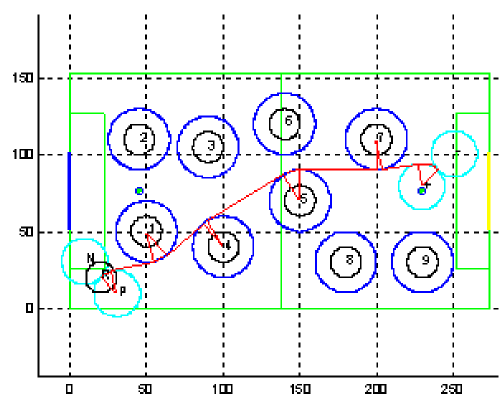

Fig. 4 The optimal path can be represented as:

```
Upper:  0  1  4  5  7 2 3 6 8 9 10
Lower: +1 -1 +1 +1 -1 0 0 0 0 0 +1
```

The algorithms were implemented in MATLAB on a PC with a Pentium III 700MHZ CPU, using the features provided above and some other parameters as follows. The population size is

50. The maximum number of generation is 50. The probabilities are selected as follows: For upper module, pc=0.4, pm = 0.02; For lower module, pc=0.4, pm = 0.02. It takes about 5 seconds to complete the searching process. Fig. 4 shows the optimal result from this experiment, which is {**0 1 4 5 7 2 3 6 8 9 10 :: +1 -1 +1 +1 -1 0 0 0 0 0 +1**}.

As the fitness over generation is shown in Fig.5, the optimal solution is show in generation 27 (assuming the initial generation is 1), and a near optimal solution is shown in generation 11. From the figure, the convergence speed is very fast.

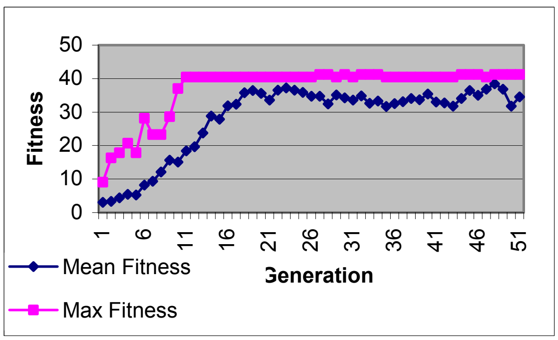

Fig. 5 Fitness over generation

5. CONCLUSIONS

In this paper, a new hierarchical genetic algorithm about path planning in a static environment with obstacles is presented. The algorithm is based on (1) the Dubins' theorem to simplify the problem model, (2) the genetic algorithm to search the best path, (3) a special hierarchical structure of chromosome to denote a possible path in the environment, (4) the special genetic operators for each module, (5) a penalty strategy to "punish" the infeasible chromosomes during searching, and (6) the theory of complex variables to make the computation of the fitness easier. The performance results presented have shown that the approach is able to produce high quality solutions in reasonable time.

There are several issues for future research. First, this algorithm may be accelerated by considering the special condition of the fields and by using special operators from the successful algorithms for TSP. The robot used here may be extended to the Reeds and Shepp cars, which can move forward and backward.

APPENDIX

A. Four kinds of relationships

The Fig. A shows a segment of a "tight path" in this paper. The point robot meats the obstacle circle O_1 first, and maybe travels a segment on the circle O_1, then travel through the line P_1P_2 to the circle O_2, and also maybe travels a segment on the circle O_2. From the discussion in the paper, the line P_1P_2 must be tangent to the circle O_1 and circle O_2. We analyse the relationships by the theory of complex variables since it is a good tool to deal with the problem of vector in a 2D space. For the space limited, we do not provide the equations here. The relationships can be denoted as '+1+1', '-1-1', '+1-1', and '-1+1' respectively.

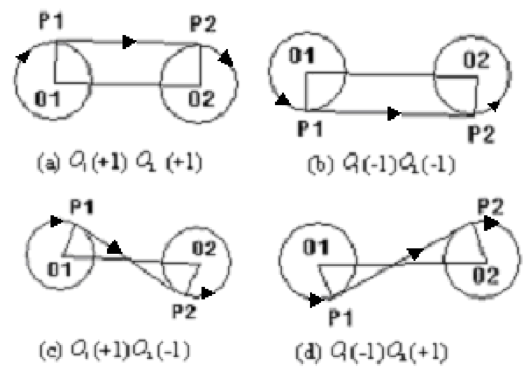

Fig. A Four relationships between two consecutive circles according to the counter-clockwise (-1) and clockwise (+1) denotes by above.

The center of circle O_1 is (x_1, y_1), and the center of the circle is O_2 is (x_2, y_2).

Let $d = |\overrightarrow{O_1O_2}|$, then $|\overrightarrow{P_1P_2}| = d \sin\alpha$, here $\alpha = \measuredangle P_1O_1O_2$. Let's make an analysis for (a), and we can use similar method to analyze (b), (c), and (d).

(a) O_1 (+1) O_2 (+1):

$$\alpha = \arccos\frac{r_1 - r_2}{d} \tag{4}$$

$$\overrightarrow{OP_1} = \overrightarrow{OO_1} + \overrightarrow{O_1P_1} = (x_1 + iy_1) + r_1\left[\frac{(x_2 - x_1) + i(y_2 - y_1)}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}\right](\cos\alpha + i\sin a) \tag{5}$$

$$\overrightarrow{OP_2} = \overrightarrow{OO_2} + \overrightarrow{O_2P_2} = (x_2 + iy_2) + r_2\left[\frac{(x_1 - x_2) + i(y_1 - y_2)}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}\right](-\cos\alpha - i\sin a) \tag{6}$$

B. Criterion for Interception

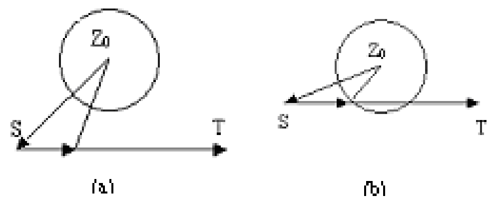

Fig.B Two relationships of a line with a circle: (a) a circle is not intercepted by a line, (b) a circle is intercepted by a line.

The criterion serves for penalty strategy. If a line is intercepted by a circle, this path is not possible, so we 'punish' this path with a penalty. Actually there are three relationships for a line segment and a circle in our problem: two are shown in Fig.B, the other one is that a line is tangent with a circle. A path with the relationship in the Fig. B(b) is not acceptable. So we need to consider the criterion for that *a circle is intercepted by a line with two points*.

The equation for the line segment is

$$\vec{S} + t(\vec{T} - \vec{S}) = Z \ , 0 \le t \le 1 \tag{7}$$

and the equation for the circle is

$$| Z - Z_0 | = R_0 \tag{8}$$

so we can called it interception if there are two solutions in the function

$$| \vec{S} + t(\vec{T} - \vec{S}) - \overrightarrow{Z_0} | = R_0 \tag{9}$$

Let $\vec{T} - \vec{S} = x_1 + iy_1$, $\vec{S} - \overrightarrow{Z_0} = x_2 + iy_2$, the equation above is transformed into

$$(x_1^2 + y_1^2)t^2 + (2x_1x_2 + 2y_1y_2)t + (x_2^2 + y_2^2 - R_0^2) = 0 \ .$$

Defining a function

$$f(t) = t^2 + \frac{(2x_1x_2 + 2y_1y_2)}{(x_1^2 + y_1^2)} t + \frac{(x_2^2 + y_2^2 - R_0^2)}{(x_1^2 + y_1^2)} \tag{10}$$

and let $b = \frac{(x_1x_2 + y_1y_2)}{(x_1^2 + y_1^2)}$, $c = \frac{(x_2^2 + y_2^2 - R_0^2)}{(x_1^2 + y_1^2)}$,

then we get

$$f(t) = t^2 + 2bt + c = (t + b)^2 + c - b^2 \tag{11}$$

So the problem is to decide if the function $f(t)$ has two solutions t_1 and t_2 ($t_1 \ne t_2$, $0 \le t_1, t_2 \le 1$).

TABLE 1.
CRITERION FOR INTERCEPTION BETWEEN A LINE SEGMENT AND A CIRCLE

<table>
<tr><th colspan="3">Criterion</th><th>Solution(s)</th></tr>
<tr><td rowspan="3">-1<b<0</td><td colspan="2">$f(-b) \ge 0$</td><td>Less than 2</td></tr>
<tr><td rowspan="2">$f(-b)<0$</td><td>$f(0) >= 0$ & $f(1) >= 0$</td><td>Equal to2</td></tr>
<tr><td>$f(0) < 0$ & $f(1) < 0$</td><td>Less than 2</td></tr>
<tr><td colspan="3">$b \ge 0$ or $b \le -1$</td><td>Less than 2</td></tr>
</table>

So the criterion for interception between a line and a circle with two points is (1) $-1<b<0$, (2) $f(-b)<0$, and (3) $f(0) >= 0$ and $f(1) >= 0$.

References

[1] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, and H. Matsubara, 1997. "RoboCup: A Challenge Problem for AI". AI Magazine 18(1): 73-85.

[2] M. Veloso, M. Bowling, S. Achim, K. Han, and P. Stone, "The CMUnited-98 champion small robot team". RoboCup-98: Robot Soccer World Cup II. Springer Verlag, Berlin, 1999.

[3] R.A.Jarvis, "Collision-Free Trajectory Planning Using the Distance Transforms," Mechanical Engineering Trans. of the Institution of Engineers, Australia, Vol. ME10, No. 3, September, 1985.

[4] C.Y. Lee. "An algorithm for path connections and its applications". IRE Trans Electronic Computers, EC10: 346--365, 1961.

[5] B. Krogh, "A Generalized Potential Field Approach to Obstacle Avoidance Control", SME - RI Technical Paper MS84-484, 1984.

[6] L.E. Dubins, "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents ", in American Journal of Mathematics, 79:497-516, 1957.

[7] J.A.Reeds, R.A.Shepp: "Optimal Paths for a Car that Goes both Forward and Backward", Pacific Journal of Mathematcis, vol. 145(2),1990.

[8] H.J. Sussmann and G.Tang, Shortest Paths for the Reed-Shepp Car: a Worked Out Example of the Usee of Geometric Techniques in Nonlinear Optimal Control. SYCON report 91-10, 1991.

[9] J.D. Boissonnat, A. Cerezo, J. Leblond: Shortest Paths of Bounded Curvature in the Plane, Proc. IEEE Int.Conf. on Robotics and Automation, pp.2315-2320, 1992.

[10] T. Horsch, F. Schwarz, and H. Tolle. "Motion planning for many degrees of freedom - random reflections at c-space obstacles". In Proc. IEEE Int. Conf. on Robotics and Automation, pages 3318--3323, 1994.

[11] T. Lozano-Perez, M.A.Wesley, An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles, Communications of the ACM, Vol. 22, No. 10, October 1979, pp.560--570.

[12] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Longman, 1989.

[13] K.F. Man, K.S. Tang, and S. Kwong, Genetic Algorithms Concepts and Designs, Springer, 1999.

[14] B. Freisleben and P. Merz, "Genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems," Proceedings of IEEE International Conference on Evolutionary Computation, IEEE-EC 96, 1996, IEEE Press, pp. 616-621.