Moving Target Search with Intelligence

Toru Ishida
NTT Communication Science Laboratories
Sanpeidani, Inuidani, Seika-cho, Soraku-gun, Kyoto 619-02, Japan
ishida@cslab.kecl.ntt.jp

Abstract

We previously proposed the *moving target search (MTS)* algorithm, where the location of the goal may change during the course of the search. MTS is the first search algorithm concerned with problem solving in a dynamically changing environment. However, since we constructed the algorithm with the minimum operations necessary for guaranteeing its *completeness*, the algorithm as proposed is neither efficient nor intelligent.

In this paper, we introduce innovative notions created in the area of *resource-bounded planning* into the formal search algorithm, MTS. Our goal is to improve the efficiency of MTS, while retaining its completeness. Notions that are introduced are (1) *commitment to goals*, and (2) *deliberation for selecting plans*. Evaluation results demonstrate that the intelligent MTS is 10 to 20 times more efficient than the original MTS in uncertain situations.

1. Introduction

Existing search algorithms can be divided into two classes: *off-line search*, which computes the entire solution path before executing the first step in the path, and *realtime search*, which always computes a plausible next move and physically executes that move in constant time. However, all these algorithms assume that the problem is fixed and does not change over the course of the search. On the other hand, in the area of *planning*, where search algorithms have been often utilized, researchers are recently focusing on dynamically changing environments, where the goal state and the problem space change in run-time, and thus agents are *resource-bounded* in constructing plans. Therefore, search algorithms capable of handling changing problems will provide a computational basis for resource-bounded planning.

In this direction, we have already investigated the case wherein the goal state changes during the course of the search, and proposed the *moving target search (MTS)* algorithm [Ishida and Korf, 1991]. Off-line search seems infeasible for implementing MTS because the search takes exponential time, and thus the target would have moved to a new position by the time a path was found. Therefore, we started from realtime search [Korf, 1990], and extended it to MTS. The MTS algorithm has been proved to be *complete* in the sense that it will eventually reach the target, assuming a finite problem space and that the speed of the problem solver is faster than that of the target.

We originally implemented MTS with the minimum operations necessary for guaranteeing its completeness. The obtained algorithm consists of a pair of steps, which are repeatedly performed: the first step is incremental learning of the estimated distance between the problem solver and the target, and the second step moves the problem solver in the problem space. As a result, MTS is *reactive*, i.e., capable of performing each move in constant time, but it is neither efficient nor intelligent. Various experiments showed that the efficiency of the algorithm decreases rapidly as *uncertainty* increases. This is because, in uncertain situations, the heuristic function does not return useful information and the learning cost becomes high.

In the area of resource-bounded planning, however, agent architectures to cope with environmental changes have been studied [Georgeff *et al.*, 1987]. Cohen *et al.* [1990] have defined the notion of *commitment* as a persistent goal. Kinny *et al.* [1991] quantitatively evaluated how the *degree of commitment* affects agents' performance.[1] The role of *deliberation* has been investigated by Bratman *et al.* [1988]. Pollack *et al.* [1990] proposed the experimental environment called *Tileworld* and have been quantitatively evaluating the *tradeoff between deliberation and reactiveness*. The challenge of this paper is to introduce these notions into formal search, and to improve the efficiency of MTS while retaining its completeness. Introduced notions are as follows:

Commitment to goals:

In MTS, the problem solver always knows the target's position, but can ignore some of its moves. Experi-

[1] Durfee *et al.*[1988] performed a similar evaluation in multi-agent environments.

ments show that changing the goal causes the problem solver to start incremental learning over again toward the new goal. Thus, in uncertain situations, where the learning cost is significant, better performance could be obtained by committing to its goal and not changing it even if the target moves.

Deliberation for selecting plans:

When the problem solver moves, MTS always selects the neighboring position that offers the minimum estimated distance to the goal. However, as the situation becomes uncertain, such a reactive decision becomes inaccurate and often does not result in better performance. Thus, deliberative investigation using off-line search, though it decreases the speed of the problem solver, might improve overall performance in uncertain situations.

2. Moving Target Search

We briefly characterize the moving target search problem [Ishida and Korf, 1991]. The problem space is represented as a connected graph. The graph is undirected, allowing motion of either the problem solver or the target along an edge in any direction. We assume that all edges in the graph have unit cost. There is an initial position of the problem solver and an initial position of the target. The problem solver does not have a map of the problem space. We assume the problem solver and the target move alternately, and can traverse at most one edge in each turn. We reduce the speed of the target by assuming that periodically the target will make no move, and remain at its current position. The problem solver has no control over the movements of the target, but always knows the target's position. There also exists a *heuristic function* that returns an estimate of the distance between any pair of states. Note that MTS must acquire heuristic information for each goal location. The heuristic function must be *admissible*, meaning it never overestimates the actual distance [Pearl, 1984]. The task is accomplished when the problem solver and the target occupy the same state.

The MTS algorithm is as follows. As the initialization of the algorithm, the current state of the problem solver is assigned to x, and the current state of the target to y. The heuristic function $h(x,y)$ represents the estimated distance between x and y. The following steps are repeatedly performed until the task is accomplished.

When the problem solver moves:

1. Calculate $h(x',y)$ for each neighbor x' of x.
2. Update the value of $h(x,y)$ as follows:

$$h(x,y) \leftarrow max \left\{ \begin{array}{l} h(x,y) \\ min_{x'}\{h(x',y)+1\} \end{array} \right\}$$

3. Move to the neighbor x' with the minimum $h(x',y)$, i.e., assign the value of x' to x. Ties are broken randomly.

When the target moves:

1. Calculate $h(x,y')$ for the target's new position y'.
2. Update the value of $h(x,y)$ as follows:

$$h(x,y) \leftarrow max \left\{ \begin{array}{l} h(x,y) \\ h(x,y')-1 \end{array} \right\}$$

3. Reflect the target's move to the problem solver's goal, i.e., assign the value of y' to y.

The MTS algorithm is *complete* in the sense that the problem solver executing MTS is guaranteed to eventually reach the target, assuming a finite problem space, in which a path exists between every pair of nodes, starting with non-negative admissible initial heuristic values, and if the target periodically skips moves.

The completeness of MTS is proved as follows: Define the *heuristic error* as the sum over all pairs of states a and b of $h^*(a,b) - h(a,b)$ where $h^*(a,b)$ is the length of the shortest path between a and b, and $h(a,b)$ is the current heuristic value. Define the *heuristic difference* as $h(x,y)$, the current heuristic value between the current state of the problem solver, x, and the current state of the target, y. Define the *heuristic disparity* as the sum of the heuristic error and the heuristic difference. Proof was made by showing that the heuristic disparity decreases by at least one unit when the problem solver moves, and increases by at most one unit when the target moves [Ishida and Korf, 1991].

3. Performance Bottleneck of MTS

Experiments

To examine MTS performance, we implemented it in a rectangular grid problem space (100×100) with obstacles. We allow motion along the horizontal and vertical dimensions, but not along the diagonal. To erase problem space boundaries, we formed a torus by connecting the opposite boundaries. Note that the rectangular grid represents one of the many problem spaces possible, but not a physical workspace. Though MTS is effective in any type of problem space, we use the rectangular grid as an experimental environment simply because it is suitable for plotting on a workstation display, and it helps humans to intuitively recognize what is going on in the problem space. Though each state basically has four neighbors, obstacles are generated by restricting the number of neighboring states.

The Manhattan distance is used as the initial heuristic value. This is because the Manhattan distance represents the actual distance if there is no obstacle, but it does becomes inaccurate as the number of obstacles increases. Thus the combination of obstacles and the Manhattan distance can naturally produce any degree of uncertainty. The problem solver and the target are initially positioned as far apart as possible in the torus, i.e., 100 units in Manhattan distance. Figure 1 illustrates the sample track of MTS. In this figure, obstacles were manually positioned, and the motion of the

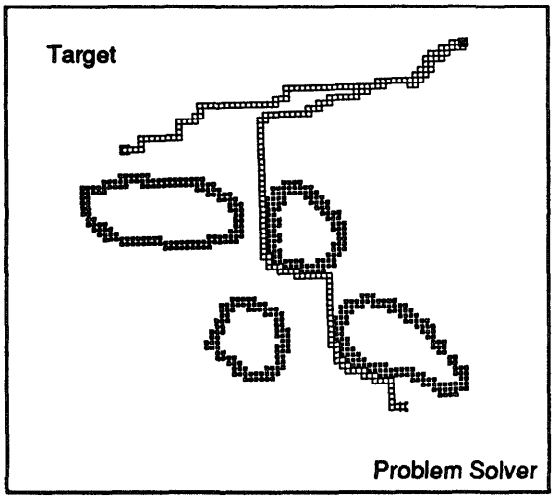

Figure 1 Sample Track of MTS

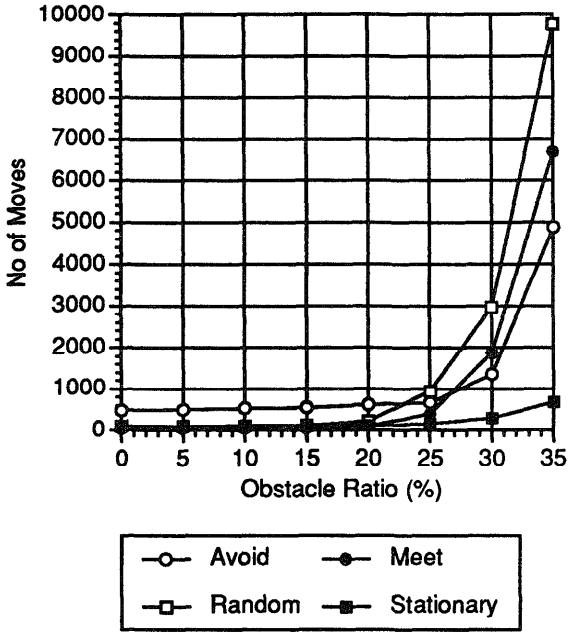

Figure 2 Performance of MTS

target was controlled by a human user to escape from the problem solver.

Figure 2 shows the performance of MTS. In this evaluation, obstacles are randomly positioned. X-axis represents the ratio of obstacles: an obstacle ratio of 20% means that 2000 junctions in the 100×100 grid are randomly replaced by obstacles. With high obstacle ratios (more than 20%), obstacles join up and form walls with various shapes. The complexity of the maze rapidly increases as the ratio increases from 25% to 35%. When the ratio reaches 40%, the obstacles tend to disconnect the problem space, separating the target from the problem solver. Y-axis represents the number of moves taken by the problem solver to catch the target. Numbers in this figure are obtained by averaging 100 trials. The speed of the target is set to 80% of the problem solver: the target skips one in every five moves. The motion of the target is controlled by programs with the following four response modes.

Avoid: The target actively avoids the problem solver: the target performs MTS to move toward the position as far apart as possible from the problem solver in the torus.

Meet: The target moves cooperatively to meet the problem solver: the target performs MTS to decrease the estimated distance from the problem solver.

Random: The target moves randomly.

Stationary: The target remains stationary. In this case, MTS behaves exactly same as the realtime search algorithm called *Learning Real-Time A** [Korf, 1990].

The results of experiments show that the performance decreases as the situation becomes uncertain. Though this phenomena is observed in all target's behavior modes, the performance decreases more rapidly when the target moves than when it remains stationary. In *Meet*, though both agents perform MTS to meet each other, the effect appears to be negative. This is because, in uncertain situations, the agents tend to reach the opposite sides of the same wall, and move back and forth in confusion.

Heuristic Depression

Let us examine the MTS behavior in more detail to figure out why MTS becomes inefficient in uncertain situations. Figure 3 represents the behavior of the problem solver in a one dimensional problem space. X-axis represents the positions of the problem solver and the target, while Y-axis represents the estimated distance between the problem solver and the target. The initial heuristic values are plotted with wide lines. Arrows indicate moves of the problem solver. Incremental updates of heuristic values performed by the problem solver are indicated by dark boxes. In this figure, the target is assumed not to move. As described in Figure 3, the problem solver performing MTS repeatedly moves along the slope of heuristic values.

To explain the problem solver's behavior, we define a *heuristic depression* for each goal state as follows: Start from a set with single state whose heuristic value is equal to or less than those of the surrounding states. Extend the set by adding any neighboring state while keeping the heuristic values in the set equal to or less

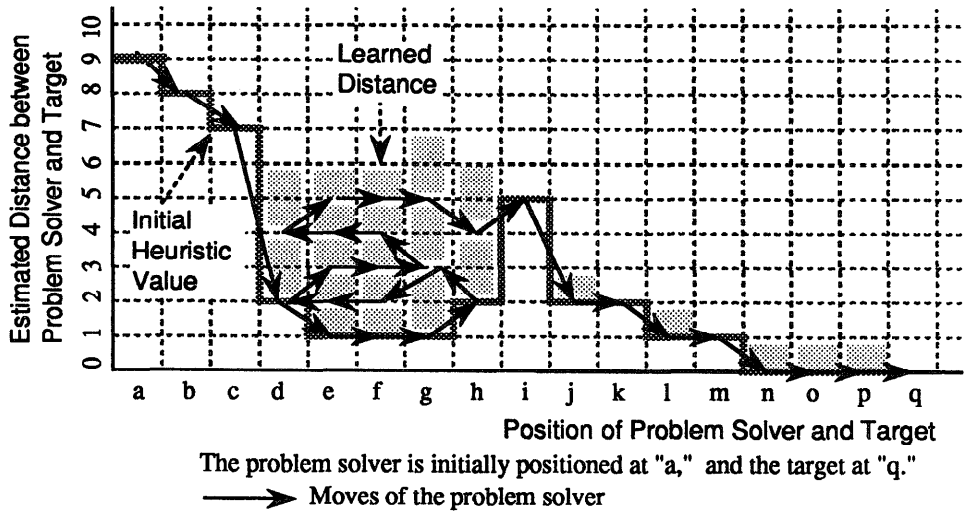

Figure 3 Behavior of MTS

than those of the states surrounding the extended set. When no more state can be added, we call the resulting set a heuristic depression. For example, in Figure 3, positions from d to h form a heuristic depression. Note that no depression exists in the actual distance. However, as the situation becomes uncertain, heuristic values differ significantly from the actual distances, and so heuristic depressions tend to appear frequently in the problem space.

When placed in a heuristic depression, the problem solver has no way to decrease the heuristic difference, and recognizes that its heuristic values are inaccurate. The problem solver cannot reach the target without filling the depression by repeatedly updating the heuristic values. Suppose the target moves during this learning process. Since MTS must maintain heuristic values for each goal location, the problem solver has to start the learning process over again for the target's new position. This is why MTS performance rapidly decreases in uncertain situations.

To summarize, *the performance bottleneck of MTS exists in its inefficiency of filling the heuristic depression.* Thus, in the following sections, we propose to introduce two notions created in the area of resource-bounded planning: (1) *commitment to goals* to ignore the target's moves and to concentrate on filling the heuristic depression, and (2) *deliberation for selecting plans* in which off-line search is performed to find a direction for getting out of the heuristic depression. The effect of introducing these notions will be reported in Section 6 using the same examples given in Figure 2. In the following discussion, we distinguish the original MTS algorithm as *BMTS (Basic Moving Target Search)* and the improved algorithm as *IMTS (Intelligent Moving Target Search).*

4. Introducing Commitment

In BMTS, the problem solver always knows the position of the target. However, we can extend BMTS so that the problem solver can ignore some of the target's moves. The extended BMTS only requires that the problem solver knows the position of the target at some point before the problem solver reaches the last known position of the target [Ishida and Korf, 1991]. The question is, when the problem solver should ignore the target's moves and when it should not.

In this paper, we propose that the problem solver *reflects the target's move in the problem solver's goal only when the heuristic difference decreases; otherwise (i.e., when placed in a heuristic depression) it commits to the target's previous position and does not change its goal even if the target moves.* However, as shown in Figure 3, when filling a large heuristic depression, updating heuristic values creates small depressions, and thus the heuristic difference may temporarily decrease (i.e., arrows are sometimes directed down). In such a situation, even if the heuristic difference decreases, retargetting still causes the learning process to be restarted again. Therefore, we introduce the *degree of commitment (doc)*, which represents the strength of the problem solver's commitment to its goal.

Let $DOWN$ be the number of problem solver's moves for which the heuristic difference continuously decreases. Let the problem solver reflect the target's move in its goal only when $DOWN \geq doc$. Obviously, when $doc = 0$ the problem solver is most sensitive to the target's moves, and behaves exactly the same as BMTS. On the other hand, when $doc = \infty$, the problem solver is least sensitive. Note that when the problem solver reaches the committed goal (i.e., $x = y$), the problem solver must always change its goal to the target's new

position. The problem solver thus changes its goal only when it reaches the last committed position of the target.

The IMTS algorithm with commitment is as follows. $DOWN$ is set to 0 before execution.

When the problem solver moves:

1. Calculate $h(x', y)$ for each neighbor x' of x.
2. If $h(x, y) > min_{x'}\{h(x', y)\}$, $DOWN \leftarrow DOWN + 1$.
   If $h(x, y) \leq min_{x'}\{h(x', y)\}$, $DOWN \leftarrow 0$.
3. Update the value of $h(x, y)$ as follows:

$$h(x, y) \leftarrow max \left\{ \begin{array}{l} h(x, y) \\ min_{x'}\{h(x', y) + 1\} \end{array} \right\}$$

4. Move to the neighbor with the minimum $h(x', y)$, i.e., assign the value of x' to x. Ties are broken randomly.

When the target moves:

When $DOWN \geq doc$ or $x = y$, perform the following:

1. Calculate $h(x, y')$ for the target's new position y'.
2. Update the value of $h(x, y)$ as follows (let t be the number of target's moves from y to y'):

$$h(x, y) \leftarrow max \left\{ \begin{array}{l} h(x, y) \\ h(x, y') - t \end{array} \right\}$$

3. Reflect the target's move in the problem solver's goal, i.e., assign the value of y' to y.

In the above algorithm, the constant t is used to represent the number of the target's moves from the problem solver's current goal y to the target's new position y'.[2] The completeness of the above algorithm can be easily obtained by extending the proof in [Ishida and Korf, 1991], and thus we omit it from this paper due to space limitations.

5. Introducing Deliberation

Realtime search is not always more efficient than off-line search. From our experience, in uncertain situations, off-line search often results in better performance. This is because, in off-line search, the problem solver does not move back and forth, but extends its wave front step by step even in uncertain situations. Our expectation is that introducing off-line search enables MTS to efficiently find the boundary of a heuristic depression, and thus overcomes the performance bottleneck. The question is, when and how far the problem solver should perform off-line search. Note that the target might run away if the problem solver inappropriately performs off-line search.

Our idea is that *the problem solver performs realtime search while the heuristic difference decreases; otherwise (i.e., when placed in a heuristic depression) performs off-line search.* To consider the cost of off-line search, we assume that, in each turn, the problem solver can expand one state in off-line search, instead of moving one edge in realtime search. This allows the target to move during the problem solver expands states in off-line search.[3] We then introduce the *degree of deliberation (dod)* to restrict the range of off-line search. Let $CLOSED$ be a set of expanded states and $OPEN$ be a set of visited but not yet expanded states. States in a heuristic depression are to be collected in $CLOSED$. The number of states in $CLOSED$ is denoted by $|CLOSED|$. We allow the problem solver to perform off-line search only when $|CLOSED| < dod$. Obviously, when $dod = 0$ and $doc = 0$, IMTS behaves exactly the same as BMTS. However, as dod increases, the problem solver tends to spend more time in deliberation.

The IMTS algorithm with deliberation is shown as follows. $CLOSED$ and $OPEN$ are cleared before execution. The algorithm starts in the realtime mode. When the target moves, the same algorithm in Section 4 is applied.

When the problem solver moves:

[A] When in the realtime mode, perform the following:

1. Calculate $h(x', y)$ for each neighbor x' of x.
2. If $h(x, y) > min_{x'}\{h(x', y)\}$, $DOWN \leftarrow DOWN + 1$.
   If $h(x, y) \leq min_{x'}\{h(x', y)\}$, $DOWN \leftarrow 0$.
3. If $h(x, y) > min_{x'}\{h(x', y)\}$ or $dod = 0$, perform the following:

3.1 Update the value of $h(x, y)$ as follows:

$$h(x, y) \leftarrow max \left\{ \begin{array}{l} h(x, y) \\ min_{x'}\{h(x', y) + 1\} \end{array} \right\}$$

3.2 Move to the neighbor with the minimum $h(x', y)$, i.e., assign the value of x' to x. Ties are broken randomly.

4. If $h(x, y) \leq min_{x'}\{h(x', y)\}$ and $dod \neq 0$, shift to the off-line mode and execute [B].

[B] When in the off-line mode, perform the following:

[2] Though it is not discussed in this paper, it might be a good idea to take account of the distance between y and y', when reflecting the target's move in the problem solver's goal, i.e., if the target has moved far from the problem solver's goal, it might be reasonable to change the goal to the target's new position.

[3] The *lookahead mechanism* has been introduced to reduce the number of moves in realtime search [Korf, 1990]. However, this mechanism is assumed to be completed in constant time (i.e., in each problem solver's turn), and thus the tradeoff between deliberation and reactiveness has not been discussed. In this paper, we introduce off-line search into MTS, taking account of its overhead.

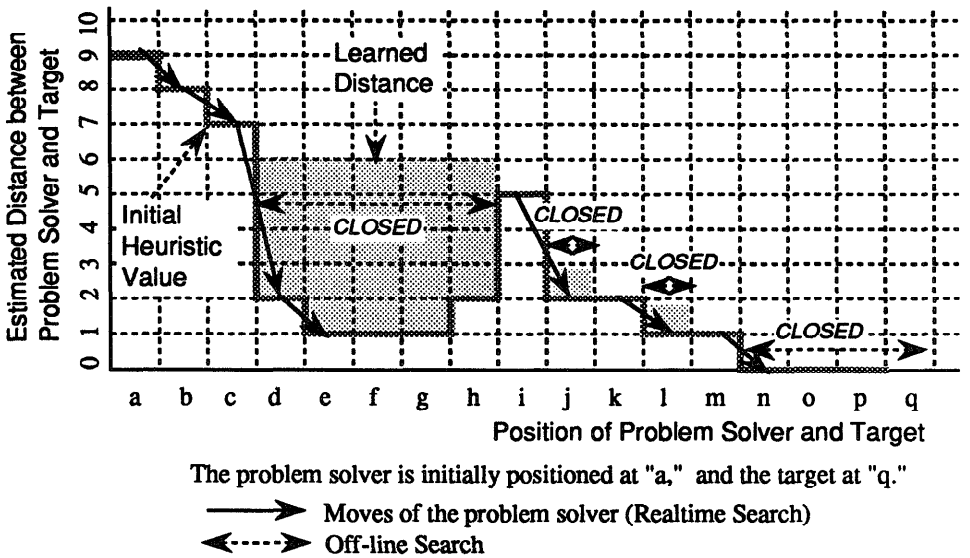

Figure 4 Behavior of IMTS with Deliberation

1. Calculate $h(x', y)$ for each neighbor $x' \notin CLOSED$ of x.
2. If $h(x, y) \leq min_{x'}\{h(x', y)\}$ and $|CLOSED| < dod$, perform the following:

2.1 Expand x: For each neighbor $x' \notin CLOSED \cup OPEN$ of x, add x' to $OPEN$. Add x to $CLOSED$.

2.2 Set x to $x_{open} \in OPEN$ with the minimum $h(x_{open}, y)$.

3. If $h(x, y) > min_{x'}\{h(x', y)\}$ or $|CLOSED| \geq dod$, perform the following.

3.1 For all $x_{closed} \in CLOSED$, update the value of $h(x_{closed}, y)$ as follows:

$$h(x_{closed}, y) \leftarrow h(x, y) + 1$$

3.2 Clear $OPEN$ and $CLOSED$.

3.3 Shift to the realtime mode and execute [A].

Figure 4 illustrates the deliberation process of the above algorithm using the same situation given in Figure 3. The problem solver starts in the realtime mode. When placed in a heuristic depression, the problem solver commits to the target's current position and shifts to the off-line mode. The problem solver then performs off-line search to find a boundary of the depression. When the boundary is found, the problem solver updates the heuristic values of all states in $CLOSED$, gets out of the depression, shifts to the realtime mode, and continues to perform realtime search while the heuristic difference decreases.

We briefly show the completeness of IMTS with deliberation. Since the number of states in $CLOSED$ is upper bounded by dod, each turn of the problem solver (both in the realtime and off-line modes) can be processed in constant time. At each turn, the problem solver selects the realtime mode or the off-line mode. When the off-line mode is selected, (1) the number of states in $CLOSED$ increases by one, or (2) the heuristic values of the states in $CLOSED$ are updated all at once. In the former case, the heuristic disparity does not decrease, but in the latter case the heuristic disparity decreases by $|CLOSED|$ units. This is because, for each state in $CLOSED$, the heuristic error decreases by $h(x, y) + 1 - h(x_{closed}, y)$, while heuristic difference might increase by $h(x, y) - h(x_{closed}, y)$, and thus the heuristic disparity decreases by one unit. Therefore, in the combined sequence of the problem solver's moves, the heuristic disparity decreases by at least $|CLOSED|$ units per $|CLOSED|$ turns. On average, the heuristic disparity is decreased by at least one unit in each turn in the off-line mode. In the realtime mode, since the process is the same as BMTS, the heuristic disparity decreases by at least one unit for each turn. On the other hand, when the target moves, the heuristic disparity increases by an average of at most one unit for each turn. Since the target periodically skips moves, the problem solver will eventually reach the target.

6. Evaluation Results

We evaluated the effectiveness of IMTS in the same situation described in Figure 2. Note that the problem solver performs IMTS, while the target performs BMTS in *Avoid*, and IMTS in *Meet*. This is to more clearly show the performance improvement possible with the IMTS algorithms: In *Avoid*, we compare the case of IMTS pursuing BMTS with the case of BMTS pursu-

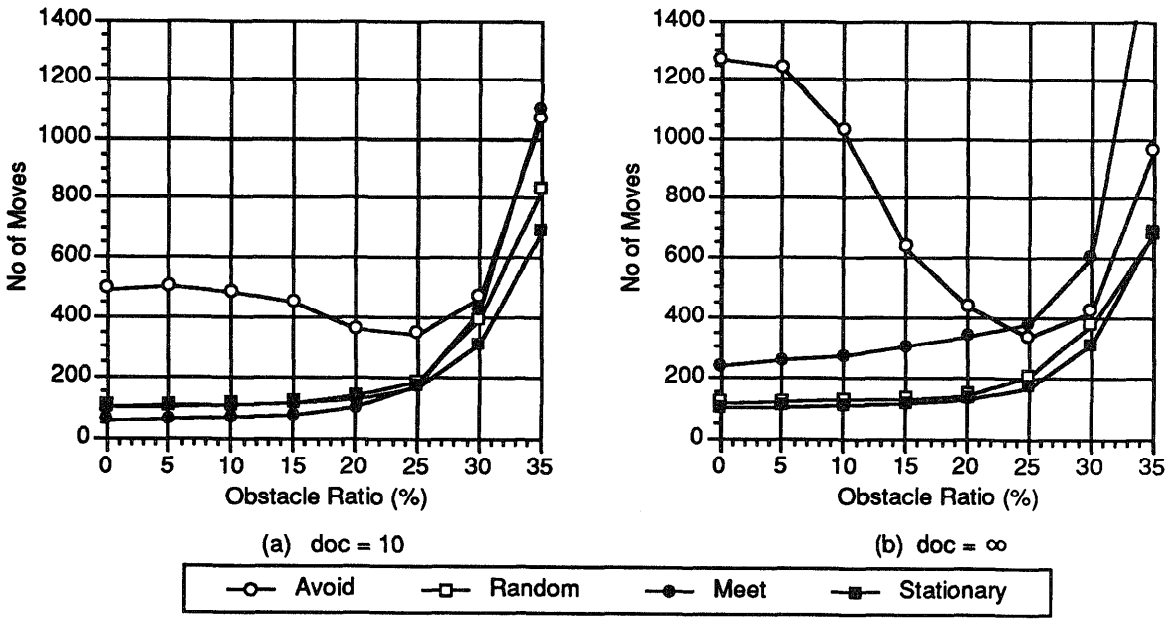

Figure 5 Performance of IMTS with Commitment

ing BMTS (as in Figure 2), and in *Meet*, the cooperative behavior of IMTS agents is compared with that of BMTS agents (as in Figure 2). The major results are as follows:

Commitment to goals:

Figure 5 plots the evaluated IMTS performance under the conditions of $doc = 10$ and $doc = \infty$. Since Figure 2 can be seen as the result of $doc = 0$, the effects of the degree of commitment to the overall performance can be evaluated by comparing these three figures.

When $doc = 10$, actually this value yields almost the best performance for all *doc* values from 0 to ∞, the performance improvement is 12 times in *Random*, 6 times in *Meet*, and 4 times in *Avoid*. The reason why the largest effect is obtained in *Random* is that the target cannot deviate from the initial position because of the random moves, and thus the problem solver can safely ignore the target's moves.

However, increasing the degree of commitment does not always improve the performance. When $doc = \infty$, in *Avoid*, the performance instead decreases when the obstacle ratio is low. That is, in certain situations, since the heuristic function returns a fairly good estimation, the problem solver had better be sensitive to the target's moves. In *Meet*, the performance also decreases but for all obstacle ratios. This decrease is considered to be caused by ignoring the target's cooperative behavior.

Deliberation for selecting plans:

Figure 6 represents IMTS performance with deliberation under the values of $dod = 5$ and $dod = 25$, which roughly mean off-line search is performed to the depth of 2 and 4, respectively. The degree of commitment (*doc*) is always set to 10. Compared with Figure 5(a), when $dod = 25$, the performance is further doubled in uncertain situations. These effects are observed in all target behavior modes including *Stationary*. This shows that deliberation is effective not only in moving target problems, but also in real-time search for fixed goals.

Unlike the introduction of commitment, the performance does not decrease even when the degree of deliberation further increases. This is because $|CLOSED|$ cannot be too large in randomly generated maps, since the ranges of heuristic depressions are naturally upper bounded. If $|CLOSED|$ is large, increasing the degree of deliberation might decrease IMTS performance.

To summarize, introducing commitment and deliberation dramatically improves the efficiency of MTS. The evaluation results clearly show that (1) MTS performance is improved by 10 to 20 times in uncertain situations depending on the target's behavior modes, and (2) controlling the degree of commitment is essential to produce the optimal performance.

7. Conclusion

MTS performance has been improved by introducing notions created in the area of resource-bounded plan-

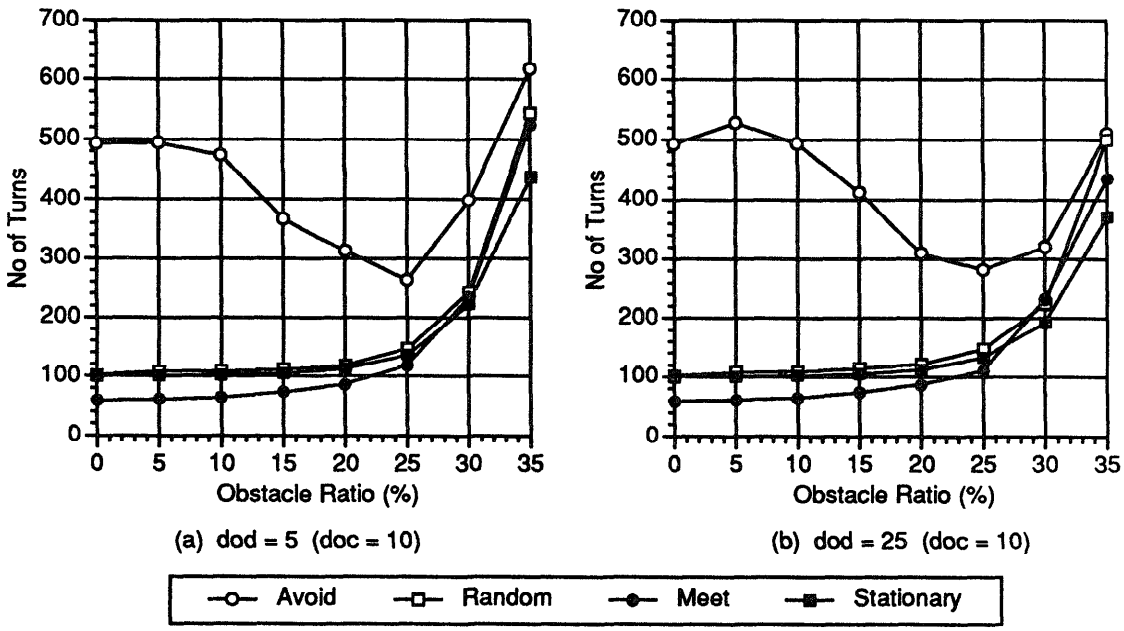

Figure 6 Performance of IMTS with Deliberation

ning. Since only a few steps are added to the original MTS, the obtained intelligent MTS has not lost its simplicity. However, the behaviors of the two algorithms as plotted on a workstation display are substantially different. The intelligent MTS behaves like a predator: In certain situations, the problem solver is always sensitive to the target's moves and reactively moves toward the target current position, while in uncertain situations, the problem solver ignores the target's moves, commits to its current goal, and deliberates to find a promising direction to reach the goal.

Throughout this work, we have tried to bridge the studies on conceptual modeling and algorithms concerned with resource-bounded planning. The results suggest that combining innovative notions and computationally sound algorithms will provide robust and efficient methodologies for problem solving in dynamically changing environments.

Acknowledgments

The author wishes to thank Kiyoshi Nishikawa and Ryohei Nakano for their support during this work at NTT Laboratories, and Richard Korf, Jun-ichi Akahani, Kazuhiro Kuwabara and Makoto Yokoo for their helpful discussions.

References

[Bratman *et al.*, 1988] M. E. Bratman, D. J. Israel and M. Pollack, "Plans and Resource Bounded Practical Reasoning," *Computational Intelligence*, 4(4), pp. 349-355, 1988.

[Cohen *et al.*, 1990] P. R. Cohen and H. J. Levesque, "Intention is Choice with Commitment," *Artificial Intelligence*, 42(3), 1990.

[Durfee *et al.*, 1988] E. H. Durfee and V. R. Lesser, "Predictability versus Responsiveness: Coordinating Problem Solvers in Dynamic Domains," *AAAI-88*, pp. 66-71, 1988.

[Georgeff *et al.*, 1987] M. P. Georgeff and A. L. Lansky, "Reactive Reasoning and Planning," *AAAI-87*, pp. 677-682, 1987.

[Ishida and Korf, 1991] T. Ishida and R. E. Korf, "Moving Target Search," *IJCAI-91*, pp. 204-210, 1991.

[Kinny *et al.*, 1991] D. N. Kinny and M. P. Georgeff, "Commitment and Effectiveness of Situated Agents," *IJCAI-91*, pp. 82-88, 1991.

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

[Pearl, 1984] J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley, Reading, Mass., 1984.

[Pollack *et al.*, 1990] M. E. Pollack and M. Ringuette, "Introducing the Tileworld: Experimentally Evaluating Agent Architectures," *AAAI-90*, pp. 183-189, 1990.