# Toward Real-World Implementation of Deep Reinforcement Learning for Vision-Based Autonomous Drone Navigation with Mission

Mozhgan Navardi†, Prakhar Dixit†, Tejaswini Manjunath †, Nicholas R. Waytowich‡, Tinoosh Mohsenin†, Tim Oates†
†Department of Computer Science & Electrical Engineering, University of Maryland Baltimore County
‡US Army Research Laboratory

***Abstract*—Though high fidelity simulators for drones and other aerial vehicles look exceptionally realistic, using simulators to train control policies and then transferring them to the real world does not work well. One reason is that real images, especially on low-power drones, produce output that look different from simulated images, ignoring for the moment that simulated worlds themselves look rather different from real ones at the level that matters for machine learning. To overcome this limitation, we focus on using object detectors that tend to transfer well from simulation to the real world, and extract features of detected objects to serve as input to reinforcement learning algorithms. Empirical results with a low-power drone show promising results. A recorded video which compares a vision-based approach with the proposed approach can be found here: Video**

***Index Terms*—Reinforcement learning, Real-world implementation, Object detection, Drone navigation.**

## I. Introduction and Related Work

Autonomous drone navigation is one of the emerging technologies that enable many capabilities such as search and rescue [1]. Moreover, indoor drones used for finding specific objects can take too long if it is done manually. Machine Learning (ML) algorithms, specifically Reinforcement Learning (RL) can be used to train an agent for goal-oriented navigation tasks. The trained model with RL can work on low-power devices such as tiny Unmanned Aerial Vehicles (UAVs) for search and rescue [2]. To achieve this goal, the RL agent needs to learn to navigate towards some predefined objects. In this paper, we propose a framework to solve the challenge of transferring the learned model from simulation to the real world in a goal oriented environment.

There are some related works which reduce the gap between simulation and real-world implementation for drone navigation and obstacle detection. [5] is a survey on object detection using ML algorithms. Kang *et al.* [2] has combined real and simulation images to help transferring sim-to-real. [6] have proposed CAD2RL for indoor flight collision avoidance. The main contribution of our work is a framework for drone navigation and obstacle avoidance using image processing and RL by (1) designing same real world and simulation environments, (2) combining simulation and real world data and (3) image processing. To the best of our knowledge there is no related work of sim-to-real which defines a mission like "reach the sphere" with collision avoidance for indoor drones.

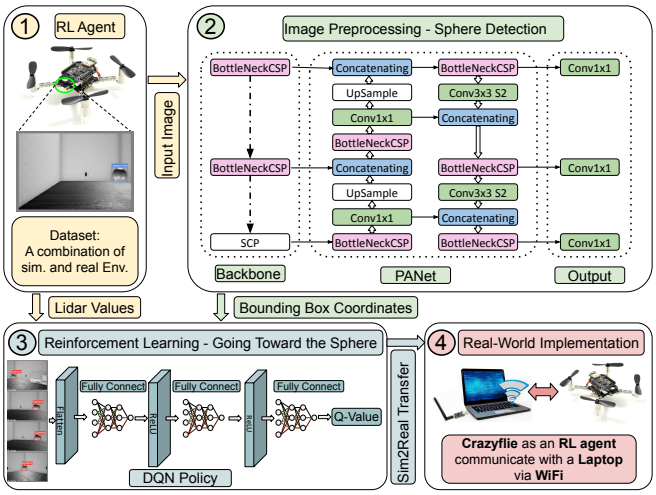


Fig. 1. Proposed end-to-end framework for real-world implementation of deep reinforcement learning including four steps: (1) Simulate an environment in Airsim [3] similar to a real environment and collect a combination dataset of real and simulated environment by using RL agent which is a drone. (2) Design a custom model by inspiring YOLO [4] to detect sphere in the environment. (3) Using generated data in two last steps to train the agent in simulated environment with DRL to fulfill a mission like going toward sphere. (4) Transferring simulation to real-world implementation.

## II. Proposed Method

Figure 1 shows an overview of the proposed framework which includes four steps: collecting data, image preprocessing, reinforcement learning and real-world implementation.

**Design Environment and Collect Data.** In the first step, we design the simulation environment to mirror the real world. Figure 2 shows two simple and complex environments which are designed for training and testing the agent. Each environment has an object as a goal. We increase the complexity of the environment by adding some obstacles similar to the goal. The combined dataset which included 3000 simulation and 1500 real images are used for training the model. This dataset is annotated manually and labelled by creating bounding boxes around each class object and considered them as the ground truth.

**YOLOv5 and RL Integration.** In the second step, instead of giving captured images directly to the RL model, collected images are given to a customized YOLOv5 model [4] to do image processing. Finally, we use a vector of eight values

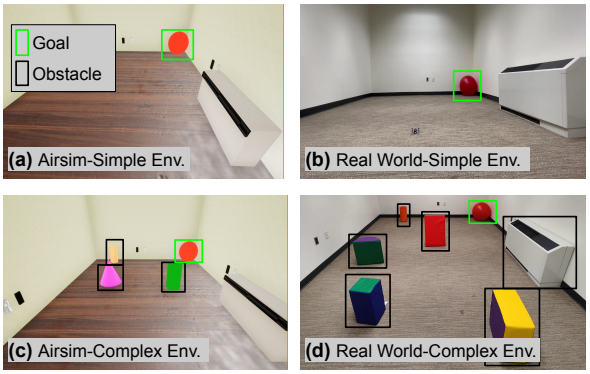


Fig. 2. Two different environments, simple and complex, are used to train and test the RL agent. (a) and (c) show simulated environments in Airsim which are similar to real world environments (b) and (d).

as the observation space to the RL algorithm. The first four values are the bounding box coordinates of the detected sphere and the last four values are the multi-ranger values used for obstacle avoidance.

**RL Reward Policy.** We developed an end-to-end navigation system based on DQN [7]. This model directly transfers the sensor readings and sphere dimensions to the commands of multi-rotor movements. In order to use RL for training a model, we define the reward as follows:

$$r_t = \begin{cases} r_{goal} & d_t \leq d_p \\ r_{col} & d_o \leq d_q \\ 1 - d_t \end{cases} \quad (1)$$

where $r_{\text{goal}}$ is the positive reward on reaching the target position, $r_{\text{col}}$ is negative reward when the multi-rotor collides with obstacles in the environment and $1-d_{\text{t}}$ in other situations. $d_{\text{t}}$ is the distance between the goal and the drone at time $t$, and $d_{\text{o}}$ is the distance from the obstacle calculated using lidar readings. $d_{\text{p}}$ and $d_{\text{q}}$ are predefined distances used to check if the target is reached or a collision is occurred.

**Simulation to Real World.** The difference between simulation and the real world means that the trained models based on simulation images are not applicable for real-world scenarios. To account for this difference, there are several approaches like Domain Adaptation [8], Domain Randomization [9] and Fine Tuning [10]. Although this generalization ability is critical for RL models, zero-shot policy transfer is still a challenging problem [11] For simpler tasks like object detection and navigation, using an object detection algorithm like YOLO seems more appropriate as the visual differences can be addressed early. For the object detection, we used the same approach as simulation in the real world by passing the images captured from the drone to the YOLO model to get the bounding box coordinates. We ensure the values in simulation and real world are consistent with one another for easy transfer from sim-to-real. Once we have a well performing model in simulation we transfer that on to a drone with AI capabilities [12].

## III. Experimental Result

To evaluate the proposed approach, we used CrazyFlie, an open-source flying development instrument with an AI-Deck extension board [13] which is shown in Figure 3. The defined mission for the CrazyFlie is reaching the sphere in the environment without collision. Figure 4 shows the result of testing the customized YOLO model in simulated and paired real world. Once we got satisfactory results with YOLOv5 training, we integrated YOLO model with our RL algorithm. We could achieve up to 92% accuracy with no penalty in simulated environment. In the simulation a vision-based approach reached 90% average success rate after 100k steps while we achieve same success rate in 400 steps. Therefore, we reduced training time significantly. In real-world testing, we observed similar results when the drone is sufficiently close to the object of interest when it can clearly see the sphere, otherwise it spends reasonable time in searching for the object which poses a problem as the CrazyFlie comes with limited power. These results are improved upon the vision-based navigation where the agent takes more time and reaches the goal in less than 20% of the cases. The real-world experiment video can be found here.

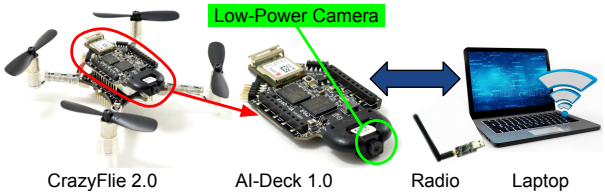


Fig. 3. Hardware setup for real-world implementation.

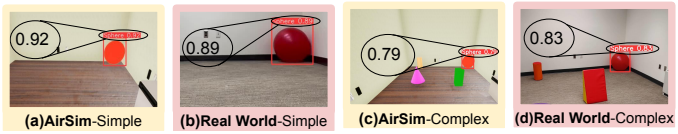


Fig. 4. Preprocessing on drone view provides sphere detection in simulation and real world environments.

## Conclusion

For end-to-end navigation tasks, the autonomous agent needs to handle a wide variety of challenges. Learning in the real world is difficult on account of unusual situations. Training in simulation is one solution to high costs in real-word testing. However, the ability to transfer learned information in the simulation to real does not translate to safe navigation in the real world. Further advancements in making simulation environment realistic is needed. Even then there are certain aspects that are left unaccounted for. To alleviate this, we implement a different approach where object detection features are combined with reinforcement learning for easier sim-to-real transfer. The main idea is to use object detection features instead of raw video. As evidenced in the results above, proposed approach outperforms vision-based navigation by a lot owing to the well performing underlying model.

## Acknowledgment

This project was sponsored by the U.S. Army Research Laboratory under Cooperative Agreement Number W911NF2120076.

## REFERENCES


[1] B. P. Duisterhof *et al.*, "Sniffy bug: A fully autonomous swarm of gas-seeking nano quadcopters in cluttered environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 9099–9106.

[2] K. Kang, S. Belkhale, G. Kahn, P. Abbeel, and S. Levine, "Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6008–6014.

[3] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," 2017. [Online]. Available: https://arxiv.org/abs/1705.05065

[4] G. Jocher *et al.*, "ultralytics/yolov5: v6.1 - TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference," Feb. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6222936

[5] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[6] F. Sadeghi and S. Levine, "Cad2rl: Real single-image flight without a single real image," *arXiv preprint arXiv:1611.04201*, 2016.

[7] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning." [Online]. Available: https://arxiv.org/abs/1312.5602

[8] S. Daftry, J. A. Bagnell, and M. Hebert, "Learning transferable policies for monocular reactive mav control," in *International Symposium on Experimental Robotics*. Springer, 2016, pp. 3–11.

[9] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," 2017. [Online]. Available: https://arxiv.org/abs/1703.06907

[10] T. Xie, N. Jiang, H. Wang, C. Xiong, and Y. Bai, "Policy finetuning: Bridging sample-efficient offline and online reinforcement learning," 2021. [Online]. Available: https://arxiv.org/abs/2106.04895

[11] D. Ghosh, J. Rahme, A. Kumar, A. Zhang, R. P. Adams, and S. Levine, "Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability," 2021. [Online]. Available: https://arxiv.org/abs/2107.06277

[12] W. Giernacki, M. Skwierczyński, W. Witwicki, P. Wroński, and P. Kozierski, "Crazyflie 2.0 quadrotor as a platform for research and education in robotics and control engineering," in *2017 22nd International Conference on Methods and Models in Automation and Robotics (MMAR)*, 2017, pp. 37–42.

[13] D. Palossi *et al.*, "A 64-mw dnn-based visual navigation engine for autonomous nano-drones," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8357–8371, 2019.