Early Results with `lf`: Online Multi-Robot Path Planning meets Optimal Trajectory Control

Ajay Shankar[1], **Keisuke Okumura**[1,2], **Amanda Prorok**[1]
[1]University of Cambridge, UK
[2]National Institute of Advanced Industrial Science and Technology (AIST), Japan
{ as3233, ko393, asp45 }@cst.cam.ac.uk

Abstract

We propose a multi-robot control paradigm to efficiently plan collision-free paths and execute them as trajectories in a fully online and optimal fashion. Our method invokes two processes asynchronously at high frequency: a centralized planner that quickly computes simultaneous multi-agent paths for an N-robot team on a discrete space in $\mathbb{R}^3$, and N order-2 optimal trajectory controllers that ensure that all agents independently follow their assigned paths reliably. By fusing a fast replanning with online control, our method, termed `lf`, provides a mechanism to perform multi-robot navigation with arbitrary goal assignments.

Introduction

Efficiently navigating a team of robots to their respective destinations – whilst maintaining responsiveness, and avoiding collisions, deadlocks or livelocks – is a crucial skill for any multi-robot team. However, this skill is non-trivial. Deploying such a team in a practical setting often faces numerous challenges in the complexity of planning and control, assumptions on dynamics and environment, and imperfections in deployment and synchronization [1]. Furthermore, factors such as non-stationary environments and uncertainties from inter-robot or human-robot interactions make a single-shot planning and execution strategy impractical. Ideally, a fast, reactive motion planner that accounts for team-level replanning is tightly coupled with a high bandwidth (and preferably onboard) low-level trajectory controller for precise individual motion control.

Planning and control in the joint-space of the team typically incurs a high computational cost. Decoupled control schemes that rely on local observations and communication have thus received considerably more attention [2, 3, 4, 5, 6]. While their computational efficiency makes them excel at coordinated behaviors in the short-term, they often lack long-term guarantees against non-smooth responses and local deadlocks, especially in dense and constrained environments (e.g., with obstacles).

In parallel, recent advances in search-based multi-agent pathfinding (MAPF) methods have demonstrated remarkable scalability, handling systems with thousands of agents in short planning timeframes [7, 8, 9, 10]. These algorithms typically use abstracted models, such as discrete synchronous actions within grid-world representations. Despite this abstraction, the inherent computational efficiency of such scalable MAPF algorithms opens the door to rapid multi-robot planning in more advanced and complex planning domains, with a slight detriment to scalability.

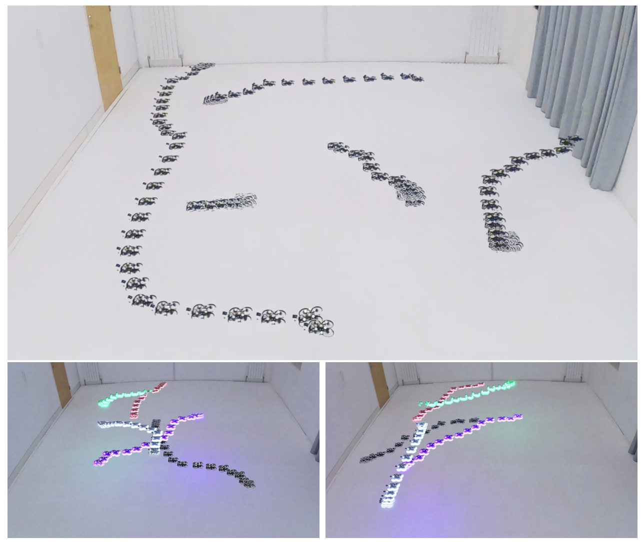


Figure 1: Some timelapse snapshots of five robots executing collision-free trajectories to random goals with `lf`.

Two key developments motivate this work: (1) fast MAPF algorithms that can be applied for online replanning, and, (2) efficient low-level optimal controllers that achieve precise individual trajectory control. We run our MAPF algorithms more frequently (e.g., 5–100 Hz), and use the latest system state to generate updated plans in real time. Concurrently, decoupled and higher-frequency low-level controllers for each robot account for real-world dynamics and track the high-level MAPF plans with high fidelity. This hierarchical scheme addresses the limitations of fully decoupled approaches without sacrificing real-time responsiveness.

This MAPF feedback control differs from conventional MAPF execution studies, which solve MAPF continuously but at low frequency (e.g. every 1–10 s), such as online replanning at each discrete timestep [11, 12], post-processing MAPF plans with liveness guarantees [13, 14, 15, 16], or offline planning robust to uncertainty [17, 18]. Our approach is much simpler; by embedding high-frequency MAPF updates directly into the feedback loop, we aim to achieve seamless and robust multi-robot control.

As a proof of concept, we present the `lf` framework for multi-robot control, which combines: *(i)* a state-of-the-art coupled MAPF algorithm called LaCAM [19], and *(ii)* a decoupled, low-level optimal trajectory control system called Freyja [20]. In the following, we describe brief preliminary results on target robotic systems, architecture, and demonstrations for drone swarm control.

Architecture

Given the current system state and the target positions assigned to each robot, the `lf` framework periodically and asynchronously performs the following two processes: *(i)* a high-level coordination planner that efficiently solves MAPF, and *(ii)* a low-level optimal control mechanism that continuously adjusts the robots' trajectories based on the latest instructions provided by the high-level planner. Typically, the high-level planner operates at a frequency (e.g., 5–100 Hz), enabling it to adapt promptly to dynamic and noisy environments. The low-level controller is executed at further higher frequencies, depending on the requirements of the robotic systems, ensuring precise and smooth control of individual robots. `lf` adapts LaCAM and Freyja as the embodiment of this concept.

Target Systems

Our approach is targeted primarily towards holonomic robot fleets in $\mathbb{R}^3$, operating in constrained indoor settings. Examples include ground-robot platforms such as the Cambridge Robomaster [21], and small aerial platforms such as the Crazyflie [22] and our in-house multirotor platforms. We note that our architecture can easily be extended to platforms with other motion models by replacing only the motion primitives and the dynamics model.

LaCAM

LaCAM is a search-based algorithm designed to efficiently solve large MAPF instances. Building on its recent implementation [19], we have developed a versatile and user-friendly multi-robot control scheme. Unlike traditional MAPF implementations, which are restricted to grid world representations, `lf` uses Octomap [23], a popular 3D environment representation in robotics. The search within LaCAM uses motion primitives inspired by [24] to generate feasible paths. During the search process, collision checking is performed using the Flexible Collision Library (FCL) [25]. To compute heuristics for the search process, `lf` integrates a Probabilistic Roadmap (PRM) [26], a sampling-based motion planning method that approximates the workspace with a discrete graph representation. Each vertex in the PRM is assigned a cost-to-go value calculated using the backward Dijkstra algorithm [27]. The heuristic of a given location is then computed via a gradient derived from the cost-to-go values at the nearest PRM vertices, efficiently found by the k-nearest neighbour search [28].

LaCAM continues to refine solutions as time allows after the initial solution discovery. Unlike LaCAM* [10], an asymptotically optimal variant with search-tree rewiring, `lf` adapts branch-and-bound refinement. It also incorporates advanced techniques from [19], including dynamic solution updates obtained from large neighbourhood search [29, 30]. In addition, path smoothing techniques [31] are introduced which optimise the paths for continuous spaces while maintaining collision-free guarantees. By combining the above techniques, our high-level planner delivers high-quality coordinated paths to the low-level controller quickly.

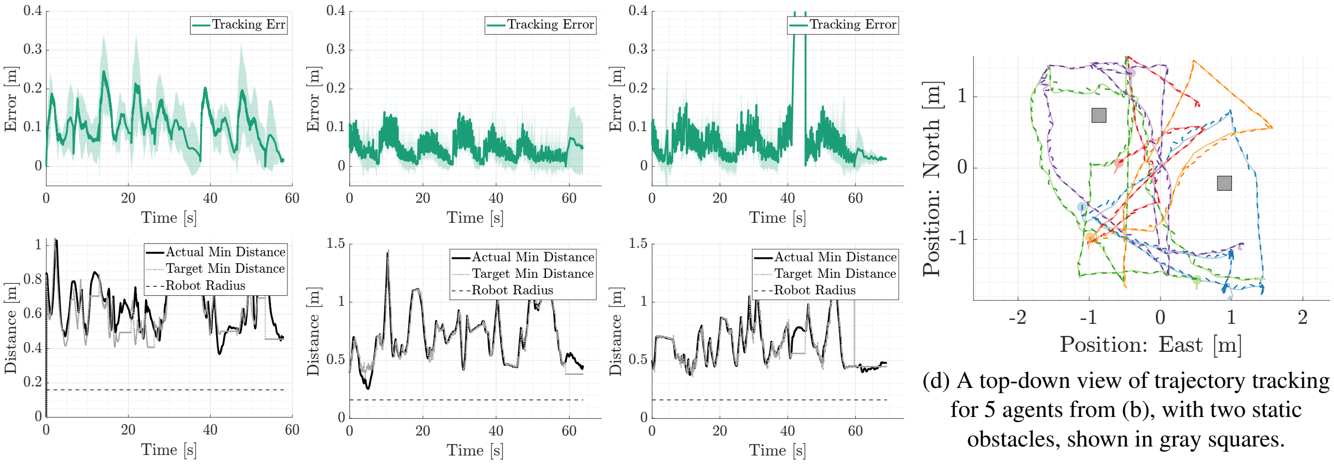
Freyja

Freyja is a model-based optimal non-linear feedback control stack for executing fast and agile robot maneuvers. Our implementation is setup as a collection of three main ROS2 'nodes' that perform state estimation, state regulation over a given trajectory, and vehicle communication handling. Freyja provides several configurable options for each module that are tunable for specific instances and use-cases. In `lf`, we use a Kalman filter to estimate the 6-DoF state (along with its first and second derivatives) using pose measurements from a motion-capture system. A Linear Quadratic Gaussian (LQG) controller is used as a state regulator to generate control actions that drive the robot along a path generated from LaCAM. Due to the differentially flat dynamics of the system, it is possible to perform planning in $\mathbb{R}^3$. Freyja exploits this property to then map the feedforward-linearized control actions [32] into the non-linear action space (target attitude and thrust vectors) of a multirotor. The LQG implementation is extremely robust for a wide range of flight regimes, and offers high computational efficiency (can be run at over 200 Hz on a Raspberry Pi Zero). For larger problems with more constraints, Freyja also supports a versatile model predictive control (MPC) architecture by interfacing with QP-solver frameworks such as OSQP [33].

Since the paths generated from LaCAM are continuous but not necessarily smooth, we implement an additional intermediate step that performs a linear 2nd-order interpolation for paths. This single-step process enables linearization over points on a trajectory (i.e., a point with a velocity vector) as opposed to individual points in space, thus producing smoother motions.

Demonstration

As a proof-of-concept demonstration, we implement the pipeline for a five-robot random path planning scenario shown in Figure 2(d). The robots are small customized multirotor platforms ($0.16\,\mathrm{m}$ diagonal) flying inside a motion-capture arena that measures $(4 \times 6 \times 4)\,\mathrm{m}^3$. The full `lf` framework is implemented in C++ and runs on a consumer laptop, which sends individual control commands to each robot at $100\,\mathrm{Hz}$ over WiFi. The robots were deployed on a lifelong basis, with the team being randomly assigned new targets within the arena after completing the current mission. The demonstrations showcase the robustness of `lf`, even in the presence of dynamic obstacles.



(a) One-shot deployment. (b) Replanning with 5 Hz. (c) With dynamic obstacles.

(d) A top-down view of trajectory tracking for 5 agents from (b), with two static obstacles, shown in gray squares.

Figure 2: Planning and tracking with `lf`: five multirotors in the presence of two pole-shaped obstacles. The figures include *(top)* the tracking error between high-level planner and actual positions, and *(bottom)* the minimum inter-robot distance for a one-minute flight. We tested three scenarios: (a) control scheme *without* online replanning, (b) MAPF replanning at 5 Hz, and, (c), same scheme as (b) but with dynamically moving obstacles (manually relocated at random during the flight). In (d), the trajectories from scenario (b) are visualized, where the dashed lines are from MAPF and the solid lines are the actual positions.

References

[1] Jennifer Gielis, Ajay Shankar, and Amanda Prorok. A critical review of communications in multi-robot systems. *Current robotics reports*, 2022.

[2] Li Wang, Aaron D Ames, and Magnus Egerstedt. Safety barrier certificates for collisions-free multirobot systems. *T-RO*, 2017.

[3] Dingjiang Zhou, Zijian Wang, Saptarshi Bandyopadhyay, and Mac Schwager. Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells. *RA-L*, 2017.

[4] Carlos E Luis, Marijan Vukosavljev, and Angela P Schoellig. Online trajectory generation with distributed model predictive control for multi-robot motion planning. *RA-L*, 2020.

[5] Jesus Tordesillas and Jonathan P How. Mader: Trajectory planner in multiagent and dynamic environments. *T-RO*, 2021.

[6] Xin Zhou, Xiangyong Wen, Zhepei Wang, Yuman Gao, Haojia Li, Qianhao Wang, Tiankai Yang, Haojian Lu, Yanjun Cao, Chao Xu, et al. Swarm of micro flying robots in the wild. *Science Robotics*, 2022.

[7] Jiaoyang Li, Wheeler Ruml, and Sven Koenig. Eecbs: A bounded-suboptimal search for multi-agent path finding. In *AAAI*, 2021.

[8] Keisuke Okumura, Manao Machida, Xavier Défago, and Yasumasa Tamura. Priority inheritance with backtracking for iterative multi-agent path finding. *AIJ*, 2022.

[9] Jiaoyang Li, Zhe Chen, Daniel Harabor, Peter J Stuckey, and Sven Koenig. Mapf-lns2: Fast repairing for multi-agent path finding via large neighborhood search. In *AAAI*, 2022.

[10] Keisuke Okumura. Improving lacam for scalable eventually optimal multi-agent pathfinding. In *IJCAI*, 2023.

[11] Jiří Švancara, Marek Vlk, Roni Stern, Dor Atzmon, and Roman Barták. Online multi-agent pathfinding. In *AAAI*, 2019.

[12] Jiaoyang Li, Andrew Tinka, Scott Kiesel, Joseph W Durham, TK Satish Kumar, and Sven Koenig. Lifelong multi-agent path finding in large-scale warehouses. In *AAAI*, 2021.

[13] Wolfgang Hönig, TK Kumar, Liron Cohen, Hang Ma, Hong Xu, Nora Ayanian, and Sven Koenig. Multi-agent path finding with kinematic constraints. In *ICAPS*, 2016.

[14] Anna Mannucci, Lucia Pallottino, and Federico Pecora. On provably safe and live multirobot coordination with online goal posting. *T-RO*, 2021.

[15] Alexander Berndt, Niels Van Duijkeren, Luigi Palmieri, Alexander Kleiner, and Tamás Keviczky. Receding horizon re-ordering of multi-agent execution schedules. *T-RO*, 2023.

[16] Yifan Su, Rishi Veerapaneni, and Jiaoyang Li. Bidirectional temporal plan graph: Enabling switchable passing orders for more efficient multi-agent path finding plan execution. In *AAAI*, 2024.

[17] Dor Atzmon, Roni Stern, Ariel Felner, Glenn Wagner, Roman Barták, and Neng-Fa Zhou. Robust multi-agent path finding and executing. *JAIR*, 2020.

[18] Keisuke Okumura, François Bonnet, Yasumasa Tamura, and Xavier Défago. Offline time-independent multiagent path planning. *T-RO*, 2023.

[19] Keisuke Okumura. Engineering lacam*: Towards real-time, large-scale, and near-optimal multi-agent pathfinding. In *AAMAS*, 2024.

[20] Ajay Shankar, Sebastian Elbaum, and Carrick Detweiler. Freyja: A full multirotor system for agile & precise outdoor flights. In *ICRA*, 2021.

[21] Jan Blumenkamp, Ajay Shankar, Matteo Bettini, Joshua Bird, and Amanda Prorok. The cambridge robomaster: An agile multi-robot research platform. In *DARS*, 2024.

[22] Crazyflie 2.0 quadrotor as a platform for research and education in robotics and control engineering. In *MMAR*, 2017.

[23] Armin Hornung, Kai M Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *AURO*, 2013.

[24] Liron Cohen, Tansel Uras, TK Kumar, and Sven Koenig. Optimal and bounded-suboptimal multi-agent motion planning. In *SoCS*, 2019.

[25] Jia Pan, Sachin Chitta, and Dinesh Manocha. Fcl: A general purpose library for collision and proximity queries. In *ICRA*, 2012.

[26] Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *ICRA*, 1996.

[27] Edsger W Dijkstra. A note on two problems in connexion with graphs. In *Numerische mathematik*. 1959.

[28] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *CACM*, 1975.

[29] Jiaoyang Li, Zhe Chen, Daniel Harabor, P Stuckey, and Sven Koenig. Anytime multi-agent path finding via large neighborhood search. In *IJCAI*, 2021.

[30] Keisuke Okumura, Yasumasa Tamura, and Xavier Défago. Iterative refinement for real-time multi-robot path planning. In *IROS*, 2021.

[31] Roland Geraerts and Mark H Overmars. Creating high-quality paths for motion planning. *IJRR*, 2007.

[32] Melissa Greeff and Angela P Schoellig. Flatness-based model predictive control for quadrotor trajectory tracking. In *IROS*, 2018.

[33] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 2020.