Exploring Semantic and Depth Penalties for Sketch Generation

Jaskirat Singh, Tanya Dixit, Kaiyuan Xing & Ruotian Zhang
College of Engineering and Computer Science
Australian National University
Canberra, Australia
jaskirat.singh@anu.edu.au

Abstract

Creativity is an essential aspect of human intelligence. In this project, we aim to impart this skill to machines by teaching them *"how to paint"* real-world scenes using deep reinforcement learning. Given an input image, we predict a sequence of brush strokes that, when transferred to a canvas, lead to a visually and semantically similar image. Previous works in this direction, limit the "painting" problem to relatively simple datasets containing a single semantic class. In this paper, we endeavor to extend this methodology to more complex datasets featuring instances belonging to one or more semantic classes. We use semantic class segmentation and monocular depth prediction to improve the granularity of generated canvases. We show that our method results in better sample complexity (faster learning speed) and qualitatively better images on the highly challenging Virtual-KITTI dataset.

1 Introduction

The most difficult things to teach a machine have a lot to do with human creativity. Visual art is one of those creative endeavours where it is difficult to model an exact process that artificial intelligence systems can imitate. However, if machines could learn to paint a scene using information that humans use [2, 12], it could shed some light on how humans go about learning to paint. The approach that we follow is also an attempt towards more explainable AI and making human intuition more accessible to a neural network model.

Earlier approaches used supervised data to learn the stroke order and characteristics in order to create a sketch, just like a human. However, such data is rare and not always readily available for complex paintings that require a large number of strokes. Reinforcement learning, coupled with adversarial learning, helps deal with the problems of supervised approaches as it allows the model to learn an abstract idea of the painting. But even after learning the stroke order, type and colour, these approaches do not take into account semantic and depth cues that humans would generally notice first. When painting a landscape, a portrait or a bird, humans would first identify what the object is, *i.e.* what class it belongs to. They also use something called "perspective", for creating the illusion of depth, which means they use a sense of depth or depth data.

To make the algorithm closer to how humans would perceive a scene, we modify the approach to include class-based segmentation and depth data. Looking at an image from semantic classes and depth is not a new concept [2, 5, 12], but it has not been previously explored for this particular problem. Modeling this problem and learning as humans would, is a crucial part of our approach, as the information we are trying to supply to the model is "human intuition" which is not so easy to define or measure. We supply this information by giving negative rewards to actions which humans would not commit while creating a painting. For instance, a painter would not paint across two separate depths in one stroke; hence we give a negative reward if the reinforcement learning agent does so. A painter would also not paint two different semantic classes with a single brush stroke;

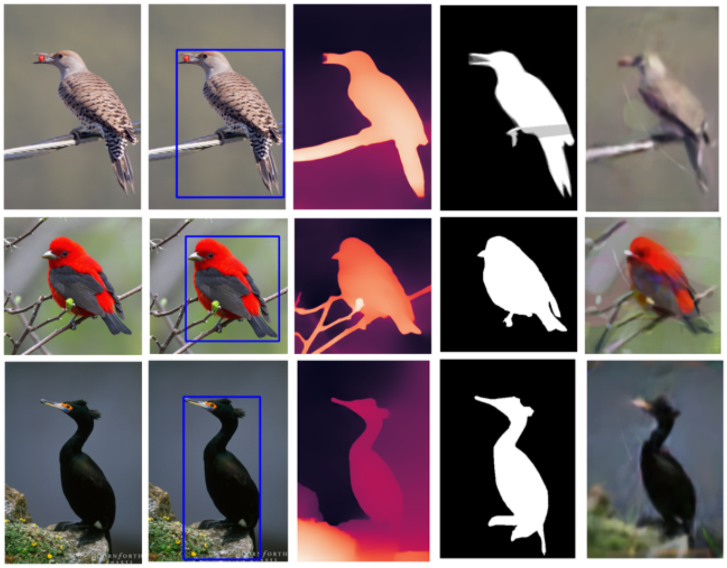

Figure 1: Examples of the AutoPaint agent input and output.**(from left)** Target Image, Bounding Box, Segmentation Map, Depth Map, Painted Canvas.

hence a negative reward is supplied in this case as well. We supply a negative reward if the next L steps are not in the same semantic class forcing the next few strokes to paint a single object. All these rewards are modeled as losses in our final loss, as explained in section 4. Our approach, however, does make an assumption; we consider paintings of real objects and exclude abstract art from our analysis as such constraints do not make sense when painting an abstract idea.

We show experiments and results on two datasets - Virtual KITTI-2 [1] and the CUB-2011 birds dataset [16]. We chose Virtual KITTI because of the challenging nature of the data containing various semantic classes. Also, this dataset had segmentation and depth data along with the images. We also found that it was easy to see qualitative differences from the baseline on these datasets.

To summarize, the main contributions of this paper are as follows,

- We propose a novel *depth variance loss* to improve our model's depth perception while encouraging a single brush stroke to cover image regions with similar depth fields.
- We propose a novel *semantic class entropy* and *bundled class entropy loss* to penalize strokes traversing multiple semantic class boundaries.
- We propose a novel *class normalized L2 loss* to highlight the importance of objects regardless of the number of pixels they occupy.

2 Related Work

Traditional Techniques. A lot of work tries to leverage machine intelligence for non-photorealistic rendering and art [4, 11, 18]. Most of the work in this area has been around stroke-based rendering algorithms that place strokes on a canvas to create images, much like human painters do. Some very early techniques required intervention from users to decide the position and shape of the stroke. Unusual others involved creating a painting in layers by blurring images and changing brush parameters accordingly [7]. However, such techniques do not learn the underlying structure of images and rely on a fixed heuristic for their image generation rather than learning it.

Supervised Learning Approaches. The recent techniques are more autonomous and have employed several different architectures. Li et al. [13] leverage adversarial learning to produce contour drawings of images and require human-generated drawings as ground truth. Ha et al. [6] use an encoder-decoder architecture using RNNs called sketch-RNN to model a sketch as a sequence of strokes. The network learns from ground-truth stroke sequences. Our approach, however, does not depend on ground truth and hence can work with any image dataset.

Reinforcement Learning Techniques. To model the exploration and exploitation that humans use to learn any task, reinforcement learning techniques have been quite successful at this task [9, 10, 15, 17]. Zhou et al. [19] use a DQN to draw human-like doodles but has several limitations in terms of stroke possibilities due to the discrete action space.

Moving on to DRL approaches adversarially trained, Ganin et al. [3] proposed SPIRAL, which trains a policy network using WGAN (Wasserstein Generative Adversarial Network) loss between the target image and an image rendered by a graphics engine based on parameters supplied by the policy network. This approach has several limitations, including the non-differentiable renderer and sparse rewards based on only the final discriminator loss. Our approach calculates the discriminator loss at every step and also has a continuous action space. Therefore, we can incorporate long episodes and hence produce more complex drawings with several strokes.

Both SPIRAL and SPIRAL++[14] deploy a non-differentiable renderer (rather than a neural network environment), and the action space is discrete and hence less precise compared to a continuous space. Huang et al. [8] propose a model-based DDPG network as the agent along with a neural renderer, which resolves the issues of model-free approaches. The agent can understand the environment effectively and get access to the gradients from the environment along with the critic and WGAN loss. Like Huang et al. [8], we, too, follow a model-based approach. Our approach improves upon [8] by taking class segmentation and depth data of the image into account while painting the strokes. We leverage depth data and impose weak supervision on our model by forcing separate brush strokes at different depths. We also use semantic segmentation data and make sure a single stroke or a stroke bundle is painting a single class. This makes our model focus on objects better than previous models. We introduce a class-normalized L2 loss, which gives faster convergence than the Wasserstein-GAN loss used in [8] and observe better qualitative performance as a result.

3 Preliminaries

3.1 Problem Setup

Similar to Huang *et al.* [8] , we model the *"learning to paint"* problem as a reinforcement learning task with an underlying Markov Decision Process (MDP). The state $s_t \in \boldsymbol{\mathcal{S}}$ at any time step t, is given by the tuple (C_t, I, t), where C_t is current canvas image at time t and I represents the target image. At each time step, the agent takes action a_t depicting the parameters of the next brush stroke using a policy network $\pi(a_t|s_t)$.

3.2 Neural Renderer: Model-based vs Model-free Reinforcement Learning

Incorporating the environment model in reinforcement learning tasks helps improve the amount of feedback per action received by the reinforcement learning agent. Nevertheless, most state-of-the-art reinforcement learning methods use a model-free approach to avoid the variance introduced through simultaneous learning of both the environment model and the agent policy.

However, the model behind the painting task, *i.e.* computation of the next state (canvas) given the current state (canvas) and the action (brush stroke), is deterministic and straightforward to model. We hence, adopt a model-based approach wherein the environment model is learned by a separate neural network, which updates the current canvas given the brush stroke parameters. We will further outline the training procedure for the neural renderer in Section 5.

4 Our Method

We propose to incorporate semantic cues from semantic class segmentation and monocular depth estimation to improve the painting process's robustness. Introducing semantic constraints within the reinforcement learning agent's objective function also helps us step closer to the actual human painting process.

In the following sections, we describe our various attempts to bridge this gap, improve the quality of results, and facilitate scene understanding. In Section 4.1, we propose a depth variance loss to

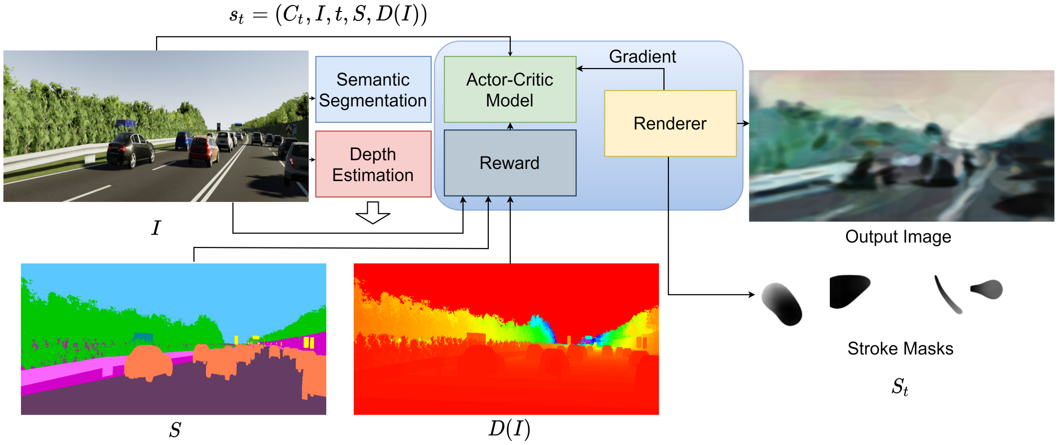

Figure 2: Basic structure, inputs, and outputs of the model. Our model will consider class segmentation and depth information when making stroke decisions.

encourage strokes covering similar depth fields. Section 4.2 introduces a penalty term for brush strokes traversing multiple object boundaries. Finally, in Section 4.4, we define a class normalization loss for normalizing the weightage of each semantic class, regardless of the raw pixel counts.

4.1 Depth Variance Loss

Depth perception is an important component of scene understanding and hence essential to the training process. We hypothesize that objects present at similar depth levels share a semantic relationship relevant to our perception of the overall image. Thus, objects present at similar depths should be given attention at similar points of the paining episode, or in other words, we want each stroke to paint image regions with similar perceived depths. For instance, use separate brush strokes when painting image foreground and background.

To discourage a single stroke from going onto different depths, we give a negative reward to a stroke with higher variance in its depth. The reward allows the model to imitate how a human would paint something. While painting, one stroke is canonically reserved for the same depth. Artists often use different colors and textures to demonstrate the effect of depth in paintings.

Given brush stroke parameters $\mathbf{a_t}$, a neural renderer model ϕ, depth prediction model $\mathcal{D}$, we propose to introduce the following depth penalty loss,

$$S_t = \{\mathbf{p} \mid \phi(\mathbf{a_t})[\mathbf{p}] \neq 0\} \tag{1}$$

$$L_{depth} = -\left(\frac{1}{N_t} \sum_{\mathbf{p} \in S_t} (\mathcal{D}_I[\mathbf{p}])^2 - \left(\frac{1}{N_t} \sum_{\mathbf{p} \in S_t} \mathcal{D}_I[\mathbf{p}] \right)^2 \right) \tag{2}$$

where, $N_t = \sum_{\mathbf{p}} \mathbb{1}_{[\mathbf{p} \in S_t]}$, is the number of pixels covered by the rendered brush stroke $\phi(\mathbf{a})$.

4.2 Semantic Class Entropy Loss

While the depth variance loss L_{depth} introduces some semantic-structure in the painting process, additional guidance is required to ensure that a brush stroke painting a given region should only remain inside the boundary of a particular semantic class. For instance, brush strokes painting a foreground car should not leak into the background road regions.

We call this phenomenon a semantic leakage and propose limiting it by introducing a novel class entropy loss L_{CE}. More specifically, we propose to measure the consistency of a particular brush stroke in focusing on a single semantic class as the negative entropy over the distribution of classes covered by the stroke pixels.

Mathematically, given action parameters $\mathbf{a}_t$, neural renderer ϕ, and a semantic class map $\mathcal{S}$, we define a discrete distribution over classes $\{\mathcal{C}_1, \mathcal{C}_2, \ldots \mathcal{C}_K\}$ covered by the stroke as,

$$S_t = \{\mathbf{p} \mid \phi(\mathbf{a_t})[\mathbf{p}] \neq 0\}, \qquad P(\mathcal{C}_i) = \sum_{\mathbf{p}\in S_t} \mathbb{1}_{[\mathcal{S}[\mathbf{p}]=\mathcal{C}_i]} \Bigg/ \sum_{\mathbf{p}} \mathbb{1}_{[\mathbf{p}\in S_t]} \tag{3}$$

The semantic class entropy loss is then defined as the entropy over the above distribution P as follows,

$$L_{CE} = -\sum_{i=1}^{K} P(\mathcal{C}_i)\ \log P(\mathcal{C}_i), \tag{4}$$

where K is the number of semantic classes.

4.3 Bundled Semantic Entropy Loss

Painting is a sequential process that requires long-term planning, which is one of the main reasons for adopting a reinforcement learning approach to this problem. To enforce the model to include further planning, we adopt an action bundle approach wherein the model at each time step is required to predict its next L actions or brush stroke parameters.

Such planning is important, especially under constraints on the overall number of strokes allowed for generating the painting. However, the presence of multiple semantic classes can act as a hindrance to this planning process. We thus propose stronger supervision by making the model plan the next L actions while painting inside only a single semantic class boundary. For instance, an agent who starts painting a car should completely focus on the next action bundle's car region.

We incorporate the concept of action bundles into the semantic class entropy penalty introduced in Eq. 4 by computing the discrete class distribution $P(C_i)$, over the entire action sequence. Mathematically, given action parameters $\mathbf{a}_t$, neural renderer ϕ, and a semantic class map $\mathcal{S}$, we define a discrete distribution over classes $\{\mathcal{C}_1, \mathcal{C}_2, \ldots \mathcal{C}_K\}$ covered by the stroke as,

$$S_t = \{\mathbf{p} \mid \phi(\mathbf{a}_t)[\mathbf{p}] \neq 0\}, \qquad P(\mathcal{C}_i) = \left(\sum_{t}^{t+L} \sum_{\mathbf{p}\in S_t} \mathbb{1}_{[\mathcal{S}[\mathbf{p}]=\mathcal{C}_i]}\right) \Bigg/ \sum_{\mathbf{p}} \mathbb{1}_{[\mathbf{p}\,\in\,\bigcup\{S_t, S_{t+1}, \ldots S_{t+L}\}]} \tag{5}$$

4.4 Class Normalized L2 Loss

The number of pixels occupied by that region will not directly determine the image region's relative importance. For instance, while the pixels representing the cars may cover less than a tenth fraction of the overall image, the presence of cars and couples of small traffic lights can completely change the perception of a scene from calm rural scenery to a busy urban traffic portrait.

Previous methods, like [8], attempt to overcome this limitation by replacing the L-2 loss with the Wasserstein-GAN (WGAN) [refer] distance. While the abstraction introduced by the WGAN penalty alleviates immediate focus on raw pixel values, we show that in the absence of any semantic information, the relative weight of a semantic class in predictions of the discriminator is still heavily affected by the associated number of pixels.

In this section, we propose to alleviate this problem by introducing a class normalized L-2 loss. More specifically, given a semantic class map $\mathcal{S}$, we define the class normalized L2 loss L_{CN} as,

$$S_t = \{\mathbf{p} \mid \phi(\mathbf{a})[\mathbf{p}] \neq 0\}, \tag{6}$$

$$L_{CN} = \sum_{i=1}^{K} \sum_{\mathbf{p}\in S_t} \left(I[\mathbf{p}] - C_t[\mathbf{p}]\right)^2 \Bigg/ \sum_{\mathbf{p}\in S_t} \mathbb{1}_{[\mathcal{S}[\mathbf{p}]=\mathcal{C}_i]}. \tag{7}$$

4.5 Complementary Discriminator

The WGAN for computing the step reward is trained in a conditional manner. That is, given the target image I_{target} and the current canvas C_t, the WGAN is trained with the following data,

$$x_{real} = [I_{target}, I_{target}] \qquad x_{fake} = [I_{target}, C_t] \tag{8}$$

However, the above configuration allows for convergence to a trivial solution where the discriminator detects the data to be real if both real and generated images are the same and give a *fake* prediction otherwise. We overcome the above mentioned trivial solution by taking a complementary discriminator approach [14]. More specifically, given target image I_{target} and the current canvas C_t, we define the training data for the WGAN to be,

$$x_{real} = [\mathbf{m} \odot I_{target}, (1 - \mathbf{m}) \odot I_{target}] \qquad x_{fake} = [\mathbf{m} \odot I_{target}, (1 - \mathbf{m}) \odot C_t], \tag{9}$$

where, the $\mathbf{m}$ is a binary mask, generated randomly such that $P(\mathbf{m} = 1) = \kappa$. κ is a hyper-parameter and is usually chosen in the range 0.5 ± 0.1.

5 Experimental Design

Model Design. Fig 2 illustrates the model pipeline. The agent is a model-based DDPG (Deep Deterministic Policy Gradient) network with an actor network $\pi(s)$ and a critic network $Q(s, a)$. The actor network is modeling the policy, which maps state s_t to action a_t. Since we follow a model-based approach, the final loss needs to be differentiable concerning the input. We achieve this by concatenating the semantic and depth maps to the input state tuple. Thus, the state s_t at time step t is given by $(C_t, I, t, \mathcal{S}_I, \mathcal{D}_I)$, where C_t is current canvas image at time t, I represents the target image, $\mathcal{S}_I$ is the class segmentation map and $\mathcal{D}_I$ represents the depth map for the target image.

The total loss of the agent is given by,

$$L = k_1\, L_{depth} + k_2\, L_{CE} + k_3\, L_{CN} + k_4\, L_{WGAN} \tag{10}$$

Note that we use $(k_1, k_2, k_3, k_4) = (1, 0, 0, 1)$ for depth variance results shown in Fig. 3, $(k_1, k_2, k_3, k_4) = (0, 1, 0, 1)$, for semantic penalty results shown in Fig. 4 and $(k_1, k_2, k_3, k_4) = (0, 0, 1, 0)$ for class normalization results shown in Fig. 5. We finally distribute the overall reward among different timesteps in an episode as follows [8],

$$r(s_t, a_t, s_{t+1}) = L_t - L_{t-1} \tag{11}$$

Neural Renderer. The neural renderer models the environment our actor-critic reinforcement learning model interacts with. The renderer is an FCN (Fully Convolutional Network) model that inputs the stroke parameters (a_t) from the actor and outputs the strokes' rendered image. We pre-train the renderer to produce these strokes, given the stroke parameters. The parameters of the stroke define the color, shape, position, and transparency of the stroke. In this paper, we use quadratic Bezier curves to model the strokes as used in [8]. The training of the renderer is done once and is independent of the dataset used. The FCN model is trained to generate the stroke masks for random stroke parameters while matching the label strokes masks (generated in a non-differentiable manner).

Dataset. We show our results on the high-fidelity synthetic dataset Virtual KITTI (VKITTI) [1] dataset. VKITTI contains 21260 images (frames) taken from 50 videos. We use the frames and their corresponding segmentation and depth data. We use 2000 images for testing and the rest for training.

6 Results

6.1 Depth Variance Loss

Fig. 3 shows the results with the depth variance loss. We see that the generated canvases are painted with a lot more granularity and consists of several distinct strokes at the same depth level. While this is desirable for some datasets, we found that this loss's benefits depend on the semantic class painted. For instance, the depth loss leads to more granularity for regions like the trees. However, it is not effective for semantic classes traversing a wide range of depths. For *e.g.*, while the road region belongs to a single semantic class, it has a varying depth structure. The introduction of depth loss makes the agent paint the road in multiple smaller strokes, which reduces the agent efficiency and leads to discontinuous perception results.

6.2 Semantic Penalty Loss

Fig. 4 shows the results with the semantic variance loss. Comparing with the baseline, we can see the canvases have higher levels of clarity. Besides, we can detect different objects better than ones

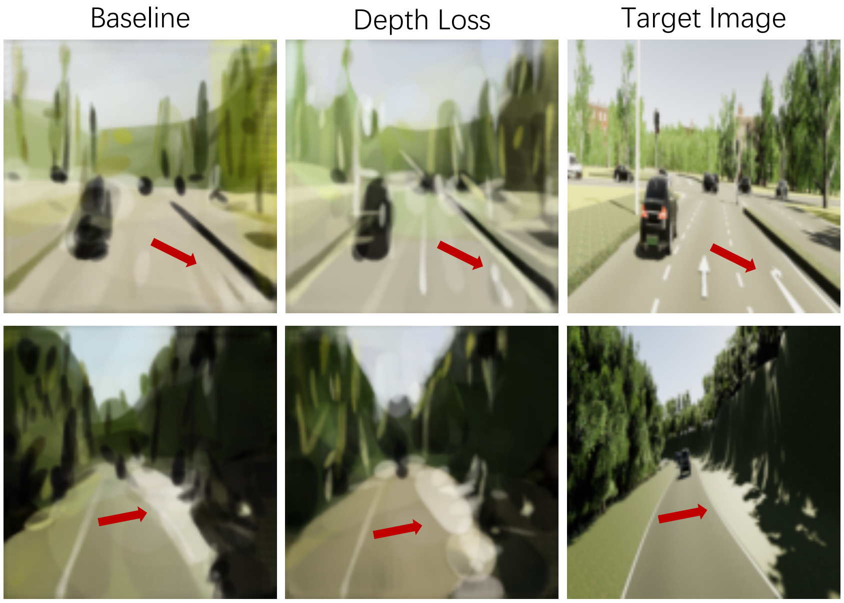

Figure 3: **Depth variance results.** (**Top row**) Introducing depth variance loss leads to better focus on granular details like road markings. (**Bottom row**) However, we also observe that the depth loss is not appropriate for semantic surfaces which are otherwise continuous but possess a high variance in depth scores. Thus, in the second example we can see that using depth loss makes the agent paint sidewalk (beside the road) through multiple distinct strokes each with a low depth.

generated without the semantic penalty loss. The penalty on crossing different items leads to a more identical boundary, just like enhancing pictures' quality. With the additional information, our model tends to use thinner but sharper strokes to demonstrate the contour.

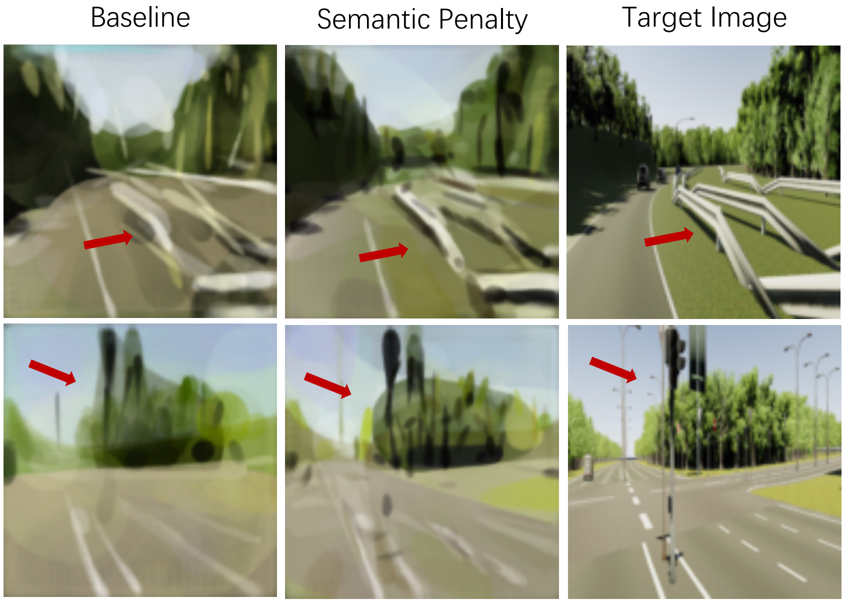

Figure 4: **Semantic penalty results.** (**Top row**) By discouraging brush strokes traversing multiple object boundaries, our novel semantic penalty loss, unlike the baseline, paints a much accurate sketch for the gaurdrails. (**Bottom row**) Similarly, we also observe that our method generates canvases with much better distinction between semantic boundaries. Here, our method by drawing brush strokes only within a particular class achieves a much better sketch for the dense vegetation and trees.

6.3 Class Normalized L2 Loss

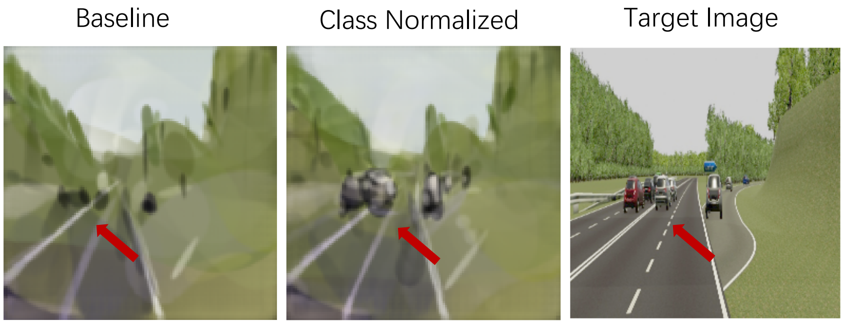

Figure 5: **Class normalized L2 loss results.** We clearly see that normalizing the L2 loss across each semantic class, leads to better focus on small object instances like cars (as shown in the above example). The generated canvas, thus with a more uniform focus on each semantically important region provides a much closer depiction of the target image. Note that the baseline hardly depicts the presence of vehicles on the road.

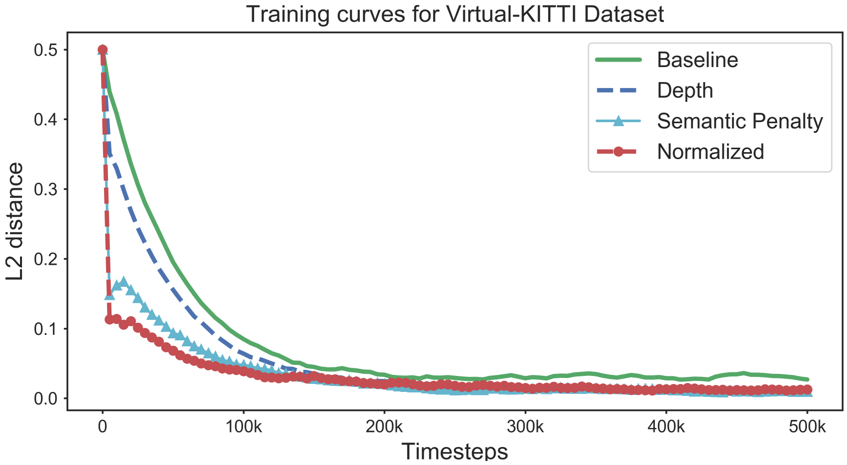

Figure 6: **Comparing training performance.** We clearly see that the introduction of each new semantic supervision loss (depth, class entropy, normalized L2 loss) leads to a faster reduction in the L2 distance (better sample complexity) between the target image and the final generated canvas. We also note that while the final L2 loss might be somewhat similar, the generated images are qualitatively different (refer Fig. 3,4,5)

7 Discussion and Conclusion

In this paper, we present an approach to utilize semantic and depth cues while teaching a reinforcement learning agent "how to paint". Building on a human painter's intuition, we define several losses that provide better supervision reward for the reinforcement learning agent and help ensure a semantic painting sequence. Our method leads to faster learning of the painting agent and the development of semantically better images.

While better from a qualitative perspective, the generated paintings have much room for improvement. Some possible directions for improvement include introducing hard constraints by changing the underlying Markov Decision Process (MDP) to separately focus on each semantic component. Another possible research direction would be to use a shared network to extract features for semantic segmentation, depth estimation, and learning to paint tasks. Since the tasks have a clear semantic connection, learning common features should benefit the RL agent's overall scene understanding.

References

[1] Y. Cabon, N. Murray, and M. Humenberger. Virtual kitti 2, 2020.

[2] P.-Y. Chen, A. H. Liu, Y.-C. Liu, and Y.-C. F. Wang. Towards scene understanding: Unsupervised monocular depth estimation with semantic-aware representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2624–2632, 2019.

[3] Y. Ganin, T. Kulkarni, I. Babuschkin, S. Eslami, and O. Vinyals. Synthesizing programs for images using reinforced adversarial learning. *arXiv preprint arXiv:1804.01118*, 2018.

[4] J. Grey. Human-computer interaction in life drawing, a fine artist's perspective. In *Proceedings Sixth International Conference on Information Visualisation*, pages 761–770. IEEE, 2002.

[5] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation. *International Journal of Computer Vision*, 112(2):133–149, 2015.

[6] D. Ha and D. Eck. A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*, 2017.

[7] A. Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 453–460, 1998.

[8] Z. Huang, W. Heng, and S. Zhou. Learning to paint with model-based deep reinforcement learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8709–8718, 2019.

[9] B. Jia, J. Brandt, R. Mech, B. Kim, and D. Manocha. Lpaintb: Learning to paint from self-supervision. *arXiv preprint arXiv:1906.06841*, 2019.

[10] B. Jia, C. Fang, J. Brandt, B. Kim, and D. Manocha. Paintbot: A reinforcement learning approach for natural media painting. *arXiv preprint arXiv:1904.02201*, 2019.

[11] M. King. Programmed graphics in computer art and animation. *Leonardo*, 28(2):113–121, 1995.

[12] C. Li, A. Kowdle, A. Saxena, and T. Chen. Towards holistic scene understanding: Feedback enabled cascaded classification models. In *Advances in Neural Information Processing Systems*, pages 1351–1359, 2010.

[13] M. Li, Z. Lin, R. Mech, E. Yumer, and D. Ramanan. Photo-sketching: Inferring contour drawings from images. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1403–1412. IEEE, 2019.

[14] J. F. Mellor, E. Park, Y. Ganin, I. Babuschkin, T. Kulkarni, D. Rosenbaum, A. Ballard, T. Weber, O. Vinyals, and S. Eslami. Unsupervised doodling and painting with improved spiral. *arXiv preprint arXiv:1910.01007*, 2019.

[15] R. Nakano. Neural painters: A learned differentiable constraint for generating brushstroke paintings. *arXiv preprint arXiv:1904.08410*, 2019.

[16] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

[17] N. Xie, H. Hachiya, and M. Sugiyama. Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting. *IEICE TRANSACTIONS on Information and Systems*, 96(5):1134–1144, 2013.

[18] S. Xu, F. C. Lau, W. K. Cheung, and Y. Pan. Automatic generation of artistic chinese calligraphy. *IEEE Intelligent Systems*, 20(3):32–39, 2005.

[19] T. Zhou, C. Fang, Z. Wang, J. Yang, B. Kim, Z. Chen, J. Brandt, and D. Terzopoulos. Learning to sketch with deep q networks and demonstrated strokes. *arXiv preprint arXiv:1810.05977*, 2018.