Decentralized Path Planning for Multi-Agent Teams in Complex Environments using Rapidly-exploring Random Trees

Vishnu R. Desaraju and Jonathan P. How

***Abstract*— This paper presents a novel approach to address the challenge of planning paths for multi-agent systems operating in complex environments. The algorithm developed, Decentralized Multi-Agent Rapidly-exploring Random Tree (DMA-RRT), is an extension of the Closed-loop RRT (CL-RRT) algorithm to the multi-agent case, retaining its ability to plan quickly even with complex constraints. Moreover, a merit-based token passing coordination strategy is developed to dynamically update the planning order based on a measure of each agent's incentive to replan, derived from the CL-RRT. Agents with a greater incentive plan sooner, yielding a greater reduction of the global cost and greater improvement in the team's overall performance. An extended version of the algorithm, Cooperative DMA-RRT, allows agents to modify others' plans in order to select paths that reduce their combined cost and thus further improve global performance. The paths generated by both algorithms are proven to satisfy inter-agent constraints, such as collision avoidance, and a set of simulation and experimental results verify performance.**

I. Introduction

Autonomous robotic systems continue to be called upon to perform a multitude of tasks. As with humans, teams of autonomous agents are able to complete multiple tasks in parallel, and moreover, are capable of performing far more complicated tasks than a single agent operating alone. Consider, for example, an automated warehouse as shown in Fig. 1. The team of autonomous agents seeks to manage the flow of inventory efficiently, performing tasks such as taking inventory and moving items. However, agents must also avoid collisions while executing these tasks. This highlights an obvious but fundamental requirement for virtually any multi-agent system: the ability to safely navigate the environment in which the tasks must be performed.

This work examines the problem of planning dynamically feasible paths through complex environments for a team of autonomous agents traveling to specified goal locations. The agents are assumed to have constrained, nonlinear dynamics, as seen in any practical system. As a result, agents may be incapable of following paths generated by simple path planners, such as piecewise-linear paths, making dynamic feasibility a crucial requirement. Agents must also be able to navigate safely in cluttered or otherwise complex real-world environments. Furthermore, these paths must satisfy all constraints on interactions between agents, the most common example being collision avoidance. In general, teams are assumed to consist of many agents operating over a large area, precluding the use of centralized planners to satisfy these requirements. Finally, paths must be selected to minimize a given cost metric, such as travel time or fuel consumption, across all agents in the team.

V. R. Desaraju, Research Assistant, Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA 02139, USA, rajeswar@mit.edu

J. P. How, Richard C. Maclaurin Professor of Aeronautics and Astronautics, Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA 02139, USA, jhow@mit.edu

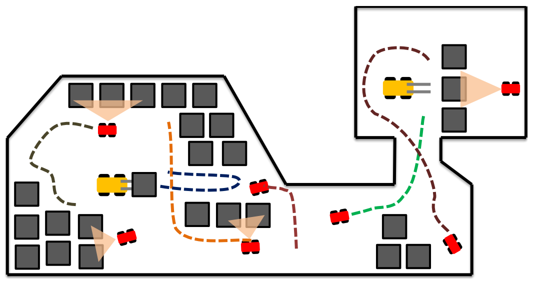


Fig. 1. A team of autonomous forklifts and ground-guidance robots operating in a constrained warehouse environment

A. Related Work

Numerous path planning algorithms have been developed for autonomous systems. Discrete search methods [1,2] have been used extensively for robot path planning, but typically do not scale well to large problems with complex agent dynamics and environments. Mixed Integer Linear Programming (MILP) [3] and Model Predictive Control (MPC) [4] retain the benefits of optimization and can account for agent dynamics more easily. However, these approaches also scale poorly with the number of constraints imposed by the complex dynamics and environment. Potential field methods have also enjoyed considerable success in simple scenarios, but have difficulty in complex environments [5].

Sampling-based motion planners, such as Probabilistic Roadmap (PRM) methods [6] and the Rapidly-exploring Random Tree (RRT) algorithm [7], have gained popularity in recent years. While the original RRT algorithm is a fast way to plan paths in complex, high-dimensional spaces, the real-time RRT algorithm proposed by Frazzoli [8] and extended by Kuwata et al. as Closed-loop RRT (CL-RRT) [9] is much more useful as an online path planner. As such, the work presented here employs the CL-RRT algorithm as the underlying path planner. The reader is encouraged to refer to references [9] and [10] for details on the CL-RRT algorithm.

Decentralization is also essential for large-scale multi-agent planning problems. Decomposing the problem into a set of single agent problems greatly reduces the complexity of each problem and enables the use of single-agent path planners to solve these smaller problems. However, some form of coordination between the decision makers is required in order to have global feasibility across all individual plans.

Scerri et al. [11] propose a deconfliction strategy in which the team must reach consensus on path feasibility for any updated path. While this reduces the amount of local knowledge required to check for path feasibility, it also introduces considerable delay into the planning process. Another notable approach is proposed by Purwin et al. [12] where agents reserve regions of the map to move in and must reach consensus on any modifications to these regions. This allows for some asynchronous planning at the expense of more conservative conflict avoidance based on these large regions. Reachability based planning approaches [13]–[15] have been studied for collision avoidance along intersecting paths. However, these methods are typically concerned with non-cooperative agents and thus consider worst-case scenarios in general. Trodden and Richards propose a version of Decentralized MPC (DMPC) where one agent replans in each iteration, while all others continue their previous plans [16]. This reduces the synchronization required but maintains a fixed planning order, which is often inefficient.

This paper introduces the Decentralized Multi-Agent Rapidly-exploring Random Tree (DMA-RRT) algorithm as a practical multi-agent path planning strategy. It combines the fast, online planning capabilities of the CL-RRT algorithm with a new a merit-based token passing coordination strategy to generate paths that guarantee all inter-agent constraints are satisfied and improve overall team performance. A cooperation strategy is also presented to further improve team performance. These performance improvements and safety guarantees are verified through simulations and experiments performed in the MIT RAVEN facility [17].

II. Decentralized Coordinated Planning

The most straightforward approach to decentralized, multi-agent path planning is to allow all agents to continuously plan their own paths subject to constraints imposed by the other agents' paths. While this has the benefit of being fully asynchronous, it may very quickly lead to conflicting plans if one agent modifies the constraints while another is planning.

A. Merit-based Token Passing Coordination Strategy

Thus a coordination strategy is needed to ensure that the decisions taken by individual agents do not conflict. The strategy in Trodden and Richards [16] requires minimal synchronization between agents to plan in real-time and also guarantees that all inter-agent constraints are fixed while an agent is replanning. However, agents still cycle through a fixed planning order, regardless of whether an agent will see any substantial benefit from replanning. An extension to this coordination strategy is presented here in the form of *merit-based token passing*. This new approach relies on a measure of Potential Path Improvement (PPI) that reflects an agent's incentive to replan, i.e., its expected improvement in path cost if allowed to update its plan next.

Rather than cycling through agents in order, a token is used to identify which agent can update its plan at each planning iteration. Each agent without the token computes its PPI and broadcasts it as a bid to be the next token holder. When the current token holder is finished replanning, it passes the token to the agent with the best bid, i.e., the greatest PPI. Ties are broken by random selection. This produces a dynamic planning order where agents that may benefit the most from replanning are able to do so sooner, without having to wait for agents that have little incentive to replan. Agents that quickly find paths to their goals will typically generate higher bids more often, allowing them to act on these plans sooner and reach more of their goals.

1) PPI from RRT: Computing the PPI requires the agent to compare costs between its current plan and the new, better plan. In general, this would require computing a new path just to check the PPI. However, the tree of feasible paths maintained by the CL-RRT simplifies this: the difference in cost between the agent's current path and the best path in the tree is its PPI. If the best path in the tree has a much lower cost than the current path, the PPI would be large and it would be beneficial to replan soon to select the better path. The merit-based token passing strategy using this PPI computation is a core component of the multi-agent path planning algorithm described in the next section.

B. Decentralized Multi-Agent RRT

A few key assumptions are made in developing the Decentralized Multi-Agent RRT (DMA-RRT) algorithm. The agents are assumed to form a fully connected, lossless network with negligible delays. Each agent has a model of both its own and the other agents' dynamics. The environment is assumed to be known and only contain static obstacles. Inter-agent constraints are assumed to be symmetric between agent pairs (i.e., agent i satisfying a constraint with agent j implies j satisfies the constraint with i). For simplicity, this paper focuses on the collision avoidance constraint.

The DMA-RRT algorithm consists of two components. The *individual component* handles the path planning, while the *interaction component* handles all information received from other agents.

1) Individual Component: The first component, described in Algorithm 1, embeds the CL-RRT algorithm. The agent is initialized with some dynamically feasible path, p_0, that satisfies all constraints for all time. This initial path need not reach the goal and may be as simple as a stopped state.

One agent is initialized as the token holder, with all others initialized to $Have\_Token \leftarrow false$. Each agent grows the CL-RRT and identifies the best path, p_k^*, in the tree according to its cost function. The merit-based token passing strategy then determines if the agent will update its plan to p_k^* and pass the token to $winner$ (the agent with the best bid), or if it will bid to be the next token holder.

If the agent updates its plan, the constraints imposed on the other agents must also be updated accordingly. Due to the closed-loop nature of the RRT, any plan can be characterized by a sparse set of waypoints and closed-loop dynamics. Thus, it is sufficient for the agent to broadcast its new waypoints to allow the others to update constraints. This update occurs in the interaction component.

Algorithm 1 DMA-RRT: Individual component

```
1: Initialize with $p_0, k = 0$
2: $Have\_Token \leftarrow false$ except for one randomly selected agent
3: while Agent is active do
4:     $k \leftarrow k + 1$
5:     Grow CL-RRT, identify best path $p^*_k$ satisfying all constraints
6:     if $Have\_Token$ then
7:         $p_k \leftarrow p^*_k$
8:         $winner \leftarrow$ agent with best $bid$
9:         Broadcast $waypoints$ of $p_k$ and $winner$ to all agents
10:        $Have\_Token \leftarrow false$
11:    else
12:        $p_k \leftarrow p_{k-1}$
13:        $bid \leftarrow (p_k.cost - p^*_k.cost)$
14:        Broadcast $bid$
15:    end if
16: end while
```

Algorithm 2 DMA-RRT: Interaction component

```
1: while Agent is active do
2:     Listen for messages
3:     if received message with $waypoints$,$winner$ then
4:         Simulate other agent's trajectory, update constraints
5:         if agent is $winner$ then
6:             $Have\_Token \leftarrow true$
7:         end if
8:     end if
9:     if Received $bid$ message then
10:        Update sender's $bid$
11:    end if
12: end while
```

2) Interaction Component: The second component of the DMA-RRT algorithm, described in Algorithm 2, manages interaction between agents. Agents communicate via two messages. The first identifies the token $winner$ and has a list of $waypoints$ defining an updated plan, while the second is the bid (PPI value) for the token.

When the $waypoints$ and $winner$ message is received, the agent must update the constraints on its plans. Rather than receiving full trajectory information, the agent can simulate the other agent's trajectory along the received waypoints using a model of its dynamics. The CL-RRT can then treat this as a time-parameterized obstacle to ensure the path satisfies the inter-agent constraints. When the token $winner$ receives this message, it can assign itself the token and begin replanning with the new constraints. When a bid message is received, it is added to the list of bids to check after the next plan update. In addition, the assumptions on the network allow all agents to receive all messages sent, maintaining data consistency across the team.

C. Path Feasibility

A key property of the CL-RRT is that any path it returns is dynamically feasible and satisfies all constraints [9]. So a new constraint is added for the end of the path. For a trajectory $p_k = \{x(k), x(k+1), \dots, x(k+N)\}$ of length $N+1$ timesteps, the agents' states must satisfy $\{x(k+M) \in \mathcal{X}_f | \forall M > N\}$ where the invariant set $\mathcal{X}_f$ does not violate any constraints. Examples include a stopped state or loiter pattern, assuming it is feasible [18]. Then an agent's behavior is well-defined and feasible even at the end of its path, allowing others to plan accordingly. This leads to the following theorem:

Theorem 1: Given a set of n cooperative agents and a set of inter-agent constraints satisfying the assumptions above, if the initial set of paths $\{p_{i,0} | \forall i, i = 1, \dots, n\}$ satisfies all constraints, then using the DMA-RRT algorithm, the set of all future paths $\{p_{i,k} | \forall i, k \geq 0\}$ will satisfy all constraints.

Proof: Assume the set of all agents' paths $\{p_{i,k} | \forall i\}$ at planning iteration k, satisfies all constraints. Then at iteration $k+1$, agent j (token holder) updates its path $p_{j,k+1} = p^*_{j,k+1}$ while all others keep their paths $\{p_{i,k+1} = p_{i,k} | \forall i \neq j\}$. Since $p_{j,k+1}$ is from the CL-RRT, it satisfies all constraints. As a result, the updated set of constraints imposed on other agents remains consistent with their existing plans (symmetric constraints assumption), and $\{p_{i,k}\}$ satisfying all constraints implies that $\{p_{i,k+1}\}$ satisfies all constraints for $i \neq j$. Thus, the set of paths $\{p_{i,k+1} | \forall i\}$ at iteration $k+1$ satisfies all constraints. Since, by assumption, $\{p_{i,k} | \forall i\}$ satisfies all constraints for $k = 0$, then by induction, $\{p_{i,k} | \forall i\}$ satisfies all constraints for all $k \geq 0$. ■

Algorithm 3 Cooperative DMA-RRT: Individual component

```
1: Initialize with $p_0, k = 0$
2: $Have\_Token \leftarrow false$ except for one predetermined agent
3: while Agent is active do
4:     $k \leftarrow k + 1$
5:     Grow CL-RRT ignoring other agents' paths, identify best path $p^*_k$
6:     if $Have\_Token$ then
7:         if $p^*_k$ conflicts with some $agent\_j$ then
8:             Check emergency stops (Algorithm 4)
9:         end if
10:        Identify emergency stop nodes on $p^*_k$
11:        $p_k \leftarrow p^*_k$
12:        if $agent\_j$'s plan was modified then
13:            $winner \leftarrow agent\_j$
14:        else
15:            $winner \leftarrow$ agent with best $bid$
16:        end if
17:        Broadcast waypoints of $p_k$ (with emergency stops) and $winner$
18:        $Have\_Token \leftarrow false$
19:    else
20:        $p_k \leftarrow p_{k-1}$
21:        $bid \leftarrow (p_k.cost - p^*_k.cost)$
22:        Broadcast $bid$
23:    end if
24: end while
```

D. Cooperative DMA-RRT

Although the DMA-RRT algorithm implements a coordination strategy, each agent only minimizes its own cost when selecting a plan, and this does not necessarily minimize the the total cost across all agents. Thus, a cooperation strategy is introduced that allows an agent to modify its own path and that of another agent to minimize the combined cost. The Cooperative DMA-RRT algorithm enables this by adding *emergency stop* nodes along each agent's path where it could safely stop if requested to by another agent. The decision to terminate another agent's path is based on a cost comparison.

1) Individual Component: The modified individual component is presented in Algorithm 3, with changes in red. When $agent\_i$ selects a plan p^*_k, it also identifies several nodes in the plan where it could stop without violating any

Algorithm 4 Emergency Stop Check

```
1: for all viable emergency stop nodes $N_l$ in $agent\_j$'s path do
2:     Assuming $agent\_j$ stops at $N_l$, find last safe stop node in $p_k^*$
3:     $TotalCost_l$ = stop node cost + $agent\_j$'s cost to stop at $N_l$
4: end for
5: Select terminal node and $N_l$ that minimize $TotalCost_l$
6: if $N_l$ is not $agent\_j$'s original terminal node then
7:     Send $estop$ message to $agent\_j$ to stop at $N_l$
8: end if
9: if selected terminal node is not the original terminal node then
10:     $p_k^*$ pruned past new terminal node
11: end if
```

Algorithm 5 Cooperative DMA-RRT: Interaction component

```
1: while Agent is active do
2:     Listen for messages
3:     if received $waypoints$,$winner$ messages then
4:         Simulate other agent's trajectory, update constraints
5:         if agent is $winner$ then
6:             $Have\_Token \leftarrow true$
7:         end if
8:     end if
9:     if Received $bid$ message then
10:         Update sender's $bid$
11:     end if
12:     if Received $estop$ message then
13:         Terminate $p_k$ at node stop specified in $estop$
14:     end if
15: end while
```

constraints, and these nodes are marked in the plan that is broadcast. As with the terminal node, no other agents can select paths through these areas, unless they do so before $agent\_i$ is expected to arrive. These nodes already have costs associated with them from the CL-RRT, so other agents know the cost of forcing this agent to stop at one of these nodes.

The test for conflicts with other agents is done outside the CL-RRT. If the path $agent\_i$ selects does not conflict with another agent, the algorithm proceeds normally. However, if it conflicts with $agent\_j$, the Emergency Stop Check in Algorithm 4 is performed.

For each of $agent\_j$'s emergency stop nodes, $agent\_i$ identifies the last node in p_k^* where it can stop. Since $agent\_i$ knows the costs of its own nodes and $agent\_j$'s costs for stopping early, it can select the combination of terminal node and emergency stop node that yields the best total cost across both agents. If this requires $agent\_j$ to stop early, an $estop$ message is sent indicating where it should stop. Then $agent\_i$ passes the token to $agent\_j$ so it can update its plan.

2) Interaction Component: The interaction component is modified (Algorithm 5) such that when an agent receives an $estop$ message, it prunes all nodes in its current path p_k after the node specified in $estop$.

3) Path Feasibility: The addition of this coordination strategy preserves the guarantee from Theorem 1. By construction, all emergency stop nodes satisfy the same requirements as terminal nodes. Thus, any path terminated at an emergency stop node satisfies the terminal node requirement. Like the terminal nodes, emergency stop nodes are known to all agents, ensuring no other plans will conflict with them.

4) Limiting Case: Consider the case where all nodes are emergency stops. This further constrains the environment, producing more conservative plans, but allows agents to stop nearly anywhere. Purwin et al. [12] describe an approach where agents cooperate to define non-intersecting regions that are safe for all time, enabling some asynchronous planning. Generalizing the emergency stop nodes to stop regions would capture some of this functionality. Moreover, emergency stops are only blocked after the agent is expected to arrive, avoiding excess cooperation checks.

III. Results

Several simulation and experimental results are presented below, demonstrating the functionality and performance of the DMA-RRT and Cooperative DMA-RRT algorithms. To maintain the decentralized nature of the algorithm, each agent is simulated on a different computer (Intel 2.13 GHz dual-core desktops with 1 GB RAM), with all the computers on the same local area network. Each agent runs a real-time Java implementation of the DMA-RRT (or Cooperative DMA-RRT) components. The agents use skid-steer bicycle models with a pure-pursuit controller [9]. CL-RRT path cost is given by travel time along the path. Two scenarios tested using this decentralized simulation setup are described below.

A. Scenario: Ten Agents on Open Map

The first scenario has a team of ten agents on an open map. Though there are no static obstacles, the density of agents on this map increases the complexity of the environment through which they must plan. Figure 2(a) shows the initial agent positions and goal locations, and agents cycle through the list of goals for ten minutes.

1) DMA-RRT without Merit-Based Token Passing: This scenario is first run *without* merit-based token passing. Instead, agents use a round-robin approach, cycling though a fixed planning order. Fig. 2(b) shows a snapshot of the world as seen by agent 1 (blue circle with red arrow). Yellow circles with blue arrows denote the other agents, and the current token-holder's arrow is highlighted. The magenta dots are the waypoints along agent 1's current path. The time-parameterized obstacles produced by the other agents' trajectories are shown by the red-to-green gradients. Red indicates the other agents' current positions, and green indicates positions at least ten seconds in the future. The gray areas are the safe terminal nodes for each path.

On average, each agent reached a total of 12.5 goals in ten minutes. The fixed planning order is evident in regular replan start times in Fig. 3(a).

2) DMA-RRT: The test is repeated with the full DMA-RRT. Adding the intelligent coordination policy yielded significantly better results: each agent reached an average of 15.1 goals in ten minutes. The dynamic planning is also apparent from a comparison of the replan start times in Fig. 3(b).

3) Analysis: This simulation scenario demonstrates the DMA-RRT's ability to handle large teams of agents. With the round-robin strategy, all agents have equal opportunities to plan. If an agent finds a new path after passing the token, it is forced to wait for the token to pass through the entire

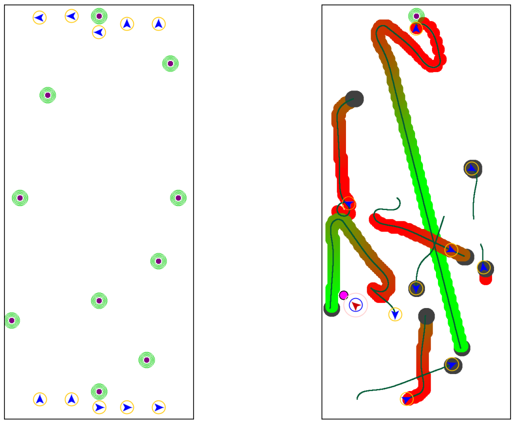

(a) Map layout (b) State of world seen by agent 1

Fig. 2. Ten Agents on Open Map scenario

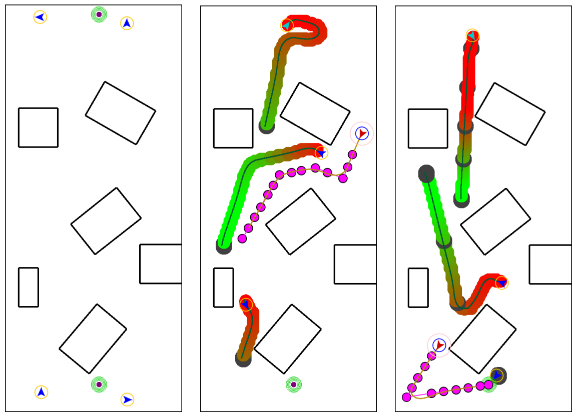
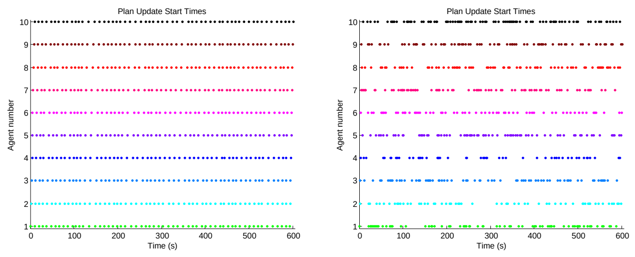

(a) Map layout (b) State of world as seen by agent 1 (c) With emergency stops (gray areas)

Fig. 4. Four Agents with Obstacles scenario



(a) Using round-robin approach (b) Using merit-based token passing

Fig. 3. Replan start times (Ten Agents on Open Map)

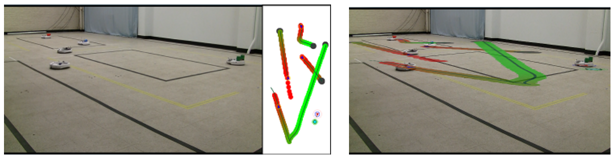

(a) Snapshot of vehicles and paths (b) Overlay of paths on testbed

Fig. 5. DMA-RRT in RAVEN

team before receiving it again. As Fig. 3(a) shows, the wait time is typically around ten seconds for this scenario. In a more complex scenario with additional agents and obstacles, this delay would only increase.

In comparison, merit-based token passing allows agents to regain the token quickly after passing it, as seen in Fig. 3(b). Thus, agents that are able to update to shorter paths can do so much more quickly, reducing the travel time between goals and increasing the number of goals that can be reached in a fixed amount of time. The simulation results reflect this quite clearly, with a 20% increase in the average number of goals per agent, from 12.5 to 15.1.

B. *Scenario: Four Agents with Obstacles*

The second scenario is in a complex environment with four agents, several static obstacles, and two goal locations, as shown in Fig. 4(a).

1) DMA-RRT without Merit-Based Token Passing: Again, the round-robin approach is tested first. Fig. 4(b) shows a snapshot of agent 1's knowledge of the world during a sample run. On average, each agent reached a total of 11.9 goals in ten minutes with this approach.

2) DMA-RRT: The same scenario is then run with the full DMA-RRT. In this case, even with merit-based token passing, each agent averages 11.9 goals in ten minutes.

3) Cooperative DMA-RRT: Finally, the emergency stop cooperation strategy is enabled for this scenario. The stop nodes are placed approximately every four seconds along each trajectory, as seen in Fig. 4(c). With emergency stops enabled, each agent averaged 13.0 goals in ten minutes.

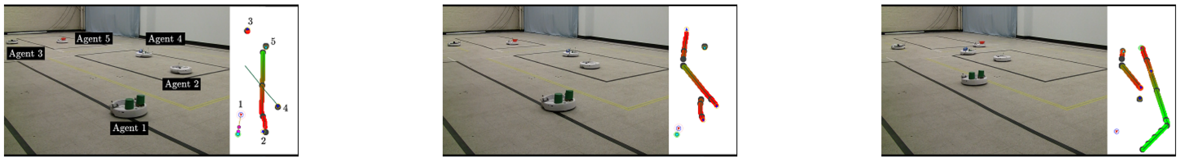
4) Analysis: For a team of only four agents, the merit-based token passing strategy is not expected to provide a significant improvement in performance. With so few agents, each receives the token almost as regularly as in the round-robin case. Since each agent's ability to update its path is not significantly altered, the overall performance does not improve.

However, introducing the emergency stop logic does make a difference due to the narrow passages formed by the obstacles. These passages are difficult to plan through, and agents often stop in or near the openings. Therefore, an agent that is able to find a path through one of the openings can use the cooperation strategy to prevent other agents from obstructing this path. Furthermore, the 95% confidence intervals for the average goal counts do not overlap – a strong indicator that the emergency stops provide a statistically significant improvement in performance.

C. *Experimental Results*

Both the DMA-RRT and Cooperative DMA-RRT were tested in the MIT RAVEN facility [17] using a team of five iRobot Create vehicles. The open map with ten goals was used in both cases. A snapshot from a run of the DMA-RRT algorithm is shown in Fig. 5. Again, the red-green gradients indicate when that area is expected to be occupied. The overlay shows the physical locations of these time-parameterized obstacles.

Fig. 6 shows one instance of the emergency stop logic activating. Agent 2 plans a path ending short of its goal due to agent 5. Agent 4 then receives the token and finds a path to its goal but identifies a conflict with agent 2. Since paths that reach the goal are preferred, agent 4 sends an emergency



(a) Agent 4 find a path to goal that conflicts with agent 2

(b) Agent 2 terminates its path early at agent 4's request, and agent 4 proceeds to the goal

(c) Agent 2 reaches the end of its plan and waits for agent 4 to pass

Fig. 6. Emergency stop logic in action

TABLE I
MINIMUM DISTANCE (M) BETWEEN AGENTS (DMA-RRT)

Agent	1	2	3	4
5	0.720	0.554	0.436	0.732
4	0.586	0.417	0.476	-
3	0.365	0.505	-	-
2	0.509	-	-	-

TABLE II
MIN DISTANCE (M) BETWEEN AGENTS (COOPERATIVE DMA-RRT)

Agent	1	2	3	4
5	0.343	0.344	0.528	0.312
4	0.564	0.641	0.441	-
3	0.313	0.308	-	-
2	0.565	-	-	-

stop message to agent 2 to terminate its path at its first stop node. Agent 2 complies, receives the token from agent 4, and updates to a plan that allows it to stop closer to its goal without interfering with agent 4's path. The agents continue along these paths, with agent 2 waiting at its terminal state for agent 4 to pass.

Table I shows the minimum distance between agents during a five minute run of the DMA-RRT algorithm. The distance between two agents placed side by side is 0.30 m. The minimum separation between each pair of agents is greater than this threshold for all combinations of agents, indicating that the collision avoidance constraint is satisfied. Table II shows similar results for Cooperative DMA-RRT.

IV. CONCLUSION

This paper has introduced the Decentralized Multi-Agent Rapidly-exploring Random Tree (DMA-RRT) and Cooperative DMA-RRT algorithms as multi-agent path planning strategies capable of handling complex environments and agent dynamics. These algorithms combine the CL-RRT path planner with a new coordination strategy, merit-based token passing, to improve team performance while ensuring all constraints are satisfied. The simulation and hardware results also confirm the benefits of the coordination and cooperation strategies. In the future, we intend to extend DMA-RRT to more general multi-agent scenarios with communication limits and dynamic obstacles. We also plan to validate these algorithms in larger simulations and experiments.

ACKNOWLEDGMENTS

The authors would like to thank Brandon Luders and Georges Aoude for their contributions to this research effort. Research funded in part by the Office of Secretary of Defense under Air Force Contract FA8721-05-C-0002

REFERENCES

[1] M. Flint, M. Polycarpou, and E. Fernández-Gaucherand, "Cooperative path-planning for autonomous vehicles using dynamic programming," in *Proceedings of the IFAC World Congress*, Barcelona, Spain, 2002.
[2] M. Likhachev and A. Stentz, "R* search," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2008, pp. 344–350.
[3] T. Schouwenaars, B. de Moor, E. Feron, and J. P. How, "Mixed integer programming for multi-vehicle path planning," in *Proceedings of the European Control Conference*. Porto, Portugal: European Union Control Association, September 2001, pp. 2603–2608.
[4] W. B. Dunbar and R. M. Murray, "Model predictive control of coordinated multi-vehicle formations," in *Proceedings of the IEEE Conference on Decision and Control*, Las Vegas, NV, December 2002 2002, pp. 4631–4636.
[5] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential fields," *IEEE Transactions on Robotics & Automation*, vol. 8, no. 5, pp. 501–518, Oct. 1992.
[6] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, August 1996.
[7] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University, Tech. Rep. 98-11, October 1998.
[8] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 116–129, January-February 2002.
[9] Y. Kuwata, J. Teo, S. Karaman, G. Fiore, E. Frazzoli, and J. P. How, "Motion planning in complex environments using closed-loop prediction," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, 2008.
[10] B. Luders, S. Karaman, E. Frazzoli, and J. P. How, "Bounds on tracking error using closed-loop rapidly-exploring random trees," in *American Control Conference (ACC)*, Baltimore, MD, 2010, pp. 5406–5412.
[11] P. Scerri, S. Owens, B. Yu, and K. Sycara, "A decentralized approach to space deconfliction," in *Proc. 10th Int Information Fusion Conf*, 2007, pp. 1–8.
[12] O. Purwin and R. D'Andrea, "Path planning by negotiation for decentralized agents," in *American Control Conference (ACC)*, 9-13 July 2007, pp. 5296–5301. [Online]. Available: 10.1109/ACC.2007.4283024
[13] G. Hoffmann and C. Tomlin, "Decentralized cooperative collision avoidance for acceleration constrained vehicles," in *IEEE Conference on Decision and Control*, 2008, pp. 4357–4363.
[14] V. Desaraju, H. C. Ro, M. Yang, E. Tay, S. Roth, and D. Del Vecchio, "Partial order techniques for vehicle collision avoidance: Application to an autonomous roundabout test-bed," in *IEEE International Conference on Robotics and Automation*, 2009, pp. 82–87.
[15] G. S. Aoude, B. D. Luders, S. Levine, and J. P. How, "Threat-aware Path Planning in Uncertain Urban Environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, October 2010 (to appear).
[16] P. Trodden and A. Richards, "Robust distributed model predictive control using tubes," in *Proceedings of the American Control Conference*, Minneapolis, MN, June 2006, pp. 2034–2039.
[17] J. P. How, B. Bethke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment," *IEEE Control Systems Magazine*, vol. 28, no. 2, pp. 51–64, April 2008.
[18] Y. Kuwata, A. Richards, T. Schouwenaars, and J. P. How, "Decentralized Robust Receding Horizon Control for Multi-vehicle Guidance," *American Control Conference (ACC)*, pp. 2047–2052, June 2006.