TU Delft

Delft University of Technology

Document Version
Final published version

Licence
Dutch Copyright Act (Article 25fa)

Citation (APA)
Yu, H., Wagter, C. D., & de Croon, G. C. H. E. (2025). Depth Transfer: Learning to See Like a Simulator for Real-World Drone Navigation. *IEEE Robotics and Automation Letters*, *10*(11), 11848-11855. https://doi.org/10.1109/LRA.2025.3617729

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
In case the licence states “Dutch Copyright Act (Article 25fa)”, this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

Green Open Access added to TU Delft Institutional Repository
as part of the Taverne amendment.

More information about this copyright law amendment
can be found at https://www.openaccess.nl.

Otherwise as indicated in the copyright section:
the publisher is the copyright holder of this work and the
author uses the Dutch legislation to make this work public.

Depth Transfer: Learning to See Like a Simulator for Real-World Drone Navigation

Hang Yu, Christophe De Wagter, and Guido C. H. E de Croon, *Member, IEEE*

***Abstract*—Sim-to-real transfer is a fundamental challenge in robot learning. Discrepancies between simulation and reality can significantly impair policy performance, especially if it receives high-dimensional inputs such as dense depth estimates from vision. We propose a novel depth transfer method based on domain adaptation to bridge the visual gap between simulated and real-world depth data. A Variational Autoencoder (VAE) is first trained to encode ground-truth depth images from simulation into a latent space, which serves as input to a reinforcement learning (RL) policy. During deployment, the encoder is refined to align stereo depth images with this latent space, enabling direct policy transfer without fine-tuning. We apply our method to the task of autonomous drone navigation through cluttered environments. Experiments in IsaacGym show that our method nearly doubles the obstacle avoidance success rate when switching from ground-truth to stereo depth input. Furthermore, we demonstrate successful transfer to the photo-realistic simulator AvoidBench using only IsaacGym-generated stereo data, achieving superior performance compared to state-of-the-art baselines. Real-world evaluations in both indoor and outdoor environments confirm the effectiveness of our approach, enabling robust and generalizable depth-based navigation across diverse domains.**

***Index Terms*—Domain adaptation, representation learning, reinforcement learning.**

I. Introduction

VISION-BASED navigation enables drones to autonomously and safely navigate through complex environments. To reduce the computational burden on onboard systems, reinforcement learning has increasingly been adopted as a key approach for autonomous obstacle avoidance [1], [2]. However, real-world training is costly and risky, motivating the use of simulators for safe and efficient policy learning [3], [4], [5].

Among visual modalities, depth maps are particularly favored for sim-to-real transfer, as they offer geometric cues and exhibit better generalization than raw RGB images. Many prior works have successfully trained obstacle avoidance policies using depth maps in simulation and deployed them in challenging real-world environments such as forests [1], [6]. However, despite this advantage, a significant perception gap remains between the ideal, noise-free depth maps generated in simulation and the noisy, artifact-laden depth maps captured in the real world. These discrepancies continue to hinder the effectiveness of reinforcement learning when deploying policies trained in simulation to real-world environments.

To address this, we propose a depth transfer method that aligns stereo depth images with simulated ground truth through representation learning and domain adaptation. As shown in Fig. 1(a), we train an encoder to extract latent features from simulated depth images, which are then combined with drone state and navigation target information to train an RL policy.

We evaluated the policy in AvoidBench [7] using an encoder refined with IsaacGym-generated stereo depth. Without using AvoidBench data during training, our method outperforms Ego-Planner [8] and Agile-Autonomy [9] in all speed settings. It surpasses MAVRL [1] in high-speed scenarios, which was trained in the AvoidBench simulator. For real-world deployment, the encoder is further refined using real stereo depth data, enabling sim-to-real transfer. Figures 1(b,c) show the evaluation environment and trajectories.

Our main contributions are as follows.

- We propose a depth transfer method that aligns stereo depth images with simulator ground truth, enabling RL policies trained entirely in the IsaacGym simulator to generalize effectively to both the photo-realistic simulator AvoidBench and real-world environments.
- We improve latent representation quality by optimizing the VAE architecture and applying min-pooling dilation to better capture obstacle structures in depth maps.

II. Related Work

A. RL-Based Obstacle Avoidance

RL has been widely applied to navigation tasks on various robotic platforms, including drones [2], [10]. Although effective in state-based settings, training RL agents directly from raw pixel data remains challenging due to the high dimensionality of visual inputs. To overcome the difficulty of training vision-based RL policies directly from high-dimensional input, previous work has adopted a two-stage strategy. First, an expert policy is trained using state-based observations, which are typically low-dimensional and easier to learn. Then, a vision-based student policy is trained via imitation learning to mimic the expert [11], [12].

An alternative solution is to use representation learning with staged pretraining, which first extracts compact and informative features from high-dimensional inputs before training the policy, offering improved sample efficiency and generalization

Received 18 May 2025; accepted 22 September 2025. Date of publication 3 October 2025; date of current version 13 October 2025. This article was recommended for publication by Associate Editor G. Shi and Editor G. Loianno upon evaluation of the reviewers' comments. This work was supported by European Commission Horizon Project SPEAR under Grant Agreement 101119774. *(Corresponding author: Hang Yu.)*

The authors are with the Faculty of Aerospace Engineering, Delft University of Technology, 2629 HS Delft, The Netherlands (e-mail: h.y.yu@tudelft.nl; g.c.h.e.decroon@tudelft.nl; c.dewagter@tudelft.nl).

Code: https://github.com/tudelft/DepthTransfer.git

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2025.3617729, provided by the authors.

Digital Object Identifier 10.1109/LRA.2025.3617729

2377-3766 © 2025 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: TU Delft Library. Downloaded on October 28,2025 at 12:02:24 UTC from IEEE Xplore. Restrictions apply.

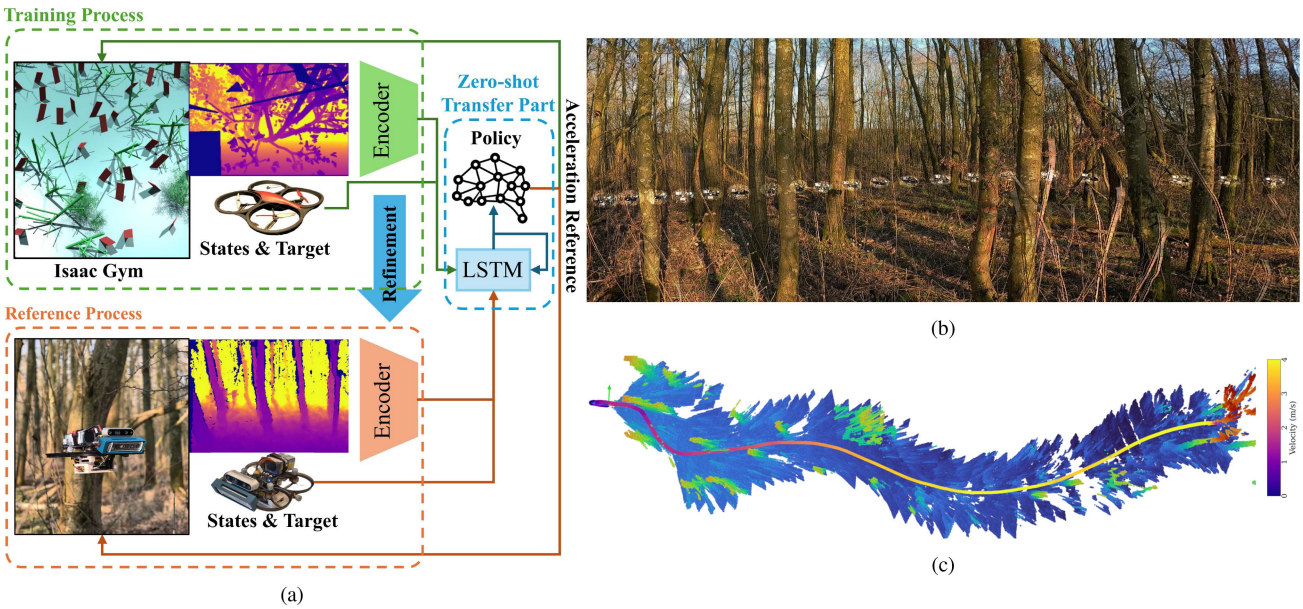


Fig. 1. (a) is the basic framework of the obstacle-free navigation system. When training, the depth images are generated from IsaacGym and the policy is trained in the simulator. During real-world deployment, the depth images are collected from real environments and the policy is evaluated using the refined encoder. (b) illustrates the forest environment for policy evaluation. (c) shows the trajectory from (b), the point clouds here are only for visualization.

compared to fully end-to-end learning. For instance, Mihir et al. [13] introduced the Depth Image-based Collision Encoder (DCE), which addresses the challenge of high-dimensional depth input by compressing it into a compact latent space, while simultaneously enhancing the preservation of fine obstacle details through geometric expansion [14]. A well-structured latent space improves both navigation and obstacle avoidance performance. To incorporate temporal context, David et al. [15] used the Long Short-Term Memory (LSTM) to predict future depth from historical latent features. Building on this idea, a subsequent approach [1] enhances memory representation by predicting both the previous and current depth images, yielding a more structured and explicit latent space that outperforms memory-free baselines and methods relying solely on future predictions. We verify that optimizing the VAE architecture to improve depth reconstruction, combined with simple pixel-level dilation, can also preserve more obstacle details and enhance navigation success.

B. Sim-to-Real Transfer and Domain Adaptation

Multiple techniques have been proposed to improve sim-to-real transfer. To improve generalization, domain randomization has been widely used by introducing physical perturbations in simulators [2], [10], [16], [17]. While effective for robotic dynamics and obstacle layouts, it struggles with complex visual discrepancies. For instance, [18] proposed using visually diverse simulation data, but this approach suffers from limited coverage and reduced training efficiency.

By optimizing contrastive loss between source and target embeddings, a drone racing policy can achieve zero-shot transfer [19]. However, this requires anchor–target image pairs with consistent scene elements, such as gate positions. In obstacle avoidance, real-world obstacles often differ in shape and size from simulation, making such correspondence difficult and reducing the effectiveness of contrastive loss.

Stereo vision has a number of challenges that lead to imperfect depth maps, including texture-poor areas, reflections, areas that are occluded in one of the images, etc. This leads to artifacts and invalid regions in stereo depth images, making them substantially different from ground-truth depth images available in simulations. One partial solution to this is to perform stereo vision with images generated in the simulator [1], [9]. However, doing this during reinforcement learning is computationally expensive and still differs from real-world data. Reducing the resolution of the depth map, as suggested in [6], can help mitigate the domain gap introduced by stereo vision. However, this process often results in the loss of fine details, making it less effective for detecting thin or small obstacles.

Domain adaptation provides an alternative by aligning models trained on one domain with data from another. It has been widely explored in computer vision [20], [21], [22], and more recently in reinforcement learning [23]. Many methods adopt a two-stage pipeline: (1) learning domain-invariant representations, followed by (2) RL training [24], [25]. However, these methods often require data from multiple domains during training and are typically still validated in visual environments like CarRacing or CARLA [26].

In this work, we propose a feature-level domain adaptation method to bridge the gap between simulated and real-world depth. A VAE trained on IsaacGym ground truth depth is aligned with stereo depth via domain adaptation, enabling an RL policy trained in simple simulation to transfer effectively to AvoidBench [7] and real-world environments.

III. Depth Transfer Based on Domain Adaptation

Following the training pipeline proposed in [1], we adopt a similar approach to train an RL agent for obstacle-free navigation. The VAE is used to embed high-dimensional depth images into a 64-dimensional latent space, which is then processed by a LSTM module. The LSTM is trained for the reconstruction

Authorized licensed use limited to: TU Delft Library. Downloaded on October 28,2025 at 12:02:24 UTC from IEEE Xplore. Restrictions apply.

of both the previous and current depth images, producing a 256-dimensional latent representation that encodes temporal information. This output is concatenated with the drone's state and target information and is used as input to the RL policy.

To bridge the visual gap between simulation and the real world, we propose a depth transfer method based on domain adaptation. This approach aligns the latent spaces of depth images across domains, allowing the policy trained in simulation to be directly deployed in real-world environments. The full training process consists of the following steps:

1) Train an initial PPO policy in IsaacGym with a few obstacles using ground-truth depth images. The encoder and LSTM are integrated but randomly initialized and kept frozen during this step.
2) Collect depth images with the initial policy and train the VAE to map into a 64-dimensional latent space.
3) Train the LSTM to reconstruct both past and current depth images from the latent embeddings.
4) Retrain the PPO policy using the outputs from the trained LSTM, concatenated with the drone's state and target position, as policy input.
5) Perform domain adaptation by retraining the encoder on stereo depth images from IsaacGym, AvoidBench, or the real world to align their latent space with that of the simulated ground truth.
6) Evaluate the policy in AvoidBench and real-world environments using the refined encoder.

A. Latent Representation Learning

We employ the VAE to learn a latent representation of depth images. The VAE consists of an encoder E and a decoder D. The encoder maps depth images X to a latent space $\mathbf{z}_{\text{vae}}$, while the decoder reconstructs the depth images from this latent representation. The VAE is trained by minimizing both the reconstruction loss and the Kullback-Leibler (KL) divergence between the learned latent distribution and a prior distribution.

Unlike previous work [1], which relied solely on convolutional layers for feature extraction, we incorporate a residual neural network (ResNet) architecture to enhance the quality of the learned latent space. The ResNet architecture consists of a series of residual blocks, each containing two convolutional layers with skip connections. These skip connections facilitate gradient flow, allowing the network to learn more complex features and improving the reconstruction quality of depth images. In addition, to solve the problem of thin obstacles in depth maps, we apply a simple pixel-level dilation to expand the obstacle sizes in the depth images, which we call the min-pooling dilation. The min-pooling dilation can be expressed using max-pooling as follows:

$$Y_{i,j} = \min_{(m,n)\in\mathcal{R}_{i,j}} X_{m,n} \tag{1}$$

where X is the input depth map, $\mathcal{R}_{i,j}$ represents the receptive field (i.e., the pooling window) centered at position (i, j), and Y is the resulting output after the min-pooling operation.

As shown in the top part of Fig. 2, the VAE is trained on depth images from IsaacGym to extract compact latent representations. The encoder consists of a 16-channel convolutional layer followed by six residual blocks, and outputs the mean and variance of the latent distribution. The decoder mirrors the encoder to reconstruct the input depth image. The VAE is optimized with a combination of reconstruction and KL divergence losses.

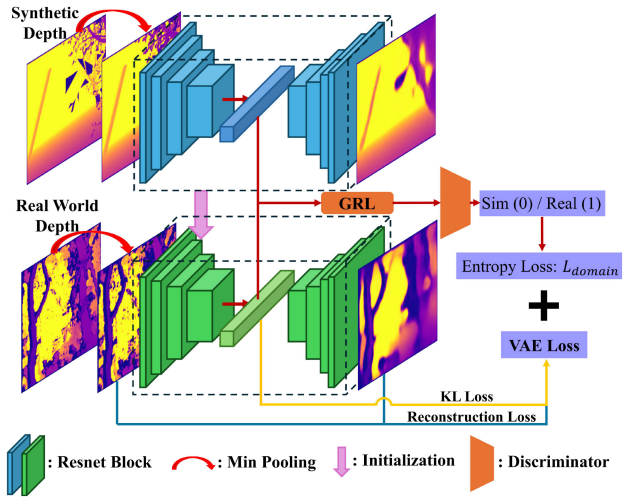


Fig. 2. Domain adaptation for depth transfer. The VAE is trained using depth images from IsaacGym. The encoder maps depth images to a latent space, while the decoder reconstructs the depth images. The GRL and discriminator are used to align the latent spaces of simulated and real-world depth images.

As illustrated in Fig. 1(a), the latent vector from the encoder is used as input to the LSTM. Following the architecture in [1], the LSTM is trained to reconstruct both the previous and current depth images to capture temporal context. Specifically, the LSTM output at time t is passed through a fully connected (FC) layer and then split into two latent vectors, $\hat{\mathbf{z}}_{t-T}$ and $\hat{\mathbf{z}}_t$, which correspond to the predicted latent space for time steps $t - T$ and t, respectively. These two latent vectors are then passed through the same VAE decoder used during pretraining to reconstruct the corresponding depth images, $\hat{Y}_{t-T}$ and $\hat{Y}_t$. Its output is concatenated with the drone's state and target position to train the PPO policy. The LSTM is optimized using the following loss:

$$\mathcal{L}_{\text{LSTM}} = \sum_{i=-1,0} \text{MSE}(Y_{t+iT}, \hat{Y}_{t+iT}), \tag{2}$$

where $\hat{Y}_{t+iT}$ denotes the reconstructed depth image at time $t + iT$ from the latent input $\mathbf{z}_t$, and T is the temporal interval between prediction steps. Following the setting validated in [1], we set $T = 20$.

B. Sim-to-Real Depth Transfer

After training the policy in IsaacGym, we collect stereo depth images from simulators or real-world environments and refine the VAE to align the resulting latent space with that learned from ground-truth depth in simulation, as shown in the bottom part of Fig. 2. Before encoding, stereo depth images are preprocessed with min-pooling dilation to enlarge obstacle regions. The VAE is initialized with weights from simulation and fine-tuned using the collected stereo data.

To bridge the domain gap, we adopt a Gradient Reversal Layer (GRL) and a discriminator. The use of GRL and domain discriminator for adversarial domain adaptation was first introduced by Ganin et al. [27]. This approach enables feature alignment

Authorized licensed use limited to: TU Delft Library. Downloaded on October 28,2025 at 12:02:24 UTC from IEEE Xplore. Restrictions apply.

through backpropagation without requiring explicit domain labels during training. The GRL is placed between the latent space and discriminator to reverse gradients during backpropagation, encouraging the encoder to generate domain-invariant representations. The discriminator is trained to distinguish between simulated and real-world latent features, thus guiding the encoder toward alignment. During the forward pass, the GRL behaves as an identity function, i.e., $\mathbf{z}_{\text{grl}} = \mathbf{z}_{\text{vae}}$. where $\mathbf{z}_{\text{grl}}$ is the output of the GRL. During backpropagation, the gradient of the loss function L with respect to the input is reversed and scaled by λ_{grl}:

$$\frac{\partial L}{\partial \mathbf{z}_{\text{vae}}} = -\lambda_{\text{grl}} \frac{\partial L}{\partial \mathbf{z}_{\text{grl}}} \tag{3}$$

where λ_{grl} is a hyperparameter that controls the strength of the gradient reversal.

The discriminator is a binary classifier consisting of two fully connected layers followed by a log-softmax layer. It outputs a probability p indicating whether the input latent vector originates from the ground depth or the stereo depth. The entropy loss of the discriminator is combined with the VAE loss to update the VAE for stereo depth images. The loss function for the VAE is then updated as follows:

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}_{\text{recon}} + \beta \mathcal{L}_{\text{KL}} + \gamma \mathcal{L}_{\text{domain}} \tag{4}$$

where $\mathcal{L}_{\text{domain}}$ is the domain loss, and γ is a hyperparameter that balances the domain loss with the reconstruction loss and KL divergence.

The domain adaptation loss $\mathcal{L}_{\text{domain}}$ is defined as a binary cross-entropy loss applied to the discriminator, which distinguishes whether a latent vector $\mathbf{z}_{\text{vae}}$ comes from ground-truth or stereo depth images:

$$\mathcal{L}_{\text{domain}} = -\mathbb{E}_{\mathbf{z}_{\text{vae}}^{\text{gt}}} \log D_{\text{dis}}(\mathbf{z}_{\text{vae}}^{\text{gt}}) - \mathbb{E}_{\mathbf{z}_{\text{vae}}^{\text{stereo}}} \log \left(1 - D_{\text{dis}}(\mathbf{z}_{\text{vae}}^{\text{stereo}})\right), \tag{5}$$

where $\mathbf{z}_{\text{vae}}^{\text{gt}} = E(X^{\text{gt}})$ and $\mathbf{z}_{\text{vae}}^{\text{stereo}} = E(X^{\text{stereo}})$ are latent vectors extracted from ground-truth and stereo depth images, respectively. The discriminator $D_{\text{dis}}(\cdot)$ outputs the probability that the input belongs to the ground-truth domain, and $\mathbb{E}[\cdot]$ denotes the batch-wise expectation.

Since the GRL inverts the gradient during backpropagation, the encoder is trained adversarially to fool the discriminator, encouraging the latent representations of ground truth and stereo depth images to become indistinguishable.

IV. Reinforcement Learning for Obstacle Avoidance

In this section, we describe the RL training process for obstacle avoidance. The RL agent is trained using the latent representation learned from the VAE and LSTM, combined with the drone's state and target information. The agent is trained using the PPO algorithm, which maximizes the expected cumulative reward by updating the policy parameters.

A. Task Formulation

The obstacle avoidance task is formulated as a Markov Decision Process (MDP), defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$. Here, $\mathcal{S}$ denotes the state space, which includes the drone's state, target information, and the latent representation extracted from depth images. The action space $\mathcal{A}$ is defined as $\boldsymbol{a} = [\boldsymbol{a}_{\text{cmd}}, \dot{\psi}]$, where $\boldsymbol{a}_{\text{cmd}}$ is the commanded acceleration and $\dot{\psi}$ is the yaw rate.

In our case, the transition function $\mathcal{P}$ is deterministic. We use the Aerial Gym simulator [5] for training the RL agent. Aerial Gym is built on top of the IsaacGym physics engine, which provides a drone model with realistic dynamics.

Given an action command $\boldsymbol{a}$, we employ the reference trajectory generator from Paparazzi[1] to produce smooth motion trajectories. By tuning the trajectory smoothing parameters, we can closely match the drone's response in simulation to that of the real platform, effectively reducing the sim-to-real gap caused by differences in dynamics and control.

A low-level controller maps the reference trajectory to the corresponding thrust and torque commands, which are applied to the drone model. The drone's state is subsequently updated through rigid-body dynamics simulated by the physics engine.

B. Observations and Reward Function

1) Observations: At each time step t, the observation $\boldsymbol{o}_t$ consists of a 14-dimensional physical state vector and a 256-dimensional perception vector obtained from the VAE and LSTM. The full observation is defined as:

$$\boldsymbol{o}_t = [\log d_{\text{hor}},\ d_z,\ \boldsymbol{d}_{\text{norm}},\ \boldsymbol{v}_{\mathcal{W}},\ \boldsymbol{e},\ \boldsymbol{\omega}_{\mathcal{B}},\ \boldsymbol{z}_t], \tag{6}$$

where d_{hor} and d_z denote the horizontal and vertical distances to the target; $\boldsymbol{d}_{\text{norm}}$ is the normalized direction vector to the target; $\boldsymbol{v}_{\mathcal{W}}$ is the linear velocity in the world frame; $\boldsymbol{e} = [\phi,\ \theta,\ \psi]^\top$ and $\boldsymbol{\omega}_{\mathcal{B}}$ are the Euler angle and the angular velocity in the body frame; and $\boldsymbol{z}_t$ is the 256-dimensional latent representation of the depth image.

2) Reward Function: The reward function consists of a **progress reward** r_{prog} and several **terminal rewards**. The progress reward is defined as:

$$\begin{aligned} r_{\text{prog}} = {} & \lambda_d \cdot \log d_{\text{hor}} + \lambda_z \cdot d_z + \max(0,\ v_{\text{hor}} - v_{\max}) \cdot \lambda_v \cdot v_{\text{hor}} \\ & + \lambda_{\text{dir}} \cdot \|\boldsymbol{d}_{\text{norm}} - \boldsymbol{v}_{\text{norm}}\|_1 + \lambda_{\text{input}} \cdot \|\boldsymbol{\omega}_{\mathcal{B}}\| \\ & + \lambda_{\text{perc}} \cdot (|v_{\mathcal{B},y}| + \max(0,\ -v_{\mathcal{B},x})), \end{aligned} \tag{7}$$

where λ_d, λ_z, λ_v, λ_{dir}, λ_{input}, and λ_{perc} are negative hyperparameters that shape the agent's behavior.

The first two terms penalize the horizontal and vertical distances to the target, encouraging the drone to approach it. The third term penalizes excessive horizontal velocity v_{hor} when it exceeds the predefined limit $v_{\max}$, ensuring safe and controlled motion. The fourth term penalizes the angular deviation between the normalized distance vector $\boldsymbol{d}_{\text{norm}}$ and velocity vector $\boldsymbol{v}_{\text{norm}}$, guiding the drone to move directly toward the target. The fifth term penalizes high angular velocity $\boldsymbol{\omega}_{\mathcal{B}}$, promoting stable attitude control. Finally, the last term introduces a perception-aware penalty by discouraging lateral and backward motion in the body-frame velocity $\boldsymbol{v}_{\mathcal{B}} = [v_{\mathcal{B},x},\ v_{\mathcal{B},y},\ v_{\mathcal{B},z}]$. This encourages the drone to maintain a forward-facing flight. This behavior does not emerge reliably without explicit penalization and was found to improve obstacle avoidance performance in our experiments.

This progress reward is designed to balance goal-reaching efficiency, flight stability, and safety, guiding the drone toward the target while maintaining a favorable attitude.

[1] wiki.paparazziuav.org

Authorized licensed use limited to: TU Delft Library. Downloaded on October 28,2025 at 12:02:24 UTC from IEEE Xplore. Restrictions apply.

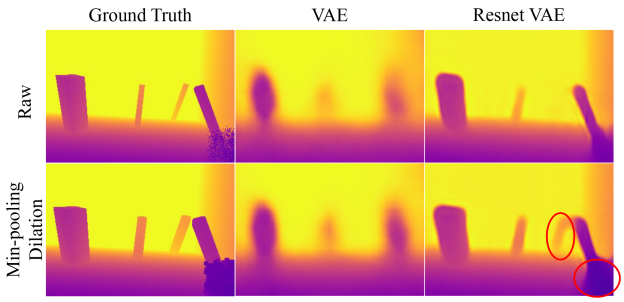


Fig. 3. Depth reconstruction with min-pooling dilation. The first column shows the original depth and after dilation. The second and third columns compare the reconstructed depth images from a standard VAE and a VAE with a ResNet architecture.

TABLE I
ABLATION STUDIES FOR RESNET VAE AND MIN-POOLING DILATION

	VAE	VAE & Dilation	Resnet VAE	Resnet VAE & Dilation
Success Rate (%)	50.4	64.2	59.6	**71.7**

The overall reward function, composed of the progress reward and several terminal rewards, is defined as:

$$r = \begin{cases} r_{\text{exceed}}, & \text{if } (p_t < p_{\min} \text{ or } p_t > p_{\max}), \quad p \in \{x, y, z\} \\ r_{\text{arrive}}, & \text{if } \|\boldsymbol{d}\| < d_{\min} \\ r_{\text{collision}}, & \text{if collision occurs} \\ r_{\text{prog}}, & \text{otherwise} \end{cases} \tag{8}$$

where $p_{\min}$ and $p_{\max}$ define the allowable position boundaries, and r_{exceed} is the penalty for flying out of bounds. The term $d_{\min}$ is the arrival threshold, and r_{arrive} denotes the reward for successfully reaching the target. The term $r_{\text{collision}}$ penalizes collisions with obstacles, while r_{prog} refers to the progress reward defined in (7).

V. EXPERIMENTS

A. Depth Reconstruction With Min-Pooling Dilation

Different from prior work [1], we apply min-pooling dilation to enlarge obstacle regions in depth images. As shown in the first column of Fig. 3, this makes obstacles more prominent and easier to detect. The second and third columns compare reconstructions from a standard VAE [1] and a ResNet-based VAE, with the latter producing outputs much closer to the ground truth. When combined with min-pooling dilation, reconstruction quality improves further—especially for nearby regions—by preserving fine details and better separating obstacles from the background.

To assess the impact of the ResNet-based VAE and min-pooling dilation on obstacle avoidance, we trained four policies with different latent representations: standard VAE, VAE with dilation, ResNet VAE, and ResNet VAE with dilation. Evaluation was performed on 24 maps of varying complexity, each with 10 trials. As shown in Table I, both the ResNet architecture and min-pooling yield substantial performance gains, with their combination achieving the highest success rate. While the absolute success rates may appear modest, the task setting is highly challenging, featuring dense and randomized obstacle layouts.

TABLE II
ABLATION STUDY OF GRL HYPERPARAMETERS

λ_{grl} / γ	0.5 / 50	1 / 1	1 / 10	1 / 20	1 / 50
Domain loss	0.542	0.231	0.309	0.453	**0.610**
Success Rate (%)	75	61.3	68.8	**77.5**	**77.5**

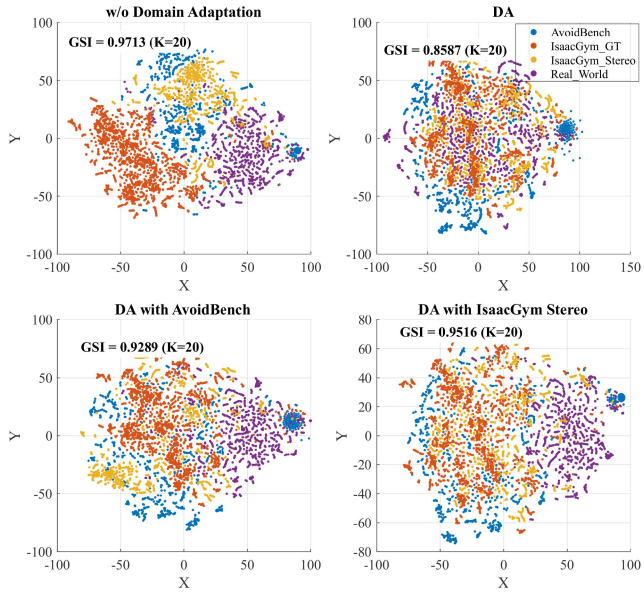


Fig. 4. t-SNE visualization of latent space alignment. The plot includes ground truth depth images from IsaacGym (orange), stereo depth images from IsaacGym (yellow), stereo depth images from AvoidBench (blue), and real-world depth images (purple).

B. Visualization and Analysis of Latent Representations

The RL policy will be trained in IsaacGym with ground truth depth images, since using the stereo depth means the simulator always need to render two images and run the stereo matching algorithm, which is time-consuming and not efficient. The goal of the domain adaptation is to align the latent spaces of the depth images from different sources, so that the policy trained in IsaacGym can be directly transferred to the visually more realistic AvoidBench and real-world environments.

To support the VAE encoder domain adaptation, we collected about 80,000 ground-truth depth images and 80,000 stereo depth images from IsaacGym, using the former as source data and the latter as target data. We further collected roughly 40,000 stereo depth images from AvoidBench and 6,000 real-world depth images with an Intel RealSense camera, both as target data. All simulation data were acquired automatically using a pre-trained policy, without manual intervention.

Table II summarizes an ablation study on the GRL-related hyperparameters λ_{grl} and γ, which control the gradient reversal strength and domain loss weight, respectively. The results show that our method remains effective across a range of values. The highest success rate (77.5%) is achieved when $\lambda_{\text{grl}} = 1.0$ and

Authorized licensed use limited to: TU Delft Library. Downloaded on October 28,2025 at 12:02:24 UTC from IEEE Xplore. Restrictions apply.

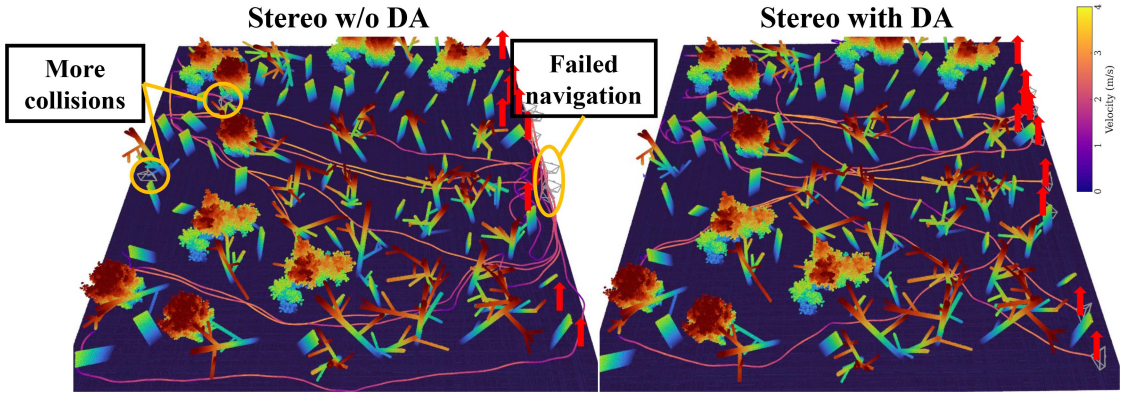


Fig. 5. Policy evaluation in IsaacGym with different encoders. The left picture shows the policy evaluated with stereo depth images without domain adaptation, and the right picture shows the policy evaluated with stereo depth images after domain adaptation.

$\gamma = 50$, corresponding to a balanced domain loss. Performance degrades only when γ is too small ($\gamma = 10$), indicating excessive discriminator strength. These findings suggest that our framework is robust to hyperparameter choices.

To evaluate the effectiveness of feature-level domain adaptation, we use t-SNE [28] to visualize the latent space (Fig. 4). The visualization includes four categories: ground-truth depth from IsaacGym (orange), stereo depth from IsaacGym (yellow), AvoidBench (blue), and real-world data (purple).

In the top-left plot of Fig. 4, a single encoder trained on ground-truth depth produces clear domain separation, especially between ground truth and the three stereo-style depth categories. After applying domain adaptation (top-right), the distributions become closely aligned, indicating effective feature-level adaptation.

The bottom plots examine whether an encoder refined with IsaacGym or AvoidBench stereo depth can generalize to real-world inputs. Compared to the top-right configuration (with dedicated encoders), both single-encoder variants show poorer alignment with real-world depth images.

We quantify the alignment using the Geometric Separability Index (GSI) [29], where lower values indicate better inter-class mixing. We computed the GSI values for the ground truth depth images from IsaacGym and real-world depth images under four different encoder configurations, as reported in each plot of Fig. 4. The configuration where all categories use different encoders achieves the lowest GSI value (0.8587), indicating the best latent space alignment. In contrast, the configuration where the encoder was refined using stereo depth images from AvoidBench results in a higher GSI value (0.9289), though still lower than the baseline where all categories use the same encoder (0.9713). Refining the encoder with stereo depth images from IsaacGym also improves alignment (0.9516) for real-world depth images, though it is less effective than using AvoidBench for adaptation.

These results confirm that domain adaptation effectively aligns the latent spaces of different depth inputs, with the best alignment achieved using separate encoders or AvoidBench-refined features for real-world deployment.

C. Zero-Shot Policy Transfer in IsaacGym

We assess the effectiveness of our depth transfer method by deploying a policy trained on ground truth depth images to stereo depth images in IsaacGym. Evaluation is conducted across eight maps with different obstacle layouts, each comprising ten trials with randomized start and target positions. The success rate is defined as the percentage of trials in which the drone reaches the target without collisions.

TABLE III
EVALUATIONS ACROSS DIFFERENT LATENT SPACE CONFIGURATIONS

	GT	Stereo Baseline	Stereo	Refinement w/o DA	Domain Adaptation	LSTM & DA
Success Rate (%)	92.5	78.8	40.0	63.8	77.5	**81.2**

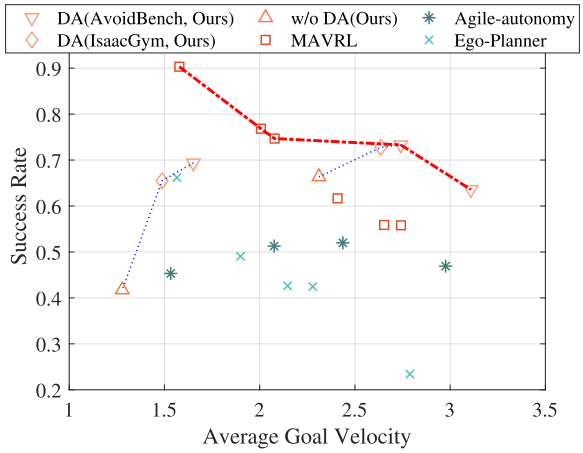
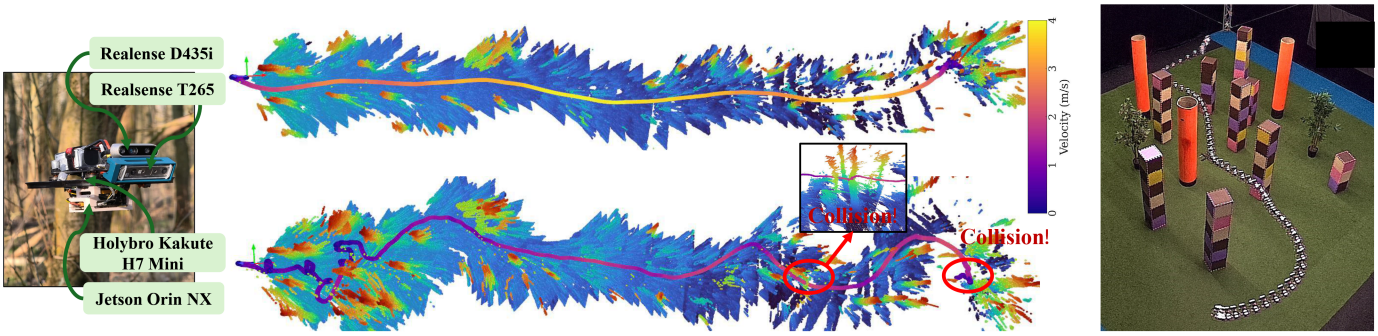


Fig. 6. Pareto front of the proposed method compared to existing approaches.

TABLE IV
STATISTICAL RESULTS OF REAL FOREST FLIGHT EXPERIMENTS

	Success rate (%)	Average speed (m/s)	Number of collisions	Average jerk (m/s^3)
w/ DA	**70.0**	**2.15**	**3**	0.0437
w/o DA	20.0	0.64	5	**0.0385**

Authorized licensed use limited to: TU Delft Library. Downloaded on October 28,2025 at 12:02:24 UTC from IEEE Xplore. Restrictions apply.



(a) (b)

Fig. 7. Real-world experiments conducted in both outdoor (a) and indoor (b) environments demonstrate the effectiveness and generalization of our domain-adapted depth transfer method.

As shown in Fig. 5, the policy trained on ground truth depth fails to generalize to stereo depth images without domain adaptation (left), resulting in frequent collisions and missed targets. After applying domain adaptation (right), the latent space is better aligned, enabling successful navigation.

We also trained a stereo baseline policy using stereo depth images from IsaacGym. Owing to the higher computational cost, it required about 50 hours to converge, compared to 18 hours for the policy trained on ground-truth depth. All experiments were run on an RTX 4090 GPU. The evaluation results are summarized in Table III. The *GT* column represents the upper bound performance, where evaluation is conducted using ground truth depth images. The *Stereo Baseline* column shows the performance of the stereo baseline policy, which is lower than the *GT* upper bound due to the challenges of training with stereo depth images. The *Stereo* column presents the evaluation results for a policy trained on ground truth depth images but evaluated directly with stereo depth images without any refinement, resulting in a significant performance drop. The *Refinement w/o DA* column shows results where the encoder is refined solely using the VAE loss, without domain adaptation. In contrast, the *Domain Adaptation* column represents a policy trained on ground truth depth images but evaluated with stereo depth images after applying domain adaptation, demonstrating improved generalization. All of the above policies use the VAE latent space as input. The highest success rate is achieved by the *LSTM & DA* configuration, which incorporates temporal memory via an LSTM and refines the encoder but not the policy using both reconstruction and domain adaptation losses.

D. Zero-Shot Policy Transfer in AvoidBench

To evaluate the overall performance of our obstacle avoidance approach, we benchmarked the proposed method using AvoidBench. Two policies with different maximum speed limits were trained by fine-tuning the progress reward hyperparameters in Equation 7. These policies were then evaluated using stereo depth images as input. As shown in Fig. 6, the x-axis denotes the average goal velocity [7], while the y-axis indicates the corresponding success rate. Each marker in the plot represents aggregated results over 1260 trials (30 trials per map across 42 maps).

Each policy is evaluated using three encoder configurations: an upward-pointing triangle denotes the encoder trained solely on ground truth depth images from IsaacGym without domain adaptation; a diamond-shaped marker represents the encoder refined with stereo depth images from IsaacGym; and a downward-pointing triangle indicates the encoder refined with stereo depth images from AvoidBench. Although the policies are trained in IsaacGym using ground truth depth images, the domain adaptation is conducted separately using stereo depth data from different sources. The points connected by blue dotted lines represent the results of the same policy using latent spaces generated by different encoders, while the red dotted line denotes the Pareto frontier.

Square markers represent MAVRL [1], which trains both the encoder and policy directly in AvoidBench using stereo depth. Star and cross markers indicate Agile-Autonomy [9] and Ego-Planner [8], respectively. MAVRL performs best in low-speed scenarios (average velocity $< 2.1\ \mathrm{m/s}$), likely due to full training in AvoidBench. In contrast, our method dominates the high-speed regime, benefiting from a reference generator that ensures consistent trajectory tracking and stable flight. Despite being trained entirely in IsaacGym, our policy generalizes effectively via domain adaptation—even when refined with stereo depth from different sources.

For the same policy, using different target data for domain adaptation leads to varying performance. The upward trend of the blue dashed line shows that employing stereo depth maps from IsaacGym and AvoidBench for domain adaptation substantially improves obstacle avoidance, with AvoidBench target data yielding the highest success rate and speed. These results highlight the effectiveness of our domain adaptation method in aligning latent spaces across domains, enabling robust zero-shot policy transfer across simulation and real-world environments.

E. Zero-Shot Policy Transfer in Real-World Environments

To validate our depth transfer method in real-world scenarios, we deploy the trained policy on a drone with a $150\,\mathrm{mm}$ wheelbase, equipped with an RealSense D435i for depth and a RealSense T265 for state estimation. The low-level controller runs on a Holybro Kakute H7 Mini with Betaflight, while the high-level policy runs on a Jetson Orin NX.

The RealSense D435i runs at 30 Hz, while the policy network outputs actions at 10 Hz to match onboard processing limits. These actions are sent to a low-level Model Predictive Control (MPC) operating at 100 Hz [30]. With a maximum speed of

Authorized licensed use limited to: TU Delft Library. Downloaded on October 28,2025 at 12:02:24 UTC from IEEE Xplore. Restrictions apply.

5.0 m/s, the drone travels 0.5 m in 100 ms, which is sufficient for timely reactions. An adaptive speed strategy [1] further improves robustness in complex environments.

Fig. 7(a) illustrates the real-world flight trajectories of the drone using two different encoders: one trained on IsaacGym ground truth depth images (bottom trajectory) and another refined with domain adaptation (top trajectory). The results demonstrate that domain adaptation enables direct transfer of the policy trained in IsaacGym to real-world environments, allowing the drone to successfully avoid obstacles and reach the target. In contrast, when using the unrefined encoder, the drone exhibited erratic lateral movements in dense obstacle regions and collided twice, while also maintaining a lower overall speed compared to the domain-adapted encoder.

Table IV presents the quantitative results from the cluttered forest flight experiments. With domain adaptation, both the success rate and average speed improved significantly, while the number of collisions was reduced. Most failures during domain-adapted flights occurred when the state estimator experienced substantial drift or when the drone encountered extremely small branches.

We also test the same policy in a challenging indoor environment (Fig. 7(b)) using the encoder refined with outdoor stereo depth. The drone navigates narrow corridors and avoids obstacles, demonstrating strong generalization of our method to diverse real-world settings.

VI. Conclusion

We propose a novel depth transfer approach for training reinforcement learning policies in simulation and deploying them in real-world environments. Our method leverages a VAE combined with an LSTM to extract latent representations from depth images, enabling the policy to capture temporal dependencies and generalize across diverse environments. To further enhance transferability, we introduce a domain adaptation technique that aligns the latent spaces of simulated and real-world depth images. Experiments show that this adaptation markedly improves policy performance, enabling effective zero-shot transfer across different environments.

References

[1] H. Yu, C. Wagter, and G. C. H. E. de Croon, "MAVRL: Learn to fly in cluttered environments with varying speed," *IEEE Robot. Automat. Lett.*, vol. 10, no. 2, pp. 1441–1448, Feb. 2025.

[2] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, no. 7976, pp. 982–987, 2023.

[3] Y. Song, S. Naji, E. Kaufmann, A. Loquercio, and D. Scaramuzza, "Flightmare: A flexible quadrotor simulator," in *Proc. Conf. RobotLearning. PMLR*, 2021, pp. 1147–1157.

[4] V. Makoviychuk et al., "Isaac Gym: High performance GPU-based physics simulation for robot learning," 2021, *arXiv:2108.10470*.

[5] M. Kulkarni, T. J. L. Forgaard, and K. Alexis, "Aerial Gym–Isaac gym simulator for aerial robots," 2023, *arXiv:2305.16510*.

[6] Y. Zhang, Y. Hu, Y. Song, D. Zou, and W. Lin, "Learning vision-based agile flight via differentiable physics," *Nature Mach. Intell.*, vol. 7, pp. 954–966, 2025.

[7] H. Yu, G. C. H. E. de Croon, and C. De Wagter, "AvoidBench: A high-fidelity vision-based obstacle avoidance benchmarking suite for multi-rotors," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 9183–9189.

[8] X. Zhou, Z. Wang, H. Ye, C. Xu, and F. Gao, "EGO-planner: An ESDF-free gradient-based local planner for quadrotors," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 478–485, Apr. 2021.

[9] A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza, "Learning high-speed flight in the wild," *Sci. Robot.*, vol. 6, no. 59, Oct. 2021, Art. no. eabg5810.

[10] Y. Song, A. Romero, M. Müller, V. Koltun, and D. Scaramuzza, "Reaching the limit in autonomous racing: Optimal control versus reinforcement learning," *Sci. Robot.*, vol. 8, no. 82, 2023, Art. no. eadg1462.

[11] Y. Song, K. Shi, R. Penicka, and D. Scaramuzza, "Learning perception-aware agile flight in cluttered environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 1989–1995.

[12] J. Xing, A. Romero, L. Bauersfeld, and D. Scaramuzza, "Bootstrapping reinforcement learning with imitation for vision-based agile flight," in *Proc. 8th Conf. Robot Learn.*, vol. 270. PMLR, 2025, pp. 2542–2556.

[13] M. Kulkarni et al., "Task-driven compression for collision encoding based on depth images," in *Advances in Visual Computing*, G. Bebis, G. Ghiasi, Y. Fang, A. Sharf, Y. Dong, C. Weaver, Z. Leo, J. J. LaViola Jr., and L. Kohli, Eds. Cham, Switzerland: Springer Nature, 2023, pp. 259–273.

[14] M. Kulkarni, H. Nguyen, and K. Alexis, "Semantically-enhanced deep collision prediction for autonomous navigation using aerial robots," in *Proc. Int. Conf. Intell. Robots Syst*. IEEE, 2023, pp. 3056–3063.

[15] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter, "Learning a state representation and navigation in cluttered and dynamic environments," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5081–5088, Jul. 2021.

[16] Y. Chebotar et al., "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 8973–8979.

[17] A. Loquercio, E. Kaufmann, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: From simulation to reality with domain randomization," *IEEE Trans. Robot.*, vol. 36, no. 1, pp. 1–14, Feb. 2020.

[18] 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 *Proc. 2019 Int. Conf. Robot. Automat.*, 2019, pp. 6008–6014.

[19] J. Xing, L. Bauersfeld, Y. Song, C. Xing, and D. Scaramuzza, "Contrastive learning for enhancing robust scene transfer in vision-based agile flight," in *Proc. 2024 IEEE Int. Conf. Robot. Automat.*, 2024, pp. 5330–5337.

[20] J. Hoffman et al., "CyCADA: Cycle-consistent adversarial domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1989–1998.

[21] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, "Domain generalization: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 4, pp. 4396–4415, Apr. 2023.

[22] J. Peng, Y. Huang, W. Sun, N. Chen, Y. Ning, and Q. Du, "Domain adaptation in remote sensing image classification: A survey," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 15, pp. 9842–9859, 2022.

[23] Z. Zhu, K. Lin, A. K. Jain, and J. Zhou, "Transfer learning in deep reinforcement learning: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 11, pp. 13344–13362, Nov. 2023.

[24] J. Xing, T. Nagata, K. Chen, X. Zou, E. Neftci, and J. L. Krichmar, "Domain adaptation in reinforcement learning via latent unified state representation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 12, 2021, pp. 10452–10 459.

[25] D. Li, L. Meng, J. Li, K. Lu, and Y. Yang, "Domain adaptive state representation alignment for reinforcement learning," *Inf. Sci.*, vol. 609, pp. 1353–1368, 2022.

[26] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. Conf. robot Learn.*, 2017, pp. 1–16.

[27] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1180–1189.

[28] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.

[29] C. Thornton "Separability is a learner's best friend," in 4th NeuralComputation and Psychology Workshop, London. Springer, 1998, pp. 40–46.

[30] P. Foehn et al., "Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight," *Sci. Robot.*, vol. 7, no. 67, 2022, Art. no. eabl6259.

Authorized licensed use limited to: TU Delft Library. Downloaded on October 28,2025 at 12:02:24 UTC from IEEE Xplore. Restrictions apply.