db-ECBS: Interaction-Aware Multi-Robot Kinodynamic Motion Planning

Akmaral Moldagalieva[1], Joaquim Ortiz-Haro[2], Wolfgang Hönig[1]

I. Introduction

An essential requirements to enhance the autonomy of a team of robots is being able to reach the goal quickly while avoiding collisions with obstacles and other robots. Moreover, the planned motions are required to respect the robots' dynamics and actuation limits. For aerial robotic teams, there are aerodynamic effects that effectively create an *interaction force* if two robots are close to each other or if a robot is close to the ground ("ground effect") [1]. These effects are difficult to model, but accurate predictions enable close-formation flights that are otherwise impossible [2].

Existing kinodynamic motion planners for multi-robot teams include search-based method [3], sampling-based methods [4], control-based methods [5], optimization-based methods [6], and hybrid approaches [7]. The current state-of-the-art, discontinuity-bounded Conflict-based Search (db-CBS), is probabilistically complete, asymptotically optimal, and demonstrates empirically that it can find high-quality solutions quickly, compared to other approaches. However, db-CBS does not scale well with the number of robots (at most 8 robots have been reported) and cannot directly reason about interaction forces that are unavoidable for aerial robots.

In this paper, we present Discontinuity-Bounded Enhanced Conflict-Based Search (db-ECBS), a generalization of db-CBS to address these shortcomings. Algorithmically, we employ ideas from the multi-agent path finding community for stronger heuristic guidance [8] and augment the state space to include interaction-forces during the planning directly. Empirically, we demonstrate that we can compute solutions for larger team sizes and compare our method with existing state-of-the-art algorithms for planning motions of aerial teams: db-CBS and MAPF/C+POST [9]. Both existing solvers use conservative collision shapes to avoid aerodynamic interactions entirely, rather than planning with them.

II. Approach

A. Problem Definition

The multi-robot kinodynamic motion planning problem is defined as follows. Given the robot's state space $\mathcal{X}$, action space $\mathcal{U}$, dynamics $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$, start state $\mathbf{x}_s$, goal state $\mathbf{x}_g$, and representation of the environment $\mathcal{W}_{\text{free}}$, find sequences of states and actions such that the robots move from their start state $\mathbf{x}_s$ to their goal state $\mathbf{x}_f$, while obeying the dynamics and avoid collisions with the environment or each other. Note that for the multi-robot system the states, actions, and dynamics are simply stacked for all robots allowing even heterogeneous robot teams. The interaction-aware planning requires an additional function that predicts the interaction forces $\boldsymbol{\psi}^{(i)}(\cdot)$ that depends on the states of nearby robots. We bound the magnitude of this force to be at most $\psi_{\max}$ and also include the forces in the dynamics $f(\mathbf{x}, \mathbf{u})$.

B. Background: db-CBS

db-CBS relies on *motion primitives*, i.e., sequences of states and actions that are consistent with the known robot dynamics. Internally, these primitives are connected with a (bounded) discontinuity. db-CBS has the following steps: i) compute single-robot motions with discontinuous jumps for each robot using db-A* (*low-level* search); ii) resolve collisions between individual robots one-by-one (*high-level* search); iii) repear discontinuous motions into smooth and feasible trajectories with *optimization*; iv) repeat steps i) to iii) with lower discontinuity bound and more motion primitives.

db-CBS defines a conflict as $C = \langle i, j, \mathbf{x}_k^{(i)}, \mathbf{x}_k^{(j)}, k\rangle$ for a collision between robot i with state $\mathbf{x}_k^{(i)}$ and robot j with state $\mathbf{x}_k^{(j)}$ identified at time k. The resulting constraint for robot i is $\langle i, \mathbf{x}_k^{(i)}, k\rangle$, which prevents it to be within a distance of δ to state $\mathbf{x}_k^{(i)}$ at time k. Similarly, the constraint for robot j is $\langle j, \mathbf{x}_k^{(j)}, k\rangle$. The notion of a discontinuity δ defines the constraint as an actual volume (around a point), which is crucial for efficiency and completeness guarantees.

C. db-ECBS

Db-ECBS builds upon db-CBS by introducing the bounded suboptimality through the use of a suboptimality factor ω. This parameter allows db-ECBS to find a suboptimal solution which is guaranteed to have a cost no more than ω times the cost of db-CBS. Db-ECBS uses a FOCAL priority queue $\mathcal{F}$ in addition to the standard OPEN queue. $\mathcal{F}$ prioritizes nodes by the lowest value f_h, the focal heuristic, which may be inadmissible. For the low-level search, the FOCAL queue is used in db-A^*_ω to avoid solutions that result in many conflicts. Similarly, in the high-level search, another FOCAL queue considers nodes with low conflict count. This approach is similar to ECBS [8], however, we operate in

[1] Technical University of Berlin, Berlin, Germany moldagalieva@tu-berlin.de.
[2] Machines in Motion Laboratory, New York University, USA

We thank Nan Cai, Jan Achermann, and Dennis Schmidt for help with the robot experiments.

Code: `https://github.com/IMRCLab/db-CBS/tree/dev-3d-main`

The research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 448549715.

TABLE I

Wall AND *Window* EXAMPLES. MEDIAN VALUES OVER 5 TRIALS PER ROW. BOLD ENTRIES ARE THE BEST FOR THE ROW, $\star$ NOT TESTED.

#	Instance	MAPF/C+POST				db-ECBS-C				db-ECBS-R			
		p	$t^{\mathrm{st}}[s]$	$J^{\mathrm{st}}[s]$	$J^{f}[s]$	p	$t^{\mathrm{st}}[s]$	$J^{\mathrm{st}}[s]$	$J^{f}[s]$	p	$t^{\mathrm{st}}[s]$	$J^{\mathrm{st}}[s]$	$J^{f}[s]$
1	window2	**1.0**	**1.0**	60.7	60.7	**1.0**	126.2	26.3	26.3	**1.0**	102.0	**26.1**	**26.1**
2	window4	**1.0**	**1.6**	147.3	147.3	0.8	353.1	**55.6**	**55.6**	**1.0**	250.7	56.3	56.3
3	window8	**1.0**	**3.2**	297.1	297.1	**1.0**	812.5	**114.9**	**114.9**	**1.0**	508.6	120.6	120.6
4	window10	**1.0**	**4.0**	373.0	373.0	0.6	1160.3	**141.6**	**141.6**	**1.0**	798.5	156.4	156.4
5	window12	**1.0**	**4.9**	439.8	439.8	0.8	2731.0	**172.6**	**172.6**	**1.0**	801.4	188.7	188.7
6	window16	**1.0**	**7.7**	657.4	657.4	0.4	3393.9	**244.0**	**244.0**	0.8	2476.4	259.3	259.3
7	wall8	$\star$	$\star$	$\star$	$\star$	0.2	434.9	72.1	72.1	**1.0**	**132.8**	**66.4**	**66.4**
8	wall10	$\star$	$\star$	$\star$	$\star$	0.4	441.7	117.5	117.5	**1.0**	**336.6**	**99.9**	**99.9**

continuous space, where the heuristic computation is more difficult.

For interaction-awareness, we augment the state of the robots to include the interaction force. Moreover, this requires to generalize the definition of motion primitives, compared to db-CBS, which now include the sequence of states, actions, and interaction forces. During the search, the focal heuristics are adapted to also count potential aerodynamic force violations ($\boldsymbol{\psi}(\cdot) \geq \psi_{\max}$). For the trajectory optimization, we plan, similar to db-CBS, on the stacked dynamics. By including the interaction force $\boldsymbol{\psi}(\cdot)$ as part of the robot's state, its value can be easily constrained to stay within a pre-defined bound.

III. Results

We report success rate (p), computational time until the first solution is found (t^{st}), cost of the first solution (J^{st}), and cost of the final solution (J^{f}) (for planners that have the anytime capability). All planners use a simplified robot model of a 3D double integrator for flying robots and first-order unicycle for ground robots.

We compute the interaction forces using a trained deep neural network that takes the set of relative states of neighbors as input [2] (db-ECBS-R). We compare with a conservative method, which assumes to have an ellipsoid shape around each robot (db-ECBS-C) and another conservative planner that relies on differential flatness (MAPF/C+POST [9]). The results in Table I can be summarized as follows: i) planning using the differential flatness property is significantly faster compared to full kinodynamic planning; ii) kinodynamic motion planning can achieve significantly lower cost solutions; iii) the benefit of planning with the full interaction-aware model is only visible in very dense settings, but there might have a high impact on success rate and cost reduction (rows 7 and 8).

We also conduct physical experiments inside a $7 \times 4 \times 2.75\ \mathrm{m}^3$ room equipped with a motion capture system with twelve Optitrack cameras. We use Bitcraze Crazyflie 2.1 drones for flying robots and control them using Crazyswarm2 [10]. For ground robots we use Polulu 3pi+ 2040 differential-drive robots. Two scenarios are considered. We first test the *Window* example with 8 homogeneous flying robots. The second scenario has 4 ground robots with vertical bamboo bars attached and 4 flying robots, see Fig. 1.

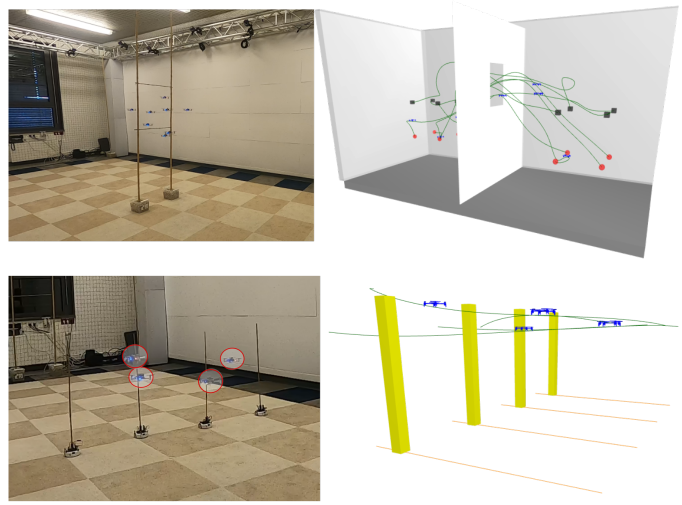

Fig. 1. Top: *Window* example with 8 flying robots. Robots are required swap their positions by passing through a small window. Red circles show the starting position, while dark boxes represent the final state. Bottom: Example with 4 ground robots and 4 flying robots. Ground robots move forward in a straight line, while the flying robots pass through the moving bamboo forest.

IV. Conclusion

We present db-ECBS, a novel kinodynamic motion planner for multi-robot teams that can directly reason about interaction forces that occur for example in aerodynamic teams that fly in close proximity to each other. Algorithmically, we generalize db-CBS to include non-admissible heuristics that guide the search to avoid conflicts and we augment the state space during the search and optimization to include aerodynamic forces. Our approach is probabilistically complete, asymptotically bounded suboptimal, and unlike differentially-flatness-based planners can directly reason about actuation constraints. Empirically, we demonstrate that db-ECBS can produce trajectories that are less than half the cost of existing planners and that the interaction-awareness is in particular important for very dense scenarios.

Future work should look at improving the scalability to larger robot teams as well as the scalability to higher-dimensional state- and action spaces.

REFERENCES

[1] K. P. Jain, T. Fortmuller, J. Byun, S. A. Mäkiharju, and M. W. Mueller, "Modeling of aerodynamic disturbances for proximity flight of multirotors," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2019, pp. 1261–1269.

[2] G. Shi, W. Hönig, X. Shi, Y. Yue, and S. Chung, "Neural-swarm2: Planning and control of heterogeneous multirotor swarms using learned interactions," *IEEE Transactions on Robotics (T-RO)*, vol. 38, no. 2, pp. 1063–1079, 2022.

[3] L. Cohen, T. Uras, T. K. S. Kumar, and S. Koenig, "Optimal and bounded-suboptimal multi-agent motion planning," in *International Symposium on Combinatorial Search (SoCS)*, 2021.

[4] J. Kottinger, S. Almagor, and M. Lahijanian, "Conflict-based search for multi-robot motion planning with kinodynamic constraints," in *International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 13 494–13 499.

[5] A. Tajbakhsh, L. T. Biegler, and A. M. Johnson, "Conflict-based model predictive control for scalable multi-robot motion planning," in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2024, pp. 14 562–14 568.

[6] J. Chen, J. Li, C. Fan, and B. C. Williams, "Scalable and safe multi-agent motion planning with nonlinear dynamics and bounded disturbances," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2021, pp. 11 237–11 245.

[7] A. Moldagalieva, J. Ortiz-Haro, M. Toussaint, and W. Hönig, "db-CBS: Discontinuity-bounded conflict-based search for multi-robot kinodynamic motion planning," in *International Conference on Robotics and Automation (ICRA)*, 2024, pp. 14 569–14 575.

[8] M. Barer, G. Sharon, R. Stern, and A. Felner, "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem," in *European Conference on Artificial Intelligence (ECAI)*, vol. 263, 2014, pp. 961–962.

[9] W. Hönig, J. A. Preiss, T. K. S. Kumar, G. S. Sukhatme, and N. Ayanian, "Trajectory planning for quadrotor swarms," *IEEE Transactions on Robotics (T-RO)*, vol. 34, no. 4, pp. 856–869, 2018.

[10] J. A. Preiss, W. Hönig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3299–3304.