TRAJECTORY PLANNING FOR QUADROTOR SWARMS

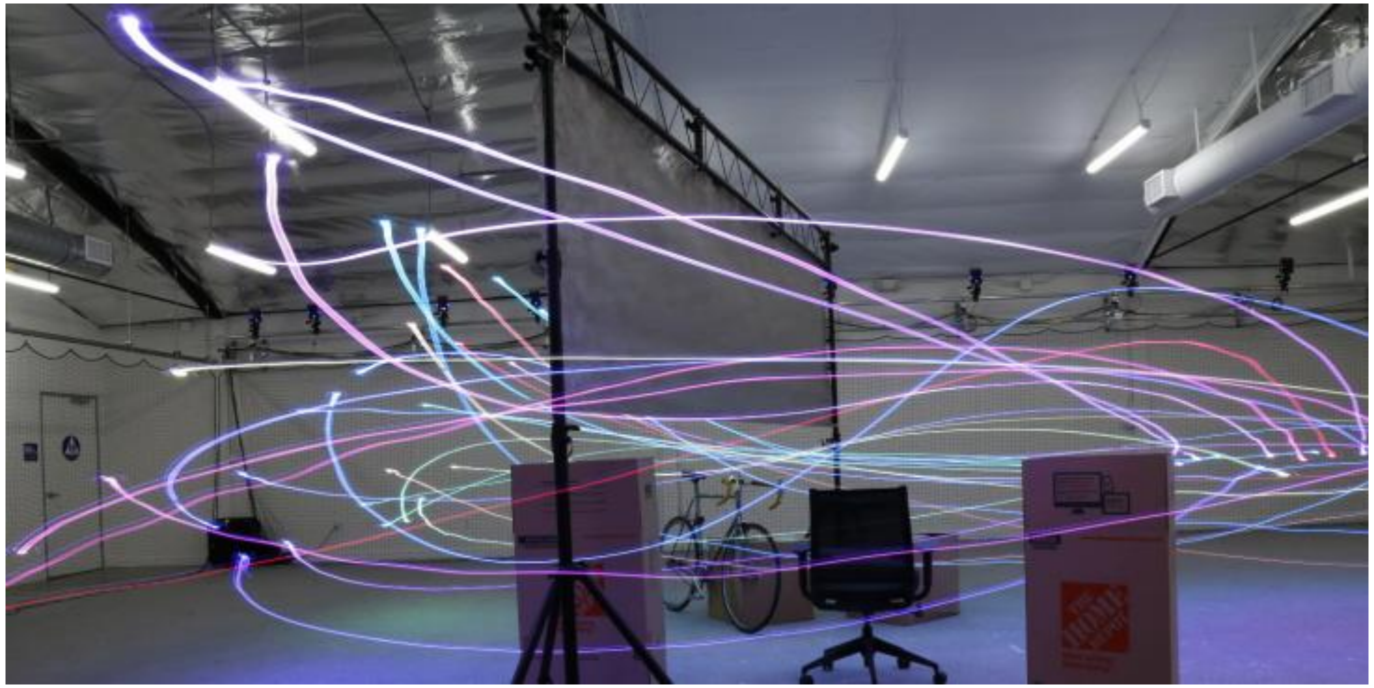
Tomáš Čelko

PRESENTATION STRUCTURE

- Task specification
- Goals
- Related Approaches
- Solution
  1. Roadmap generation
  2. Conflict annotation
  3. Discrete planning
  4. Continuous refinement
- Experiments
  - Trying different methods
  - Trying different environments

THE TASK DESCRIPTION

- N drones
- Bounded environment
- Dense obstacles
- Each robot has collision model
- Labeled vs unlabeled



GOALS AND DESIRED PROPERTIES

Find time T and function $f^i \colon [0, T] \rightarrow R^3$ and aim for:

- Soundness
  - No collisions
- Completeness
- Optimality
  - Time (makespan)
  - Energy (sum of costs)
- Computational performance
- Physical plausability

RELATED APPROACHES

- Carthesian product of simple robot planning
  - Robot-robot collisions = space obstacle
  - Computationally infeasible
- Graph search perspective
  - Reasonable for high congestion problems
  - Piecewise linear path
- Continuous settings
  - Solving one big optimization task
  - Tightly coupled
  - Usually do not scale well

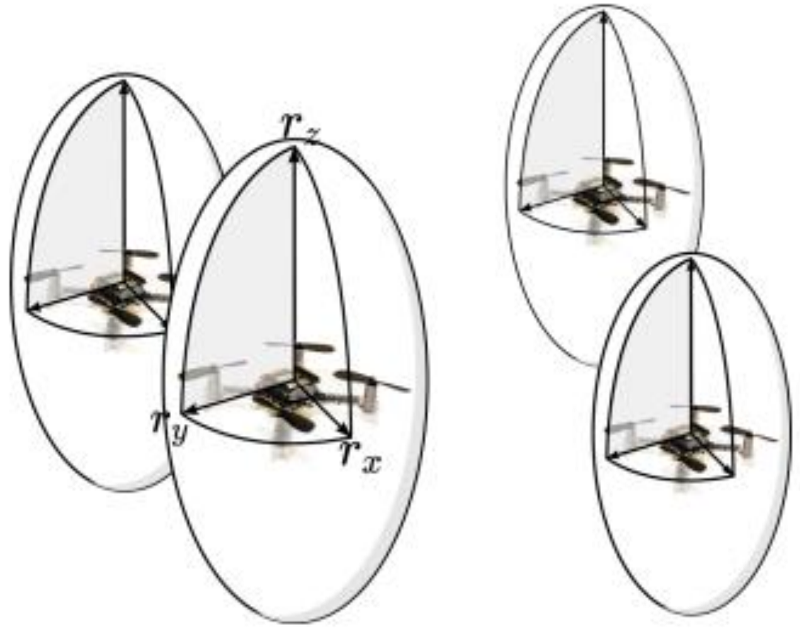
ENVIRONMENT

- Environment bounded
- Obstacles are convex polytopes
- Drones often identified by 6 coordinates
  - But authors use only 3
- Collision model is an ellipsoid

$$\mathcal{R}_{\mathcal{R}}(q) = \{Ex + q : \|x\|_2 \leq 1\}$$

where $E = \mathbf{diag}(r_x, r_y, r_z)$.

- Downwash effect
- Continuity up to 4th derivative



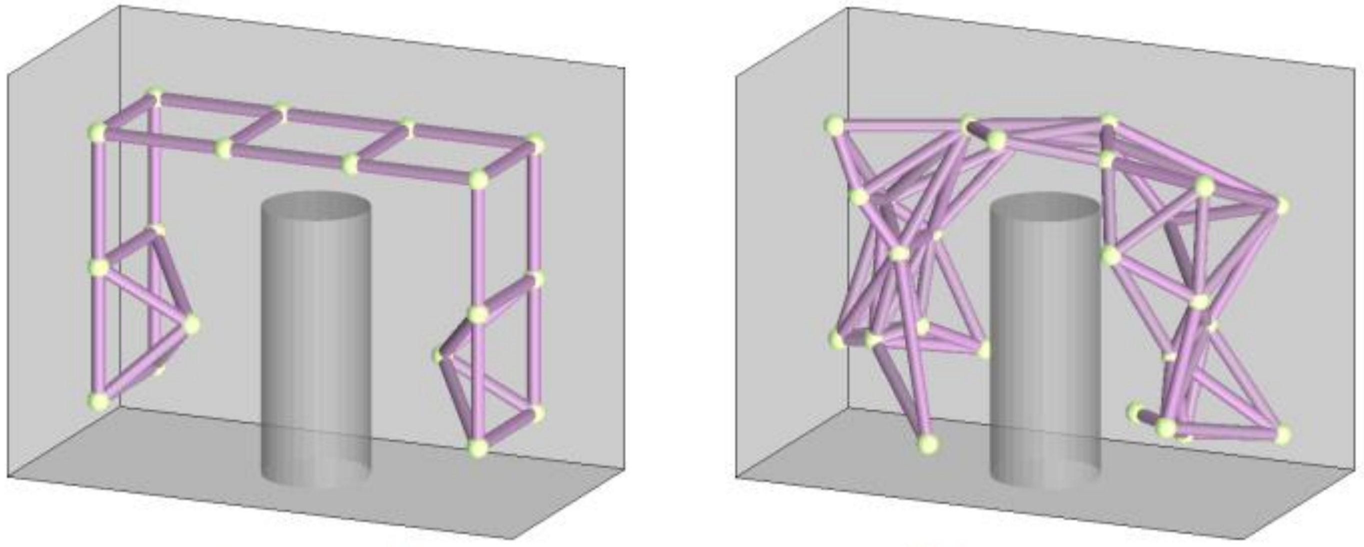
1.ROADMAP GENERATION

- Undirected connected graph
- We define $loc : V \rightarrow R^3$
- Dispersion parameter = diameter of the largest sphere not containing any vertex
- Desired properties
  - Connected
  - Well approximating
  - Sparse (otherwise many robot-robot conflicts)

1.ROADMAP GENERATION (2)

- Two approaches:
  - Grid-based roadmap
    - 6-connected
    - Optimality when dispersion approaches zero
  - Sparse roadmap spanners
    - Spanner (graph theory) = subgraph with dropped

      edges preserving shortest distances up to a constant factor
    - First create dense graph
    - Drop edges and vertices



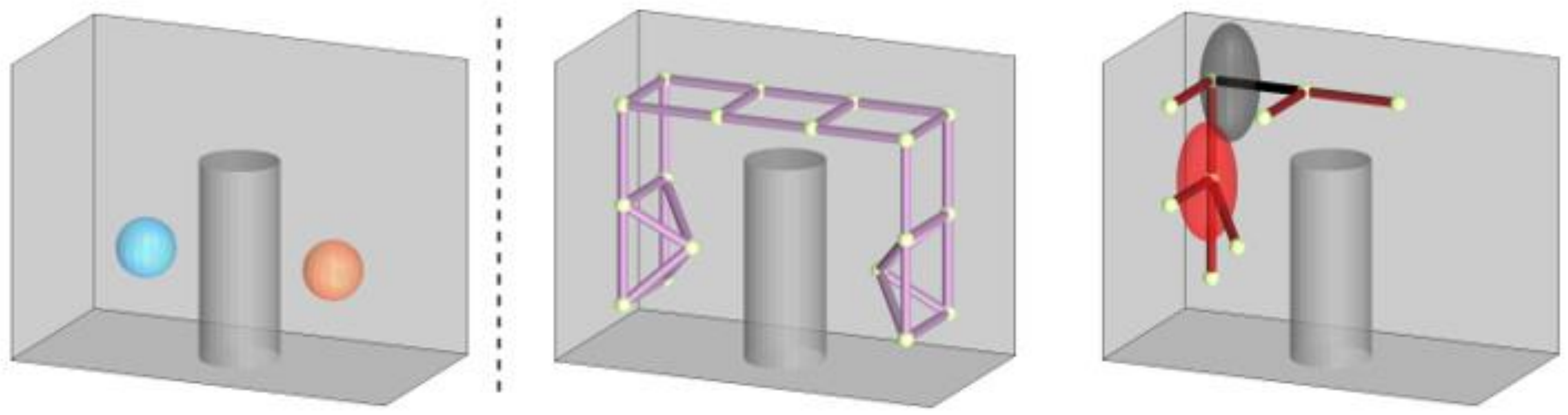
(a) Grid-based roadmap. (b) SPARS-based roadmap.

2.CONFLICT ANNOTATION

Output = the graph where edges and vertices are annotated with a conflict set

Conflict types:

- Vertex-vertex (VV)
- Edge-vertex (EV)
- Edge-edge (EE)



2.CONFLICT ANNOTATION (2)

- Two conflict models:
  - SWEPT collision model
    - Simple
    - Allows arbitrary velocity profiles
  - FCL collision model
    - Define $mot(e,t);\ t \in [0,1]$
    - Uses known velocity profiles (such as constant)
    - Why wouldn't this work with arbitrary velocities?
- Quadratic time complexity

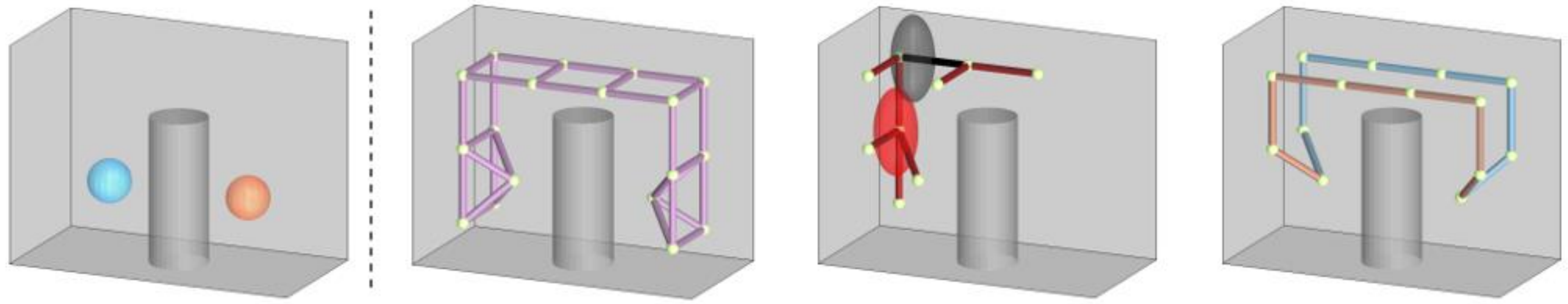
$$conEE'(e) = \{d \in \mathcal{E}_E \mid \exists t \in [0,1]$$
$$\mathcal{R}_\mathcal{R}(mot(d,t)) \cap \mathcal{R}_\mathcal{R}(mot(e,t)) \neq \emptyset\}$$
$$conEV'(e) = \{u \in \mathcal{V}_E \mid \exists t \in [0,1]$$
$$\mathcal{R}_\mathcal{R}(loc(u)) \cap \mathcal{R}_\mathcal{R}(mot(e,t)) \neq \emptyset\}$$

3.DISCRETE PLANNING

- We are given an annotated roadmap
- Traverse edge / wait actions
- Schedule for robot = $(loc(u_0^i),\ loc(u_1^i), \ldots loc(u_K^i))$
- Robot travels on a line segment along edges
- We formulate additional constraints => MAPF/C

robot

timestep



MAPF/C PROPERTIES

Authors formulated 8 properties that should hold in MAPF/C:

- P1: Start vertices
- P2: Unique goal locations
- P3: Stay at vertex/move along the edge
- P4: No vertex collisions
- P5: No edge collisions
- P6: No ConVV conflicts
- P7: No ConEV conflicts
- P8: No ConEE conflicts

Recap:

$A \leq_p B \leftrightarrow \exists f\ \forall x\colon x \in A \leftrightarrow f(x) \in B; f\ polynomial.$

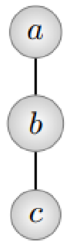
$\mathrm{A} \leq_p \mathrm{B}\ \&\ \mathrm{A}\ is\ NP$ hard $\rightarrow B\ is\ NP\ hard$

MAPF/C PROPERTIES (2)

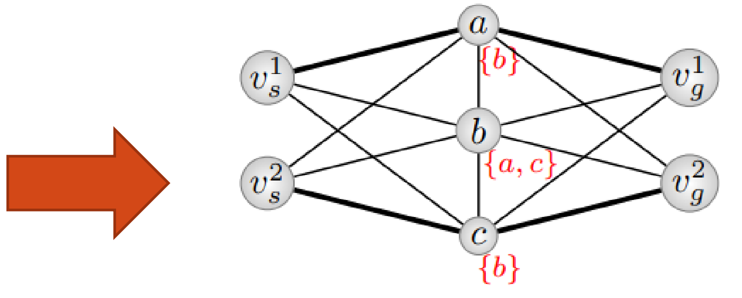
Theorem 1: *Solving **labeled** MAPF/C optimally with respect to **cost** or **makespan** is NP-hard.*

- Proof: Follows from NP-hardness of the same problem for labeled MAPF.

- Theorem 2: *Solving **unlabeled** MAPF/C optimally with respect to **makespan** is NP-hard.*
  - *Proof:* Reduction from independent set problem (G_{MIS}, k_{MIS}).
  - How do we construct IS of size k from the plan?



(a) $\mathcal{G}_{MIS}$



(b) $\mathcal{G}_{MAPF/C}$

UNLABELED MAPF/C

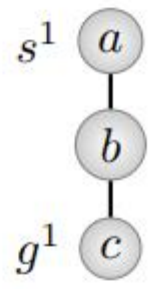
- MAPF Tractable in polynomial time

  1. Generate time-expanded graph
  2. Find maximal flow
  3. Reconstruct the plan from the maximal flow

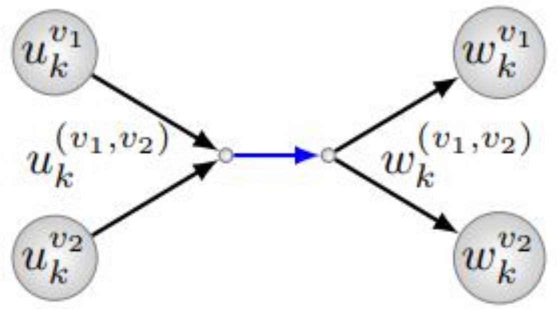
- Q: Why wouldn‘t this work in the labeled case?
- Q: And what about unlabeled MAPF/C ?

GENERATING TIME EXPANDED GRAPH (P1-P5)

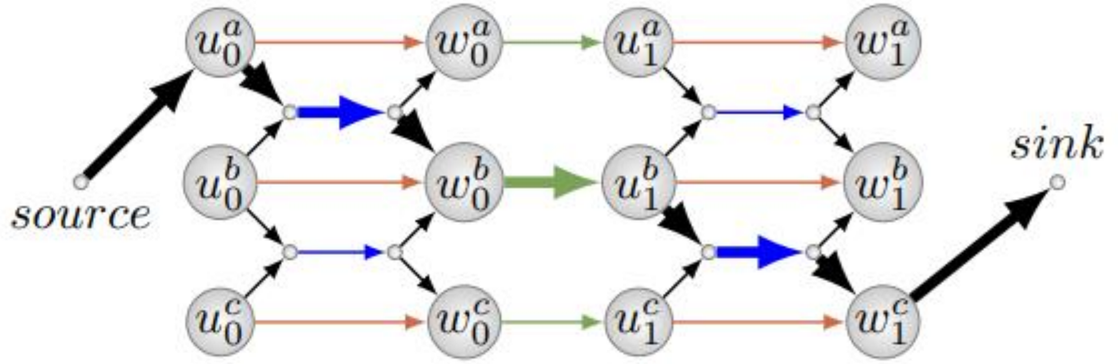
- For each vertex *a* in roadmap and timestep we add two vertices and an edge
- For each edge and timestep we add gadget (P5 – no swapping)
- Connect consecutive timesteps for each vertex (P4 – no vertex collisions)
- Add source and sink
- Actions stay in vertex/move along edge



(a) $\mathcal{G}_E$



(b) "Gadget" for flow-graph construction.

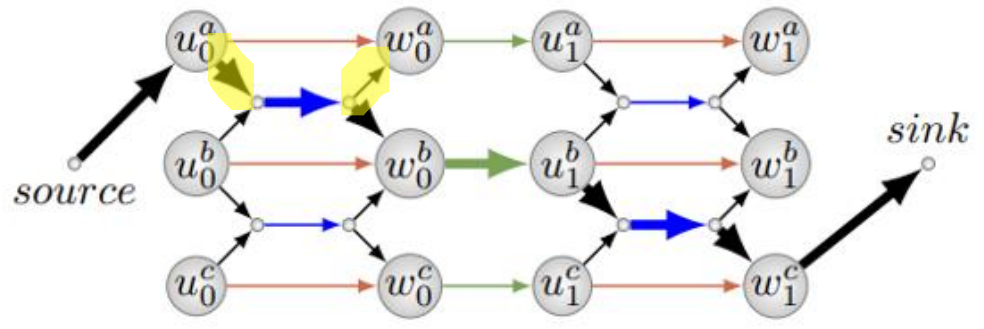


(c) $\mathcal{G}_F$ with $K = 2$.

ADDING ADDITIONAL CONSTRAINTS (P6-P8)

- It is not possible to encode generalized conflicts into the time expanded graph itself => adding constraints instead
- All necessary constraints can be handled via conflicting edges annotations: $con : E \to 2^E$ (conflict set)
  - VV generalized conflicts (P6)
  - EE generalized conflicts (P7)
  - EV generalized conflicts (P8)
- Blue edges should not be used for waiting

  problem => add pairs to conflict sets
- Now we have graph + con(e)



(c) $\mathcal{G}_F$ with $K = 2$.

FINDING OUT ACTUAL MAKESPAN

- To construct graph we need to know the number of timesteps
- How to find it?
- Two step approach:
  - Solve classic unlabeled MAPF instance (no generalized conflicts, only checking feasibility)

    For k = 1,2,4,8.. apply Edmond's Karp algorithm to get lower bound k_{min}
  - Linear search with all constraints (ILP)

```
For k = $k_{min} + 1; k++$
    Solution = SolveILP(G(k))
    If solution != Fail
            return solution
```

FINDING MAXIMUM FLOW

- We are given graph and conflicting pairs of edges
- Represent problem as ILP (NP-hard, unlabeled case -> we cannot do much better - Theorem 2)
- $z_{u,v}$ is flow on edge (u,v)

$$\begin{aligned}
\text{maximize} \quad & \sum_{(source,v)\in\mathcal{E}_F} z_{(source,v)} \\
\text{subject to} \quad & \sum_{(u,v)\in\mathcal{E}_F} z_{(u,v)} = \sum_{(v,w)\in\mathcal{E}_F} z_{(v,w)} \ \forall v \in \mathcal{V}'_F \\
& z_e + \sum_{e' \in con(e)} z_{e'} \leq 1 \quad \forall e \in \mathcal{E}_F
\end{aligned}$$

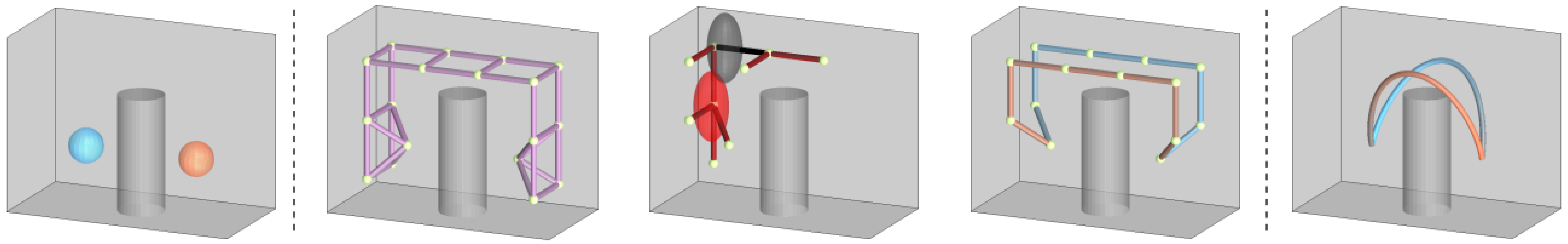
LABELED MAPF/C

- Conflict-Based search – resolving conflicts one by one
  - High level search
  - Low level search
  - Extension to MAPF/C is simple

- ECBS = Suboptimal variant of CBS
  - Searches binary conflict tree
  - Uses focal search (variant of A*) – frontier satisfies suboptimality factor

TRAJECTORY OPTIMIZATION

- Given discrete plan = sequence of waipoints
- Convert it to continuous and smooth trajectory
- Approach:
  - Create safe corridors for each drone
  - Optimize trajectory within given corridor
  - Iteratively refine trajectories



FINDING SAFE CORRIDORS

- Corridor = sequence of convex polyhedra P_t^d
- Safe corridor = drone can fly in it without collisions
- P_t^d is intersection of n halfspaces
- Notes:
  - P_t^d is not necessarily bounded
  - P_t^d and P_{t+1}^d can and usually will overlap

$$\ell_k^i \subset \{x : \alpha_k^{(i,j)^T} x < \beta_k^{(i,j)}\}$$
$$\ell_k^j \subset \{x : \alpha_k^{(i,j)^T} x > \beta_k^{(i,j)}\}$$

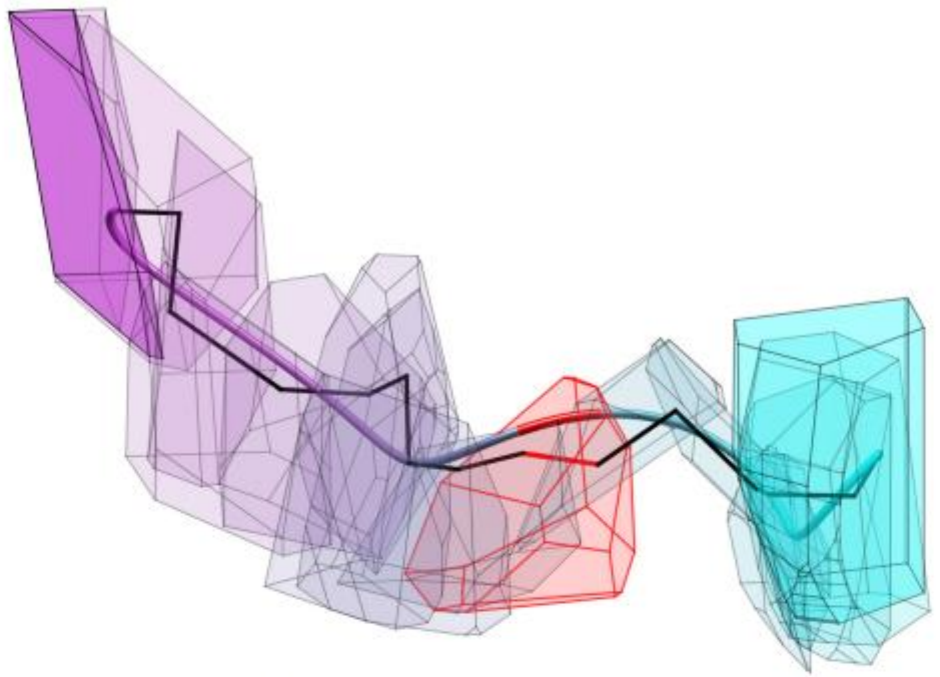


Fig. 7. Safe corridor for one robot over entire flight. Corridor polytopes are colored by timestep. *Black line*: underlying discrete plan. *Shaded tube*: smooth trajectory after first iteration of refinement. *Highlighted in red*: polytope, discrete plan segment, and trajectory polynomial piece for a single timestep.

FINDING SAFE CORRIDORS (2)

- Our planes only separate line segments, We need to account for model and trajectory curve.

$$\alpha_k^{(i,j)^T} f^i(t) < \beta_k^{(i,j)} - \|E\alpha_k^{(i,j)}\|_2 \quad \forall t \in [t_{k-1}, t_k]$$
$$\alpha_k^{(i,j)^T} f^j(t) > \beta_k^{(i,j)} + \|E\alpha_k^{(i,j)}\|_2 \quad \forall t \in [t_{k-1}, t_k]$$

- Use SVM with margin $2\|E\alpha\|$ to find the hyperplanes (a lot of SVM subprograms $O(KN^2 + KNN_{obs})$ −> parallelizable with redundancy = 2)

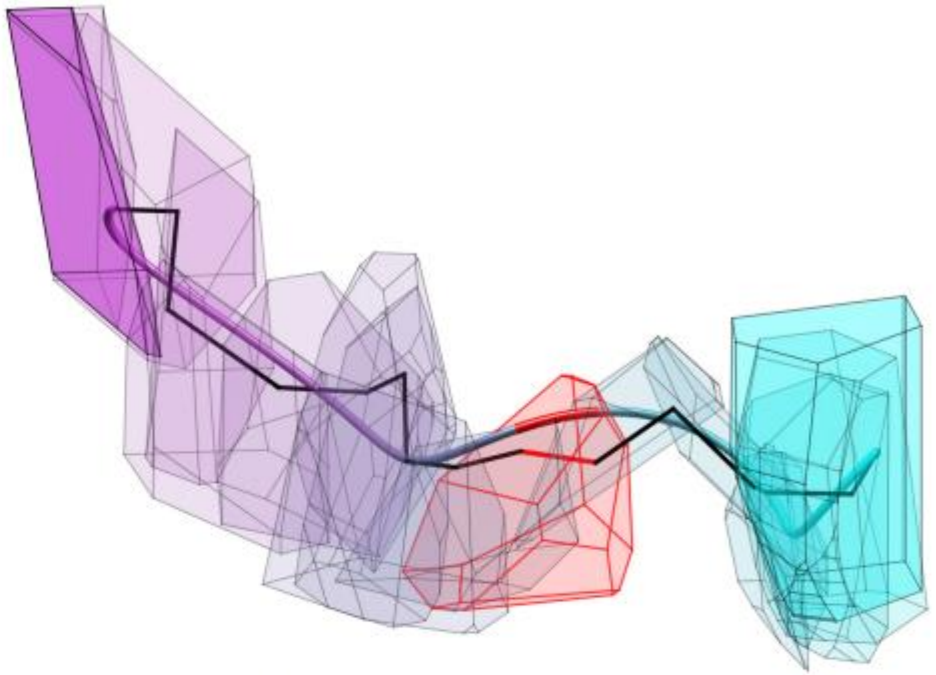
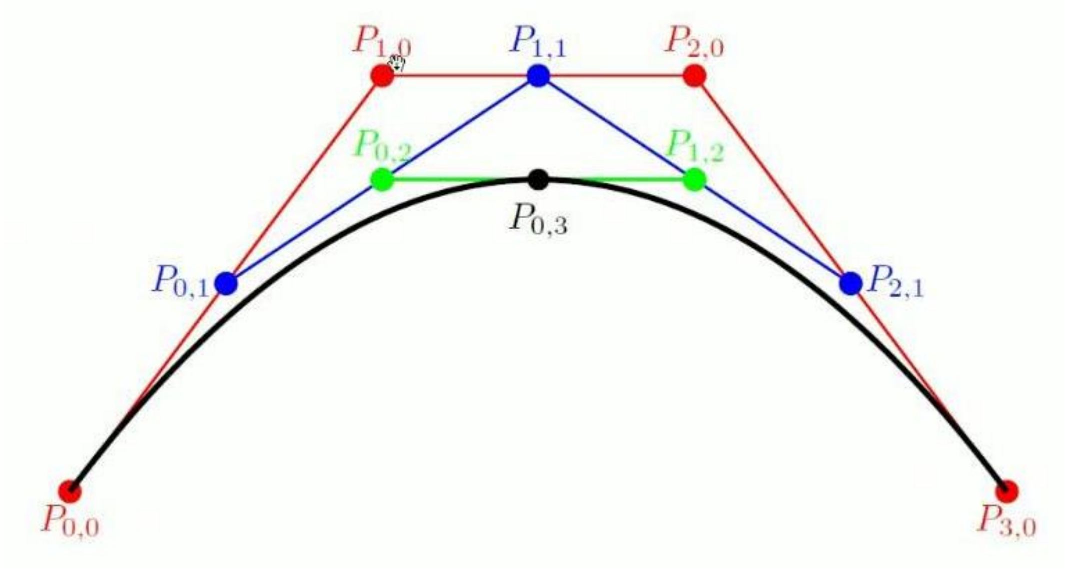


Fig. 7. Safe corridor for one robot over entire flight. Corridor polytopes are colored by timestep. *Black line*: underlying discrete plan. *Shaded tube*: smooth trajectory after first iteration of refinement. *Highlighted in red*: polytope, discrete plan segment, and trajectory polynomial piece for a single timestep.

TRAJECTORY AND PROBLEM REPRESENTATION



- We have safe corridors
- Choose trajectory basis:
- A) Piecewise **Polynomial** -> polynomial inside polyhedron is non-convex constraint

$$p(t) = a_0 + a_1 t + a_2 t^2 + \cdots + a_D t^D$$

- B) Piecewise **Bezier curves**
  - can be represented by convex constraints
  - not passing through control points, lies in their convex hull

$$f(t) = b_{0,D}(t) y_0 + b_{1,D}(t) y_1 + \cdots + b_{D,D}(t) y_D$$

$$b_{\nu,n}(x) = \binom{n}{\nu} x^{\nu} (1-x)^{n-\nu}, \quad \nu = 0, \ldots, n,$$

TRAJECTORY OPTIMIZATION

- Objective function: $$\text{cost}(f^i) = \sum_{c=1}^{C} \gamma_c \int_0^T \left\| \frac{d^c}{dt^c} f^i(t) \right\|_2^2 dt$$

- Can be rewritten as a quadratic program:

$$\begin{array}{ll} \text{minimize} & \mathbf{y}^T (B^T Q B) \mathbf{y} \\ \text{subject to} & y^i_{k,d} \in \mathcal{P}^i_k \quad \forall\, i, k, d \\ & f^i(0) = s^i, \;\; f^i(T) = g^{\phi(i)} \\ & f^i \text{ continuous up to derivative } C \end{array}$$

- We have n such programs – parallel execution

ITERATIVE REFINEMENT & POSTPROCESSING

- Continuous trajectories often deviate heavily from discrete plan
- We can iteratively improve the trajectory:

1. Sample continuous trajectories
2. Replan
3. Perform continuous optimization
4. If not done, go to 1

Postprocessing:

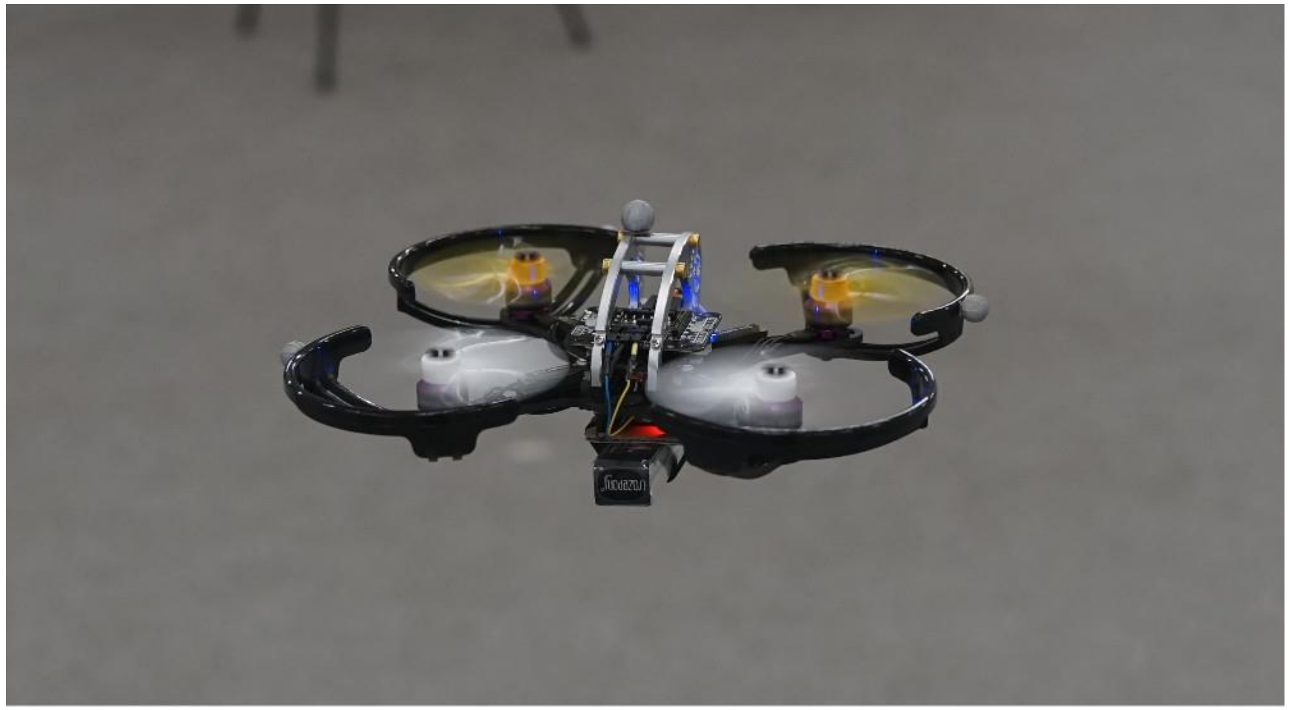
FCL collision model + arbitrary velocity profiles => prevent vertex collisions (by edge subdivision)

EXPERIMENTS

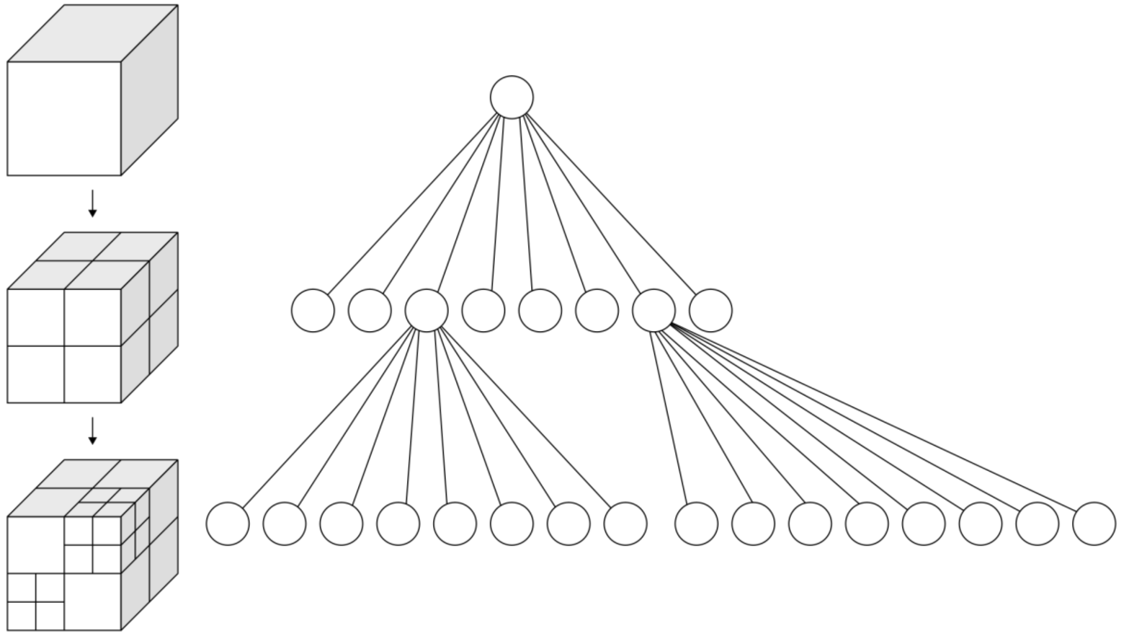
- Simulations (200 drones)
- Swarm of quadrotors
- Crazyswarm

Trying various approaches for:

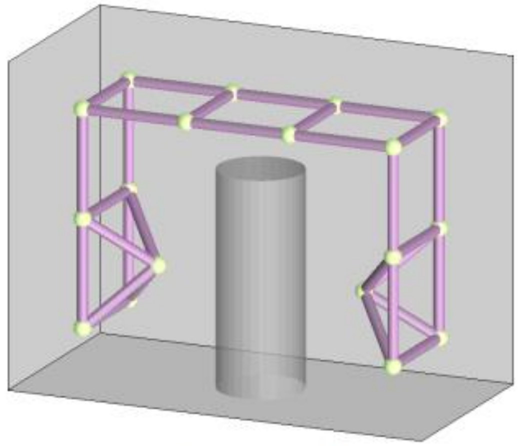
- Mapping
- Roadmap generation
- Conflict annotation
- Discrete planning
- Continuous optimization



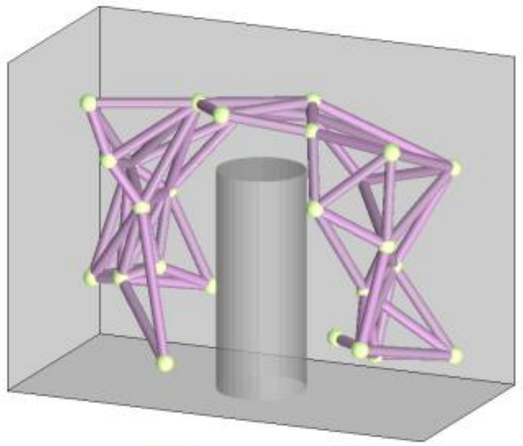
MAPPING AND ROADMAP EXPERIMENTS



- Mapping
  - Oct-tree
  - Choose correct leaf size (≈dispersion)
- Roadmap generation
  - 6 neighbor grid
  - SPARS



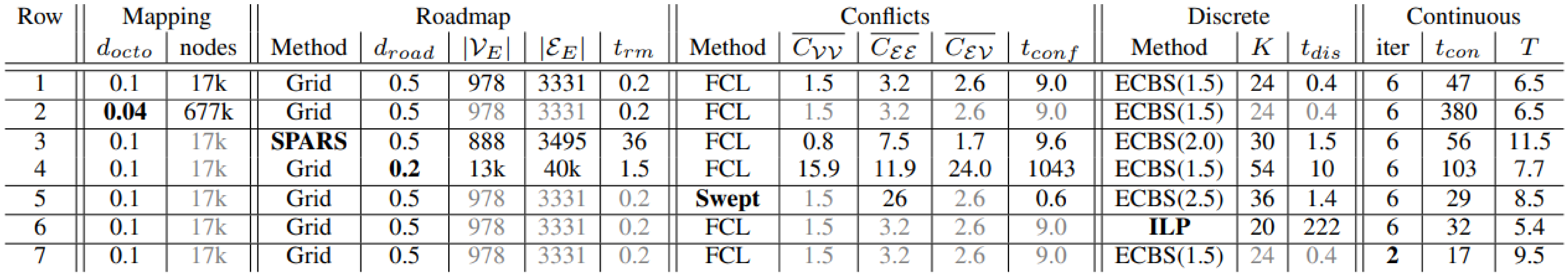
(a) Grid-based roadmap.



(b) SPARS-based roadmap.

CONFLICTS AND DISCRETE PLANNING

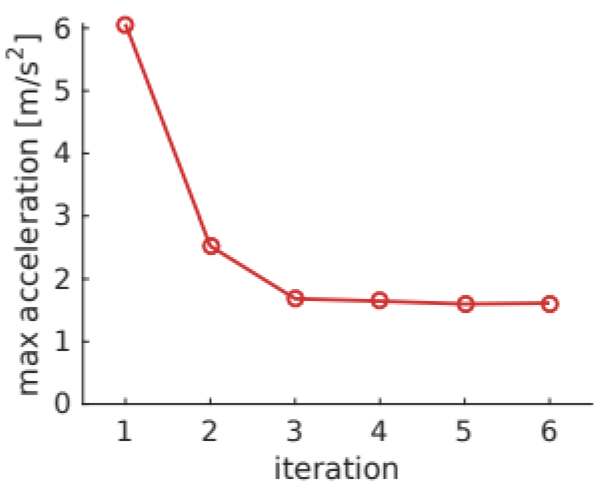
- Collision model
  - FCL
  - SWEPT
- Discrete planning
  - Labeled – ECBS
  - Unlabeled
    - Labeling + ECBS
    - ILP

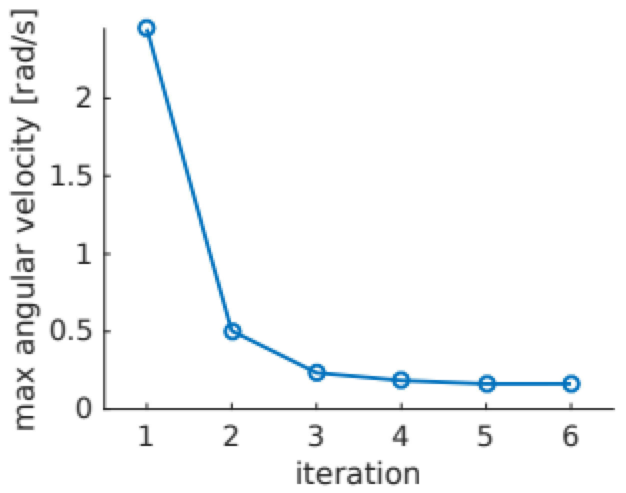


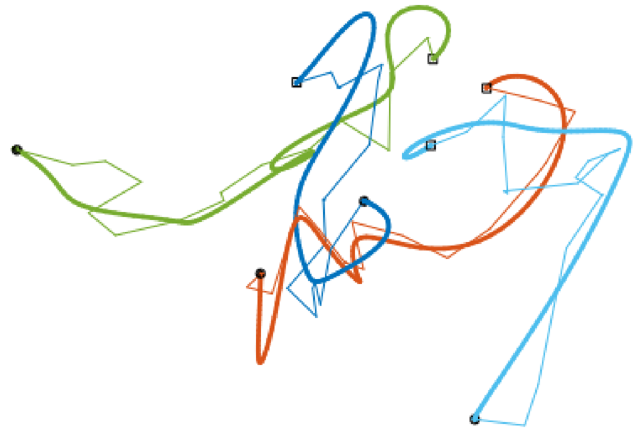
Row	Mapping		Roadmap					Conflicts					Discrete			Continuous		
	d_{octo}	nodes	Method	d_{road}	$\lvert\mathcal{V}_E\rvert$	$\lvert\mathcal{E}_E\rvert$	t_{rm}	Method	$\overline{C_{\mathcal{V}\mathcal{V}}}$	$\overline{C_{\mathcal{E}\mathcal{E}}}$	$\overline{C_{\mathcal{E}\mathcal{V}}}$	t_{conf}	Method	K	t_{dis}	iter	t_{con}	T
1	0.1	17k	Grid	0.5	978	3331	0.2	FCL	1.5	3.2	2.6	9.0	ECBS(1.5)	24	0.4	6	47	6.5
2	**0.04**	677k	Grid	0.5	978	3331	0.2	FCL	1.5	3.2	2.6	9.0	ECBS(1.5)	24	0.4	6	380	6.5
3	0.1	17k	**SPARS**	0.5	888	3495	36	FCL	0.8	7.5	1.7	9.6	ECBS(2.0)	30	1.5	6	56	11.5
4	0.1	17k	Grid	**0.2**	13k	40k	1.5	FCL	15.9	11.9	24.0	1043	ECBS(1.5)	54	10	6	103	7.7
5	0.1	17k	Grid	0.5	978	3331	0.2	**Swept**	1.5	26	2.6	0.6	ECBS(2.5)	36	1.4	6	29	8.5
6	0.1	17k	Grid	0.5	978	3331	0.2	FCL	1.5	3.2	2.6	9.0	**ILP**	20	222	6	32	5.4
7	0.1	17k	Grid	0.5	978	3331	0.2	FCL	1.5	3.2	2.6	9.0	ECBS(1.5)	24	0.4	**2**	17	9.5

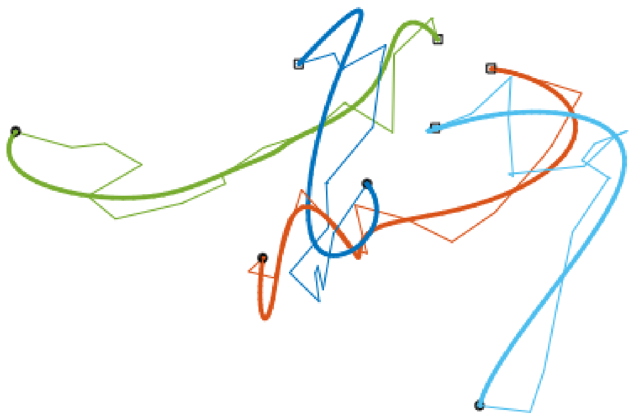
CONTINUOUS OPTIMIZATION

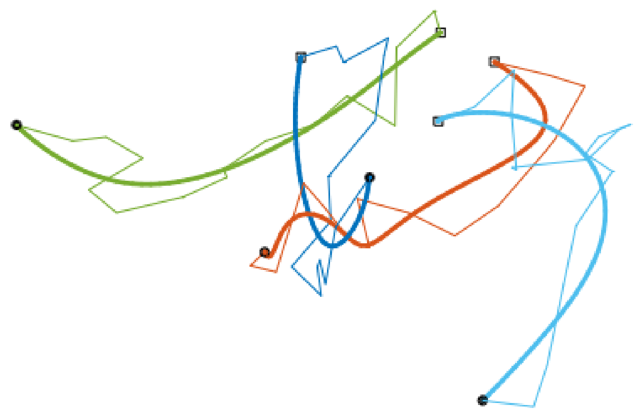
- Examined iterative refinement
- Lowering acceleration peaks





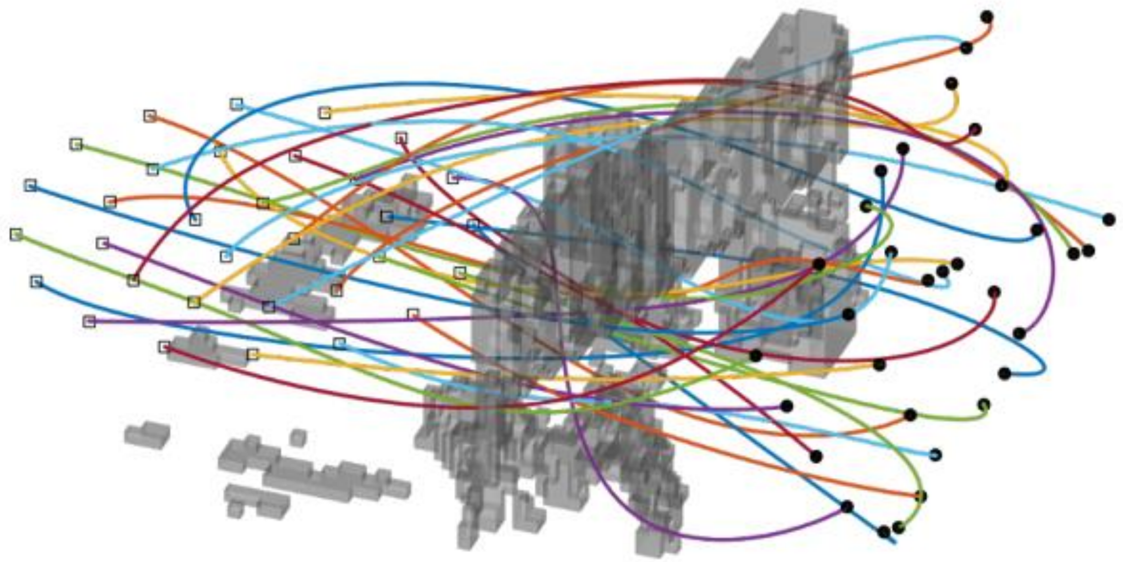

(a) 1 iteration


(b) 2 iterations


(c) 6 iterations

OTHER EXPERIMENTS

					Roadmap			Conflicts				Discrete		Continuous			
Example	labeled	N	Env. Size	occupied	$\lvert\mathcal{V}_E\rvert$	$\lvert\mathcal{E}_E\rvert$	t_{rm}	$\overline{C_{\mathcal{V}\mathcal{V}}}$	$\overline{C_{\mathcal{E}\mathcal{E}}}$	$\overline{C_{\mathcal{E}\mathcal{V}}}$	t_{conf}	K	t_{dis}	$t_{1(hp)}$	$t_{1(qp)}$	t_{con}	T
Flight Test	No	32	$9 \times 5.5 \times 2.2$	4 %	873	3430	50	0.8	28	1.8	0.8	28	0.5	2.0	3.9	28	6.5
Wall32	No	32	$7.5 \times 6.5 \times 2.5$	6 %	921	3536	36	0.8	27.7	1.7	0.8	41	4.5	1.6	3.5	35	11.6
Maze50	No	50	$10 \times 6.5 \times 2.5$	30 %	1045	3221	82	1.1	24.2	2.2	0.7	48	4.4	7.3	12.3	133	16.3
Sort200	No	200	$14.5 \times 14.5 \times 2.5$	31 %	3047	7804	85	1.1	18.8	2.0	3.5	39	25	21	34	411	11.7
Swap50	Yes	50	$7.5 \times 6.5 \times 2.5$	6 %	869	3371	34	0.8	27	1.6	0.7	48	15	3.4	9.4	78	14.4


(a) Full 32-robot trajectory plan after six iterations of refinement. The start and end positions are marked by squares and filled circles, respectively.

(b) Picture of the final configuration after the test flight. A video is available as supplemental material.

SOURCES

- W. Hönig, J. A. Preiss, T. K. S. Kumar, G. S. Sukhatme and N. Ayanian, "Trajectory Planning for Quadrotor Swarms," in *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 856-869, Aug. 2018, doi: 10.1109/TRO.2018.2853613.
- D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2520-2525, doi: 10.1109/ICRA.2011.5980409.

THANK YOU FOR YOUR ATTENTION

QUESTIONS?