Bridging the Gap Between Simulations and Reality: A CycleGAN-Based Approach for Drone Landing Systems

1st Diyar Altinses
Department of Automation Technology and learning systems
South Westphalia University of Applied Sciences
Soest, Germany
altinses.diyar@fh-swf.de

2nd David Orlando Salazar Torres
Department of Automation Technology and learning systems
South Westphalia University of Applied Sciences
Soest, Germany
salazartorres.davidorlando@fh-swf.de

3th Andreas Schwung
Department of Automation Technology and learning systems
South Westphalia University of Applied Sciences
Soest, Germany
schwung.andreas@fh-swf.de

***Abstract*—Drone delivery systems are increasingly gaining importance in modern logistics due to their efficiency and potential to revolutionize the supply chain. However, among the various stages of drone operations, the landing phase stands out as the most critical, as it requires precise navigation and control to ensure the safe and successful delivery of goods. Therefore, testing simulations are required to accurately replicate the complex dynamics of real-world environments. In this paper, we propose a CycleGAN-based framework for bidirectional domain translation between simulated and real-world data, with a focus on GPS features. The proposed architecture facilitates seamless transfer learning by enabling simulation-trained controllers to operate effectively in real-world environments and vice versa. Additionally, we integrate the translated data into a downstream control framework for autonomous landing tasks, validating the practicality of the approach.**

***Index Terms*—Domain adaptation, precision drone landing, sim-to-real, drone control**

I. Introduction

Unmanned aerial vehicles (UAV) have gained widespread attention for their diverse applications, ranging from surveillance and aerial mapping to delivery systems and emergency response [1]. A critical phase in the operation of drones, particularly in environments where precision and safety are important, is the landing process [2]. Ensuring a smooth and precise landing is essential for minimizing the risk of damage to the drone, its payload, and the surrounding environment [3].

Achieving consistently accurate and smooth drone landings remains a significant challenge, despite considerable advancements in control systems. This difficulty stems from a combination of unpredictable environmental factors, such as wind and turbulence, inaccuracies inherent in onboard sensors, and the dynamic conditions present during flight [4] [5]. These challenges introduce considerable uncertainty, making precise and reliable landings a complex task that demands advanced control algorithms capable of adapting in real-time [6].

While real-world deployment of drone control systems is the ultimate goal, it is first necessary to test these systems extensively in simulated environments. Simulations serve as a safe and controlled space to model various flight conditions without the risk of hardware damage or safety breaches [7]. However, simulations often fall short of accurately representing the complexities of real-world environments, creating a substantial gap between the simulated and actual conditions [8]. As a result, controllers developed and optimized within these artificial environments frequently fail to perform effectively when exposed to the unpredictable dynamics of real-world scenarios [9]. Closing this gap is essential, as the inability to translate simulated performance into practical application undermines the reliability and utility of such control systems in real-world operations.

In this study, we propose a domain adaptation algorithm with additional regularization to bridge the gap between simulated data and real-world conditions. For this purpose, we employ a CycleGAN architecture that includes two generators and discriminators to learn a bidirectional mapping between simulated and real-world data. The generators aim to translate the simulated data into a realistic domain and vice versa, while the discriminators evaluate the authenticity of the generated samples, ensuring that the translated data closely resembles the target domain. We include a regularization loss to push the synthetic real data to be similar to the simulated data. To validate our approach, we collect real-world data using a PX4 flight controller in a quadrocopter. PX4 leverages onboard sensors to estimate the vehicle state, which is critical for stabilizing the vehicle and enabling autonomous control. With this approach, we can convert a simple simulation into a high-fidelity simulation. High-fidelity simulations incorporate essential factors such as aerodynamics, weather variations, and sensor noise, significantly reducing the discrepancy between simulated and real-world performance. By minimizing this gap, developers can ensure that the performance observed in simulated environments closely mirrors the behavior of drones in practical applications, thereby enabling a smoother and more reliable transition from simulation to real-world deployment. Our key contributions can be summarized as follows:

(i) We design a CycleGAN architecture to perform bidirectional translation between simulated and real-world data, with a focus on GPS-based features (longitude, latitude, and altitude).
(ii) We successfully incorporate the CycleGAN-generated data into a downstream control framework, demonstrating its utility in enabling robust landing control by bridging the gap between simulated and real-world domains.
(iii) Comprehensive evaluation of all CycleGAN components and domain translation samples validate the model's capacity to generate realistic and semantically consistent data for both domains.

The structure of this paper is as follows: First, we offer an in-depth review of the current literature, underscoring key studies. Next, we describe our methodology, focusing on the neural domain adaptation architecture, with a detailed explanation of the key development stages. We then present the experimental results, analyzing the findings and evaluating the control mechanisms to determine their effectiveness and practical relevance. Lastly, we summarize our contributions and propose future research directions.

II. Related Work

This section reviews the relevant literature on position estimation for drone landing techniques and sim-to-real domain adaptation methods and machine learning applications. The contributions of this paper are built upon these areas, with the goal of improving the precision and smoothness of drone landings in real-world scenarios.

A. Drone landing

The challenge of drone landing has been a prominent area of focus within UAV research, given the complexities associated with achieving precision and coping with environmental factors. PID controllers have traditionally been the go-to solution for managing descent and landing tasks. However, these controllers often face limitations when subjected to disturbances or inaccuracies in sensor readings.

Mahony et al. provided an insightful analysis of quadrotor dynamics, pointing out the drawbacks of conventional control techniques like PID in the context of landing in dynamic conditions [10]. To overcome these challenges, alternative strategies, such as adaptive controllers and neural networks, have been proposed. One such solution, introduced by Bouadi et al., is a sliding mode controller that adjusts to uncertainties during the landing phase, demonstrating enhanced resilience to external disturbances [3]. Additionally, recent research has seen the integration of machine learning-based algorithms to refine landing processes, leading to improvements in both stability and precision. Notably, Mellinger et al. explored the use of trajectory optimization techniques for aggressive maneuvers, including precise landing control, a study that continues to influence the development of autonomous drone systems [11].

The application of visual-inertial systems for drone landings has also gained attention in recent studies. For instance, Joo et al. combined camera and inertial data to improve landing accuracy, particularly for small target landings [12]. Furthermore, Zhang et al. introduced an adaptive neural network control framework aimed at achieving safe and smooth landings in complex, cluttered environments, further advancing the field of autonomous landing technologies [13]. Alongside control systems, sensor-based assistance has been explored as a means of enhancing landing precision. A comprehensive review by Noor et al. highlighted the critical role of integrating various sensors to improve the reliability and accuracy of autonomous landing systems [14].

Our approach differs by addressing the critical gap in sim-to-real transfer in drone landing applications. While simulations are essential for reducing costs and risks, their accuracy must closely mirror real-world conditions to be effective. To bridge this gap, we propose a domain adaptation method based on CycleGANs, which facilitates the conversion of simulated environments into realistic scenarios for testing real-world controllers while also transforming real-world data into simulated data to leverage controllers developed in simulation.

B. Domain adaptation

Domain adaptation techniques are crucial for bridging the gap between simulated and real-world data, particularly in control systems where models trained in simulation must perform reliably in real environments. Cycle-consistent generative Adversarial Networks (CycleGANs) have emerged as a prominent method for unpaired image-to-image translation, facilitating sim-to-real domain adaptation by enabling the transformation of simulated data to resemble real-world information and vice versa [15].

Rao et al. introduced RL-CycleGAN, a reinforcement learning-aware CycleGAN variant designed for sim-to-real transfer in vision-based robotic grasping tasks. By incorporating an RL-scene consistency loss, their model ensures that the translation process preserves task-relevant features, thereby enhancing policy transfer from simulation to reality [16]. Similarly, Yuan et al. proposed a sim-to-real learning framework for vision-based robotic assembly tasks, utilizing CycleGAN for domain adaptation alongside force control strategies. Their approach successfully transferred policies trained in simulation to real-world peg-in-hole assembly scenarios, demonstrating the effectiveness of combining visual domain adaptation with control theory principles [17].

Autonomous navigation has become a popular machine-learning application. In the context of LiDAR-based object detection, Barrera et al. developed a CycleGAN-based domain adaptation method to reduce the sim-to-real gap in bird's eye view representations. By enforcing cycle and semantic consistency, their approach improved detection performance for smaller road agents, highlighting the applicability of CycleGANs in diverse sensing modalities [18]. Xu et al. propose a novel Curriculum Domain Adaptation method that transfers smooth semantic knowledge from daytime to nighttime [19]. Yoo et al. propose a model to improve adversarial learning for domain adaptation. In addition, they perform a theoretical analysis that supports the empirical results of the method [20].

These studies underscore the potential of CycleGAN-based domain adaptation methods in enhancing the transferability of control policies from simulation to the real-world. Nevertheless, there is a gap in sim-to-real drone landing control methods that specifically address the challenges of varying environmental conditions, such as wind, lighting, and surface irregularities, which can significantly impact landing accuracy and system performance in real-world scenarios.

III. Domain adaptation for drone control

In this section, we begin by defining the problem related to sim-to-real domain adaptation, particularly in the context of UAV systems. Once the problem is defined, we introduce our framework for the sim-to-real translation and the simulated landing control. We start by providing an overview of our approach, outlining the key components and objectives. Following this, we detail the data collection process, describing how both simulated and real-world data are gathered and utilized to train and validate our models. Finally, we present the neural domain adaptation architecture we developed, which enables the generation of more realistic synthetic data by bridging the gap between simulated and real-world environments.

A. *Problem defintion*

Let $\{\mathbf{x}_i^{\text{sim}}\}_{i=1}^{N} \in \mathcal{X}_{\text{sim}}$ and $\{\mathbf{x}_j^{\text{real}}\}_{j=1}^{M} \in \mathcal{X}_{\text{real}}$ represent the domains of simulated and real-world GPS data, respectively. The goal is to learn three functions. First, the forward function $G : \mathcal{X}_{\text{sim}} \rightarrow \mathcal{X}_{\text{real}}$ such that $\mathbf{x}_i^{\text{real}} = G(\mathbf{x}_i^{\text{sim}})$ for all $\mathbf{x}_i^{\text{sim}} \in \mathcal{X}_{\text{sim}}$. Then the backward mapping $F : \mathcal{X}_{\text{real}} \rightarrow \mathcal{X}_{\text{sim}}$ such that $\mathbf{x}_j^{\text{sim}} = F(\mathbf{x}_j^{\text{real}})$ for all $\mathbf{x}_j^{\text{real}} \in \mathcal{X}_{\text{real}}$. Finally, the cycle consistency that enforces $F(G(\mathbf{x}_i^{\text{sim}})) \approx \mathbf{x}_i^{\text{sim}}$ and $G(F(\mathbf{x}_j^{\text{real}})) \approx \mathbf{x}_j^{\text{real}}$, ensuring consistency in both directions. These mappings allow for bidirectional data transformation between these domains while preserving task-specific features relevant to landing control.

B. *Approach overview*

The primary objective of our approach is to bridge the gap between simulated data and real-world behavior by converting simulated data to closely resemble real-world data. This transformation enables us to create a wide range of simulated scenarios, which can then be effectively translated into real-world situations. This domain adaptation is modelled by a CycleGAN architecture which we will introduce at the end of this section.

Once the simulated data is converted into a real format, it is used as input to the controller, which generates control actions. These actions are designed to closely mimic the responses of a real-world controller operating in an actual environment. Following this, the simulated position generated by the environment is again transformed into a real-world equivalent, creating a continuous loop where simulated inputs are transformed into real-world outputs and vice versa. This iterative process ensures that the system can accurately simulate realistic behaviors while using real-world-like data to train and validate control strategies. An overview of this entire process and framework is illustrated in Figure 1.

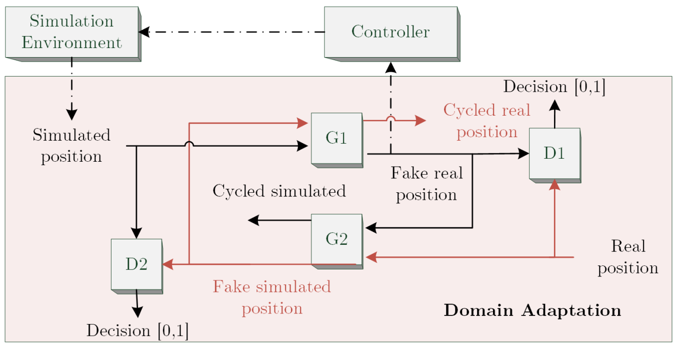


Fig. 1: The domain adaptation framework (red box) and the testing scenario (dashed lines).

With this setup, we can effectively simulate various real-world scenarios, including environmental disturbances, sensor inaccuracies, and other uncertainties that might occur in real-life operations. This allows us to rigorously test and refine the developed controller under conditions that closely mirror real-world environments, ensuring its robustness and effectiveness in practical applications. Additionally, this framework enables testing across a range of conditions that might be difficult or costly to reproduce in physical experiments, providing valuable insights into the controller's performance before deployment in real-world settings.

C. *Data collection*

In this section, we provide an overview of the collection process for both the simulated and real-world datasets. The simulated data was generated in a controlled virtual environment, allowing us to test various scenarios and flight conditions in a safe and reproducible manner. In contrast, the real-world data was collected using a quadcopter equipped with advanced sensors, capturing information from actual flight tests under real environmental conditions.

1) Simulation environment: To generate synthetic data for the drone landing process, we leverage the environment and control models presented in [21]. Specifically, we utilize their sensor topology and dynamics modeling framework, which includes detailed formulations for GPS sensor measurements. The simulated landing process is executed within a controlled environment where the drone dynamics include translational and rotational forces. These dynamics are based on the drone's state vector, which includes its position, velocity, orientation, and angular velocity. The control of the drone is handled by a Proportional-Derivative controller, where both translational and rotational controls are calculated based on error terms relative to target positions and orientations [21].

2) Simulated data collection: To generate clean data, we remove the uncertainties in position estimation from the sensor models. Using this simulation setup, we generate $n = 50$ landing scenarios, each with a simulation duration of $T = 2000\,\mathrm{ms}$. The dataset includes precise measurements of position, velocity, and sensor readings. Let the GPS-derived position be (ϕ, λ, h), where $\phi \in [-\pi/2, \pi/2]$ (latitude), $\lambda \in [-\pi, \pi]$ (longitude), and $h \in \mathbb{R}$ (altitude). The transformation to Cartesian coordinates $(x, y, z) \in \mathbb{R}^3$ is given by $x = R \cdot \cos(\phi) \cdot \cos(\lambda)$, $y = R \cdot \cos(\phi) \cdot \sin(\lambda)$, $z = h$, where $R = 6,371\,\mathrm{km}$ is Earth's radius. For each landing process, initial conditions $(\phi_0, \lambda_0, h_0, v_0)$ are randomly sampled from uniform distributions. The dataset records the trajectory as a time-series function $\mathbf{p}(t) = \begin{bmatrix} x(t) & y(t) & z(t) \end{bmatrix}^\top$, $\mathbf{v}(t) = \frac{d\mathbf{p}(t)}{dt}$, $\mathbf{a}(t) = \frac{d^2\mathbf{p}(t)}{dt^2}$. This dataset provides a high-dimensional representation of landing dynamics with minimized uncertainties.

3) Real-world data collection: For our data collection, we used an Auriol quadcopter (Figure 2) by Third Element Aviation, equipped with the PX4 flight controller as the central unit for real-time sensor management and flight commands. The dataset $D = \{X_1, X_2, \ldots, X_N\}$, where $X_i \in \mathbb{R}^{p \times q}$, contains sensor data matrices, with p as features and q as samples. A key subset is GPS data, capturing spatial coordinates: latitude $\phi \in [-90^\circ, 90^\circ]$, longitude $\lambda \in [-180^\circ, 180^\circ]$, and altitude $h \in \mathbb{R}$ (above WGS84 ellipsoid). Additionally, GPS-related features include timestamps, position accuracy, velocity, signal quality, heading, and fixed states. Data was logged across varying days/times for diverse real-world scenarios.

Fig. 2: The Auriol quadrocopter from Third Element Aviation used for the real-world data collection.

D. *CycleGAN for domain adaptation*

This section presents the methodology employed for developing a CycleGAN-based model for domain adaptation tasks. The proposed architecture leverages multilayer perceptrons as the building blocks for generators and discriminators, enabling bidirectional domain transformations while maintaining consistency and task-specific features.

The CycleGAN model consists of two generators G and F and two discriminators D_{sim} and D_{real}, designed to perform bidirectional mapping between two domains $\mathcal{X}_{\text{sim}}$ and $\mathcal{X}_{\text{real}}$. The generator G is configured to map inputs from domain $\mathcal{X}_{\text{sim}}$ to domain $\mathcal{X}_{\text{real}}$. The generator F is configured to map inputs from domain $\mathcal{X}_{\text{real}}$ to domain $\mathcal{X}_{\text{sim}}$. The discriminators D_{sim} and D_{real} are responsible for distinguishing real inputs from fake inputs generated by G and F, respectively. Each generator and discriminator is implemented using a multilayer perception architecture. The discriminators include an additional sigmoid activation module at their output to provide probability estimates.

The training objective of the CycleGAN is formulated as a combination of adversarial $\mathcal{L}_{\text{GAN}}$, cycle-consistency, and identity losses:

$$\min_{G,F} \mathcal{L}_{\text{total}} = \mathcal{L}_{\text{GAN}} + \lambda_{\text{cycle}} \mathcal{L}_{\text{cycle}} + \lambda_{\text{idt}} \mathcal{L}_{\text{idt}}, \tag{1}$$

where λ_{cycle} and λ_{idt} are hyperparameters controlling the relative importance of cycle consistency and identity losses.

The adversarial loss $\mathcal{L}_{\text{GAN}}$ is the sum of $\mathcal{L}_{\text{GAN},1}$ and $\mathcal{L}_{\text{GAN},2}$ and ensures that the generators produce outputs indistinguishable from real samples by their respective discriminators:

$$\mathcal{L}_{\text{GAN},1} = \mathbb{E}_{\mathbf{x}_{\text{real}} \sim \mathcal{X}_{\text{real}}} [\log D_{\text{real}}(\mathbf{x}_{\text{real}})] \tag{2}$$
$$+ \mathbb{E}_{\mathbf{x}_{\text{sim}} \sim \mathcal{X}_{\text{sim}}} [\log(1 - D_{\text{real}}(G(\mathbf{x}_{\text{sim}})))], \tag{3}$$

where D_{real} is the discriminator for the real domain. Similarly, for the simulated domain:

$$\mathcal{L}_{\text{GAN},2} = \mathbb{E}_{\mathbf{x}_{\text{sim}} \sim \mathcal{X}_{\text{sim}}} [\log D_{\text{sim}}(\mathbf{x}_{\text{sim}})] \tag{4}$$
$$+ \mathbb{E}_{\mathbf{x}_{\text{real}} \sim \mathcal{X}_{\text{real}}} [\log(1 - D_{\text{sim}}(F(\mathbf{x}_{\text{real}})))]. \tag{5}$$

To preserve the semantic structure of the input data, the cycle-consistency loss enforces that the mappings $G_{\text{sim}} \circ G_{\text{real}}$ and $G_{\text{real}} \circ G_{\text{sim}}$ approximate identity transformations. The cycle consistency loss is defined as:

$$\mathcal{L}_{\text{cyc}} = \mathbb{E}_{\mathbf{x}_{\text{sim}} \sim \mathcal{X}_{\text{sim}}} [\|F(G(\mathbf{x}_{\text{sim}})) - \mathbf{x}_{\text{sim}}\|_1] \tag{6}$$
$$+ \mathbb{E}_{\mathbf{x}_{\text{real}} \sim \mathcal{X}_{\text{real}}} [\|G(F(\mathbf{x}_{\text{real}})) - \mathbf{x}_{\text{real}}\|_1]. \tag{7}$$

The identity loss promotes the preservation of features during mapping by penalizing deviations from the input:

$$\mathcal{L}_{\text{idt}}(G, F) = \mathbb{E}_{\mathbf{x}_{\text{real}} \sim \mathcal{X}_{\text{real}}} [\|G(\mathbf{x}_{\text{real}}) - \mathbf{x}_{\text{real}}\|_1] \tag{8}$$
$$+ \mathbb{E}_{\mathbf{x}_{\text{sim}} \sim \mathcal{X}_{\text{sim}}} [\|F(\mathbf{x}_{\text{sim}}) - \mathbf{x}_{\text{sim}}\|_1]. \tag{9}$$

This methodology leverages a CycleGAN framework with MLP-based generators and discriminators to achieve effective domain adaptation. The combination of adversarial, cycle-consistency, and identity losses ensures the mappings are robust and preserve task-specific features.

To ensure the transformed data preserves features critical for the landing control task, we define a task-specific loss:

$$\mathcal{L}_{\text{task}} = \mathbb{E}_{\mathbf{x}_{\text{real}} \sim p_{\text{real}}} [\mathcal{T}(G(\mathbf{x}_{\text{real}}), \mathbf{x}_{\text{real}})] \tag{10}$$
$$+ \mathbb{E}_{\mathbf{x}_{\text{sim}} \sim p_{\text{sim}}} [\mathcal{T}(F(\mathbf{x}_{\text{sim}}), \mathbf{x}_{\text{sim}})], \tag{11}$$

where $\mathcal{T}(\cdot, \cdot)$ is a task-specific objective function related to landing control and is defined as the mean squared error. The primary objective is to ensure that the translated data preserves amplitude and other crucial signal characteristics while exclusively modifying the inter-domain disturbances.

IV. Evaluation

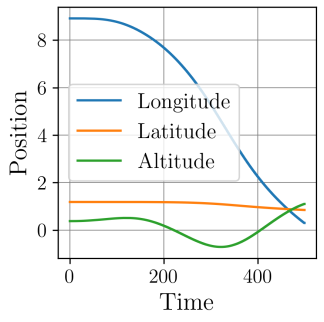
In this section, we evaluate the proposed framework to highlight its key features. We begin by describing the dataset pre-processing and the experimental setup used for the optimization process. Following this, we assess the performance of the model during both training and testing. To conclude, we showcase the model's characteristics by integrating it into the controller framework and comparing the resulting outcomes.

A. Dataset

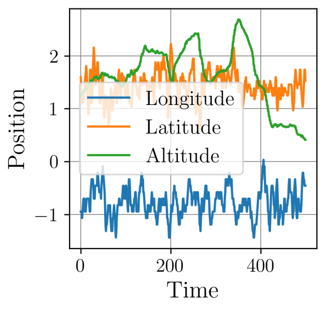
For data generation, we create $K = 500$ synthetic landing scenarios, each containing $N = 2000$ samples for the three axes: longitude, latitude, and altitude. These samples are structured in the form $\{\mathbf{p}_{\text{gps},i}\}_{i=1}^{N}$, where $\mathbf{p}_{\text{gps},i}$ represents the corresponding simulated position. Similarly, we extract the same sensor information of the real dataset.

A sliding window approach is applied to the simulated, as well as, to the real dataset. Each sliding window has a size $w_s = 500$ and a step size $w_j = 500$, which determine how the data is segmented. The total number of samples after applying the sliding window is defined as $L = \left(\left\lfloor \frac{N-w_s}{w_j} \right\rfloor + 1\right) \cdot K = 2000$. This results in a reformulated dataset structured as $\{\mathbf{P}_{\text{gps},i}\}_{i=1}^{L}$, where each input sequence for a window is represented as $\mathbf{P}_i \in \mathbb{R}^{w_s \times 3}$.

Since we have more simulated samples than the real ones, we expand the real dataset to the simulated length. Therefore, we create a set of uniformly sampled indices and construct the extended dataset by selecting elements from the real dataset corresponding to the indices. This ensures that the real dataset is expanded to the size of the simulated dataset by resampling its elements according to randomly generated indices. Then, all K datasets are concatenated to form a comprehensive dataset that covers the entire domain of interest. Finally, the datasets are normalized based on the z-norm and presented in Figure 3.



(a) Simulated data sample.



(b) Real data sample.

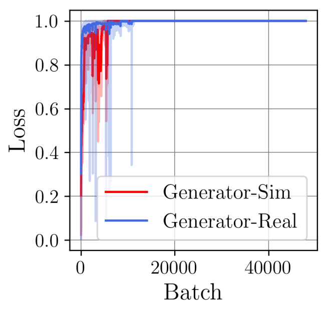
Fig. 3: A sample of the drone's time series GPS position data is shown, with figure a) representing the real-world dataset and figure b) the simulated.

B. Experimental Setup

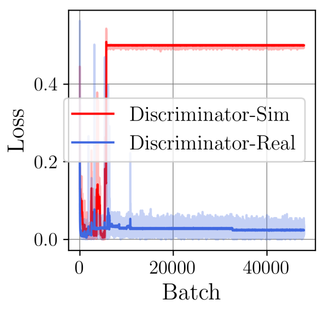
The model is trained for a total of $4 \cdot 10^4$ epochs with a batch size of 128, leveraging the ADAM optimizer to facilitate efficient updates throughout the training process. The initial learning rate, $\lambda = 10^{-4}$, is progressively reduced every 10^4 epochs by a factor of 10^{-1}. The generator networks consist of 10 hidden layers, each with 1024 neurons, and have an input and output size of 1500. The discriminators follow a layered architecture with a configuration of 1500-512-32-1, where 1500 neurons form the input layer, leading to three hidden layers, and ending with a single output neuron. ReLU activation functions are applied between all layers to introduce non-linearities into the system.

C. Results

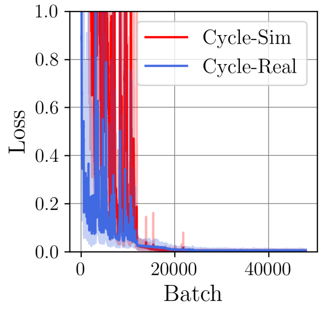
The evaluation of the CycleGAN architecture focuses on assessing its ability to achieve effective domain adaptation while maintaining semantic consistency. In Figure 4, we present the test losses of the model, including generator, discriminator, cycle consistency, and identity losses, as key metrics to evaluate the performance and stability of the adversarial training process. These results provide insights into the effectiveness of the architecture in learning bidirectional mappings between the domains.



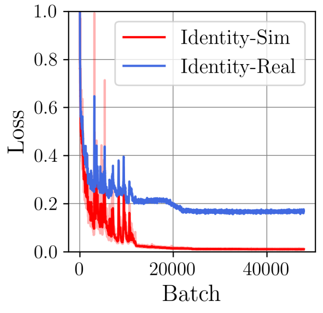
(a) Generator loss.



(b) Discriminator loss.



(c) Cycle consistency loss.

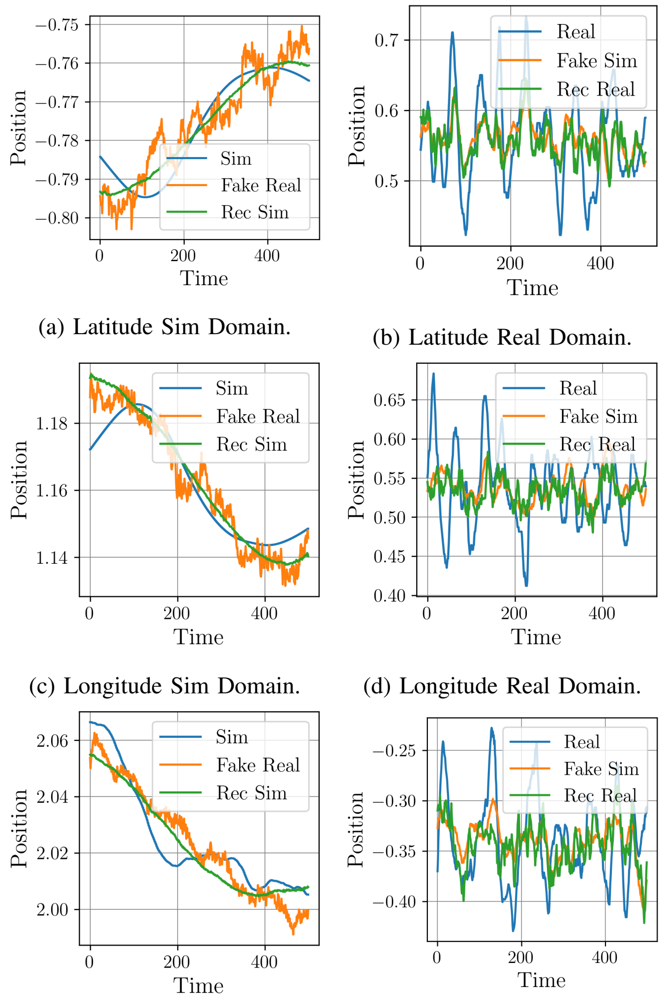


(d) Identity loss.

Fig. 4: The four primary components of the loss function employed in a CycleGAN: (a) Generator loss, which encourages the generator to produce realistic images, (b) Discriminator loss, which aims to distinguish real images from generated ones, (c) Cycle consistency loss, which ensures that images can be translated back and forth between domains without significant information loss, and (d) Identity loss, which preserves identity mappings between domains.

Early fluctuations indicate the adversarial game between the generators and discriminators before achieving equilibrium. Stable generator losses indicate that the generators have learned to produce realistic outputs for both domains. Cycle losses decrease during the training phase and stabilize near zero, indicating that the generators effectively reconstruct the original inputs after domain translation. Cycled-Sim and Cycled-Real exhibit symmetric behavior, demonstrating balanced performance between the two domains. The convergence to near-zero cycle losses is a strong indicator of the model's ability to maintain semantic consistency during translation and reconstruction.

To further evaluate the effectiveness of the proposed CycleGAN architecture, we analyze its performance in translating samples between the two domains in Figure 5. By leveraging the bidirectional nature of the model, we generate domain-specific translations for given inputs and visually inspect their quality and consistency. The translations from the source domain to the target domain and their subsequent reconstructions highlight the model's ability to preserve semantic information while adapting the style and features of the target domain.



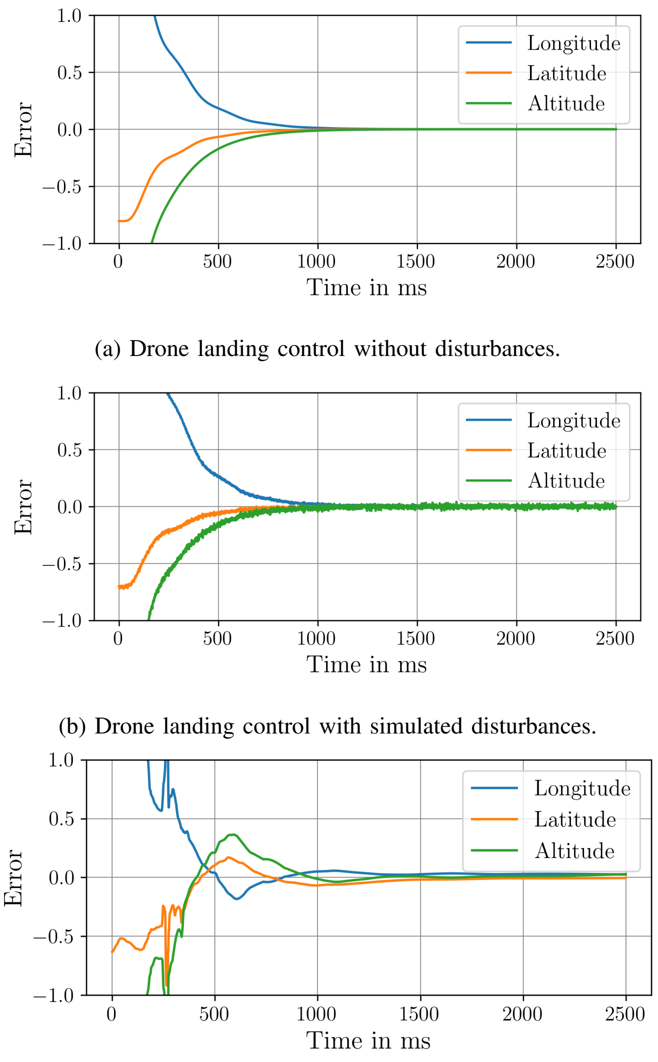
(a) Latitude Sim Domain. (b) Latitude Real Domain.

(c) Longitude Sim Domain. (d) Longitude Real Domain.

(e) Altitude Sim Domain. (f) Altitude Real Domain.

Fig. 5: Domain-specific translations of samples from both domains using the CycleGAN architecture.

These results provide qualitative evidence of the model's capacity to perform accurate and meaningful domain adaptation. In addition to evaluating CycleGAN's performance on domain translation, we integrate the architecture into our downstream control framework to demonstrate its practical utility. By incorporating CycleGAN as a core component in our control framework, we aim to showcase the practicability of a simulation-tuned controller in real-world domains. The results of this integration are shown in Figure 6 and provide insights into the feasibility and reliability of using domain-adapted data for robust control in complex, real-world environments.



(a) Drone landing control without disturbances.

(b) Drone landing control with simulated disturbances.

(c) Drone landing control using domain adaptation.

Fig. 6: This figure illustrates the drone landing control systems under different scenarios: (a) ideal conditions with no external disturbances, (b) showcasing the robustness of the control system in the presence of simulated disturbances based on the paper [21], and (c) demonstrating the effectiveness of domain adaptation techniques in generating real-world variations.

The first scenario in Figure 6 represents a clean, idealized simulation without any disturbances. In this case, the controller converges very quickly and operates without noise. The second scenario includes simulated sensor disturbances based on the methodology described in [21] and has similar behavior including noise. The final scenario involves domain adaptation, where the input data is processed by Generator A (simulated-to-real transformation) before being used in the control architecture to mimic real behavior. The quantitative comparison is presented in Table I. With these results, we can estimate whether the controller works in the real world or not.

TABLE I: Comparison of three scenarios based on classical control quality criteria.

Criterion	**Clean**	**Sim. noisy**	**Trans. real**
Settling time (ms)	631,0	747,0	836,0
Rise time (ms)	631,0	747,0	483,0
Overshoot (%)	0	0	39,40
Steady-state error (%)	0	0	0,061

V. Conclusion

In conclusion, drone delivery systems hold immense potential to revolutionize modern logistics, with the landing phase being the most critical. Accurate testing methodologies are essential, as traditional simulations often fail to capture the complexities of real-world environments. To address this challenge, we proposed a CycleGAN-based framework for bidirectional domain translation between simulated and real-world data, focusing on GPS features. We include a task-specific penalty function to force the model to keep the input signal characteristics and only transfer disturbances and sensor uncertainties. This architecture enables transfer learning, allowing simulation-trained controllers to perform effectively in real-world scenarios and vice versa. Furthermore, integrating the translated data into a control framework for autonomous landing tasks demonstrated the practicality and effectiveness of simulation-based controllers in real-world scenarios.

Future research could explore expanding the framework to incorporate additional sensory data, such as visual and LiDAR information, to enhance the fidelity of the domain translation process further. Real-world testing at larger scales and in diverse operational environments will also be crucial for validating and refining the proposed approach.

Funding

The research project was funded by "The Ministry of the Environment, Nature Conservation and Transport of the State of North Rhine-Westphalia" in Germany and co-financed by the European Union for the research project "SIDDA - Sustainable Intermodal Drone Delivery Airline" with grant number IN-ML-1-013b.

References

[1] D. Altinses, D. O. Salazar Torres, M. Schwung, S. Lier, and A. Schwung, "Optimizing drone logistics: A scoring algorithm for enhanced decision making across diverse domains in drone airlines," *Drones*, vol. 8, no. 7, p. 307, 2024.

[2] Y. Feng, C. Zhang, S. Baek, S. Rawashdeh, and A. Mohammadi, "Autonomous landing of a uav on a moving platform using model predictive control," *Drones*, vol. 2, no. 4, p. 34, 2018.

[3] H. Bouadi, M. Bouchoucha, and M. Tadjine, "Sliding mode control based on backstepping approach for an uav type-quadrotor," *International Journal of Mechanical and Mechatronics Engineering*, vol. 1, no. 2, pp. 39–44, 2007.

[4] D. Altinses, D. O. S. Torres, A. M. Gobachew, S. Lier, and A. Schwung, "Synthetic dataset generation for optimizing multimodal drone delivery systems," *Drones*, vol. 8, no. 12, p. 724, 2024.

[5] D. Altinses and A. Schwung, "Deep multimodal fusion with corrupted spatio-temporal data using fuzzy regularization," in *IECON 2023-49th Annual Conference of the IEEE IES*. IEEE, 2023, pp. 1–7.

[6] D. O. S. Torres, D. Altinses, and A. Schwung, "Data imputation techniques using the bag of functions: Addressing variable input lengths and missing data in time series decomposition," in *2025 IEEE International Conference on Industrial Technology (ICIT)*. IEEE, 2025, pp. 1–7.

[7] V. Kangunde, R. S. Jamisola Jr, and E. K. Theophilus, "A review on drones controlled in real-time," *International journal of dynamics and control*, vol. 9, no. 4, pp. 1832–1846, 2021.

[8] D. Altinses and A. Schwung, "Multimodal synthetic dataset balancing: a framework for realistic and balanced training data generation in industrial settings," in *IECON 2023-49th*. IEEE, 2023, pp. 1–7.

[9] A. Mairaj, A. I. Baba, and A. Y. Javaid, "Application specific drone simulators: Recent advances and challenges," *Simulation Modelling Practice and Theory*, vol. 94, pp. 100–117, 2019.

[10] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *IEEE robotics & automation magazine*, vol. 19, no. 3, pp. 20–32, 2012.

[11] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

[12] S. Joo, C. Ippolito, K. Al-Ali, and Y.-H. Yeh, "Vision aided inertial navigation with measurement delay for fixed-wing unmanned aerial vehicle landing," in *2008 IEEE aerospace conference*. IEEE, 2008, pp. 1–9.

[13] H.-T. Zhang, B.-B. Hu, Z. Xu, Z. Cai, B. Liu, X. Wang, T. Geng, S. Zhong, and J. Zhao, "Visual navigation and landing control of an unmanned aerial vehicle on a moving autonomous surface vehicle via adaptive learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 12, pp. 5345–5355, 2021.

[14] M. Noor, M. Ismail, M. Khyasudeen, A. Shariffuddin, N. Kamel, and S. R. Azzuhri, "Autonomous precision landing for commercial uav: A review," *Fuzzy Systems and Data Mining III*, pp. 459–468, 2017.

[15] X. Jin, Y. Park, D. Maddix, H. Wang, and Y. Wang, "Domain adaptation for time series forecasting via attention sharing," in *International Conference on Machine Learning*. PMLR, 2022, pp. 10 280–10 297.

[16] K. Rao, C. Harris, A. Irpan, S. Levine, J. Ibarz, and M. Khansari, "Rl-cyclegan: Reinforcement learning aware simulation-to-real," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 157–11 166.

[17] C. Yuan, Y. Shi, Q. Feng, C. Chang, M. Liu, Z. Chen, A. C. Knoll, and J. Zhang, "Sim-to-real transfer of robotic assembly with visual inputs using cyclegan and force control," in *2022 IEEE International Conference on Robotics and Biomimetics*. IEEE, 2022, pp. 1426–1432.

[18] A. Barrera, J. Beltrán, C. Guindel, J. A. Iglesias, and F. García, "Cycle and semantic consistent adversarial domain adaptation for reducing simulation-to-real domain shift in lidar bird's eye view," in *2021 IEEE International Intelligent Transportation Systems Conference*. IEEE, 2021, pp. 3081–3086.

[19] Q. Xu, Y. Ma, J. Wu, C. Long, and X. Huang, "Cdada: A curriculum domain adaptation for nighttime semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2962–2971.

[20] J. Yoo, Y. Hong, Y. Noh, and S. Yoon, "Domain adaptation using adversarial learning for autonomous navigation," *arXiv preprint arXiv:1712.03742*, 2017.

[21] D. Altinses, D. O. S. Torres, S. Lier, and A. Schwung, "Neural data fusion enhanced pd control for precision drone landing in synthetic environments," in *2025 IEEE International Conference on Mechatronics (ICM)*. IEEE, 2025, pp. 1–7.