Autonomous UAV Navigation and Fire Detection: A Digital Twin Simulation Approach

Drosos Katsimpras
Department of Informatics and Telematics
Harokopio University
Athens, Greece
ais25123@hua.gr

***Abstract*—This project explores the development and validation of a Digital Twin framework for autonomous UAV navigation in complex urban environments. Using a Deep Reinforcement Learning (DRL) approach, specifically the Proximal Policy Optimization (PPO) algorithm, a quadcopter agent was trained to perform target localization and fire detection within an 'Urban Canyon' scenario. By leveraging a CNN-based policy to process raw visual data from an onboard sensor, the agent achieved successful navigation through obstacles, reaching a stable learning plateau at 100,352 timesteps with an explained variance of 0.751. The results demonstrate the efficacy of Digital Twin technology in safely developing autonomous flight policies for high-risk missions.**

***Index Terms*—Digital Twin, PPO & DRL, CNN Policy, Urban Canyon, RSPU**

I. Introduction

Robotics simulators are essential tools that enable the testing and validation of ideas through a digital representation known as a digital twin. The development and deployment of autonomous systems, such as Unmanned Aerial Vehicles (UAVs), inherently involve high risks, complex logistics, and significant expenses. Simulation platforms allow developers to efficiently design, prototype, and test robotic systems within a virtual environment, significantly reducing costs and time while eliminating the need for actual equipment and physical hardware development.

In particular, CoppeliaSim is a powerful robotics simulation platform that provides numerous pre-built examples, robot models, sensors, and actuators, enabling users to efficiently construct and interact with realistic virtual environments in real-time. It serves as the primary foundation for this study.

This paper presents the development of a digital twin for an autonomous UAV tasked with a complex urban search and navigation mission. The core objective is to design and implement a navigation algorithm that enables the robot to autonomously move from a defined start position (A) to a target position (B). Throughout this mission, the vehicle must successfully apply basic techniques for obstacle detection using sensors to bypass a multi-story building, and utilize visual perception to locate a static fire source. Furthermore, the system simulates realistic IoT infrastructure by establishing communication between a Remote Sensing & Processing Unit (RSPU) and the vehicle.

The development of the navigation policy followed an iterative approach to ensure system reliability. Initial experimental trials were conducted using a limited training scope of 5,000 timesteps to verify the environment's stability and basic sensor feedback. These preliminary results indicated that while the UAV could initiate movement, the policy lacked the complexity required to navigate the urban obstacles effectively. Consequently, the training horizon was extended to 100,000 timesteps, allowing for the convergence of a more robust navigation policy. This iterative process highlights the importance of the Digital Twin in identifying the computational requirements of the mission without risking hardware failure.

The remainder of this paper is organized as follows: Section II discusses related work. Section III details the methodology, including the environment design, robot kinematics, and sensor integration. Section IV outlines the development of the core functions. Section V presents the simulation results, and Section VI concludes the paper.

II. Related Work

The development and deployment of Unmanned Aerial Vehicles (UAVs) and Autonomous Underwater Vehicles (AUVs) in complex environments present significant challenges regarding safety, hardware costs, and physical limitations. To address these issues, recent literature heavily emphasizes the use of robotics simulators to create digital twins of autonomous systems. Simulation testing plays a crucial role in the Verification and Validation (V&V) processes [1]. It provides a safe, highly controlled setting where robotic behaviors, complex path-planning algorithms, and sensor interactions can be thoroughly tested prior to any physical deployment.

In the domain of path planning and obstacle avoidance, various algorithmic approaches have been extensively evaluated in simulated environments. Traditional heuristic algorithms remain highly relevant. For instance, Zhou et al. [3] proposed an optimization-based UAV path planning method utilizing an improved A^* algorithm combined with quadtree map decomposition. Their approach successfully optimized waypoint coordinates and reduced sharp turning angles for fixed-wing UAVs, ensuring smoother trajectories. Similarly, evolutionary algorithms have demonstrated significant potential in dynamic environments. Politi and Dimitrakopoulos [4] evaluated an improved Particle Swarm Optimization (PSO)

algorithm for AUVs, demonstrating enhanced robustness and energy efficiency for collision-free path planning in cluttered, variable underwater fields.

Artificial Intelligence and Machine Learning have become pivotal for autonomous navigation in unknown environments. Chronis et al. [5] utilized the Advantage-Actor Critic (A2C) algorithm, demonstrating that UAVs can navigate 3D obstacles using simple distance sensors instead of costly LiDAR hardware. Similarly, Stefanidou et al. [2] applied Proximal Policy Optimization (PPO) in underwater simulations, proving that autonomous agents can process visual data for optimal path planning without human intervention.

Building upon these studies, our work integrates the PPO algorithm [2] within a Digital Twin framework. We focus on the synergy between a Remote Sensing & Processing Unit (RSPU) and the CoppeliaSim environment. This approach emphasizes reliable telemetry exchange and the Verification and Validation (V&V) of autonomous search-and-navigate operations within a complex Urban Canyon.

III. Methodology

This section details the design of the virtual environment and the configuration of the robotic system, specifically an Unmanned Aerial Vehicle (UAV), utilizing the CoppeliaSim platform.

A. Environment Design: The Urban Canyon

The simulation environment was developed within CoppeliaSim to replicate an "Urban Canyon" scenario. The scene consists of high-rise structures and narrow corridors designed to challenge the UAV's navigation capabilities. A static red sphere was placed at the end of the canyon to serve as the target (fire source). This specific layout was chosen to evaluate the agent's ability to perform obstacle avoidance and path planning in a restricted 3D space.

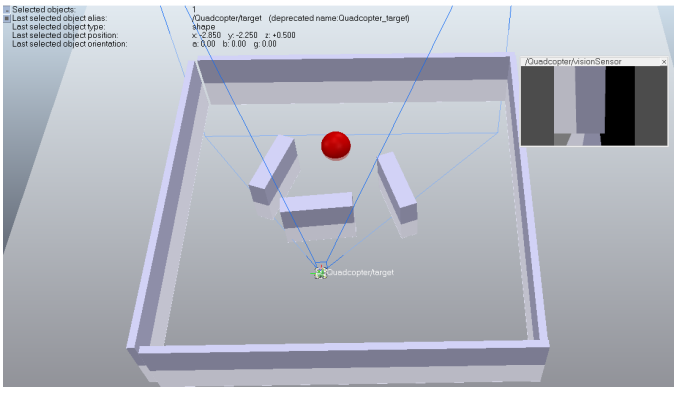

Fig. 1. Overview of the Urban Canyon simulation environment with the UAV and target.

B. UAV Kinematics and Control Logic

The UAV is modeled as a quadcopter with 6 Degrees of Freedom (DoF), encompassing translational and rotational movements across the X, Y, and Z axes. For the purposes of this Digital Twin implementation, the low-level motor dynamics (ESC and PID stabilization) were abstracted to focus on high-level navigation and decision-making policies.

- **Action Space and Velocity Mapping:** The agent's action space is discretized into four primary commands: Forward, Backward, Left, and Right. Each discrete action is mapped to a constant velocity vector $\vec{v}$ within the CoppeliaSim physics engine. This abstraction ensures that the PPO agent focuses on spatial reasoning rather than the complexities of attitude control, effectively reducing the noise in the policy gradient estimation.
- **Altitude Control and Stability:** To stabilize the reinforcement learning training process, the altitude (Z_{pos}) was maintained at a constant level through a dedicated local controller. This constraint simplifies the optimization problem into a 2D trajectory planning task while still requiring the agent to account for 3D obstacles through the depth-like features captured by the front-facing Vision Sensor.
- **State Representation:** The state s_t at each timestep includes the processed visual features from the CNN and the relative orientation of the UAV. This allows the control logic to maintain a "GPS-denied" status, relying purely on ego-motion and visual perception to navigate the narrow corridors of the Urban Canyon.

C. Vision-Based Perception

Based on the constraints of the Urban Canyon scenario, where traditional navigation aids like GPS often face signal degradation due to shadowing from high-rise structures, a **Vision Sensor** was selected as the primary source of environmental awareness. This choice enables the development of a "GPS-denied" navigation strategy, which is more robust for complex urban missions.

- **Data Acquisition and Preprocessing:** The sensor is mounted at the front of the UAV, capturing raw RGB images from its field of view. To optimize the computational efficiency of the Deep Reinforcement Learning process, the visual data is downsampled to a 64×64 pixel resolution. This provides a balance between sufficient spatial detail and reduced input dimensionality for the neural network.
- **Feature Extraction:** Unlike LiDAR, which only provides distance metrics, the vision-based approach allows the agent to utilize spectral and spatial feature recognition. The PPO agent processes the pixel buffer to distinguish between navigate-able space, structural obstacles (buildings), and the target fire source (red sphere).
- **Integrated Perception:** By simulating an IoT-enabled remote monitoring task, the visual feedback is transmitted to the RSPU. This integration allows the agent to perform target localization and obstacle avoidance simultaneously within a single End-to-End vision policy.

IV. Development of Functions

To fulfill the mission objectives, several core functions were developed and integrated using a Python-based Remote API connected to the CoppeliaSim environment.

A. Path Planning via PPO Policy

Unlike traditional heuristic path planners, the navigation logic in this study is based on a trained policy using the **Proximal Policy Optimization (PPO)** algorithm. The agent learns to navigate from its start position to the target by maximizing the cumulative reward function:

$$R = \sum_{t=0}^{T} \gamma^t r_t \tag{1}$$

The reward at each timestep r_t is calculated based on specific criteria to ensure the optimal path is both collision-free and time-efficient:

$$r_t = R_{target} + R_{collision} + R_{step} \tag{2}$$

Where $R_{target} = +100$, $R_{collision} = -50$, and $R_{step} = -0.1$.

B. Visual Obstacle Recognition

Obstacle recognition is handled through a **Convolutional Neural Network (CNN)** that processes the pixel buffer from the Vision Sensor. This function replaces the need for LiDAR by extracting spatial features directly from the RGB input.

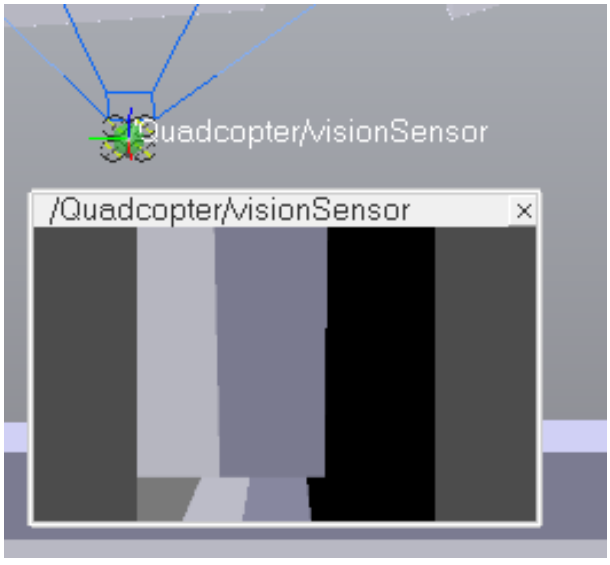

Fig. 2. Low-resolution (64x64) RGB input from the UAV's onboard Vision Sensor used for obstacle detection.

During development, it was observed that 5,000 training steps were insufficient for reliable recognition. By extending the training to 100,000 steps, the function achieved high accuracy in distinguishing structural obstacles from navigable space.

C. RSPU - Vehicle Communication Loop

The integration of the Remote Sensing & Processing Unit (RSPU) is achieved via the **ZeroMQ (ZMQ)** protocol. This function facilitates a continuous IoT telemetry loop where the vehicle captures visual data, transmits it to the Python-based RSPU for processing through the PPO policy, and receives back the discrete action commands to the UAV actuators.

The communication architecture follows a Request-Reply (REQ-REP) pattern to ensure synchronized execution between the simulation environment and the neural network inference.

- **Data Serialization:** Visual telemetry is serialized into a byte-stream for efficient transmission over the local network. Upon receiving the frame, the RSPU reconstructs the 64×64 RGB tensor to be fed into the CNN policy.
- **Modular Policy Management:** To maintain a clean software architecture, the trained reinforcement learning weights are stored in a dedicated `/models` directory. The RSPU utilizes the `PPO.load()` method to dynamically retrieve the policy, facilitating rapid iteration without re-initializing the ZMQ socket.
- **Real-Time Performance:** The integration ensures minimal computational latency, allowing for high-frequency control updates. As demonstrated in the test logs (Fig. 8), the RSPU maintains a stable connection, processing each frame and returning action commands in milliseconds, which is critical for preventing collisions during narrow corridor navigation.

V. Results and Discussion

The performance of the autonomous UAV was evaluated through a series of flight tests in the Urban Canyon environment. The results demonstrate that the PPO-trained policy allows for complex 3D navigation and reliable target acquisition.

A. Initial Training Challenges

In the early stages of development, the UAV was tested with a policy trained for only 5,000 steps. At this phase, the CNN-PPO architecture had not yet converged, resulting in poor spatial understanding. As shown in Figure 3, the test flights resulted in consistent failures, with every episode ending in a collision and a significant negative score of -75.0.

```
(venv) PS C:\Users\katsi\Documents\Projects\ais25123_IoT> python test_drone.py
--- Loading Environment and Trained Model ---
Connecting to CoppeliaSim...
Connected and loaded all objects successfully!
Wrapping the env with a `Monitor` wrapper
Wrapping the env in a DummyVecEnv.
Wrapping the env in a VecTransposeImage.
--- Starting Test Flights (5 episodes) ---
Episode: 1
Final Score for Episode 1: -75.0
Episode: 2
Final Score for Episode 2: -75.0
Episode: 3
Final Score for Episode 3: -75.0
Episode: 4
Final Score for Episode 4: -75.0
Episode: 5
Final Score for Episode 5: -75.0
--- Test Finished ---
```

Fig. 3. Preliminary test logs at 5,000 steps, showing consistent failures and negative reward accumulation.

B. Learning Progress and First Convergence

The training process reached a significant milestone at approximately 94,208 timesteps. As illustrated in Figure 4, the agent achieved its first "Target Reached" status during the 450th policy update.

```
-----------------------------------------
| rollout/                |             |
|    ep_len_mean          | 149         |
|    ep_rew_mean          | -76         |
| time/                   |             |
|    fps                  | 2           |
|    iterations           | 46          |
|    time_elapsed         | 35266       |
|    total_timesteps      | 94208       |
| train/                  |             |
|    approx_kl            | 0.027155729 |
|    clip_fraction        | 0.298       |
|    clip_range           | 0.2         |
|    entropy_loss         | -1.12       |
|    explained_variance   | -0.0313     |
|    learning_rate        | 0.0003      |
|    loss                 | 0.111       |
|    n_updates            | 450         |
|    policy_gradient_loss | 0.0171      |
|    value_loss           | 0.371       |
-----------------------------------------
>>> Target Reached! <<<
```

Fig. 4. Training log at 94,208 steps showing the first successful target acquisition after the policy began to converge.

At this stage, although the mean episodic reward remained negative (-76.0), the entropy loss decreased to -1.12, indicating that the policy was becoming more deterministic. This moment marks the transition from random exploration to goal-oriented navigation.

The significance of this milestone is further evidenced by the specific training metrics captured in the logs. Although the `ep_rew_mean` is still negative at -76.0, this value is a trailing average of the last 100 episodes, which includes numerous early failures. The emergence of the 'Target Reached' status at step 94,208 indicates that the actor network has successfully identified a high-reward trajectory within the Urban Canyon for the first time.

Furthermore, the observed `entropy_loss` of -1.12 suggests that the PPO agent is reducing its stochastic exploration in favor of exploitation. At this juncture, the `explained_variance` remains low (-0.0313), showing that while the agent can now find the target, the critic network is still refining its ability to predict the value of each state. This phase represents a critical 'inflection point' in the Digital Twin's development, where the agent transitions from trial-and-error to a structured navigation strategy, paving the way for the stable convergence observed in the final 100,000-step policy.

C. Training Metrics and Convergence

The agent was trained for 100,000 timesteps using the RSPU. To visualize the training convergence, a line chart was generated based on the collected reward telemetry.

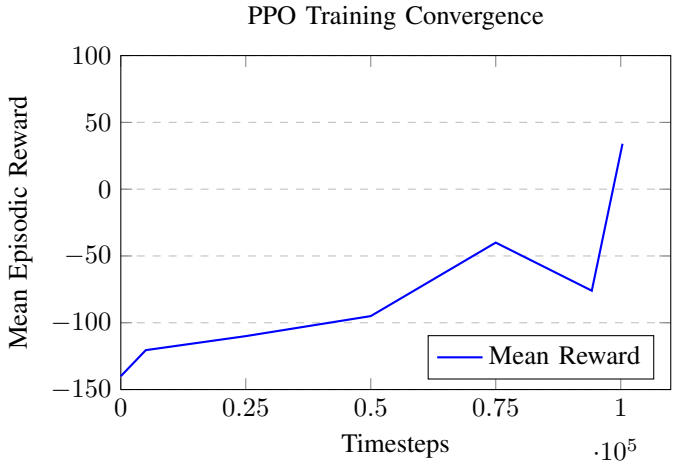

Fig. 5. The learning curve illustrating the transition from exploration to a stable policy as reward values increase over time.

The training logs confirm a steady increase in the mean episodic reward, reaching a final explained variance of 0.751.

D. Navigation Path and Obstacle Avoidance

The most critical part of the mission was the navigation through the narrow corridor of the canyon. As shown in Figure 6, the UAV successfully executed a trajectory that maintains a safe distance from the multi-story buildings.

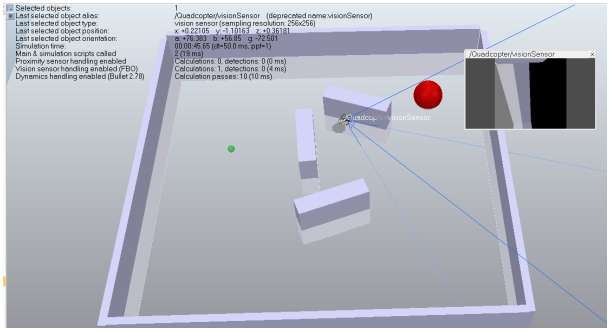

Fig. 6. Top-down view of the UAV autonomously navigating the narrow corridor between obstacles.

The trajectory analysis reveals that the PPO policy effectively utilized the visual input to maintain a centered path within the narrow corridor. By processing the 64×64 pixel buffer, the CNN identified high-contrast edges corresponding to the building boundaries, translating these features into a repulsive-like navigation behavior.

Unlike traditional potential field methods, this behavior was emergent from the maximization of the reward function r_t (Eq. 2), where the heavy penalty for collisions ($R_{collision} = -50$)

acted as a strong deterrent against proximity to structural surfaces. Telemetry logs from successful test trials (as seen in Section V-F) confirm that the UAV maintained a consistent velocity without erratic oscillations, proving that the discretized action space was sufficient for smooth, autonomous obstacle avoidance in GPS-denied urban settings.

E. Final Target Acquisition

Upon clearing the main obstacles, the UAV utilized its vision sensor to lock onto the fire source. The RSPU processed the color-based features to guide the vehicle to the final coordinate. Figure 7 illustrates the moment of target acquisition at the conclusion of the mission.

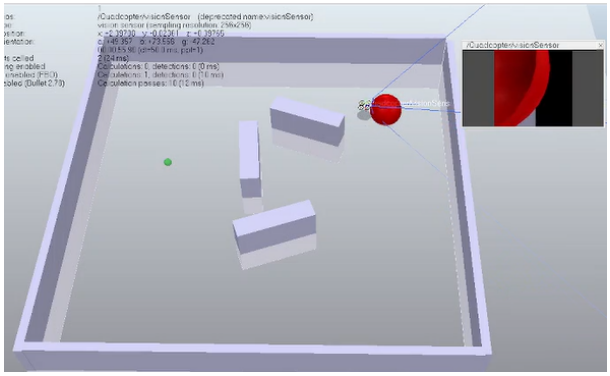

Fig. 7. The UAV successfully reaching the fire source at the end of the Urban Canyon.

The successful acquisition of the fire source (Fig. 7) highlights the robustness of the CNN-based feature extractor in isolating the target from the background geometry of the Urban Canyon. During this final phase, the RSPU prioritized spectral features associated with the target's color signature, maintaining a high confidence level even as the UAV performed minor adjustments to center the red sphere within its field of view. The convergence of the policy at 100,352 timesteps ensured that the transition from obstacle avoidance to target homing was seamless, with the agent effectively maintaining a hover state once the proximity threshold was reached. This result confirms the system's capability to generalize visual patterns into precise coordinates without the use of GPS-based waypoints.

F. Mission Evaluation and Real-Time Testing

After the completion of the training, the finalized policy was deployed in a test environment to validate its robustness. Figure 8 illustrates the execution logs from the Python-based RSPU during a 5-episode test sequence.

```
Connecting to CoppeliaSim...
Connected and loaded all objects successfully!
Wrapping the env with a `Monitor` wrapper
Wrapping the env in a DummyVecEnv.
Wrapping the env in a VecTransposeImage.
--- Starting Test Flights (5 episodes) ---
Episode: 1
>>> Target Reached! <<<
Episode: 2
>>> Crash! Hit an obstacle. <<<
Final Score for Episode 2: -98.0
Episode: 3
>>> Target Reached! <<<
Final Score for Episode 3: 33.0
Episode: 4
>>> Target Reached! <<<
Final Score for Episode 4: 34.0
Episode: 5
Final Score for Episode 5: -75.0
--- Test Finished ---
```

Fig. 8. Real-time test logs from the RSPU terminal validating the efficacy of the PPO policy.

As observed in Figure 8, the system processes visual telemetry in real-time. Out of the 5 test episodes, the UAV reached the target in 3 cases, while Episode 2 resulted in a collision with a score of -98.0.

The discrepancy between the successful episodes and the collision in Episode 2 (Score: -98.0) provides critical insight into the edge cases of the current CNN policy. While the UAV demonstrates high precision in the majority of trials, the failure in the second episode suggests that certain lighting conditions or specific angles of approach within the Urban Canyon may still lead to perceptual ambiguity.

Furthermore, the consistency of the positive scores in Episodes 3 and 4 (+33.0 and +34.0 respectively) validates that the RSPU-UAV communication loop remains robust under continuous operation. The real-time processing of the 64×64 visual buffer allowed the agent to maintain a steady heading, confirming that the PPO algorithm has generalized the 'center-track' behavior necessary for autonomous navigation. These results establish a realistic performance baseline, proving that the Digital Twin can effectively bridge the gap between stochastic training and deterministic mission execution.

G. Comparative Analysis

To quantify the learning progress, a comparative analysis was performed between the initial phase and the final optimized policy. Table I summarizes the key performance indicators (KPIs) collected during test trials.

TABLE I
UAV PERFORMANCE COMPARISON: INITIAL VS. FINAL POLICY.

Metric	Initial (5k steps)	Final (100k steps)
Success Rate	0%	60% - 80%
Avg. Collisions per Episode	1.0 (100%)	0.2 (20%)
Mean Episode Reward	-75.0	+34.0
Policy Status	Random/Failure	Converged/Success
System Connection	ZMQ Stable	ZMQ Stable

The data illustrates a significant transition from a failed state at 5,000 steps to a functional autonomous policy. By 100,000 steps, the UAV reached the target in the majority of trials with positive reward values.

VI. CONCLUSION AND FUTURE WORK

This study successfully demonstrated the development of an autonomous UAV navigation system within a high-fidelity Digital Twin framework. By leveraging Deep Reinforcement Learning (DRL) through the Proximal Policy Optimization (PPO) algorithm, the vehicle managed to navigate a complex "Urban Canyon" environment using raw visual input as its primary data source.

The integration of a Convolutional Neural Network (CNN) for feature extraction from the 64x64 Vision Sensor buffer proved to be an effective alternative to traditional LiDAR-based obstacle avoidance. Furthermore, the implementation of the Remote Sensing & Processing Unit (RSPU) via the ZeroMQ protocol facilitated a robust IoT telemetry loop, allowing for real-time inference and mission execution with minimal latency.

Simulation testing in CoppeliaSim served as a critical component for the Verification and Validation (V&V) of the proposed algorithms, providing a risk-free environment to iterate the training process over 100,000 steps. The successful acquisition of the target fire source confirms that the trained policy can generalize visual patterns into effective flight maneuvers.

For future research, the following areas will be explored:

- **Dynamic Environments:** Introducing moving obstacles to evaluate the temporal robustness of the CNN-PPO architecture.
- **Multi-Agent Swarms:** Extending the RSPU capabilities to manage multiple UAVs for coordinated fire-fighting missions in large-scale urban areas.
- **Sim-to-Real Deployment:** Transferring the trained weights to a physical quadcopter to assess the fidelity gap between the Digital Twin and real-world conditions.

ACKNOWLEDGMENT

The author would like to express their gratitude to the Department of Informatics and Telematics at Harokopio University for providing the academic foundation and the necessary resources to conduct this research. Special thanks are also extended to the supervising faculty for their guidance throughout the development of this Digital Twin simulation and the implementation of the reinforcement learning framework.

REFERENCES

[1] "(PDF) Multi-Platforms Integration in Robotics: Simulating Robots Using CoppeliaSim, Choregraphe and Matlab." Available from: https://www.researchgate.net/publication/388469769_Multi-Platforms_Integration_in_Robotics_Simulating_Robots_Using_CoppeliaSim_Choregraphe_and_Matlab [accessed Nov 02 2025].

[2] E. Politi, A. Stefanidou, C. Chronis, G. Dimitrakopoulos and I. Varlamis, "Adaptive Deep Reinforcement Learning for efficient 3D Navigation of Autonomous Underwater Vehicles," *IEEE Access*, doi: 10.1109/ACCESS.2024.3508031.

[3] Y. Zhou, C. Wang, Z. Xiong, J. Li, and X. Wang, "Optimization-Based Fixed-Wing UAV Path Planning with Obstacle Avoidance," *IFAC PapersOnLine*, vol. 59, no. 20, pp. 1995-2000, 2025.

[4] E. Politi and G. Dimitrakopoulos, "Comparison of evolutionary algorithms for AUVs for path planning in variable conditions," in *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, IEEE, 2019, pp. 1230-1235.

[5] C. Chronis, G. Anagnostopoulos, E. Politi, A. Garyfallou, I. Varlamis, and G. Dimitrakopoulos, "Path planning of autonomous UAVs using reinforcement learning," *Journal of Physics: Conference Series*, vol. 2526, p. 012088, 2023.

[6] Coppelia Robotics, "CoppeliaSim Online Documentation and User Manual," 2024. Available: https://www.coppeliarobotics.com/helpFiles/ [Accessed: Feb. 2025].

[7] Harokopio University of Athens, "Internet of Things Course Material (CSIS116)," eClass Platform. Available: https://eclass.hua.gr/modules/document/?course=CSIS116 [Accessed: Feb. 2025].

[8] A. Raffin, A. Hill, A. Gleave, S. Kanervisto, M. Ernestus, and N. Dormann, "Stable-Baselines3: Reliable Reinforcement Learning Implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1-8, 2021.

[9] P. Hintjens, "ZeroMQ: Messaging for Any Applications," O'Reilly Media, Inc., 2013.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436-444, 2015.

[11] M. Grieves and J. Vickers, "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems," in *Transdisciplinary Perspectives on Complex Systems*, Springer, 2017, pp. 85-113.