Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Virtual-to-Real: Learning to Control in Visual Semantic Segmentation

Zhang-Wei Hong, Yu-Ming Chen*, Hsuan-Kung Yang*, Shih-Yang Su*, Tzu-Yun Shann*, Yi-Hsiang Chang, Brian Hsi-Lin Ho, Chih-Chieh Tu, Tsu-Ching Hsiao, Hsin-Wei Hsiao, Sih-Pin Lai, Yueh-Chuan Chang, Chun-Yi Lee
Elsa Lab, Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan

Abstract

Collecting training data from the physical world is usually time-consuming and even dangerous for fragile robots, and thus, recent advances in robot learning advocate the use of simulators as the training platform. Unfortunately, the reality gap between synthetic and real visual data prohibits direct migration of the models trained in virtual worlds to the real world. This paper proposes a modular architecture for tackling the virtual-to-real problem. The proposed architecture separates the learning model into a perception module and a control policy module, and uses semantic image segmentation as the meta representation for relating these two modules. The perception module translates each perceived RGB image to semantic image segmentation. The control policy module is implemented as a deep reinforcement learning agent, which performs actions based on the translated image segmentation. Our architecture is evaluated in an obstacle avoidance task and a target following task. Experimental results show that our architecture significantly outperforms all of the baseline methods in both virtual and real environments, and demonstrates a faster learning curve than them. We also present a detailed analysis for a variety of variant configurations, and validate the transferability of our modular architecture.

1 Introduction

Visual perception based control has been attracting attention in recent years for controlling robotic systems, as visual inputs contain rich information of the unstructured physical world. It is usually necessary for an autonomous robot to understand visual scene semantics to navigate to a specified destination. Interpreting and representing visual inputs to perform actions and interact with objects, however, are challenging for robots in unstructured environments as colored images are typically complex and noisy [Maier *et al.*, 2012; Biswas and Veloso, 2012]. It is especially difficult to design a rule-based robot satisfying such requirements.

Both modular and end-to-end learning-based approaches have been proven effective in a variety of vision-based robotic control tasks [Sadeghi and Levine, 2017; Finn and Levine, 2017; Gupta *et al.*, 2017; Smolyanskiy *et al.*, 2017; Zhu *et al.*, 2017]. A modular cognitive mapping and planning approach has been demonstrated successful in first-person visual navigation [Gupta *et al.*, 2017]. Vision-based reinforcement learning (RL) has been attempted to train an end-to-end control policy for searching specific targets [Zhu *et al.*, 2017]. Applying end-to-end supervised learning to navigate a drone along a trail with human-labeled image-action pairs is presented in [Smolyanskiy *et al.*, 2017]. An end-to-end training methodology for object manipulation tasks using unlabeled video data is described in [Finn and Levine, 2017]. While these learning-based approaches seem attractive, they typically require a huge amount of training data. Collecting training data for learning a control policy in the physical world is usually costly and poses a number of challenges. First, preparing large amounts of labeled data for supervised learning takes considerable time and human efforts. Second, RL relies on trial-and-error experiences, which restrict fragile robots from dangerous tasks. Online training and fine-tuning robots in the physical world also tend to be time-consuming, limiting the learning efficiency of various RL algorithms.

An alternative approach to accelerate the learning efficiency and reduce the cost is training robots in virtual worlds. Most of the recent works on robot learning collect training data from simulators [James and Johns, 2016; Rusu *et al.*, 2016; Sadeghi and Levine, 2017; Peng *et al.*, 2018; Tobin *et al.*, 2017; Zhu *et al.*, 2017]. However, the discrepancies between virtual and real worlds prohibit an agent trained in a virtual world from being transferred to the physical world directly [James and Johns, 2016]. Images rendered by low-fidelity simulators are unlikely to contain as much rich information as real ones. Therefore, bridging the reality gap [Tobin *et al.*, 2017] has been a challenging problem both in computer vision and robotics. Many research efforts have been devoted to tackling this problem by either domain adaption (DA) [Rusu *et al.*, 2016; Ghadirzadeh *et al.*, 2017; Zhang *et al.*, 2017] or domain randomization (DR) [Sadeghi and Levine, 2017; Peng *et al.*, 2018; Tobin *et al.*, 2017; Zhang *et al.*, 2017]. Both of these methods train agents by simulators. DA fine-tunes simulator-trained models with real-world data. DR, on the other hand, trains agents with ran-

* indicates equal contribution.

domized object textures, colors, lighting conditions, or even physical dynamics in virtual worlds. The diversified simulated data enable the agents to generalize their models to the real world. Unfortunately, collecting the real-world data required by DA for control policy learning is exceptionally time-consuming. Although DR does not require real-world data, the technique lacks a systematic way to determine which parameters are required to be randomized. Furthermore, DR requires considerably large number of training samples in the training phase to achieve acceptable performance.

Instead of training a vision-based control policy in an end-to-end fashion, we propose a new modular architecture to address the reality gap in vision domain. We focus on the problem of transferring deep neural network (DNN) models trained in simulated virtual worlds to the real world for vision-based robotic control. We propose to separate the learning model into a perception module and a control policy module, and use semantic image segmentation as the meta-state for relating these two modules. These two modules are trained separately and independently. Each of them assumes no prior knowledge of the other module. The perception module translates RGB images to semantic image segmentations. This module can be any semantic image segmentation models pre-trained on either commonly available datasets [Cordts *et al.*, 2016; Zhou *et al.*, 2017] or synthetic images generated by simulators [Richter *et al.*, 2017]. As annotated datasets for semantic segmentation are widely available nowadays and cross dataset domain adaptation has been addressed [Chen *et al.*, 2017], training and fine-tuning a perception module for segmenting outdoor [Cordts *et al.*, 2016] or indoor [Zhou *et al.*, 2017] scenarios have become straightforward. The control policy module employs deep RL methods and takes semantic image segmentation as its input. In the training phase, the control policy module only receives the image segmentation rendered by a low-fidelity simulators. The RL agent interacts with the simulated environments and collects training data for the control policy module. While in the execution phase, the control policy module receives image segmentation from the perception module, enabling the RL agent to interact with the real world. As the image segmentations rendered by the simulator and those generated by the perception module are invariant [Pan *et al.*, 2017], the control policy learned from the simulator can be transferred to the real world directly. The proposed architecture provides better efficiency than the conventional learning-based methods, as simulators are able to render image segmentations at a higher frame rate, enabling the RL agent to collect training or trial-and-error experiences faster. It also allows the RL agent to learn faster, as semantic image segmentation contains less noise than raw images. We believe that our methodology is more general and applicable to different scenarios and tasks than the method proposed in [Yan *et al.*, 2017], which uses a binary segmentation mask to bridge the visual and control modules. Using binary segmentation can result in a loss of semantic information, which are essential in a number of complex tasks. On the other hand, our proposed methodology not only preserves semantic information, but is also generalizable to complex scenarios.

Another advantage of the proposed architectures is its modularity. Both the perception and control policy modules can be flexibly replaced in a plug-and-play fashion, as long as their meta-state representation (i.e., image segmentation) formats are aligned with each other. Replacing the perception module allows a pre-trained control policy module to be applied to different scenarios. Replacing the control policy module, on the other hand, changes the policy of the robot in the same environment. Furthermore, our architecture allows another visual-guidance module to be incorporated. The visual-guidance module adjusts the meta-state representation, such that the behavior of the robot can be altered online without replacing either the perception or control policy modules.

To demonstrate the effectiveness of the proposed modular architecture, we evaluate it against a number of baseline methods in two benchmark tasks: obstacle avoidance and target following. These tasks are essential for autonomous robots, as the former requires collision-free navigation, while the latter requires understanding of high-level semantics in the environments. To validate the transferability of our architecture, we train our models and the baseline methods in our simulator, and compare their performance in the benchmark tasks in both virtual and real worlds. We focus on RL-based robotic control policy, though the control policy module can be implemented by various options (e.g., imitation learning). Our results show that the proposed architecture outperforms all the baseline methods in all the benchmark tasks. Moreover, it enables our RL agent to learn dramatically faster than all the baseline methods. By simply replacing the perception module, we also show that no additional fine-tuning is necessary for the control policy module when migrating the operating environment. We further demonstrate the versatility of the visual-guidance module in Section 5. The contributions of this work are summarized as follows:

1. A straightforward, easy to implement, and effective solution (which has not yet been proposed in the literature) to the challenging virtual-to-real transferring problem.
2. A new modular learning-based architecture which separates the vision-based robotic learning model into a perception module and a control policy module.
3. A novel concept for bridging the reality gap via the use of semantic image segmentation, which serves as the meta-state representation for relating the two modules.
4. A way to migrate the operating environment of a robot to another one without further fine-tuning its control policy module.
5. A visual-guidance module for altering the behavior of a robot via adjusting the meta-state representation.

The remainder of this paper is organized as follows. Section 2 introduces background material. Section 3 walks through the proposed modular architecture. Section 4 describes the evaluation setup, tasks, and scenarios. Section 5 presents the experimental results. Section 6 concludes.

2 Background

This section briefly reviews background material on semantic image segmentation, RL, and policy gradient methods.

2.1 Semantic Image Segmentation

The goal of semantic segmentation is to perform dense predictions at pixel level. It has received great attention due to its applicability to a number of research domains (e.g., visual understanding, autonomous driving, and robotic control). Fully convolutional network (FCN) [Long *et al.*, 2015] pioneered the replacement of fully-connected (FC) layers by convolutional layers. A number of successive works have further enhanced the accuracy and efficiency [Chen *et al.*, 2016; Zhao *et al.*, 2017a; Zhao *et al.*, 2017b; Paszke *et al.*, 2016; He *et al.*, 2015], making them more promising for robotic control tasks. Unfortunately, relatively fewer attempts have been made to incorporate them into robotic learning models.

2.2 RL and Policy Gradient Methods

RL trains an agent to interact with an environment $\mathcal{E}$. An RL agent observes a state s from $\mathcal{E}$, takes an action a according to its policy $\pi(a|s)$, and receives a reward $r(s,a)$. $\mathcal{E}$ then transitions to a new state s'. The agent's objective is to maximize its accumulated rewards G_t with a discounted factor γ, expressed as $G_t = \sum_{\tau=t}^{T} \gamma^{\tau-t} r(s_\tau, a_\tau)$, where t is the current timestep, and T is the total number of timesteps. Policy gradient methods are RL approaches that directly optimize π in the direction: $\nabla_\pi \sum_{t=0}^{T} \log \pi(a_t|s_t)(G_t - b(s_t))$, where $b(s_t)$ is a baseline function. A common choice for $b(s_t)$ is the value function $V^\pi(s) = \mathbb{E}[G_t|s_t = s, \pi]$. This approach is known as the actor-critic algorithm [Williams, 1992]. An asynchronous variant of the actor-critic algorithm, namely asynchronous advantage actor-critic (A3C) [Mnih *et al.*, 2016], has been proven data-efficient and suitable for vision-based control tasks [Wu and Tian, 2017; Parisotto and Salakhutdinov, 2018; Pan *et al.*, 2017; Zhu *et al.*, 2017].

3 Proposed Methodology

In this section, we present the modular architecture of our model. We first provide an overview of the model architecture, followed by the implementation details of the modules. Fig. 1 illustrates the proposed modular architecture. It consists of a perception module, a control policy module, and a visual guidance module. Semantic image segmentation s_t serves as the meta-state representation for relating the former two modules, as shown in Fig. 1 (a). The perception module generates s_t from an RGB input image x_t, which comes from different sources in the training (x_t^{syn}) and execution (x_t^{real}) phases. The control policy module takes s_t as its input, and reacts with a_t according to π. The visual guidance module enables high-level planning by modifying s_t, as shown in Fig. 1 (b).

3.1 Perception Module

The main function of the perception module is to generate s_t, and passes it to the control policy module. In the training stage, s_t is rendered by a segmentation shader from a synthetic image x_t^{syn}. While in the execution phase, the perception module can be any semantic segmentation model. Its function is expressed as $s_t = \phi^{percept}(x_t^{real}; \theta^{percept})$, where $\phi^{percept}$ represents the semantic segmentation model, x_t^{real}

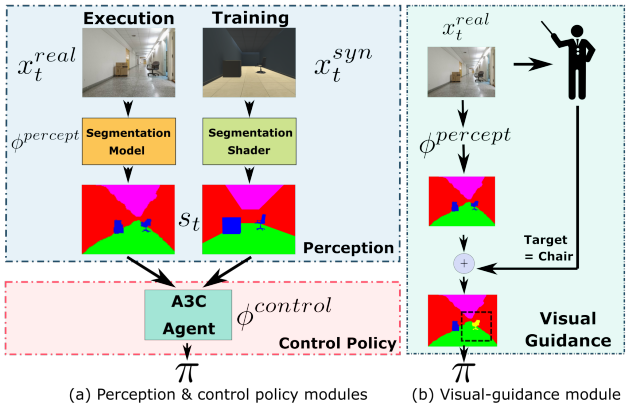


Figure 1: Model architecture

is the RGB image from the real world, and $\theta^{percept}$ denotes the parameters of the perception module. We directly employ existing models, such as DeepLab [Chen *et al.*, 2016] and ICNet [Zhao *et al.*, 2017a], and pre-train them on datasets ADE20K [Zhou *et al.*, 2017] and Cityscape [Cordts *et al.*, 2016] for indoor and outdoor scenarios, respectively. The format of meta-state representation s_t is the same for both phases. The effect of using different semantic image segmentation models as the perception module is analyzed in Section 5.

The benefit of using semantic segmentation as the meta-state representation is threefold. First, although the visual appearances of x_t^{real} and x_t^{syn} differ, their semantic segmentations are almost identical [Pan *et al.*, 2017] (Fig. 1 (a)). This allows us to use semantic segmentation to bridge the reality gap. Second, semantic segmentation preserves only the most important information of objects in an RGB image, resulting in a more structured representation. It also provides much richer information than other sensory modalities, as they typically fail to cover crucial high-level scene semantics. Third, the perception module in the execution phase only requires a monocular camera, which is a lot cheaper than depth cameras or laser rangefinders. As a result, semantic segmentation is inherently an ideal option for meta-state representation.

3.2 Control Policy Module

The control policy module is responsible for finding the best control policy π for a specific task. In this work, it employs the A3C algorithm, and takes s_t as its inputs (Fig. 1 (a)). In the training phase, the model $\phi^{control}$ is trained completely in simulated environments, and learns π directly from s_t rendered by a low-fidelity simulator. While in the execution phase, it receives s_t from the perception module, allowing the RL agent to interact with the real world. As semantic segmentation is used as the meta-state representation, it enables virtual-to-real transferring of π without fine-tuning or DR.

3.3 Visual Guidance Module

The use of semantic segmentation as the meta-state representation gives the proposed architecture extra flexibility. Our modular architecture allows a visual-guidance module to be

Action	(Linear (m/s), Angular (rad/s))
Forward	$(0.2, 0.0)$
TurnLeft	$(0.2, -0.2)$
TurnRight	$(0.2, 0.2)$

Table 1: Mapping actions to robot velocities.

augmented to guide the control policy module to perform even more complex tasks by manipulating the meta-state s_t. The manipulation can be carried out by either human beings or other RL agents. One can easily modify a robot's task through manipulating the class labels in s_t. This implies that a roadway navigation robot can be transformed to a sidewalk navigation robot by swapping the labels of the roadway and the sidewalk in s_t. The visual guidance module can also alter a target following robot's objective online by modifying the target label to a new one, as shown in Fig. 1 (b) (yellow chair). Note that visual guidance does not require any retraining, fine-tuning, or extra data in the above scenarios, while the previous works [Gupta *et al.*, 2017; Zhu *et al.*, 2017] demand additional training samples to achieve the same level of flexibility.

4 Virtual-to-Real Evaluation Setup

In this section, we present the evaluation setup in detail. We evaluate the proposed architecture both in virtual and real environments. We begin with explaining the simulator used for training and evaluation, followed by an overview of the tasks and scenarios for performance evaluation. Finally, we discuss the baseline models and their settings.

Simulator. We use the Unity3D [1] engine to develop the virtual environments for training our RL agent. As Unity3D supports simulation of rigid body dynamics (e.g., collision, friction, etc.), it allows virtual objects to behave in a realistic way. In a simulated environment, our RL agent receives its observations in the form of semantic segmentation from a first-person perspective. It then determines the control action to take accordingly. The virtual environment responds to the action by transitioning to the next state and giving a reward signal. The geometric shape of our simulated agent is set to be similar to that of the real robot.

Tasks. The proposed model is evaluated against the baselines in virtual and real environments on two benchmark tasks: obstacle avoidance and target following. In both tasks, an RL agent starts at a random position in the environment at the beginning of each episode. At each timestep, it selects an action from the three possible options: moving forward, turning left, and turning right. Each action corresponds to the linear and angular velocities of the robot specified in Table 1.
(1) Obstacle Avoidance: The agent's goal is to navigate in a diverse set of scenarios, and avoid colliding with obstacles.
(2) Target Following: The agent's objective is to follow a moving target (e.g., human) while avoiding collisions.

[1] Unity3D: https://unity3d.com/

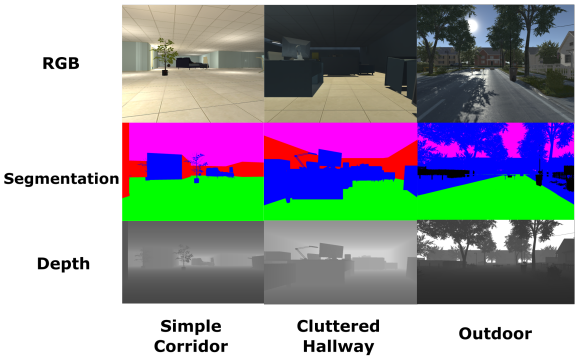


Figure 2: Samples of evaluation scenarios. From left to right: *Simple Corridor*, *Cluttered Hallway*, and *Outdoor*. From top to bottom: the corresponding RGB image, semantic segmentation, and depth map.

We validate the proposed methodology on these simple tasks, as we consider our work as a proof-of-concept of transferring deep reinforcement learning (DRL) agents from virtual worlds to the real world by the use of semantic image segmentation. To the best of our knowledge and a rigorous literature survey, our work is the first one to address the virtual-to-real control problem with semantic image segmentation as the meta-state representation. Hence, we focused our attention on the effectiveness and efficiency of the proposed methodology, and validated it on the obstacle avoidance and target following tasks.

Scenarios. We evaluate the models with the following three scenarios. Fig. 2 illustrates a few sample scenes of them.
(1) Simple Corridor: This scenario features indoor straight passages, sharp turns, static obstacles (e.g., chairs, boxes, tables, walls, etc.), and moving obstacles (e.g., human).
(2) Cluttered Hallway: This scenario features a hallway crammed with static and moving obstacles for evaluating an agent's capability of avoiding collisions in narrow space.
(3) Outdoor: This scenario features an outdoor roadway with sidewalks, buildings, terrain, as well as moving cars and pedestrians. This is used to evaluate how well the control policy trained for the tasks can be transfered from an indoor environment to an outdoor environment. Note that the agent is not allowed to move on the sidewalks in this scenario.

Models. For all the experiments, our model is trained with semantic segmentations generated from the simulator. We adopt DR, depth perception, and ResNet [He *et al.*, 2016] as the baseline models. For the DR baseline, the texture of each mesh in a scene is randomly chosen from 100 textures. We augment the 100 textures by randomly changing their color tones and lighting conditions, which results in over 10,000 visually different textures. This ensures that the DR baseline used in our evaluation is actually trained with many different rendering conditions. For the depth perception baseline, we use the depth maps generated from the simulator. We set the minimum and maximum sensible depths of the agent in the simulator to be 0.3m and 25m, respectively, which are close to those of the depth cameras (e.g., ZED) mounted on real

Model	Dimension	Format
Seg (Ours)	84×84×3	RGB Frame
Seg-S (Ours)	84×84×3×4	
DR-A3C	84×84×3	RGB Frame
DR-A3C-S	84×84×3×4	
Depth-A3C	84×84×1	Depth Map
Depth-A3C-S	84×84×4	
ResNet-A3C	224×224×3	RGB Frame

Table 2: Settings of the model inputs.

Reduced class labels	Original class labels in the ADE20K dataset
Wall	Window, Door, Fence, Pillar, Signboard, Bulletin board
Floor	Road, Ground, Field, Path, Runway
Ceil (Background)	Ceil
Obstacle	Bed, Cabinet, Sofa, Table, Curtain, Chair, Shelf, Desk, Plant
Target	(Depends on scenario)

Table 3: Class label reduction from the ADE20K dataset.

robots. We summarize the detailed settings of the model inputs in Table 2. All the models in Table 2 adopt the vanilla A3C architecture [Mnih *et al.*, 2016] except for ResNet-A3C, in which the convolutional layers are replaced by ResNet-101 pre-trained on ImageNet [Krizhevsky *et al.*, 2012]. ResNet-A3C is mainly used to justify that directly applying features extracted by ResNet as the meta-state representation does not lead to the same performance as ours. In Table 2, we denote our model as Seg, and the other baselines as DR-A3C, Depth-A3C, and ResNet-A3C, respectively. The models named with "-S" (e.g., DR-A3C-S) indicate that these models concatenate the latest four frames as their inputs. We do not include DA in our experiments, as we only focus on control policies solely trained in simulated environments without further fine-tuning. We use Adam optimizer [Kingma and Ba, 2015], and set both the learning rate and epsilon to 0.001. Each model is trained for 5M frames, and the training data are collected by 16 worker threads for all experimental settings. For the evaluation tasks in the real world, we train DeepLab-v2 [Chen *et al.*, 2016] on ADE20K [Zhou *et al.*, 2017] and ICNet [Zhao *et al.*, 2017a] on Cityscapes [Cordts *et al.*, 2016] as the indoor and outdoor perception modules, respectively. The class labels in ADE20K are reduced as Table 3 for better accuracy and training efficiency.

5 Experimental Results and Analysis

In this section, we present the experimental results and their implications. We comprehensively analyze the results, and perform an ablative study of our methodology. In Sections 5.1 and 5.2, we compare the performance of our architecture and the baseline models in the tasks mentioned in Section 4. We demonstrate the concept of visual guidance in Section 5.3. We present the results of the ablative analysis in Section 5.4.

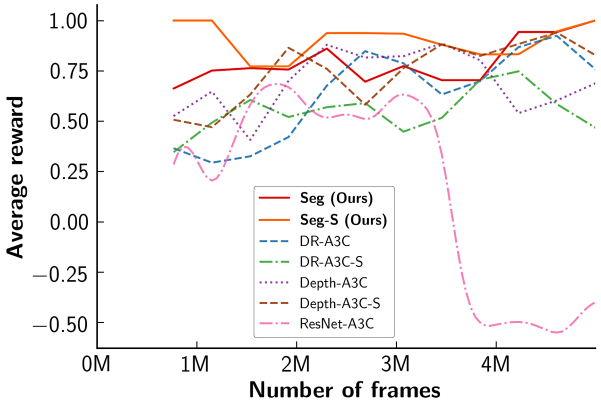


Figure 3: Learning curves in the obstacle avoidance task.

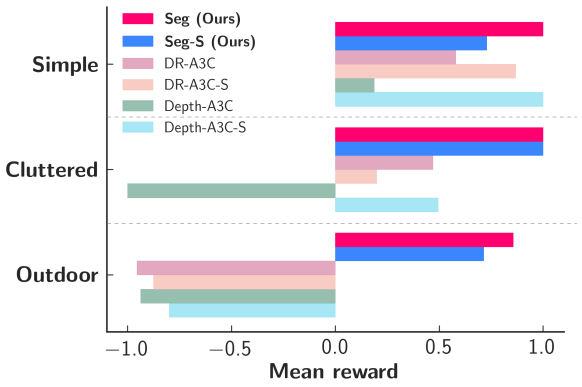


Figure 4: Evaluation results in the obstacle avoidance task (evaluated in simulated environments).

5.1 Comparison in the Obstacle Avoidance Task

To compare the performance of our models against those of the baselines in the obstacle avoidance tasks, we designed a total of 16 indoor scenarios for training the agents. Each scenario features a pre-defined set of attributes (e.g., hallway, static obstacles), along with randomly located obstacles and moving objects. A scenario is randomly selected for each training episode. Each episode terminates after 1,000 timesteps, or when a collision occurs. The agent receives a reward of 0.001 at each timesteps. Note that all the models are trained in simulation. Fig. 3 plots the learning curves of the models in the obstacle avoidance task. While most of the models achieve nearly optimal performance at the end of the training phase, our models learn significantly faster than the baseline models. This is due to the fact that semantic segmentation simplifies the original image representation to a more structured representation. On the contrary, the poor performance of ResNet-A3C indicates that the deep feature representations extracted by ResNet does not necessarily improve the performance. In addition, we observe that frame stacking has little impact on performance in the training phase. Depth-A3C shows a slightly better performance than DR-A3C and ResNet-A3C, implying the usefulness of depth information.

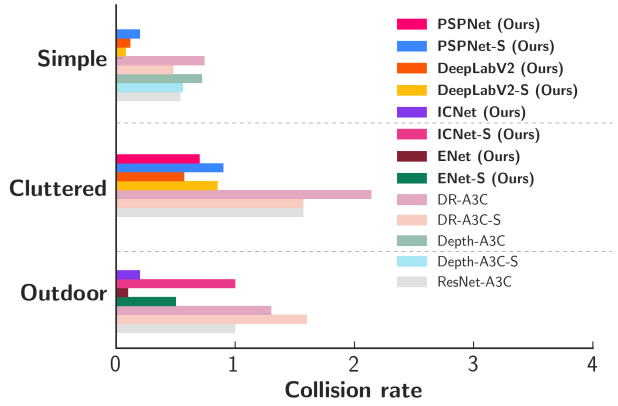


Figure 5: Collision rate in the obstacle avoidance task (evaluated in the real world).

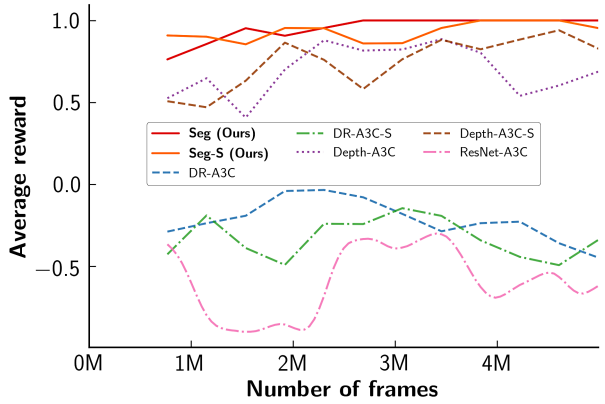


Figure 6: Learning curves in the target following task

Evaluation in Simulation. Fig. 4 compares the mean rewards of the agents in three scenarios for seven types of models. The mean rewards are evaluated over 100 episodes, and the scenarios are totally different from those used at the training phase. It can be observed that our methods outperform the other baseline models in all of the three scenarios. In *Simple Corridor*, noticeable performance can be observed from the other models. Most of the agents navigate well along the corridor, where there are relatively fewer obstacles compared to the other scenarios. In a more challenging scenario *Cluttered Hallway*, it can be seen that both Depth-A3C and Depth-A3C-S experience a considerable drop in performance. This is because depth map becomes too noisy in resolution to navigate in narrow space. In *Outdoor*, all models receive negative mean rewards, except for our models. We observe that the baseline models tend to drive on the sidewalks, or rotate in place. In contrast, our models focus on driving on the roadway. As a result, we conclude that using semantic segmentation as the meta-state representation enables a control policy to be transfered from an indoor to an outdoor environment.

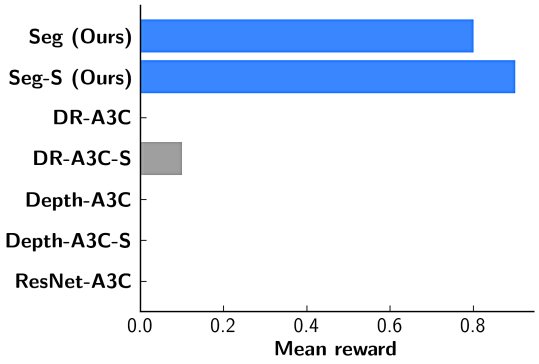


Figure 7: Evaluation results in the target following task (evaluated in simulated environments).

Evaluation in the Real World. To test the virtual-to-real transferability of the models, we replicate the settings of the scenarios mentioned above in the real world. We measure the performance by recording the number of collisions per minute. We manually position the robot back to a random position when it collides with an obstacle. We report the results in Fig. 5. In *Simple Corridor*, we observe that Depth-A3C experience a huge performance drop from simulation to the real world, since the depth map estimated in the real world is too noisy for the model to adapt. The performances of the other baselines are similarly decreased, while our methods still maintain the same level of performance. We exclude Depth-A3C from the rest of the real world scenarios due to the poor quality of depth estimation of the depth cameras in complex environments. In *Cluttered Hallway*, while all of the models degrade in performance, our methods still significantly outperform the baselines. In *Outdoor*, we observe that both DR-A3C and ResNet-A3C fail to transfer their knowledge from simulation to the real world, and tend to bump into cars and walls. We also notice that frame stacking leads to lower performance, because it amplifies the errors resulting from the changes in environmental dynamics. Additionally, to demonstrate the robustness of the proposed architecture to the quality of segmentation, we compare the performance of our methods with different segmentation models. They include PSPNet [Zhao *et al.*, 2017b] (high) and DeepLab [Chen *et al.*, 2016] (low) for indoor environments, and ICNet [Zhao *et al.*, 2017a] (high) and ENet [Paszke *et al.*, 2016] (low) for outdoor environments. Fig. 5 shows that our agents are not quite sensitive to the quality of image segmentations. Note that in *Outdoor* scenario, ICNet performs worse than ENet due to its inevitable computational delay. From the results in Figs. 3-5, we conclude that our methods are robust, generalizable, and transferable in the obstacle avoidance tasks.

5.2 Comparison in the Target Following Task

We further compare the performance of our methods against those of the baselines in the target following task. For each episode, a simulated scenario featuring static and moving obstacles is similarly selected at random. An episode ends immediately after 1,000 timesteps, or when a collision with the target/obstacles occurs. The agent receives a reward of

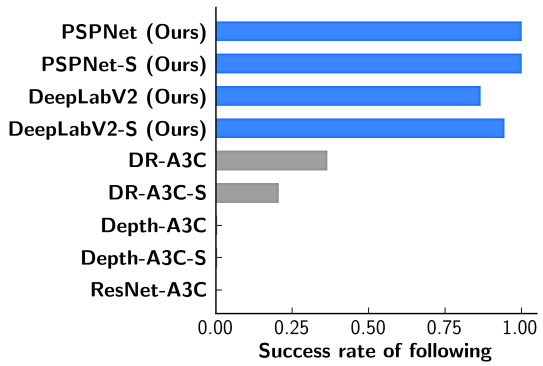
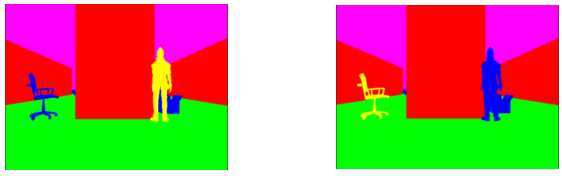


Figure 8: Success rate in the target following task (evaluated in the real world).



(a) The target is a person (b) The target is a chair

Figure 9: Visual guidance allows the target to be switched.

	VW (mean reward)	RW (success rate)
Seg	0.824	80%
Seg-S	0.925	90%

Table 4: Performance of visual guidance in the virtual world (VW) and real world (RW) in the *Switching-Target Following* task.

0.001 at each timestep if the target is within its sight, (1.0-"cumulative reward") if it touches the target, and 0.0 otherwise. At the beginning of each episode, the target is located in front of the agent. The target then navigates to a randomly chosen destination along the path computed by the A* algorithm. The ultimate goal of the agent is to avoid any collision on the way, and keep up with the target. For a fair comparison, the geometry of the target is always fixed. Fig. 6 plots the learning curves of our models and the baselines in the training phase. It can be observed that our models are much superior to the other baseline models. We also notice that our agents learn to chase the target in the early stages of the training phase, while the baseline models never learn to follow the target. We conclude that the superiority of our models over the baselines results from our meta-state representation, which makes the target easily recognizable by our agents.

Evaluation in Simulation. To validate the generalizability, we evaluate the agent in a virtual scenario, which is not included in the training scenario sets. In each episode, the agent starts from a random position in the scenario. We evaluate the mean rewards over 100 episodes, and present the results in Fig. 7. Our models achieve higher mean rewards than all the baselines, indicating the effectiveness of our models in capturing and tracking a fixed moving object. We observe that DR-A3C and ResNet-A3C always fail to follow the target at the fork of a corridor, because they are easily distracted by other objects. On the other hand, although Depth-A3C has higher mean rewards than the other baselines in the training phase, it often fails when the target is far away from it in the evaluation phase. The results in Fig. 7 validate that our models are generalizable and transferable to new scenarios.

Evaluation in the Real World To evaluate if a learned policy can be successfully transfered to the real world, we further conduct experiments in real indoor environments, which consist of corridors, forks, sharp turns, and randomly placed static obstacles. The target is a human who moves in the environment. The task is considered failed if the agent loses the target or collides with an obstacle, and successful if the agent follows the target to the destination. We report the success rate for each model in Fig. 8. The success rate is evaluated over ten episodes. The results show that all the baseline models fail in the real-world scenarios. Specifically, as the models trained with DR never learn a useful policy even in the simulated training environments, there is no doubt that they fail in the real-world evaluation. As for the cases of Depth-A3C and Depth-A3C-S, despite the high mean rewards they attained in the training phase, the learned policies do not transfer successfully to the real world. During the evaluation, we observe that DR-A3C and ResNet-A3C agents treat the target person as an obstacle, and avoid the target as it comes close. We also notice that our models do not lose track of the target even at sharp turns. We deduce that this is due to the temporal correlations preserved by frame stacking. Fig. 8 also provides evidence that our architecture is robust to the quality of image segmentation even in the target following task.

5.3 Demonstration of Visual Guidance

We demonstrate the effectiveness of the visual guidance module of our modular architecture by switching the target in a *Switching-Target Following* task. We perform experiments in both virtual and real environments. Both environments are the same as those in the *Target Following* task. To demonstrate visual guidance, we randomly select an object in the environment, and re-label it as the target at the beginning of each episode. The initial location of the agent is chosen such that the selected target falls in the field of view (FOV) of the agent. The selected object is rendered to yellow, as shown in Fig. 9. Fig. 9 illustrates the procedure of switching the target from Fig. 9 (a) a human to Fig. 9 (b) a chair. We measure the mean rewards over 1,000 episodes in the virtual environments, and the success rate over ten episodes in the real world. Table 4 summarizes the results. They reveal that our agents can successfully catch the randomly specified targets in both environments, no matter what target size and shape are.

5.4 Robustness to Non-Reduced Labels

We further show that our control policy module can retain its performance, even if its input contains non-reduced labels (i.e., the original class labels in the ADE20K dataset). We conduct obstacle avoidance experiments in *Cluttered Hallway*, as it is the most complex scenario for our agents.

Model	Collision rate
Reduced-Seg	0.7
Reduced-Seg-S	0.9
Non-Reduced-Seg	0.1
Non-Reduced-Seg-S	0.2

Table 5: Comparison of collision rate in *Cluttered Hallway*

When training in the simulated environments, we include non-reduced class labels in semantic segmentations. There are totally 27 non-reduced labels used for validating the robustness of our method. This enforces the agents to learn more labels in the training phase. Given the same amount of training time, the agents are able to obtain similar learning curves as those shown in Fig. 3. For evaluation in the real world, we compare the agents trained with non-reduced labels to those trained with reduced labels (i.e., the agents mentioned in Section 5.1). We adopt PSPNet as the perception module for both agents, and modify its output labels to be matched with the labels rendered by our simulator. The experimental results in Table 5 show that the agents trained with non-reduced labels demonstrate lower collision rate than their counterparts mentioned in Section 5.1. We attribute this improvement to the richer context of image segmentations perceived by the agents in the training phase.

6 Conclusion

In this paper, we presented a new modular architecture for transferring policies learned in simulators to the real world for vision-based robotic control. We proposed to separate the model into a perception module and a control policy module, and introduced the concept of using semantic image segmentation as the meta state for relating these two modules. We trained our model with a standard RL algorithm, and did not apply any DR technique. We performed experiments in two benchmark tasks: an obstacle avoidance task and a target following task, and demonstrated that the proposed method outperforms the baseline models in both virtual and real environments. We further validated the generalizability of our model in handling unfamiliar indoor and outdoor scenarios, and the transferability of our model from simulation to the real world without fine-tuning. Finally, in the switching-target following task, we proved that our model is flexible such that the target can be easily switched by visual guidance.

Acknowledgments

The authors thank Lite-On Technology Corporation for the support in researching funding, and NVIDIA Corporation for the donation of the Titan X Pascal GPU used for this research.

References

[Biswas and Veloso, 2012] Joydeep Biswas and Manuela Veloso. Depth camera based indoor mobile robot localization and navigation. In *Proc. Int. Conf. Robotics and Automation (ICRA)*, pages 1697-1702, May 2012.

[Chen *et al.*, 2016] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Trans. Pattern Analysis and Machine Intelligence*, Jun. 2016.

[Chen *et al.*, 2017] Yi-Hsin Chen, Wei-Yu Chen, Yu-Ting Chen, Bo-Cheng Tsai, Yu-Chiang Frank Wang, and Min Sun. No more discrimination: Cross city adaptation of road scene segmenters. In *Proc. Int. Conf. Computer Vision (ICCV)*, pages 2011-2020, Oct. 2017.

[Cordts *et al.*, 2016] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes dataset for semantic urban scene understanding. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 3213-3223, Jun. 2016.

[Finn and Levine, 2017] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *Proc. Int. Conf. Robotics and Automation (ICRA)*, pages 2786-2793, May 2017.

[Ghadirzadeh *et al.*, 2017] Ali Ghadirzadeh, Atsuto Maki, Danica Kragic, and Mårten Björkman. Deep predictive policy training using reinforcement learning. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pages 2351-2358, Sep. 2017.

[Gupta *et al.*, 2017] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 7272-7281, Jul. 2017.

[He *et al.*, 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 37, no.9, pp. 1904-1916, Sep. 2015.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 770-778, Jun. 2016.

[James and Johns, 2016] Stephen James and Edward Johns. 3D simulation for robot arm control with deep Q-learning. *arXiv:1609.03759*, Sep. 2016.

[Kingma and Ba, 2015] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. Int. Conf. Learning Representations (ICLR)*, May 2015.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 1097-1105. Dec. 2012.

[Long *et al.*, 2015] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 3431-3440, Jul. 2015.

[Maier *et al.*, 2012] Daniel Maier, Armin Hornung, and Maren Bennewitz. Real-time navigation in 3D environments based on depth camera data. In *Proc. Int. Conf. Humanoid Robots (Humanoids)*, pages 692-697, Nov. 2012.

[Mnih *et al.*, 2016] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proc. Int. Conf. Machine Learning (ICML)*, pages 1928-1937, Jul. 2016.

[Pan *et al.*, 2017] Xinlei Pan, Yurong You, Ziyan Wang, and Cewu Lu. Virtual to real reinforcement learning for autonomous driving. In *Proc. The British Machine Vision Conference (BMVC)*, Sep. 2017.

[Parisotto and Salakhutdinov, 2018] Emilio Parisotto and Ruslan Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. In *Proc. Int. Conf. Learning Representations (ICLR)*, Apr.-May 2018.

[Paszke *et al.*, 2016] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. ENet: A deep neural network architecture for real-time semantic segmentation. *arXiv:1606.02147*, Jun. 2016.

[Peng *et al.*, 2018] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *Proc. Int. Conf. Robotics and Automation (ICRA)*, May 2018.

[Richter *et al.*, 2017] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *Proc. Int. Conf. Computer Vision (ICCV)*, pages 2232-2241, Oct. 2017.

[Rusu *et al.*, 2016] Andrei A Rusu, Matej Vecerik, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. Sim-to-real robot learning from pixels with progressive nets. In *Proc. Conf. Robot Learning (CoRL)*, pages 262-270, Nov. 2016.

[Sadeghi and Levine, 2017] Fereshteh Sadeghi and Sergey Levine. CAD^{2}RL: Real single-image flight without a single real image. In *Proc. Robotics: Science and Systems (RSS)*, Jul. 2017.

[Smolyanskiy *et al.*, 2017] Nikolai Smolyanskiy, Alexey Kamenev, Jeffrey Smith, and Stan Birchfield. Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pages 4241-4247, Sep. 2017.

[Tobin *et al.*, 2017] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pages 23-30, Sep. 2017.

[Williams, 1992] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *J. Machine learning*, vol. 8, no. 3-4, pp. 229-256, May 1992.

[Wu and Tian, 2017] Yuxin Wu and Yuandong Tian. Training agent for first-person shooter game with actor-critic curriculum learning. In *Proc. Int. Conf. Learning Representations (ICLR)*, Apr. 2017.

[Yan *et al.*, 2017] Mengyuan Yan, Iuri Frosio, Stephen Tyree, and Jan Kautz. Sim-to-real transfer of accurate grasping with eye-in-hand observations and continuous control. *arXiv:1712.03303*, Dec. 2017.

[Zhang *et al.*, 2017] Fangyi Zhang, Jürgen Leitner, Michael Milford, and Peter Corke. Sim-to-real transfer of visuo-motor policies for reaching in clutter: Domain randomization and adaptation with modular networks. *arXiv:1709.05746*, Sep. 2017.

[Zhao *et al.*, 2017a] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. ICNet for real-time semantic segmentation on high-resolution images. *arXiv:1704.08545*, Apr. 2017.

[Zhao *et al.*, 2017b] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 2881-2890, Jul. 2017.

[Zhou *et al.*, 2017] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 5122-5130, Jul. 2017.

[Zhu *et al.*, 2017] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *Proc. Int. Conf. Robotics and Automation (ICRA)*, pages 3357-3364, May 2017.