Real Time Target Evaluation Search

Cagatay Undeger
Savunma Teknolojileri Muhendislik ve Ticaret A.S.
Mecnun Sk. No: 58 Bestepe
06510 Ankara, Turkey

cundeger@ceng.metu.edu.tr

Faruk Polat
Department of Computer Engineering
Middle East Technical University
06531 Ankara, Turkey

polat@ceng.metu.edu.tr

ABSTRACT
In this paper we propose a real-time search algorithm called Real-Time Target Evaluation Search (RTTES) for the problem of searching a route in grid worlds from a starting point to a static or dynamic target point in real-time. The algorithm makes use of a new effective heuristic method which utilizes environmental information to successfully find solution paths to the target in dynamic and partially observable environments. The method requires analysis of obstacles to determine closed directions and estimate the goal relevance of open directions in order to identify the most beneficial move. The environment is assumed to be a planar grid and the agent has limited perception. In this paper, we compared RTTES with Real-Time A* (RTA*) and Real-Time Edge Follow (RTEF), and observed a significant improvement.

Categories and Subject Descriptors
I.2 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

General Terms
Algorithms, Performance, Experimentation

Keywords
Path planning, Grid world, Real-time search, Heuristic search

1. INTRODUCTION
Path planning can be described as finding a path from an initial point to a target point if there exists one. Path planning algorithms are either off-line or on-line. Off-line algorithms find the whole solution in advance before starting. These algorithms suffer from execution time in dynamic or partially observable environments due to frequent re-planning requirements. In on-line case, an agent repeatedly plans its next move in limited time and executes it, but they are not designed to be optimal and usually find poor solutions with respect to path length. Furthermore, there exist some hybrid solutions such as incremental heuristic search algorithms that are more efficient than off-line path planning. However, they are still slow for some real-time applications and are not applicable to moving targets. There are some on-line algorithms trying to overcome these difficulties. For instance, Real-Time Edge Follow (RTEF) [16, 17] uses a powerful heuristic function that can discard some non-promising alternative moving directions in real-time to guide the agent to a static or dynamic target. Although RTEF is able to determine the closed (non-promising) directions successfully, it is weak in selecting the right move from the rest of the alternatives since it uses poor Euclidian distance heuristic. Therefore, we focused on a new method for better selection.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
AAMAS'06, May 8–12, 2006, Hakodate, Hokkaido, Japan.
Copyright 2006 ACM 1-59593-303-4/06/0005…$5.00.

In this paper, we propose a real-time search algorithm, Real-Time Target Evaluation Search (RTTES), and a heuristic method, Real-Time Target Evaluation (RTTE) capable of estimating distance to the target considering the obstacles. The method sends rays away from the agent in four directions, and determines obstacles that the rays hit. For each such obstacle, we extract its border and determine best direction that avoids the obstacle. Finally, by using a resolution mechanism, one of the proposed directions is chosen.

In the next section, the related work is given. In section 3, RTTES is described in brief. The performance analysis is presented in section 4. And finally, the conclusion is given in section 5.

2. RELATED WORK
Optimal [11, 14] and probabilistic [2, 4, 7, 9, 10] off-line path planning algorithms are hard to use for large dynamic environments because of their time requirements. One solution is to make off-line algorithms to be incremental [5, 6, 12, 13] to avoid re-planning from scratch. Although incremental algorithms are efficient in most cases, sometimes a small change in the environment may cause to re-plan almost from scratch. Due to the efficiency problems of off-line techniques, a number of on-line approaches such as Learning Real-Time A* (LRTA*), Real-Time A* (RTA*) [8], Real-Time Horizontal A* (RTHA*) [15], Execution Extended Rapidly Exploring Random Trees [1], and PRM with Kinodynamic Motion Planner [3] are proposed.

Recently, a new on-line path search algorithm (Real-Time Edge Follow - RTEF) [16, 17] is proposed for grid-type environments. RTEF uses a new heuristic (RTEF-ARM) that effectively makes use of global environmental information. With this heuristic, the agent can detect closed directions (directions that cannot reach the target) using perceptual data and tentative map, and determine its next move from open directions. Experimental study shows that RTEF performs better than RTA* and RTA* with n-look-ahead depth in terms of solution path and execution time [17].

3. RTTES & RTTE

We assume that the environment is a planar grid and is partially known by the agent. The agent located at a particular cell is required to reach a target cell avoiding obstacles in real-time. At any step, agent can move north, south, east or west, and is able to maintain a tentative map in its memory while exploration.

RTTES uses the heuristic method RTTE, which analyzes obstacles and proposes moving directions to reach the target through approximately short paths avoiding the obstacles. For this, RTTE geometrically analyzes the obstacles nearby and tries to estimate lengths of paths around obstacles to reach the target. To avoid infinite loops and re-visiting the same locations redundantly, RTTES uses two mechanisms (*visited count grid* & *history*). *Visited count grid* is a 2D matrix storing the number of visits to the cells, and *history* is the set of previously visited marked cells. History cells are assumed to be obstacle. When exploring an unknown environment, the agent may realize that all the ways leading to the target are blocked. In such a case the history should be cleared in order to let the agent backtrack as history cells may cause this. RTTES repeats the following steps until reaching the target or detecting that the target is inaccessible:

1) Use RTTE to compute proposed direction and utilities of neighbor cells
2) If a direction is proposed by RTTE:
   a) Select the neighbor cell with the highest utility from the set of non-obstacle neighbors with the smallest visited count
   b) Move to the selected direction, increment the visited count of the previous cell by one, and insert the cell into the *History*
3) Else if *History* is not empty, clear all the *History*
4) Else destination is unreachable, stop the search with failure.

The RTTE algorithm works as follows:

1) Mark all the moving directions as open
2) Propagate four rays away from the agent to north, south, east and west
2) For each ray hitting an obstacle:
   a) Trace and extract the border of the obstacle
   b) Analyze the border by re-tracing it from left and right sides
   c) Detect closed directions
   d) Evaluate results and determine a direction to avoid obstacle
3) Merge individual results, propose a direction to move, and compute utilities of neighbor cells

As seen above, RTTE propagates four rays from the agent location to north, south, east and west directions, and analyzes the obstacles these rays hit to find out the best direction to move. If a ray hits an obstacle before exceeding the maximum ray distance, the obstacle border is extracted by tracing cells on the border starting from the hit-point. Next the border is re-traced from both left and right sides to determine the geometric features of the obstacle. The closed moving directions are determined similar to RTEF [17]. The obstacle features are evaluated and a moving direction to avoid the obstacle is identified. After all the obstacles are evaluated, the results are merged in order to propose a final moving direction.

4. PEFORMANCE ANALYSIS

In this paper, we compared RTTES with RTA* and RTEF on randomly generated sample grids. We used RTTES and RTEF with two variations: "VC" that only uses visited counts and "VCH" that uses both history and visited counts. Thus, we tested five algorithms: RTTES-VC, RTTES-VCH, RTEF-VC, RTEF-VCH and RTA*. We used grids of three different types: *random*, *maze* and *U-type* with size 200x200. *Random* grids are generated randomly based on different obstacle ratios. *Maze* grids are the ones where every two non-obstacle cells are always connected through a path (usually one path). *U-type* grids are created by randomly putting U-shaped obstacles. We assume that the agent has limited vision (called *visual depth v)* and perceives only the grid cells within the square region *(2v+1)*(2v+1)* centered at agent's location. We take the visual depth as 20 or *infinite* (*full vision*) which means the agent knows the entire grid world in advance.

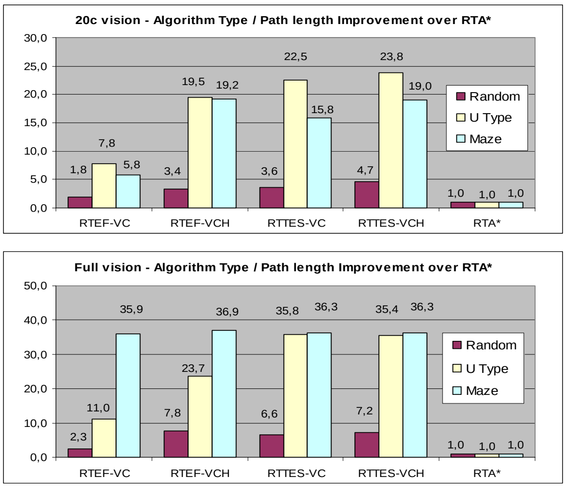

Figure 1. Path Length comparison of 20 cell and full vision

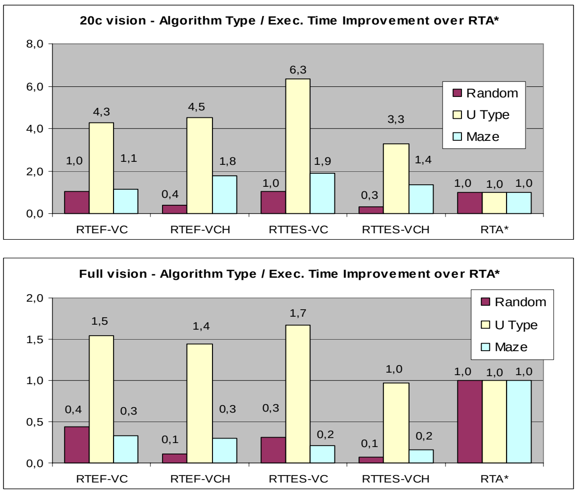

Figure 2. Execution Time comparison of 20 cell and full vision

Figure 1 contains test results that show improvements over RTA* in path length (i.e., path length of RTA* divided by these of compared algorithms) for different grid types. We observed that RTEF and RTTES performed significantly better than RTA* all the time. In random grids, RTEF-VCH and RTTES variations were almost head to head. In maze grids, all the RTEF and RTTES variations performed almost the same with full vision. With 20 cell vision, the results of RTEF-VCH and RTTES-VCH were very close to each other, and they performed better than RTEF-VC and RTTES-VC on the average. In U-type grids, the results of RTTES variations were almost the same, and they performed better than RTEF variations on the average.

Figure 2 contains comparison of total execution times of previous runs similarly. We report ratios in terms of execution times taking RTA* as basis. In random grids with 20 cell vision RTEF-VC, RTTES-VC and RTA* performed almost the same, and were better than RTEF-VCH and RTTES-VCH. In random grids with full vision, RTA* is better since the efficiency of RTEF and RTTES usually drop in fully known grids. In maze grids, RTEF-VCH and RTTES-VC performed similarly, and were better than the others with 20 cell vision. With full vision, RTA* is better. In U-type grids, RTEF and RTTES usually performed much better than RTA*. The results of RTTES-VC were the best among all.

We also compared the path length of the algorithms with the optimal solutions generated by A* in fully known grids. The ratios of the algorithms' solution path lengths over optimal paths lengths are presented in Tables 1-2. The results show that solutions of both RTTES variations were only 1.14 times longer than the optimal ones on the average, whereas the solutions of RTEF-VC, RTEF-VCH and RTA* were 4.39, 1.5 and 33 times longer respectively. The standard deviations of RTTES algorithms were significantly less than RTEF algorithms. And we see that best improvement was obtained in U-type grids.

Table 1. Average ratios of algorithms' path lengths over optimal path lengths

	RT-VC	RT-VCH	RTTES-VC	RTTES-VCH	RTA*
Average	4,394	1,501	1,142	1,140	33,022
Standart Deviation	8,555	1,068	0,194	0,169	50,417

Table 2. Average ratios of algorithms' path lengths over optimal path lengths in random, U-type, and maze grids

	RT-VC	RT-VCH	RTTES-VC	RTTES-VCH	RTA*
Random Grids	3,577	1,432	1,369	1,339	10,532
Maze Grids	1,967	1,346	1,078	1,081	37,800
U-Type Grids	10,467	1,903	1,113	1,123	39,141

5. CONCLUSION

In this paper, we have focused on real-time search for grid-type problems, and presented an effective heuristic method (RTTE) and a real-time search algorithm (RTTES).

We have compared RTA*, RTEF and RTTES with the help of 1600 test runs. With respect to the path length, experimental results showed that RTTES is able to make use of environmental information very successfully to improve the solutions. In U-type grids, RTTES discovers much shorter paths compared to RTEF and RTA*. In random and maze grids, RTTES and RTEF are almost head to head, and both are much better than RTA*. With respect to execution time, we have observed that RTTES-VC is highly efficient in U-type grids. In random and maze grids, RTEF and RTTES almost perform the same, but it is hard to say about RTA* since the results change much depending on the vision range. With limited vision, the agent knows less about its environment, which increases the efficiency of RTEF and RTTES, and makes them perform usually better than RTA*. The performance decrease in case of unlimited vision is due to the fact that the agent knows more than it requires. We have also seen that RTTES converges to almost optimal solutions in fully known grids with respect to path length.

6. REFERENCES

[1] Bruce, J. and Veloso, M. *Real-time randomized path planning for robot navigation*. In Proceedings of IEEE Int'l Conf. on Intelligent Robots and Systems, pp. 2383-2388, Piscataway, N.J.: IEEE, 2002.

[2] Cheng, P. and LaValle, S. M. *Resolution complete rapidly-exploring random trees*. In Proceedings of IEEE Int'l Conf. on Robotics and Automation, pp. 267-272, 2002.

[3] Hsu, D., Kindel, R., Latombe, J.C. and Rock, S. *Randomized Kinodynamic Motion Planning with Moving Obstacles*. Int. J. Robotics Research, 21(3):233–255, 2002.

[4] Kavraki, L. and Latombe, J.C. *Probabilistic Roadmaps for Robot Path Planning*. In Practical Motion Planning in Robotics: Current and Future Directions, K. Gupta & A. del Pobil Eds., Addison-Wesley, 1998.

[5] Koenig, S. and Likhachev, M. *D* Lite*. In Proceedings of the National Conference on Artificial Intelligence, 2002.

[6] Koenig, S. and Likhachev, M. *Improved Fast Replanning For Robot Navigation in Unknown Terrain*. In Proceedings of the Int'l Conf. on Robotics and Automation, 2002.

[7] Konar, A. *Artificial Intelligence and Soft Computing: Behavioral and Cognitive Modeling of Human Brain*, CRC Press LLC, 2000.

[8] Korf, R.E. *Real-Time Heuristic Search*. Artificial Intelligence, Vol. 42, No: 2-3, pp. 189-211. 1990.

[9] LaValle, S. M. and Kuffner, J. J. *Rapidly-exploring random trees: Progress and prospects*. Algorithmic and Computational Robotics: New Directions, pp. 293-308. A K Peters, Wellesley, MA, 2001.

[10] Michalewicz, Z. *Genetic Algorithms + Data Structure = Evolution Programs*, Springer-Verlag, New York, 1986.

[11] Russell, S. and Norving, P. *Artificial Intelligence: a modern approach*. Prentice Hall, Inc. 1995.

[12] Stentz, A. *Optimal and Efficient Path Planning for Partially-Known Environments*. In Proceedings of IEEE Int'l Conf. on Robotics and Automation, May, 1994.

[13] Stentz, A. *The Focussed D* Algorithm for Real-Time Replanning*. In Proceedings of Int'l Joint Conf. on Artificial Intelligence, August, 1995.

[14] Tanenbaum, A.S. *Computer Networks*, Prentice-Hall, Inc., pp 349-351, 1996.

[15] Undeger, C. *Real-Time Mission Planning For Virtual Human Agents*. M.S. Thesis in Computer Engineering Department of Middle East Technical University. Jan 2001.

[16] Undeger, C., Polat, F. and Ipekkan, Z. *Real-Time Edge Follow: A New Paradigm to Real-Time Path Search*. In Proceedings of GAME-ON 2001, London, England, 2001.

[17] Undeger, C. and Polat, F. *Real-Time Edge Follow: A Real-Time Path Search Approach*. IEEE Transaction on Systems, Man and Cybernetics, Part C (Accepted for publication).