Optimized and kinematically feasible multi-agent motion planning

Anja Hellander, Kristoffer Bergman and Daniel Axehill

I. Introduction

An important capability for an autonomous system is the ability to plan in advance what action to take to achieve a goal. Motion planning is the problem of deciding what path or trajectory to follow to safely and efficiently move the system from its current state to a desired state. If the system consists of multiple agents, the resulting multi-agent motion-planning problem (MAMP) quickly becomes more complicated as the planned trajectories for the agents must be feasible and collision-free with respect to other agents as well.

In this work we focus on optimizing and finding kinematically feasible trajectories for multi-agent systems with nontrivial dynamics and large bodies, such as tractor-trailer systems. Our proposed framework consists of two steps: a discrete optimization step where we find a collision-free, kinematically feasible solution to a discretization of the original problem, and a continuous improvement step where the solution found in the discrete optimization step is improved using local optimization.

II. Problem Formulation

We consider k agents operating within the same workspace $W \in \mathcal{R}^2$ containing an obstacle region $O \in W$. Each agent $i \in \{1, \ldots, k\}$ is a nonlinear dynamical system that can be described by

$$\dot{x}_i(t) = f_i(x_i(t), u_i(t)), \quad x(t_0) = x_{i,0} \tag{1}$$

where $x_i \in \mathcal{X}_i \subset \mathcal{R}^{n_i}$ is the state vector, $u_i \in \mathcal{U}_i \subset \mathcal{R}^{m_i}$ is the control input, $t \geq t_0$ is the time elapsed, and $x_{i,\text{init}}$ is the initial state of the agent. Let $R_i(x_i) \subset \mathcal{R}^2$ denote the area occupied by agent i when the agent is in state x_i.

Given a set of initial states $x_{i,0}$ and a set of desired final states $x_{i,f}$, the multi-agent motion-planning problem consists of finding a set of feasible and collision-free trajectories $(x_i(t), u_i(t), t_{i,f})$ such that $x_i(t_0) = x_{i,0}$ and $x_i(t_{i,f}) = x_{i,f}$ for all agents i.

The optimal MAMP problem for k agents can then be posed as

$$\begin{aligned}
&\underset{\{x_i(t),u_i(t),t_{i,f}\}_{i=1}^k}{\text{minimize}} && \sum_{i=1}^{k} J_i(x_i, u_i, t_{i,f}) && \text{(2a)}\\
&\text{s.t.} && \dot{x}_i(t) = f_i(x_i(t), u_i(t)) && \text{(2b)}\\
& && R_i(x_i(t)) \in W \setminus O && \text{(2c)}\\
& && u_i(t) \in \mathcal{U}_i && \text{(2d)}\\
& && R_i(x_i(t)) \cap R_j(x_j(t)) = \emptyset \quad i \neq j && \text{(2e)}\\
& && x_i(t_0) = x_{i,0} && \text{(2f)}\\
& && x_i(t_{i,f}) = x_{i,f} && \text{(2g)}
\end{aligned}$$

where the cost functions J_i are defined as

$$J_i = \int_{t_0}^{t_f} l(x_i(t), u_i(t))\, dt \tag{3}$$

where $l(x(t), u(t)) \geq 0$ is the user-defined running cost. For example, selecting $l(x, u) = 1$ results in a minimum-time problem.

III. The Planning Framework

The proposed framework can be seen as an extension of the single-agent framework proposed in [1]. The underlying principle is to first solve a discretization of the motion-planning problem to obtain an initial feasible solution. An improvement step is then applied where an optimal control problem (OCP) is posed and solved using the initial feasible solution found in the previous step as initial guess to warm-start the solver.

A. Discrete Optimization Step

As (2) is difficult to solve, we discretize the problem and construct a state-lattice as described in [1]. Similarly, we divide the work space W into a grid of cells and compute for each motion primitive which cells are swept by the agent when executing the motion.

Given a set of initial and goal states for each agent, we propose to find a solution using either conflict-based search (CBS) [2] or priority-based search (PBS) [3]. The general idea of both CBS and PBS is to perform graph search on a graph where each node contains a set of constraints on the trajectory of each agent, and a trajectory for each agent that obeys these constraints. These trajectories are computed using a single-agent motion planner. Iteratively, the nodes in the graph are examined and if they are found to contain conflicts between the trajectories new nodes are constructed and additional constraints added to solve the conflicts. As CBS is complete and optimal, the resulting solution after the discrete optimization step is resolution optimal. PBS,

This work was partially supported by the Wallenberg Artificial Intelligence, Autonomous Systems and Software Program (WASP), funded by the Knut and Alice Wallenberg Foundation.

A. Hellander and D. Axehill are with the Division of Automatic Control, Linköping University, Sweden {anja.hellander, daniel.axehill}@liu.se

K. Bergman is with Saab AB, kristoffer.bergman@rnfrst.io

however, is neither complete nor optimal and consequently the resulting solution lacks optimality guarantees.

As single-agent motion planner we use Safe Interval Path Planning with Interval Projection (SIPP-IP) [4]. We extend the SIPP-IP algorithm so that it can handle agents that cover more than one grid cell and cost functions that contain terms other than time.

B. Continuous Improvement Step

To improve the solution found by the discrete optimization step, we pose (2) as a multi-phase OCP and solve it using the found solution to warm-start the solver.

Given a set of trajectories $(x_i(t), u_i(t), t_{i,f})$ we assume without loss of generality that the agents are ordered so that $i < j$ implies $t_{i,f} \leq t_{j,f}$, i.e., that agent i reaches is goal no later than agent j. We then divide the problem into $M = k$ phases where the first phase covers the time until the first agent reaches its goal, and each subsequent phase m covers the time between the time when agent $m-1$ reaches its goal and agent m reaches its goal. For each phase m we introduce the decision variable t_m denoting the time duration of the phase. For each phase it is then possible to use the same time discretization for all agent trajectories to allow for easy collision checking, while different phases having different discretization allows the optimization to extend the duration of the trajectories of some agents while reducing the duration for others.

Note that the order in which the agents reach their goal will be the same as in the initial feasible solution. Similarly, the combinatorial aspect of how to navigate around static obstacles as well as other agents is also implicitly encoded by the initial feasible solution and will be maintained.

IV. Numerical experiments

The proposed planning framework has been implemented and evaluated for a tractor-trailer system. Each tractor-trailer combination is modeled as a general 2-trailer with a car-like truck. The dynamical model of the 2-trailer system is the same as described in [5].

To compare the performance of CBS and PBS we generate 100 problem instances for $n = 2, 3, 4, 5$ agents in an obstacle-free 200×200 map. All problem instances are generated so that there is no overlap between initial positions or between terminal positions, and so that there exists a feasible trajectory for each agent if all other agents are ignored. Both planners were given up to $100\,s$ to solve each problem instance. If successful, the resulting solution was then used for the improvement step. Figure 1 shows the number of problems that were successfully solved by the planners. It can be seen that PBS solves a higher number of problems than CBS does. It can also be seen that as the number of agents increase the number of problems where an improved solution is found decreases. Table I shows the results for the problems solved by both planners, to facilitate a direct comparison on execution time and cost function values.

As can be seen in Table I, the computation time t_p for the initial solution is similar for CBS and PBS. The cost J_p of the initial solution is slightly lower for CBS, which is to be expected as CBS is (resolution) optimal and PBS is not. Perhaps more interesting is that the cost J_{imp} after the improvement step is sometimes lower for CBS and sometimes for PBS. This is likely due to the fact that sometimes the numerical solver fails to solve a problem for either CBS or PBS, leading to the initial solution being kept and the cost function not being improved. The optimization step improves the cost function with around 10%.

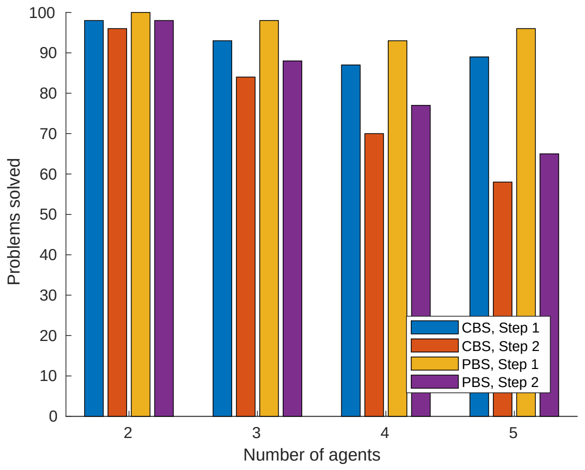

Fig. 1: The number of problems successfully solved by the two planners. Successfully solved during Step 2 means that the numerical optimization solver was able to improve the solution.

TABLE I: Comparison of CBS and PBS in an obstacle-free environment. Problems solved by both planners.

N	Planner	t_p	t_{imp}	J_p	J_{imp}	r_p	r_{imp}
2	CBS	0.17	2.84	258.81	219.52	98	96
2	PBS	0.19	2.91	258.89	220.50	98	96
3	CBS	0.22	13.24	389.78	338.62	92	83
3	PBS	0.22	13.65	390.02	342.44	92	84
4	CBS	0.37	26.60	503.87	448.82	83	67
4	PBS	0.35	26.23	504.83	451.92	83	71
5	CBS	0.46	41.66	632.40	572.27	87	58
5	PBS	0.46	41.66	636.03	576.53	87	60

References

[1] K. Bergman, O. Ljungqvist, and D. Axehill, “Improved path planning by tightly combining lattice-based path planning and optimal control,” *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 1, pp. 57–66, 2020.

[2] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, “Conflict-based search for optimal multi-agent pathfinding,” *Artificial intelligence*, vol. 219, pp. 40–66, 2015.

[3] H. Ma, D. Harabor, P. J. Stuckey, J. Li, and S. Koenig, “Searching with consistent prioritization for multi-agent path finding,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, pp. 7643–7650, 2019.

[4] Z. A. Ali and K. Yakovlev, “Safe interval path planning with kinodynamic constraints,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 12330–12337, 2023.

[5] K. Bergman, O. Ljungqvist, T. Glad, and D. Axehill, “An optimization-based receding horizon trajectory planning algorithm,” *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15550–15557, 2020.