2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Enhancing Safety via Deep Reinforcement Learning in Trajectory Planning for Agile Flights in Unknown Environments

Lidia Rocha[1], Jorge Bidinotto[2], Fredrik Heintz[3], Mattias Tiger[3], and Kelen Vivaldini[1]

***Abstract*— Unmanned aerial vehicles (UAVs), known for their agile flight capabilities, require safe trajectory planning to achieve high-speed flights. This is necessary to swiftly evade obstacles and adapt trajectories under hard real-time constraints. These adjustments are essential to generate viable paths that prevent collisions while maintaining high speeds with minimal tracking errors. This paper addresses the challenge of enhancing the safety of agile trajectory planning. The proposed method combines a supervised learning approach, as teacher policy, with deep reinforcement learning (DRL), as student policy. Initially, we train the teacher policy using a path planning algorithm that prioritizes safety while minimizing jerk and flight time. Then, we use this policy to guide the learning of the student policy in various unknown environments. Testing in simulation demonstrates noteworthy advancements, including an 80% reduction in tracking error, a 31% decrease in flight time, a 19% increase in high-speed duration, and a success rate improvement from 50% to 100%, as compared to baseline methods.**

I. Introduction

Unmanned Aerial Vehicles (UAVs) represent highly capable platforms for achieving rapid and agile movements, as noted in recent studies [1]. There has been a notable surge in research endeavors pushing UAVs to achieve higher speeds [2], [3], emphasizing the need for advancements in control algorithms to ensure quicker response times [4], refine trajectory planning [5], and uphold flight safety [6].

Previous researches in agile trajectory planning have explored optimization-based [7], geometry-based [8], and sampling-based [9] approaches. More recently, machine learning models have been employed for predicting optimal guidance paths [4], [10]. However, it is essential to note that these efforts primarily operate in known environments and require extensive information for reliable outcomes, which is seldom the case in real-world missions.

To address this challenge, our contribution introduces a trajectory planning method for generating agile flight trajectories in unknown environments solely based on data from onboard sensors (Fig. 1). Our approach utilizes privileged reinforcement learning [10] to ensure high precision through expert-guided teacher policies, while enabling real-time predictions via the student policy. The framework integrates two neural networks: the teacher policy, trained via supervised learning, incorporates a geometry-based trajectory planning strategy enriched with a heuristic to optimize flight time and enhance safety. The teacher policy evaluates the training of the student policy, which exclusively processes real-time data collected by the UAV.

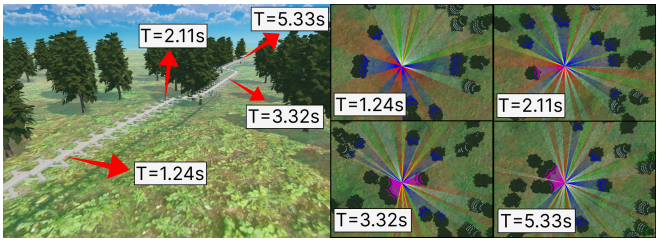


Fig. 1: Our approach concentrates on learning navigation policies from discrete maps. The primary task is to efficiently guide Unmanned Aerial Vehicles toward their goals, achieving minimal flight time and maximize safety, even when operating at high speeds in unknown environments. (Left) We conducted a high-speed flight in a software-in-the-loop simulator in a rainforest environment. (Right) The top view of the 3D Lidar sensor behavior at different time intervals.

We validate our approach through testing in 30 diverse synthetic scenarios, utilizing Software-in-the-Loop (SITL) simulations. SITL ensures real-world constraints are considered, involving simulations that integrate virtual software components. This approach allows for comprehensive software performance assessment without live deployment risks and facilitates a smooth transition from simulation to reality.

II. Related Work

Trajectory planning techniques for UAVs can be categorized into polynomial approaches, geometric-based methods, sampling-based approaches, optimization methods, and learning-based approaches [1]. Additionally, these techniques can be classified based on continuous-time or discrete-time representations of the state space [11].

Polynomial methods, such as those presented in [12], [13], work with continuous-time trajectories but face challenges in representing rapid changes [14]. In contrast, geometric, sampling, and optimization-based methods [15], [13], [16] operate in discrete-time spaces, allowing the utilization of full actuator capabilities [11]. However, computational demands arise when sampling across multiple time steps [17]. Learning-based approaches, exemplified by neural networks, offer a promising solution by effectively capturing rapid changes and actuator potential while mitigating computational challenges associated with discretization. These ap-

[1]Department of Computer, Federal University of São Carlos, Brazil lidiarocha@ieee.org, vivaldini@ufscar.br
[2]Department of Aeronautics Engineering, Federal University of São Paulo, Brazil jhbidi@sc.usp.br
[3]Department of Computer and Information Science, Linköping University, Sweden fredrik.heintz@liu.se, mattias.tiger@liu.se

proaches, as demonstrated by studies like [4], learn from data, and provide adaptability and the capability to handle complex dependencies.

One notable advantage of learning-based methods is their ability to be trained beforehand and used in real-world flights within milliseconds, facilitating quick responses for obstacle avoidance, especially at high speeds [18], [4]. The Deep Q-Network (DQN) serves as an effective example, demonstrating prowess in fast tracking, exploration, and obstacle avoidance in complex and uncertain environments [4], [19]. Techniques like Prioritized Experience Replay (PER) enhance DQN's performance [20], [21], [22], and privileged learning is emerging to enable safer decisions in unknown situations [18], [10].

The resulting trajectories often undergo optimization to adhere to system dynamics, frequently employing B-spline methods [17], [23], [24] that exploit UAVs' differential flatness properties [25]. These methods optimize for smoothness, dynamic feasibility, collision costs, and safety [24].

While trajectory planning methods traditionally focus on generating trajectories, the enhancement of safety is often assigned to control methods. However, combining safe trajectory planning and control safety significantly reduces accident rates and enhances mission success rates [26], [27]. Although several approaches can track reference trajectories [9], [28], they may struggle with disturbances during high-speed flight. Consequently, the development of Agilicious [29] addresses this gap, enhancing control for high-speed flights and demonstrating effectiveness in various works within the field [7], [11], [10].

III. Methodology

The primary aim of our proposed approach is to facilitate the safe and agile navigation of UAVs in unknown environments, leveraging 3D LiDAR for environmental perception. Our strategy involves establishing the UAV trajectory with a minimal flight time based on real-time sensory data while prioritizing safety.

Figure 2 illustrates the proposed privileged reinforcement learning framework aimed at addressing the challenges inherent in large-scale observations. Initially, a Deep Feedforward Neural Network (DFNN) serves as the teacher policy, trained using an expert algorithm proficient in determining the optimal trajectory within minimal time.

The teacher policy provides optimal action insights across diverse environments. It evaluates the student policy, a DRL approach without privileged information, using current sensor data and UAV state. This method is chosen for its adaptive response to dynamic environments and real-time feedback, optimizing safety protocols effectively. Initial supervised learning guidance refines policies iteratively, balancing exploration and exploitation to enhance robustness and applicability in diverse operational scenarios. DRL's ability to generalize and adapt ensures tailored safety solutions in complex environments [30].

The expert's output, the ideal next waypoint for each waypoint in the scenario, relies on a comprehensive understanding of the entire environment. This reliance introduces computational challenges during real-time execution, particularly evident in expansive environments, where the real-time generation of trajectories becomes unfeasible [31]. To address this, we employ the expert to train the teacher policy, a DFNN network designed to generalize data considering the entire environment, defining the model's behavior and strategies. The student policy, a Deep Q-Network with Prioritized Experience Replay (DQN-PER), operates based on data identifiable by the UAV's 3D Lidar around its current pose. Both networks output the new ideal waypoints to be followed (F, B, R, L, U, D). The distilled knowledge from the teacher policy is integrated into the student policy, functioning without privileged information. This approach ensures training with constrained data to mitigate overfitting and enhances adaptability to real-time environmental variations. Notably, this is particularly relevant when the UAV navigates unknown environments and continually explores new facets of the surroundings during each flight. This allows the student policy to surpass the constraints of the original teacher policy, enabling effective performance across a broader spectrum of scenarios and conditions, as in unknown environments.

A. Safety Trajectories for Agile Flights

Our focus is on formulating a practical and reliable trajectory planning strategy, aiming to optimize three key objectives: (1) minimizing jerk, (2) reducing flight time, and (3) enhancing safety by maximizing the distance to the obstacles in the environment.

The spatial domain is denoted as $W = \mathbb{R}^3$, with obstacles $O = \{O_1, \ldots, O_m\} \subset W$. The trajectory planning task involves determining a collision-free path for the UAV ($A \subset W$) connecting two locations within W while avoiding obstacles O. The state of the UAV in $W(t)$ is represented by $x(t)$, $y(t)$, and $z(t)$. The destination node is defined by x_{goal}, y_{goal}, z_{goal}, with $\mathbf{p}_{\text{goal}}$ as the position vector of the goal node, and $\mathbf{p}(t)$ representing the vector of the UAV's current state in W.

The UAV, as explained in [29], is a differential flatness model capable of navigating within a defined free space $C_{\text{free}} = C \setminus C_{\text{obs}}$. Here, C_{obs} represents configurations where the UAV encounters potential collisions ($\delta(\cdot,\cdot) \leq d_c$). The solution involves initiating from a configuration $x_s \in C_{\text{free}}$ and concluding at $x_e \in C_{\text{free}}$, with a continuous series of trajectories $\tau_i : [0,1] \rightarrow C_{\text{free}}$ satisfying the conditions $\tau_0(0) = x_s$, $\tau_N(1) = x_e$, and $\tau_{i-1}(1) = \tau_i(0)$.

The goal is to minimize equations (1), (2), and (3) while maximizing (4) to enhance safety and ensure collision-free trajectories for high-speed flights. In this context, n represents the total number of waypoints, k is the number of local obstacles, j iterates through all obstacles, p_{goal} signifies the goal node, p_i represents the current position of the UAV, D_{obs} signifies the distance between the current node and obstacles in W, D_{jerk} stands for the trajectory's jerk, D_{goal} represents the distance between the current node and the goal node, and D_{next} indicates the distance between the current

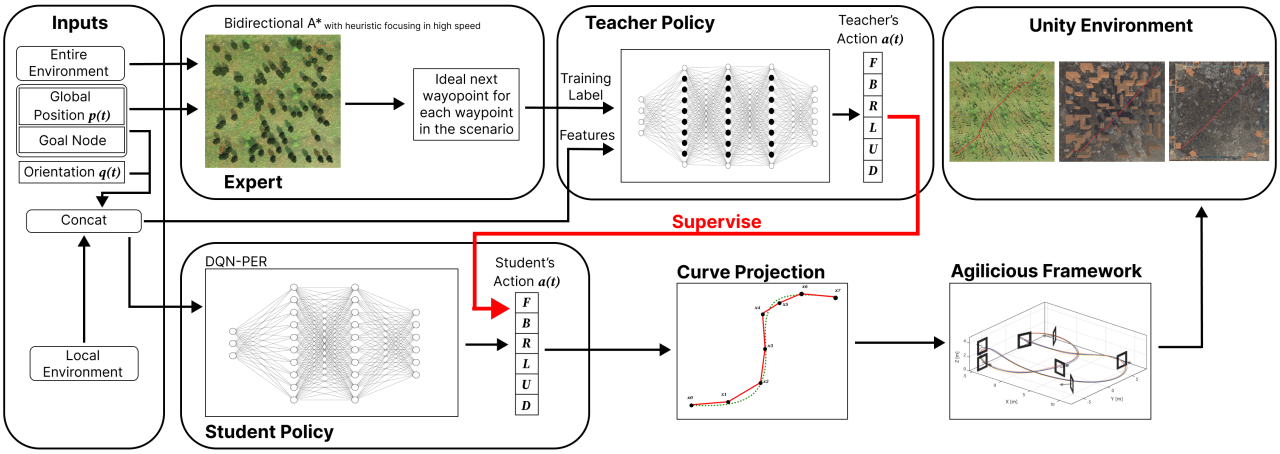


Fig. 2: Method overview. We train the teacher policy using expert data to identify optimal subsequent nodes that minimize flight time and enhance safety. We then transfer the distilled knowledge from the teacher to a student policy, facilitating evaluation through experience replay. Once the training process is complete, we employ the model to make trajectory predictions in new environments. These predictions are subsequently mapped onto a polynomial trajectory space, guiding the MPC system on Agilicious to navigate the track. Additionally, we can visualize the UAV's behavior within the Unity simulator.

node and the subsequent one. This collective set of variables orchestrates trajectory planning to achieve minimized jerk, reduced path length, and heightened safety considerations, contributing to the formulation of optimal flight paths.

$$D_{goal} = \sum_{i=0}^{n} \sqrt{(\mathbf{p}_{goal} - \mathbf{p}_i)^2} \tag{1}$$

$$D_{next} = \sum_{i=0}^{n-1} \sqrt{(\mathbf{p}_{i+1} - \mathbf{p}_i)^2} \tag{2}$$

$$D_{jerk} = \int (\frac{d^3\mathbf{p}}{dt^3})^2 \, dt \tag{3}$$

$$D_{obs} = max \sum_{j=0}^{k} \sqrt{(\mathbf{p}_i - O_j)^2} \tag{4}$$

Utilizing the bidirectional A* algorithm [32], we integrate these equations as heuristics for trajectory generation, as demonstrated in Equation 5. Equations 1 and 2 serve as conventional heuristics from the A* algorithm, guiding the search towards finding the shortest path. Equation 3 is employed to minimize trajectory jerk, promoting sustained high speeds and, consequently, reducing flight time. This jerk minimization also decreases the Root Mean Square Error (RMSE) by fostering smoother trajectories, enhancing safety through more accurate and predictable movements. Additionally, Equation 4 is incorporated to increase the distance between the current position and local obstacles. In this way, the approach ensures increased clearance around obstacles, offering resilience in scenarios where the controller may deviate from the guided trajectory at high speeds, further enhancing safety.

$$f_(i) = (D_{goal} + D_{obs} + D_{next}) * D_{jerk} \tag{5}$$

B. Teacher Policy

Our teacher policy is composed of a DFNN with three hidden layers, with 128 neurons in the first hidden layer, 64 in the second, and 32 in the third. A dropout rate of 0.5 is applied between these layers to promote generalization through dropout regularization, mitigating overfitting by stochastically deactivating neurons during training [33].

The policy is trained across multiple randomly generated scenarios that simulate environments where agile UAVs prove valuable. These environments include rainforests, mazes, and disaster areas, as outlined in [18]. To ensure precision, a distinct model is learned for each scenario, resulting in 30 models trained across different scenarios, with 10 models for each environment. Training involves 100 trajectories for each model, with randomly assigned start and goal nodes.

The inputs during training include the goal node, global position (p), orientation (q), and the environment around within a $5 \times 5 \times 2$ meter radius range around the global position. The output is the subsequent ideal action determined by our expert and is encoded in a one-hot encoding format. For instance, if the optimal action is to proceed forward, the output vector would be represented as $[1, 0, 0, 0, 0, 0]$. This subsequent ideal node serves as privileged information within our trajectory planning, functioning akin to an expert system, and provides valuable insights for efficient and safe trajectory planning. Furthermore, this configuration tailors the network to specialize in returning actions aimed at minimizing flight time and enhancing safety by increasing the distance between obstacles, as determined by our expert algorithm.

The teacher policy's outputs correspond to probabilities associated with optimal actions (a), including moving forward (F), backward (B), right (R), left (L), upward (U), and downward (D).

C. Student Policy

The student policy efficiently produces real-time, collision-free trajectories for agile flights, relying solely on onboard sensor measurements. These measurements encompass obstacle positions within a $5 \times 5 \times 2$ meter radius obtained from the 3D Lidar, UAV position and orientation data, and the goal node.

The student policy, structured as a DQN-PER, is designed to generalize UAV behavior across diverse scenarios with limited environmental information. This architecture excels in learning from experiences [10], prioritizing novel or challenging situations using the Prioritized Experience Replay mechanism. The dual hidden layer network (64 and 32 neurons) effectively captures complex relationships, trained with a batch size of 32 and an adaptive learning rate of 0.001 using the Adam optimizer [34]. A discount rate (γ) of 0.999 prioritizes long-term rewards.

Starting with an exploration rate (ϵ) of 1 and gradually decreasing by a factor of 0.995 to strike a balance between exploration and exploitation.This design facilitates comprehensive strategy discovery followed by a shift to exploitation. The policy adapts to new environments without extra information, determining optimal UAV actions by following a set of actions based on predefined rules and guidelines. The student policy learns from the teacher policy, emphasizing adaptability to diverse scenarios.

The student policy derives insights from the teacher policy, focusing less on specific situations for enhanced adaptability in new environments without requiring additional information. It autonomously determines the best UAV flight paths by prioritizing obstacle avoidance and minimizing flight time. Guided by a set of actions adhering to predefined rules and guidelines, the student policy leverages the teacher policy, trained by an expert algorithm, to navigate safely in varied conditions while optimizing flight efficiency.

In real missions, the student policy guides the UAV in unknown environments. Upon generating a trajectory, it utilizes Bézier curves to transform the anticipated trajectory into a comprehensive state representation. This transformation is executed autonomously for each axis, ensuring the management of multi-dimensional motion complexities. Bézier curves, known for continuity and smoothness, align with the intrinsic property of differential flatness in UAVs [25].

For instance, we explore the details of the x-axis, specifically defining the Bézier curve projection (Eq. 6). Here, $\mu_x(t)$ signifies the trajectory projected onto the Bézier curve for the x-axis, $B_i^N(t)$ denotes the Bernstein basis functions of order N, and P_i represents the control points influencing the shape of the Bézier curve. This process is replicated for all axes, ensuring a comprehensive representation of trajectories.

$$\mu_x(t) = \sum_{i=0}^{N} B_i^N(t) \cdot C_i \tag{6}$$

Strategically positioned control points (C_i) ensure the curve captures sudden trajectory shifts [35], aligning with orientation changes for seamless attitudes. This design guarantees consistent attitude transitions by harnessing the constrained acceleration characteristic of cubic B-splines. The differential flatness of quadrotors enables bounded acceleration to translate into continuous attitudes, facilitating accurate trajectory tracking.

During flight, the UAV records obstacle positions, triggering policy execution upon identifying new obstacles. The policy generates a new trajectory using obstacle positions, providing a single waypoint per iteration. Multiple policy runs are conducted to create the final trajectory. Obstacle positions are updated based on the UAV's existing knowledge; undiscovered areas are treated as obstacle-free.

The privileged learning framework prioritizes a dynamic and context-aware strategy, intentionally aligning diverse elements for effective learning. The student policy, derived from a neural network, dynamically adapts to disturbances, ensuring robust flight performance even in challenging conditions [4].

IV. Results

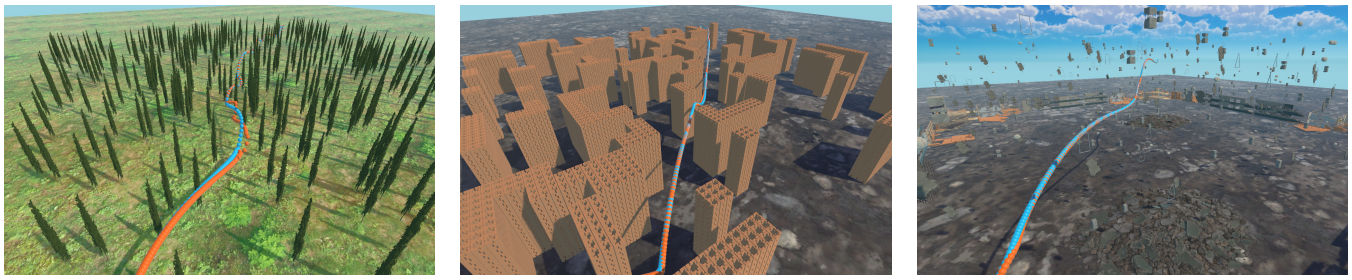
The proposed method is evaluated based on: flight time (seconds), processing time (seconds) - the time taken by the algorithm to return the whole trajectory, CPU frequency (MHz), memory usage (MB), the duration the UAV is in high speed (T_{high_speed}) in seconds - representing the time the UAV is at least at 80% of the maximum speed, RMSE from the guidance trajectory (meters), and success rate (%) as metrics.

Our proposed methodology is compared against several baseline methods, serving as benchmarks for our approach. These baselines encompass the standard bidirectional A* in an unknown environment, our expert operating in an unknown environment, the original DQN-PER, and our teacher policy. Additionally, we conduct a comparison with an optimal algorithm represented by our expert in a known environment.

The experiments involve 10 new scenarios within each environment mentioned in Section III-B, with each environment having dimensions of (100, 100, 25) meters. The start node in all environments is located at (5, 5, 5) meters, and the goal node is at (95, 95, 5) meters, except for the disaster area where the goal node is at (95, 95, 20) meters. Figure 3 illustrates the environments alongside the trajectories generated by both the student policy and the optimal algorithm.

A. Experimental Design

The simulations utilize Agilicious [29], an open-source and open-hardware framework supporting model-based and neural-network-based controllers for high-speed flights [36]. Agilicious employs Blade-Element-Momentum (BEM) theory for accurate modeling of lift and drag produced by each rotor based on the platform's current ego-motion and individual rotor speeds [37]. This framework offers high thrust-to-weight and torque-to-inertia ratios, ensuring agility, and features a real-time flight controller.



(a) Rainforest (b) Cave (c) Disaster 3D

Fig. 3: The environments used for baseline comparisons. The displayed trajectories correspond to the student policy (orange) and the optimal algorithm (blue).

TABLE I: Comparison between baseline algorithms and the proposed approach in varied scenarios.

Env	Algorithm	Flight Time (s)	Process Time (s)	CPU (MHz)	Memory (MB)	T_{high_speed} (s)	RMSE (m)	SR(%)
Rainforest	Standard Bi A*	55.31±1.77	2.10±0.13	**1253±259**	**191.5±20.6**	73.63±1.27	10.95±0.36	60
	Our Bi A*	46.49±3.85	3.10±0.31	1733±624	193.6±19.8	74.20±2.73	9.90±0.83	80
	DQN-PER	∞	∞	∞	∞	∞	∞	0
	Teacher Policy	40.17±3.12	0.30±0.07	1595±540	192.4±19.3	73.90±4.15	7.94±1.05	**100**
	Our Approach	32.10±3.08	**0.27±0.07**	1566±565	193.1±20.0	77.49±3.28	6.12±0.67	**100**
	Optimal	**26.24±1.88**	6.12±0.74	3276±492	321.87±26.31	**81.24±3.02**	**5.23±1.62**	**100**
Maze	Standard Bi A*	27.11±1.42	4.39±1.04	**1182±225**	**158.8±14.2**	71.15±0.98	13.23±0.97	50
	Our Bi A*	21.16±2.12	4.76±0.60	2480±332	193.5±19.6	71.89±2.46	11.50±1.12	90
	DQN-PER	∞	∞	∞	∞	∞	∞	0
	Teacher Policy	19.75±1.89	**0.27±0.12**	1566±394	192.0±19.7	73.36±4.63	8.19±0.76	**100**
	Our Approach	**16.98±1.95**	**0.28±0.06**	1262±330	193.9±18.6	**75.11±3.79**	7.08±0.94	**100**
	Optimal	**16.51±1.17**	7.40±0.80	3554±412	314.28±22.17	**75.31±3.56**	**5.88±1.92**	**100**
Disaster Area	Standard Bi A*	22.70±1.85	3.27±0.24	**1073±68**	195.1±18.99	70.75±1.62	8.78±0.71	80
	Our Bi A*	20.75±2.23	6.49±4.86	1397±177	170.6±14.0	71.50±2.31	7.46±1.19	80
	DQN-PER	∞	∞	∞	∞	∞	∞	0
	Teacher Policy	19.43±2.10	0.37±0.08	1389±183	166.6±11.8	84.16±4.01	6.79±1.01	**100**
	Our Approach	**17.82±2.01**	**0.32±0.10**	1376±193	**160.5±11.6**	86.59±2.92	4.99±0.98	**100**
	Optimal	**16.33±1.82**	18.23±4.91	2917±245	302.74±19.38	**89.67±3.77**	**3.97±2.11**	**100**

The UAV model, as mentioned in Section III-A, uses a Model Predictive Controller (MPC) as a high-level controller and Betaflight[1] as a low-level controller. Betaflight facilitates precise tracking of body rate and collective thrust commands, utilizing high-frequency IMU (Inertial Measurement Unit) measurements [37].

Simulations leverage Agilicious for physics and aerodynamics, while Unity [38] handles visualization. Environments are randomly generated following the approach defined in [18]. Sensors, including the 3D Lidar Velodyne VLP-16 and IMU[2], are implemented in Unity for localization. Communication between Unity and Agilicious is achieved through integration between the Robot Operating System (ROS) [39] and the MQTT [40] framework. This setup enables the use of ROS as the communication base, allowing data from Unity to mimic real-world sensor inputs. The received data serve as input for our algorithm, which communicates with the controller to determine optimal UAV actions. The resulting positions are replicated in Unity, providing a virtual representation emulating a real environment.

[1]https://betaflight.com/

[2]https://github.com/Field-Robotics-Japan/UnitySensors

B. Baseline Comparisons

This section demonstrates the effectiveness of our approach in navigating environments exclusively relying on onboard UAV information. The comparison of baseline methods is in Table I. The optimal algorithm, our approach (the student policy) and the teacher policy achieved a remarkable 100% success rate across all environments. In contrast, the DQN-PER encountered difficulties in a 1-hectare area, failing to find a collision-free path despite extensive training iterations, attributed to its reliance solely on onboard information without privileged insights.

Our bidirectional A* algorithm, while proficient in finding collision-free paths, struggled with sharp turns created due to sudden obstacle discovery, leading to crashes, and yielding a success rate between 80% and 90%. The standard Bidirectional A* achieved a success rate of up to 50%. This emphasizes the significance of incorporating the heuristics proposed in Section III-A. The jerk heuristic governs UAV acceleration changes, promoting smoother transitions. Trajectories with lower jerk values enhance stability, reduce unpredictable movements, and minimize mechanical stress. Furthermore, by maximizing the distance to obstacles, the UAV is afforded a margin for slight deviations from the guided path with low collision risk.

Upon scrutinizing computational processing, it is evident that, generally, the techniques exhibit minimal distinctions in terms of CPU or memory utilization in unknown environments as the flights take place in unknown environments, and solely process information about nearby obstacles. The low number of obstacles results in a correspondingly low computational cost, rendering it feasible for onboard processing and facilitating fully autonomous flights. However, the optimal algorithm runs in known environments with several obstacles, and despite the low memory usage, it requires a high CPU usage, which poses challenges when integrating with other algorithms such as mapping or computer vision algorithms with onboard compute.

Considering the processing time, the optimal algorithm underperforms in large environments, as it considers all obstacles, leading to longer computation times. While the standard and our Bidirectional A* algorithm in an unknown environment demonstrate reduced processing times, handling fewer obstacles per iteration. Our approach significantly outperforms both, generating trajectories with considerably less processing time. Moreover, our approach demonstrates comparable processing times to the teacher policy, consistently producing faster trajectories across diverse environments.

The generation of faster trajectories is especially apparent in rainforest environments, where closely spaced obstacles prompt the UAV to decelerate. As described in Section III-C, the UAV detects obstacles solely during flight, necessitating frequent trajectory adjustments. Both Bidirectional A* algorithm, lacking consideration for the UAV's current orientation, leads to unnecessary turns that contribute to a decrease in speed. This is particularly evident in the standard Bidirectional A*, which prioritizes only the shortest path without incorporating heuristics to enhance speed during the flight.

In contrast, our approach takes into account the UAV's orientation, emphasizing actions that minimize jerk during flights. Notably, it demonstrates superior adaptability to dynamic environments compared to classical algorithms. This adaptability stems from its ability to continuously learn and adjust trajectories, effectively accommodating changes over time. Unlike traditional methods dependent on predetermined data or fixed rules, our approach seamlessly adapts to new information. However, our approach had a sacrifice on flight time between 2% and 19% compared to the optimal algorithm operating in a known environment within our scenarios.

The discernible difference in speed retention among the techniques becomes apparent when evaluating the duration each method maintains high speed. Our approach demonstrates the capability to sustain high speed for an additional 5% of the time in both maze and rainforest environments, and for 19% longer in disaster areas compared to baseline algorithms. The reduction compared to the optimal algorithm is marginal, standing at 4.6%, 0.3%, and 3.4% in each respective environment. This analysis is visually depicted in Figure 4, portraying the speed profiles of each algorithm throughout the flights in a randomly selected environment within the evaluated scenarios. To ensure a consistent scale for comparison, we performed linear interpolation along the trajectory, considering the varied duration of each flight.

The velocity profiles in the X and Y axes graphs illustrate that our approach consistently maintains a higher speed than the baseline methods throughout the entire trajectory. Our method not only achieves maximum speed in most trajectories among the baselines but also surpasses them in terms of sustained speed. Notably, our approach exhibits results comparable to the optimal algorithm, with the primary difference being that the optimal algorithm attains higher speeds even during low-speed flight.

In terms of the Z-axis, little differentiation is observed among the approaches, particularly in the 2D environments. It can be asserted that all techniques demonstrated similar stability along the Z-axis. However, in the disaster environment, characterized by a more significant variation in the Z-axis, our approach exhibited movements closely resembling those of the expert, while the teacher policy maintained a lower velocity. This discrepancy may be attributed to potential overfitting in the network, hindering its optimal adaptation to the test scenario.

Also, an analysis incorporating the RMSE between the proposed trajectory for each algorithm and the UAV's actual trajectory was conducted. The total error along the entire trajectory for each algorithm is presented in Table I, revealing a reduction by 80% in error for our approach compared to the baseline methods. However, there is still a difference by 25% from the optimal algorithm, whereas the baselines exhibit errors exceeding 200% in comparison to the optimal algorithm.

These results emphasize the robustness and reliability of our approach, with increased precision contributing to heightened safety. While the teacher policy achieved comparable results, our method maintains higher speeds for longer. Also, more sophisticated networks may be necessary in complex environments, increasing prediction times beyond what is feasible for high-speed real-time flights.

Our jerk heuristic, prioritizing smoother trajectories with lower acceleration changes, also plays a pivotal role in enhancing trajectory stability, minimizing unpredictable movements, and reducing mechanical stress on the system, thereby contributing to heightened safety during UAV flights. This precision is reflected in the small RMSE results, demon-

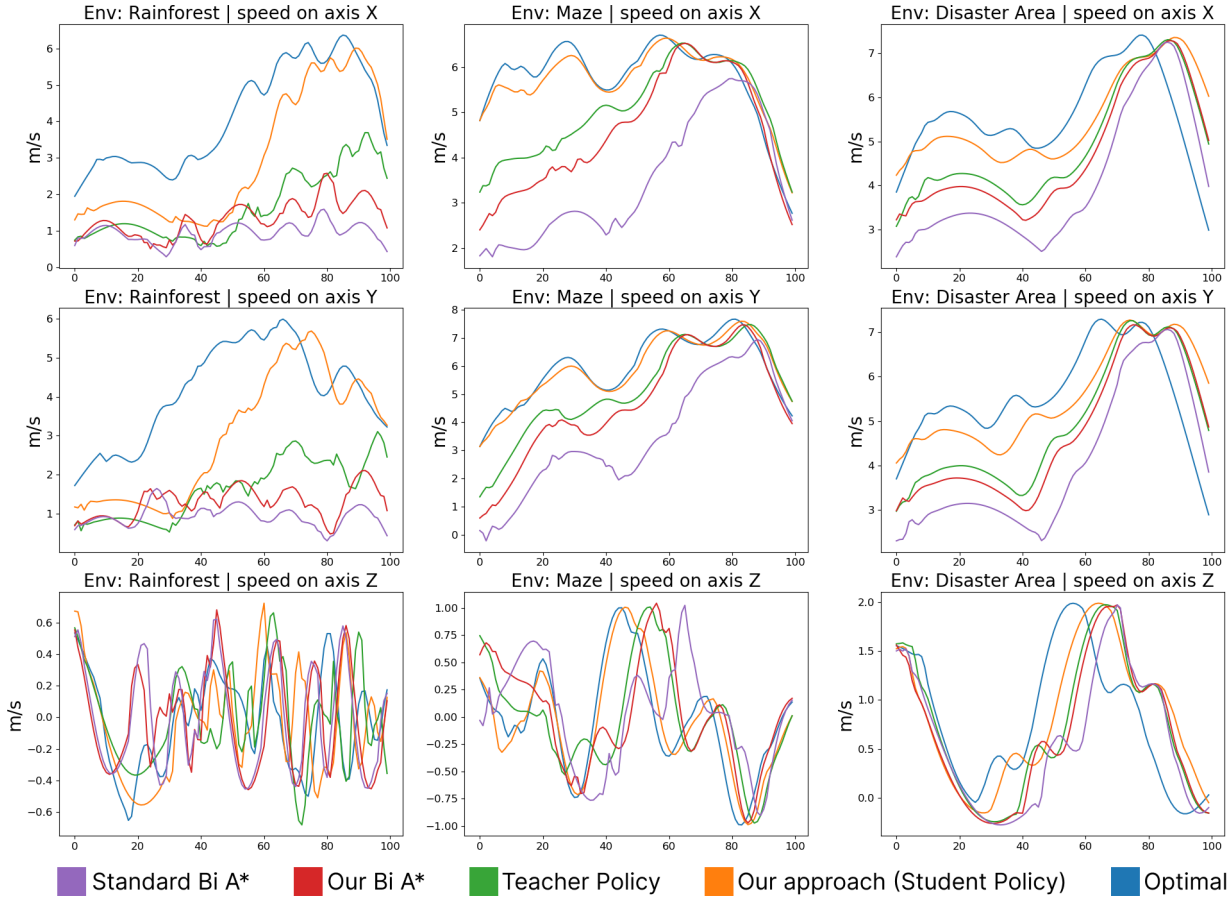


Fig. 4: Comparing the speed in each axis between our approach and the baseline methods across all the environments.

strating the accuracy and effectiveness of our trajectory planning algorithm. Additionally, the heuristic that increases the distance to obstacles contributes to overall safety by establishing a safety margin, permitting minor deviations from the guided trajectory without causing collisions. This is highlighted by the results from the standard Bidirectional A* algorithm, which, without considering any of our heuristics, exhibited imprecise trajectory tracking by the controller, resulting in collisions despite finding collision-free paths.

V. Conclusion

This paper introduces an approach focused on deriving optimal trajectories that minimize jerk and flight time, and enhance safety in unknown environments. Our method leverages a privileged reinforcement learning framework in tandem with a deep reinforcement learning network, yielding a robust policy for generating agile and safe trajectories within unknown settings. This paper showcases the efficacy of our approach by demonstrating its ability to produce reliable trajectories for high-speed flights, relying solely on onboard UAV information, achieving a 100% success rate. Additionally, the proposed method significantly improves the trajectory performance during high-speed flight, leading to a substantial improvement in success rate from 50% to 100%, an 80% reduction in RMSE for trajectory tracking, a 31% reduction in flight time, and a 19% increase in the duration of high-speed trajectory maintenance compared to baseline methods. Moreover, our approach delivers a reliable trajectory within 0.4 seconds. These findings underscore the promise of employing an end-to-end policy for crafting secure guiding paths, addressing real-world challenges in real-time scenarios.

Acknowledgment

The authors would like to acknowledge for their financial support from the Federal Agency for Support and Evaluation of Graduate Education (CAPES), the National Council for Scientific and Technological Development (CNPQ), and the Swedish-Brazilian Research and Innovation Center (CISB) grants n.o 200056/2022-0 and n.o 200051/2022-9. We also extend our thanks to The Reasoning and Learning (ReaL) group at Linköping University for their invaluable contributions.

References

[1] L. Rocha, M. Saska, and K. Vivaldini, "Overview of uav trajectory planning for high-speed flight," in *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2023, pp. 110–117.

[2] Y. Song, M. Steinweg, E. Kaufmann, and D. Scaramuzza, "Autonomous drone racing with deep reinforcement learning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 1205–1212.

[3] J. Fu, Y. Song, Y. Wu, F. Yu, and D. Scaramuzza, "Learning deep sensorimotor policies for vision-based autonomous drone racing," *arXiv preprint arXiv:2210.14985*, 2022.

[4] R. Penicka, Y. Song, E. Kaufmann, and D. Scaramuzza, "Learning minimum-time flight in cluttered environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7209–7216, 2022.

[5] X. Zhou, C. Xu, and F. Gao, "Automatic parameter adaptation for quadrotor trajectory planning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 3348–3355.

[6] A. Saviolo and G. Loianno, "Learning quadrotor dynamics for precise, safe, and agile flight control," *Annual Reviews in Control*, 2023.

[7] R. Penicka and D. Scaramuzza, "Minimum-time quadrotor waypoint flight in cluttered environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5719–5726, 2022.

[8] X. Gu, B. Xian, and Y. Wang, "Agile flight for a quadrotor via robust geometry control: Theory and experimental verification," *International Journal of Robust and Nonlinear Control*, vol. 32, no. 7, pp. 4236–4250, 2022.

[9] P. Foehn, D. Brescianini, E. Kaufmann, T. Cieslewski, M. Gehrig, M. Muglikar, and D. Scaramuzza, "Alphapilot: Autonomous drone racing," *Autonomous Robots*, vol. 46, no. 1, pp. 307–320, 2022.

[10] Y. Song, K. Shi, R. Penicka, and D. Scaramuzza, "Learning perception-aware agile flight in cluttered environments," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 1989–1995.

[11] P. Foehn, A. Romero, and D. Scaramuzza, "Time-optimal planning for quadrotor waypoint flight," *Science Robotics*, vol. 6, no. 56, p. eabh1221, 2021.

[12] Z. Han, Z. Wang, N. Pan, Y. Lin, C. Xu, and F. Gao, "Fast-racing: An open-source strong baseline for se(3) planning in autonomous drone racing," 2021.

[13] A. Romero, R. Penicka, and D. Scaramuzza, "Time-optimal online replanning for agile quadrotor flight," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7730–7737, 2022.

[14] S. LaValle, "Planning algorithms: Cambridge university press, 2006," 2006.

[15] Y. Song and D. Scaramuzza, "Policy search for model predictive control with application to agile drone flight," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2114–2130, 2022.

[16] J. Tordesillas, B. T. Lopez, and J. P. How, "Faster: Fast and safe trajectory planner for flights in unknown environments," in *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2019, pp. 1934–1940.

[17] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3529–3536, 2019.

[18] A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza, "Learning high-speed flight in the wild," *Science Robotics*, vol. 6, no. 59, p. eabg5810, 2021.

[19] T. Zhang, G. Kahn, S. Levine, and P. Abbeel, "Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 528–535.

[20] L. Liu, B. Tian, X. Zhao, and Q. Zong, "Uav autonomous trajectory planning in target tracking tasks via a dqn approach," in *2019 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, 2019, pp. 277–282.

[21] S. Li, F. Wu, S. Luo, Z. Fan, J. Chen, and S. Fu, "Dynamic online trajectory planning for a uav-enabled data collection system," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 12, 2022.

[22] A. Khan, J. Zhang, S. Ahmad, S. Memon, B. Hayat, and A. Rafiq, "Dqn-based proactive trajectory planning of uavs in multi-access edge computing," *Comput. Mater. Contin*, vol. 74, pp. 4685–4702, 2023.

[23] B. Zhou, F. Gao, J. Pan, and S. Shen, "Robust real-time uav replanning using guided gradient-based optimization and topological paths," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1208–1214.

[24] B. Zhou, J. Pan, F. Gao, and S. Shen, "Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1992–2009, 2021.

[25] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011.

[26] S. Liu, K. Guo, X. Yu, L. Ma, L. Xie, and L. Guo, "Safe maneuvering planning for flights in complex environments," *IEEE Transactions on Industrial Electronics*, 2023.

[27] J. Jia, K. Guo, X. Yu, L. Guo, and L. Xie, "Agile flight control under multiple disturbances for quadrotor: Algorithms and evaluation," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 58, 2022.

[28] A. Loquercio, A. Saviolo, and D. Scaramuzza, "Autotune: Controller tuning for high-speed flight," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4432–4439, 2022.

[29] P. Foehn, E. Kaufmann, A. Romero, R. Penicka, S. Sun, L. Bauersfeld, T. Laengle, G. Cioffi, Y. Song, A. Loquercio, and D. Scaramuzza, "Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight," *AAAS Science Robotics*, 2022.

[30] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, 2021.

[31] L. Rocha, M. Aniceto, I. Araújo, and K. Vivaldini, "A uav global planner to improve path planning in unstructured environments," in *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2021, pp. 688–697.

[32] Z. Xie, X. Zhao, Z. Jiang, H. Yang, and C. Li, "Trajectory planning and base attitude restoration of dual-arm free-floating space robot by enhanced bidirectional approach," *Frontiers of Mechanical Eng.*, 2022.

[33] A. H. Qureshi and M. C. Yip, "Deeply informed neural sampling for robot motion planning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018.

[34] Y. Zhu, T. Hayashi, and Y. Ohsawa, "Gradient descent optimization by reinforcement learning," in *Proceedings of the Annual Conference of JSAI 33rd (2019)*. The Japanese Society for Artificial Intelligence, 2019, pp. 2H4E204–2H4E204.

[35] L. R. Howell and B. D. Allen, "Spline trajectory algorithm development: Bézier curve control point generation for uavs," in *8th AIAA Atmospheric and Space Environments Conference*, 2016, p. 4197.

[36] M. Faessler, A. Franchi, and D. Scaramuzza, "Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories," *IEEE Robotics and Automation Letters*, 2018.

[37] 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 Trans. on Robotics*, 2022.

[38] M. S. P. De Melo, J. G. da Silva Neto, P. J. L. Da Silva, J. M. X. N. Teixeira, and V. Teichrieb, "Analysis and comparison of robotics 3d simulators," in *2019 21st Symposium on Virtual and Augmented Reality (SVR)*. IEEE, 2019, pp. 242–251.

[39] A. Koubâa *et al.*, *Robot Operating System (ROS)*. Springer, 2017, vol. 1.

[40] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, "Mqtt-s—a publish/subscribe protocol for wireless sensor networks," in *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08)*. IEEE, 2008, pp. 791–798.