

Virtual to Real Reinforcement Learning for Autonomous Driving

Xinlei Pan^{*1}

xinleipan@berkeley.edu

Yurong You^{*2}

yurongyou@sjtu.edu.cn

Ziyan Wang³

zy-wang13@mails.tsinghua.edu.cn

Cewu Lu²

lu-cw@cs.sjtu.edu.cn

^{*}Indicates equal contribution.

¹ University of California, Berkeley
Berkeley, CA, USA

² Shanghai Jiao Tong University
Shanghai, China

³ Tsinghua University
Beijing, China

Abstract

Reinforcement learning is considered as a promising direction for driving policy learning. However, training autonomous driving vehicle with reinforcement learning in real environment involves non-affordable trial-and-error. It is more desirable to first train in a virtual environment and then transfer to the real environment. In this paper, we propose a novel realistic translation network to make model trained in virtual environment be workable in real world. The proposed network can convert non-realistic virtual image input into a realistic one with similar scene structure. Given realistic frames as input, driving policy trained by reinforcement learning can nicely adapt to real world driving. Experiments show that our proposed virtual to real (VR) reinforcement learning (RL) works pretty well. To our knowledge, this is the first successful case of driving policy trained by reinforcement learning that can adapt to **real world** driving data.

Autonomous driving aims to make a vehicle sense its environment and navigate without human input. To achieve this goal, the most important task is to learn the driving policy that automatically outputs control signals for steering wheel, throttle, brake, etc., based on observed surroundings. The straight-forward idea is end-to-end supervised learning [3, 4], which trains a neural network model mapping visual input directly to action output, and the training data is labeled image-action pairs. However, supervised approach usually requires large amount of data to train a model [3] that can generalize to different environments. Obtaining such amount of data is time consuming and requires significant human involvement. By contrast, reinforcement learning learns by a trial-and-error fashion, and does not require explicit supervision from human. Recently, reinforcement learning has been considered as a promising technique to learn driving policy due to its expertise in action planning [5, 6, 7].

However, reinforcement learning requires agents to interact with environments, and undesirable driving actions would happen. Training autonomous driving cars in real world will cause damages to vehicles and the surroundings. Therefore, most of current research in autonomous driving with reinforcement learning focus on simulations [8, 9, 10] rather than training in real world. While an agent trained with reinforcement learning achieves

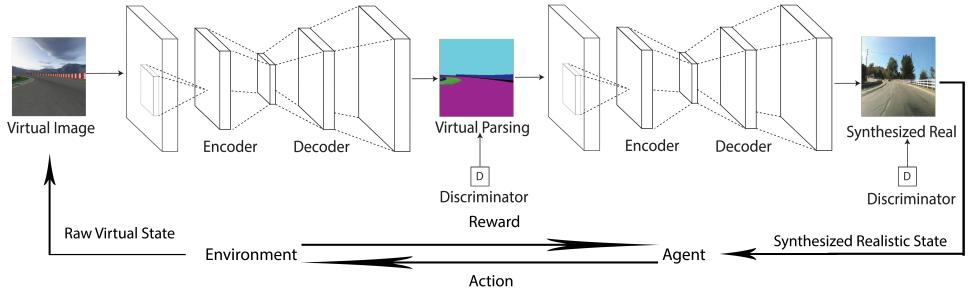


Figure 1: Framework for virtual to real reinforcement learning for autonomous driving. Virtual images rendered by a simulator (environment) are first segmented to scene parsing representation and then translated to synthetic realistic images by the proposed image translation network (VISRI). Agent observes synthetic realistic images and takes actions. Environment will give reward to the agent. Since the agent is trained using realistic images that are visually similar to real world scenes, it can nicely adapt to real world driving.

near human-level driving performance in virtual world [18], it may not be applicable to real world driving environment, since the visual appearance of virtual simulation environment is different from that of real world driving scene.

While virtual driving scenes have a different visual appearance compared with real driving scenes, they share similar scene parsing structure. For example, virtual and real driving scenes may all have roads, trees, buildings, etc., though the textures may be significantly different. Therefore, it is reasonable that by translating virtual images to their realistic counterparts, we can obtain a simulation environment that looks very similar to the real world in terms of both scene parsing structure and object appearance. Recently, generative adversarial network (GAN) [9] has drawn a lot of attention in image generation. The work by [10] proposed an image-to-image translation network that can translate images from one domain to another using paired data from both domains. However, it is very hard to find paired virtual-real world images for driving, making it difficult to apply this method to our case of translating virtual driving images to realistic ones.

In this paper, we propose a *realistic translation network* to help train self-driving car entirely in virtual world that can adapt to real world driving environment. Our proposed framework (shown in Figure 1) converts virtual images rendered by the simulator to a realistic one and train the reinforcement learning agent with the synthesized realistic images. Though virtual and realistic images have a different visual appearance, they share a common scene parsing representation (segmentation map of roads, vehicles etc.). Therefore, we can translate virtual images to realistic images by using scene parsing representation as the interim. This insight is similar to natural language translation, where semantic meaning is the interim between different languages. Specifically, our realistic translation network includes two modules. The first one is a virtual-to-parsing or virtual-to-segmentation module that produces a scene parsing representation of input virtual image. The second one is a parsing-to-real network that translates scene parsing representations into realistic images. With realistic translation network, reinforcement learning model learnt on the realistic driving data can nicely apply to real world driving.

To demonstrate the effectiveness of our method, we trained our reinforcement learning model by using the realistic translation network to filter virtual images to synthetic realistic

images and feed these realistic images as state inputs. We further compared with supervised learning and other reinforcement learning approaches that use domain randomization [29]. Our experiments illustrate that a reinforcement learning model trained with translated realistic images has better performance than reinforcement learning model trained with only virtual input and virtual to real reinforcement learning with domain randomization.

1 Related Work

Supervised Learning for Autonomous Driving. Supervised learning methods are obviously straightforward ways to train autonomous vehicles. ALVINN [19] provides an early example of using neural network for autonomous driving. Their model is simple and direct, which maps image inputs to action predictions with a shallow network. Powered by deep learning especially a convolutional neural network, NVIDIA [8] recently provides an attempt to leverage driving video data for simple lane following task. Another work by [2] learns a mapping between input images to a number of key perception indicators, which are closely related to the affordance of a driving state. However, the learned affordance must be associated with actions through hand-engineered rules. These supervised methods work relatively well in simple tasks such as lane-following and driving on a highway. On the other hand, imitation learning can also be regarded as supervised learning approach [32], where the agent observes the demonstrations performed by some experts and learns to imitate the actions of the experts. However, an intrinsic shortcoming of imitation learning is that it has the covariate shift problem [20] and can not generalize very well to scenes never experienced before.

Reinforcement Learning for Autonomous Driving. Reinforcement learning has been applied to a wide variety of robotics related tasks, such as computer games [2], robot locomotion [8, 12], and autonomous driving [11, 23]. One of the challenges in practical real-world applications of reinforcement learning is the high-dimensionality of state space as well as the non-trivial large action range. Developing an optimal policy over such high-complexity space is time consuming. Recent work in deep reinforcement learning has made great progress in learning in a high dimensional space with the power of deep neural networks [10, 15, 17, 18, 24]. However, both deep Q-learning method [10] and policy gradient method [15] require the agent to interact with the environment to get reward and feedback. However, it is unrealistic to train autonomous vehicle with reinforcement learning in a real world environment since the car may hurt its surroundings once it takes a wrong action.

Reinforcement Learning in the Wild. Performing reinforcement learning with a car driving simulator and transferring learned models to the real environment could enable faster, lower-cost training, and it is much safer than training with a real car. However, real-world driving challenge usually spans a diverse range, and it is often significantly different from the training environment in a car driving simulator in terms of their visual appearance. Models trained purely on virtual data do not generalize well to real images [8, 23]. Recent progress of transfer and domain adaptation learning in robotics provides examples of simulation-to-real reinforcement training [11, 21, 26]. These models either first train a model in virtual environment and then fine-tune in the real environment [21], or learn an alignment between virtual images and real images by finding representations that are shared between the two domains [26], or use randomized rendered virtual environments to train and then test in real environment [21, 26]. The work of [21] proposes to use progressive network to transfer network weights from model trained on virtual data to the real environment and then fine-

tune the model in a real setting. The training time in real environment has been greatly reduced by first training in a virtual environment. However, it is still necessary to train the agent in the real environment, thus it does not solve the critical problem of avoiding risky trial-and-error in real world. Methods that try to learn an alignment between virtual images and real images could fail to generalize to more complex scenarios, especially when it is hard to find a good alignment between virtual images and real images. As a more recent work, [22] proposed a new framework for training a reinforcement learning agent with only a virtual environment. Their work proved the possibility of performing collision-free flight in real world with training in 3D CAD model simulator. However, as mentioned in the conclusion of their paper [22], the manual engineering work to design suitable training environments is nontrivial, and it is more reasonable to attain better results by combining simulated training with some real data, though it is unclear from their paper how to combine real data with simulated training.

Image Synthesis and Image Translation. Image translation aims to predict image in some specific modality, given an image from another modality. This can be treated as a generic method as it predicts pixels from pixels. Recently, the community has made significant progress in generative approaches, mostly based on generative adversarial networks [8]. To name a few, the work of [24] explored the use of VAE-GAN [25] in generating 3D voxel models, and the work of [26] proposed a cascade GAN to generate natural images by structure and style. More recently, the work of [27] developed a general and simple framework for image-to-image translation which can handle various pixel level generative tasks like semantic segmentation, colorization, rendering edge maps, etc.

Scene Parsing. One part of our network is the semantic image segmentation network. There are already many great works in the field of semantic image segmentation. Many of them are based on deep convolutional neural network or fully convolutional neural network [16]. In this paper, we use the SegNet for image segmentation, the structure of the network is revealed in [9], which is composed of two main parts. The first part is an encoder, which consists of Convolutional, Batch Normalization, ReLU and max pooling layers. The second part is a decoder, which replaces the pooling layers with upsampling layers.

2 Reinforcement Learning in the Wild

We aim to successfully apply a driving model trained entirely in virtual environment to real-world driving challenges. One of the major gaps is that what the agent observes are frames rendered by a simulator, which are different from real world frames in terms of their appearance. Therefore, we proposed a *realistic translation network* to convert virtual frames to realistic ones. Inspired by the work of image-to-image translation network [28], our network includes two modules, namely virtual-to-parsing and parsing-to-realistic network. The first one maps virtual frame to scene parsing image. The second one translates scene parsing to realistic frame with similar scene structure as the input virtual frame. These two modules generate realistic frames that maintain the scene parsing structure of input virtual frames. The architecture of realistic translation network is illustrated on Figure 1. Finally, we train a self-driving agent using reinforcement learning method on realistic frames obtained by realistic translation network. The approach we adopt is developed by [18], where they use the asynchronous actor-critic reinforcement learning algorithm to train a self-driving vehicle in the car racing simulator TORCS [30]. In this section, we will first present proposed realistic translation network and then discuss how to train driving agent under a reinforcement

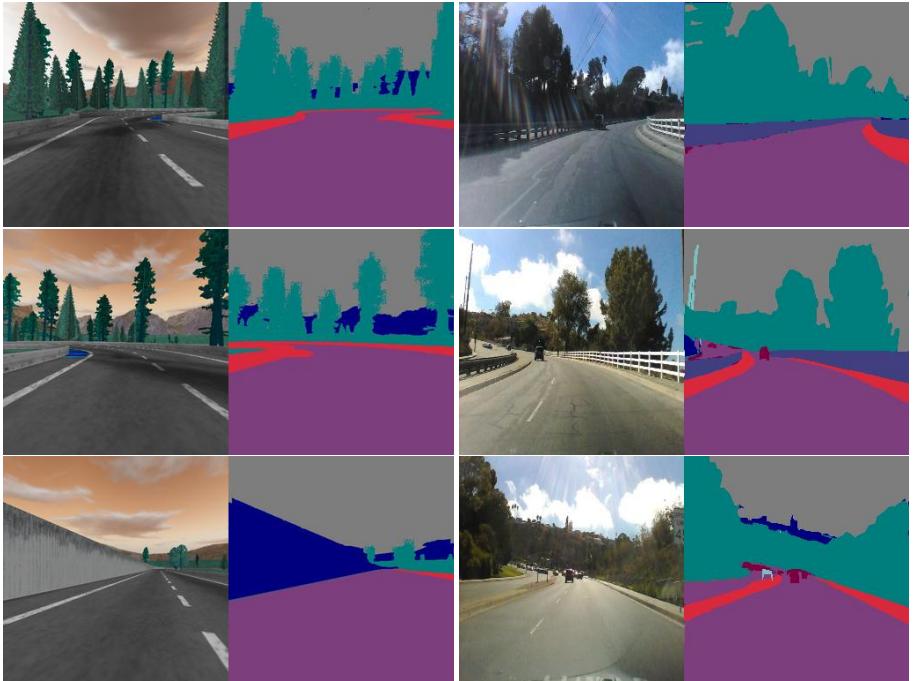


Figure 2: Example image segmentation for both virtual world images (Left 1 and Left 2) and real world images (Right 1 and Right 2).

learning framework.

2.1 Realistic Translation Network

As there is no paired virtual and real world image, a direct mapping from virtual world image to real world image using [10] would be awkward. However, as these two types of images both express driving scene, we can translate them by using scene parsing representation. Inspired by [10], our realistic translation network is composed of two image translation networks, where the first image translation network translates virtual images to their segmentations, and the second image translation network translates segmented images to their real world counterparts.

The image-to-image translation network proposed by [10] is basically a conditional GAN. The difference between traditional GANs and conditional GANs is that GANs learn a mapping from random noise vector z to output image $s : G : z \rightarrow s$, while conditional GANs take in both an image x and a noise vector z , and generate another image $s : G : \{x, z\} \rightarrow s$, where s is usually in a different domain compared with x (For example, translate images to their segmentations).

The objective of a conditional GAN can be expressed as,

$$\begin{aligned} \mathcal{L}_{cGAN}(G, D) = & \mathbb{E}_{x, s \sim p_{data}(x, s)} [\log D(x, s)] \\ & + \mathbb{E}_{x \sim p_{data}(x), z \sim p_z(z)} [\log(1 - D(x, G(x, z)))], \end{aligned} \quad (1)$$

where G is the generator that tries to minimize this objective and D is the adversarial discriminator that acts against G to maximize this objective. In other words, $G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D)$. In order to suppress blurring, a L1 loss regularization term is added, which can be expressed as,

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x, s \sim p_{data}(x, s), z \sim p_z(z)} [\|s - G(x, z)\|_1]. \quad (2)$$

Therefore, the overall objective for the image-to-image translation network is,

$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G), \quad (3)$$

where λ is the weight of regularization.

Our network consists of two image-to-image translation networks, both networks use the same loss function as equation 3. The first network translates virtual images x to their segmentations $s : G_1 : \{x, z_1\} \rightarrow s$, and the second network translates segmented images s into their realistic counterparts $y : G_2 : \{s, z_2\} \rightarrow y$, where z_1, z_2 are noise terms to avoid deterministic outputs. As for GAN neural network structures, we use the same generator and discriminator architectures as used in [10].

2.2 Reinforcement Learning for Training a Self-Driving Vehicle

We use a conventional RL solver Asynchronous Advantage Actor-Critic (A3C)[18] to train the self driving vehicle, which has performed well on various machine learning tasks. A3C algorithm is a fundamental Actor-Critic algorithm that combines several classic reinforcement learning algorithms with the idea of asynchronous parallel threads. Multiple threads run at the same time with unrelated copies of the environment, generating their own sequences of training samples. Those actors-learners proceed as though they are exploring different parts of the unknown space. For one thread, parameters are synchronized before an iteration of learning and updated after finishing it. The details of A3C algorithm implementation can be found in [18].

In order to encourage the agent to drive faster and avoid collisions, we define the reward function as

$$r_t = \begin{cases} (v_t \cdot \cos \alpha - \text{dist}_{\text{center}}^{(t)}) \cdot \beta & \text{no collision,} \\ \gamma & \text{collision,} \end{cases} \quad (4)$$

where v_t is the speed (in m/s) of the agent at time step t , α is the angle (in rad) between the agent's speed and the tangent line of the track, and $\text{dist}_{\text{center}}^{(t)}$ is the distance between the center of the agent and the middle of the track. β, γ are constants and are determined at the beginning of training. We take $\beta = 0.006, \gamma = -0.025$ in our training.

3 Experiments

We performed two sets of experiments to compare the performance of our method and other reinforcement learning methods as well as supervised learning methods. The first sets of experiments involves virtual to real reinforcement learning on real world driving data. The second sets of experiments involves transfer learning in different virtual driving environments. The virtual simulator used in our experiments is TORCS[30].

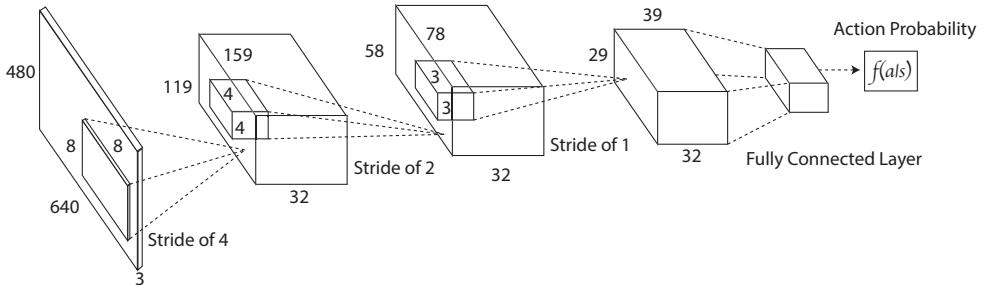


Figure 3: Reinforcement learning network architecture. The network is an end-to-end network mapping state representations to action probability outputs.



Figure 4: Examples of Virtual to Real Image Translation. Odd columns are virtual images captured from TORCS. Even columns are synthetic real world images corresponding to virtual images on the left.

3.1 Virtual to Real RL on Real World Driving Data

In this experiment, we trained our proposed reinforcement learning model with realistic translation network. We first trained the virtual to real image translation network, and then used the trained network to filter virtual images in simulator to realistic images. These realistic images were then feed into A3C to train a driving policy. Finally, the trained policy was tested on a real world driving data to evaluate its steering angle prediction accuracy.

For comparison, we also trained a supervised learning model to predict steering angles for every test driving video frame. The model is a deep neural network that has the same architecture design as the policy network in our reinforcement learning model. The input of the network is a sequence of four consecutive frames, the output of the network is the action probability vector, and elements in the vector represent the probability of going straight, turning left and turning right. The training data for the supervised learning model is different from the testing data that is used to evaluate model performance. In addition, another baseline reinforcement learning model (B-RL) is also trained. The only difference between B-RL and our method is that the virtual world images were directly taken by the agent as state inputs. This baseline RL is also tested on the same real world driving data.

Dataset. The real world driving video data are from [8], which is collected in a sunny day with detailed steering angle annotations per frame. There are in total around 45k images in this dataset, of which 15k were selected for training the supervised learning model, and another 15k were selected and held out for testing. To train our realistic translation net-

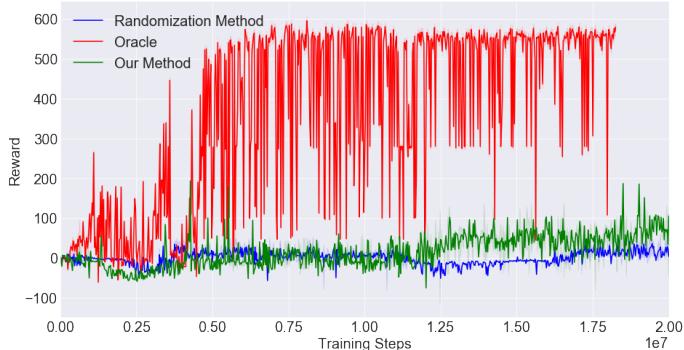


Figure 5: Transfer learning between different environments. Oracle was trained in Cgtrack2 and tested in Cgtrack2, so its performance is the best. Our model works better than the domain randomization RL method. Domain randomization method requires training in multiple virtual environments, which imposes significant manual engineering work.

work, we collected virtual images and their segmentations from the Aalborg environment in TORCS. A total of 1673 images were collected which covers the entire driving cycle of Aalborg environment.

Scene Segmentation. We used the image semantic segmentation network design of [1] and their trained segmentation network on the CityScape image segmentation dataset [2] to segment 45k real world driving images from [3]. The network was trained on the CityScape dataset with 11 classes and was trained with 30000 iterations.

Image Translation Network Training. We trained both virtual-to-parsing and parsing-to-real network using the collected virtual-segmentation image pairs and segmentation-real image pairs. The translation networks are of a encoder-decoder fashion as shown in figure 1. In the image translation network, we used U-Net architecture with skip connection to connect two separate layers from encoder and decoder respectively, which have the same output feature map shape. The input size of the generator is 256×256 . Each convolutional layer has a kernel size of 4×4 and striding size of 2. LeakyReLU is applied after every convolutional layer with a slope of 0.2 and ReLU is applied after every deconvolutional layer. In addition, batch normalization layer is applied after every convolutional and deconvolutional layer. The final output of the encoder is connected with a convolutional layer which yields output of shape $3 \times 256 \times 256$ followed by Tanh. We used all 1673 virtual-segmentation image pairs to train a virtual to segmentation network. As there are redundancies in the 45k real images, we selected 1762 images and their segmentations from the 45k images to train a parsing-to-real image translation network. To train the image translation models, we used the Adam optimizer with an initial learning rate of 0.0002, momentum of 0.5, batchsize of 16, and 200 iterations until convergence.

Reinforcement Training. The RL network structure used in our training is similar to that of [18] where the actor network is a 4-layer convolutional network (shown in figure 3) with ReLU activation functions in-between. The network takes in 4 consecutive RGB frames as state input and output 9 discrete actions which corresponds to “go straight with acceleration”, “go left with acceleration”, “go right with acceleration”, “go straight and brake”, “go left and brake”, “go right and brake”, “go straight”, “go left”, and “go right”. We trained the reinforcement learning agent with 12 asynchronous threads, and with the RMSProp op-

timizer at an initial learning rate of 0.01, $\gamma = 0.9$, and $\epsilon = 0.1$.

Evaluation. The real world driving dataset [8] provides the steering angle annotations per frame. However, the actions performed in the TORCS virtual environment only contain "going left", "going right", and "going straight" or their combinations with "acceleration" or "brake". Therefore, we define a label mapping strategy to translate steering angle labels to action labels in the virtual simulator. We relate steering angle in $(-10, 10)$ to action "going straight" (since small steering angle is not able to result in a distinct turning in a short time), steering angle less than -10 to action "going left" and steering angle more than 10 to action "going right". By comparing output actions generated from our method with ground truth, we can obtain the accuracy of driving action prediction.

3.2 Transfer Learning in Virtual Driving Environments

We further performed another sets of experiments and obtained results of transfer learning between different virtual driving environments. In this experiments, we trained three reinforcement learning agents. The first agent was trained with standard A3C in the Cg-track2 environment in TORCS, and evaluated its performance frequently in the same environments. It is reasonable to expect the performance of this agent to be the best, so we call it "Oracle". The second agent was trained with our proposed reinforcement learning method with realistic translation network. However, it was trained in E-track1 environment in TORCS and then evaluated in Cg-track2. It is necessary to note that the visual appearance of E-track1 is different from that of Cg-track2. The third agent was trained with domain randomization method similar to that of [20], where the agent was trained with 10 different virtual environments and evaluated in Cg-track2. For training with our methods, we obtain 15k segmented images for both E-track1 and Cg-track2 to train virtual-to-parsing and parsing-to-real image translation networks. The image translation training details and reinforcement learning details are the same as that of section 3.1.

4 Results

Image Segmentation Results. We used image segmentation model trained on the cityscape [9] dataset to segment both virtual and real images. Examples are shown in figure 2. As shown in the figure, although the original virtual image and real image look quite different, their scene parsing results are very similar. Therefore, it is reasonable to use scene parsing as the interim to connect virtual images and real images.

Qualitative Result of Realistic Translation Network. Figure 4 shows some representative results of our image translation network. The odd columns are virtual images in TORCS, and the even columns are translated realistic images. The images in the virtual environment appears to be darker than the translated images, as the real images used to train the translation network is captured in a sunny day. Therefore, our model succeed to synthesize realistic images of similar appearance with the original ground truth real images.

Reinforcement Training Results. The results for virtual to real reinforcement learning on real world driving data are shown in table 1. Results show that our proposed method has a better overall performance than the baseline method (B-RL), where the reinforcement training agent is trained in a virtual environment without seeing any real data. The supervised method (SV) has the best overall performance, however, was trained with large amounts of supervised labeled data.

Table 1: Action prediction accuracy for the three methods.

Accuracy	Ours	B-RL	SV
Dataset in [3]	43.40%	28.33%	53.60%

The result for transfer learning in different virtual environments is shown in figure 5. Obviously, standard A3C (Oracle) trained and tested in the same environment gets the best performance. However, our model performs better than the domain randomization method, which requires training in multiple environments to generalize. As mentioned in [22], domain randomization requires lots of engineering work to make it generalize. Our model succeeds by observing translated images from E-track1 to Cg-track2, which means the model already gets training in an environment that looks very similar to the test environment (Cg-track2), thus the performance is improved.

5 Conclusion

We proved through experiments that by using synthetic real images as training data in reinforcement learning, the agent generalizes better in a real environment than pure training with virtual data or training with domain randomization. The next step would be to design a better image-to-image translation network and a better reinforcement learning framework to surpass the performance of supervised learning.

Thanks to the bridge of scene parsing, virtual images can be translated into realistic images which maintain their scene structure. The learnt reinforcement learning model on realistic frames can be easily applied to real-world environment. We also notice that the translation results of a segmentation map are not unique. For example, segmentation map indicates a car, but it does not assign which color of that car should be. Therefore, one of our future work is to make parsing-to-realistic network output various possible appearances (e.g. color, texture). In this way, bias in reinforcement learning training would be largely reduced.

We provide the first example of training a self-driving vehicle using reinforcement learning algorithm by interacting with a synthesized real environment with our proposed image-to-segmentation -to-image framework. We show that by using our method for RL training, it is possible to obtain a self driving vehicle that can be placed in the real world.

References

- [1] Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Y Ng. An application of reinforcement learning to aerobatic helicopter flight. *Advances in neural information processing systems*, 19:1, 2007.
- [2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015.
- [3] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016. URL <http://arxiv.org/abs/1604.07316>.
- [4] Chenyi Chen, Ari Seff, Alain L. Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. *CoRR*, abs/1505.00256, 2015. URL <http://arxiv.org/abs/1505.00256>.
- [5] Sully Chen. Autopilot-tensorflow, 2016. URL <https://github.com/SullyChen/Autopilot-TensorFlow>.
- [6] Paul Christiano, Zain Shah, Igor Mordatch, Jonas Schneider, Trevor Blackwell, Joshua Tobin, Pieter Abbeel, and Wojciech Zaremba. Transfer from simulation to real world through learning deep inverse dynamics model. *arXiv preprint arXiv:1610.03518*, 2016.
- [7] 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. *CoRR*, abs/1604.01685, 2016. URL <http://arxiv.org/abs/1604.01685>.
- [8] Gen Endo, Jun Morimoto, Takamitsu Matsubara, Jun Nakanishi, and Gordon Cheng. Learning cpg-based biped locomotion with a policy gradient method: Application to a humanoid robot. *The International Journal of Robotics Research*, 27(2):213–228, 2008.
- [9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. URL <http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf>.
- [10] Abhishek Gupta, Coline Devin, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. *arXiv preprint arXiv:1703.02949*, 2017.
- [11] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016. URL <http://arxiv.org/abs/1611.07004>.

-
- [12] Nate Kohl and Peter Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 3, pages 2619–2624. IEEE, 2004.
 - [13] Jan Koutník, Giuseppe Cuccu, Jürgen Schmidhuber, and Faustino Gomez. Evolving large-scale neural networks for vision-based reinforcement learning. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pages 1061–1068. ACM, 2013.
 - [14] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *CoRR*, abs/1512.09300, 2015. URL <http://arxiv.org/abs/1512.09300>.
 - [15] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
 - [16] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
 - [17] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.
 - [18] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *CoRR*, abs/1602.01783, 2016. URL <http://arxiv.org/abs/1602.01783>.
 - [19] Dean A Pomerleau. Alvinn, an autonomous land vehicle in a neural network. Technical report, Carnegie Mellon University, Computer Science Department, 1989.
 - [20] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *AISTATS*, volume 3, pages 3–5, 2010.
 - [21] Andrei A. Rusu, Matej Vecerik, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. Sim-to-real robot learning from pixels with progressive nets. *CoRR*, abs/1610.04286, 2016. URL <http://arxiv.org/abs/1610.04286>.
 - [22] Fereshteh Sadeghi and Sergey Levine. (cad)\$^2\$rl: Real single-image flight without a single real image. *CoRR*, abs/1611.04201, 2016. URL <http://arxiv.org/abs/1611.04201>.
 - [23] Ahmad El Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. End-to-end deep reinforcement learning for lane keeping assist. *arXiv preprint arXiv:1612.04340*, 2016.
 - [24] John Schulman, Sergey Levine, Pieter Abbeel, Michael I Jordan, and Philipp Moritz. Trust region policy optimization. In *ICML*, pages 1889–1897, 2015.

-
- [25] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *CoRR*, abs/1610.03295, 2016. URL <http://arxiv.org/abs/1610.03295>.
 - [26] 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. *arXiv preprint arXiv:1703.06907*, 2017.
 - [27] Eric Tzeng, Coline Devin, Judy Hoffman, Chelsea Finn, Pieter Abbeel, Sergey Levine, Kate Saenko, and Trevor Darrell. Adapting deep visuomotor representations with weak pairwise constraints. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2016.
 - [28] Xiaolong Wang and Abhinav Gupta. Generative image modeling using style and structure adversarial networks. In *ECCV*, 2016.
 - [29] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016.
 - [30] Bernhard Wyman, Eric Espié, Christophe Guionneau, Christos Dimitrakakis, Rémi Coulom, and Andrew Sumner. Torcs, the open racing car simulator. *Software available at http://torcs.sourceforge.net*, 2000.
 - [31] Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. *arXiv preprint arXiv:1612.01079*, 2016.
 - [32] Jiakai Zhang and Kyunghyun Cho. Query-efficient imitation learning for end-to-end autonomous driving. *arXiv preprint arXiv:1605.06450*, 2016.