Towards Event-Driven, End-to-End UAV Tracking Using Deep Reinforcement Learning

Ala Souissi, Hajer Fradi, Panagiotis Papadakis

Towards Event-Driven, End-to-End UAV Tracking Using Deep Reinforcement Learning

Ala Souissi
IMT Atlantique
Lab-STICC, UMR CNRS 6285
Brest, Brittany, France
ala.souissi20@gmail.com

Hajer Fradi
IMT Atlantique
Lab-STICC, UMR CNRS 6285
Brest, Brittany, France
hajer.fradi@imt-atlantique.fr

Panagiotis Papadakis
IMT Atlantique
Lab-STICC, UMR CNRS 6285
Brest, Brittany, France
panagiotis.papadakis@imt-atlantique.fr

ABSTRACT

To enable autonomous active tracking by a drone in applications where RGB cameras would be insufficient, in this paper, we propose an event-based pipeline exhibiting enhanced robustness and performance in adverse and highly dynamic conditions. In particular, we design a event data-driven controller trained end-to-end via deep reinforcement learning to respond to raw event streams to follow a target/leader drone. To favor policy generalization across diverse conditions, the agent is exposed during training to intensive domain randomization in terms of scene layout, as well as target trajectory shape and dynamics. Experiments are performed in simulation in a framework reminiscent of search and rescue scenarios by training and testing in simulated DARPA subterranean challenge environments, showing superior performance compared to conventional RGB-based tracking under high-speed and low-lighting. Related source-code is available at: https://github.com/AlaSouissi5/DRL-Drone-Tracking.

KEYWORDS

Event cameras, tracking, UAV, Deep Reinforcement Learning

1 INTRODUCTION

Unmanned aerial vehicles (UAVs), commonly known as drones, have been increasingly used in humanitarian missions, including search and rescue, safety surveillance, emergency contingency planning, and guidance tasks [1, 2, 8, 9, 12, 16]. This rising interest emphasizes the need for autonomous navigation skills that can be complementary to manual control. A key component of this autonomy is active tracking, which keeps the target centered in the field of view (FOV) using mostly visual input [18]. Such tracking capabilities support applications like search and rescue, where a tele-operated drone can lead exploration while autonomous drones follow in GPS-denied environments [14, 25].

Active tracking with drones is challenging due to the complexity of the task and the system nonlinear dynamics. Drone dynamics are affected by uncertain environmental conditions, aerodynamic forces, payload changes, and control noise. In rapidly changing environments, the difficulty of maintaining accurate tracking increases, making it challenging to develop controllers that ensure stable performance. Earlier works mostly employed Proportional-Integral-Derivative (PID) control [23] or Linear Quadratic Regulator (LQR) [19], but these require highly accurate drone models, which are often difficult to obtain in practice.

Recent developments in the field have explored the use of reinforcement learning (RL). Unlike classic controllers, RL-based trackers adapt to environmental changes by learning from past experiences, enabling model refinement without altering system components. Additionally, a highly accurate dynamic model is not required, as RL algorithm can learn from data even when the model does not fully capture the system complexities. However, these approaches, including [7, 24], typically require access to the target state,which is challenging to estimate accurately during continuous motion of both the tracker and the target. Instead of training a detection-based policy, an alternative approach consists in developing end-to-end, deep reinforcement learning-based controllers [4, 6, 26].

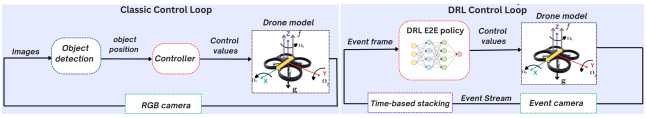

Figure 1: Conventional RGB-based controller for UAV active tracking vs event-based, DRL controller.

Building on these works showing the potential of RL for control tasks using RGB images and motivated by the recent success of event cameras in vision tasks [10], in this work we develop an event-based, end-to-end deep reinforcement learning controller (cf. Fig. 1 for the two paradigms). In particular, we employ event-camera data to address intrinsic RGB camera limitations, such as low frame rate and limited dynamic range. Their high dynamic range, temporal resolution, and low latency have drawn growing interest, particularly in autonomous vehicles where fast response, adaptability to lighting, and robust perception are essential [11].

Following standard practice, training is conducted in simulation and to favor real-world transfer, training is complemented by domain randomization and parallel training to manage complexity. In comparison to D-VAT [6] which is conceptually the closest to our work, this article contributes in the following directions:

- Use of event frames instead of RGB frames as input modality.
- Experiments in DARPA SubTerranean Challenge scene layouts [3] alongside box-like environments and comparisons against [6] in adverse conditions.

The following sections provide a summary of the problem statement (Section 2), the evolution of D-VAT to an event-based end-to-end architecture (Section 3) and the corresponding experiments (Section 4) and conclusion.

2 PROBLEM STATEMENT

2.1 Dynamic Quadrotor Model

Following [20], the quadrotor model represents the drone as a rigid body with 6 degrees of freedom, namely, 3 translational and 3 rotational along the 3D body axes. This model is controlled by a scalar f representing the total thrust value along the Z axis, and angular velocity expressed in the fixed body frame as $\mathbf{\Omega} = (\Omega_1, \Omega_2, \Omega_3)$.

The quadrotor state consisting of position, velocity, and orientation, is defined by the following differential equations:

$$\ddot{\mathbf{p}}(t) = \mathbf{R}_3(t)\frac{f(t)}{m} + \mathbf{g}, \tag{1}$$

$$\dot{\mathbf{R}_3}(t) = \mathbf{R}(t)[\mathbf{\Omega}(t)]_X, \tag{2}$$

$$\mathbf{\Omega}_\times = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix}, \tag{3}$$

where $\mathbf{p}(t)$, $\mathbf{R}(t)$, and $\mathbf{\Omega}(t)$ denote the absolute position, orientation, and angular velocity of the tracker drone at time t, respectively. $\mathbf{R}_j(t)$ is the j-th column of the orientation matrix $\mathbf{R}(t)$. The total thrust is $f(t)$, and m is the drone mass. The gravity vector is $\mathbf{g} = [0\ \ 0\ \ -9.8]^\top$ m/s^2, and $[\mathbf{\Omega}(t)]_X$ is the skew-symmetric matrix of $\mathbf{\Omega}(t)$. The thrust $f(t)$ and angular rates are constrained within the following ranges: $0 \leq f_{\text{min}} \leq f \leq f_{\text{max}}$, $-\Omega_{\text{max}} < \Omega_i < \Omega_{\text{max}}$ for $i \in \{1, 2, 3\}$. The limit values f_{min}, f_{max}, and Ω_{max} can be set to match certain drone specifications.

2.2 Reinforcement Learning-based Tracking

For UAVs, the active tracking problem involves using visual sensory data to generate actions that keep the target centered and maintain a desired distance. We formulate UAV active tracking as a Partially Observable Markov Decision Process (POMDP) [17], an extension of the Markov Decision Process (MDP). POMDP is defined by the tuple $(\mathbb{S}, \mathbb{A}, \mathbb{O}, T, R, Z, \gamma)$, where $\mathbb{S}$, $\mathbb{A}$, and $\mathbb{O}$ are the state, action, and observation spaces, respectively. The discount factor γ balances immediate and future rewards. T is the transition probability, Z is the observation model, and R is the reward function.

At each time step, the agent (UAV) interacts with the environment as follows: (i) it receives a visual observation $o_t \in \mathbb{O}$; (ii) based on this observation, it selects an action using a stochastic policy $a_t \sim \pi(a_t \mid o_t)$; (iii) it receives an immediate reward $r_{t+1} = R(s_{t+1})$ based on the new state and a new observation $o_{t+1} \sim Z(o_{t+1} \mid s_{t+1})$ correlated with that state. This formulation aims to find the optimal policy π^* that maximizes the expected cumulative reward. In the following, we define the observation, state, and action spaces:

Observation Space: The observation derived from visual sensors is a sequence of the N latest images $O(t) = (I(t-N+1), ..., I(t))$, where $I(t)$ is the current image. The observation space is: $\mathbb{O} = (H, W, C)^N$, with $W \times H$ as spatial resolution and C as number of channels. **Action Space:** The actions are the control decisions for tracking, defined as $a_t = (f(t), \Omega(t))$. The action space is continuous and is defined as: $\mathbb{A} = \underbrace{[0, f_{\text{max}}]}_{\text{thrust}} \times \underbrace{[-\Omega_{\text{max}}, \Omega_{\text{max}}]^3}_{\text{angular rates}} \subset \mathbb{R}^4$.

State Space: The state space at time t is defined as $s_t = (P_t, V_t, A_t) \in \mathbb{R}^9$, where P_t, V_t, and A_t are the relative position, velocity, and acceleration of the target with respect to the tracker in 3D.

2.3 Asymmetric Soft Actor-Critic Framework

The learning framework is based on the asymmetric formulation of the Soft Actor-Critic algorithm, as shown in Fig. 2. In its basic

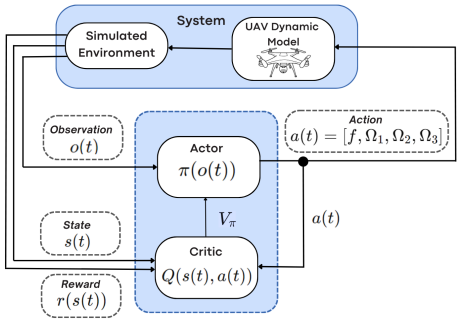

Figure 2: The flowchart of Asymmetric Soft Actor-Critic (ASAC) learning framework: the actor and critic networks work together to optimize the policy.

version, actor-critic uses a stochastic policy to generate actions and a critic to evaluate them using a Q-function. Both are implemented as neural networks and trained jointly. Soft actor-critic extends this by maximizing both expected rewards and policy entropy, encouraging a balance between exploration and exploitation. The asymmetric formulation uses different inputs for the actor and critic. In robotics, the policy often relies on partial, noisy sensor observations, while the critic, during training, has access to the full system state. This setup, adopted in some prior works, namely, [5, 6, 21], enhances learning by allowing the critic to provide more accurate feedback, despite the actor limited input. The resulting asymmetric actor-critic algorithm improves convergence and overall learning efficiency.

Building on the asymmetric formulation, we adopt the reward from D-VAT [6], where the objective is to maintain the target centered in the tracker field of view at an optimal distance. The defined reward encourages alignment of the target relative position along the three axes of the tracker body frame. Additional penalties are considered to discourage high speeds and collisions. This design promotes stable and efficient target tracking; for full formulation details, refer to [6].

3 END-TO-END EVENT-BASED TRACKING ARCHITECTURE

Within the presented ASAC framework, we train a single end-to-end model to generate actions directly from the raw event data. The overall architecture is shown in Fig. 3.

3.1 Event Data Processing

Compared to conventional cameras capturing images at a fixed frame rate, event cameras respond to brightness changes for every pixel asynchronously and independently. This results in a stream of events that are spatially sparse and asynchronous. Each event is a tuple $e = (t, x, y, p)$, where t is the timestamp at which the event is

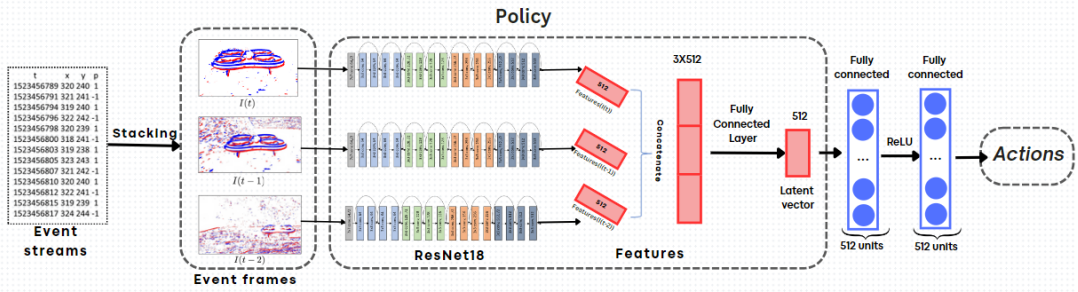

Figure 3: Overview of the event-driven end-to-end UAV active tracking using deep reinforcement learning.

triggered, (x, y) are the spatial coordinates, and p is the polarity indicating the sign of the change. To align with conventional image-based vision, event streams need to be transformed into a 2D spatial grid representation. A common way to represent events is time-based stacking which involves incorporating a sequence of events $E = \{e_i \mid t \leq i < t + \Delta t\}$ within a time interval Δt, resulting in an event frame. To train our policy, the observation is defined as a sequence of the N latest event frames ($N = 3$, in our case): $O(t) = (I(t-2), I(t-1), I(t))$. To process event frames and capture task-relevant features, the policy architecture is designed as follows.

3.2 Neural Networks of Actor and Critic

Actor-NN: The actor network consists of two blocks. The first block is a feature extractor that maps observations (event frames) to a feature vector, capturing spatial and temporal information. The resulting feature space has a higher correlation with the state space, giving a richer representation of the current state. Event frames, used as observations, are processed by ResNet18 [13] as the feature extractor, following [6], for its balance of complexity and performance. Asides very local temporal information that is encoded via event stacking, the feature extractor takes a sequence of 3 event frames each mapped to a 512-dimensional feature vector that are concatenated and passed through a fully connected layer to reduce the dimension from 3×512 to 512. The second block consists of two linear layers with 512 neurons each, followed by $\tanh$ activations. It outputs the mean and log standard deviation of a Gaussian distribution over actions, which are used to select the best action.
Critic-NN: The critic receives the full environment state represented by a 9-dimensional vector s_t, and the chosen action $a(t)$. It is designed using a straightforward architecture based on fully connected layers. A flatten layer first transforms the input into a 1-dimensional vector. The main part consists of three dense layers: the first maps the input to 512 neurons, followed by a second hidden layer with 512 neurons. The final output layer produces a single scalar value, representing the estimated action value $Q^{\pi}(s(t), a(t))$. A $\tanh$ activation function is applied at the output to constrain the value range.

4 EXPERIMENTS

This section describes the experiments performed to train and test our method alongside the main findings. Further insights and secondary details are further provided via an on-line video.

Setup : A simulated environment was employed based on AirSim simulator [22] that supports aerial vehicles. AirSim provides flexibility in drone dynamics modeling by supporting both internal and external dynamic models. This enables to leverage the previously defined drone dynamic model while using the simulator for accurate sensor feedback based on the drone position and orientation.

Training and evaluation environments : We instantiate a training environment called *box* where the drone learns optimal tracking strategies under varying conditions. To favor transfer and generalization to diverse environments, we apply domain randomization [15], introducing variability in wall textures, colors, lighting, and orientation.

We evaluate the trained model in environments [3] inspired by the DARPA Subterranean Challenge[1], featuring complex settings such as caves, urban areas, tunnels, and mountains. These environments are challenging due to low lighting and high-texture scenes, making them ideal for evaluating our policy. We also assess generalization in various *box* environments. In both cases, we introduce changing lighting and fast-moving targets with randomly generated, multi-directional trajectories and random pauses. Example environments are shown in Fig. 4.

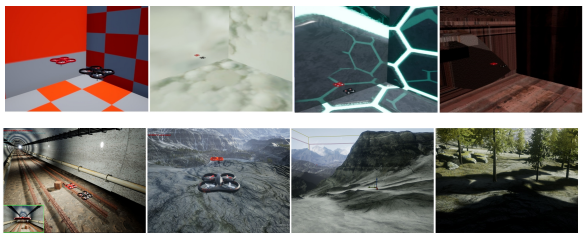

Figure 4: Examples of training *box* environments with varying textures (first row) and DARPA environments for evaluation (second row). The leader and follower drones are shown in red and black colors, respectively.

Comparisons and implementation parameters : The training was conducted on an NVIDIA RTX A4500 GPU with a 12-core i7 CPU and 32GB RAM, using parallel training with 7 agents for efficiency. We compare the end-to-end event-based UAV tracking with two settings: (1) a baseline method that trains the policy based on object detection, with observations comprising target positions and distances. A simple MLP processes the flattened input through three fully connected layers with ReLU activations. (2) an RGB-based

[1] https://github.com/osrf/subt/tree/master/subt_ign/worlds

baseline [6] that uses RGB images keeping all other architectural details unchanged.

All settings use 70 epochs of 50,000 timesteps each, with evaluation episodes set to 6. We use a buffer size of 10,000, batch size of 64, training every 8 timesteps, a learning rate of 0.0003, and a discount factor of 0.99.

In the reinforcement learning environment, each episode lasts 40 seconds, with an optimal tracking distance $d^* = 0.2$ meters. The action space includes angular rates $[-3.5, 3.5]$ rad/s and thrust $[-18, 18]$ N. Reward parameters are the penalty weight $\alpha = 0.4$ and the reward penalty constant $k_c = 10$. Events are stacked every $\Delta t = 0.005$s.

Training process :

To ensure diverse training episodes, target motion follows randomized sinusoidal trajectories with varying speeds, amplitudes, phases, and frequencies. Occasional random During training, the drone tracks the target for up to 40 seconds per episode, which ends upon collision, loss of visibility, or timeout, encouraging the target to stay within view and at an optimal distance. After each epoch, the policy is evaluated over 6 episodes using the mean and standard deviation of cumulative rewards. This process repeats for 70 epochs. The mean cumulative reward per episode curves of the training phase are shown in Fig.5.

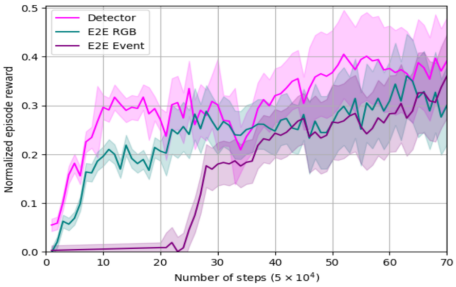

Figure 5: Learning plots of the detection-based (baseline), end-to-end RGB-based (E2E RGB) [6], and our proposed end-to-end event-based (E2E event) approach.

For the object detection-based approach, the agent learns faster thanks to the low-dimensional vector input and the simplicity of the policy architecture. The learning curve stabilizes around 0.35, achieving a maximum mean reward of 0.4050 ± 0.09. It is important to this policy is based on the ground truth pose of the tracked drone, thus setting an upper-bound on performance in scenarios where the drone is always perfectly detected. However, as shown in the rightmost part of the training plots, the rewards of both end-to-end approaches (RGB and event) are very close to it, with E2E event method performing slightly better at the end.

Limitations of object detection-based policy : To validate the end-to-end approach, we evaluate a policy trained on ground-truth object detections. While detection-based input can speed up learning, performance heavily depends on detection accuracy. Fig. 6 shows how added noise affects tracking by comparing target and tracker trajectories.

In this example, the target trajectory remains fixed while noise levels vary. As shown in the figure, tracking performance degrades with increased noise: at $\eta = 0.06$, performance drops moderately,

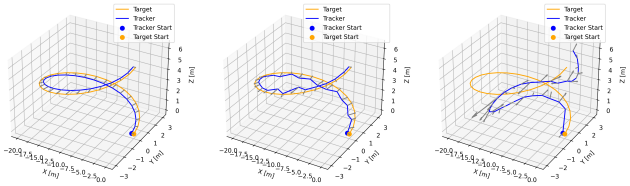

Figure 6: Example of the tracker and the target trajectories using object detector-based policy with different noise values: $\eta = 0$, $\eta = 0.06$, and $\eta = 0.12$.

while at $\eta = 0.12$, tracking fails. This demonstrates the sensitivity of detection-based methods to noise and highlights the benefit of end-to-end architectures that learn task-relevant features directly from raw inputs. The reward system encourages the agent to encounter scenarios relevant to the tracking task, allowing the jointly optimized feature extractor and policy network to focus on task-specific data and improve performance.

Comparison between RGB and event modalities : Both RGB and event-based pipelines use the ASAC algorithm with identical training conditions in the *box* environment. For fairness, both policies use the same architecture with identical parameter initialization and unchanged task prediction head. Table 1 presents the evaluation results of both approaches.

Scenarios	E2E Event	E2E RGB ([6])
Box environment (Low-light)	**0.45 ± 0.20**	0.35 ± 0.22
Box environment (Fast target)	**0.47 ± 0.19**	0.34 ± 0.21
Box environment (Normal)	0.50 ± 0.10	**0.52 ± 0.09**
DARPA environment (Low-Light)	**0.28 ± 0.17**	0.19 ± 0.16
DARPA environment (Fast target)	**0.26 ± 0.12**	0.22 ± 0.08
DARPA environment (Normal)	0.28 ± 0.11	**0.48 ± 0.09**

Table 1: Cumulative reward comparison of event and RGB policies in different scenarios (best performances are shown in bold).

As shown in Table 1, event-based tracking achieves the highest performance in low-light conditions due to their high dynamic range, outperforming underexposed RGB images. In contrast, surface reflections in textured scenes cause pixel-level brightness changes, leading to noisy event data, where RGB cameras perform better. Additionally, in fast-target scenarios, RGB cameras with low frame rate can cause the target to move out of the frame leading to poor tracking performance. In contrast, event cameras capture high-frequency changes, enabling the drone to react to rapid movements resulting in superior performance for fast-moving targets.

5 CONCLUSION

We addressed the problem of active drone tracking sing event streams and ASAC-based, deep reinforcement learning, highlighting the benefits of event cameras in terms of reactivity and robustness under varying conditions. A multi-modal solution that leverages the advantages of both RGB and event modalities could be a subject of further investigation, for example via hierarchical RL. However, such extension could require additional attention so as not to compromise real-time tracking performance.

6 ACKNOWLEDGMENT

This work was financed in the context of Labex CominLabs excellence laboratory managed by the National Research Agency in the "Investing for the Future" program under reference ANR-10-LABX-07-01, project *LEASARD* (Low-Energy deep neural networks for Autonomous Search-And-Rescue Drones).

REFERENCES

[1] Rubina Akter, Van-Sang Doan, Thien Huynh-The, and Dong-Seong Kim. 2021. RFDOA-Net: An Efficient ConvNet for RF-Based DOA Estimation in UAV Surveillance Systems. *IEEE Transactions on Vehicular Technology* 70, 11 (2021), 12209–12214.

[2] Antonio Albanese, Vincenzo Sciancalepore, and Xavier Costa-Pérez. 2022. SARDO: An Automated Search-and-Rescue Drone-Based Solution for Victims Localization. *IEEE Transactions on Mobile Computing* 21, 9 (2022), 3312–3325.

[3] DARPA. 2020. DARPA Subterranean Challenge. https://www.subtchallenge.com/

[4] Alessandro Devo, Alberto Dionigi, and Gabriele Costante. 2021. Enhancing continuous control of mobile robots for end-to-end visual active tracking. *Robotics and Autonomous Systems* 142 (2021).

[5] Alberto Dionigi, Alessandro Devo, Leonardo Guiducci, and Gabriele Costante. 2022. E-vat: An asymmetric end-to-end approach to visual active exploration and tracking. *IEEE Robotics and Automation Letters* 7, 2 (2022), 4259–4266.

[6] Alberto Dionigi, Simone Felicioni, Mirko Leomanni, and Gabriele Costante. 2024. D-VAT: End-to-end visual active tracking for micro aerial vehicles. *IEEE Robotics and Automation Letters* (2024).

[7] Alberto Dionigi, Mirko Leomanni, Alessandro Saviolo, Giuseppe Loianno, and Gabriele Costante. 2023. Exploring deep reinforcement learning for robust target tracking using micro aerial vehicles. In *International Conference on Advanced Robotics (ICAR)*.

[8] Zixuan Fang and Andrey V Savkin. 2024. Strategies for Optimized UAV Surveillance in Various Tasks and Scenarios: A Review. *Drones* 8, 5 (2024), 193.

[9] Hajer Fradi, Lorenzo Bracco, Flavia Canino, and Jean-Luc Dugelay. 2018. Autonomous person detection and tracking framework using unmanned aerial vehicles (UAVs). In *European Signal Processing Conference (EUSIPCO)*. 1047–1051.

[10] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. 2020. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence* 44, 1 (2020), 154–180.

[11] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. 2021. Dsec: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters* 6, 3 (2021), 4947–4954.

[12] Yassine Habib, Panagiotis Papadakis, Cédric Le Barz, Antoine Fagette, Tiago Gonçalves, and Cédric Buche. 2024. Real-time, dense UAV mapping by leveraging monocular depth prediction with monocular-inertial SLAM. *Advanced Robotics* 38, 21 (2024), 1555–1566.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385

[14] Larkin Heintzman, Amanda Hashimoto, Nicole Abaid, and Ryan K Williams. 2021. Anticipatory planning and dynamic lost person models for human-robot search and rescue. In *IEEE International Conference on Robotics and Automation (ICRA)*.

[15] Dániel Horváth, Gábor Erdős, Zoltán Istenes, Tomáš Horváth, and Sándor Földi. 2022. Object detection using sim2real domain randomization for robotic applications. *IEEE Transactions on Robotics* 39, 2 (2022), 1225–1243.

[16] Geert-Jan M. Kruijff, Fiora Pirri, Mario Gianni, Panagiotis Papadakis, Matia Pizzoli, Arnab Sinha, Viatcheslav Tretyakov, Thorsten Linder, Emanuele Pianese, Salvatore Corrao, Fabrizio Priori, Sergio Febrini, and Sandro Angeletti. 2012. Rescue robots at earthquake-hit Mirandola, Italy: A field report. In *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*.

[17] Michael L. Littman. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings*. 157–163.

[18] Bodi Ma, Zhenbao Liu, Wen Zhao, Jinbiao Yuan, Hao Long, Xiao Wang, and Zhirong Yuan. 2023. Target tracking control of UAV through deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems* 24, 6 (2023), 5983–6000.

[19] Luís Martins, Carlos Cardeira, and Paulo Oliveira. 2019. Linear Quadratic Regulator for Trajectory Tracking of a Quadrotor. In *IFAC Symposium on Automatic Control in Aerospace ACA*.

[20] Mark W. Mueller, Markus Hehn, and Raffaello D'Andrea. 2015. A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation. *IEEE Transactions on Robotics* 31, 6 (2015), 1294–1310.

[21] Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. 2017. Asymmetric actor critic for image-based robot learning. *arXiv preprint arXiv:1710.06542* (2017).

[22] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. 2018. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *International Conference Field and Service Robotics*. 621–635.

[23] Hongfei Wang and Yongkang Shi. 2022. Design of UAV target tracking controller based on visual servo. *Journal of Physics: Conference Series* 2246, 1 (2022), 012051.

[24] Mao Xi, Yun Zhou, Zheng Chen, Wengang Zhou, and Houqiang Li. 2021. Anti-distractor active object tracking in 3D environments. *IEEE Transactions on Circuits and Systems for Video Technology* 32, 6 (2021), 3697–3707.

[25] Lijuan Xu, Qinghai Yang, Meng Qin, Weihua Wu, and KyungSup Kwak. 2023. Collaborative Human Recognition with Lightweight Models in Drone-based Search and Rescue Operations. *IEEE Transactions on Vehicular Technology* (2023).

[26] Jiang Zhao, Han Liu, Jiaming Sun, Kun Wu, Zhihao Cai, Yan Ma, and Yingxun Wang. 2022. Deep Reinforcement Learning-Based End-to-End Control for UAV Dynamic Target Tracking. *Biomimetics* 7, 4 (2022), 197.