VR-Goggles for Robots: Real-to-sim Domain Adaptation for Visual Control

Jingwei Zhang*[1] Lei Tai*[2] Peng Yun[2] Yufeng Xiong[1] Ming Liu[2] Joschka Boedecker[1] Wolfram Burgard[1]

***Abstract*—In this paper, we deal with the *reality gap* from a novel perspective, targeting transferring Deep Reinforcement Learning (DRL) policies learned in simulated environments to the real-world domain for visual control tasks. Instead of adopting the common solutions to the problem by increasing the visual fidelity of synthetic images output from simulators during the training phase, we seek to tackle the problem by translating the real-world image streams back to the synthetic domain during the deployment phase, to *make the robot feel at home*. We propose this as a lightweight, flexible, and efficient solution for visual control, as 1) no extra transfer steps are required during the expensive training of DRL agents in simulation; 2) the trained DRL agents will not be constrained to being deployable in only one specific real-world environment; 3) the policy training and the transfer operations are decoupled, and can be conducted in parallel. Besides this, we propose a simple yet effective *shift loss* that is agnostic to the downstream task, to constrain the consistency between subsequent frames which is important for consistent policy outputs. We validate the *shift loss* for *artistic style transfer for videos* and *domain adaptation*, and validate our visual control approach in indoor and outdoor robotics experiments.**

***Index Terms*—Deep Learning in Robotics and Automation, Visual-Based Navigation, Model Learning for Control.**

I. Introduction

PIONEERED by the Deep Q-network [1] and followed up by various extensions and advancements [2]–[5], Deep Reinforcement Learning (DRL) algorithms show great potential in solving high-dimensional real-world robotics sensory control tasks. However, DRL methods typically require several millions of training samples, making them infeasible to train directly on real robotic systems. As a result, DRL algorithms are generally trained in simulated environments, then transferred to and deployed in real scenes. However, the *reality gap*, namely the noise pattern, texture, lighting condition discrepancies, etc., between synthetic renderings and real sensory readings, imposes major challenges for generalising the sensory control policies trained in simulation to reality.

In this paper, we focus on visual control tasks, where autonomous agents perceive the environment with their onboard cameras, and execute commands based on the colour image reading streams. A natural way and also the typical choice in the recent literature on dealing with the *reality gap* for visual control, is by increasing the visual fidelity of the simulated images [6], [7], by matching the distribution of synthetic images to that of the real ones [8], [9], and by gradually adapting the learned features and representations from the simulated domain to the real-world domain [10]. These *sim-to-real* methods, however, inevitably have to add preprocessing steps for each individual training frame to the already expensive learning pipeline of DRL policies; or a policy training or finetuning phase has to be conducted for each visually different real-world scene.

This paper attempts to tackle the *reality gap* in the visual control domain from a novel perspective, with the aim of adding minimal extra computational burden to the learning pipeline. We cope with the *reality gap* only during the actual deployment phase of agents in real-world scenarios, by adapting the real camera streams to the synthetic modality, so as to translate the unfamiliar or unseen features of real images back into the simulated style, which the agents have already learned how to deal with during training in the simulation.

Compared to the *sim-to-real* methods bridging the *reality gap*, our proposed *real-to-sim* approach, which we refer to as the *VR-Goggles*, has several appealing properties: (1) Our proposed method is highly lightweight: It does not add any extra processing burden to the training phase of DRL policies; and (2) Our approach is highly flexible and efficient: Since we decouple the policy training and the adaptation operations, the preparations for transferring the polices from simulation to the real world can be conducted in parallel with the training of the control policies. From each visually different real-world environment that we expect to deploy the agent in, we just need to collect several (typically on the order of 2000) images, and train a *VR-Goggles* model for each of them. More importantly, we do not need to retrain or finetune the visual control policy for new environments.

As an additional contribution, we propose a new *shift loss*, which enables generating consistent synthetic image streams without imposing temporal constraints, and does not require sequential training data. We show that *shift loss* is a promising and cheap alternative to the constraints imposed by optical flow, and demonstrate its effectiveness in *artistic style transfer for videos* and *domain adaptation*.

This work was supported by: the Shenzhen Science, Technology and Innovation Commission (SZSTI) JCYJ20160428154842603; the BrainLinks-BrainTools cluster of excellence funded by the DFG (German Research Foundation), grant number EXC 1086; the Research Grant Council of Hong Kong SAR Government, China, under Project No. 11210017, No. 16212815 and No. 21202816, the National Natural Science Foundation of China (Grant No. U1713211) awarded to Prof. Ming Liu. *(Corresponding authors: Jingwei Zhang and Lei Tai.)*

*The first two authors contributed equally to this work.

[1]Jingwei Zhang, Yufeng Xiong, Joschka Boedecker and Wolfram Burgard are with the Department of Computer Science, University of Freiburg. Breisgau 79110, Germany (e-mail: {zhang, xiongy, jboedeck, burgard}@informatik.uni-freiburg.de)

[2]Lei Tai, Peng Yun and Ming Liu are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong (e-mail: {ltai, pyun, eelium}@ust.hk)

II. Related Works

A. Domain Adaptation

Visual *domain adaptation*, or *image-to-image translation*, targets translating images from a source domain into a target domain. We here focus on the most general unsupervised methods that require minimal manual effort and are applicable in robotics control tasks.

CycleGAN [11] introduced a cycle-consistent loss to enforce an inverse mapping from the target domain to the source domain on top of the source to target mapping. It does not require paired data from the two domains of interest and shows convincing results for relatively simple data distributions containing few semantic types. However, in terms of translating between more complex data distributions containing many more semantic types, its results are not as satisfactory, in that permutations of semantics often occur. Several works investigate imposing semantic constraints [12], [13], e.g., *CyCADA* [12] enforces a matching between the semantic map of the translated image and that of the input.

B. Domain Adaptation for Learning based Visual Control

Learning-based methods such as DRL and imitation learning have been applied to robotics control tasks including manipulation and navigation. Below we review the recent literature mainly considering the visual *reality gap*.

Bousmalis et al. [6] bridged the *reality gap* for manipulation by adapting synthetic images to the realistic domain during training, with a combination of image-level and feature-level adaptation. Also following the *sim-to-real* direction, Stein et al. [7] utilized *CycleGAN* to translate every synthetic frame to the realistic style during training navigation policies. Although effective, these approaches still add an adaptation step before each training iteration, which can slow down the whole learning pipeline.

The method of domain randomization [8], [9], [14] is proposed to randomize the texture of objects, lighting conditions, and camera positions during training, such that the learned model could generalize naturally to real-world scenarios. However, such randomizing might not be efficiently realized by some robotic simulators at a relatively low cost. Moreover, there is no guarantee that these randomized simulations can cover the visual modality of an arbitrary real-world scene.

Rusu et al. [10] deals with the *reality gap* by progressively adapting the features and representations learned in simulation to that of the realistic domain. This method, however, still needs to go through a policy finetuning phase for each visually different real-world scenario.

Apart from the approaches mentioned above, some works chose special setups to circumvent the *reality gap*. For example, $2D$ Lidar [15]–[17] and depth images [18], [19] are sometimes chosen as the sensor modality, since the discrepancies between the simulated domain and the real-world domain for them can be smaller than those for colour images. Zhu et al. [20] conducted real-world experiments with visual inputs. However, in their setups, the real-world scene is highly visually similar to the simulation, a condition that can be relatively difficult to meet in practice.

Very related to our method is the work of Inoue et al. which also adopts a *real-to-sim* direction [21]. They train VAEs to perform the adaptation during deployment of the trained object detection model in the real world. However, their method relies on paired data between two domains and focuses on supervised perception tasks.

In this paper, we mainly consider *domain adaptation* for learning-based visual navigation. In terms of visual aspects, the adaptation for navigation is quite challenging, since navigation agents usually work in environments at relatively larger scales compared to the relatively confined workspaces for manipulators. We believe our proposed *real-to-sim* method could be potentially adopted in other control domains.

An essential aspect of *domain adaptation*, within the context of dealing with the *reality gap* is the consistency between subsequent frames, which has not been considered in any of the adaptation methods mentioned above. As an approach for solving sequential decision making, the consistency between the subsequent inputs for DRL agents can be critical for the successful fulfilment of their final goals. Apart from solutions for solving the *reality gap*, the general *domain adaptation* literature also lacks works considering sequential frames instead of single frames. Therefore, we look to borrow techniques from other fields that successfully extend single-frame algorithms to the video domain, among which the most applicable methods are from the *artistic style transfer* literature.

C. Artistic Style Transfer for Videos

Artistic style transfer is a technique for transferring the artistic style of artworks to photographs [22]. *Artistic style transfer for videos* works on video sequences instead of individual frames, targeting generating temporally consistent stylizations for sequential inputs. Ruder et al. [23] provides a key observation that: a trained stylization network with a total downsampling factor of K (e.g., $K = 4$ for a network with 2 convolutional layers of stride 2), is shift invariant to shifts equal to the multiples of K pixels, but can output substantially different stylizations otherwise. This undesired property (of not being shift invariant) causes the output of the trained network to change substantially for even very tiny changes in the input, which leads to temporal inconsistency (under the assumption that only relatively limited changes would appear in subsequent input frames). However, their solution of adding temporal constraints between generated subsequent frames, is rather expensive, as it requires optical flow as input during deployment. Huang et al. [24] offers a relatively cheap solution, requiring the temporal constraint only during training single-frame *artistic style transfer*. However, we suspect that constraining optical flow on single frames is not well-defined. We suspect that their improved temporal consistency is actually due to the inexplicitly imposed consistency constraints for regional shifts by optical flow. We validate this suspicion in our experiments (Sec. IV-A).

We propose that the fundamental problem causing the inconsistency can be solved by an additional constraint of *shift loss*, which we introduce in Sec. III-D. We show that the *shift loss* constrains the consistency between generated subsequent

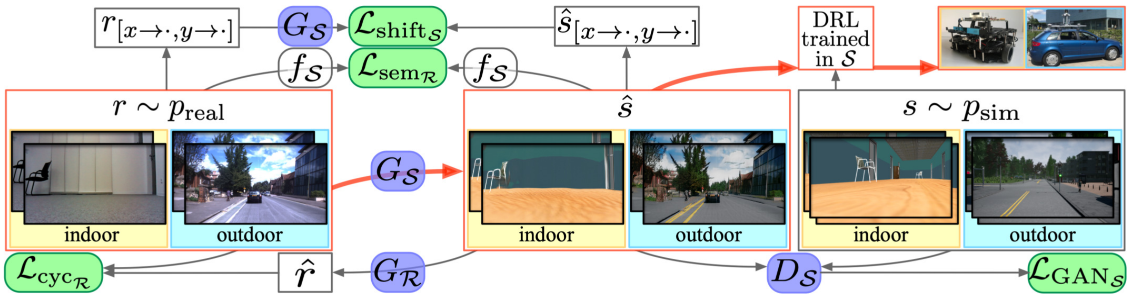

$r_{[x\to\cdot,y\to\cdot]}$
$G_{\mathcal{S}}$
$\mathcal{L}_{\text{shift}_{\mathcal{S}}}$
$\hat{s}_{[x\to\cdot,y\to\cdot]}$
DRL trained in $\mathcal{S}$
$f_{\mathcal{S}}$
$\mathcal{L}_{\text{sem}_{\mathcal{R}}}$
$f_{\mathcal{S}}$
$r \sim p_{\text{real}}$
$\hat{s}$
$s \sim p_{\text{sim}}$
indoor
outdoor
$G_{\mathcal{S}}$
indoor
outdoor
indoor
outdoor
$\mathcal{L}_{\text{cyc}_{\mathcal{R}}}$
$\hat{r}$
$G_{\mathcal{R}}$
$D_{\mathcal{S}}$
$\mathcal{L}_{\text{GAN}_{\mathcal{S}}}$

Fig. 1: The *VR-Goggles* pipeline. We depict the computation of the losses $\mathcal{L}_{\text{GAN}_{\mathcal{S}}}$, $\mathcal{L}_{\text{cyc}_{\mathcal{R}}}$, $\mathcal{L}_{\text{sem}_{\mathcal{R}}}$ and $\mathcal{L}_{\text{shift}_{\mathcal{S}}}$. We present both *outdoor* and *indoor* scenarios, where the adaptation for the *outdoor* scene is trained with the semantic loss $\mathcal{L}_{\text{sem}}$ (since its simulated domain *CARLA* has ground truth semantic labels to train a segmentation network $f_{\mathcal{S}}$), and the *indoor* one without (since its simulated domain *Gazebo* does not provide semantic ground truth). The components marked in *red* are those involved in the final deployment: a real sensor reading is captured ($r \sim p_{\text{real}}$), then passed through the generator $G_{\mathcal{S}}$ to be translated into the simulated domain $\mathcal{S}$, where the DRL agents were originally trained; the translated image $\hat{s}$ is then fed to the DRL policy, which outputs control commands. For clarity, we skip the counterpart losses $\mathcal{L}_{\text{GAN}_{\mathcal{R}}}$, $\mathcal{L}_{\text{cyc}_{\mathcal{S}}}$, $\mathcal{L}_{\text{sem}_{\mathcal{S}}}$ and $\mathcal{L}_{\text{shift}_{\mathcal{R}}}$.

frames, without the need for the relatively expensive optical flow constraint. We argue that for a network that has been properly trained to learn a smooth function approximation, small changes in the input should also result in small changes in the output.

III. Methods

A. Problem formulation

We consider visual data sources from two domains: $\mathcal{S}$, containing sequential frames $\{s_0, s_1, s_2, \cdots\}$ (e.g., synthetic images output from a simulator; $s \sim p_{\text{sim}}$, where p_{sim} denotes the simulated data distribution), and $\mathcal{R}$, containing sequential frames $\{r_0, r_1, r_2, \cdots\}$ (e.g., real camera readings from the onboard camera of a mobile robot; $r \sim p_{\text{real}}$, where p_{real} denotes the distribution of the real sensory readings). We emphasize that, although we require our method to generate consistent outputs for sequential inputs, we do not need the training data to be sequential; we formalize it in this way only because some of our baseline methods have this requirement.

DRL agents are typically trained in the simulated domain $\mathcal{S}$, and expected to execute in the real-world domain $\mathcal{R}$. As we have discussed, we choose to tackle this problem by translating the images from $\mathcal{R}$ to $\mathcal{S}$ during deployment. In the following, we introduce our approach for performing *domain adaptation*. Also to cope with the sequential nature of the incoming data streams, we introduce a *shift loss* technique for constraining the consistency of the translated subsequent frames.

B. CycleGAN Loss

We first build on top of *CycleGAN* [11], which learns two generative models to map between domains: $G_{\mathcal{R}} : \mathcal{S} \to \mathcal{R}$, with its discriminator $D_{\mathcal{R}}$, and $G_{\mathcal{S}} : \mathcal{R} \to \mathcal{S}$, with its discriminator $D_{\mathcal{S}}$, via training two GANs simultaneously:

$$\begin{aligned}\mathcal{L}_{\text{GAN}_{\mathcal{R}}}(G_{\mathcal{R}}, D_{\mathcal{R}}; \mathcal{S}, \mathcal{R}) =& \mathbb{E}_{p_{\text{real}}}\left[\log D_{\mathcal{R}}(r)\right] + \\ & \mathbb{E}_{p_{\text{sim}}}\left[\log(1 - D_{\mathcal{R}}(G_{\mathcal{R}}(s)))\right], \\ \mathcal{L}_{\text{GAN}_{\mathcal{S}}}(G_{\mathcal{S}}, D_{\mathcal{S}}; \mathcal{R}, \mathcal{S}) =& \mathbb{E}_{p_{\text{sim}}}\left[\log D_{\mathcal{S}}(s)\right] + \\ & \mathbb{E}_{p_{\text{real}}}\left[\log(1 - D_{\mathcal{S}}(G_{\mathcal{S}}(r)))\right],\end{aligned}$$

in which $G_{\mathcal{R}}$ learns to generate images $G_{\mathcal{R}}(s)$ matching those from domain $\mathcal{R}$, while $G_{\mathcal{S}}$ translats r to domain $\mathcal{S}$. We also constrain mappings with the *cycle consistency loss* [11]:

$$\begin{aligned}\mathcal{L}_{\text{cyc}_{\mathcal{R}}}(G_{\mathcal{S}}, G_{\mathcal{R}}; \mathcal{R}) = \mathbb{E}_{p_{\text{real}}}\left[||G_{\mathcal{R}}(G_{\mathcal{S}}(r)) - r||_1\right], \\ \mathcal{L}_{\text{cyc}_{\mathcal{S}}}(G_{\mathcal{R}}, G_{\mathcal{S}}; \mathcal{S}) = \mathbb{E}_{p_{\text{sim}}}\left[||G_{\mathcal{S}}(G_{\mathcal{R}}(s)) - s||_1\right].\end{aligned}$$

C. Semantic Loss

Since our translation domains of interest are between synthetic images and real-world sensor images, we take advantage of the fact that many recent robotic simulators provide ground truth semantic labels and add a semantic constraint inspired by *CyCADA* [12]. (For simplicity in the following we use *CyCADA* to refer to *CycleGAN* plus this semantic loss instead of the full *CyCADA* approach [12]).

Assuming that for images from domain $\mathcal{S}$, the ground truth semantic labels $\mathbf{Y}$ are available, a semantic segmentation network $f_{\mathcal{S}}$ can be obtained by minimizing the *cross-entropy* loss $\mathbb{E}_{s\sim\mathcal{S}}[\text{CrossEnt}(\mathbf{Y}_s, f_{\mathcal{S}}(s))]$. We further assume that the ground truth semantic for domain $\mathcal{R}$ is lacking (which is the case for most real scenarios), meaning that $f_{\mathcal{R}}$ is not easily obtainable. In this case, we use $f_{\mathcal{S}}$ to generate "semi" semantic labels for domain $\mathcal{R}$. Then semantically consistent image translation can be achieved by minimizing the following losses, which imposes consistency between the semantic maps of the input and that of the generated output:

$$\begin{aligned}\mathcal{L}_{\text{sem}_{\mathcal{R}}}(G_{\mathcal{S}}; \mathcal{R}, f_{\mathcal{S}}) = \mathbb{E}_{p_{\text{real}}}[\text{CrossEnt}(f_{\mathcal{S}}(r), f_{\mathcal{S}}(G_{\mathcal{S}}(r)))]. \\ \mathcal{L}_{\text{sem}_{\mathcal{S}}}(G_{\mathcal{R}}; \mathcal{S}, f_{\mathcal{S}}) = \mathbb{E}_{p_{\text{sim}}}[\text{CrossEnt}(f_{\mathcal{S}}(s), f_{\mathcal{S}}(G_{\mathcal{R}}(s)))],\end{aligned}$$

D. Shift Loss for Consistent Generation

Different from the current literature of *domain adaptation*, our model is additionally expected to output consistent images for sequential inputs. Although with $\mathcal{L}_{\text{sem}}$, the semantics of the consecutive outputs are constrained, inconsistencies and artifacts still occur quite often. Moreover, in cases where ground truth semantics are unavailable from either domain, the sequential outputs are even less constrained, which could potentially lead to inconsistent policy outputs. Following the discussions in Sec. II-C, we introduce the *shift loss* to constrain the consistency even in these situations.

For an input image s, we use $s_{[x\to i,y\to j]}$ to denote the result of a shift operation: shifting s along the X axis by i pixels, and j pixels along the Y axis. We sometimes omit $y \to 0$ or $x \to 0$ in the subscript if the image is only shifted along the X or Y axis. According to [23], a trained stylization network is shift invariant to shifts of multiples of K pixels (K represents the total downsampling factor of the network), but can output significantly different stylizations otherwise. This causes the output of the trained network to change greatly for even very small changes in the input. We thus propose to add a simple yet direct and effective *shift loss* (u denotes uniform distribution):

$$\begin{aligned}\mathcal{L}_{\text{shift}_\mathcal{R}}(G_\mathcal{R};\mathcal{S}) &= \mathbb{E}_{p_{\text{sim}},\ i,j\sim u(1,K-1)}\\ &\quad\left[\left|\left|G_\mathcal{R}(s)_{[x\to i,y\to j]} - G_\mathcal{R}(s_{[x\to i,y\to j]})\right|\right|_2^2\right],\\ \mathcal{L}_{\text{shift}_\mathcal{S}}(G_\mathcal{S};\mathcal{R}) &= \mathbb{E}_{p_{\text{real}},\ i,j\sim u(1,K-1)}\\ &\quad\left[\left|\left|G_\mathcal{S}(r)_{[x\to i,y\to j]} - G_\mathcal{S}(r_{[x\to i,y\to j]})\right|\right|_2^2\right].\end{aligned}$$

Shift loss constrains the shifted output to match the output of the shifted input, regarding the shifts as image-scale movements. Assuming that only limited regional movement would appear in subsequent input frames, *shift loss* effectively smoothes the mapping function for small regional movements, restricting the changes in its outputs for subsequent inputs. This can be regarded as a cheap alternative for imposing consistency constraints on small movements, eliminating the need for the optical flow information, which is crucial for meeting the requirements of real-time robotics control.

E. Full Objective

Our full objective for learning *VR-Goggles* (Fig. 1) is (λ_{cyc}, λ_{sem} and λ_{shift} are the loss weightings):

$$\begin{aligned}&\mathcal{L}(G_\mathcal{R}, G_\mathcal{S}, D_\mathcal{R}, D_\mathcal{S}; \mathcal{S}, \mathcal{R}, f_\mathcal{S})\\ &\quad= \mathcal{L}_{\text{GAN}_\mathcal{R}}(G_\mathcal{R}, D_\mathcal{R}; \mathcal{S}, \mathcal{R}) + \mathcal{L}_{\text{GAN}_\mathcal{S}}(G_\mathcal{S}, D_\mathcal{S}; \mathcal{R}, \mathcal{S})\\ &\quad+ \lambda_{\text{cyc}}\left(\mathcal{L}_{\text{cyc}_\mathcal{R}}(G_\mathcal{S}, G_\mathcal{R}; \mathcal{R}) + \mathcal{L}_{\text{cyc}_\mathcal{S}}(G_\mathcal{R}, G_\mathcal{S}; \mathcal{S})\right)\\ &\quad+ \lambda_{\text{sem}}\left(\mathcal{L}_{\text{sem}_\mathcal{R}}(G_\mathcal{S}; \mathcal{R}, f_\mathcal{R}) + \mathcal{L}_{\text{sem}_\mathcal{S}}(G_\mathcal{R}; \mathcal{S}, f_\mathcal{S})\right)\\ &\quad+ \lambda_{\text{shift}}\left(\mathcal{L}_{\text{shift}_\mathcal{R}}(G_\mathcal{R}; \mathcal{S}) + \mathcal{L}_{\text{shift}_\mathcal{S}}(G_\mathcal{S}; \mathcal{R})\right).\end{aligned}$$

This corresponds to solving the following optimization:

$$G^*_\mathcal{R}, G^*_\mathcal{S} = \arg\min_{G_\mathcal{R}, G_\mathcal{S}} \max_{D_\mathcal{R}, D_\mathcal{S}} \mathcal{L}(G_\mathcal{R}, G_\mathcal{S}, D_\mathcal{R}, D_\mathcal{S}).$$

IV. Experiments

A. Validating Shift Loss: Artistic Style Transfer for Videos

To evaluate our method, we firstly conduct experiments for *artistic style transfer* for videos, to validate the effectiveness of *shift loss* on constraining consistency for sequential frames. We collect a training dataset of 98 HD video footage sequences (from *VIDEVO*[1] containing 2450 frames in total); the *Sintel* [25] sequences are used for testing, as their ground-truth optical flow is available. We compare the performance of the models trained under the following setups: (1) ***FF*** [22]: Canonical feed forward style transfer trained on single frames; (2) ***FF+flow*** [24]: *FF* trained on sequential images, with optical flow added for imposing temporal constraints on subsequent frames; (3) ***Ours***: *FF* trained on single frames, with an additional *shift loss* as discussed in Sec. III-D.

As a proof of concept, we begin our evaluation by comparing the three setups on their ability to generate shift invariant stylizations for shifted single frames. In particular, for each image s in the testing dataset, we generate 4 more test images by shifting the original image along the X axis by $1, 2, 3, 4$ pixels respectively, and pass all 5 frames (s, $s_{[x\to 1]}$, $s_{[x\to 2]}$, $s_{[x\to 3]}$, $s_{[x\to 4]}$) through the trained network to examine the consistency of the generated images. The results shown in Fig. 2 validate the discussion from [23], since the stylizations for s and $s_{[x\to 4]}$ from *FF* are almost identical ($K = 4$ for the trained network), but differ substantially otherwise. *FF-flow* improves the invariance by a limited amount; *Ours* is capable of generating consistent stylizations for shifted inputs, with the *shift loss* directly reducing the shift variance.

We then evaluate the consistency of stylized sequential frames, computing the temporal loss [24] using the ground truth optical flow for the *Sintel* sequences (Table I). Although the temporal loss is part of the optimization objective of *FF-flow*, and our method does not have access to any optical flow information, *Ours* is still able to achieve lower temporal loss with the *shift loss* constraint.

We further visualize the consistency comparison in Fig. 3, where we show the *temporal error maps*, the same metric as in [24], of two stylized consecutive frames for each method. The error increases linearly as shown from black to white in grayscale. *Ours* (bottom row) achieves the highest temporal consistency. Further details about style transfer training and the calculation of *temporal error map* are available in the supplement file [26].

B. Quantitative Evaluation: Carla Benchmark

Secondly, we conduct a quantitative evaluation of our proposed *real-to-sim* policy transfer pipeline. Since there are no publicly available common benchmarks for real-world autonomous driving evaluation, we test our pipeline in the *Carla* simulator following its benchmark setup [27], [28]. We choose the imitation learning pipeline because the reinforcement learning policy in [27] performs substantially worse. In [27], the expert datasets for *Carla* benchmark are collected under 4 different weather conditions (*daytime*, *daytime after rain*, *daytime hard rain* and *clear sunset*), and the policy is tested on benchmark tasks under *cloudy daytime* and *soft rain at sunset*. Since the datasets under the testing benchmark conditions are not available[2] for us to conduct domain adaptation, we split the provided training datasets into three training conditions (*daytime*, *daytime after rain*, *clear sunset*) and one testing condition (*daytime hard rain*) as shown in Fig. 4.

We present comparisons for both phases in the policy transfer pipeline: *policy training* and *domain adaptation*.

For the *policy training* phase, we adopt the following training regimes: (1) ***Single-Domain***: We train one policy

[1]http://www.videvo.net

[2]https://github.com/carla-simulator/imitation-learning

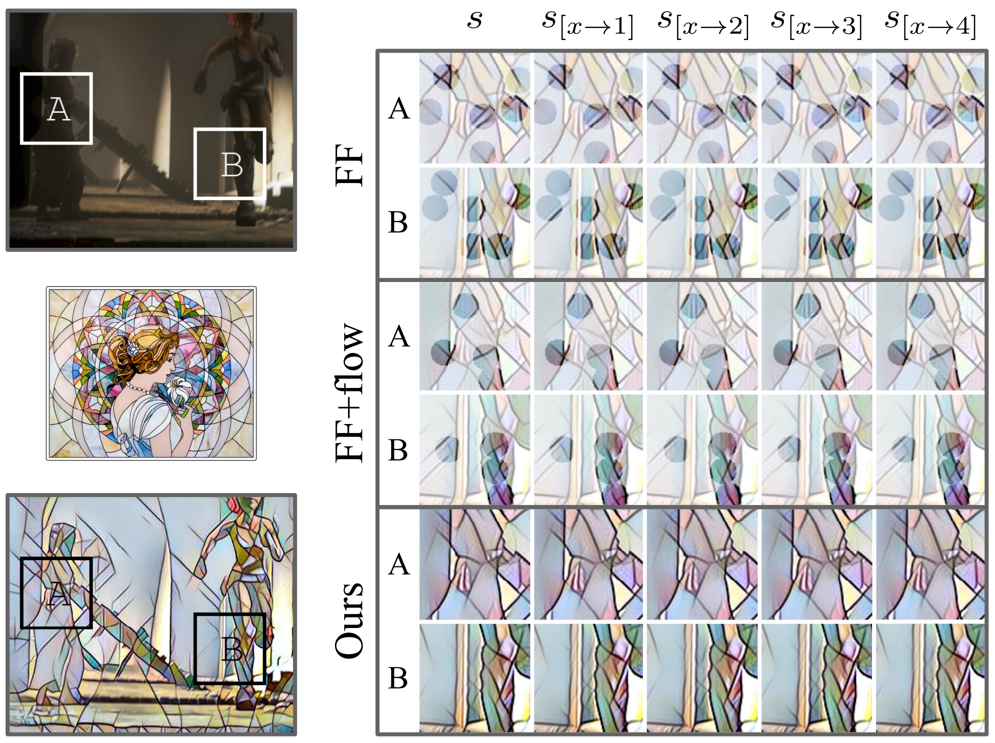


Fig. 2: Shift-invariance evaluation, comparing between *FF*, *FF+flow* and *Ours*. We shift an input image s along the X axis by $1, 2, 3, 4$ pixels respectively and feed all 5 frames through the networks trained via *FF*, *FF+flow* and *Ours* and show the generated stylizations. We mark the most visible differences with small circles and dim the rest of the generated images. As is discussed in [23], *FF* generates almost identical stylizations for s and $s_{[x\to 4]}$ (because 4 is a multiple of the total downsampling factor of the trained network), but those for $s_{[x\to 1]}$,$s_{[x\to 2]}$,$s_{[x\to 3]}$ differ significantly. *FF+flow* improves the shift-invariance, but we suspect the improvement is due to the inexplicit consistency constraint on regional shifts imposed by optical flow. *Ours* is able to generate shift-invariant stylizations with the proposed *shift loss*.

	FF	FF+flow	Ours
		mosaic	
ambush5	0.152	0.130	**0.127**
bamboo1	0.119	0.093	**0.086**
market6	0.132	0.110	**0.108**
temple2	0.127	0.104	**0.098**
sleeping2	0.115	0.089	**0.083**
shaman3	0.124	0.095	**0.087**
alley2	0.122	0.096	**0.090**
bamboo2	0.113	0.091	**0.089**
alley1	0.113	0.085	**0.078**
sleeping1	0.125	0.099	**0.092**
		lamuse	
ambush5	0.154	0.131	**0.130**
bamboo1	0.123	0.097	**0.090**
market6	0.135	**0.112**	**0.112**
temple2	0.138	0.108	**0.107**
sleeping2	0.121	0.100	**0.092**
shaman3	0.132	0.106	**0.094**
alley2	0.129	0.104	**0.094**
bamboo2	0.114	**0.090**	0.091
alley1	0.127	0.096	**0.083**
sleeping1	0.132	0.102	**0.101**

TABLE I: Comparing temporal loss between *FF*, *FF+flow* and *Ours*. *FF+flow* directly optimizes on this metric, while optical flow is never provided to *Ours*; yet *Ours* achieves lower temporal loss on the evaluated *Sintel* sequences.

under each of the three training weather conditions; (2) ***Multi-Domain***: A policy is trained under a combined dataset containing all three training weather conditions. We note that since the imitation policy is trained with datasets instead of interacting with the simulation environment, the full approach of *Domain Randomization* [8], [9] could not be directly applied, as it requires to randomize the textures of each object, lighting conditions and viewing angles of the rendered scenes. Thus the *Multi-Domain* can be considered as a relatively limited realization of the *Domain Randomization* approach in the *Carla* benchmark dataset setup. As for the progressive nets approach [10], it requires a finetuning phase of the policy in the real world, which for autonomous driving means that we need to deploy the trained policy onto a real car and finetune it through rather expensive real-world interactions. Thus we do not consider this approach in this evaluation. (An additional comparison experiment with the progressive nets can be found in the supplementary materials [26].)

For the *domain adaptation* phase, we compare the following adaptation methods: (1) ***No-Goggles***: Feed the testing data directly to the trained policy; (2) ***CycleGAN*** [11]: Use *CycleGAN* to translate the test data to the training domain before feeding to policy nets and (3) ***Ours***: Add *shift loss* on top of (2) as ***VR-Goggles*** to translate the inputs. For both *CycleGAN* and *VR-Goggles*, we train an adaptation network from the testing weather condition to each of the three training conditions. (For more details about the training of the policy and adaptation models, please refer to the supplementary materials [26].)

The four benchmark tasks (*Straight*, *One Turn*, *Navigation* and *Nav. dynamic*) are in order of increasing difficulty and each of them consists of 25 different preset trajectories. Since the *Multi-Domain* policy is trained with three weather conditions instead of four as in the original setup [27] due to the reason discussed earlier, directly deploying the *Multi-Domain* policy fail to finish any of the two harder tasks under the relatively extreme testing weather condition. For the different adaptation strategies, our *VR-Goggles* outperforms *CycleGAN* on almost all of the metrics, especially the two harder tasks (*Navigation* and *Nav. dynamic*) in terms of both the success rate and the average percentage of distance to goal traveled. The average distance traveled between two infractions is reported only for the hardest task [27]: navigating in the presence of dynamic objects (*Nav. dynamic*). The adaptation models of *Ours* enable the agents to drive safely with mostly lower infraction frequencies compared with *CycleGAN*. *CycleGAN* collides with pedestrians less often with the *Multi-Domain*

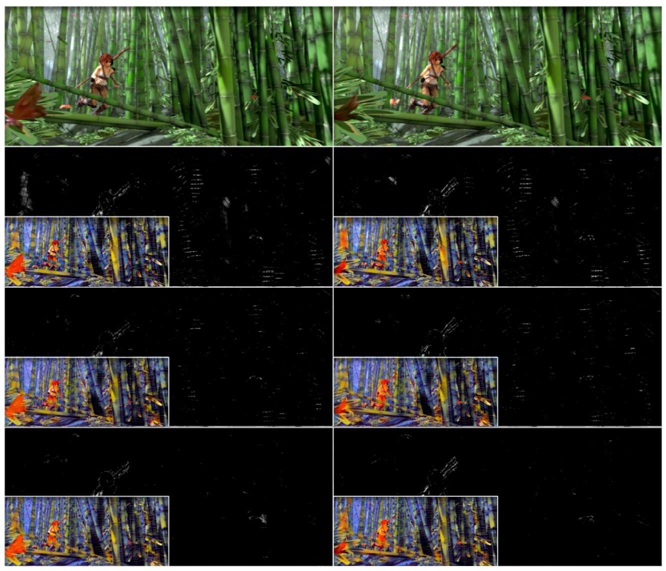
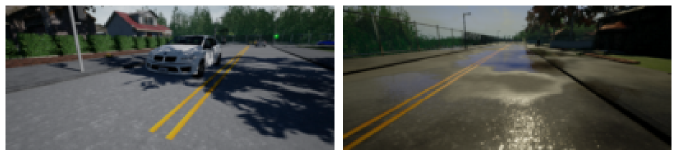


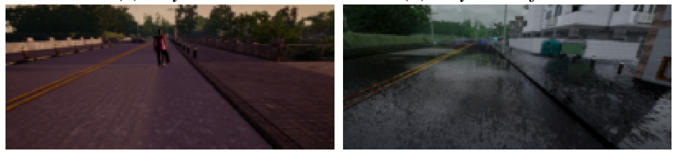
Fig. 3: *Temporal error maps* between generated stylizations for subsequent input frames. The error increases linearly as shown from black to white in grayscale. $1st$ *row*: input frames; $2nd \sim 4th$ *row*: temporal error maps (with the corresponding stylizations shown on top) of outputs from *FF*, *FF+flow*, and *Ours*. We here choose a very challenging style (*mosaic*) for temporal consistency, as it contains many fine details, with tiny tiles laid over the original image in the final stylizations. Yet, *Ours* achieves very high consistency.

policy. A probable explanation is that most episodes under this setup end due to collision with cars and static obstacles, so there does not occur too many challenging pedestrian conditions. For example, for direct deployment without adaptation (*No-Goggles*), the average distance between collisions with pedestrians is higher than 4.6 km, because the total navigation distance for all 25 episodes in this task is only 4.6 km which is too short to encounter pedestrians.

We note that the transfer pipelines of *Single-Domain* policies behave much better than directly deploying the *Multi-Domain* policy, and the training time of the former policy is also much shorter than that of the latter [26].



(a) *daytime* (b) *daytime after rain*



(c) *clear sunset* (d) *daytime hard rain*

Fig. 4: *Carla* weather conditions used in benchmarking: three training conditions (a), (b), (c) and one testing condition (d).

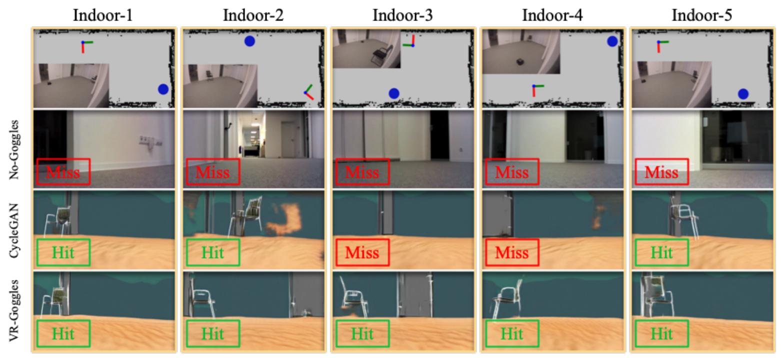
C. Real-world Indoor & Outdoor Navigation

Finally, we conduct real-world robotics experiments for both indoor and outdoor visual navigation tasks. We begin by training learning-based visual navigation policies, taking simulated first-person-view images as inputs, outputting moving commands for specific navigation targets. Then, we deploy the trained policy onto real robots, comparing the following *domain adaptation* approaches: (1) ***No-Goggles***: Feed the sensor readings directly to the trained policy; (2) ***CycleGAN/CyCADA*** [11], [12]: Use *CycleGAN* (when semantic ground truth is not available) / *CyCADA* (when ground truth semantic maps are provided by the simulator) to translate the real sensory inputs to the synthetic domain before feeding to the policy nets; (3) ***Ours***: Add *shift loss* on top of (2) as the ***VR-Goggles***.

For ***indoor*** office experiments, we build an office environment in *Gazebo* [29] and render $s \sim p_{\text{sim}}$ from this simulation environment (Fig. 6a). We capture $r \sim p_{\text{real}}$ from a real office (Fig. 6b) using a *RealSense R200 camera* mounted on a *Turtlebot3 Waffle*. For conducting the *domain adaptation*,

		Training		Testing					
		Single-Domain	Multi-Domain	Single-Domain			Multi-Domain		
				No-Gog.	CycleGAN	VR-Gog.	No-Gog.	CycleGAN	VR-Gog.
Success rate (%)	Straight	81.3	97.3	13.3	**93.3**	90.7	64.0	96.0	**100**
	One turn	64.0	85.3	1.3	**54.7**	**54.7**	36.0	61.3	**76.0**
	Navigation	60.0	84.0	0.0	21.3	**45.3**	0.0	42.7	**61.3**
	Nav. dynamic	58.7	74.7	0.0	21.3	**32.0**	0.0	34.7	**56.0**
Ave. distance to goal travelled (%)	Straight	89.7	96.5	37.8	**95.8**	94.7	83.6	95.9	**98.3**
	One turn	73.6	71.1	18.4	36.3	**48.4**	24.7	52.0	**71.3**
	Navigation	68.6	88.8	7.3	36.7	**51.0**	7.4	60.8	**73.1**
	Nav. dynamic	68.2	80.7	5.0	32.6	**51.3**	5.2	55.7	**72.2**
Ave. distance travelled between two infractions in Nav. dynamic (km)	Opposite lane	2.83	2.55	0.23	**0.77**	0.72	0.26	0.83	**2.22**
	Sidewalk	6.47	9.70	0.21	1.15	**2.62**	0.38	1.29	**2.46**
	Collision-static	2.38	3.03	0.14	0.52	**0.87**	0.16	0.77	**1.26**
	Collision-car	2.06	1.03	0.29	1.01	**1.40**	0.27	0.59	**0.77**
	Collision-pedestrian	15.10	16.17	2.17	4.03	**6.98**	>4.60	**7.29**	4.67

TABLE II: Quantitative evaluation of goal-directed *Carla* navigation benchmark tasks [27]. We train imitation policies under single weather condition (*Single-Domain*) and three training weather conditions (*Multi-Domain*). Policies are evaluated in testing weather condition through direct deploying (*No-Goggles*), translating the input image through *CycleGAN* and through *VR-Goggles* (for transferring the *Multi-Domain* policy with *CycleGAN* and *VR-Goggles* we train one adaptation network from the testing weather condition to each of the three training weather conditions and report the average results under those three adaptations). Higher is better.



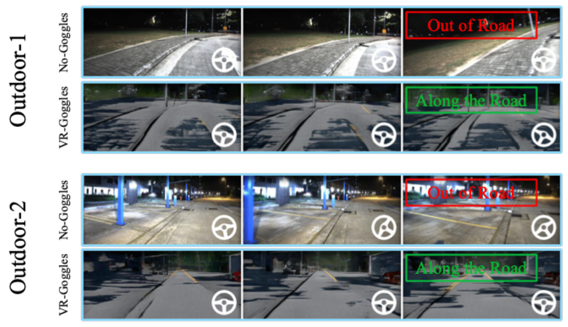
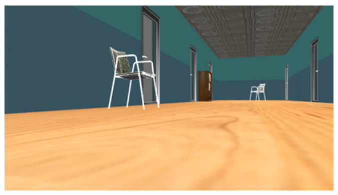
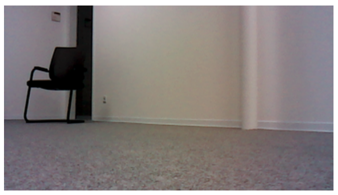
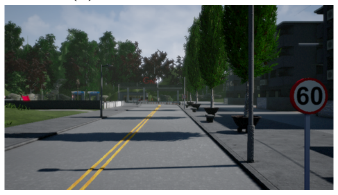


Fig. 5: Real-world visual control experiments. ***Indoor*** (yellow):. A navigation policy is firstly trained in a simulated environment (Fig. 6a) that is able to navigate to chairs based on visual inputs. Without retraining or finetuning, our proposed *VR-Goggles* enables the mobile robot to directly deploy this policy in a real office environment (Fig. 6b), achieving 100% success rate in a set of real-world experiments. Here *Miss* refers to test runs where the agent stays put or rotate in place and simply ignores the chair even when they are in sight as the policy trained in the simulation could not cope with the drastically visually different inputs (*No-Goggles*), or due to the inconsistency of the translated subsequent outputs which hinders the successful fulfilment of the goal-reaching task (*CycleGAN*). *Hit* refers to frames where the agent captures the chair in sight and outputs commands to move towards it. ***Outdoor*** (cyan): An autonomous driving policy (via conditional imitation learning [28]) is trained in *Carla* daytime (Fig. 6c), a *VR-Goggles* model is trained to translate between *Carla* daytime and *Robotcar* nighttime (Fig. 6d), which enables the real-world nighttime deployment of the trained policy.


(a) simulated indoor


(b) real indoor


(c) simulated outdoor

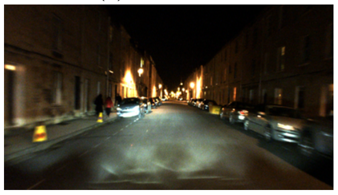

(d) real outdoor

Fig. 6: Samples from the simulated environment (left) and the real world (right) used in our indoor (top) and outdoor (bottom) navigation experiments.

as the simulator (*Gazebo*) does not provide ground truth semantics, we drop the semantic constraint $\mathcal{L}_{\text{sem}}$. The input images are of size 640×360 and the adaptation network is trained with 256×256 crops. We use the same network architecture as in *CycleGAN*, and train for 50 epochs with a learning rate of $2e-4$ as we observe no performance gain training for longer iterations.

We train the navigation policy using Canonical A3C with 8 parallel workers [2] in *Gazebo*, and deploy the trained policy onto *Turtlebot3 Waffle* and compare the three *domain adaptation* approaches (Fig. 5). Without *domain adaptation*, *No-Goggles* fails completely in the real-world tasks; our proposed *VR-Goggles* achieves the highest success rate (0%, 60% and 100% for *No-Goggles*, *CycleGAN* and *Ours* respectively) due to the quality and consistency of the translated streams. The control cycle runs in real-time at 13Hz on a *Nvidia TX2*.

Finally, we conduct ***outdoor*** autonomous driving experiments (we sample $s \sim p_{\text{sim}}$ from the *Carla* daytime [27] environment Fig. 6c and sample $r \sim p_{\text{real}}$ from a nighttime dataset of *Robotcar* [30] Fig. 6d) with input images of size 640×400. Considering that *VR-Goggles* outperforms *CycleGAN* in *indoor* experiments, and since outdoor robotics experiments are relateively expensive, we only compare *No-Goggles* and *VR-Goggles* in the *outdoor* autonomous driving scenario. We take the driving policy trained through conditional imitation learning [28] as in Section IV-B. This policy takes as inputs the first person view RGB image and a high-level command, which falls in a discrete action space and is generated through a global planner (*straight, left, right, follow, none*). In our real-world experiments, this high-level direction command is set as *straight*, indicating the vehicle (a *Bulldog* with a *PointGrey Blackfly* camera mounted on it) to always go along the road. The control policy outputs the steering angle.

The control policy is trained purely in *Carla* simulated daytime, while it is tested in a nighttime town street scene (Fig. 5). It is non-trivial to quantitatively evaluate the control policy in the real world, so we show two representative sequences marked with the output steering commands. The top row of each sequence shows the continuous outputs of *No-Goggles*. Due to the huge difference between the real nighttime and the simulated daytime, the vehicle failed to move along the road. Our *VR-Goggles*, however, successfully guides the vehicle along the road as instructed by the global planner (the policy prefers to turn right since it is trained in a right-driving environment) [3].

[3]A video demonstrating our approach and much more experimental results are available at https://sites.google.com/view/zhang-tai-19ral-vrg/home, where we also show that the *VR-Goggles* can easily train a new model for a new type of chair without finetuning the indoor control policy.

V. Conclusions

In this paper, we tackle the *reality gap* occurring when deploying learning-based visual control policies trained in simulation to the real world, by translating the real images back to the synthetic domain during deployment. Due to the sequential nature of the incoming sensor streams for control tasks, we propose *shift loss* to increase the consistency of the translated subsequent frames, and validate it both in *artistic style transfer for videos* and *domain adaptation*. We verify our proposed *VR-Goggles* pipeline as a lightweight, flexible and efficient solution for visual control through *Carla* benchmark as well as a set of real-world robotics experiments. It would be interesting to apply our method to manipulation, as this paper has been mainly focused on navigation. Also, evaluating our method in more challenging environments on more sophisticated control tasks could be another future direction.

Acknowledgment

The authors would like to thank Christian Dornhege and Daniel Büscher for the discussion of the initial idea.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, p. 529, 2015.

[2] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, “Asynchronous methods for deep reinforcement learning,” in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1928–1937.

[3] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” *arXiv preprint arXiv:1509.02971*, 2015.

[4] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 1889–1897.

[5] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.

[6] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, *et al.*, “Using simulation and domain adaptation to improve efficiency of deep robotic grasping,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4243–4250.

[7] G. J. Stein and N. Roy, “Genesis-rt: Generating synthetic images for training secondary real-world tasks,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7151–7158.

[8] F. Sadeghi and S. Levine, “Cad2rl: Real single-image flight without a single real image,” in *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017.

[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,” in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 23–30.

[10] A. A. Rusu, M. Večerík, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, “Sim-to-real robot learning from pixels with progressive nets,” in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 262–270.

[11] J. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 2242–2251.

[12] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, “CyCADA: Cycle-consistent adversarial domain adaptation,” in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. Stockholmsmässan, Stockholm Sweden: PMLR, 10–15 Jul 2018, pp. 1989–1998.

[13] A. Cherian and A. Sullivan, “Sem-gan: Semantically-consistent image-to-image translation,” *arXiv preprint arXiv:1807.04409*, 2018.

[14] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, “Asymmetric actor critic for image-based robot learning,” in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[15] L. Tai, G. Paolo, and M. Liu, “Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 31–36.

[16] J. Zhang, L. Tai, J. Boedecker, W. Burgard, and M. Liu, “Neural SLAM: Learning to explore with external memory,” *arXiv preprint arXiv:1706.09520*, 2017.

[17] O. Zhelo, J. Zhang, L. Tai, M. Liu, and W. Burgard, “Curiosity-driven exploration for mapless navigation with deep reinforcement learning,” *arXiv preprint arXiv:1804.00456*, 2018.

[18] J. Zhang, J. T. Springenberg, J. Boedecker, and W. Burgard, “Deep reinforcement learning with successor features for navigation across similar environments,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 2371–2378.

[19] L. Tai, J. Zhang, M. Liu, and W. Burgard, “Socially-compliant navigation through raw depth inputs with generative adversarial imitation learning,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1111–1117.

[20] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, “Target-driven visual navigation in indoor scenes using deep reinforcement learning,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 3357–3364.

[21] T. Inoue, S. Choudhury, G. De Magistris, and S. Dasgupta, “Transfer learning from synthetic to real images using variational autoencoders for precise position detection,” in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 2725–2729.

[22] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in *European Conference on Computer Vision*. Springer, 2016, pp. 694–711.

[23] M. Ruder, A. Dosovitskiy, and T. Brox, “Artistic style transfer for videos and spherical images,” *International Journal of Computer Vision*, vol. 126, no. 11, pp. 1199–1219, Nov 2018. [Online]. Available: https://doi.org/10.1007/s11263-018-1089-z

[24] H. Huang, H. Wang, W. Luo, L. Ma, W. Jiang, X. Zhu, Z. Li, and W. Liu, “Real-time neural style transfer for videos,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 7044–7052.

[25] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, “A naturalistic open source movie for optical flow evaluation,” in *European Conference on Computer Vision*. Springer, 2012, pp. 611–625.

[26] J. Zhang, L. Tai, Y. Xiong, M. Liu, J. Boedecker, and W. Burgard, “Supplement file of VR-Goggles for robots: Real-to-sim domain adaptation for visual control,” Tech. Rep., 2018. [Online]. Available: https://ram-lab.com/file/tailei/vr_goggles/supplement.pdf

[27] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 1–16.

[28] F. Codevilla, M. Miiller, A. López, V. Koltun, and A. Dosovitskiy, “End-to-end driving via conditional imitation learning,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1–9.

[29] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, Sep. 2004, pp. 2149–2154 vol.3.

[30] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, “1 Year, 1000km: The Oxford RobotCar Dataset,” *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 1, pp. 3–15, 2017.