Dynamic Programming Agent for Mobile Robot Navigation with Moving Obstacles

D .Tamilselvi
Sr.Gd. Lecturer
Department of Computer Science and Engineering
Thiagarajar College of Engineering
Madurai 625 015, Tamilnadu, India
dtamilselvi@tce.edu

P.Rajalakshmi
M.E. II Year
Department of Computer Science and Engineering
Thiagarajar College of Engineering
Madurai 625 015, Tamilnadu, India
rajime@tce.edu

S.Mercy Shalinie
Asst.Prof. & Head
Department of Computer Science and Engineering
Thiagarajar College of Engineering,
Madurai 625 015, Tamilnadu, India
shalinie@tce.edu

Abstract - Dynamic programming (DP) approach to shortest path algorithm provides a global optimal solution to robot path planning in an indoor environment. The proposed DP shortest path algorithm for real-time provides collision-free robot-path planning in which the barriers is permitted to move in an indoor environment. Environment is represented as a grid map, each grid point records the distance to the target. The information stored at each point is a current estimate of the distance to the nearest target and the neighbor from which this distance was determined. Updating the distance estimate at each grid point is done using the information gathered from the point's neighbors. Dynamic Programming Agent requires the distance information in the neighbor grid points and no prior knowledge about the obstacle position is required. At each time step collision free shortest path is generated by propagating the distance to the target and updating them in an order depending on the distance to the target for intelligent navigation

Key words: Dynamic programming Agent, Path planning, Mobile Robot, path planning, dynamic obstacle.

I. INTRODUCTION

In recent years a new approach to Artificial Intelligence has developed which are based on building behavior-based programs to control situated and embodied robots in unstructured dynamically changing environments [1]. Rather than modularize perception, world modeling, planning, and execution, the new approach builds intelligent control systems where many individual modules directly generate some part of the behavior of the robot.

Intelligence is added to the mobile robot by agents who perceive environment, capable of taking actions based on its own and other agents exist in the environment. Agents need to coordinate to achieve their individual goals and common goals. Agent systems are considered to be intelligent because they posses a high degree of autonomy and can make decisions by perceiving the environment in which they reside. Multiagent systems are developed to address complex systems [2]where a human being cannot predict beforehand its behavior at a particular instance. This leads to the desirability of coordination and agent autonomy in multiagent systems and their adaptability to the changing environment. Topological, metric or hybrid navigation schemes can make use of different types of environment features [9] on various levels of perceptual abstraction leading to different environment models.

The AI Community uses the control of the mobile robot with three functional units.[3] The sensing ,planning and execution system. The intelligence of the system is performed in the planner of the Program, not the execution mechanism. Research efforts through 1985 focused almost exclusively on planning and world modeling.

Mobile Robot Builders ([Nilsson 84],[Moravec 83],[Giralt et al 83] have explored the path planning into subtask of Sensing, Mapping Sensor data into a world representation, planning,[4] task execution and motor control. Theses subtask form a chain through which information flows from the robot's environment, via sensing, through the robot and back to the environment, via action and closing the task while it reaches the target. Sensing, Planning and Execution System. One of the ultimate goals of indoor mobile robotics research is to build robots that can safely carry out missions in hazardous and populated environments which can be achieved by implementing the algorithms to overcome various navigation issues. In this paper moving obstacle problems are addressed and methodology to overcome the Dynamic Programming (DP) algorithm proposed by Allan R. Willms and Simon X. Yang[1] is implemented. The simulation results shows the shortest path algorithm for real-time collision- free robot-path planning approach in which the barriers are permitted to move in an indoor environment

II. PATH PLANNING

There exists a large number of methods for solving the basic navigation issue. However, not all of them solve the problem in its full generality. For instance, some methods require the workspace to be two dimensional and the objects to be polygonal, and some would need the workspace to be always static.

These methods are [6] briefly introduced in this section. The roadmap approach to path planning represents the free-space for a robot as a collection of connected collision-free paths. This set of collision free paths, called roadmap, is used to plan a path as follows. A path is constructed from the start configuration to some part of the roadmap. Similarly, a path is constructed from the goal configuration to the roadmap. Next, using standard graph algorithms, the roadmap is searched for a path between the two points of connection on the roadmap. For a static environment, the roadmap is constructed once, and can be used to solve multiple planning problems. The many variations of the roadmap approach differ mainly in the method for constructing the roadmap. These variations include: visibility graphs and Voronoi diagrams Cell decomposition techniques divide the area of the free space in two-dimensional configuration spaces into disjoint units of simple shape, called cells. A path can be generated by searching the connectivity graph describing the adjacency of the cells in the cell decomposition methods. The shape of the cells is chosen in such a manner that motion generation within a cell is simple. A path then consists of a sequence of cells and points at which the transition from one cell to another occurs. Planners based on these techniques tend to be more of theoretical interest as they are complex to implement and are inefficient

A. Obstalce Avoidance

Recently, many researches turned their attention to obstacle avoidance problem developing interesting real-time approaches for narrow and cluttered spaces. However, there are some classic obstacle avoidance methods [5] that must be cited (Borenstein and Koren, 1991): *edge-detection*, *certainty grids*, and *potential field methods*. The first one, edgedetection, is a very popular method that extracts the obstacle vertical edges and drives the robot around either one of the visible edges. This approach was early commonly combined with ultrasonic sensors. Due to the limited accuracy of the sensor, the approach presented some shortcomings: poor directionality, frequent misreading, and specular reflections.

On the other hand, Moravec and Elfes (1985) pioneered the concept of certainty grid, a map representation that is well suited for sensor data accumulation and fusion. Certainty grid is an obstacle probabilistic representation method that uses a grid-type world model. The robot's work area is modeled as a 2-D array of square elements, called cells. Each cell has a certainty value (*CV*) that indicates the measure of confidence that an obstacle is within the cell area. The *CV* is a probability function that depends on the sensor characteristics. As each cell has its *CV* updated constantly by the sensor readings, after a period moving across an area, the robot has a fairly accurate map of that area. The method accuracy is a function of the cell size and may be considered as its drawback as well. The third method, potential field method is based on the idea that obstacles exert imaginary repulsive forces, while the goal position applies an imaginary attractive force to the robot. The resultant robot behavior is obtained summing all attractive and repulsive forces

B Moving Obstacle

In an effort to solve the problem of motion planning [10] in a dynamic environment, Conn and Kam [11] included time as one of the dimensions and thus the moving obstacles can be regarded as stationary in the extended world. The major problem in this approach is that the trajectories of the moving obstacles are assumed known a priori, which are often inapplicable in real applications. Then, Vadakkepat *et al*. [12] proposed a new methodology named Evolutionary Artificial Potential Field (EAPF) to solve moving-obstacle problem.

III. DP ALGORITHM

The DP path planning algorithm is for mobile robot navigation is suitable for both static and dynamic environments. This algorithm works in real time and requires no prior knowledge of the barrier locations. Robot environment is discretized into a grid of M points, [7] labeled by an index i, each point being either a free space or a barrier location, the targets and the robot may occupy any free space. We define $d_{\min}$ and $d_{\max}$ to be the minimum and maximum distances between any two adjacent neighbors in the grid, and we define B_i to be the set of free spaces that are adjacent neighbors to point i. The distance d_{ij} between any two free spaces i and j is defined to be the minimum Euclidean length of all paths joining i and j through non barrier adjacent neighbors.

IV. ENHANCED DYNAMIC PATH PLANNING AGENT

Environment is discretized and represented by a topologically organized map. Each grid point has only local connections to its neighboring grid points. Unlike the neural network based path planning model proposed by Yang and Meng [8], the proposed robot path algorithm efficiently propagates the distance instead of the neural activity from the target to the entire robot workspace. Unlike Zelinsky's distance model, where the transformed distance at each grid point is a function of target and obstacle locations in the entire workspace making it suitable for static environments only, our algorithm is capable of dealing with changing environments.

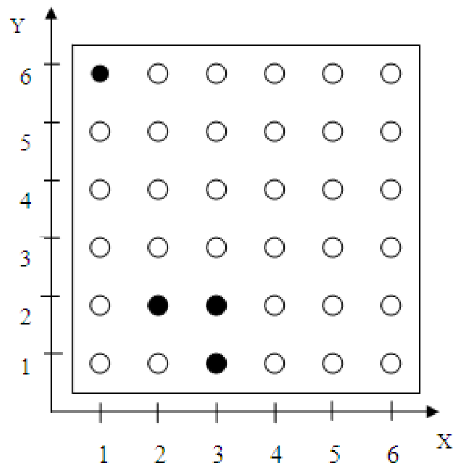


Figure 1. Grid Environment

Fig.1 shows an example environment discretized in to 6×6 grid map. Robot environment is discretized into a grid of M points as shown in figure 1, labeled by an index i, each point being a free space or a barrier location, the targets and the robot may occupy any free space. We define $d_{\min}$ and $d_{\max}$ to be the minimum and maximum distances between any two adjacent neighbors in the grid, and we define B_i to be the set of free spaces that are adjacent neighbors to point i. The distance d_{ij} between any two free spaces i and j is defined to be the minimum Euclidean length of all paths joining i and j through non barrier adjacent neighbors.

A Distance Propagation

Each grid point has an associated variable $x_i(n)$, a real value that records the distance to the nearest target at time step n. We define $d_{\min}$ and $d_{\max}$ to be the minimum and maximum distances between any two adjacent neighbors in the grid, and we define B_i to be the set of free spaces that are adjacent neighbors to point i. The distance d_{ij} between any two free spaces i and j is defined to be the minimum Euclidean length of all paths joining i and j through non barrier adjacent neighbors. The system is initialized by setting the variables $x_i(0)$ to 0 for all target locations and all other locations to some large maximal distance D. $D > (M - 1)d_{\max}$ is sufficiently large. The dynamic system then evolves by updating the grid points in an order depending on the distance to the target.

$$x_i(n+1) = \begin{cases} 0, & \text{if } i = \text{target} \\ D, & \text{if } i = \text{barrier} \\ \min(D, f(i,n)), & \text{otherwise} \end{cases} \tag{1}$$

where,

$$f(i,n) = \min_{j \in B_i}(d_{ij} + x_j(n))$$

using Eq.(1) each neighbor grid point is updated for the current distance information. After one time step, the target locations will be zero and their adjacent neighbors will record the correct distance; after two time steps, the adjacent neighbors of the target neighbors will record the correct distance, and so on, so that the distance to each target location will spread outward by one grid step each time step. In other words, the distance variable x_i can temporarily reflect the distance to the target that is the fewest grid steps away. Once the number of time steps is greater than the number of grid steps from a point to each of the targets along the minimal distance paths, the value of x for that point will not change and is guaranteed to be the exact distance to the target.

IV. B PATH SELECTION

We assume that the robot can move from any grid point to any neighboring grid point, that is, point robot dynamics. The robot location $r(t)$ is specified as an index of one of the points on the grid, and is a function of real time $t \geq t_0$. Initially, $r(t_0) = i_0$. Robot's travel path is updated at a set of real time values $t_1 < t_2 < t_3 < \ldots$ At time t_k, the next update time, t_{k+1}, and next update location, $r(t_{k+1})$, are determined. $r(t_{k+1})$ can be defined as,

$$r(t_{k+1}) = Ind(r(t_k), n(r(t_k), t_k)) \tag{2}$$

where $n(i, t_k)$ means the highest time step n for which x_i has been computed up to time t_k, and Ind(i, n) is the index of the closest neighbor through which the value of $x_i(n)$ was calculated.

Let

$$B_i^*(n) = \{j \in B_i \mid f(i, n-1) = (d_{ij} + x_j(n-1))\}$$

then

$$Ind(i,n) = \begin{cases} i, & \text{if } x_i(n) \in \{0, D\} \\ j \in B_i^*(n) \mid d_{ij} = \min\limits_{k \in B_i^*(n)} d_{ik}, & \text{otherwise} \end{cases}$$

If barriers are moving, then the robot location can be easily updated by simply keeping track of Ind(i, n) for the robot location i while updating x_i.

The robot location is updated at times t_k according to Eq. (2). Let k^* be the smallest integer such that we have $r(t_{k^*+1}) \neq r(t_{k^*})$. In other words, t_{k^*} is the first time at which the robot starts to move. Update times prior to this occurred when the distance information had not yet propagated to the robot's location, i.e., $x_{r(t_k)} = D$ for $k < k^*$. The robot does not move until $x_{r(t_k)} < D$ and, by definition of the robot update function Ind(i, n), it always moves to a location which must also have once been less than D. once x is below D it remains below D forever, it follows that the robot will continue to move until $x_{r(t_k)} = 0$.

V. SIMULATION RESULTS

In this section simulation results are presented and several implementation issues are discussed. Figuers show simulation runs in several static and dynamic environments demonstrating the inclination of the Ordered distance propagation approach to maximize the distance from the obstacles as the robot is driven along the path. The proposed navigation system has been implemented using java. To use the distance propagation approach to establish the dynamic path planning model, the following assumptions need to be made,

- Point Robot dynamics-The robot can move from any grid point to any neighboring grid point.
- Obstacle movements are unknown.
- Since obstacles can move, collisions with robots is possible. To avoid collision with robot the simulations are designed with the assumption that in general Robot has the higher speed than obstacles in the environment.

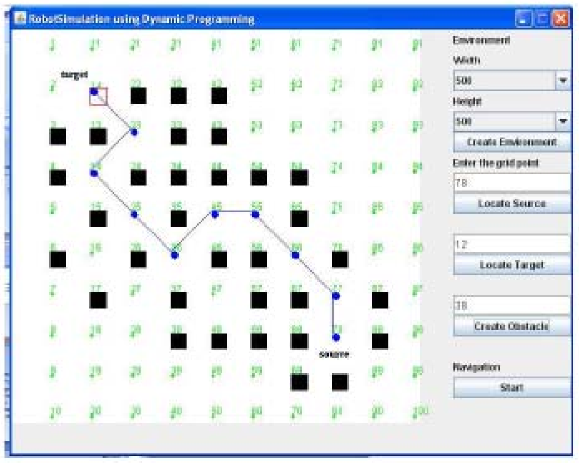
In more complicated situations we prevented collisions between barriers and robots by specifying that if at time n a barrier appears at location pi to which a robot is currently moving,

- pi = r(tk), tk−1 < n < tk, then the robot returns from where it came
- (r(tk) is set to r(tk−1)).

All the simulation trials are carried out on the discretized grid map Environment. Table describes the notations used in the simulation. Computational efficiency is calculated by retrieving the system time using java.

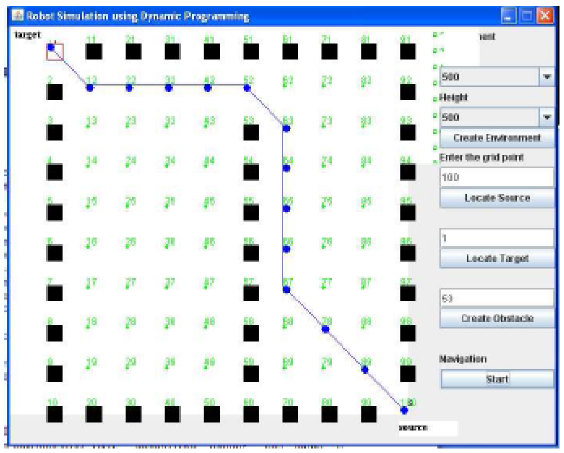
Table 1

Notation	Description
Blue Oval	Robot
Blue, Red unfilled rectangles	Target
Blue Objects	Obstacles
Blue Line	Robot Movement
Red Line	Obstacle Movement
Greed unfilled ovals	Unvisited free locations
Block unfilled ovals	Location occupied by the obstacles in the previous time step but free during the current time step



Environment
Width 500
Height 500
Create Environment
Enter the grid point
78
Locate Source
12
Locate Target
38
Create Obstacle
Navigation
Start

Figure 5.1: Environment with complex set of static obstacles



Height: 500 / 500

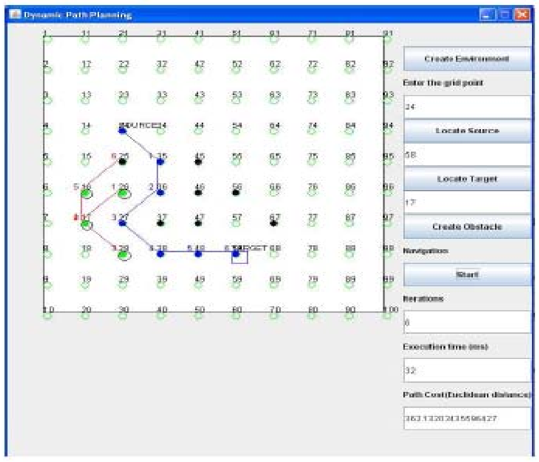
Enter the grid point: 100

Locate Target: 1

Create Obstacle: 53

Navigation

Figure 5.2: Environment with static walls



Enter the grid point: 24

Locate Target: 58

Create Obstacle: 17

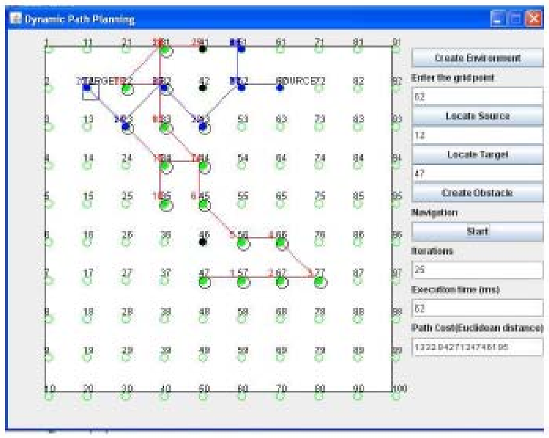
Navigation

Iterations: 6

Execution time (ms): 32

Path Cost(Euclidean distance): 363.13202135596427

5.3 Environment with Dynamic Obstacles



Enter the grid point: 62

Locate Target: 12

Create Obstacle: 47

Navigation

Iterations: 25

Execution time (ms): 62

Path Cost(Euclidean distance): 1322.0427134746195

5.4 Environment with Dynamic Obstacles

VI. CONCLUSION

The enhanced DP algorithm simulation results show the effective path selection along with time steps taken for the mobile robot to reach the target in an indoor environment with the obstacles moving. The performance is compared with other dynamic approaches Dynamic Programming and Distance Propagation. The computation time is less for reaching the target in the ordered distance propagation method. This simulation work will be extended to real time robots.

REFERENCES

[1] Rodney A. Brooks, " Artificial Life and Real Robots**"** , *MIT Artificial Intelligence Laboratory*

[2] P. Gaudiano, E. Zalama, and J. L. Coronado, "An unsupervised neural network for low- level control of a mobile robot: Noise resistance, stability, and hardware implementation," *IEEE Trans. Syst., Man, Cybern. B,Cybern., vol. 26, no. 3, pp. 485–496, Jun. 1996.*

[3] Erann Gat, "On Three Layer Architectures " , Jet Propulsion Laboratory, California Institute of Technology, *AAAI Press, Artificial Intelligence and Mobile Robots*

[4] Rodney A.Brooks, "A Robust Layered Control System for a Mobile Robot", Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1985

[5] Marcelo Becker, Carolina Meirelles ,Weber Perdigao Macedo," Obstacle Avoidance Procedure for Mobile Robots" *ABCM Symposium Series in Mechatronics, Vol.2-pp 250- 257*

[6] Lim Chee Wang, Lim Ser Yong, Marcelo H Ang Jr, " Hybrid of Global Path Planning and Local Navigation Implemented on a Mobile Robot in Indoor Environment" *National University of Singapore*

[7] Allan R. Willms, Simon X.Yang, " An Efficient Dynamic System for Real time Robot Path Planning" IEEE Transactions of System, Man and Cybernetics – Part B: Cybernetics, Vol.36, No.4, August 2006 Page No.755 – 765

[8] S.X.Yang, M.Q.Meng, "Real-time collision-free motion planning of mobile robots using neural dynamics based approaches," IEEE Trans.Neural Netw., vol. 14, no. 6, pp. 1541–1552, Nov. 2003

[9] Nicola Tomatis,"A Hybrid Approach for Robust and Precise Mobile Robot Navigation with Compact Environment Modeling" *Proceedings IEEE International Conference on Robotics and Automation,2001*

[10] Guan-Chun Luh, " Dynamic Mobile Robot Navigation using Potential field based immune network" *Systems, Cybernetics and Informatics,2003*

[11] R.A. Conn, and M. Kam, "Robot motion planning on N dimensional star worlds among moving obstacles", *IEEE Transactions on Robotics and Automation, Vol. 14, No. 2, 1998, pp. 320-325*