Moving Target Search in Grid Worlds

Cagatay Undeger
Savunma Teknolojileri Muhendislik ve Tic.A.S.
Kafkas Sk. No:56 Bestepe
06510 Ankara, Turkey
cundeger@stm.com.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 moving target search algorithm for dynamic and partially observable environments, modeled as grid world. The proposed algorithm, Real-time Moving Target Evaluation Search (MTES), is able to detect the closed directions around the agent, and determine the best direction that avoids the nearby obstacles, leading to a moving target which is assumed to be escaping almost optimally. We compared our proposal with Moving Target Search (MTS) and observed a significant improvement in the solution paths. Furthermore, we also tested our algorithm against A* in order to report quality of our solutions.

Categories and Subject Descriptors

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

General Terms

Algorithms, Performance, Experimentation

Keywords

Moving target search, Grid world, Path planning, Predator

1. INTRODUCTION

Pursuing a moving target is one of the most challenging problems in areas such as robotics, virtual simulations, computer games, etc. Off-line and incremental path planning algorithms are not able to handle moving targets in real-time, and most of the on-line search algorithms are specifically designed for partially observable environments with static targets. The most well known algorithm for moving targets is Moving Target Search (MTS) [3], which maintains a heuristic table that contains estimated costs of paths between every pair of coordinates. Convergence of the estimated costs takes considerable time making MTS a poor algorithm to be used in practice. Upon seeing such inefficiency, the authors developed two extensions namely *Commitment to Goal* (MTS-c) and *Deliberation* (MTS-d).

In this paper, we propose a moving target search algorithm, Real-Time Moving Target Evaluation Search (MTES), which is able to make use of environmental information available to the agent successfully in order to select the best moving direction avoiding nearby obstacles to capture a moving target as soon as possible. To show its performance, we compared MTES with MTS-c, MTS-d and A*.

The related work is given in Section 2. In section 3, MTES is described in brief. The performance analysis is presented in Section 4. Finally, the conclusion is given in Section 5.

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

2. RELATED WORK

Off-line path planning algorithms [7] are hard to use for large dynamic environments and for moving targets because of their time requirements. One way to make these algorithms more efficient is to change them from off-line to incremental [8, 4] in order to avoid re-planning from scratch. Although these algorithms work fine with partially observable environments, they are not capable of handling moving targets. There are also a number of on-line approaches [6, 5, 9, 11, 10, 3]. As a matter of fact, only few of these algorithms can be adapted against a moving target.

Moving Target Search (MTS) [3] is a well known real-time search algorithm for pursuing a moving target. The algorithm maintains a table of heuristic values, presenting the function $h(x, y)$ for all pairs of locations x and y in the environment, where x is the location of the agent and y is the location of the target. The original MTS is a poor algorithm in practice. Therefore, two MTS extensions called *Commitment to Goal* (MTS-c) and *Deliberation* (MTS-d) are also proposed to improve the solution quality [3]. In order to use the learned table values more effectively, MTS-c ignores some of the target's moves, and MTS-d performs an off-line search (deliberation) while in a heuristic depression.

When we look at the prey algorithms, we usually cannot see very successful studies. Mostly the focus is on the predators, and the prey algorithms commonly use hybrid techniques mixing the reactive strategies [1, 3, 2]. Since these reactive algorithms are not good enough for our predator algorithms, we developed an off-line strategy, which is slow but more powerful with respect to its escape capability.

3. MTES

We assume that the environment is a planar grid world and partially observed by the agent. There is a single agent

(predator) that aims to reach a static or moving target (prey) avoiding obstacles in real-time. The agents are able to move north, south, east or west direction in each step. The predator knows the location of the prey all the time, but perceives the grid world up to a predefined vision range, thus maintains a tentative map that holds the known part of the grid world, as he explores the environment. The prey has unlimited perception and knows all the grid world and the location of the predator all the time. The search continues until the predator reaches the prey.

MTES makes use of a heuristic (RTTE-h), which geometrically analyzes the obstacles nearby, tries to estimate the lengths of paths to the prey around these obstacles, and proposes a moving direction in order to lead the predator to the prey through shorter paths. MTES executes the steps shown in Algorithm 1 until reaching the target or determining that the target is unreachable. The algorithm internally calls RTTE-h heuristic to determine the proposed moving direction and the utilities of neighbor cells according to that proposed direction. RTTE-h propagates four diagonal rays away from the agent location to split north, south, east and west moving 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. Then, the closed moving directions and the geometric features of the obstacles are determined. Next, these 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.

Algorithm 1 An Iteration of MTES Algorithm

```
1: Call RTTE-h to compute the proposed direction and the utilities
   of the neighbor cells.
2: if a direction is proposed by RTTE-h then
3:    Select the neighbor cell with the highest utility from the set of
      neighbors cells with non-zero utility and smallest visit count.
4:    Move to the selected direction.
5:    Increment the visit count of previous cell by one.
6:    Insert the previous cell into the history.
7: else
8:    if History is not empty then
9:       Clear all the History.
10:      Jump to 1.
11:   else
12:      Destination is unreachable, stop search with failure.
13:   end if
14: end if
```

The complexity of MTES is $O(w.h)$ per step, where w is the width and h is the height of the grid world. Since increasing the grid size decreases the efficiency, a *search depth (d)* can be introduced in order to limit the worst case complexity. With this limitation, the complexity becomes $O(d^2)$.

4. PERFORMANCE ANALYSIS

In this section, we present our experimental results on MTS, MTES and A*. As being an off-line algorithm, we executed A* in each step from scratch. We used 9 randomly generated sample grids of size 150x150. Six of them were the *maze* grids, and three of them were the *U-type* grids (see Fig. 1). The location of predators were randomly chosen from left or right sides of the grids, and the prey locations were chosen randomly from the middle part. The predator always knows the location of the prey, but perceives the

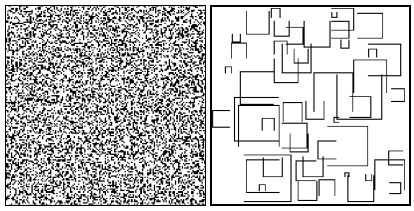

Figure 1: A maze grid (left), a U-type grid (right)

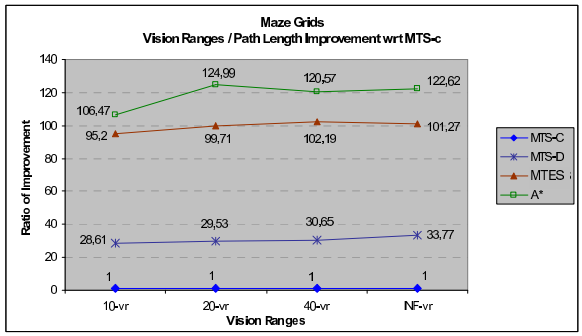

Figure 2: Average of path length results of maze grids for increasing vision ranges

grid world up to a limit, which is called *vision range (v)*. The predator can only sense the cells within the rectangular area of size $(2v+1).(2v+1)$ centered at the agent location. We used the statement *infinite vision* to emphasize that the predator has unlimited sensing capability. Our tests are performed with 10, 20, 40 and *infinite* vision ranges and search depths. Additionally, we assumed that the prey is slower than the predator, and skips 1 move after each 7 moves. To test the algorithms, we also developed an off-line prey algorithm, (Prey-A*), which is powerful but not very efficient. To prevent the side effects caused by the efficiency difference, the predator and the prey algorithms are executed alternately in performance tests.

With respect to vision ranges and search depths, the averages of path lengths on maze grids are given in Figures 2 and 3, and the averages of path lengths on U-type grids are given in Figures 4 and 5, respectively. In the charts, the horizontal axis is either the vision range or the search depth, and the vertical axis contains the ratio of improvement in

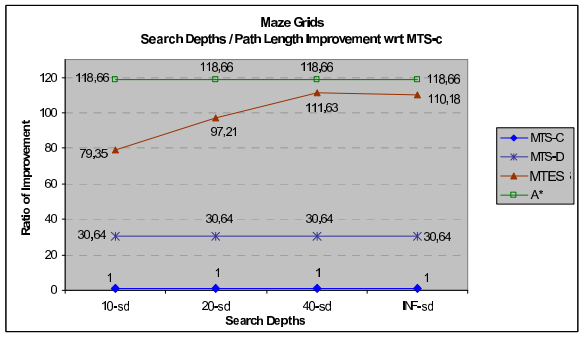

Figure 3: Average of path length results of maze grids for increasing search depths

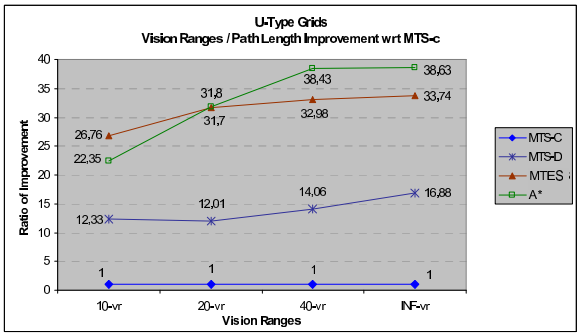

Figure 4: Average of path length results of U-type grids for increasing vision ranges

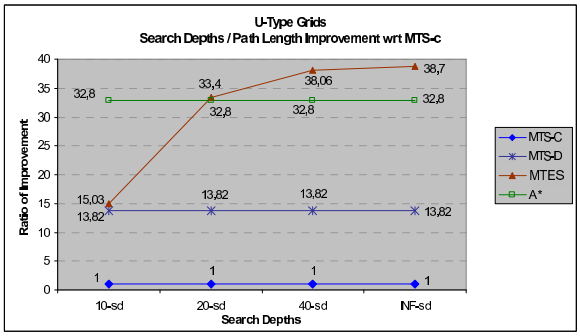

Figure 5: Average of path length results of U-type grids for increasing search depths

the path length with respect to MTS-c. The results showed that MTES performs significantly better than MTS-c and MTS-d even with small search depths, and usually offers near optimal solutions that are almost as good as the ones produced by A*. Especially in U-type grids, MTES mostly outperforms A*. When we examined this interesting result, we observed that they behave very differently in sparse parts of the grid. MTES prefers performing diagonally shaped manoeuvres for approaching targets located in diagonal directions, whereas A* prefers performing L-shaped manoeuvres. Since the agents are only permitted to move in horizontal and vertical directions, these two manoeuvre patterns have equal path distances to a fixed location, but this is not the case for a moving target since the strategy difference significantly affects the behavior of the prey in U-type grids.

We also examined the step execution times of the algorithms running on an AMD Athlon 2500+ computer. In Table 1, the worst case and the average number of moves executed per second in maze and U-type grids are shown. The rows are for the compared algorithms and the columns are for the search depths. According to the results, we can conclude that MTS-c and MTS-d have low and almost constant step execution times whereas the efficiency of MTES is tied to the search depth and obstacle ratio, and hence the appropriate depth should be chosen according to the required efficiency. A* is the worst as expected.

5. CONCLUSION

In this paper, we have examined the problem of pursuing a moving target in grid worlds, and introduced a moving target search algorithm, MTES. To see the performance of MTES, we have compared MTS-c, MTS-d, MTES and A* against a successful prey algorithm, Prey-A*.

With respect to path lengths, the experimental results showed that MTES performs significantly ahead of MTS, and competes with A*. We also observed that the two MTS versions are significantly different from each other. Although, MTS-d performs acceptably good, MTS-c almost never offers good solutions. In terms of step execution times, we observed that MTS is the most efficient algorithm, and almost spend constant time in each move. But its solution path lengths are usually unacceptably long. MTES follows MTS, and its efficiency is inversely proportional to the increase in obstacle density. Finally, A* is always the worst.

Table 1: The worst case and average number of moves per second for increasing search depths

Maze grids

Depth	10-sd	20-sd	40-sd	INF-sd
MTS-c	1063/2676	1063/2676	1063/2676	1063/2676
MTS-d	937/2412	937/2412	937/2412	937/2412
MTES	531/1101	212/692	82/413	23/283
A*	20/189	20/189	20/168	20/189

U-type grids

Depth	10-sd	20-sd	40-sd	INF-sd
MTS-c	1063/2855	1063/2855	1063/2855	1063/2855
MTS-d	1062/2498	1062/2498	1062/2498	1062/2498
MTES	793/1257	400/747	133/348	57/233
A*	8/104	8/104	8/104	8/104

6. REFERENCES

[1] M. Goldenberg, A. Kovarsky, X. Wu, and J. Schaeffer, 'Multiple agents moving target search', 1536–1538, (2003).

[2] T. Haynes and S. Sen, 'Evolving behavioral strategies in predators and prey', (1996).

[3] T. Ishida and R.E. Korf, 'Moving target search: A real-time search for changing goals', *IEEE Trans Pattern Analysis and Machine Intelligence*, **17**(6), 97–109, (1995).

[4] S. Koenig and M. Likhachev, 'Fast replanning for navigation in unknown terrain', *Transactions on Robotics*, **21**(3), 354–363, (2005).

[5] S. Koenig and M. Likhachev, 'Real-time adaptive a*', *5th Int'l Joint Conf. on Autonomous Agents and Multiagent Systems*, 281–288, (2006).

[6] R.E. Korf, 'Real-time heuristic search', *Artificial Intelligence*, **42**(2-3), 189–211, (1990).

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

[8] A. Stentz, 'The focussed D* algorithm for real-time replanning', *In Proceedings of the Int'l Joint Conference on Artificial Intelligence*, (1995).

[9] C. Undeger and F. Polat, 'Real-time edge follow: A real-time path search approach', *IEEE Transaction on Systems, Man and Cybernetics, Part C, (In press)*.

[10] C. Undeger and F. Polat, 'Rttes: Real-time search in dynamic environments', *Applied Intelligence (In press)*.

[11] C. Undeger and F. Polat, 'Real-time target evaluation search', *5th Int'l Joint Conf. on Autonomous Agents and Multiagent Systems, AAMAS-06*, 332–334, (2006).