A decentralized collision avoidance algorithm of two mobile robots using potential fields

Dong-Hoon Yang* and Suk-Kyo Hong**

*Department of Electronics Engineering, Ajou University, Suwon 442-749, Korea.
(Tel: +82-31-219-2489; Fax: +82-31-212-9531; Email: yanh@ajou.ac.kr)

**Department of Electronics Engineering, Ajou University, Suwon 442-749, Korea.
(Tel: +82-31-219-2489; Fax: +82-31-212-9531; Email: skhong@ajou.ac.kr)

Abstract: A new collision avoidance algorithm is presented for two mobile robots in narrow corridor environments. When two robots meet each other in a narrow corridor, one should yield the way to the other robot. To solve the problem arising in this situation, they exchange their path to get information about crossing-points to check avoidance conditions, which are necessary for choosing the robot to yield. The conditions are summarized as follows. 1) If one robot blocks the path to the closest crossing-point in front of the other robot. 2) If the closest crossing-point of each robot is the same point. 3) Which robot is closer to the closest crossing-point. In this paper, we propose a path planning algorithm for the robot which yield the way. Simulation results are presented to verify the feasibility of the proposed collision avoidance algorithm.

Keywords: mobile robot, path planning, potential field, collision avoidance

1. Introduction

Path planning algorithms and motion control methods of multiple mobile robots have been studied recently. Mobile robots will be used in various areas and they will have many tasks such as exploration and mapping of unknown or partially known environments [6][4], search and rescue [5], carrying large objects [3]. To carry out their tasks, one of the important functions that they should have is to move to the goal from the initial position. Many path planning algorithms have been proposed for mobile robots to reach their goal position such as roadmap methods, cell decomposition methods, potential field methods, and so on [1]. Obstacle avoidance algorithms are also required for mobile robots because there can be obstacles in environments [7]. Researches on moving obstacle avoidance algorithms have been conducted [8]. This method includes time as one of the dimensions of the model world.

When multiple mobile robots move in an environment, they can avoid each other if they have sensors and avoidance algorithms unlike moving obstacles. Therefore they can carry on tasks in an environment. Multiple robots can accomplish some tasks that a single robot cannot. Performance benefits (e.g. time, energy, etc.) can be gained using multiple robots.

Planning schemes for multiple robots are divided into group architectures such as centralized and decentralized. The behavior of decentralized systems have several advantages over centralized architectures such as fault tolerance, natural exploitation of parallelism, reliability and scalability [9]. Decentralized architectures are divided into distributed and hierarchical architectures. Centralized architectures have a single agent. The other agents receive orders and execute them. In decentralized architectures, there is no agent like in centralized architectures.

Multiple robot systems in both architectures may need to communicate to exchange some information with each other. In order to exchange their information or commands multiple mobile robot can use communication or sensors. When multiple robots carry a large object they can communicate by detecting force without telecommunication [10]. In this paper two mobile robots use telecommunication to exchange their paths when they meet while they move to the goal.

There are several planning algorithms in decentralized planning. In prioritized planning one robot is considered at a time, a path of one robot is created in the configuration space [11]. Stationary obstacles and moving obstacles (other robots) are taken into account during path planning of the one robot. The prioritized planning approach would have great difficulty in solving the planning problem in Fig. 1. Robot1 plans its path first without considering Robot2, Robot2 should create its path with considering Robot1. But Robot1 blocks the path of Robot2, so Robot2 cannot find the path. In path coordination method [12] the trajectory coordination problem is considered as a scheduling problem, like job-shop scheduling, where space is the shared resource. Each of path is generated independently first, and then two paths are coordinated so that the robots do not collide with each other.

In this paper we propose a collision avoidance algorithm for two mobile robots in an environment that has narrow corridors. This algorithm makes the robots avoid safely when the robots meet each other in narrow corridors. An algorithm that finds the spots for avoiding and plans the path to the spot is proposed.

This paper is organized as follows. The following sec-
2. Problem Statements

Let $A$ be a single robot as a rigid object and it moves in a Euclidean space $W$, called workspace, represented as $R^N$, with $N=2$. Each individual robot is described as $A_i$. A configuration $q$ of $A_i$ is a specification of the position $T$ with respect to Cartesian frames. The configuration space of $A$, represented as $A(q)$, is the space $C$ of all the configurations of $A$. Each robot starts from its initial configuration and moves to its own goal configuration. The environment has a couple of corridors and crossing-points, which are too narrow and small for the two robots to pass over each other without collision. When the two mobile robots meet in a narrow corridor, one should move to a spot which is not in the path of the other robot in order to avoid collision. There are a lot of cases that two robots meet, so the robots should determine which car should give way in each case. The cases are summarized as follows. Fig. 2 shows various situations that the robots can meet.

(a) $A_1, A_2$ move in the opposite direction. There are crossing-points behind each robot, not between them.
(b) $A_1, A_2$ move in the opposite direction. There are crossing-points between them.
(c) $A_1, A_2$ would move in the same direction. There are crossing-points between them.
(d) $A_1, A_2$ move in the opposite direction. Only one robot has crossing-points behind.

In Fig.1.(a), both robots have a crossing-point behind them, so they both can yield the way to each other. It is more reasonable that robot where is closer to its crossing-point yield the way. In Fig.1.(b), one should move to the side path between them first. After the other robot goes past the crossing-point, the one should go to its goal. In Fig.1.(c), they would use the same path, one should wait for the other to go and be apart enough from it, then it moves to the goal. In Fig.1.(d), it should be considered whether one blocks the way other robot. The robot in side mid-path can block the path of the robot in bottom. Therefore, the one in bottom must go to the right side of the crossing-point. After the other goes past the crossing-point, the one can go along its path.

3. Collision Avoidance of Two Mobile Robots

3.1. Path Planning Using Potential Field

There have been proposed a few path planning algorithms using potential fields. The potential maps created by those planning algorithms may have local minima. The improved navigation function was proposed to solve the local minimum problem [1][2].

In this algorithm, the grid potential is used. The configuration is discretized into a rectangular grid. $GC$ represents discretized configuration. This algorithm uses the skeleton of an environment to find safe way to the goal. Fig. 3 shows the skeleton of the environment. The skeleton $S$ is composed of the configurations that lie halfway between obstacles.

After the goal is linked with the skeleton, the potential of the goal is assigned by zero. $L^1$ distance (Manhattan distance) is used in computing the potential of the remaining configurations in the skeleton. The potential $U$ of the neighbors of the goal corresponds to the $L^1$ distance of the configurations. These two steps will be used in making the paths for avoiding.

To make the robot at the start position move to the skeleton, the potential $U$ of the besides of the skeleton in free space should be assigned. These potentials are assigned by using $L^1$ distance from the skeleton. The potential maps created by this function have no local minima. A path from any start position to the goal can be created by the best-first planning.

3.2. Collision Avoidance Algorithm

Each robot moves to its goal using the path that was planned by improved navigation function. Fig. 4 shows the collision avoidance behaviors of $A_1$ when it meets $A_2$. To choose the behaviors of robots, several factors should be taken into account. $nc$ expresses the nearest crossing-point of the robot.
1. Whether one blocks the path to the nc of the other.
2. Whether nc is same to nc or not, where nc is the nearest crossing-point of A.
3. Which robot is closer to the nc.

When the robots meet each other, they exchange their paths and find several parameters and flag to determine their behaviors. These are in Fig. 7. nc1 = 1, nc2 = 1 means there is no crossing-point in remaining paths of A, A2. If the environment has only one corridor, this case can be occurred but it is assumed that the environment in this paper has two crossing-points.

Fig. 4 presents the determination of driving modes according as which robot blocks the path of the other robot. This means the goal of the one robot is in the path of the other robot. If both robot do not block each other, whether crossing-points exist or not in the remaining paths should be checked. By exchanging the paths, the robot can get information of the other.

Fig. 5 shows the selection of the driving modes according as there are same crossing points in their remaining paths or not. When they have the same crossing-point, the behavior of the robot is determined by the relations of the configurations just before and after the crossing-point. Condition 1 presents that the two robots would exchange their configurations as Fig. 2(b). Condition 2 shows that a robot would move to the corridor where the other robot is and the other robot would move to the third way that both robot would not go along like Fig. 2(d). Condition 3 is the opposite case to condition 2. Condition 4 presents that the two robots would go along the same way when they meet. Condition 5 means that one robot get close to the other robot that went passed the crossing-point. These five conditions are not same rank.

Condition 2 is checked when the condition 1 is not satisfied. The other conditions are checked when high rank conditions are not satisfied.

Fig. 6 shows the procedure after the mode was determined as the avoiding behavior. The robot moves along the path that was generated by the path planning process for avoidance until it reaches the spot for avoiding. Then it moves to the nearest crossing or the saved configuration or the same crossing-point with the other robot's. And, the robot moves along the path that was generated by the improved navigation function again.
$d_1$: if this is 1, the goal of $A_2$ is in the path of $A_1$.
$d_2$: if this is 1, the goal of $A_1$ is in the path of $A_2$.
$nc_1$: if this is 1, there is no crossing-point in the remaining path of $A_1$.

$dm$: driving mode
$n$: robot moves along the path created by improved navigation function.
$a$: robot moves to the spot for avoiding.
$r$: robot moves back to the saved configuration.

same crossing: if this is 1, there are same crossing-points between the two robots.
same config.: if this is 1, there are same configurations in the paths of the two robots.

$bci$: the configuration just before the nearest crossing-point of the $A_i$.
$aci$: the configuration just after the nearest crossing-point of the $A_i$.
$dc_i$: the distance between the nearest crossing-point and the configuration of the $A_i$.
$nc_i$: the nearest crossing-point of $A_i$.

Fig. 7. Parameters and flags to determine behaviors

The procedure of generating the path for avoiding is shown in Fig. 8. This procedure is similar to the process of making the potential map in the improved navigation function. Since the robot is assumed to be on the path, the potential of the configurations in outside of the skeleton do not have to be assigned. The potential is assigned on the current configuration as zero.

4. Simulation Results

Simulation results show that the proposed collision avoidance algorithm can be applied to various situations that two robots meet in a narrow corridor. Four cases of encounter of two robots are chosen to inspect the proposed algorithm.

Fig. 8 shows the case that two robots meet each other in a narrow corridor. There is no crossing-point in their fronts. The upper robot in Fig. 8(a) moves backward and goes to the different way from the way that the lower robot would use until reaching the spot for avoiding. Then, the robot moves back to the crossing-point and moves along the original path.

In Fig. 8(b) two robot meet each other and there is a crossing-point between them. Hence, they can use this crossing-point to avoid each other. The robot that is closer to the crossing-point moves to the spot for avoiding. While the robot moves to the spot, the other moves along the its path at a regular distance. The robot at the spot comes back to the crossing-point, and goes along the original path.

Fig. 8 presents that two robot would move along the same path after a crossing-point when they meet each other. In this case, the robot that is closer to the crossing-point has the priority. Therefore, this robot moves first along the path. While this robot is getting closer to the crossing-point, the other robot moves back not to collide. And this robot follows the robot that has the priority at a regular distance.

1. $k=1,2,...,spot_b=q; q \in S$ and $d_1(q, nc) = \alpha$
2. Find the paths $P_{nc}$ from $spot_b$ to $nc$.
3. Find the spot $spot_a$, for avoiding of the path; $spot_a \leftarrow spot_b; \forall j, P_{nc}(j) \cap P_{other} = \emptyset$ ($P_{other}$: path of the other robot)
4. Compute potential $U_{nc}$
   $\forall q, U_{nc}(q) \leftarrow M (M is a large number.)$
   $U_{nc}(spot_a) = 0$
   $Q \leftarrow spot_a (Q is a queue of configurations)$
   While $\exists U_{nc} \leq \alpha$
      $\forall q' \in Q$
      $For all neighbors q' of q in S$
      
      if $U_{nc}(q') \neq M$ and $U_{nc}(q') = M$
      
      $U_{nc}(q') \leftarrow U_{nc}(q) + 1$
      $Q \leftarrow q'$
      
   end
5. Find the path using best-first planning

From the current configuration to the $spot_a$

In Fig. 8 the goal of the upper robot is in where the robot of bottom is. If the upper robot moves to the crossing-point, the path of the other robot is blocked up. Therefore, the robot of the bottom should move to the spot for avoiding first regardless of the distance between the current configuration and the crossing point. While this robot moves to the spot, the other robot should move along its original path.

Through these simulations, this collision avoidance algorithm can makes the two robots avoid each other using cross-points.

5. Conclusions

We proposed a collision avoidance algorithm for two mobile robots in an environment that has narrow corridors. The conditions for the determination of the robot that should go to the spot for avoiding were summarized. The path to the spot can be generated by the proposed planning algorithm. Capability of the proposed collision avoidance algorithm was shown through the simulations about four cases.

Environment does not have only narrow corridors. There can be wide corridors that two robots can pass each other without collision. Collision avoidance algorithm for the environments that have narrow and wide corridors will be developed.
Fig. 9. Two robots meet each other in a narrow corridor. There is no crossing-point in their fronts. The upper robot moves backward and goes to the different way from the way that the lower robot would use until reaching the spot for avoiding. Then, the robot moves back to the crossing-point and moves along the original path.

Fig. 10. There is a crossing-point between two robots. The robot that is closer to the crossing-point moves to the spot for avoiding. While the robot moves to the spot, the other moves along the its path at a regular distance. The robot at the spot comes back to the crossing-point, and goes along the original path.

Fig. 11. Two robot would move along the same path when they meet each other. In this case, the robot that is closer to the crossing-point moves first along the path. While this robot is getting closer to the crossing-point, the other robot moves back not to collide. And this robot follows the other robot at a regular distance.

Fig. 12. The upper robot blocks the robot in the bottom corridor. Therefore, the robot of the bottom should move to the spot for avoiding first regardless of the distance between the current configuration and the crossing point. While this robot moves to the spot, the other robot should move along the its original path.
References