

# Dynamic Programming Equation Based Multi-Robot Searching

YAN MENG

Department of Electrical and Computer Engineering  
Stevens Institute of Technology  
Hoboken, NJ 07030  
USA  
yan.meng@stevens.edu

**Abstract:** It is a challenging task for multiple robots working together efficiently to search for multiple targets in an indoor office environment. It is assumed that a rough priori probability map of the targets' distribution is given. To coordinate the behaviours of individual robots and consider the interaction between the robots, a dynamic-programming equation (DPE) based algorithm is proposed to estimate the utility function of each robot so that overall global performance can be achieved. To reduce the computational complexity of the dynamic equation based method, event-driven discretization and one-step dynamic programming are presented. Simulation results in a searching task in an indoor office environment demonstrate the efficiency and robustness of the DPE-based method.

**Key-Words:** - multi-robot systems, dynamic programming equation, and multi-target searching

## 1. INTRODUCTION

It is an important and challenging field of research to design and deploy a multi-robot system to perform complex coordinate tasks, such as exploration, searching, and map-building. One of the key issues for a multi-robot system is how to coordinate the behaviors of individual robots so that the overall global performance can be optimized.

To be more efficient for a searching task, it is reasonable to assume that some partial information is available either through distributed sensors installed in the searching area, or based on some heuristics from human beings under the emergency situations. One natural way to capture available information is to represent it as probability map, which is the likelihood of the target presence in the search space.

Generally, an approach that proved to be more efficient for searching tasks consists of discretizing time and partitioning the continuous space into a finite collection of cells. The search problem is then reduced to deciding which cell to visit at each time interval. In the UAV search scenarios, Bertuccelli and

How [2] introduced a statistical framework for an UAV searching for static targets, where the probability associated with each cell was described by a Beta distribution that was updated based on whether a target was detected or not by the sensor. This approach was extended later in [3] for an UAV search with uncertain probability maps to the case of dynamic targets, where the Beta distributions were updated using a recursive method that includes both propagation and measurement update steps.

Eagle [5] noted that a discrete search can be formulated as an optimization on a partially observable Markov decision process (POMDP) and proposed a dynamic programming solution to it. However, since the optimization of POMDPs is extremely computationally extensive, this approach is often not practical. Instead, Eagle and Yee [6], and Stewart [18] formulated the discrete search as nonlinear integer programming problem and proposed branch-and-bound procedures to solve it, which in the case of [6] are optimal.

However, updating the state of each cell at each time interval would be computational/memory expensive and impractical in real-world scenarios. Therefore, other discretization methods have been proposed with probability maps. In Urban Search and Rescue (USAR) scenarios, Murphy [16] proposed a biomimetic search strategy where a robot partitions the search space based on the semantic understanding of the expected distribution of survivors, then systematically searches each of the volumes in ranked order. In the case of finding targets in free space, such as boats lost at sea, Bourgault et al. [4] employed a Bayesian approach where the target probability density function (PDF) is used as a priori information. As rescue air vehicles cover the ocean, the target PDF is updated using the model of the sensor and expected target motion. Optimal trajectories for the search are those that maximize the cumulative probability of detection over a limited time horizon. Lau et al. [13] divided a searching space into different regions where available information about the targets was expressed by the expected proportion of targets present in each region. A search strategy that minimizes the expected average time for detection of each target was

proposed, which deals with an environment where multiple targets may be present and allows travel and search costs of individual regions to be arbitrarily specified.

However, all of these searching strategies only deal with a single robot. As we know, a multi-robot system is more desirable in some scenarios, such as exploration, and searching, due to the robustness, stability, adaptability, and scalability. Meanwhile, the mutual interactions between individual robots sharing a common workspace could be much more complex in general cases.

Some research work have been conducted on the multi-robot searching. Kok et. al. [12] described a framework to coordinate multiple robots using coordinate graphs (CG). First a discretization of the state by assigning roles to the agents was conducted. Then a CG-based method was applied to the derived set of roles. The capability of dynamic update of the topology of the CG and the role-based coordination rules made this approach to be practical for large scale teams. In [15], an optimal searching strategy for a target on multiple concurrent rays in parallel using  $p$  robots was presented based on a solution of *cow path problem* that can be treated as a special condition with  $p = 1$  and has been solved in [1][11]. The interaction between the robots is relatively simple in this case due to the special configurations. In [10], several different targets were considered in a heterogeneous team of UAVs, where some target locations are suspected a priori with a certain probability, while the others are initially unknown.

For those more complex dynamic environments, another methodology of solving coordination problem utilizes a game theory, which is a convenient tool for modeling and solving multi-robot interaction problems. Skrzypczyk [17] proposed an architecture of a control system for a real time collision-free movement coordination in a multi-robot environment, performing their navigational tasks by using the normal form games. In [14], an approach for analyzing and selecting time-optimal coordination strategies for multiple robots whose configurations were constrained to lie on C-space road map was proposed. The maximal Nash Equilibrium concept was used to find the optimal strategies for each robot. A pursuit-evasion problem as a Markov game was described in [9], which was the generalization of a Markov decision process to the case when the system evolution is governed by a transition probability function depending on two or more players' actions. This probabilistic setting made it possible to model the uncertainty affecting the player's motion. Combining exploration and pursuit in a single problem was then translated into learning the transition probability function while playing the game. A dynamic

programming solution to a Stackelberg equilibrium of a partial-information Markov process was proposed. However, this approach required that the pursuit-evasion policies be learned for each new obstacle configuration. To reduce the complexity of a large-scale team, the architecture model needs to be dynamically simplified. Vidal et al. [19] extended and improved this approach to a probabilistic game theoretical framework to control a team of unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs) to pursue a second team of evaders while concurrently building a map in an unknown environment in. To reduce the computational complexity, two computationally feasible greedy pursuit policies: local-max and global-max, were proposed. However, the main limitation of game-theory based approaches is that the methods are extremely computational expensive, which significantly hamper the practical utilization in real-world scenarios.

In this paper, we aim at developing a robust, real-time dynamic equation based multi-robot search approach for multiple targets in a dynamic environment with a rough priori probability map. To cut down the request of system computational time and memory, the environment is partitioned into regions instead of cells. Although a priori probability of targets (i.e., probability map) in each region is assumed to be provided, however, these probabilities may be changed under dynamic environment. Therefore, dynamic updating of the probability map is necessary based on the current sensor information. A dynamic programming equation (DPE) is applied to formalize the utility function for each robot, where not only the probability map and travel costs are considered, but also the interaction between the robots. This utility function would guide the robots' behaviors to achieve an optimal overall performance. Due to the mutual understanding of robots, the dynamic equation based algorithm can achieve higher robustness and fault-tolerance under dynamic environment compared to other heuristic methods.

The paper is organized as followings. Section 2 describes the problem statement. Section 3 explains how the probability map is updated, and an event-triggered system discretization is proposed to reduce the computation complexity. Dynamic programming based utility function is described in Section 4. Simulation and experimental results are given in Section 5. Section 6 concludes the paper and discusses the future work.

## 2. PROBLEM FORMULATION

The searching environment is discretized into  $J$  regions. There are  $N$  homogeneous mobile robots and  $G$  stationary targets, which may be distributed within one region or at different regions. A rough priori *probability map*, which contains the priori probabilities of target existence in each region, is assumed to be provided. When a searching task starts, this probability map will be updated dynamically based on the new perception information of the robots. In terms of capability of each physical robot, the following assumptions are made:

- Each robot can localize itself within the map using an on-board laser range finder and odometry measurements.
- Each robot can detect the targets using an on-board camera system.
- The robots can communicate with each other through wireless communication within the map.
- Each robot can avoid obstacles and plan an optimal global path to the destination.
- The searching task stops when all targets are found.

The searching problem can be stated as follows: given a rough priori probability map, develop an efficient and robust strategy for multiple robots searching for multiple targets in a dynamic environment so that the expected average searching time can be minimized or near-minimized.

### 3. PROBABILITY MAP UPDATE AND EVENT-DRIVEN DYSYSTEM DESCITIZATION

A rough priori probability map (i.e. likelihood distribution) is installed in the robots. When the searching task starts, this probability map is updated dynamically based on the current searching results. The vector of priori probability of targets within each region can be represented as:

$$\tilde{\mathbf{P}}(k|k) = [\tilde{P}_1(k|k), \tilde{P}_2(k|k), \dots, \tilde{P}_J(k|k)]^T, \quad \text{where } \tilde{P}_j(k|k) \in [0,1]$$

where  $M$  is the maximum region number. The prediction of the probability can be estimated based on its previous probability (at time step  $k$ ) and current measurement observed by robot at time step  $k+1$ , which is represented as follows:

$$\begin{aligned} \hat{\mathbf{P}}(k+1|k) &= f(\hat{\mathbf{P}}(k|k), \mathbf{z}(k+1)) \\ \hat{\mathbf{P}}(k|k) &= \tilde{\mathbf{P}}(k|k) \end{aligned} \quad (3.1)$$

where  $\hat{\mathbf{P}}_i(k+1|k)$  represents the vector of probabilities  $[\hat{P}_1(k+1|k), \hat{P}_2(k+1|k), \dots, \hat{P}_J(k+1|k)]^T$  predicted at time step  $k+1$ ,  $f$  represents the update function,  $\hat{\mathbf{P}}(k|k)$  denotes the vector of probabilities at time step  $k$  which are estimated based on previous probabilities.  $\mathbf{z}(k)$  represents the vector of robot observations  $[z_1(k), z_2(k), \dots, z_N(k)]^T$ , which registers target detection or absence at time step  $k$ , where  $z_i(k)$  denotes the observation by robot  $i$  at time step  $k$ , and  $N$  is the number of robots. Let  $P_j^n(k|k)$  denotes the probability of robot  $n$  choosing region  $j$  at time step  $k$ , we have

$$\sum_{j=1}^J P_j^i(k|k) = 1, \quad i = 1, 2, \dots, N. \quad (3.2)$$

To update the probability map, Bayesian measurement update is one straight forward approach, which computes the new probability belief state as a function of robot observation inputs and its former probability belief state. However, this procedure can only be applied to each individual region with new observations from the robot which is searching that region. If we want to propagate this update to other regions, it would be computational intensive. Therefore, a simple update approach is applied in this paper to meet the real-time performance.

Whenever a robot finishes searching region  $i$  at time step  $k$ , where a target may be detected or not, the current probability in this region will be evenly distributed in unsearched regions at next time step  $k+1$ , and the probability of the searched region at next time step  $k+1$  will be set to zero. This procedure would keep the sum of probabilities of unsearched regions always equals to 1. This procedure can be represented as followings:

$$\begin{aligned} \hat{P}_i(k+1|k) &= 0; \\ \hat{P}_j(k+1|k) &= \frac{\hat{P}_j(k|k)}{1 - \hat{P}_i(k|k)}, \quad j \neq i, j = 1, 2, \dots, J \end{aligned} \quad (3.3)$$

where  $\hat{P}_i(k|k)$  represents the estimated probability of a target in region  $i$  at time step  $k$ ,  $\hat{P}_i(k+1|k)$  denotes the predicted probability of a target in region  $i$  at time step  $k+1$ , and  $M$  is the maximum region number within the map.

If the system is discretized at a predefined time interval, the probability map needs to be updated at each time step. However, in a large-scale searching area with low density of targets, the region-based probability map may stay the same most of the time. To reduce the computational complexity, an event-

triggered system discretization is applied in this paper, where the time step and probability map are only updated as necessary instead of at every predefined time step.

Once the searching task starts, the finite state machine of a robot consists of two states: busy and free. The robot state is defined as busy when it is searching inside a region. Otherwise, its state is defined as free. Initially all robots are set as free, and the robots' states are updated dynamically during searching. A new event happens when a robot enters a region or finishes searching its current region, which would trigger the update of the robot states.

The event-trigger is defined as a series of discrete time when each robot changes its state, which can be represented as follows:

$$evt\_trigger = [trig_1, trig_2, \dots, trig_N], \quad (3.4)$$

where  $N$  is the number of robots. Next event will be triggered at  $\min(trig_1, trig_2, \dots, trig_N)$ , which will be accumulated with the total searching time and be subtracted from  $\max(trig_1, trig_2, \dots, trig_N)$ . With this event-triggered discretization, the probability map is only updated at each event. Since the robots only communicate with each other upon new event, the communication overhead can be obviated significantly compared with the fixed-time-interval discretization method.

#### 4. DYNAMIC PROGRAMMING EQUATION (DPE) BASED ALGORITHM

Generally, a utility is defined as a payoff value of a robot selecting a region to search at next discrete time. For a multi-robot system, to improve the collective searching efficiency, the utility of each robot does not only depend on its own payoff value, but also on other robots' decisions.

Obviously, the utility associated with each robot depends on the probability of the target at each region. The higher the probability, the higher the utility value should be. However, this probability-only based approach may try to achieve the most valuable goal (i.e. highest probability of target detection) irrespective of the difficulty of that goal. For example, the agent may skip the nearest region which has lower probability, and go for a very far region which has higher probability but may take much longer time to reach. To build more sensible agents, in this paper, we combine travel cost and utility calculations for each action and then calculate the expected utility.

The set of decisions made by robots from 1 to  $N$  at time step  $k$  is denoted by  $\mathbf{D}(k) = [d_1(k), d_2(k), \dots, d_N(k)]^T$ . The set of probabilities of the target from region 1 to  $J$  at time step  $k$  is denoted by  $\hat{\mathbf{P}}(k|k) = [\hat{P}_1(k|k), \hat{P}_2(k|k), \dots, \hat{P}_J(k|k)]^T$ . The travel cost can be divided into two time-based vectors,  $\mathbf{T}^i(k)$  and  $\mathbf{T}_c$ , where  $\mathbf{T}^i(k) = [t_1^i(k), t_2^i(k), \dots, t_J^i(k)]^T$  represents the time vector that robot  $i$  takes to navigate from its current position at time step  $k$  to different regions from 1 to  $M$ , and  $\mathbf{T}_c = [t_1, t_2, \dots, t_J]$  denotes the set of time required for a robot to cover each region, where  $t_i$  denotes the time required to cover the region  $i$  for each robot. The utility function of robot  $i$  can be defined as a function of  $(\mathbf{D}(k), \hat{\mathbf{P}}(k|k), \mathbf{T}^i(k), \mathbf{T}_c)$ .

First, let us define the payoff value of robot  $i$  searching region  $n$  as follows:

$$g(\mathbf{D}(k), \hat{\mathbf{P}}(k|k), \mathbf{T}^i, \mathbf{T}_c) = \frac{\hat{P}_n(k|k)}{k_1 t_n^i(k) + k_2 t_{cn}}, \quad (4.1)$$

where  $t_n^i(k)$  and  $t_{cn}$  represent the time required for robot  $i$  to navigate from its current position at time step  $k$  to region  $n$ , and the time required for robot  $i$  to cover region  $n$ , respectively.  $k_1$  and  $k_2$  are scale factors which can be adjusted based on different environmental structures. For example, robots may move faster in empty corridors and move slower in the rooms with higher density of obstacles. In this case,  $k_1$  is set greater than  $k_2$ .

To achieve an optimal overall performance, each robot has to take the current decisions of other team members into consideration when it makes its own decision. To resolve this interaction issue, a Dynamic Programming Equation (DPE) is applied to define the utility function for robot  $i$  as follows.

$$U_i(d_1(k), d_2(k), \dots, d_N(k)) = f_i(\mathbf{D}(k), \hat{\mathbf{P}}(k|k), \mathbf{T}^i(k), \mathbf{T}_c) = \begin{cases} 0, & \text{if } P_n(k|k) = 0 \\ g(\mathbf{D}(k), \hat{\mathbf{P}}(k|k), \mathbf{T}^i(k), \mathbf{T}_c) & \text{if } P_n(k|k) = 1 \\ h(\mathbf{D})[g(\mathbf{D}(k), \hat{\mathbf{P}}(k|k), \mathbf{T}^i(k), \mathbf{T}_c) + (1 - P_n(k|k)) \max_n \{f(\hat{\mathbf{D}}(k), \hat{\mathbf{P}}(k|k), \hat{\mathbf{T}}^i(k), \mathbf{T}_c)\}] & \text{otherwise} \end{cases} \quad (4.2)$$

where  $U_i$  represents the utility function of robot  $i$ ,  $\hat{P}_n(k|k)$  represents the estimated probability of target detection in region  $n$  at time step  $k$ .  $g(\mathbf{D}(k), \hat{\mathbf{P}}(k|k), \mathbf{T}^i, \mathbf{T}_c)$  is defined in (4.1).

$(1 - P_n(k|k)) \max_n \{f(\hat{\mathbf{D}}(k), \hat{\mathbf{P}}(k|k), \hat{\mathbf{T}}^i(k), \mathbf{T}_c)\}$  represents the maximum expected utility of robot  $i$  by selecting

different  $\hat{n}$  for the rest of the unsearched regions after finishing the region  $n$ .  $\hat{\mathbf{D}}(k)$ ,  $\hat{\mathbf{P}}(k|k)$ , and  $\hat{\mathbf{T}}^i(k)$  represent the estimated values after robot  $i$  finishes its current searching at region  $n$  at time step  $k$ , and are defined as follows respectively:

$$\hat{\mathbf{D}}(k) = [d_1(k), \dots, \hat{d}_i(k), \dots, d_N(k)]^T$$

$$\hat{\mathbf{P}}(k|k) = \left[ \frac{\hat{P}_1(k-1|k)}{1-\hat{P}_n(k-1|k)}, \dots, \frac{\hat{P}_{n-1}(k-1|k)}{1-\hat{P}_n(k-1|k)}, 0, \frac{\hat{P}_{n+1}(k-1|k)}{1-\hat{P}_n(k-1|k)}, \dots, \frac{\hat{P}_j(k-1|k)}{1-\hat{P}_n(k-1|k)} \right]^T$$

and  $\hat{\mathbf{T}}^i(k) = [\hat{T}_1^i, \hat{T}_2^i, \dots, T_n^i, \hat{T}_j^i]^T$ , (4.3)

The definition of  $\hat{\mathbf{D}}(k)$  assumes that when robot  $i$  is making decision through this recursive procedure, other robots keep their current decision. Considering the situation where several robots may choose same region simultaneously based on their own utility functions, which will definitely decrease the overall searching performance. We define a factor  $h(\mathbf{D})$  as follows:

$$h(\mathbf{D}) = \begin{cases} k_3 * t_n^i(k) / \sum_{j \neq i, j=1}^N t_n^j(k), & j \neq i \\ 1 & otherwise \end{cases} \quad (4.4)$$

where  $t_n^i(k)$  is the travel time that robot  $i$  takes from its current position at time step  $k$  to region  $n$ ,  $\sum_{j \neq i} t_n^j(k)$  is the total travel time that other robots (except  $i$ ) take from their current positions to region  $n$ , and  $k_3$  is a scale factor to be adjusted based on different environmental structures.

The definition of  $h(\mathbf{D})$  actually embeds the coordination between the multiple robots. In other words, cutting down the utility value helps to prevent multiple robots picking up the same region simultaneously, which will eventually improve the overall searching efficiency.

The coefficients  $k_1, k_2$ , and  $k_3$  are gains to adjust the influence of individual factors of the utility function. Various searching behaviors can be achieved by tuning these parameters, which will be discussed in simulation section.

The dynamic programming equation, however, is somewhat intractable for large-scale region numbers. It is possible to approximate the dynamic programming without iterating all possible states. To reduce the computational complexity, we propose one-step dynamic programming solution for Equation (4.2). Basically, the average expected contribution from team members is calculated as the contribution that the team members would make from their current

positions. This approximation is reasonable when each step is relatively small.

In general, the utility is zero if the probability of target detection in region  $n$  is zero. If the probability of the target detection in region  $n$  is 1, it means that there is at least one target left and this is the last region need to be searched. In this case, the utility function is only related to the payoff value by searching region  $n$ . Otherwise, the utility is a recursive function defined in (4.2).

## 5. SIMULATION RESULTS AND DISCUSSIONS

### 5.1 Other Heuristic Searching Strategies

In order to evaluate the searching performance of the proposed dynamic programming equation (DPE) based method, comparison with other heuristic searching strategies is necessary. Several searching strategies are proposed here. First one is called *randomly selection (RS)* approach, where each robot randomly selects the next region to search from the list of unsearched regions without taking into account of prior probability of the target distribution. Second strategy is called *probability-based (PB)* approach, where each robot only picks the region with the highest probability as its next objective region. If more than one region has the same highest probability, the robot randomly picks one from them. The one we proposed is represented by *dynamic programming equation (DPE)* approach. Basically, each robot with DPE strategy defines its own utility function based on (4.2), and picks next searching region with the highest utility value. If more than one region has the same highest utility values, the robot randomly picks one from them.

### 5.2 Simulation Environment and Setups

To evaluate the performance of above approaches, RS, PB, and DPE, simulation experiments are carried out on a *Player/Stage* simulator. *Player/Stage* simulator is used to simulate the control and sensing of the robots.

We carried out our search experiment with a team of two homogeneous Pioneer 3DX mobile robots. Each robot is equipped with a pan-tilt-zoom CCD camera, a laser range finder and one front SONAR ring and one back SONAR ring. The camera sensor is utilized for target detection, the laser range finder is used for the range estimation and robot localization, and SONAR rings are used for obstacle avoidance. There are three targets scattered in the environment, which can be easily identified from their colors. All robots move at a predefined constant speed.

The Adaptive Monte-Carlo Localization algorithm [7] is used for the localization method. Each robot has its

own global path planner, which is wavefront path planner, and local obstacle avoidance algorithms, where the vector filed histogram algorithm is adopted here.

A set of snapshots of one searching experiment using game theory approach in a configuration with 25 regions are shown in Fig. 5.1, where three stationary targets are distributed in different regions. Initially, two robots start from the left corner. The searching task stops when all three targets are detected.

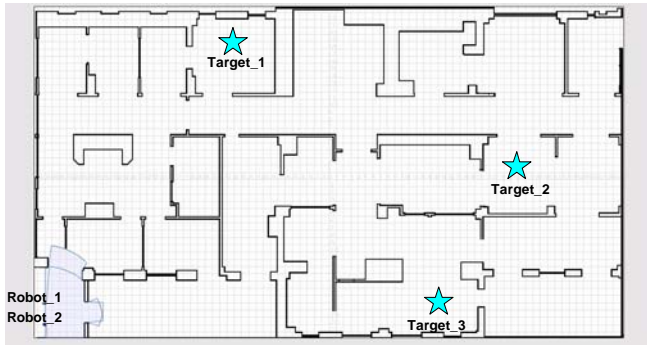


Fig. 5.1: One snapshot of two robots searching for three targets in a 25-region simulation environment using Player/Stage.

### 5.3 Simulation Results

To evaluate the searching performance in a scalable environment, different simulation configurations are applied, where each configuration has different number of regions including 6, 10, 15, 20, 25, 30, and 40. Each searching approach runs 35 times at each configuration. At each configuration, three targets are distributed according to a priori probability map. The simulation results of mean value and the corresponding variance of searching times are shown in Fig. 5.2.

It is obviously that DPE outperforms all other methods. The searching time of DPE is much less than that of PB. The reason for this is because the travel and searching time was ignored in the latter case. RS has the worst performance since neither priori probability map nor the travel/searching time is considered.

In a real world scenario, it is difficult to obtain an accurate priori probability map especially in urban search and rescue situations where the variance may be very large. To explore the system robustness with respect to priori probability map variations, another set of simulations are implemented. 35 runs are conducted for each searching approach, where the target is distributed in the searching regions with the variations of priori probability map ranging from 10% to 50%. Simulation results in the 10-region configuration are shown in Fig. 5.3.

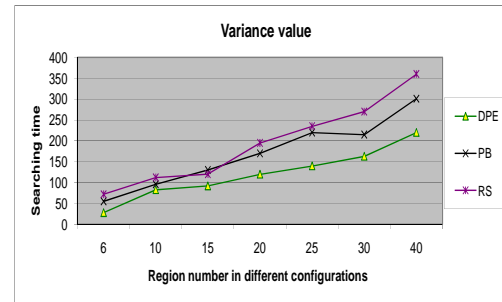
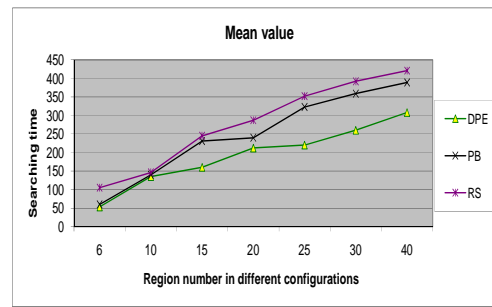


Fig. 5.2: Searching time (mean and variance) with different configurations

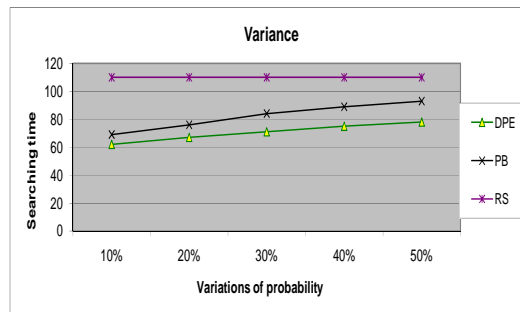
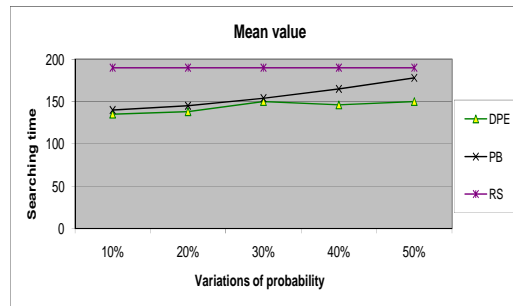


Fig. 5.3: Searching time (mean and variance) with different variations in probability of target distribution in a 10-region configuration

It can be seen that PB is very sensitive to the probability variation since the probability is the only

criteria for the robot to make searching decisions. The probability variation has no effect on the RS at all, which is easily to be understood. The DPE IS much more robust than PB.

When an obstacle is detected on the way to the robot's destination, different approaches would have different behaviors. (1) For PB, the robot has to dynamically re-plan the path to the original higher probability region even though the travel cost is extremely high with the new path; (2) For DPE, a new decision has to be made considering the travel cost, therefore, different region may be selected. To evaluate the robustness of the proposed approaches, 35 simulation runs have been conducted in a 25-region environment, where 5 obstacles are randomly distributed within the searching area at each run. The simulation results are shown in Fig. 5.4. It can be seen that PB is very vulnerable to the environmental change. Although the mean searching time of RS is relatively high, RS is very robust to the environmental change. The DPE approach has much better performance in terms of mean and variance compared to the other two methods.

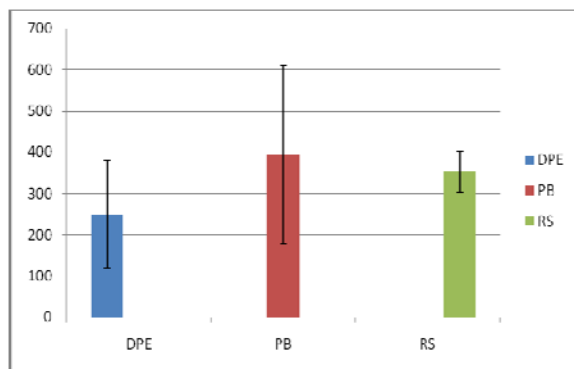


Fig. 5.4: Searching time with different obstacle distributions in a 25-region configuration, where the bars represent the mean values and the error bars represent the variance values.

## 6. CONCLUSION AND FUTURE WORK

A dynamic programming equation based strategic searching approach is proposed in this paper to cooperate a multi-robot system in a multi-target searching task. To take the interaction between the robots into consideration, a dynamic programming equation is applied to estimate the utility function, where not only a priori probability map but also the travel costs are considered, as well as other robots' current decisions. To improve the real-time performance of the proposed approach, two different mechanisms have been proposed in this paper. First, event-driven discretization method is utilized to reduce the unnecessary probability map update and

communication cost. Second, instead of using full-length dynamic equation to calculate the utility, one-step dynamic equation is applied to reduce the computation complexity significantly. Comparing to other heuristic searching strategies, such as random selection and probability-based methods, the simulation results demonstrated that the DPE-based approach has better performance and is more robust to handle the environmental uncertainty.

Although homogeneous robots are utilized in our simulation, the proposed algorithm can easily be extended to the heterogeneous robots with different moving speeds and local sensing capabilities by setting up different travel and covering time for each robot. In the future, we will extend this dynamic equation based approach to a real-world application with physical mobile robots. Since each robot has its own on-board processor and can execute in parallel, so the computational complexity will not be a big issue here.

## REFERENCES

- [1] R. Baeza-Yates, J. Culberson, and G. Rawlins, Searching in the plane, *Information and Computation*, 106:234–252, 1993.
- [2] L. F. Bertuccelli and J. P. How, Robust UAV search for environments with imprecise probability maps, *IEEE Conference on Decision and Control*, 2005.
- [3] L. F. Bertuccelli and J. P. How, Search for dynamic targets with uncertain probability maps, *Proceedings of the 2006 American Control Conference*, Minnesota, USA., 2006, pp. 737-742.
- [4] F. Bourgault, T. Furukawa, and H. F. Durrant-Whyte, Coordinated decentralized search for a lost target in a Bayesian world, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.
- [5] J. N. Eagle, Optimal search for a moving target when the search path is constrained, *Operations Research*, 32(5): 1107-1115, 1984.
- [6] J. N. Eagle, and J. R. Yee, An optimal branch and bound procedure for the constrained path, moving target search problem. *Operations Research*, 38(1):110-114, 1990.
- [7] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. *AAAI*. 1999.
- [8] J. C. Harsanyi and R. Selten, *A general theory of equilibrium selection in games*, the MIT Press, Cambridge, Massachusetts, 1998.
- [9] J. P. Hespanha, M. Prandini, and S. Sastry, Probabilistic pursuit-evasion games: a one-step Nash approach. In *Proc. of the 39<sup>th</sup> conf. on decision and control*, vol. 3, Dec. 2000.

- [10] Y. Jin, A. Minai, and M. Polycarpou, Cooperative real-time search and task allocation in UAV teams, *IEEE Conference on Decision and Control*, 2003.
- [11] M. Y. Kao, J. H. Reif, and S. R. Tate, Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem, *In Proc. 4th ACM-SIAM Sympos. Discrete Algorithms*, pages 441–447, 1993.
- [12] J. R. Kok, M. T. J. Spaan, and N. Vlassis, Multi-robot decision making using coordination graphs, *In Proceedings of the International Conference on Advanced Robotics (ICAR)*, pp. 1124–1129, Coimbra, Portugal, June 2003.
- [13] H. Lau, S. Huang, and G. Dissanayake, Optimal search for multiple targets in a built environment, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'05)*, Edmonton, Alberta, Canada, August 2005.
- [14] S. LaValle, and S. Hutchinson, Path selection and coordination for multiple robots via Nash equilibria, *Proc. IEEE Int. Conf. Robot and Automation*, 1994, pp. 1847-1852.
- [15] A. L'opez-Ortiz<sup>1</sup> and S. Schuierer, S. Online parallel heuristics and robot searching under the competitive framework, *8th Scandinavian Workshop on Algorithm Theory*, Turku, Finland, July 3-5, 2002.
- [16] R. Murphy, Biomimetic search for urbane search and rescue, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, pp. 2073-2078, 2000.
- [17] K. Skrzypczyk, Game theory based target following by a team of robots, *Fourth International Workshop on Robot Motion and Control*, June 17-20, 2004.
- [18] T. J. Stewart, Searching for a moving target when searcher motion is restricted. *Computers and Operations Research*, 6:129-140, 1979.
- [19] R. Vidal, O. Shakernia, H. J. Kim, D. H. Shim, and S. Sastry, Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation, *IEEE Trans. on Robotics and Automation*, Vol. 18, No. 5, Oct. 2002, pp.662-669.