Perception-Aware Multiagent Trajectory Planner for UAVs Using Imitation Learning

Kota Kondo, AeroAstro

kkondo@mit.edu

May 8, 2023

Background 1: What are UAVs?

- UAVs (Unmanned Aerial Vehicles) and Drones
  - Commercial use
    - Video/photo
    - Package delivery
    - New mobility?
- Trajectory Planner
  - Control where they will go
  - Complex problem
    - Surrounding environment changes
    - More agents more complex

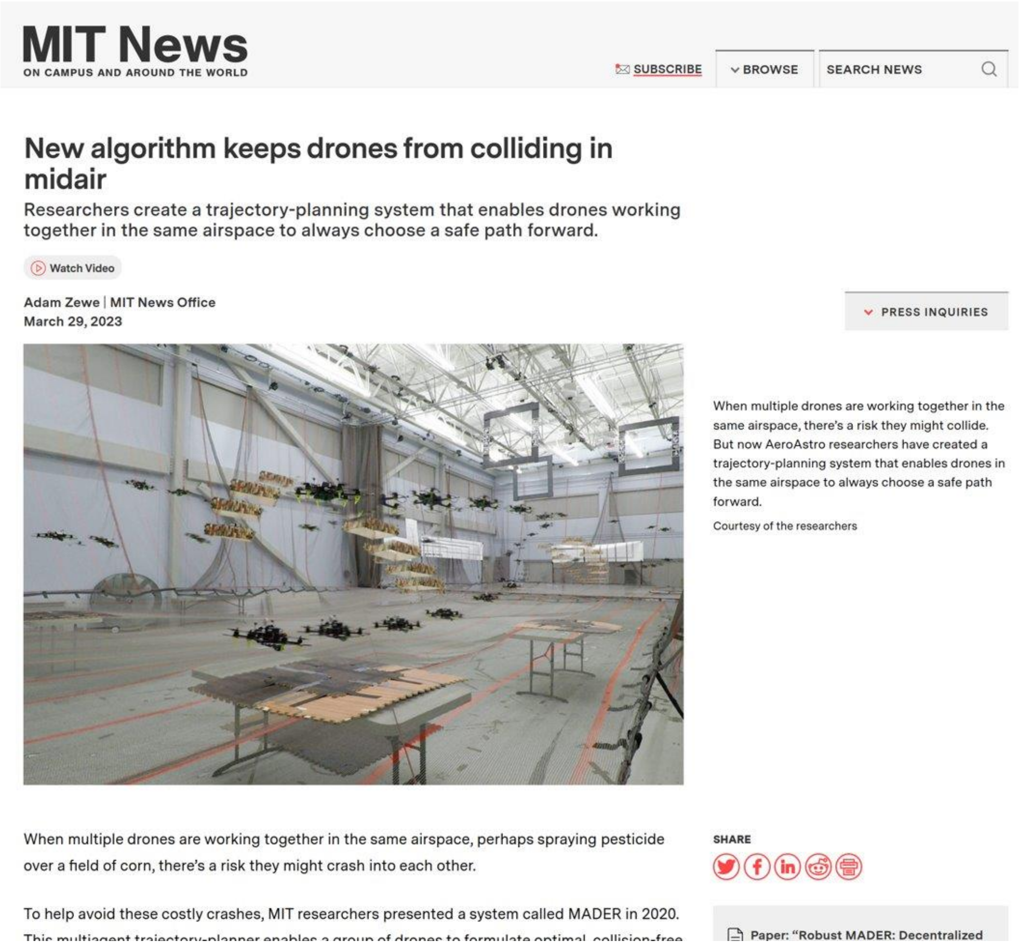
Background 2: Multiagent & Perception-aware

- Multiagent traj. planning
  - Decentralized vs. Centralized
  - Asynchronous vs. Synchronous

Table 1. Multiagent Trajectory Planner Category

	Synchronous	Asynchronous
Centralized	Not Scalable	Not Possible
Decentralized	Somewhat Scalable	**Most Scalable** (our approach)

- Perception-aware algorithm
  - Onboard sensing
  - Plans traj. depending on the env.



MIT News
ON CAMPUS AND AROUND THE WORLD

SUBSCRIBE | BROWSE | SEARCH NEWS

New algorithm keeps drones from colliding in midair

Researchers create a trajectory-planning system that enables drones working together in the same airspace to always choose a safe path forward.

Watch Video

Adam Zewe | MIT News Office
March 29, 2023

PRESS INQUIRIES

When multiple drones are working together in the same airspace, there's a risk they might collide. But now AeroAstro researchers have created a trajectory-planning system that enables drones in the same airspace to always choose a safe path forward.

Courtesy of the researchers

When multiple drones are working together in the same airspace, perhaps spraying pesticide over a field of corn, there's a risk they might crash into each other.

SHARE

To help avoid these costly crashes, MIT researchers presented a system called MADER in 2020.

Paper: "Robust MADER: Decentralized

Background 2: Multiagent & Perception-aware

Table 2. State-of-the-art UAV Trajectory Planners

Method	Multiagent	Perception-aware
EGO-Swarm [31]	Yes	No
DMPC [10]		
MADER [22]		
decMPC [26]		
RMADER [9]		
Raptor [30]	No	Yes
Time-opt [19]		
PANTHER [23]		
PA-RHP [29]		
Deep-PANTHER [24]		
Proposed approach	Yes	Yes

Background 3: Opt-based vs. IL-based

- Optimization-based
  - Solve optimization problem
    - Optimal traj. generation
    - Slow
    - Not scalable
- Imitation Learning (IL)-based
  - Imitate expert (usually opt-based) trajectory planner
    - Close-to-optimal
    - Fast
    - Scalable

	Optimal?	Computation	Scalability
Opt-based	**Yes**	**slow**	**No**
IL-based	**Close-to-optimal**	**fast**	**Yes**

Background 3: Opt-based vs. IL-based

Table 3. State-of-the-art Perception-aware Obstacle Tracking Trajectory Planners

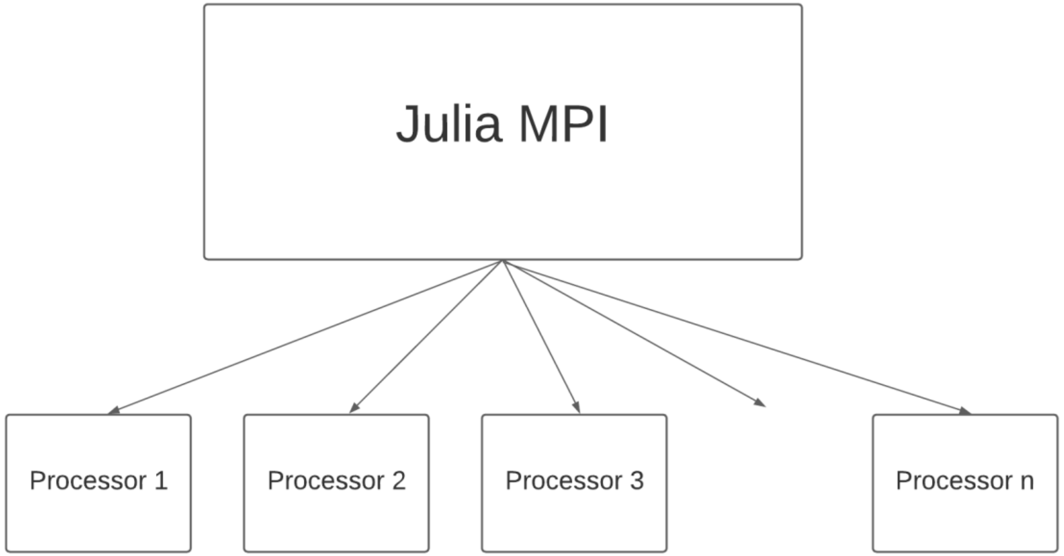
Method	Tracking Multi-obstacles	Multi-agents	Trajectory	Planning
[21]	No	No	Only Position	Optimization-based (slow & not scalable)
[14]	No	No	Position & Yaw	Optimization-based (slow & not scalable)
PANTHER / PANTHER* [23, 24]	No	No	Position & Yaw	Optimization-based (slow & not scalable)
Deep-PANTHER [24]	No	No	Only Position [1]	IL-based (faster & scalable)
Expert	Yes	Yes	Position & Yaw	Optimization-based (slow & not scalable)
Student (proposed)	Yes	Yes	Position & Yaw	IL-based (faster & scalable)

Motivation

- Want to create the first "Perception-aware Multiagent traj. Planner using Imitation Learning"
  - Perception information
    - Flexible trajectory planning in real-world
  - Multiagent
    - Large-scale task
  - Imitation Learning
    - Fast

Julia MPI for IL (Behavior Cloning)

- MPI for fast data (trajs) collection
  - Parallelize data collection process for trajectory behavior cloning
  - Each processor generates expert trajectories
  - Collected 10K trajs (48606 seconds)



Julia MPI for IL (Behavior Cloning)

- Performance Comparison
  - Logged 100 trajectories collection speed

Table 2. Data Collection Time for 100 trajectories

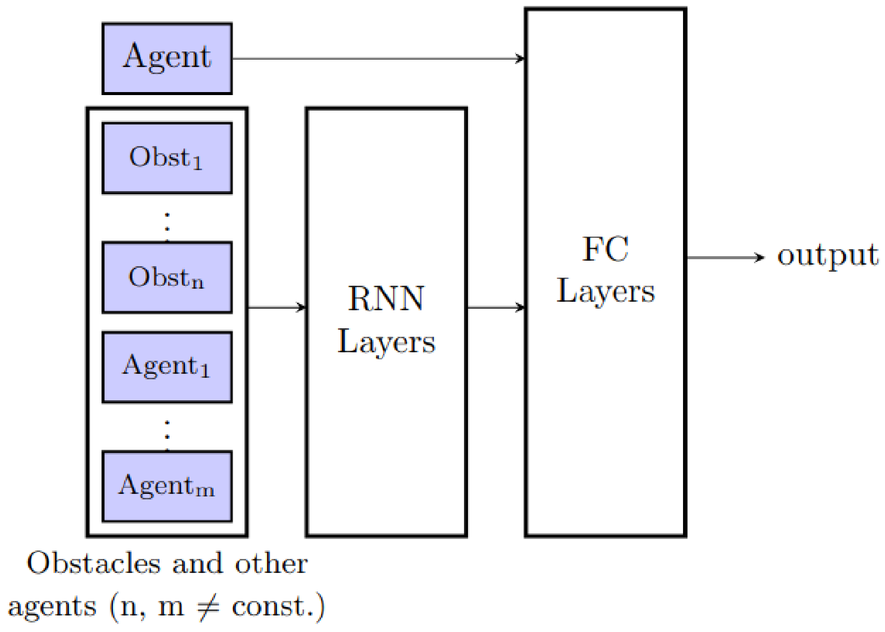
		Data Collection Time [s]
Not Parallelized		320.4
MPI Parallelized	2 processors	181.6
	5 processors	86.8

1.75 times

3.68 times

Planner Framework 1

- Multiagent in Neural Net
  - Issue: Fully-connected (FC) layers have a fixed input size
  - Solution: Use RNN: Long Short-Term Memory (LSTM)
- NN details
  - 4 FC layers with 1024 neurons
  - ReLu
  - Adam optimizer
  - Learning rate decay
  - BC / DAgger



Planner Framework 2

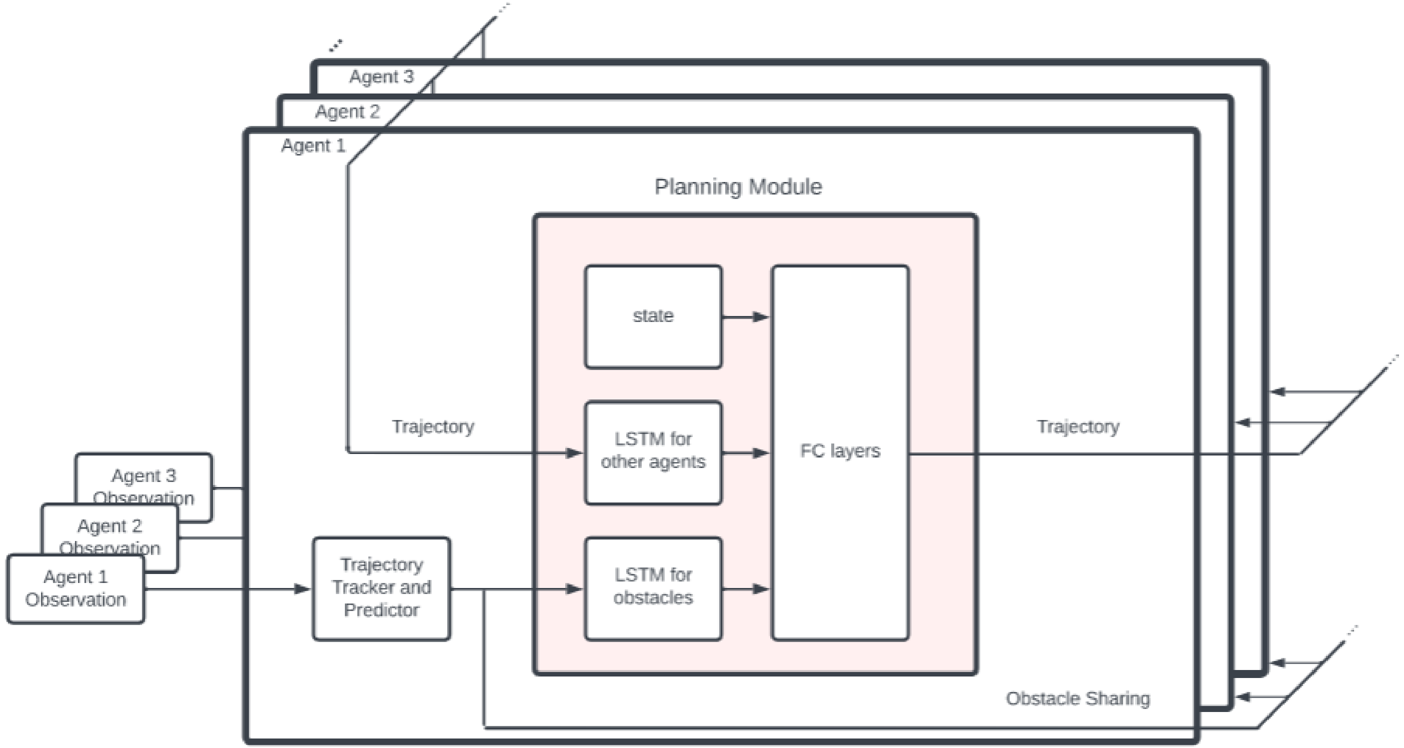


Fig. 3. Student Planning and Sharing Trajectory Architecture

Simulation Results 1: Student Policy Analysis

- BC is not so great to train student -> Data Aggregation (DAgger)
  - Trajectory Cost: FOV + Terminal Goal + Obst. Avoidance + Dyn. Limit. Constr.

Table 5. Expert vs. Student

	Avg. Cost	Computation Time [ms]
Expert	1317.0	**5363.4**
Student (BC)	2055.4	**0.5634**
Student (BC + DAgger)	1550.3	**0.8978**

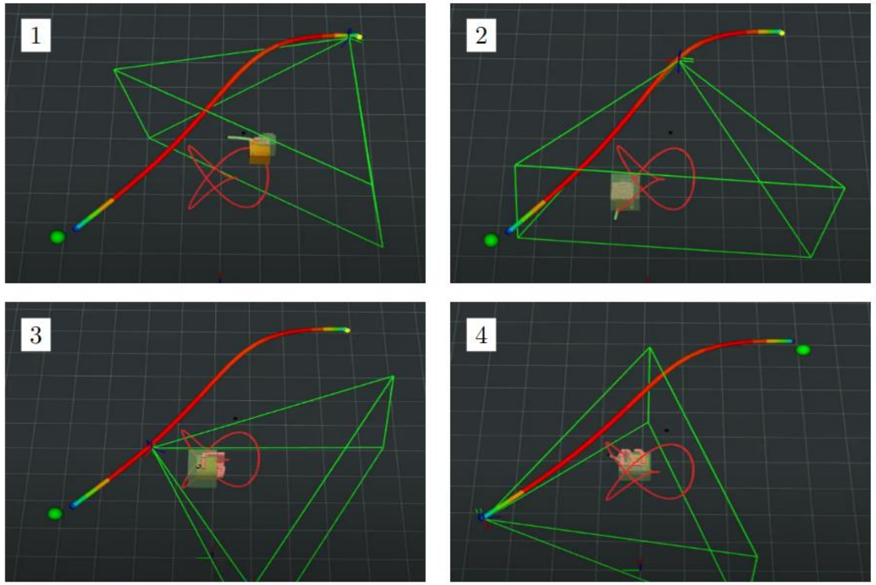


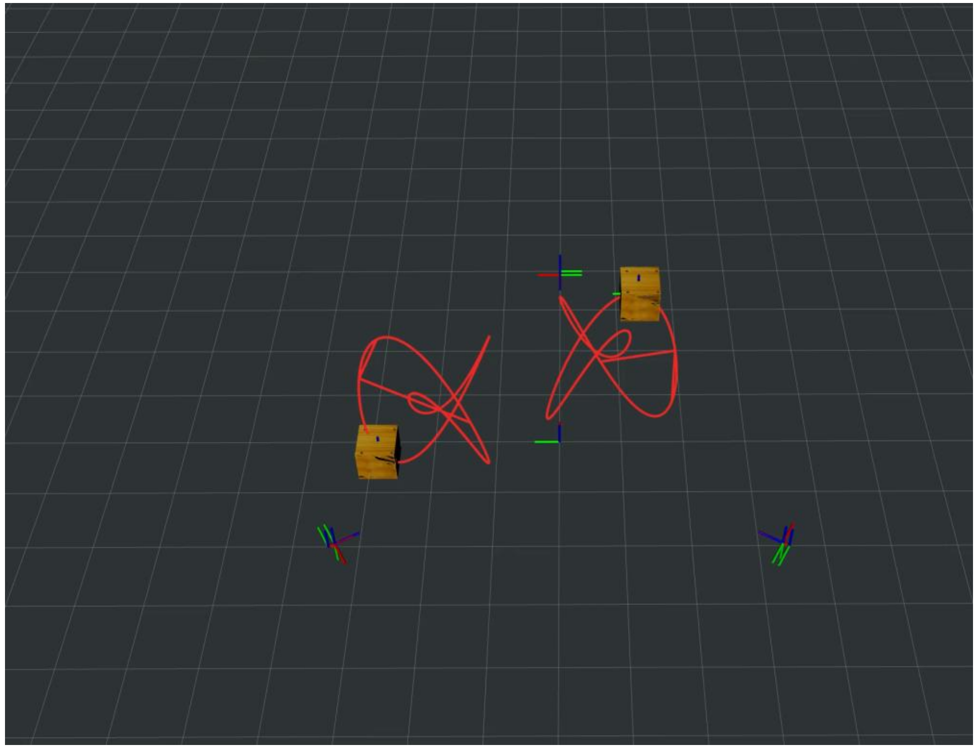
Fig. 5. Student single-agent, single-obstacle, simulation result: We made the Student agent fly around a trefoil-trajectory dynamic obstacle. The agent started at the top-right corner and was commanded to fly to the down-left.

Simulation Results 2: Benchmarking

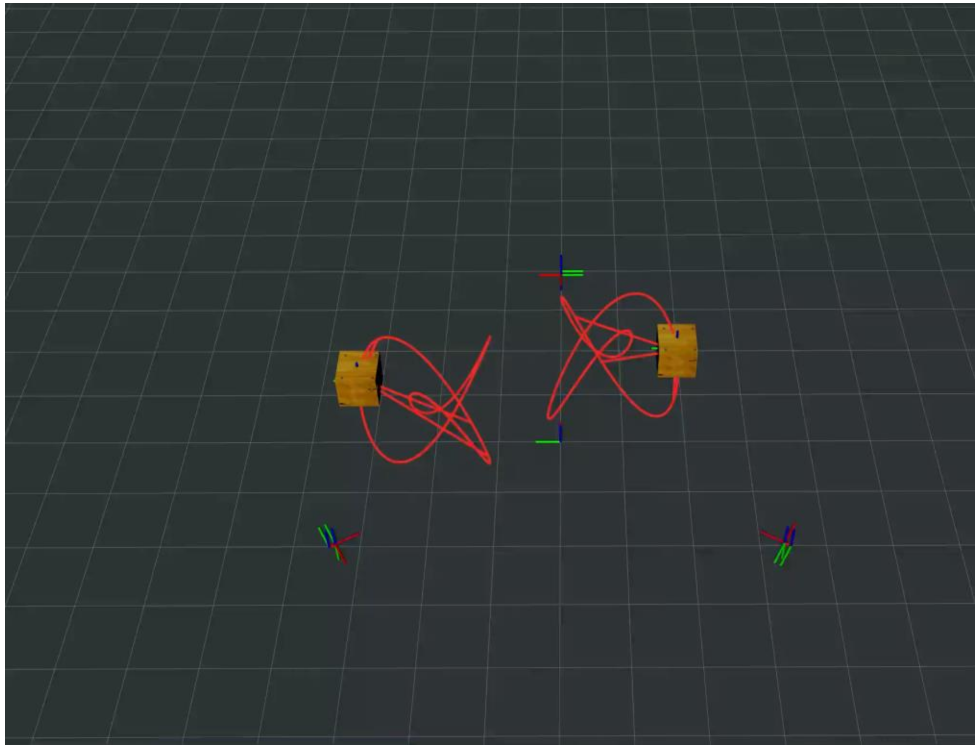
Table 6. Benchmarking

Env.	Method	Compu. Time [ms]	Success Rate [%]	Travel Time [s]	FOV Rate [%]	# Conti. FOV Detection Frames	Dyn. Constr. Violation Rate [%]
1 agent + 2 obst.	Expert	3456.13	100.0	7.87	29.0	19.8	0
	Student	57.11	100.0	4.45	28.0	31.0	10.3
3 agents + 2 obst.	Expert	6212.13	0.0	13.00	19.6	65.7	0.0
	Student	119.82	80.0	5.83	25.0	35.3	5.4

Simulation Results 2: Benchmarking (videos)



Student 3 agents + 2 obsts



Student 3 agents + 2 obsts w/o FOV

Conclusions & Future work

- First Multiagent Perception-aware traj. Planner using IL
  - Decentralized
  - Asynchronous
  - RNN (LSTM) -> multi-obstacles + multiagent
- Fast training done parallelly using Julia MPI
- Benchmarking with multiple obstacles and agents
  - Faster Computation with good performance
- Hardware flight experiments