HARVARD UNIVERSITY, COMPUTER SCIENCE DEPARTMENT

Cooperative multi-agent path planning under strict real-time constraints

Nathaniel Tucker & Svilen Kanev

May 9, 2013

In this project, we propose to incorporate collaboration for real-time path planning in a contiguous world. The main purpose of the project is to demonstrate the benefits of collaboration between multiple agents planning potentially intersecting paths in a real-world environment, and to balance those benefits with the cost of communicating intended paths.

1 INTRODUCTION

"*By mid-21st century, a team of fully autonomous humanoid robot soccer players shall win the soccer game, comply with the official rule of the FIFA, against the winner of the most recent World Cup.*" [11]. This project's inspiration is robotic soccer and, more specifically, the Robocup challenge (its mantra stated above). More specifically, we look into the Robocup small-size league [2], where the focus falls mainly on the fast-paced nature of the game. Ball speeds in this league are limited at 10 m/s, and are often achieved. Similarly, robot movement speeds of 2-4 m/s can be observed[1]. To be able to meaningfully compete at such speeds, teams have developed high-performing systems that are able to re-evaluate plans and decisions many times in a second, and actuate robots accordingly. As of 2013, typical stacks for sensing and motion planning run at frequencies of 20-60 Hz.

A major component of such a real-time motion planning stack is the path planner. Its purpose can be synthesized to: *find a dynamically feasible path from point A to point B avoid-*

[1] An example at speeds several times slower can be seen here [1]

ing (possibly mobile) obstacles. In this definition, points A and B are the agent's current position and a target set by higher levels of AI in a physical, contiguous, two-dimensional space.

The result of a path planner is a set of waypoints that the agent follows in the physical world until the next replanning step. Commonly, these waypoints also contain a target velocity, and thus fully define the expected robot trajectory in space and time.

The target of a planner in this domain is to minimize the time until reaching the goal point. This is often done by selecting the shortest feasible path that avoids collisions with obstacles. A collision is assumed to have a very high time cost.

In a collaborative multi-agent scenario, this high cost can be avoided in certain cases. When the two colliding agents can collaborate (one of them is not a static obstacle, or an adversary), they can coordinate such that their paths do not cross. Developing a technique for such coordination in a real-time setting is the purpose of this project.

2 Background and Related Work

There are two domains that this work falls under: pathfinding and communication in multi-agent systems. Both fields are very prolific and we only provide a sample of relevant approaches. LaValle [12] provides a significantly more in-depth overview on path planning, while Ferber [8] summarizes different approaches to multi-agent coordination.

2.1 Pathfinding

Pathfinding can easily be posed as a search problem, and solved by traditional search algorithms. The classic solution to such problems in discretized space is the A* algorithm, which is provably optimal with an appropriate heuristic [15]. However, using such algorithms in a continuous setting that requires high precision is often impractical, especially so while retaining optimality. Recently, R* [14] and Randomized A* [7] have gone in the direction of relaxing the optimality constraint of A*, providing suboptimal paths, but at significantly lower overheads.

One class of algorithms that takes such relaxation even further is *Rapidly-exploring random trees* (RRT-s). RRT is a randomized data structure designed for a broad class of path planning problems, and specifically amenable to nonholonomic constraints with a high degree of freedom. Two properties make this class of algorithms very suitable to a fast-paced robotic environment: the relatively low cost of replanning, and the ability to easily discard paths that would be infeasible under dynamic constraints (f.e. taking too sharp a turn). Originally proposed to quickly search state spaces with high dimensionality [13], RRT algorithms have become very popular in dynamic planning in 2D spatial environments [4]. The real-time RRT algorithm proposed by Frazzoli [9] and extended by Kuwata et al. as Closed-loop RRT (CL-RRT) [19] is much more useful as an online path planner. The material here employs the CL-RRT algorithm as the principal single-agent path planer.

2.2 Coordinated Planning

Coordinated planning is also heavily studied. A large body of research solves the multi-agent decision issue by creating a single point of centralization and finding the optimal team policy at that point. Some examples include modelling the problem as a DEC-(PO)MDP [16, 10] and finding appropriate heuristics to execute a centralized plan in a decentralized manner. Generally these centralized policies will sacrifice runtime for the boon of better performance and are not immediately applicable to a fast-paced real-time scenario, where optimality is not strictly necessary.

Decentralized Coordinated Planning A plethora of approaches avoid the high cost of centralization. For example, Bhattacharya et al. [3] model the set of planning goals as a task graph that achieves implicit communication through time parametrizing goals. Snape et al. [17] reason about other agents by predicting their trajectories and projecting their potential positions. Finally, Trodden and Richards [18] envision a decentralized scheme based on a fixed cycled planning order, where agents take turns in forcing others to avoid their path.

Recently, Desaraju and How [6] have proposed a cooperative RRT-based algorithm that appears to fit well the robotic soccer domain by using an constant-time single agent planner and very limited communication. Similarly to Trodden and Richards [18], Decentralized Moving Agent RRT (DMA-RRT) avoids the large cost of collision by having each agent broadcast their current set of waypoints to all other agents. After a voting procedure to determine conflict resolution, a winner modifies others' waypoints to avoid collisions, and broadcasts the updated set of waypoints.

Desaraju and How incorporate the benefit for replanning into deciding "who" takes precedence in replanning. Instead of arbitrary precedence selection the algorithm quantifies Potential Path Improvement (PPI), and gives precedence to the agent whose path is deemed "best" for the team. The computation of PPI requires the agent to compare the costs of its current path to the new path. CL-RRTs dramatically simplify this step by maintaining the tree of feasible paths.

In designing the algorithm, Desaraju and How made a few key assumptions:

1. A path remains "good" for the time it takes the whole team to re-plan.
2. The environment is assumed to be known and only contain static obstacles.
3. Each agent has a model of both its own and the other agents' dynamics.

Remain cognizant of these three points as we move forward, as the algorithm we propose seeks to address all three.

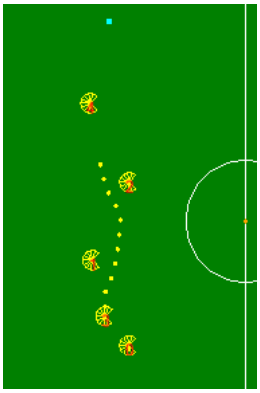


Figure 3.1: Single-agent RRT-based planner in action. Simulation environment.

3 PARTIALLY-COORDINATED PLANNING

The environment that we target can benefit a lot from the approaches described in the previous section. The high speeds relative to the world size imply that replanning often suboptimally can be more beneficial than generating an optimal, but stale path slowly. Coupled with a continuous world, this suggests that RRT-based planners are a good match for the domain.

Furthermore, the robot soccer setting often contains a set of shared goals (f.e. two agents reaching for the ball). Around these, coordination is required – a naive CL-RRT is very susceptible to collisions and intractable positions (deadlock). However, a centralized approach is impractical because of the time cost of communication. A distributed scheme such as DMA-RRT is therefore an appealing solution.

Finally, while the set of goals in the target domain can be shared, goals are not necessarily spatially close ot each other. Thus, the global resolution scheme of DMA-RRT can often hurdle performance – for example, the agent winning the auction can be far away from all other participants and irrelevant for collision avoidance. This requires either a global Potential Path Improvement (PPI) function or localized auctions. Of these, localized auctions appear more appealing because they require less reasoning about global state.

In the rest of this section, we show the following:

1. An implementation of naive CL-RRT without communication.
2. A modification of Desaraju and How's DMA-RRT, fitting it to a faster-paced domain: *Extended DMA-RRT (E-DMA RRT)*.
3. A further improvement, with "islands of communication" for subsets of agents: *Partitioned DMA-RRT (P-DMA RRT)*.

3.1 NAIVE CL-RRT SANS COMMUNICATION

We have a large amount of supporting infrastructure to carry out this project, courtesy of the RFC Cambridge robotic soccer team (http://rfcbots.com). This infrastructure crucially includes a real-world robot testbed, a simulator that properly models the physics of a soccer-playing robot, and a single-agent CL-RRT-based planner, finely tuned to these physical parameters. Figure 3.1 shows a snapshot of this planner, avoiding several static obstacles in the simulation environment.

Thus the implementation of the naive system has been created for us. However, it is strictly single-agent – the only multi-agent-like reasoning is projecting others' velocities forward similar to Snape et al. [17]. This strategy helps avoidance to an extent, but is relatively limited when agents change direction often. It also does not address the issue of shared goals.

We extended their existing codebase to include multi-agent path planning tests and measurement infrastructure to quantify multi-agent performance. Running the naive CL-RRT planner in a multi-agent setting with shared goals demonstrates a propensity toward collision and deadlock when two agents are trying to reach the same goal. This occurs so commonly that the naive planner is barely useful when goals are shared.

3.2 EXTENDED DMA-RRT

The implementation of E-DMA RRT follows two components, and individual and a cooperative component.

INDIVIDUAL E-DMA RRT In this part the agent is initialized with some dynamically feasible path p_0. One agent will be initialized as the token holder while all others will lack the winner token. According to a cost function PPI (which we will go into later) the agents will grow their paths. Then based on this function for each agent we will determine the new winner. If the winner agent updates its plan, the constraints on the other agents must also be updated accordingly. This update occurs in the interaction component.

INTERACTION E-DMA RRT The second component to the DMA-RRT would be the interactive component. Here the agents communicate via two messages. The first identifies the token winner and has its plan, the second is the bid (PPI value) for the token.

When the path and the winner message are received, the agent must update the obstacles in its plans. We model each of the agents' plans by a set of waypoints and thus add them to the other agents' obstacles. When the token winner receives this message, it can assign itself the token and take on its plan. Conceptually, the previous bid winner decides the winner for the next round, after receiving everyone else's messages. Because we operate in a simulation environment, we could afford to simulate this procedure centrally without affecting the results and maintain consistency easier.

Thus the extensions to the original DMA-RRT are as follows:

Algorithm 1 DMA-RRT: Individual Component

Initialize with $p_0, k = 0$
$Have\_Token \leftarrow false$ except for one agent
while Agent is active **do**
    $k \leftarrow k + 1$
    Grow CL-RRT and identify best path p_k^*
    if $Have\_Token$ **then**
        $p_k \leftarrow p_k^*$
        Assign the best bid the winner
        Broadcast waypoints of p_k and winner to all agents
        $Have\_Token \leftarrow false$
        $bid \leftarrow PPI(p_k^*)$
        Broadcast bid
    else
        $p_k \leftarrow p_k^*$
        $bid \leftarrow PPI(p_k^*)$
        Broadcast bid
    end if
end while

Algorithm 2 DMA-RRT: Interactive Component

while Agent is active **do**
    Listen for messages
    if received message with winner and waypoints **then**
        if agent is not the winner **then**
            Update obstacles
        end if
    end if
    if Received bid message **then**
        Update sender's bid
    end if
end while

- The bid winner participates in the next auction round.
- The bid losers get to re-plan, but need to take the previous round's winner into account.
- We extended their notion of PPI to better fit the domain.

The first two changes were necessary, once again, because of the dynamicity of the domain. The original DMA-RRT only assumes static obstacles and infrequent enough replanning, such that an agent does not need to re-plan until it eventually becomes the bid winner. In our case, this performed significantly more poorly than the naive planner, and we do not include it as a comparison point. However, having the bid winner from the previous round participate in the auction can easily lead to one agent dominating planning. Thus, we allowed the rest to re-plan at every time step, too.

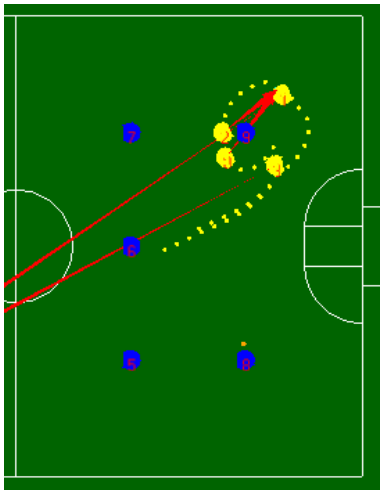


Figure 3.2: Example of P-DMA RRT coordinating avoidance around a shared goal.

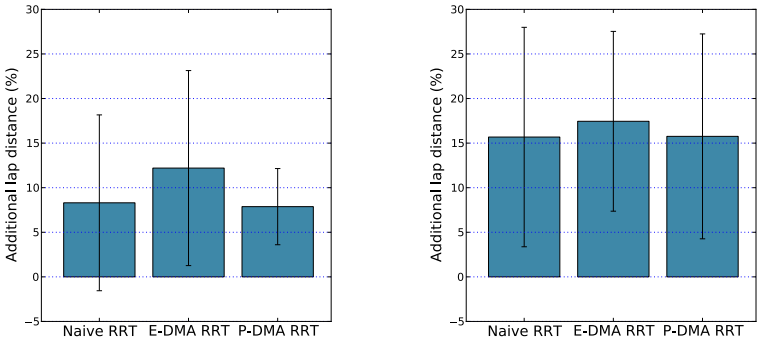
Finally, our model of PPI encapsulates a few metrics, shared with the single-agent planner:

1. Lose points for the length of the path (i.e. prefer shorter paths).
2. Lose points if the path has too sharp a curve (thus the robot cannot execute it).
3. Win points for current path agreement (the robot will not need to expend energy changing its current velocity).

3.3 Partitioned DMA-RRT

The final extension to the DMA-RRT algorithm originally imagined by Desaraju and How was to partition the auctions. The implementation of the Extended DMA-RRT, showed good improvement in multi-agent environments when goals are close to each other (see Section 4), but incurred unnecessary communication when clusters of agents shared a set of spare goals. Agents would avoid the winner of the bid's path, however would not avoid each others' paths. This would cause situations where the bid winner would be in a spatially different location, winning the bid, while a number of robots in spatially similar locations with lower PPI would still collide.

Thus, P-DMA RRT partitions the environment into regions of a fixed radius around each agent. Robots in each such "island of communication" conduct their own auction and choose a single winner (Figure 3.2 shows one such island around a shared goal). Then, the algorithm continues exactly like E-DMA RRT – the auction winner for each region broadcasts their planned path to everyone else in the region and the bid losers make sure to avoid the path. Auctions regions are re-evaluated at a fixed number of time steps. Thus, the resulting



(a) Scenario 1 (far). (b) Scenario 2 (near).

Figure 4.1: Additional accumulated distance over a naive straight-line path. Scenario 2 requires significantly more obstacle avoidance and coordination from the agents.

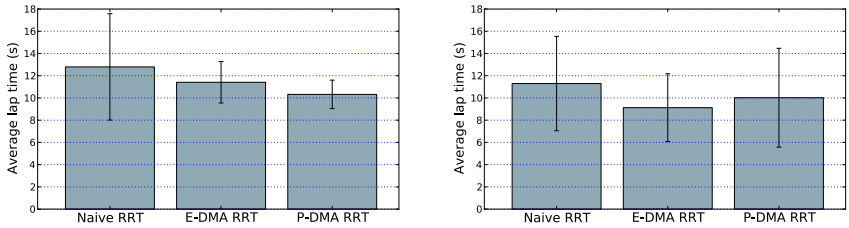
algorithm is more general than E-DMA RRT (E-DMA is the corner case with an infinite region radius).

4 Results

We evaluated the different algorithms described above in simulation. The simulation environment is set up to reproduce the highly dynamic nature of the robotic soccer domain, and its relatively low agent count. More specifically, we model two teams of 4 soccer-playing robots that move with a maximum velocity of 1 m/s on a 4×6 m field. A more thorough description of the simulated agent dynamics is presented in prior work [5]. Agent reaction times are on the order of 10 ms and are highly dependant on current velocity. With such parameters, the agents can easily cross the entire field in a very limited time, creating an environment that favors constant and cheap replanning to avoid mobile obstacles.

In the evaluation setup, each of the four agents goes independently through a shared set of goal states. A set of static obstacles prevents the trivial straight-line paths between the goals. The agents themselves act as dynamic obstacles, especially when near the shared goal states. We show results from two simulated scenarios – with different distances between the goal states. In Scenario 1, the shared goals are relatively far away from each other, and agents only need to avoid each other when they are aiming for the same goal. Contrary, the goals in Scenario 2 are near each other, requiring coordination between all agents for the majority of time. Figure 4.1 demonstrates this difference between the two scenarios. It shows that, compared to Scenario 1, agents in Scenario 2 accumulate a significantly larger distance than the optimal because of the increased pressure on obstacle avoidance. Note that this additional accumulated distance is independent of the particular planner used.

We measure the average time the agents take to complete the set of goals. Figure 4.2 shows the results from this experiment. The first conclusion that can be drawn is that any coordination strategy performs better than the naive case, with a 10-20% average improvement



(a) Scenario 1 (far). (b) Scenario 2 (near).

Figure 4.2: Average time to reach a set of goals for evaluated planners. The effectiveness of coordination depends on the sparsity of goals in the planning scenario.

in total time. Thus, any form of communication is effective in reducing collisions between agents. Second, the two proposed algorithms perform differently based on the sparsity of goal states in Scenario 1 and Scenario 2.

E-DMA RRT, which always selects a single source of authority for the entire team shows a smaller improvement when goals are sparse (Scenario 1). If the agents are going towards two relatively unrelated goals, choosing only one auction winner with the highest PPI near one goal can leave multiple agents near another goal uncoordinated. As shown in Figure 4.2a, P-DMA RRT optimizes for this case and its concept of "islands of communication" results in shorter average times. However, when the goal states are in close proximity, partial communication performs slightly worse than the global case (Figure 4.2b).

Finally, both communication strategies significantly reduce the number of agent collision compared to Naive RRT. While part of this observation is included in the time results (via the costs of slowing down and speeding up after a collision), the mechanical wear costs of collisions are not modelled since they are harder to quantify. The cases that result in deadlock towards a shared goal are not included in the measurements in Figure 4.2, but, qualitatively, such cases were much more rare for the coordinated planners.

5 Future Work

One area for improvement of the demonstrated algorithms is the localization heuristics. Choosing better how to localize communication and how to be ancillary to full communication could further improve planning performance in a sparse scenario. If we assume that each agent can reason about the goals of other agents, two immediate approaches are possible: (i) evaluating the sparsity of team goals and adjusting the communication radius appropriately; (ii) cheaply estimating other agents' paths by emulating several iterations of RRT between them and their goals.

Furthermore, a comparison with other approaches to planning would be interesting. For example, the global communication scheme shares some properties with the centralized DEC-MDP algorithm and comparing the two might could be beneficial.

6 Conclusion

In this project, we proposed including collaboration in real-time path planning through a contiguous world. We implemented a real-time path planning algorithm in a multi-agent domain, sans communication, with full communication, and with local communication. We have concluded that indeed, in a multi-agent domain sans communication, not only do collisions invariably occur, but also intractable situations, deadlock, happen almost surely in multi-agent environments with more than two agents.

With full communication, and a single-winner heuristic, collisions occur, and deadlock is very infrequent. Such an approach can see real gains in speed, but the global nature of a single-winner heuristic causes it to perform poorly for a sparse set of goals. We addressed this problem through local communication, a system where each agent communicates with the ones most relevant to it. This shows an improvement in collision avoidance and nearly no deadlock.

7 Appendix I

Since we used a large amount of external supporting infrastructure from the RFC Cambridge soccer team, re-running our experiments requires a rather specific development environment (which fits in a several-gigabyte virtual machine image). We will be more than happy to demo the system in person on the laptops we have already set up. Just send us an email.

8 Appendix II

Our domain is not text-based. Instead of a trace, we provide video snapshots of the system simulating various planning algorithms here: http://youtu.be/zBPUBSTnBrw.

9 Appendix III

The code for this project is attached separately. Other than the single-agent RRT planner (SmoothRRTPlanner.cs), which we did not write, the rest is in diff format, highlighting the changes we made to the whole system for this assignment. This is only done so that we do not include 50,000+ lines of irrelevant supporting code with the assignment.

References

[1] RFC Cambridge Training, https://www.youtube.com/watch?v=amESKG0rjQg.

[2] RoboCup Small-Size League, http://robocupssl.cpe.ku.ac.th/.

[3] S. Bhattacharya, M. Likhachev, and V. Kumar. Multi-agent path planning with multiple tasks and distance constraints. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 953–959. IEEE, 2010.

[4] J. Bruce and M. Veloso. Real-time randomized path planning for robot navigation. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2383–2388. IEEE, 2002.

[5] A. Colin, J. Henion, R. Jin, B. Johnson, S. Kanev, L. Liu, A. Rameriz, M. Thomas, and D. Wu. RoboCup Team Description Paper: RFC Cambridge. 2011.

[6] V. R. Desaraju and J. P. How. Decentralized path planning for multi-agent teams in complex environments using rapidly-exploring random trees. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4956–4961. IEEE, 2011.

[7] R. Diankov and J. Kuffner. Randomized statistical path planning. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 1–6. IEEE, 2007.

[8] J. Ferber. *Multi-agent systems: an introduction to distributed artificial intelligence*, volume 1. Addison-Wesley Reading, 1999.

[9] E. Frazzoli, M. A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *Journal of Guidance, Control, and Dynamics*, 25(1):116–129, 2002.

[10] C. V. Goldman and S. Zilberstein. Decentralized control of cooperative systems: Categorization and complexity analysis. *J. Artif. Intell. Res. (JAIR)*, 22:143–174, 2004.

[11] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, and H. Matsubara. RoboCup: A challenge problem for AI. *AI magazine*, 18(1):73, 1997.

[12] S. M. LaValle. *Planning algorithms*. Cambridge university press, 2006.

[13] S. M. LaValle and J. J. Kuffner Jr. Rapidly-exploring random trees: Progress and prospects. 2000.

[14] M. Likhachev and A. Stentz. R* search. In *Proceedings of the 23rd national conference on Artificial intelligence-Volume 1*, pages 344–350. AAAI Press, 2008.

[15] J. Pearl. Heuristics: intelligent search strategies for computer problem solving. 1984.

[16] R. S. Roth, Maayan and M. Veloso. What to communicate? execution-time decision in multi-agent pomdps. *Distributed Autonomous Robotic Systems*, 7:177–186, 2006.

[17] J. Snape, J. van den Berg, S. J. Guy, and D. Manocha. The hybrid reciprocal velocity obstacle. *Robotics, IEEE Transactions on*, 27(4):696–706, 2011.

[18] P. Trodden and A. Richards. Robust distributed model predictive control using tubes. In *American Control Conference, 2006*, pages 6–pp. IEEE, 2006.

[19] S. K. G. F. E. F. Y. Kuwata, J. Teo and J. P. How. Motion planning in complex environments using closed-loop prediction. AIAA Guidance, Navigation, and Control Conference (GNC), 2008.