Finding Near-optimal Solutions in Multi-robot Trajectory Planning

Michal Čáp[1], Peter Novák[2], Alexander Kleiner[3]

***Abstract*— We deal with the problem of planning collision-free trajectories for robots operating in a shared space. Given the start and destination position for each of the robots, the task is to find trajectories for all robots that reach their destinations with minimum total cost such that the robots will not collide when following the found trajectories. Our approach starts from individually optimal trajectory for each robot, which are then penalized for being in collision with other robots. The penalty is gradually increased and the individual trajectories are iteratively replanned to account for the increased penalty until a collision-free solution is found. Using extensive experimental evaluation, we find that such a penalty method constructs trajectories with near-optimal cost on the instances where the optimum is known and otherwise with 4-10 % lower cost than the trajectories generated by prioritized planning and up to 40 % cheaper than trajectories generated by local collision avoidance technique ORCA.**

I. INTRODUCTION

An important problem in multi-robotics is the coordination of trajectories of individual robots. That is, given n circular robots, together with their starting and destination positions, we are interested in finding a set of individual trajectories $\pi_1, \ldots, \pi_n$ that do not collide with each other, i.e. $\forall_{i,j} \forall t : |\pi_i(t) - \pi_j(t)| > d^{\text{sep}}$, where d^{sep} is the required separation distance – usually the sum of radii of the two robots, while at the same time the overall costs (e.g., sum of trajectory lengths) is minimized.

It is known that path coordination of circular vehicles among polygonal obstacles is NP-hard [4]. While the problem is relatively straightforward to formulate as a planning problem in the Cartesian product of the state spaces of the individual robots, the solutions are difficult to find using standard search techniques because the joint state-space grows exponentially with the number of robots.

Prioritized planning [2] is a heuristic approach based on the idea of sequential planning for the individual robots in the order of their priorities, where each robot considers the higher-priority robots as moving obstacles and plans its trajectory to avoid them. While fast, prioritized planning is incomplete and often fails to find a solution even if one exists. Futher, the resulting trajectories are typically noticeably suboptimal.

The techniques of mathematical optimization such as the penalty-based approach [3] have been also studied in the context single-robot and multi-robot trajectory generation. In particular, a distributed penalty-based method has been used to solve the multi-robot rendezvous problem [1].

[1] Agent Technology Center, Dept. of Computer Science, Faculty of Electrical Engineering, CTU in Prague

[2] Algorithmics, EEMCS, Delft University of Technology

[3] iRobot Inc., Pasadena

We use similar distributed penalty-based approach to find coordinated non-colliding trajectories for multiple robots and propose the *k-step penalty method* that can be seen as a generalization of prioritized planning approach. The algorithm performs a series of single-robot path planning queries in a dynamic environment such that the trajectories that are too close to trajectories of other robots are penalized. Starting from trajectories that disregard collisions with other robots, the collisions between robots' trajectories are gradually being penalized with increasing severity so that they are finally forced out of collision as the penalties tend to infinity. Using extensive experiments, we demonstrate that this heuristic approach tends to generate near-optimal trajectories that are of significantly lower cost than the trajectories generated by prioritized planning and reactive techniques.

II. PENALTY-BASED METHOD

We propose a penalty-based approach that attempts to mitigate the low success rate and low solution quality of prioritized planning, but in the same time retain its tractability. We combine the idea of decoupled planning as used in prioritized planning with iterative increasing of penalty assigned to each robot in collision.

The requirement on minimal separation between trajectories of a pair of robots i, j is modelled by a penalty function assigning penalty to each part of the trajectory of robot i that gets closer to the trajectory of robot j than the required separation distance d^{sep}. The penalty function has the form

$$\Omega_{ij}(\pi_i, \pi_j) = \int_0^\infty \omega_{ij}(|\pi_i(t) - \pi_j(t)|)\, dt\,,$$

where $\pi_i(t)$ is a trajectory of robot i and $\omega_{ij}(d)$ is a bump function:

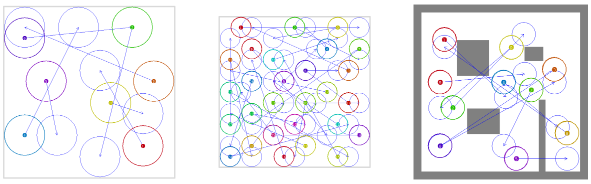
$$\omega_{ij}(d) = \begin{cases} \frac{1}{e^{-1}} \cdot e^{-\frac{1}{1-(d/d^{sep})^2})} & \text{for } d < d^{sep} \\ 0 & \text{otherwise} \end{cases} .$$

Algorithm 1 exposes the *k-step Penalty Method* (PM) algorithm that replans the trajectory of each robot exactly k-times. The algorithm starts by finding a cost-optimal trajectory for each robot using $w = 0$, i.e., while ignoring interactions with other robots. Then, it gradually increases the weight w and thus the penalties start to be taken into account. After each increase of the weight coefficient, one of the robots is selected and its trajectory is replanned to account for the increased penalty. The trajectory is planned by performing space-time search in time-extended roadmap using A*. After the iterative phase finishes, the trajectories of all robots are replanned for the last time with the weight coefficient set to infinity. If the final set of trajectories is

Algorithm 1: k-step Penalty Method

Algorithm PM(k)
- **for** $i \leftarrow 1 \ldots n$ **do** $\pi_i \leftarrow$ Replan($i, 0$)
- **for** $i \leftarrow 1 \ldots n(k-2)$ **do**
  - $r \leftarrow i \bmod n(k-2)$
  - $w_i \leftarrow \tan(\frac{i}{n(k-2)+1} \cdot \frac{\pi}{2})$
  - $\pi_i \leftarrow$ Replan(i, w_i)
- **for** $i \leftarrow 1 \ldots n$ **do** $\pi_i \leftarrow$ Replan(i, ∞)
- **if** $\forall_{ij}\ \Omega_{ij}(\pi_i, \pi_j) = 0$ **then return** $\langle \pi_1, \ldots, \pi_n \rangle$
- **else** report failure

Function Replan(r, w)
- return trajectory π for robot r that minimizes $c(\pi) + w \sum_{j \neq r} \Omega_{rj}(\pi, \pi_j)$



Scenario A Scenario B Scenario C

Fig. 1. Experimental environments

conflict-free, the algorithm returns the trajectories as a valid solution, otherwise it reports failure.

III. Experimental Evaluation

We compared the performance of the proposed k-step penalty method (PM) against prioritized planning (PP), a state-of-the-art optimal algorithm called operator decomposition (OD) [5], and a reactive method ORCA [6] on a range of randomly generated dense multi-robot path planning instances in three synthetic environments shown in Figure 1. We focused on dense collision situations in which all robots are involved in a single conflict cluster. The runtime of each algorithm was limited to 1 hour. The results of the comparison are plotted in Figure 2. We can see that due to the exponentially-growing state space, the optimal algorithm OD was not able to solve the instances with more than 5 robots within 1 hour limit, while PM scales even to instances with more than 20 robots and in the same time generates high-quality solution with relatively low runtime requirements.

IV. Conclusion

We have explored the applicability of penalty-based method to improve success rate and the quality of returned solution of prioritized planning. Our experimental results show that with increasing number of iterations, the algorithm constructs solutions with near-optimal cost (on instances where the optimum was known). On the instances where the optimum was not known, our method consistently provided solutions that are 4-10 % cheaper than solutions provided by prioritized planning and up to 40 % cheaper than the solutions provided by a widely-used reactive technique ORCA.

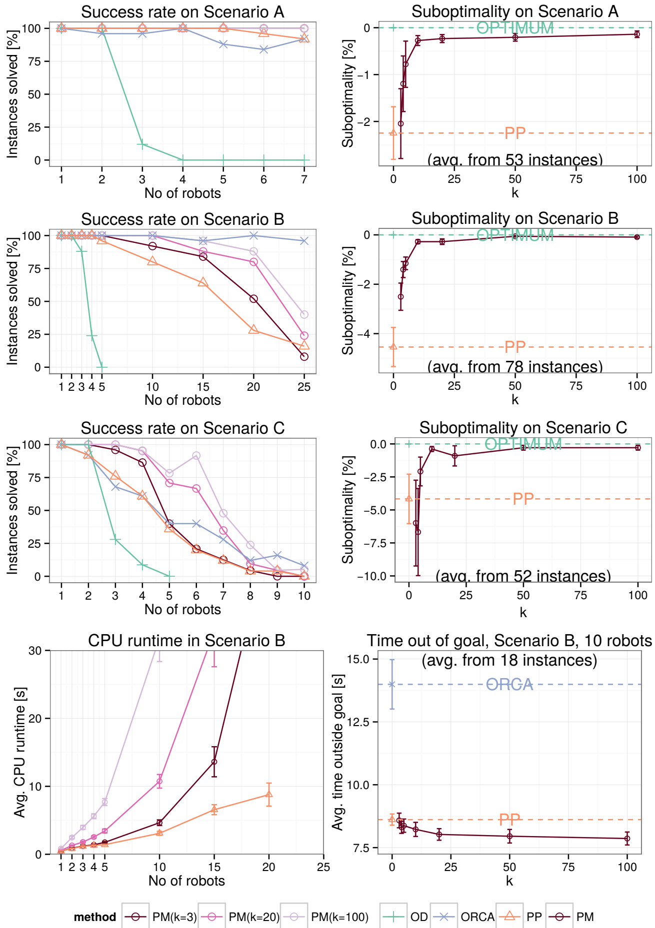


Fig. 2. Results. The plot shows: **Success rate:** the percentage of instances successfully solved by PM($k = 3, 20, 10$), PP, OD, and ORCA in each scenario. **CPU runtime:** average CPU runtime to find a solution by PM($k = 3, 20, 10$) and PP. **Suboptimality:** average suboptimality of solution generated by PM($k = 1, \ldots, 100$) and PP on instances where optimum was known. **Time out of goal:** average difference in solution quality generated by PM($k = 1, \ldots, 100$), PP, and ORCA on instances with 10 robots in Scenario B.

In future, we plan to focus on the investigation of theoretical properties of the method on our problem.

References

[1] Subhrajit Bhattacharya, Vijay Kumar, and Maxim Likhachev. Distributed optimization with pairwise constraints and its application to multi-robot path planning. In *Robotics: Science and Systems*, 2010.

[2] Michael Erdmann and Tomas Lozano-Pérez. On multiple moving objects. *Algorithmica*, 2:1419–1424, 1987.

[3] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer series in operations research and financial engineering. Springer, New York, NY, 2. ed. edition, 2006.

[4] Jacob T Schwartz and Micha Sharir. On the piano movers' problem: Iii. coordinating the motion of several independent bodies: the special case of circular bodies moving amidst polygonal barriers. *The International Journal of Robotics Research*, 2(3):46–75, 1983.

[5] Trevor Scott Standley. Finding optimal solutions to cooperative pathfinding problems. In Maria Fox and David Poole, editors, *AAAI*. AAAI Press, 2010.

[6] Jur Van Den Berg, Stephen Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. *Robotics Research*, pages 3–19, 2011.