Trajectory Planning for Heterogeneous Robot Teams

Mark Debord, Wolfgang Hönig, and Nora Ayanian

***Abstract*— We describe a method for trajectory planning for heterogeneous mobile robot teams in known environments. We consider two core problems that arise with heterogeneous robot teams: 1) asymmetric inter-robot collision constraints and 2) varying dynamic limits. Asymmetric collision constraints complicate the spatial coordination and are important for close-proximity flight of rotorcraft because of the downwash effect. Varying dynamic limits complicate the temporal coordination between robots and must be taken into account during planning.**

I. Introduction

Trajectory planning for heterogeneous teams of robots is a core problem for many potential applications of multi-robot systems. In order to accomplish complex tasks it could be beneficial for a team to be composed of different types of robots with varied capabilities. This complicates trajectory planning due to each robot having different dynamics and mixed requirements for allowed interactions with each other. Fig. 1 shows an example of a physical experiment in which many quadrotors of different sizes must fly in close proximity and thus be aware of other quadrotors' downwash while also considering the motion of ground robots.

Consider a team of N robots, each of which are one of M types. Each robot is denoted by $r^{(i,k)}$ with $i \in \{1 \ldots N\}$ and $k \in \{1 \ldots M\}$. The operating environment is defined by a set of N_{obs} convex obstacles and a convex boundary. The obstacles and boundary are used to define M configuration spaces $\mathcal{F}^k$. We are given a set of starting locations $s^{(i,k)}$ and goal locations $g^{(i,k)}$ for each robot. We seek to find the time T in which the last robot in the team reaches its goal and collision free trajectories $f^{(i,k)} : [0, T] \rightarrow \mathcal{F}^k$ for each robot such that $f^{(i,k)}(0) = s^{(i,k)}$ and $f^{(i,k)}(T) = g^{(i,k)}$.

The presented method is an extension to prior work done on downwash-aware trajectory planning for large quadrotor teams [1]. The high-level structure of the original approach is retained. First, a graph-based planning method is used to compute a collision-free discretized schedule for all robots in the team. We consider spatial and temporal differences of the given robot types by constructing an annotated *super roadmap*. The discrete solution is then used to partition the free space for each robot in a parallelizable trajectory optimization stage. Spatial partitioning is generalized to consider asymmetric collision constraints and a new method of temporal scaling is used to address varied dynamic limits. Our method scales well with respect to the number of robot types and with respect to the total number of robots.

This paper is an extended abstract of work submitted to IROS 2018.
All authors are with the Department of Computer Science, University of Southern California, Los Angeles, CA, USA. Email: {mjdebord, whoenig, ayanian}@usc.edu

Fig. 1. A robot team with 10 small UAVs (blue, 8 visible), 2 medium UAVs (red), 1 large UAV (green), and 2 ground robots (yellow) is tasked with navigating a cluttered environment.

Motion planning for heterogeneous robots have employed a variety of methods including optimization-based methods [2], graph-based methods [3], and reactive planning [4]. In contrast to the mentioned work, our method demonstrates better scalability as well as the ability to account for asymmetric collision constraints.

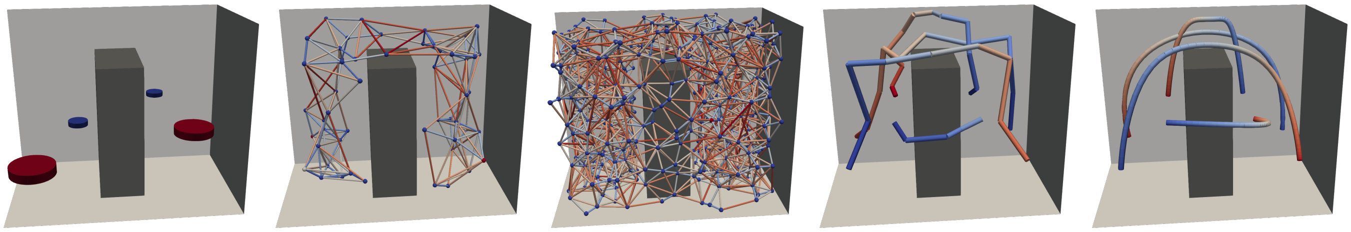
II. Approach

We generalize the hybrid planning approach to account for the additional difficulties with spatial and dynamic heterogeneity. In the discrete scheduling phase we first construct a *super roadmap* that is used for multi-agent path-finding. In the trajectory optimization phase we generalize the spatial partitioning and trajectory scaling steps. Fig. 2 shows an example of the entire approach for the case of two types of UAVs.

A. Heterogeneous Collision Model

Trajectory planning for heterogeneous teams requires the ability to account for non-uniform inter-robot collision constraints. For example, due to the downwash effect a large quadrotor is likely able to fly underneath a smaller one without difficulty, but the opposite is not true.

To account for these kinds of asymmetric collision constraints, we define independent collision volumes for each pair of types in the team. Specifically, for each pair of types we define a cylindrical collision volume $\mathcal{R}^{(k,l)}, k, l \in \{1 \ldots M\}$ that is parameterized by a tuple $\langle r, a, b \rangle$. The parameter r specifies the safe horizontal distance between the positions of robots that are of k, l types, and the parameters a, b specify the distance that a robot of type l must maintain above and below a robot of type k. This collision model is used to check for conflicts during the discrete scheduling phase, and to compute free-space partitions during the trajectory optimization phase.



(a) Initial configuration. (b) Roadmap for large UAVs. (c) Roadmap for small UAVs. (d) Discrete trajectories. (e) Continuous trajectories.

Fig. 2. Example with two small and two large rotorcraft UAVs, one of each type on each side of the obstacle. The UAVs are tasked with moving to goal locations on the opposite side of the obstacle. The smaller UAVs fly either above, or far below the larger UAVs to avoid downwash.

B. Discrete Scheduling

The discrete scheduling phase takes the robot starting positions, goal positions, and collision models as input and computes a piecewise linear path on a constructed roadmap. The schedule assigns a sequence of waypoints for each robot such that if all robots traverse their waypoints on fixed timesteps they are guaranteed to be collision-free.

Spatial coordination of a heterogenous team is complicated due to non-uniform spatial extents. Temporal coordination is also difficult because the different types of robots in the team may have different physical limits and so the original assumption that each robot can traverse an edge on the same fixed timestep is violated.

Both spatial and temporal coordination are achieved by constructing a *super roadmap* that is a disjoint union of M sub-roadmaps that are generated for each robot type. These sub-roadmaps are independently generated using the SPARS algorithm such that the ratio of average edge lengths between the sub-roadmaps reflect the robots relative velocity limits. For example, if a robot of type k is able to move twice as fast as a robot of type l, then the average length of the edges of the k type sub-roadmap are twice as long as the l type.

The super roadmap is then annotated with constraints that specify which vertices and edges can be simultaneously occupied without inter-robot collisions. The final computation of the schedule is done using the ECBS/C algorithm on the super roadmap.

C. Trajectory Optimization

The trajectory optimization stage takes the robot collision models and the computed discrete schedules and returns a smooth collision-free trajectory for each robot. This is done by first using the discrete schedule and robot collision models to partition the free space into collision-free volumes called *safe corridors* and then performing independent trajectory optimizations for each robot within its corridor.

Our method offers a new way to compute the safe corridors that takes into account the non-uniform spatial extents of the robots. The corridors are a set of K convex polyhedra where K is the number of timesteps in the discrete solution. The polyhedra are defined such that if every robot stays within their safe polyhedron for the corresponding timestep no collisions occur. Each polyhedron is computed as the intersection of $N-1$ half-spaces separating a robot from every other team member and N_{obs} half-spaces separating the robot from obstacles.

Each half-space separating a pair of robots of types k, l is defined by a maximum margin hyperplane that separates a trajectory slice of the k type robot from the convex hull defined by sweeping the $\mathcal{R}^{(k,l)}$ collision geometry along the corresponding trajectory slice of the l type robot. The computation of the half-spaces separating robots from the environment is modified compared to our previous work such that each robot can have a separate specified size.

To ensure that the dynamic limits of all robots in the team are enforced we utilize the differential flatness property and uniformly scale all trajectories by a constant factor. We compute this stretching factor for each robot using a binary search approach. The maximum of all those factors is then applied uniformly to all trajectories.

III. Experiments

We analyze the scalability of our approach in simulation. First, we fix the total number of robots at 50 and vary M from 2 to 10 types. Roadmap conflict annotation scales roughly quadradically with M while the scheduling and optimization steps stay roughly constant at a total time of $62\,\mathrm{s}$. Second, we fix both N and M but adjust the ratio of maximum velocity limits. We find that if the ratio is large the discrete solution requires many more timesteps and thus results in longer run times for both scheduling and optimization.

We set up a cluttered obstacle course that a team of 15 robots must swap sides across. The team consists of two ground robots and three types of quadrotors with masses ranging from $30\,\mathrm{g}$ to $500\,\mathrm{g}$. Generating sub-roadmaps for each of the four types takes roughly $300\,\mathrm{s}$ total. We merge the individual roadmaps into our super roadmap and annotate it with potential conflicts in $4\,\mathrm{s}$. Discrete scheduling, four iterations of spatial partitioning and trajectory optimization, and temporal scaling takes about $19\,\mathrm{s}$ total. Fig. 1 shows a snapshot of the physical execution of the trajectories. A video of the entire execution can be seen at `https://youtu.be/OzXUV4GQ7Qs`.

References

[1] W. Hönig *et al.*, “Trajectory planning for quadrotor swarms,” *IEEE T-RO, Special Issue on Aerial Swarm Robotics*, 2018, to appear.

[2] D. R. Robinson *et al.*, “An efficient algorithm for optimal trajectory generation for heterogeneous multi-agent systems in non-convex environments,” *IEEE RA-L*, vol. 3, no. 2, pp. 1215–1222, 2018.

[3] J. Butzke *et al.*, “Planning for a ground-air robotic system with collaborative localization,” in *ICRA*, 2016, pp. 284–291.

[4] J. Alonso-Mora *et al.*, “Collision avoidance for aerial vehicles in multi-agent scenarios,” *Autonomous Robots*, vol. 39, no. 1, pp. 101–121, 2015.