Proceedings of the 2007 IEEE
International Conference on Mechatronics and Automation
August 5 - 8, 2007, Harbin, China

Collision Avoidance in Multi-Robot Systems

Chengtao Cai and Chunsheng Yang
College of automation
Harbin Engineering University
Harbin, 150001, CHINA
caichengtao@163.com

Qidan Zhu and Yanhua Liang
National Research Council Canada
Montreal Road, Ottawa
ON K1A 0R6, CANADA
Chunsheng.Yang@nrc-cnrc.gc.ca

***Abstract* - Multi-robot systems have been widely applied to various applications to perform a given task collaboratively and cooperatively. In a multi-robot environment, path-planning or collision avoidance is an important problem. This paper tackles this important but challenging problem. We developed a step-forward approach for collision avoidance in multi-robot systems, building on the established techniques from omni-directional vision systems, automatic control, and dynamic programming. The developed collision avoidance algorithms can help avoid the collision from any static obstacles and any dynamic objects such as a moving robot. In this paper, we report the developed collision avoidance algorithms, along with simulation-based experimental results. The results show that the collision avoidance strategies are effective and useful for making decisions on collision avoidance in multi-robot systems.**

Index Terms – Multi-robot system, Collision Avoidance, Prediction algorithms.

I. INTRODUCTION

With the rapid development of artificial intelligence, autonomous robots, sensors, automatic control, and computer vision systems, multi-robot systems have been widely applied to various real-world applications such as search and rescue, surveillance, collaborative exploration, and exploration of large and unmapped areas. The objective is to use a team of robots to perform a given task collaboratively and cooperatively. Up to date, there have been a large number of research efforts focusing on different research topics related to multi-robot systems, such as communication techniques among multiple robots [1], behavior-based control [2], vision systems, and path planning [3]. For instance, C. Behring, et al, investigated robot path planning and algorithm complexity [3] using cellular automation simulations. Some researchers worked on sensor techniques to improve sensor reliability [4] and sensor coverage [5].

In the research and development of multi-robot systems, one of the most important issues is path planning with collision avoidance [13]. Because of dynamic characteristics and specificities of multi-robot systems, it is impossible to establish a scheduled path for each robot in performing tasks cooperatively with other robots [12] [14]. A robot in multi-robot systems is one member of a team. Its motion is restricted with other robots and environments given by the performed task. We need an effective dynamic approach for planning the path for a robot in a dynamic environment. There is some work on using centralized algorithms to plan the path for each robot. In such a case, the algorithm can only work for a simple problem that involves two or three robots [6]. Potential field techniques [7] have been proposed for more complex path-planning problems. In [8] [9], Javier Minguez, et al, implemented a single robot with collision avoidance support in troublesome scenarios using the ND (Nearness Diagram) navigation algorithm. In this case, the collision avoidance strategies must rely on sensory information to compute the movement according to unforeseen circumstances. To advance decision-making on collision avoidance for path planning, we have to develop a more effective collision avoidance approach. Without an effective and feasible collision avoidance approach, it will be difficult to plan a safe path for a robot to reach its given goal. In this work, we focus on an approach for multi-robot systems. Building on techniques from sensor, automatic control, and omni-directional vision systems, we developed a step-forward approach. Our algorithms or strategies can help make decisions on collision avoidance; including avoiding multiple static obstacles as well as dynamic objects such a moving robot. To validate the effectiveness and usefulness of these algorithms, we developed a simulation environment for conducting simulation-based experiments in different scenarios. In this paper, we present the collision avoidance algorithms in detail. We also describe the simulation environment and report our experimental results.

The reminder of this paper is organized as follows: Section 2 briefly introduces an environment on multi-robot systems; Section 3 presents the collision avoidance algorithms for multi-robot systems; Section 4 describes the simulation environment and some simulation-based experimental results; Section 5 concludes the paper and discusses future work.

II. THE MULTI-ROBOT SYSTEM ENVIRONMENT

A Multi-robot system is designated to cooperatively perform a given task in various applications. Such a multi-robot system usually consists of multiple robots, a given task, and environment for performing the task. Each robot in the system has to be an autonomous robot, which is equipped with devices or systems such as an omni-directional vision system, a target-identifying system, communication systems, and control systems. These devices or systems provide basic functionalities and abilities for the robot to perform the assigned task in a teamwork environment. In this section, we briefly introduce the techniques for identifying an environment and preprocessing the environmental information around a robot.

A. Identifying environment

To identify the task-performing environment around a robot, an Omni-directional Vision System (OVS) can be used. The

OVS is one of the most important components in an autonomous robot system [10]. It is used as an advanced sensor for detecting the environment around a robot. It provides a 360° view of the environment in a single image. Such an omni-directional image is usually obtained with catadioptric panoramic cameras, which combine conventional cameras (lenses) and convex mirrors. After applying some existing image processing technologies to these images, we can restore the real image and identify the environmental information around a robot. This information includes the position and **ID**s of other robots, the positions of obstacles and other potential dynamic objects. The precision of an omni-directional vision system gradually deteriorates as the distance between its mirror and an object increases. As a result, we define a circular area with a given radius as the vision area in our multi-robot system. The radius of this circle is defined as a vision radius for a single robot. Decisions on collision avoidance will be made based on the information collected within this circular area.

B. Preprocessing environment information

The information obtained from the OVS has to be preprocessed in order to effectively perform two main tasks: identifying static obstacles and identifying dynamic objects.

1). Identifying static obstacles

This task is to identify the position and size of a static obstacle by using the information from OVS. It consists of three steps as shown in Figure 1:

a. Make a circular area for an obstacle in which the diameter equals the length from the top-left point to the bottom-right point of the obstacle. Then, substitute this area for the original obstacle.

b. Check the distance between any two obstacles. If the distance is smaller than the minimum size for a robot to pass through, then combine them into one obstacle, defining a new circular area that covers both of them.

c. Update the obstacle information for decision-making on collision avoidance.

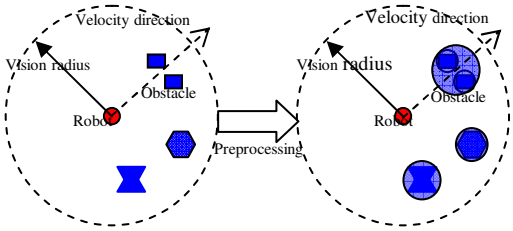


Fig. 1. Preprocessing the static obstacles

2). Identifying dynamic objects

The second task is to identify dynamic objects such as moving robots. Figure 2 shows the process of detecting the velocity (direction and speed) for a moving robot.

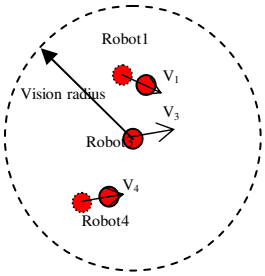


Fig. 2. Detecting the velocity direction

III. COLLISION AVOIDANCE STRATEGIES

In order for a multi-robot system to cooperatively perform a given task in real-world applications, one of the main tasks for each robot in the system is to reach its individual goal without colliding with a static obstacle or other robots. Due to the dynamic environment, it is difficulty to plan a scheduled path for each robot before performing a given task. Traditionally, expert systems have been a useful approach for collision avoidance. For instance, Kose and Yang developed an expert system for ship collision avoidance [11]. However, in a multi-robot system, a robot is controlled by itself and we are unable to obtain expert experience for building a knowledge base for collision avoidance. Considering specificities of multi-robot systems, we propose a novel collision avoidance strategy that we call a step-forward dynamic moving approach. In general, any single robot system must avoid any obstacles based on the information obtained from preprocessing of environmental information. The idea is that a single robot keeps updating its motion path by using recent information. The encountering situation in a multi-robot system can be summarized as 3 cases: an encountering with a static obstacle, an encountering with a dynamic object, and an encountering with both static obstacles and dynamic objects. For these situations, we developed three different algorithms for providing collision avoidance strategies to any single robot in a multi-robot system. In the following subsections, we present these algorithms in detail. The notational conventions used in this paper are summarized in Table Ⅰ.

TABLE Ⅰ
THE NOTATION CONVENTIONS USED IN THIS PAPER

Notation	The mean of the notations
R	A robot's current position.
R_{n1}	The next position following the established path.
R_{no}	The next position for avoiding obstacle.
R_{nr}	The next position for avoiding another robot
O_i	The *i*th obstacle's position.
O_{radius}	The radius of an obstacle.
P	The goal's position
P_R	The priority of Robot R.
P_i	The priority of the *i*th robot.
L	The escape point position.
N_o	The number of dangerous obstacles.
ε	A small positive number.
N_R	The number of dangerous robots.
Q_i	The position of the *i*th dangerous robot.
$Result$	The prediction result. 1 means collision , 0 means safety

A. *The algorithm for avoiding static obstacles*

We assume that the obstacle's radius is smaller than the robot vision radius. Once an OVS captures an image, the relative coordinate information is determined by preprocessing the information as described previously.

Before discussing the collision avoidance strategy, we introduce the some concepts such as *near path*, *far path*, and *escape point*. In Figure 3, the degree of risk for obstacle m exceeds the threshold value of Robot n. So Robot n takes it as a dangerous object. At point **A**, the distance between m and n is less than or equal to a predetermined value. The robot must change its established path to avoid a collision with the obstacle. For a single obstacle and a single goal, there exist two tangent points, shown as point **B** and **C.**

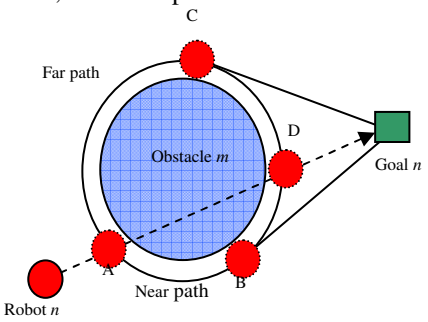


Fig. 3. The motion path for avoiding static obstacle

We call point **B** and **C** the escape points. Since arc **AB** is shorter in length than arc **AC**, arc **AB** is defined as the near path and arc **AC** as the far path. To keep a shorter path to the goal, the robot should follow arc **AB** to avoid the obstacle. When the robot reaches the escape point **B**, there are two possible paths to follow: one is to return to the previously established path, i.e., first move to the point **D**, then follow the original established path to move toward its goal; another possibility is to re-plan the path toward its goal. Apparently, the newly established path is shorter. Using these concepts, we developed an algorithm for avoiding static. The algorithm is shown in Table Ⅱ. In the algorithm, ε is a given small number, which is tunable and determined by the requirements of a given task.

TABLE Ⅱ

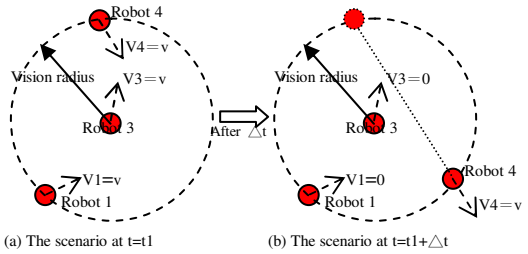
THE ALGORITHM FOR AVOIDING STATIC OBJECTS

Input: ε, R, O_i, N_o, L, P, O_{radius}
Output: R_{nl}, R_{no}

Process:
If ($|P-R|>\varepsilon$) *do {*
  For (all the obstacles $\in N_o$*){*
    If ($|R-O_i|>\varepsilon+O_{radius}$) *do* $\{R \leftarrow R_{nl}\}$ // *moving a step following the //established path.*
    Else If ($L==R$) *do* $\{R \leftarrow R_{nl}\}$ // *moving a step following the //established path.*
    Else do $\{ (R \leftarrow R_{no})\}$ } // *moving a step for avoiding an obstacle*
  }
 }
 Else do $\{R \leftarrow P\}$

B. *The algorithm for avoiding dynamic objects*

In this study, a dynamic object is defined as a moving robot. As we mentioned, each robot has its own unique **ID**. Other robots in the system can identify this **ID** number. Now we take Robot 3 as an example to illustrate the strategy for avoiding a dynamic object. Figure 4(a) shows that Robot 3 encounters two robots, Robot 4 and Robot 1. In order to avoid the deadlock among robots, we adopt the concept of priority and use the robot ID number as the robot's priority. When two robots encounter each other, the lower priority one must take an action to avoid the higher priority one, while the higher priority robot keeps its own moving state unless it encounters a new object or obstacle. The action that the lower priority robot will take could be "stopping" or "speed reduction". Here, the lower priority robot takes a "stopping" action to avoid a collision. In Figure 4(a), because Robot 3 identifies two target robots: Robot 1 and Robot 4, Robot 3 takes a "stopping" action to avoid Robot 4. Meanwhile, Robot 1 takes a stopping action as well to avoid Robot 3. As a result, only Robot 4 keeps moving with its current states. After Robot 4 gets out of the vision area of Robot 3, Robot 3 and Robot 1 start to move following their established path as shown in Figure 4(b).



(a) The scenario at t=t1 (b) The scenario at t=t1+△t

Fig. 4.The motion strategy without prediction

Based on the collision avoidance strategy, Robot 3 and Robot 1 stop their motion to avoid Robot 4. It would be much preferred if these three robots could keep moving without taking "stopping actions" , because "stopping" requires a complicated control system and the robots are delayed in the execution of their task. To this end, we propose to predict other robot's motion before making a decision. As showing in Figure 4(a), Robot 3 first predicts the other robots' motion state based on current moving states, including velocity direction and value in a given time. If the two robots will encounter each other, they will rigorously follow the priority strategy. Otherwise, they will keep moving in current motion states. Figure 5 shows a prediction procedure from Robot 3. Because each robot is homogenous in a multi-robot system, the prediction procedure of Robot 4 and Robot 1 is similar to Robot 3's.

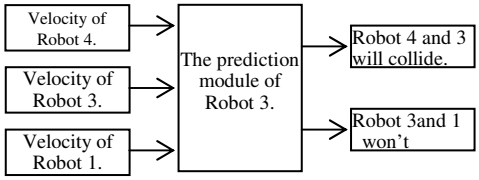


Fig. 5. The prediction procedure of Robot 3

Based on the predicted results from Robot 3, the moving states of three robots are shown in Figure 6. Because Robot 4 and Robot 3 will encounter each other and the priority of Robot 3 is lower than Robot 4's one, Robot 3 will stop and Robot 4 will keep moving. Meanwhile, since Robot 3 and Robot 1 will not encounter each other, Robot 1 will keep moving as shown in Figure 6(b) even through Robot 1 is identified in Robot 3's vision area. After some time Δt, Robot 3 predicts that Robot 4 and Robot 3 will not collide, so Robot 3 will start to move as shown in Figure 6(c). By predicting the other robot's moving status, the robots reduce the frequency of taking a "stopping" action to avoid the higher priority robot. The algorithm for avoiding dynamic objects is shown in Table Ⅲ.

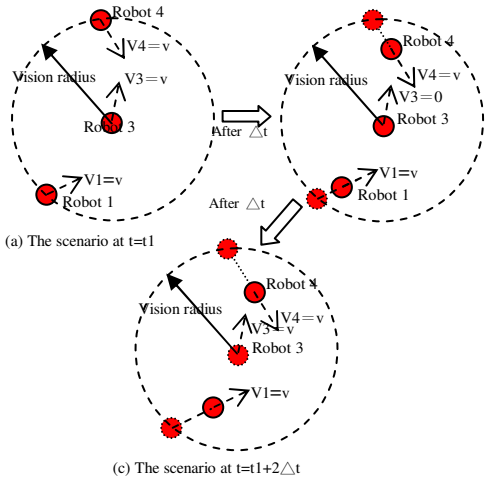


Fig. 6. the motion strategy with prediction

TABLE Ⅲ

THE ALGORITHM FOR AVOIDING DYNAMIC OBJECTS

Input: ε, R, O_i, N_R, L, P, Q_i,P_R, P_i
Output: R_{nl}, R_{nr},*Result*

```
Process:
If (|P-R|>ε) {
    If(all the robots, i ∈N_R,P_R==max(P_i))do{
     If(|Q - R| >ε) do {R←R_nl}   // moving a step following the
                                  // established path.
     Else do { R←R_nr }}          // moving a step for avoiding
                                  // other robots.
  Else{
    Result=predictingTargets(R, Q_i)!  // making prediction
      If (Result=1) do {R←R}       // taking a "stop" action for
                                   //avoiding other robot
      Else do {{R←R_nl}}}          // moving a step following the
                                   //established path.
  Else do {R←P}
```

C. The algorithm for avoiding static obstacles and dynamic objects

In practice, a robot in a multi-robot system needs to avoid not only static obstacles but also dynamic objects at the same time. We need a collision avoidance strategy for avoiding both static obstacles and dynamic objects. In this case, the lower priority robot will be considered as a static obstacle from the viewpoint of the higher priority robots. On the other hand, the higher priority robot will keep moving regardless of the prediction results. After combining the algorithms for avoiding static obstacles and dynamic objects discussed above, we have an algorithm for avoiding both static obstacles and dynamic objects. This algorithm is shown in Table Ⅳ.

TABLE Ⅳ

THE ALGORITHM FOR AVOIDING STATIC OBSTACLES AND DYNAMIC OBJECTS

Input: ε, R, O_i, N_R, L, P, Q_i,P_R, P_i, O_{radius}
Output: R_{nl}, R_{nr}, R_{no} ,*Result*

```
Process:
If (|P-R|>ε) {
 If(|R-O_i|>ε+O_radius)do{R←R_nl}    //moving a step following the
                                     // established pat.
   Else do{If(L==R) do {(R←R_nl)}    //moving a step following the
                                     // newly established path
        Else do { (R←R_no)}}}        // moving a step for avoiding
                                     // an  obstacle
If (|P-R|>ε) {
If (N_R!=0){
   Result=predictingTargets(R, Q_i)!  // making prediction for other
                                      //robots
     If (Result=1){
    If(all the robots, i ∈N_R,P_R==max(P_i)){
       If(|Q- R|>ε) do {R←R_nl}      //moving a step following the
                                     //established path
      Else do {If(L==R) do (R←R_nl)} //moving a step following
                                      // the newly established
path
           Else do {{R←R_nr}}}        // moving a step for
                                      //  avoiding other robots
     Else do { R←R }}
    Else do {R←R_nl}}                // moving a step following
                                     // the established path
   Else{R←R_nl}}}                    //moving a step following
                                     // the  established path
Else{R←P}
```

Ⅳ. SIMULATION-BASED EXPERIMENTS AND RESULTS

In order to validate the effectiveness and usefulness of the collision avoidance strategies for multi-robot systems, we developed a simulation environment for conducting experiments for collision avoidance in multi-robot systems. As we introduced in Section 2, all robots are autonomous and homogenous robotic systems, which are equipped with OVS and other devices for identifying the environment around them. Our simulation environment was developed using the Java programming language. Using this simulation environment, we conducted different experiments for validating the collision avoidance algorithms that we discussed in Section 3. Within this environment, we can set different multi-robot system parameters. For instance, we can specify the number of robots, the positions of static obstacles, the position of each robot's goal, and the initial positions for each robot. Figure 7 shows an example of a multi-robot system in which there are 5 robots, 10 static obstacles, and 5 different goals corresponding to each robot. As shown in Figure 7, the simulation environment is a two-dimension space of 600 pixels by 800 pixels. The size of a single robot is 30 pixels, and all robots in the system have the same speed of 5 pixels/per second. For simplifying the simulation process, all robots start from the left side and move

toward their goals on the right hand side. All static obstacles have different sizes varying from 20 pixels to 60 pixels and are randomly placed in the environment.

We present some of these experiments and results in the next sub-section.

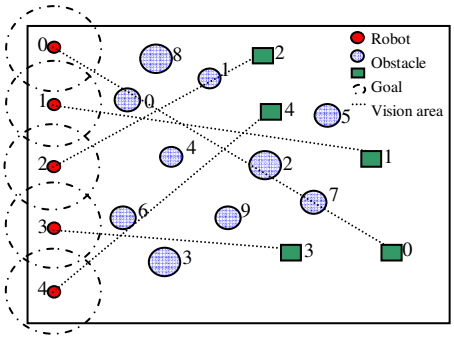


Fig. 7. An example of a multi-robot system

A. *Experiments of a single robot and static obstacles*

Figure 8 shows the simulation result for collision avoidance in the case of a single robot and multiple static obstacles. The robot started from the top-left corner. The goal is set near the bottom-right corner. The robot encountered 3 static obstacles on its way to the goal. At the points **A**, **B**, and **C**, the robot took the near path as determined by the collision avoidance strategy to avoid these obstacles. When the robot arrived at these escape points, it rescheduled a new path for collision avoidance. From the trajectory of the simulation, it is obvious that the robot successfully avoided the static obstacles and reached its goal safely.

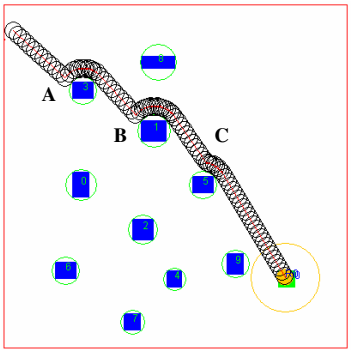


Fig. 8. The simulation result of a single robot and several static obstacles

B. *Experiments of collision avoidance for dynamic objects*

Figure 9 shows the simulation result for collision avoidance in the case of multiple dynamic objects. In this scenario, there were 5 robots in the system. They were positioned on the left side, while 5 goals corresponding to each robot were positioned on the right side. The five robots started at the same time. In area **M**, Robot 0, Robot 1, and Robot 2 encountered each other, so each robot had to take an action to avoid the other robots, in this case without predicting the moving status of other robots. After each robot performed the prediction for its own target robots, they found there was no dangerous collision. As a result, none of them needed to take a "stopping" action. However, in area **N**, Robot 3 and Robot 4 encountered each other. The prediction result shows that they would have a collision. As determined by the algorithm, Robot 3 has priority to take an action to avoid Robot 4. As a result, Robot 3 stopped a while waiting for Robot 4 to pass. The final trajectories of each robot in this scenario are shown in Figure 9.

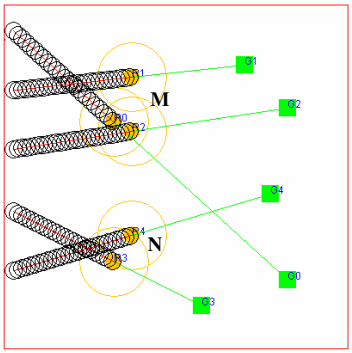


Fig. 9. The simulation result of 5 robots without static obstacles

C. *Experiments of collision avoidance for multiple robots and static obstacles*

Figure 10 shows the simulation results for a scenario in which there were 5 robots and 10 static obstacles. In this complex scenario, a robot needs to avoid not only collision from the static obstacles but also from other moving robots in the system. Each robot performed the collision avoidance strategies very well. As shown in Figure 10, Robot 0 avoided obstacle 3 and took a stopping action to avoid Robot 1, and then move to his goal (G0). Robot 1 took an action to avoid obstacle 3 and 8. Robot 2 took an action to avoid obstacle 1. Similarly, Robot 4 has highest priority and only needed to take an action to avoid obstacle 6 and obstacle 2. But Robot 3 took a stopping action to avoid Robot 4 and another action to avoid obstacle 4. From the simulation result, we can see that all robots reached their goals successfully without any collision with either static obstacles or other robots.

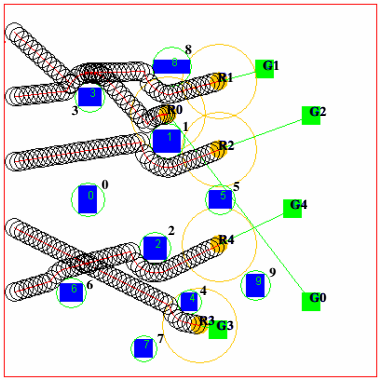


Fig. 10.The simulation result of multiple robots and static obstacles

V .CONCLUSIONS AND FURTHER WORK

In this paper, we presented the developed collision avoidance strategies for multi-robot systems. The proposed collision avoidance approaches rely on the techniques from omni-directional vision sensor systems that are able to identify the position of static obstacles and the velocity of other robots.

To validate the effectiveness and usefulness of the developed collision avoidance algorithms or strategies, we also developed a simulation environment. Using this simulation environment, we conducted experiments for different scenarios. The simulation-based experimental results showed that these algorithms are effective and useful for collision avoidance in multi-robot systems. They help make decision on collision avoidance from any static obstacles or any dynamic objects.

Although we implemented the collision avoidance algorithms in a simulation environment, we still need to validate these algorithms in a physical multi-robot system. Since the control method, mechanics of robot motion and environment detection precision may be different in realistic multi-robot systems, the parameters in the developed collision avoidance algorithms have to be tuned. Therefore, our future work is to conduct validation for the developed algorithms in a real multi-robot system.

ACKNOWLEDGEMENT

This work was done when the author, Chengtao Cai, was working with National Research Council Canada as a Guest Worker. The authors would like to thank all staff at Integrated Reasoning Group of Institute for Information Technology. Special thank goes to Bob Orchard for his English proofreading and valuable suggestions on revising the paper.

REFERENCES

[1] Balch, T. and Arkin, R. C.: "Communication in Reactive Multiagent Robotic Systems," *Autonomous Robots*, vol. 1 No. 1, pp. 27–52, 1995.

[2] Mataric, M.: Behavior-based Control, "Main Properties and Implications," *In Proceedings of IEEE International Conference on Robotics and Automation, Workshop on Intelligent Control Systems, Nice, France*, pp. 46–54, 1992.

[3] Behring, C., Bracho, M., Castro, M. and Moreno, J. A., "An Algorithm for Robot Path Planning with Cellular Automata," *In Proceedings of the Fourth International Conference on Cellular Automata for Research and Industry, Springer-Verlag*, pp. 11–19, 2000.

[4] Parker, L. E., "Distributed Algorithms for Multi-robot Observation of Multiple Moving Targets," *Autonomous Robots*, vol. 12 No. 3, pp. 231–255, 2002.

[5] Balch, T. and Arkin, R.C. "Behavior-based Formation Control for Multi-robot Teams*," IEEE Transactions on Robotics and Automation*, vol.14. pp. 926–939, 1998.

[6] Schwartz J. T. and Sharir, M. "On the Piano Movers Problem: Coordinating the Motions of Several Independent Bodies," *Int. J. Robot. Res.*, vol. 2, pp. 46–75, 1983.

[7] Tournassoud, P. "A Strategy for Obstacle Avoidance and Its Application to Multi-robot Systems," *In IEEE Int. Conf. on Robotics and Automation, San-Francisco, CA*, pp. 1224–1229, 1986.

[8] Javier Minguez, Luis Montano, "Nearness Diagram（ND）Navigation: Collision Avoidance in Troublesome Scenarios," *IEEE Transactions on Robotics and Automation*, vol. 20: pp. 45-59, 2000.

[9] W.Feiten, et al, "Robust Obstacle Avoidance in Unknown and Cramped Environment," *In Proc. IEEE Int. Conf. Robotics and Automation, San Diego, CA*. pp. 2412-2417, 1994.

[10] Jos´e Gaspar, et al, "Vision-based Navigation and Environmental Representations with an Omni-directional Camera," *IEEE Transactions on Robotics and Automation*, vol. 16. pp. 890-898, 2000.

[11] Kose, K., Yang, C., et al, "A Collision Avoidance Expert System for Integrated Navigation System and Its Brush-Up," *Journal of Society of Naval architects of Japan,* vol. 177, pp. 399-407, 1995.

[12] Zhang, F., Tan D. A new real-time and dynamic collision avoidance method of mobile robots based on relative coordinates, Robot Vol. 25. pp.31-34 ,2003

[13] Xu, T., TANG Z., Research on obstacle avoidance planning for mobile robots in dynamic world. Robot Vol. 25. pp.117-122,2003

[14] Zhao, F., CUI P., et al, Approach to collision avoidance plan for robot in dynamic environment. Journal of Kunming University of Science and Technology (Science and Technology) Vol. 28. pp.64-66, 2003