TU Delft

Delft University of Technology

Document Version
Final published version

Licence
Dutch Copyright Act (Article 25fa)

Citation (APA)
Agarwal, S., Alonso-Mora, J., & Sun, S. (2026). Decentralized Real-Time Planning for Multi-UAV Cooperative Manipulation via Imitation Learning. In *Proceedings of the IEEE International Symposium on Multi-Robot and Multi-Agent Systems, MRS 2025* IEEE. https://doi.org/10.1109/MRS66243.2025.11357262

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
In case the licence states “Dutch Copyright Act (Article 25fa)”, this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

Decentralized Real-Time Planning for Multi-UAV Cooperative Manipulation via Imitation Learning

Shantnav Agarwal[1], Javier Alonso-Mora[1], and Sihao Sun[1]

2025 IEEE International Symposium on Multi-Robot and Multi-Agent Systems (MRS) | 979-8-3315-9359-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/MRS66243.2025.11357262

***Abstract*— Existing approaches for transporting and manipulating cable-suspended loads using multiple UAVs along reference trajectories typically rely on either centralized control architectures or reliable inter-agent communication. In this work, we propose a novel machine learning–based method for decentralized kinodynamic planning that operates effectively under partial observability and without inter-agent communication. Our method leverages imitation learning to train a decentralized student policy for each UAV by imitating a centralized kinodynamic motion planner with access to privileged global observations. The student policy generates smooth trajectories using physics-informed neural networks that respect the derivative relationships in motion. During training, the student policies utilize the full trajectory generated by the teacher policy, leading to improved sample efficiency. Moreover, each student policy can be trained in under two hours on a standard laptop. We validate our method in both simulation and real-world environments to follow an agile reference trajectory, demonstrating performance comparable to that of centralized approaches.**

I. Introduction

Unmanned aerial vehicles (UAVs) have gained significant traction across domains such as surveillance, agriculture, and infrastructure inspection due to their agility and versatility. However, their limited payload capacity restricts their effectiveness in applications involving the transportation of heavy or bulky objects which is common in construction and large-scale logistics. A scalable and cost-effective solution to this limitation is cable-suspended cooperative aerial manipulation [1], where multiple UAVs cooperatively transport and control a cable-suspended payload. This method enables full pose manipulation of objects whose weight may exceed the capacity of a single UAV.

Numerous control strategies have been proposed for cooperative transportation of suspended payloads using UAV teams. These approaches vary in terms of modeling accuracy, scalability, communication requirements, and capability to regulate the full pose of the payload. Given the focus of this work on decentralized cooperative aerial manipulation, prior methods are categorized into three primary frameworks: centralized control, decentralized control with communication, and decentralized control without communication.

1) Centralized Approaches: A formation-based geometric control approach was introduced in [2], modeling a point-mass suspended via massless rigid links. This method centrally solves for force equilibrium using full system state information, allowing formation maintenance but neglects payload attitude. Moreover, the approach has only been validated in simulation.

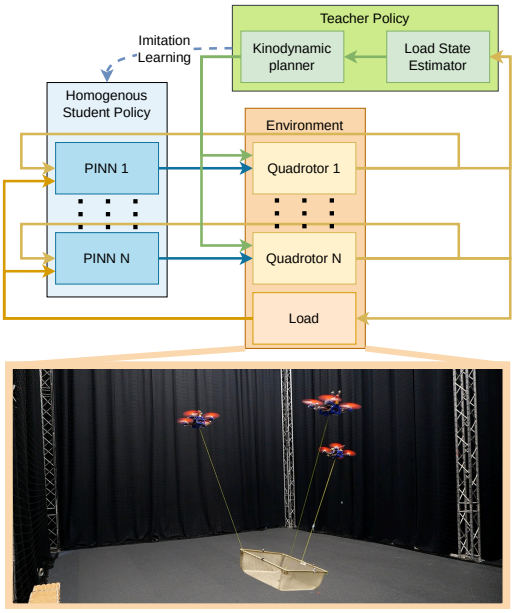


Fig. 1: We enable decentralized cooperative aerial manipulation through student policies that operate independently using only the ego UAV's state and the pose of the load. These student policies are trained via imitation learning from a centralized teacher policy with privileged observations, including the full state of the other UAVs and the load. The policy has been tested in real-world environments, where three UAVs cooperatively manipulate a cable-suspended load.

Model Predictive Control (MPC) is commonly used in centralized frameworks due to its ability to handle constraints and optimize control over a receding time horizon. Linear MPC approaches typically model the payload as a point mass and use linearized system dynamics [3], [4]. While computationally efficient, these methods cannot control the payload's orientation and often under perform on dynamic trajectories [5].

Nonlinear MPC (NMPC) techniques have been introduced that allow manipulating the full-pose of the load along agile trajectories. For example, a cascaded NMPC controller has been developed that enables full pose manipulation of the load[6] . However, this controller assumes load dynamics are a magnitude slower than quadrotor dynamics, which reduces control performance on agile trajectories. Recently, in [7], an online kinodynamic motion planner utilizing a load-cable-UAVs model without assumption of time-scale separations

[1] Authors are with the Department for Cognitive Robotics, ME, Delft University of Technology, Delft, Netherlands

Authorized licensed use limited to: TU Delft Library. Downloaded on May 04,2026 at 08:01:50 UTC from IEEE Xplore. Restrictions apply.

has been proposed, enabling agile full-pose control.

While centralized controllers have achieved agile and accurate load manipulation, they rely on high-bandwidth, low-latency communication networks with a central coordinator. This assumption is often impractical in real-world scenarios where communication may be unreliable or unavailable. Further, the computational cost of aforementioned approaches grows exponentially with the number of agents thereby limiting scalability. Hence, the community has also explored decentralized methods for cooperative aerial manipulation.

2) Decentralized Approaches with Communication: A distributed model predictive control framework has been developed which demonstrates performance comparable to centralized MPC while distributing the computational burden across agents[8]. The approach assumes a fully-connected communication graph, wherein each UAV shares its previous control inputs with neighboring agents and independently optimizes its own input sequence at each iteration. Though validated only in simulation, the framework exhibits scalability and coordination capabilities. While the method is scalable, it still relies on continuous inter-agent communication, motivating the need for decentralized control strategies that operate without explicit communication.

3) Decentralized Approaches without Communication: A decentralized control scheme combining distance-based formation control with incremental nonlinear dynamic inversion (INDI) is proposed in [9]. UAVs compute local acceleration commands based solely on relative position measurements, eliminating the need for global positioning or inter-agent communication. This method, however, does not regulate the payload's orientation. Master-slave architectures have also been employed, where a designated master UAV generates a reference trajectory, and slave UAVs follow using force-based admittance control [10], [11]. These approaches are limited in their ability to manipulate the payload's full pose and have not been extended beyond two-UAV systems.

While decentralized approaches without inter-agent communication offer benefits like scalability and robustness, they struggle to achieve agile and precise full-pose manipulation of a payload. To address this challenge, we present a decentralized, communication-free online kinodynamic motion planner for cooperative aerial manipulation. Our method enables multiple autonomous UAVs to perform agile and accurate full-pose control of a rigid object. Each UAV operates independently, relying solely on local observations and the object's reference trajectory, to generate it's own receding-horizon trajectories. This trajectory is then followed by low-level trajectory tracking controllers deployed onboard the UAV. To achieve this, each UAV is equipped with a neural network policy trained through imitation learning from a privileged expert [12]. A centralized receding-horizon kinodynamic motion planner serves as the teacher policy [7], guiding the training of the strongly homogeneous [13] decentralized student policies. We assume that each UAV can accurately estimate both its own state and the state of the payload. To the best of our knowledge, this is the first work to develop a decentralized, communication-free planner for cooperative cable-suspended aerial manipulation. We have validated our approach in both simulation and real-world experiments.

II. Problem Statement

A. Notations

We consider a system composed of n unmanned aerial vehicles (UAVs) and a single rigid load body. The pose and twist of the i^{th} UAV are denoted by $[\boldsymbol{p}_i \quad \boldsymbol{q}_i]$ and $[\boldsymbol{v}_i \quad \boldsymbol{\omega}_i]$, respectively, where $\boldsymbol{p}_i \in \mathbb{R}^3$ and $\boldsymbol{q}_i \in \mathbb{S}^3$ represent the position and orientation (quaternion), and $\boldsymbol{v}_i, \boldsymbol{\omega}_i \in \mathbb{R}^3$ denote the linear and angular velocities. We use bold lowercase letters to denote vectors, and bold capitalized letters for matrices; otherwise, variables are scalars.

Similarly, the pose and twist of the load are represented as $[\boldsymbol{p}_L \quad \boldsymbol{q}_L]$ and $[\boldsymbol{v}_L \quad \boldsymbol{\omega}_L]$, with analogous definitions.

A load-attached reference frame, denoted $\mathcal{L}$, is defined such that its origin coincides with $\boldsymbol{p}_L$, and its orientation is aligned with the inertial frame $\mathcal{I}$.

Each UAV has an associated body-fixed frame $\mathcal{B}_i = [\boldsymbol{O}_i \quad \boldsymbol{x}_i \quad \boldsymbol{y}_i \quad \boldsymbol{z}_i]$, where the origin $\boldsymbol{O}_i$ is located at $\boldsymbol{p}_i$, and the z_i-axis points vertically upward, coinciding with the thrust direction.

Unless otherwise stated, all vectors are expressed in the inertial frame $\mathcal{I}$. Frame-specific quantities are indicated using superscripts when necessary.

B. Cooperative Aerial Manipulation

cooperative aerial manipulation refers to the coordinated use of multiple aerial robots to jointly transport and manipulate a common object. A representative depiction of this setup is provided in Figure 1. The system consists of multiple UAVs, each connected to a rigid load via cables. The cables are attached below the center of mass of each UAV, and at arbitrary points on the load, using universal joints that allow the transmission of forces but not moments.

Each UAV is modeled as a unidirectional thrust platform, capable of generating thrust solely along its body-fixed positive z_i-axis. The primary objective is for the UAVs to collaboratively control the full pose of the load—both its position and orientation—so that it tracks a desired trajectory over time.

C. Fully Decentralized Cooperative Aerial Manipulation

Fully decentralized cooperative aerial manipulation aims to achieve full-pose control of a load using multiple UAVs, where each UAV operates independently based on local observations and without direct communication with other UAVs.

This decentralized setting introduces several key constraints and challenges - *Distributed Control:* Each UAV is governed by an onboard control policy and functions autonomously, without access to the control inputs or internal states of other UAVs. *Local Observations:* Each UAV's planning and control policy relies solely on locally available sensor data, including its own pose and twist, as well as the pose of the load. No information about the states or

Authorized licensed use limited to: TU Delft Library. Downloaded on May 04,2026 at 08:01:50 UTC from IEEE Xplore. Restrictions apply.

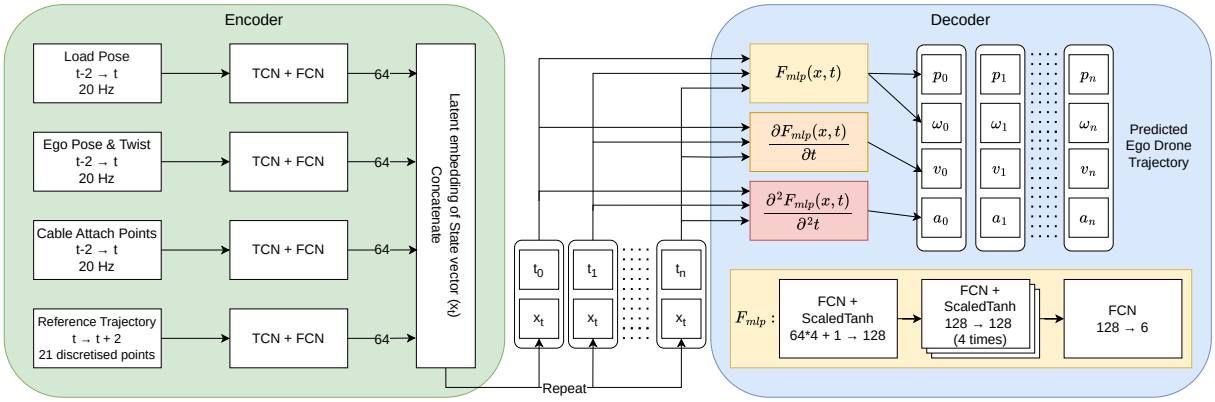


Fig. 2: The architecture of the student policy is made up of two parts - Encoder (Sec. III-B.1) & Decoder (Sec. III-B.2). The Encoder network maps the observation histories to a latent vector x_t. This latent vector is then copied for all nodes in the horizon and passed to the Decoder network.

actions of other UAVs is accessible. *Implicit Coordination:* UAVs must learn to collaborate effectively during the training phase, such that during deployment, coordinated behavior emerges through physical interactions alone and without any explicit communication channels. This framework reflects practical limitations in real-world multi-robot systems, where communication constraints, scalability, and robustness to failure are critical considerations.

III. Method

This section introduces the proposed control policy, including details on the training procedure, observation & action space as well as the architecture of the learned policy.

A. Teacher Policy

We use the planning algorithm introduced in [7] as the teacher policy, which is the state-of-the-art centralized method for the multi-lifting system for full-pose control of a cable-suspended load, with high agility and robustness. Given the reference trajectory of the load, the teacher policy solves an optimization-based kinodynamic motion planning problem and generates receding-horizon reference trajectories for each UAV at 10 Hz. The reference trajectories are then followed by low-level controllers deployed on each UAV, using a flatness-based controller and INDI [14] to compensate for external forces and torques from the cable.

B. Student Policy

While the centralized policy demonstrates high agility and robustness against dynamic model uncertainties, it relies on access to the full state information of all UAVs and the load. In contrast, the student policy learns to generate trajectories for the ego UAV in a decentralized manner, using only the ego UAV's state and the load's pose. Notably, the student policy shares the same low-level controllers as the teacher policy. It is implemented as a strongly homogeneous machine learning policy—meaning all UAVs run identical copies of the policy, with the same architecture and weights [13]. The architecture of the student policy is depicted in Figure 2, and an overview of the overall control framework is illustrated in Figure 1. The policy operates at a control frequency of 10 Hz, consistent with the teacher policy.

1) Observation Space: The student policy has access to the following information:

$$\boldsymbol{o}_i = \begin{bmatrix} \boldsymbol{p}_L^{\mathcal{L}} & \boldsymbol{q}_L^{\mathcal{L}} & \boldsymbol{p}_i^{\mathcal{L}} & \boldsymbol{v}_i^{\mathcal{L}} & \boldsymbol{q}_i^{\mathcal{L}} & \boldsymbol{\omega}_i^{\mathcal{L}} & \boldsymbol{p}_{C1,i}^{\mathcal{L}} & \boldsymbol{p}_{C2,i}^{\mathcal{L}} \end{bmatrix} \quad (1)$$

The decentralized policy receives the *Pose* of the load as well as the *Pose* and *Twist* of the ego UAV. Notably, the policy does not receive any information about the other UAVs in the team.

To enable generalization across environments, all observations are expressed in the load frame $\mathcal{L}$. This makes the policy invariant to absolute world position allowing it to operate in spaces of arbitrary size.

It is demonstrated in [15] that leveraging load dynamics as implicit communication effectively facilitates cooperative load transportation. Their method encodes the load's state via the positions and velocities of eight bounding-box vertices, where agents' actions alter these dynamics and influence others' strategies, enabling scalable multi-agent collaboration.

Similarly, we provide the student policy with the position of the ends of the cable on the load ($\boldsymbol{p}_{C1,i}^{\mathcal{L}}$) and on the UAV ($\boldsymbol{p}_{C2,i}^{\mathcal{L}}$) The cable attach points, represented in the load-fixed frame $\mathcal{L}$, convey both the spatial position where each UAV applies force and the direction of that force. The exact location of these points on the load defines the moment arm, which determines the torque generated around the load's center of mass. This information is crucial for accurately controlling the load's pose during cooperative manipulation. The positions of these points are computed from known attachment positions in the UAV and load frames:

$$\boldsymbol{p}_{C1,i} = \boldsymbol{p}_L + \boldsymbol{R}_L \boldsymbol{\rho}_i^{\mathcal{L}}, \quad \boldsymbol{p}_{C2,i} = \boldsymbol{p}_i + \boldsymbol{R}_i \boldsymbol{r}_i^{\mathcal{B}_i}, \quad (2)$$

where $\boldsymbol{R}_L$, $\boldsymbol{R}_i$ is the rotation matrix of the load and the UAV, respectively. $\boldsymbol{\rho}_i^{\mathcal{L}}$ represents the vector from load origin to the i^{th} UAV's cable attach point on the load. Similarly, $\boldsymbol{r}_i^{\mathcal{B}_i}$ represents the vector from the the ego UAV's origin to the cable attach point.

Instead of using quaternions, we represent rotations using the 6D continuous formulation proposed by Zhou et al. [16].

Authorized licensed use limited to: TU Delft Library. Downloaded on May 04,2026 at 08:01:50 UTC from IEEE Xplore. Restrictions apply.

This representation avoids gimbal lock and discontinuities inherent in Euler angles or quaternions and therefore improves differentiability and learning stability.

To reason under partial observability and infer the intentions from other agents, the policy uses a history of past observations. Specifically the observation matrix at time t is defined as

$$\boldsymbol{O}_i = [\boldsymbol{o}_{i,t}, \boldsymbol{o}_{i,t-dt}, \ldots, \boldsymbol{o}_{i,t-Ndt}] \tag{3}$$

where $\boldsymbol{o}_{i,t}$ is the observation vector of the i-th UAV at time t.

Typically, time parameterized representations of future reference states help the agent develop a deeper understanding of the scenario, leading to improved performance [17]. Therefore, we provide the policy with the reference trajectory for the next 2 seconds at $N = 21$ nodes, which are the same references for the teacher policy, thereby ensuring that both the teacher and student get the same reference data as input during training.

$$\boldsymbol{Y} = [\boldsymbol{y}_0 \quad \boldsymbol{y}_1 \ldots \boldsymbol{y}_N] \tag{4}$$

where

$$\boldsymbol{y}_j = \begin{bmatrix} \boldsymbol{p}_L^{\mathcal{L}} & \boldsymbol{v}_L^{\mathcal{L}} & \boldsymbol{a}_L^{\mathcal{L}} & \boldsymbol{q}_L^{\mathcal{L}} & \boldsymbol{\omega}_L^{\mathcal{L}} \end{bmatrix}_j, \; j \in \{0, \ldots, N\}. \tag{5}$$

Here $\boldsymbol{a}_L$ represents the linear acceleration of the load.

To process sequential data of observation histories and reference trajectories, we use Temporal Convolutional Networks (TCN), which offer longer memory, stable gradients, and fast, parallel inference. Then the outputs of the TCN network are passed through a dense layer to create a vector of latent embeddings. The four streams of information as shown in Figure 2 are recorded at different time steps and therefore processed using separate TCN networks. The outputs of all TCNs are concatenated into a latent vector $\boldsymbol{x}_i$ that encodes the current state of the system as well as the desired future reference states:

$$\boldsymbol{x}_i = \pi_{\text{encoder}}(\boldsymbol{O}_i, \quad \boldsymbol{Y}) \tag{6}$$

2) Action Space: The student policy, as an online planner, directly generates reference trajectories for the ego UAV. In the training time, we use the output of the centralized teacher policy's online planner directly as the target for the student policy. Therefore, the action space for the i-th UAV can be represented by the following equations. Note that we omit the subscript i for readability.

$$\boldsymbol{U} = [\boldsymbol{u}_0 \quad \boldsymbol{u}_1 \ldots \boldsymbol{u}_N] \tag{7}$$

with

$$\boldsymbol{u}_j = [\boldsymbol{p} \quad \boldsymbol{\omega} \quad \boldsymbol{v} \quad \boldsymbol{a}]_j, \quad j \in \{0, \ldots, N\} \tag{8}$$

where $\boldsymbol{p}$, $\boldsymbol{v}$, $\boldsymbol{a}$ represent desired position, velocity and acceleration; $\boldsymbol{\omega}$ represents the desired body rates of the ego UAV. The onboard low-level controller typically employs a differential-flatness-based method to follow the reference trajectory generated by the planner for the UAV. Incorporating higher-order derivatives of the reference trajectory, such as velocity and acceleration, can significantly improve tracking performance [18]. A key challenge in predicting higher-order terms with a learning-based policy is ensuring kinematic consistency. The network may generate trajectories wherein the time derivative of position deviates from the predicted velocity, and likewise, the derivative of velocity from the predicted acceleration. To address this problem, we propose the use of Physics Informed Neural Networks (PINNs) with architectural constraints applied as described below:

$$[\boldsymbol{p}_j, \quad \boldsymbol{\omega}_j] = F_{mlp}(\boldsymbol{x}_i, t_j) \tag{9}$$

$$\boldsymbol{v}_j = \frac{\partial F_{mlp}(\boldsymbol{x}_i, t_j)}{\partial t} \tag{10}$$

$$\boldsymbol{a}_j = \frac{\partial^2 F_{mlp}(\boldsymbol{x}_i, t_j)}{\partial t^2} \tag{11}$$

where t is the instantaneous time and t_j is the timestamp of the j-th node in the UAV reference trajectory calculated using the equation:

$$t_j = t_{j-1} + 0.01 + 0.009 \cdot (j-1), \quad t_0 = t \tag{12}$$

We therefore employ PINNs as the decoder (Fig. 2) to generate trajectories from latent vectors that are inherently feasible, as they explicitly enforce kinematic constraints, i.e., that velocity is the derivative of position and acceleration is the derivative of velocity.

C. Training environment

The student policy is exclusively trained within the Gazebo (RotorS) [19] simulation environment, and the resulting policy is subsequently deployed in the real world without any modification. The training process employs privileged learning [12] in conjunction with the Dataset Aggregation (DAgger) algorithm [20]. DAgger is an iterative imitation learning algorithm wherein each iteration consists of two primary phases: data collection and policy training. During data collection, either the teacher or the student policy is used to interact with the environment to collect demonstrations. Afterwards the policy is trained to predict the teacher's action given the corresponding observations as input. In the initial iteration, only the teacher policy is used to determine actions. In subsequent iterations, the probability of using the student policy to control the system is gradually increased. This approach enables the generation of a more diverse dataset that encompasses a wider range of system states, thereby enhancing the robustness of the learned student policy [20].

To further improve the quality and diversity of the training dataset, we introduce randomized external disturbances during the data collection phase. Specifically, at the start of an episode, a random force-torque vector is selected to define the disturbance direction. During each time step of the episode, a force with random magnitude is applied along this predefined direction, effectively simulating a consistent yet stochastic perturbation. This method promotes the exploration of a broader range of system dynamics, which contributes to the robustness and generalization capability of the resulting policy.

Authorized licensed use limited to: TU Delft Library. Downloaded on May 04,2026 at 08:01:50 UTC from IEEE Xplore. Restrictions apply.

IV. Results

A. Environment for practical experiments

The proposed algorithm is validated through real-world experiments involving cooperative manipulation of a 1.4 kg rigid-body payload using three quadrotors, each with a mass of 0.6 kg. The payload is suspended by three cables, each 1 m in length, connected to distinct attachment points on the payload to facilitate full pose control. The opposing ends of the cables are affixed 0.03 m below the center of gravity (CoG) of each quadrotor. The aerial platforms are customized from the Agilicious open-source hardware framework [21], with onboard computation for low-level controllers handled by Raspberry Pi 5 mini PCs.

The student policy for each UAV is deployed as an independent ROS node on the aforementioned laptop. The predicted trajectories are transmitted to the Agilicious flight controller onboard each UAV via a ROS topic over Wi-Fi at 10 Hz. We use a motion capture system for high-precision state estimation of both the UAVs and the suspended payload.

B. Pose Trajectory Tracking

The student policy was trained on a figure-eight trajectory with dimensions 2.2 m $\times$ 2 m over 20 DAgger rounds, each consisting of a single episode. After each round, the probability of selecting actions from the student policy was linearly increased from 0 to 1 by round 20. During each round, the policy was trained on the aggregated dataset for 16 epochs. To enhance robustness against real-world perturbations, consistent yet stochastic disturbances, as detailed in Section III-C, were applied to the load during the final 5 episodes of training. The entire training process, including data collection and policy updates, was executed on the same laptop used for deployment and completed within 80 minutes. The trained policy was then evaluated on the identical figure-eight trajectory in the real world via zero-shot sim-to-real transfer. We perform two flights for both student and teacher policies on the same hardware and report the average of the metrics in Table I. We also show the trajectory tracking performance of both in Fig 3. In real-world experiments, the student policy, executed in a fully decentralized manner using only ego-UAV observations, achieves a comparable orientation RMSE to the teacher policy but exhibits a higher position RMSE.

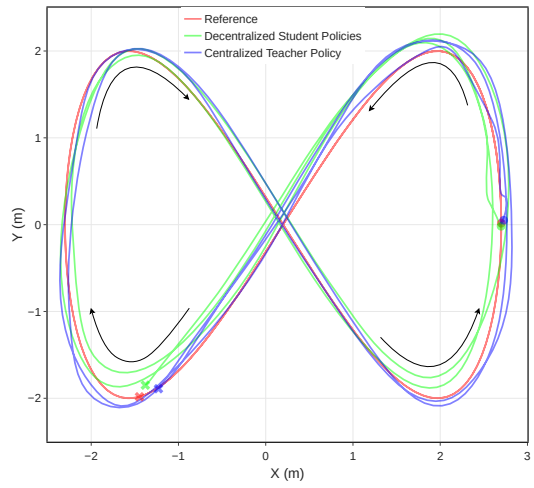


Fig. 3: Top view of tracking performance. Circle, cross indicate start, end of trajectory. Two laps of Eight are performed.

Next, we train the student policy in a similar manner, but using two distinct trajectories. Training alternated between these two trajectories over 20 DAgger rounds, with perturbations applied during the final 4 rounds. The resulting policy was then evaluated in simulation on multiple unseen trajectories. The results are summarized in Table II.

Metric	Teacher Policy	Student Policies
RMSE Posn. (m)	0.171	0.377
RMSE Orient. (°)	10.447	9.335

TABLE I: Comparison of metrics from real-world flights following the figure-eight trajectory shown in Fig 3. The values are the means of the two flights.

Ref. Size		Train	Student Policies		Teacher Policy	
X(m)	Y(m)	Dataset	Posn. m	Orient. °	Posn. m	Orient. °
2.7	2.7	Yes	0.166	6.768	0.089	4.111
2.7	2.3	Yes	0.155	5.684	0.091	4.324
2.0	2.0	No	0.377	10.877	0.073	3.397
3.0	3.0	No	0.175	7.531	0.105	5.371
3.5	3.5	No	0.707	12.013	0.192	6.912
2.5	2.5	No	0.205	6.903	0.090	4.196

TABLE II: Comparison of the student and teacher policy performance in simulation for multiple trajectories. The table reports the RMSE for position and orientation. Student policies were trained using the first two trajectories. X and Y denote the amplitudes of the sine and cosine functions used to generate the trajectories.

C. Advantage of PINN over MLP as Decoder

A benefit of using PINNs as decoder is that the trajectories are guaranteed to be consistent and smooth. Consistency comes from the fact that in all target trajectories in training data start at ego UAV's pose and therefore the policies learn this behavior as well. Smoothness are guaranteed by the partial derivative constraints on velocity and acceleration. We compare the PINN architecture with a baseline policy where the *Decoder* network employs a single Multi-layer Perceptron (MLP) to generate all four trajectory components. The trajectories produced by this model are visibly jagged (shown in the supplementary video), lacking the smoothness enforced by the PINN's constraints. Moreover, they exhibit inconsistencies, where the numerical derivative of predicted position does not match the predicted velocity as highlighted in Table III

Method	X (m/s)	Y (m/s)	Z (m/s)	Overall (m/s)
PINN	0.010	0.030	0.004	0.015
MLP	0.080	0.116	0.048	0.081
Teacher	0.012	0.033	0.003	0.016

TABLE III: The Mean Absolute Error (MAE) between the numerical derivative of predicted position and the predicted velocity was evaluated for each method. For PINN and the teacher planner, the errors primarily arise from numerical differentiation. In contrast, the MLP exhibits higher errors due to both numerical differentiation and additional inaccuracies stemming from infeasible position and velocity predictions.

Authorized licensed use limited to: TU Delft Library. Downloaded on May 04,2026 at 08:01:50 UTC from IEEE Xplore. Restrictions apply.

D. Generalizability to Unseen Trajectories

We train the student policy to execute different trajectories—namely, Eight, Circle, and Square each with varying sizes. These set of trajectories contain a very diverse set of curves with a large range of radii of curvature which needs to be followed by the load. Further, we train the student policy to follow two different orientation modes: 1) Constant orientation where the orientation of the load is constant; and 2) Zero Side-slip orientation where the orientation of the load changes according to the velocity vector.

The trained student policy is then evaluated on the Zandvoort F1 Track located in The Netherlands. This trajectory differs substantially from those encountered during training, exhibiting greater variability in curvature radii, path length, and overall duration. The quantitative results are summarized in Table IV. Additional details on training trajectories and results are provided in supplementary material available here.

Metric	Zandvoort (Constant)	Zandvoort (ZSS)
RMSE Posn. (m)	0.092	0.378
RMSE Orient. (°)	3.157	93.575

TABLE IV: Comparison of metrics from simulated flights around the Zandvoort F1 Track. The Student Policy (Both) was deployed for this experiment without any fine-tuning.

E. Computational Time

Real-time inference is possible on the Raspberry Pi 5 CPU and takes 97 milliseconds, as the planner generates trajectories at 10 Hz. However, during experiments, we realized that running both the Agilicious flight control stack and PINN inference on the Raspberry Pi leads to a degradation in the inner control loop update rate. Therefore, we eventually deployed PINNs on a laptop and sent them to each UAV through Wi-Fi. The average inference time of the student policy is 27 milliseconds, on a NVIDIA RTX 3060 Laptop GPU. In comparison, the teacher policy requires an average computation time of 15.3 milliseconds (for 3 UAVs) running on an Intel Core i7-13700H CPU according to [7], and its computational complexity increases exponentially with the number of UAVs. In contrast, the student policy, due to its decentralized and partially observable architecture, exhibits the potential for scalability. By enabling each UAV to independently predict its own trajectory, the framework could support horizontal scaling, making it feasible to accommodate a large number of UAVs without any increase in computation time.

V. Discussions

The proposed student policy demonstrates the capability to control the full pose of a load along agile reference trajectories without relying on networked communication. Trained entirely in simulation, it achieves zero-shot sim-to-real transfer thanks to the onboard-deployed trajectory tracking controller. Despite being a decentralized machine learning policy, it retains several key benefits of the teacher policy: *Sample Efficiency*, as the entire prediction horizon is leveraged during training; *System Dynamics*, since the learned trajectories are physically consistent by design due to architectural constraints; *Receding Horizon Planning*, which can increase the situational awareness of human operators and aid the development of safety modules; and *Lower Frequency*, as the trajectory-based structure requires execution at lower update rates, allowing for larger models and reduced computational load.

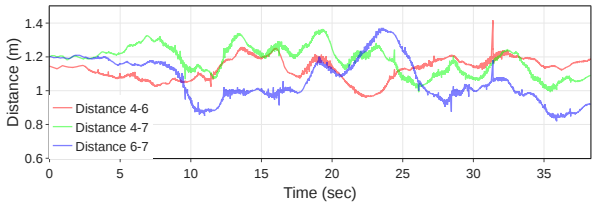


Fig. 4: Despite lacking awareness of each other's positions, the quadrotors successfully maintain safe inter-agent distances

One notable limitation of the student policies is the absence of hard guarantees for inter-agent collision avoidance. Unlike the teacher policy, which explicitly enforces minimum separation constraints, the decentralized policy employed in our method lacks direct awareness of other UAVs' positions. Consequently, the risk of inter-agent collisions is inherently higher for the decentralized student policy. While the student policy learns to approximate safe behavior, primarily by imitating the teacher and placing UAVs in safe configurations relative to the current load pose and reference trajectory, it does not explicitly account for neighboring agents. In real-world experiments, occasional breaches of the 1 m minimum separation threshold were observed; however, the UAVs typically recovered and reestablished safe distances, as illustrated by the trajectory data in Fig. 4. Enhancing the system with onboard sensing for inter-UAV perception and obstacle detection represents a promising avenue for future research, potentially enabling safer decentralized coordination in aerial manipulation scenarios.

Simulation results on Zandvoort F1 Track indicate that the student policies are able generalize to very different trajectories when the orientation of the load is kept constant. However, the pose-tracking performance deteriorates for unseen trajectories with zero side-slip orientation. In these cases, the student policy struggles to compute optimal trajectories, likely because the required motion patterns deviate substantially from those encountered during training. A more comprehensive analysis of the method's ability to handle a wider range of trajectory types remains an open research direction for future work.

VI. Conclusion

In this work, we have designed and evaluated a fully decentralized real-time planning method for cooperative aerial manipulation of a cable-suspended load using multiple UAVs, without inter-UAV communications. Our method leverages imitation learning to train a student policy by imitating a centralized teacher policy with privileged information of the environment. We have experimentally validated the proposed planning method, showing its effectiveness in real-world flights.

Authorized licensed use limited to: TU Delft Library. Downloaded on May 04,2026 at 08:01:50 UTC from IEEE Xplore. Restrictions apply.

References

[1] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, present, and future of aerial robotic manipulators," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 626–645, 2022. DOI: 10.1109/TRO.2021.3084395

[2] T. Lee, K. Sreenath, and V. Kumar, "Geometric control of cooperating multiple quadrotor uavs with a suspended payload," in *52nd IEEE Conference on Decision and Control*, 2013, pp. 5510–5515. DOI: 10.1109/CDC.2013.6760757

[3] G. Tartaglione, E. D'Amato, M. Ariola, P. S. Rossi, and T. A. Johansen, "Model predictive control for a multi-body slung-load system," *Robotics and Autonomous Systems*, vol. 92, pp. 1–11, 2017, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2017.02.007 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889016305516

[4] I. Moreno Caireta, "Planning and control of a multiple-quadcopter system cooperatively carrying a slung payload in dynamical environments. a centralized model predictive control solution," Ph.D. dissertation, UPC, Facultat d'Informàtica de Barcelona, 2018. [Online]. Available: http://hdl.handle.net/2117/121147

[5] M. Kamel, M. Burri, and R. Siegwart, "Linear vs nonlinear mpc for trajectory tracking applied to rotary wing micro aerial vehicles," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3463–3469, 2017.

[6] G. Li and G. Loianno, "Nonlinear model predictive control for cooperative transportation and manipulation of cable suspended payloads with multiple quadrotors," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Oct. 2023, 5034–5041. DOI: 10.1109/iros55552.2023.10341785 [Online]. Available: http://dx.doi.org/10.1109/IROS55552.2023.10341785

[7] S. Sun, X. Wang, D. Sanalitro, A. Franchi, M. Tognon, and J. Alonso-Mora, *Agile and cooperative aerial manipulation of a cable-suspended load*, 2025. arXiv: 2501.18802 [cs.RO]. [Online]. Available: https://arxiv.org/abs/2501.18802

[8] J. Wehbeh, S. Rahman, and I. Sharf, "Distributed model predictive control for uavs collaborative payload transport," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 11 666–11 672. DOI: 10.1109/IROS45743.2020.9341541

[9] H. G. d. Marina and E. Smeur, "Flexible collaborative transportation by a team of rotorcraft," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1074–1080. DOI: 10.1109/ICRA.2019.8794316

[10] A. Tagliabue, M. Kamel, S. Verling, R. Siegwart, and J. Nieto, "Collaborative transportation using mavs via passive force control," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5766–5773. DOI: 10.1109/ICRA.2017.7989678

[11] M. Tognon, C. Gabellieri, L. Pallottino, and A. Franchi, "Aerial co-manipulation with cables: The role of internal force for equilibria, stability, and passivity," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2577–2583, 2018. DOI: 10.1109/LRA.2018.2803811

[12] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," *CoRR*, vol. abs/1912.12294, 2019. arXiv: 1912.12294. [Online]. Available: http://arxiv.org/abs/1912.12294

[13] S. V. Albrecht, F. Christianos, and L. Schäfer, *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024. [Online]. Available: https://www.marl-book.com

[14] E. J. J. Smeur, Q. Chu, and G. C. H. E. de Croon, "Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 3, pp. 450–461, 2016. DOI: 10.2514/1.G001490 [Online]. Available: https://doi.org/10.2514/1.G001490

[15] J. Gao et al., *Coohoi: Learning cooperative human-object interaction with manipulated object dynamics*, 2024. arXiv: 2406.14558 [cs.RO]. [Online]. Available: https://arxiv.org/abs/2406.14558

[16] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5738–5746. DOI: 10.1109/CVPR.2019.00589

[17] J. Mayer, J. Westermann, J. P. G. H. Muriedas, U. Mettin, and A. Lampe, "Proximal policy optimization for tracking control exploiting future reference information," *CoRR*, vol. abs/2107.09647, 2021. arXiv: 2107.09647. [Online]. Available: https://arxiv.org/abs/2107.09647

[18] S. Sun, A. Romero, P. Foehn, E. Kaufmann, and D. Scaramuzza, "A comparative study of nonlinear mpc and differential-flatness-based control for quadrotor agile flight," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3357–3373, 2022.

[19] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, "Robot operating system (ros): The complete reference (volume 1)," in A. Koubaa, Ed. Cham: Springer International Publishing, 2016, ch. RotorS—A Modular Gazebo MAV Simulator Framework, pp. 595–625, ISBN: 978-3-319-26054-9. DOI: 10.1007/978-3-319-26054-9_23 [Online]. Available: http://dx.doi.org/10.1007/978-3-319-26054-9_23

[20] S. Ross, G. J. Gordon, and J. A. Bagnell, *A reduction of imitation learning and structured prediction to no-regret online learning*, 2011. arXiv: 1011.0686 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1011.0686

[21] P. Foehn et al., "Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight," *Science Robotics*, vol. 7, no. 67, eabl6259, 2022. DOI: 10.1126/scirobotics.abl6259 eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.abl6259. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abl6259

Authorized licensed use limited to: TU Delft Library. Downloaded on May 04,2026 at 08:01:50 UTC from IEEE Xplore. Restrictions apply.