# Simulation of Path Planning of Mobile Robot in Dynamic Environment

Prabal Bhatnagar
Assistant Professor
CS & IT Deptt.
MIT, Moradabad

Shivanshu Rastogi
Assistant Professor
CS & IT Deptt.
MIT, Moradabad

Vikas Kumar
Associate Professor
CS & IT Deptt.
MIT, Moradabad

## ABSTRACT

Path planning of a robot [1] is a problem in which a problem space or a problem domain with a number of obstacles is given and the aim is to find a collision free path which a robot can follow in order to reach its destination from the start position. Here in this paper environment is represented by ordered grids, each of which represents a location in the mobile robot movement space. Along with this we would be able to place dynamic no. of static and movable obstacles at run time. The boundary of obstacles is formed by their actual boundary plus minimum safety distance considering the size of the mobile robot, which makes it possible to treat the mobile robot a point in the environment. The path [2] which the robot will follow is desired to be optimal in terms of distance and time taken by the robot to reach the destination. Here in this problem we have used Genetic Algorithm for path planning which is a search algorithm based on the mechanics of natural selection and natural genetics. Potential solutions of a problem are chromosomes, which form a population of possible solutions which compete with each other on the basis of fitness function. A selection mechanism based on the fitness is applied to the population and the individuals strive for the survival.

## Keywords

Fitness function, Obstacles, mobile robots, path planning.

## 1. INTRODUCTION

The moment we try to find out a solution for path planning of a mobile robot, the first thing which came to our mind is about some sort of sensors mounted on the head of robot that are supposed to sense the obstacles and other difficulties that may come in calculating the path of that particular robot. Secondly we think about the nature of obstacles that may be static or dynamic (moving obstacles).[3] This paper is mainly devoted to the planning in the dynamic environment and the Genetic Algorithm is implemented thoroughly instead of using sensors in order to find the path and estimation of obstacles. Now the problem is to find a collision free path between two points in a known environment with different static and moving obstacles for motion of a mobile robot. The desired path should be the shortest path where there are no collisions between robot and obstacles and the speed of robot is also constant. One issue is to produce a path that satisfies the criteria of optimality. Very often the collision free path is highly desirable to be optimal or near-optimal with respect to time, distance or energy [4].

Genetic Algorithms are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial creatures (strings) is created using bits and pieces of the fittest of the old.

## 2. PROBLEM DESCRIPTION

In this paper the mobile robot environment is treated as a grid based environment in which each grid is represented by an ordered pair of row number and column number. The size of grids is specified by the user at the run time. Both Static and Dynamic Obstacles are also placed by the user in the workspace in which static obstacles can be placed by simply clicking on the grid and the dynamic obstacles can be placed by clicking the starting and the ending point in between of which the dynamic obstacle will move to and fro with a constant velocity. Here we are considering the mobile robot as a point in the environment. [5] The size of mobile robot can be added to the obstacles boundaries as a safety distance which allows us to treat the mobile robot as a point object. The environment of mobile robot is depicted as a rectangular grid is shown in Fig.1(Static obstacles) & Fig.2(Static and Dynamic obstacles).

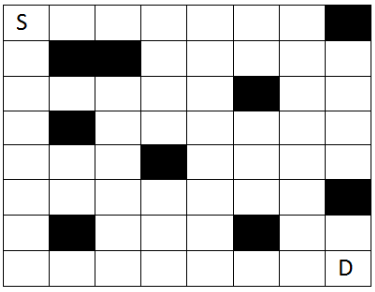


**Fig.1 Static Obstacles**

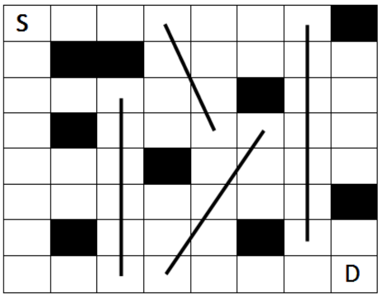


**Fig.2 Static and Dynamic Obstacles**

Along with finding the collision free path from source to destination, arrival at destination within minimum time and the minimum distance is also the key problem to be dealt.

## 3. WORKING PRINCIPLE

Genetic Algorithms [6] are basically the search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial creatures (chromosomes) is created using random bits and pieces of the fittest of the old. an occasional new part is tried for good measure which is known as mutation. Genetic algorithms despite being randomized are not a simple random walk. They efficiently use historical information to speculate on new search points which can be used to get a solution with improved performance.

### 3.1 What are Chromosomes?

Chromosomes are the initial random solutions which constitute the population. The individual chromosomes compete for survival on the basis of their evaluation by a fitness function. The process is iterative in nature and the chromosomes evolve from one generation to the other using both the old and new random solutions. After many generations, the fittest individual survives and the optimal or near optimal solution can be obtained. A chromosome may look like this:

**11111110101010010101001010101011101101**

### 3.2 What's the Crossover Rate?

This is simply the chance that two chromosomes will swap their bits. A good value for this is around 0.7(based on our calculated results). Crossover [7] is performed by selecting a random gene along the length of the chromosomes and swapping all the genes after that point.

e.g. Given two chromosomes

**10011111011000010**

**11100011101011010**

Choose a random bit along the length, say at position 9, and swap all the bits after that point. So the above become:

**10011111001011010**

**11100011111000010**

### 3.3 What's the Mutation Rate?

Mutation [8] is based on the fact of a rare chance of genetic disorder. This is the process in which a random bit within a chromosome will be flipped (0 becomes 1, 1 becomes 0). The mutation rate is usually a very low value for genes, say 0.01. So whenever chromosomes are randomly selected from the population for mutation process the algorithm iterates down the length of each chromosome mutating the bits if applicable.

## 4. FLOW CHART FOR PATH PLANNING IN STATIC ENVIRONMENT

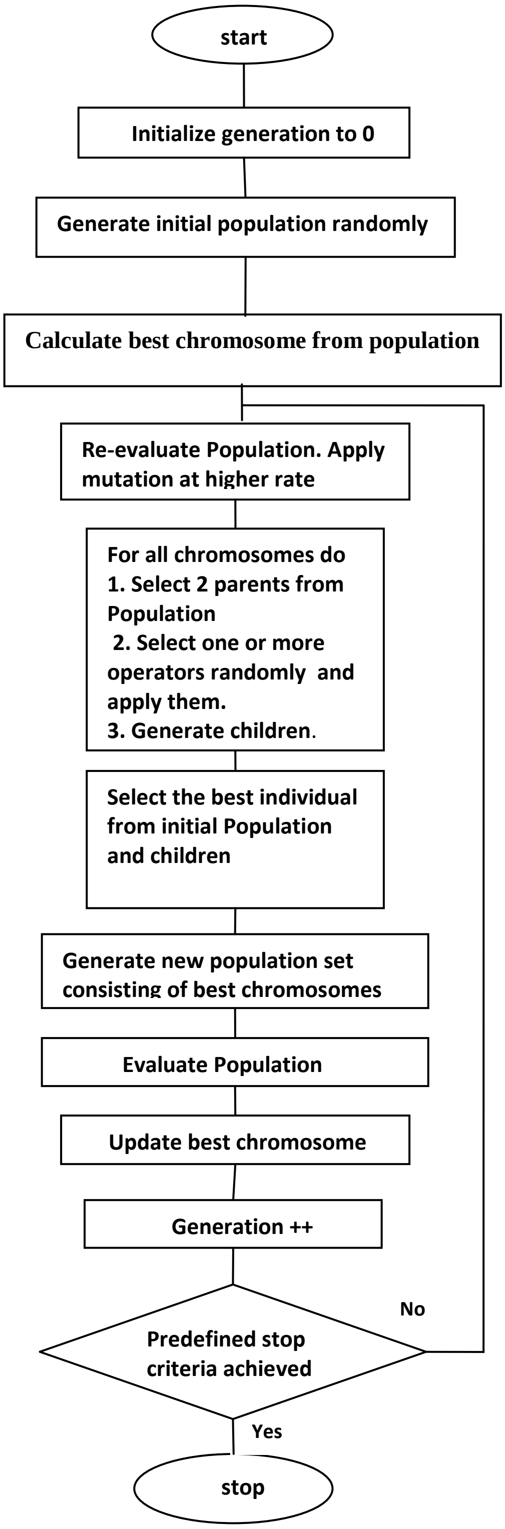

start
Initialize generation to 0
Generate initial population randomly
Calculate best chromosome from population
Re-evaluate Population. Apply mutation at higher rate
For all chromosomes do
1. Select 2 parents from Population
2. Select one or more operators randomly and apply them.
3. Generate children.
Select the best individual from initial Population and children
Generate new population set consisting of best chromosomes
Evaluate Population
Update best chromosome
Generation ++
Predefined stop criteria achieved
No
Yes
stop

**Fig.3 Flowchart of Path Planning in Static Environment**

## 5. PATH PLANNING OF ROBOT WITH DYNAMIC OBSTACLES

First of all in order to place the dynamic obstacles we need to mark the starting and the ending position of the obstacles so as to keep moving between the specified positions. A sample environment is shown in Fig.4. Some assumptions are also to be kept in mind that are:

- The dynamic obstacles are point obstacles.
- They have certain velocity.
- They move in a to and fro motion.
- They don’t have the power to think so they will not wait for the robot.

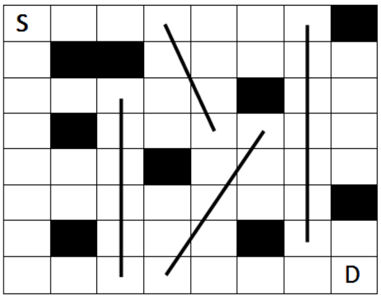


**Fig.4 Representation of environment of a mobile robot with obstacles (filled grids), S-source, D-destination, Lines shows the dynamic obstacles**.

### 5.1 Evaluation

A path can be either feasible (collision free) or infeasible because intermediate nodes can fall on any of the grids. The evaluation should be able to distinguish feasible and infeasible paths and tell the difference of path qualities within either category. The evaluation function is presented below:

$$F_{Cost} = \Sigma di + K*P$$

$$Cost = Fcost + N*P$$

Where $F_{cost}$ is the cost of path planning in static environment

Σdi is the Euclidean distance from source to destination

K is the penalty cost in for static collision

N is the penalty for each collision, for ex. N=50

P is the number of pixels colliding with dynamic obstacles.

This evaluation gives penalty to infeasible paths, but still keeps them in the population because they might become good feasible solutions after certain genetic transformations. Besides, this evaluation allows some overlap between finesses of feasible and infeasible solutions because a very poor feasible path is not necessarily better than a very good near-feasible path in the sense of evolving solutions. It is beneficial to give more chance to some good infeasible solutions that are easily to be evolved to good solutions. To save computational time, some information obtained by the evaluation needs to be recorded so that later on it can be used by some specialized genetic operators as heuristic knowledge without re-calculation. The information includes feasibility (feasible or infeasible, node-infeasible or line-infeasible), number of infeasible nodes or line segments, and which obstacle(s) a path intersects.

### 5.2 Proposed Algorithm:

1. Evaluate the location of each dynamic obstacle.
2. Choose initial population.
3. Evaluate the fitness of each chromosome in the population.
4. Repeat
    a. Select the best chromosomes among the population.
    b. Breed the selected parents using genetic operators to generate offsprings.
    c. Evaluate the fitness of each chromosome in the offsprings.
    d. Select the best among them and replace worst ranked chromosome with the best one in offsprings.
5. Repeat the process until termination condition is satisfied.
6. Exit

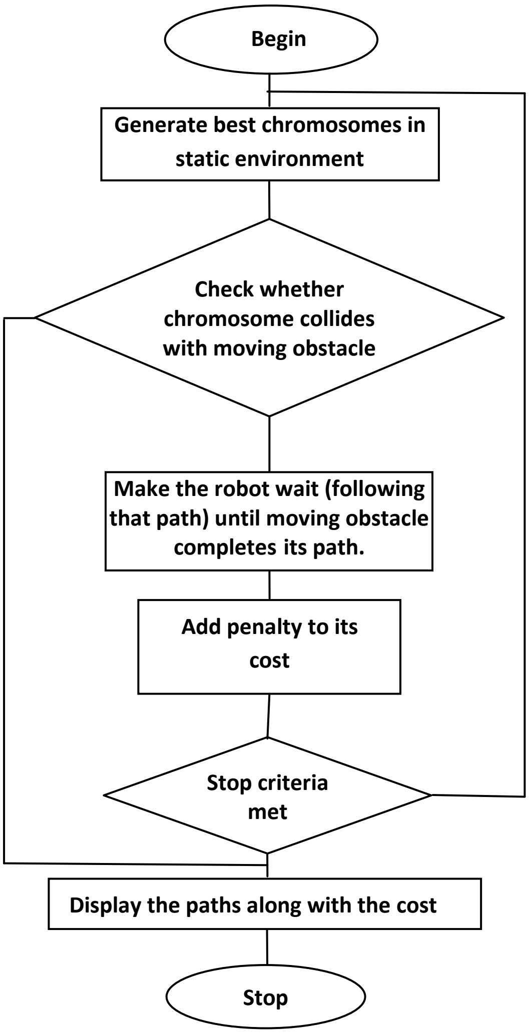

Begin
Generate best chromosomes in static environment
Check whether chromosome collides with moving obstacle
Make the robot wait (following that path) until moving obstacle completes its path.
Add penalty to its cost
Stop criteria met
Display the paths along with the cost
Stop

**Fig.5 Flowchart for Path Planning in Dynamic Environment**

## 6. GENETIC OPERATOR

Here to make the Genetic Algorithm more effective four specialized operators are designed to make use of available problem-specific knowledge including knowledge of the environment. These operators are introduced as follows:

**6.1 Crossover** is the operator that randomly chooses a node from Parent 1 and the other node from Parent 2. Exchange the part after these two nodes. Check the two offspring, and delete the part between two same nodes if it happens. The choice of different crossover sites in different parents can increase the variability of chromosome length, which benefits exploration of the solution space. The new population goes through other operators that make the solution full proof and optimistic.

**Chromosomes before Crossover**

| | | | | |
|---|---|---|---|---|
| 2 | 9 | 7 | 10 | 90 |

Crossover Point

| | | | | |
|---|---|---|---|---|
| 1 | 11 | 19 | 49 | 85 |

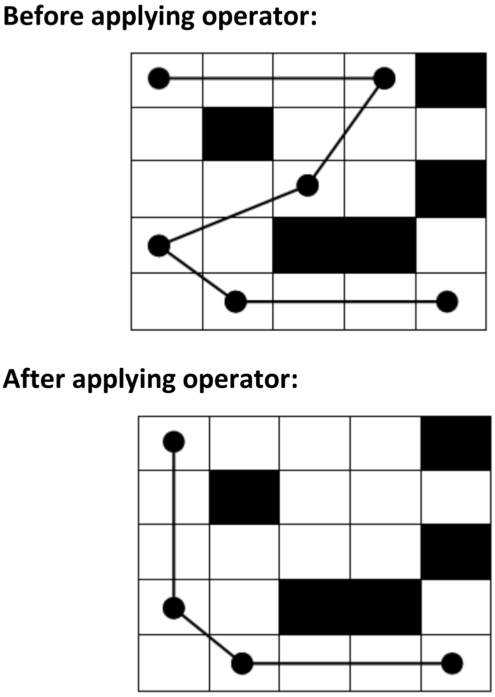
Crossover Point

**Chromosomes after Crossover**

| | | | |
|---|---|---|---|
| 2 | 9 | 49 | 85 |

| | | | | | |
|---|---|---|---|---|---|
| 1 | 11 | 19 | 7 | 10 | 90 |

**Fig. 6 Crossover Process**



Before applying operator:

After applying operator:

**Fig. 7 Crossover operator**

**6.2 Mutation** is that randomly chooses a node and replaces it with a node that is not included in the path. Mutation is served as a key role to diversify the solution population. Therefore, it is not necessary that a solution is better after it is mutated. This is also a basic genetic operator. The drawback of mutation operator is that it sometimes causes the generation of most infeasible paths and the most non-optimized paths.

**Chromosomes before Mutation**

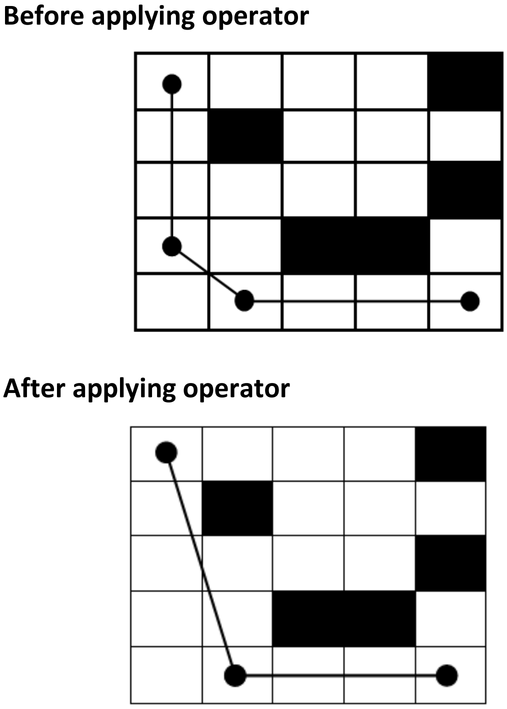
| | | | | |
|---|---|---|---|---|
| 4 | 9 | 47 | 56 | 90 |

Mutation Point

**Chromosomes after Mutation**

| | | | | |
|---|---|---|---|---|
| 4 | 9 | 32 | 56 | 90 |

**Fig.8 Mutation Process**



Before applying operator

After applying operator

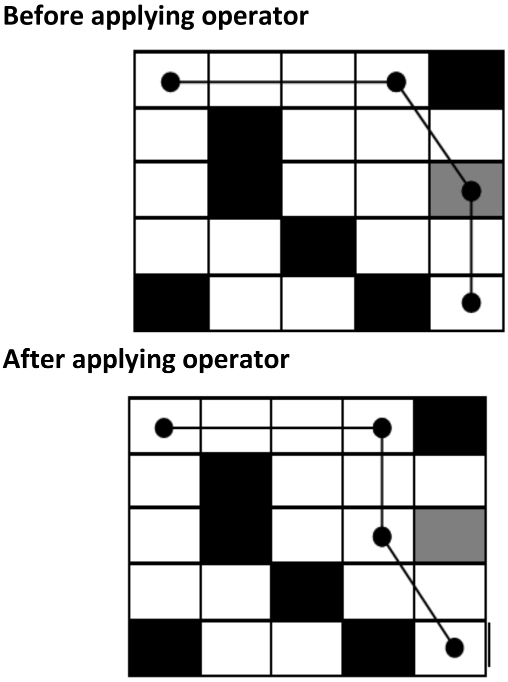
**Fig.9 Mutation operator**

**6.3 Node-repair** is used to move a node falling on an obstacle out of the obstacle and to a best grid around the obstacle. To locate the best grid, a small-scaled local search in the neighbors of the obstacle is applied. The 8 nearest neighbor of the grid is considered. There are three cases in searching for nearest neighbors:

- **Case 1:** if the node under consideration is a centre node, it will have 8 neighbors.
- **Case 2:** if the node under consideration is the corner node, it will have 3 neighbors.

- **Case 3:** if the node under consideration is the boundary node, it will have 5 neighbors.

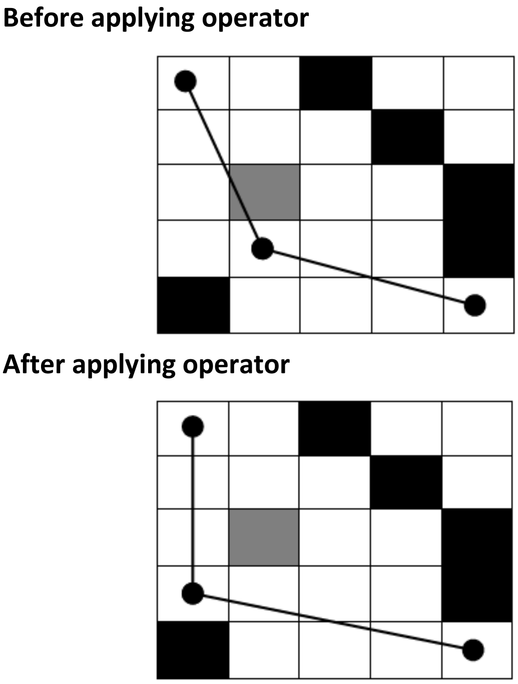
Node repair operator is used to improve the path.


Before applying operator

After applying operator

**Fig.10 Node-Repair Operator**

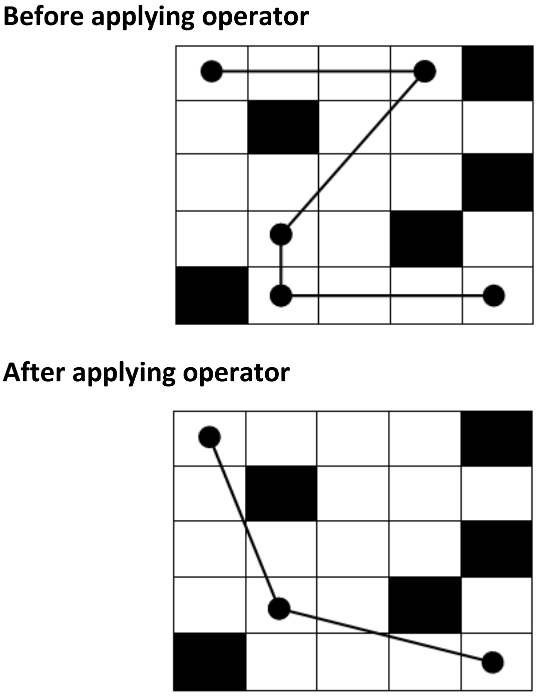
**6.4 Line-repair** is used to repair an infeasible line segment by inserting a suitable node between the two nodes of the segment. Again, to locate a best node, similar local search is applied in the all of neighboring grids of the intersected obstacle.


Before applying operator

After applying operator

**Fig.11 line repair operator**

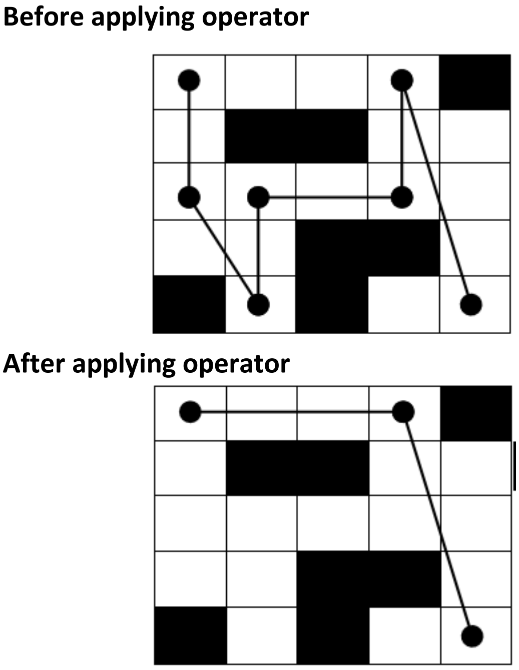
**6.5 Deletion** is applied to both feasible and infeasible path. Randomly choose one node, check its two adjacent nodes, and connected segments, if the deletion of the chosen node is beneficial (turn the infeasible to the feasible, reduce the cost), delete the node.


Before applying operator

After applying operator

**Fig.12 Delete operator**

**6.6 Improvement** is designed for feasible solutions. Randomly chose one node, do a local search in the neighboring grids of the node, move to a best grid.


Before applying operator

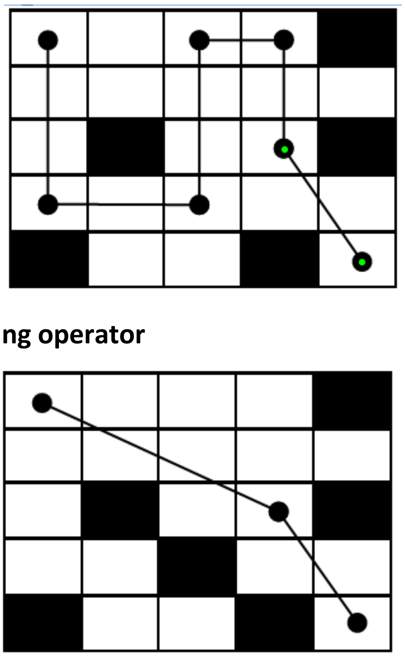
After applying operator

**Fig.13 Improvement Operator**

'This operator is used for fine tuning of a feasible solution. The fining of these operators depends on two criteria: probability and heuristic knowledge (e.g., if feasible then improve).

**6.7 Optimize** is designed to obtain more feasible solutions. This operator is based on the triangle inequality. It takes three nodes and tries to find if the sum of distance of two nodes is greater than third one. If yes, then the middle node is deleted. A more enhanced path is obtained.

**Before applying operator**

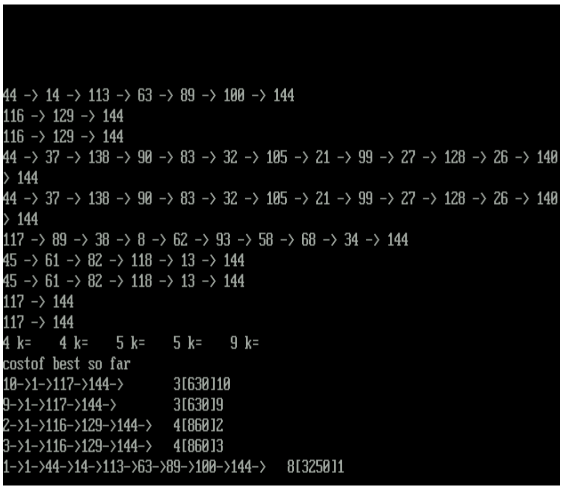


**After applying operator**

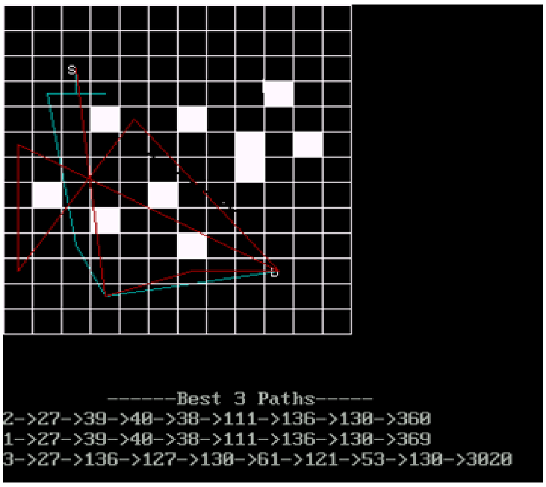
Fig.14 Optimize operator

## 6. RESULTS

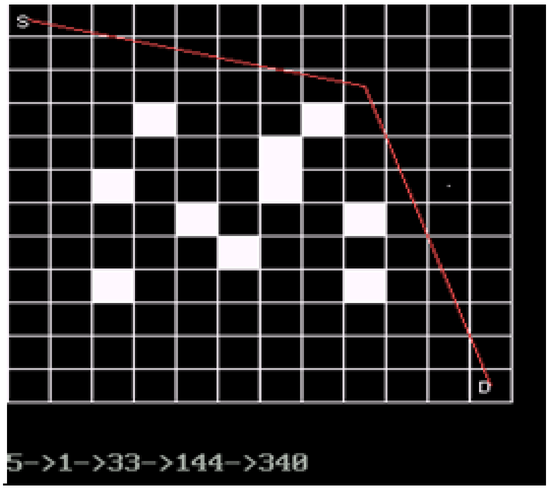
Some results are shown in the following figures:



```
44 -> 14 -> 113 -> 63 -> 89 -> 100 -> 144
116 -> 129 -> 144
116 -> 129 -> 144
44 -> 37 -> 138 -> 90 -> 83 -> 32 -> 105 -> 21 -> 99 -> 27 -> 128 -> 26 -> 140
> 144
44 -> 37 -> 138 -> 90 -> 83 -> 32 -> 105 -> 21 -> 99 -> 27 -> 128 -> 26 -> 140
> 144
117 -> 89 -> 38 -> 8 -> 62 -> 93 -> 58 -> 68 -> 34 -> 144
45 -> 61 -> 82 -> 118 -> 13 -> 144
45 -> 61 -> 82 -> 118 -> 13 -> 144
117 -> 144
117 -> 144
4 k=   4 k=   5 k=   5 k=   9 k=
costof best so far
10->1->117->144->       3[630]10
9->1->117->144->        3[630]9
2->1->116->129->144->   4[860]2
3->1->116->129->144->   4[860]3
1->1->44->14->113->63->89->100->144->   8[3250]1
```



```
-------Best 3 Paths------
2->27->39->40->38->111->136->130->360
1->27->39->40->38->111->136->130->369
3->27->136->127->130->61->121->53->130->3020
```



```
5->1->33->144->340
```

## 7. CONCLUSION

In this work a genetic algorithm with variable chromosome length is developed for a mobile robot to obtain a shortest path in 2-D environment with ‘m’ static obstacles and ‘n’ dynamic obstacles which can move to and fro on the given area. The developed algorithm uses an efficient coding scheme with variable length integer coded chromosome that act as the natural selection of genetic algorithms, then the specialized operators in coordination with the basic operators of genetic algorithm are used to find a collision free path.

The research work uses a simple robot path representation that combines grids to show the environment representations. The genetic algorithm incorporates the domain knowledge into its problem specific genetic operators. The developed GA also features its efficient evaluation method i.e. an effective fitness function which is greatly beneficial for evolving good solutions from infeasible solutions. The effectiveness of the use of knowledge based operators is demonstrated by simulation studies. The simulation results also show that the proposed genetic algorithm is effective in both complex static and dynamic environments.

## 8. REFERENCES

[1] Rastogi Shivanshu, Kumar Vikas, “ path planning of mobile robot in static environment” MITIJCSIT,2012

[2] T. Lozano-Perez, “Spatial planning: A configuration approach,” IEEE Trans. on Computers, vol. C-32, no. 2, pp. 108-120, Feb. 1983

[3] M.W. Spong and M. Vidyasagar, Robot dynamics and control. New York: Wiley, 1989.

[4] Ananya Das, Priyadarsini ohapatra “ Improved real time A* algorithm for Path Planning of Mobile Robot in Quadrant Based Environment”

[5] Torvald Ersson and Xiaoming Hu, “Path Planning and Navigation of Mobile Robots in Unknown Environments”

[6] Ismail AL-Taharwa, Alaa Sheta and Mohammed Al-Weshah,” A Mobile Robot Path Planning Using Genetic Algorithm in Static Environment

[7] Zhang Qi-yi, Chang Shu-chun,” An Improved Crossover Operator of Genetic Algorithm, “Second International Symposium on Computational Intelligence and Design, 2009. ISCID '09

[8] Yuan-Qing Qin De-Bao Sun ; Ning Li ; Yi-Gang Cen,” Path planning for mobile robot using the particle swarm optimization with mutation operator Proceedings of 2004 International Conference on Machine Learning and Cybernetics.