Effective Methods for Generating Collision Free Paths for Multiple Robots based on Collision Type (Demonstration)

Fan Liu, Ajit Narayanan, Quan Bai
School of Computing and Mathematical Sciences
Auckland University of Technology
Auckland, New Zealand
{richard.liu, ajit.narayanan, quan.bai}@aut.ac.nz

1. COLLISION TYPES IN MULTI-ROBOT SYSTEMS

Collision avoidance is an important topic in multi-robot systems. Existing multi-robot pathfinding approaches ignore sideswipe collisions among robots (i.e., only consider the collision which two agents try to occupy the same node during the same time-step) [1, 3, 4], and allow diagonal move between two adjacent nodes (e.g., Figure 1(b)). However, in many real world applications, sideswipe collisions may also block robots' movements or cause deadlocks. For example, as shown in Figure 1, if the size of two robots is as big as the grid size they occupied, collisions will happen not only between robots R1 and R2 in the situation depicted in Figure 1(a), but also that in Figure 1(b), which is typically not considered as a collision in existing multi-robot systems.

Figure 1: (a) Occupy the same position. (b) Sideswipe collision.

To overcome the limitation depicted in Figure 1(b), we investigate all possible collision scenarios in a multi-robot system (the speed / velocity of robots is taken into consideration when describing these collisions) when robots are moving, and identify one deadloop type and five collision types. Other collision types involving non-movement of robots due to breakdown are not included in our scenarios. We claim that all possible scenarios that may hinder a robot's planned motion in a two-dimensional space can be covered by these collision / deadloop types (with symmetry). The scenario that may cause a deadloop situation in a multi-robot system, on the other hand, is depicted in Figure 2. The five collision types are head-on, front sideswipe, rear sideswipe, front-end swipe and front-end sideswipe, which are illustrated from Figure 3(a) to (e), respectively. Front sideswipe (Figure 3(b)) and rear sideswipe (Figure 3(c)) can occur only on diagonal moves for both robots.

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), June, 4–8, 2012, Valencia, Spain.

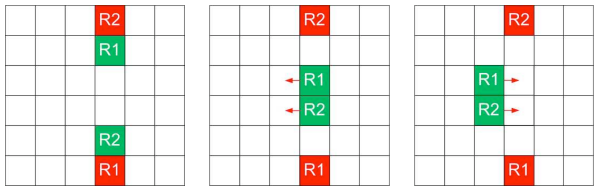

Figure 2: Illustration of deadloop. The green square and the red square are the robot positions and the goal positions for two robots, respectively. R1 and R2 are robot 1 and robot 2. (a) The initial position for two robots. (b) and (c) The dead looping condition is encountered and repeated in-between (b) and (c) infinitely as each robot makes a move that mirrors the other robot's.

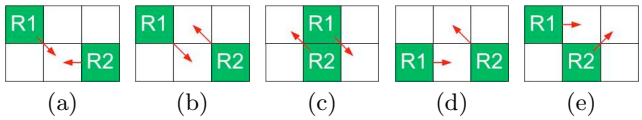

Figure 3: Illustration of 5 collision types. (a) Head-On. (b) Front Sideswipe. (c) Rear Sideswipe. (d) Front-End Swipe. (e) Front-End Sideswipe.

2. ROBUST COLLISION AVOIDANCE STRATEGY

In this work, we also propose a coordinator-based (centralized) strategy to coordinate robots' movements. The strategy repeats a 'plan-evaluate-move' process to plan robots' routes and avoid potential collisions. In details, collision avoidance is achieved through the following steps.

1. Each robot computes its optimal path by using a classical path finding algorithm, i.e., the A* algorithm [2].
2. Each robot reports to the coordinator its current node position, previous node position, intended node position and estimated distance remaining to the goal node.
3. The coordinator detects potential deadloop and collisions based on robots' intentions (i.e., the nodes the robots want to move to). If no collision or deadloop is detected, goto Step (6), otherwise goto the next step.
4. Robots with potential collisions or deadloop will use the Super A* algorithm, which is described in Algorithm 1, to replan their routes in a decoupled manner.

Robots with less remaining distance will get higher priority for path planning purposes.

5. Robots will then report their re-planned intentions to the coordinator. Repeat Step (3)-(5) until no collision or deadloop can be detected by the coordinator.

6. Each robot moves to their intended node. If the goal node is achieved, then the algorithm is stopped for that robot. Otherwise, go back to Step (1) and repeat Step (1)-(6) until all robots achieve their goals.

Algorithm 1 Super A* Algorithm

Input: Input two nodes n_0 (Start Node), n (Goal Node) and L (Robot Label)
Output: Output a set of nodes $N_{close}, n_i \in N_{close}$

```
1: N_open ← n_0, N_close ← ∅, flagged ← false
2: loop
3:    pn ← Compute the lowest cost of node, in N_open
4:    if pn = n then
5:       N_close ← N_close ∪ {pn}
6:       Return N_close
7:    else
8:       for all neighbours n_new do
9:          if L accords with a fixed priority scheme then
10:            flagged ← Check Deadloop and five Collision types
11:         end if
12:         if flagged is true then
13:            Continue Loop
14:         end if
15:         if n_new ∈ N_open and G_new < current G then
16:            G ← G_new
17:            Continue Loop
18:         end if
19:         if n_new ∈ N_close and G_new < current G then
20:            G ← G_new
21:            Continue Loop
22:         end if
23:         N_open ← N_open ∪ n_new
24:      end for
25:   end if
26:   N_close ← N_close ∪ {pn}
27: end loop
```

3. DEMONSTRATIONS

We have conducted both real robot and simulator-based simulations. Through these simulations, we try to evaluate the following three aspects of the proposed strategy: (1) Practicability: is the strategy relevant and applicable to real robot systems? (2) Solvability: can the strategy find valid collision-free multi-robot paths? (3) Optimality: is the strategy able to generate the best paths despite collision-avoidance behaviour? Experiments were carried out using different configuration environments. The proposed method is applied to a two-robot system with the deadloop and collision conditions described in Section 1. The video link is http://youtu.be/gEHRxpbD_LY.

3.1 Demo 1: Real Robot Application

The goal of the first demo is to demonstrate the applicability of our approach to real robot systems. This experiment has been carried out using the Rovio robots of WowWee Technologies. The task of the robots is to find the optimal / shortest path and move from their initial positions to their goal positions without collision. In the demo, the robots are deployed on two sides of the 5x5 grid and have to move to their goal positions on the other side using the proposed strategy avoiding deadloop (Figure 2) and collisions. The demo shows that one robot moves away from its optimal path given the initial situation. The path of one robot is changed to avoid collisions between each other. For collision avoidance, one robot has to take a detour around the other and then return to its planned path as quickly as possible. At the end, the two robots achieve their goal positions. This is the resolution to this possible collision without introducing a sideways move or a collision involving front-end swipe or sideswipe, given that the robots are as big as the nodes they occupy.

3.2 Demo 2: Simulation

In demo 2, three scenarios are simulated in a 10 by 10 grid. The first scenario demonstrates deadloop and all five collision types, with dynamic path replanning demonstrated according to steps 1-6 above. It can be seen that R1 lets R2 take the optimal path, and R1 selects an avoidance strategy that allows it to return to the optimal path after collision is avoided. In addition, the proposed strategy can also cater for a combination of possible collisions. For instance, if a dynamic change to one or more robots' goal states leads to a deadloop condition, the proposed strategy 1-6 can resolve the problem effectively. The second scenario shows the tunnel-like environment, where two robots need to pass through a tunnel to reach their goals. R1 gets to the tunnel first, so R1 gets the priority to go through the tunnel. This is an example of allocating priority based on time. That is, the proposed strategy not only considers the optimal path cost but also takes optimal time cost into account. Finally, the last scenario shows that, with randomly changing goals in real-time, the proposed strategy is capable of avoiding collisions and returning to the planned optimal path for the new goal nodes. A 50x50 grid for 20 and 50 robots, respectively, with 10% obstacle density randomly generated environment.

4. CONCLUSION AND FUTURE WORK

In real-world multi-robot systems, deadloop and collision types must be clearly identified and managed to ensure that robots reach their destinations as optimally as possible. The proposed strategy, according to our real-world and simulated experiments, is robust and able to handle the deadloop and collisions effectively. We have also shown that the strategy is capable of dealing with both static and dynamic obstacles, and allows robots to resume their planned paths after collision avoidance. In future work, we will design a decentralized approach which can allow robots to achieve peer-to-peer communication for collision avoidance as well as investigate optimality preservation in more detail.

5. REFERENCES

[1] M. Jansen and N. Sturtevant. Direction maps for cooperative pathfinding. In *AIIDE*, pages 185–190, October 2008.

[2] N. J. Nilsson. *Principles of Artificial Intelligence*. Springer Publisher, Berlin, New York, 1982.

[3] D. Silver. Cooperative pathfinding. In *AIIDE*, pages 117–122, June 2005.

[4] T. Standley. Finding optimal solutions to cooperative pathfinding problems. In *AAAI*, pages 173–178, July 2010.