CS231N Final Report: Style Transfer Incorporating Depth Perception

Xu, YingHao
Stanford University
ericx@stanford.edu

Zhang, Yun
Stanford University
yunzhang@stanford.edu

Yang, Yuxin
Stanford University
yuxiny@stanford.edu

1 Introduction

Image stylization with deep networks has been a vibrant area of research in recent years. With the help of deep Convolutional Neural Networks (CNN) we are able to transfer the styles of famous paintings to any images, adding a touch of artistic feel to our photos. Such applications are widely used in photo retouching Apps, or digital media productions.

However, one downside of such CNN-based transformation is that the original content of the images cannot be well preserved. Specifically, after transferring the texture and strokes of a painting to our image, the image looks cluttered and depth information of the image is mostly discarded. Instead, the existing style transfer networks mainly put emphasis on the graphical feature including colors, edges or patterns. Although artistic style is a rather subjective topic, depth prediction, the important missing piece of a picture, is largely agreed to be important to an aesthetically pleasing perception when it comes to our daily photos. Not only is depth an important piece of information for art consumers like us, it is also an extremely important information for visually impaired people to better understand art pieces [4]. Some of the ways to process images before to include depth information include purposely blur the background[2] to achieve depth or using different color tones[3]. In this project, we explore ways to incorporate depth information into existing style transfer algorithm in order to achieve a cleaner and more realistic retouch of images with deep learning CNN networks.

2 Related Work

2.1 Image Style Transfer

The origin of the idea of style transfer with CNN can be traced to Gatys *et al*'s work[6], in which a style transfer problem is formulated into an optimization problem. Gatys *et al* define two loss terms, one for style and one for content, which are evaluated in high-level feature space as the distance between feature representations of output image and that of content/style images. The feature representations are extracted from different layers of a VGG [12] network. The intuition behind is that the VGG network, originally developed for image classification and localization, can effectively learn the content representation of images. In addition, to represent stylistic features, Gatys *et al* introduce the use of a Gram Matrix that essentially calculates the covariance of different feature channels. The weighted sum of both loss terms of the network is optimized with forward and backward passes. The computational cost of this method is very high due to the explicit optimization and the generation can take a long time because each content and style image pair would require retraining the network.

Later Johnson *et al* [1] devised an image transformation network that largely relieved the computational cost while producing reasonably good transfered images. Instead of solving the optimization problem posed in [6] explicitly, Johnson trained a deep feed-forward residual network as the transformation network with a pretrained VGG network as its fixed loss network. The transformation network would need to be trained for each different style image in order for the transformation network to learn to transfer the style to input content images. After training the network for a particular style image, generating new image would only need a forward pass of the image transformation network. The quality of the transfered images are close to that of Gatys *et al*'s method. Subsequently, there have been several other techniques introduced to improve the transfered image quality of a image transformation network such as Ulyanov's instance normalization [19].

2.2 Depth Estimation with Deep Networks

In recent years deep neural networks are applied to predict depth in a single image. This is a particular difficult task if the goal is to estimate absolute depth because gathering correctly labeled training data in a large amount would be hard. However, in many real applications we only need relative depth information. Eigen *et al* [15] proposed an effective multi-scale convolutional network that performs well on both indoor and outdoor scenes. Laina *et al* [7] later proposed a new deep network for the same task that achieves slightly better results. There are several other pretrained networks available online, including a network trained with cGAN (Conditional Generative Adversarial Networks) model for monocular depth perception [9], “Res-Net-UpProj” trained on NYU Depth v2 & Make 3D[5]. In the latter paper, they learned the mapping between a single RGB image and its corresponding depth map using a CNN.

3 Methodology

Our approach to incorporate depth information in style transfer is similar to Johnson's[1] work, in which an image transformation network is trained using perceptual losses that considers not only style and content, but also depth. Similar to Johnson's work, the system consists of two components - image transformation network and perceptual loss network. But now the perceptual loss network contains an additional pretrained depth network for computing depth loss of generated images compared to the content images. Our implementation of the overall network architecture is adapted from two public github repositories: Lengstorm's implementation of Johnson's fast neural transfer [18] and iro-cp's implementation of FCRN depth prediction network[8].

3.1 Image Transformation Networks

The first part is an image transformation network f_W, which is a deep residual CNN parametrized by weights W. It transforms input image x to output image $\hat{y}$ via equation: $\hat{y} = f_W(x)$. This part of the network largely follows the same architecture in Johnson's [1] except that the spatial batch normalization is replaced by Ulyanov's instance normalization [19] due to its proven improvement on image quality.

3.2 Perceptual Loss Functions

The second part of the system, the perceptual loss network consist of a VGG network ϕ for style and content losses, plus a depth prediction network Ψ for depth loss. The goal is to train the image transformation network so that the weighted combination of the three loss functions is minimized:

$$W^{\star} = \arg\min_{W} E_{x,y_i}[\sum_{i=1} \lambda_i l_i(f_W(x), y_i)]$$

In our project, specifically, we construct our weighted combination of loss functions to be:

$$\begin{aligned} W^{\star} = \arg\min_{W} E(\lambda_1 l^{\Phi,j}_{style}(\hat{y}, y_{style}) + \lambda_2 l^{\Phi,j}_{content}(\hat{y}, y_{content}) + \\ \lambda_3 l^{\Psi,j}_{depth}(\hat{y}, y_{content})) \end{aligned} \tag{1}$$

Where Φ is the pretrained VGG loss network, Ψ is the depth loss network. $\Phi_j(x)$ and $\Psi_j(x)$ are the activations of the jth layer of the network Φ and Ψ (j is not necessarily the same in different loss networks). If j is a convolutional layer then $\Phi_j(x)$ and $\Psi_j(x)$ will be a feature map of shape $C_j \times H_j \times W_j$. G is the gram matrix defined in Johnson's paper.

3.2.1 Loss Network - Style Loss

The loss network, originally developed by Johnson[1], is used to define targets for high level features loss functions $l_1 \dots l_k$. We pass both the generated image and the style target image through the loss network ϕ. And we take the difference, which is denoted as $l_i(y, y_i)$, between the output of the generated image $\hat{y}$ and the target image y_i at several layers from the network to be the losses we have for the generated image.

The feature reconstruction loss penalizes the generated image y when it deviates in content from the target y. We also wish to penalize differences in style: colors, textures, common patterns, etc. To achieve this effect, Gatys propose the following style reconstruction loss [6].

$$l_{style}^{\Phi,j}(\hat{y}, y_{style}) = \|G_j^{\Phi}(f_W(x)) - G_j^{\Phi}(y_{style})\|_F^2 \tag{2}$$

In our project, we chose to use the output of Relu 1-1, 2-1, 3-1, 4-1, and 5-1 as the outputs that we compute loss from.

3.2.2 Loss Network - Content Loss

The content loss, similar as the the style loss is defined by the difference of outputs between higher level features extracted from passing the generated image and the content image at some higher level in the loss network.

As Johnson mentioned in his paper, minimizes the feature reconstruction loss for early layers tends to produce images that are visually indistinguishable from y [1]. Therefore, we reconstruct from higher layers. So that the image content and overall spatial structure are preserved but color, texture, and exact shape are not. Using a feature reconstruction loss for training our image transformation networks encourages the output image y to be perceptually similar to the target image y, but does not force them to match exactly.

The exact loss is computed by:

$$l_{content}^{\Phi,j}(\hat{y}, y_{content}) = \frac{1}{C_j H_j W_j}\|\Phi_j(f_W(x)) - \Phi_j(y_{content})\|_2^2 \tag{3}$$

Where Φ is the pretrained VGG loss network, and if j is a convolutional layer then $\Phi_j(x)$ and $\Psi_j(x)$ will be a feature map of shape $C_j \times H_j \times W_j$.

3.2.3 Depth Prediction Network

Following the idea in [1] that CNN trained for classification tasks has already learned how to represent high-level features in images, the style and content losses are formulated as the difference in feature space between our output image and the input style and content images given by a VGG network[12]. Inspired by this use of perceptual loss network, we hypothesize that a deep CNN specializing well in single image depth estimation, such as the one developed by Eigen *et al* [15] would be very capable of capturing depth information in both high-level feature space and pixel space.

Based on these hypotheses, we define the depth loss to be the differences of original and the generated image in some feature space given by a single image depth estimation network. To be specific, we define a new term - the depth loss, by computing the difference between the depth predicted from the original content image and the newly generated image's depth by a pre-trained depth perception network.[7]. We compute the depth loss by the following formula:

$$l_{depth}^{\Psi,j}(\hat{y}, y_{content}) = \frac{1}{C_j H_j W_j}\|\Psi_j(f_W(x)) - \Psi_j(y_{content})\|_2^2 \tag{4}$$

Where Ψ is the depth loss network. If j is a convolutional layer then $\Phi_j(x)$ and $\Psi_j(x)$ and $\Omega_j(x)$ will be a feature map of shape $C_j \times H_j \times W_j$. G is the gram matrix defined in Johnson's paper.

3.2.4 Overall System

Now the structure of our proposed method, as shown in Fig 3, consists three components: a image transformation network, a VGG network for style and content losses, and a depth network for depth loss. With our new system set up, we trained the image transformation network to minimize the newly defined weighted loss.

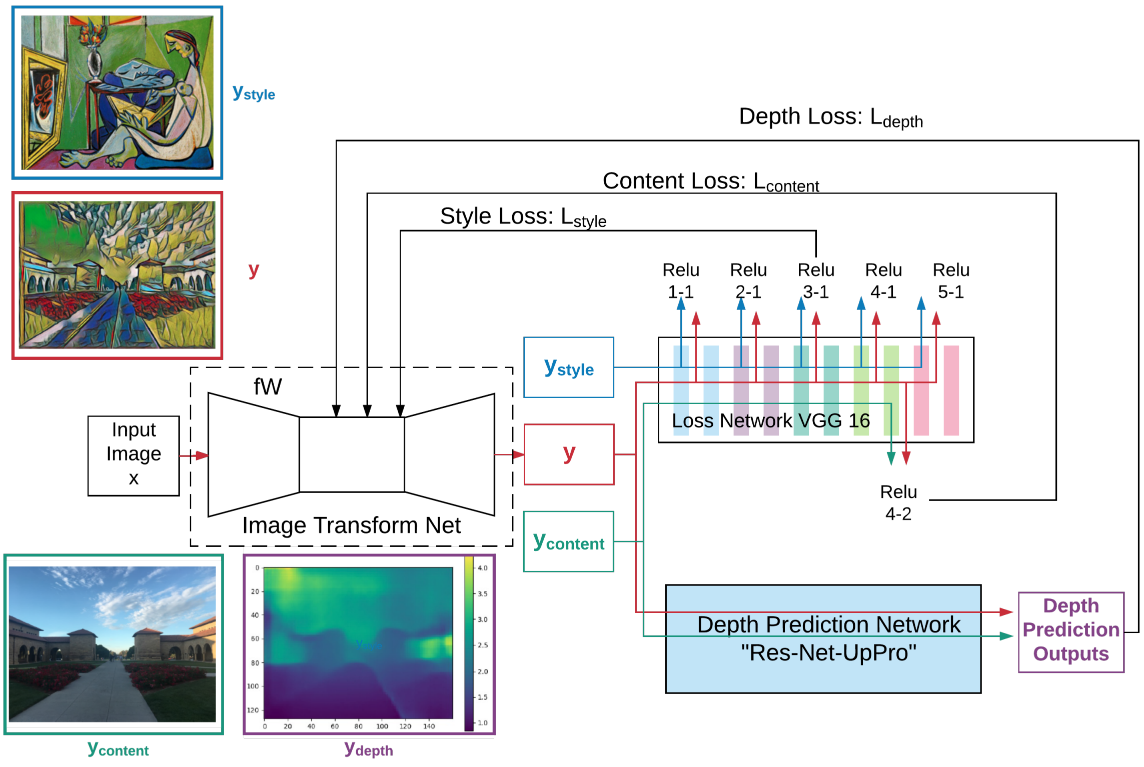


Figure 1: System overview - We train an image transformation network to transform input images into output images. We use a loss network pretrained for image classification to define perceptual loss functions that measure perceptual differences in content, style and depth prediction results between generated images and training images. The loss network and depth prediction networks remained fixed during the training process.

4 Dataset and Features

In this project, we utilized two ready to use CNNs to generate training information for image transformation network.

4.1 Obtained Models

4.1.1 VGG-19 for Image Classification

One of which is the VGG-19 loss network pre-trained for image classification purpose[18]. This network was developed by Logan Engstrom in his project "Fast Style Transfer in TensorFlow". His implementation is based off of a combination of Gatys' A Neural Algorithm of Artistic Style [6], Johnson's Perceptual

Losses for Real-Time Style Transfer and Super-Resolution [1], and Ulyanov's Instance Normalization [19]. [18]. Engstrom's implementation uses TensorFlow to train a fast style transfer network, which is the same transformation network as described in Johnson, except that batch normalization is replaced with Ulyanov's instance normalization [19], and the scaling/offset of the output tanh layer is slightly different. The loss function he used was close to the one described in Gatys, using VGG19 instead of VGG16 and typically using "shallower" layers than in Johnson's implementation (e.g. we use relu1-1 rather than relu1-2).

4.1.2 Depth Prediction Network

For depth prediction network, we obtained another pre-trained model called "ResNet-UpProj" [7]. It is a model developed by By Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, Nassir Navab. They trained their model to learn the mapping between a single RGB image and its corresponding depth map using a CNN, first, by introducing a fully convolutional architecture to depth prediction, endowed with novel up-sampling blocks, that allows for dense output maps of higher resolution and at the same time requires fewer parameters and trains on one order of magnitude fewer data than the existing state of the art. In their work they thoroughly evaluate the influence of the network's depth, the loss function and the specific layers employed for up-sampling in order to analyze their benefits. [7]

4.1.3 Microsoft COCO

For the training of our image transformation network, We used Microsoft COCO [11] dataset.

Microsoft COCO is a new image recognition, segmentation, and captioning dataset. COCO has several features: Object segmentation, Recognition in Context, Multiple objects per image, and 80 object categories. There are in total of 80k training images in the COCO dataset. The images we used has been resized to 256 x 256. And then, we train with a batch size of 4 for 20k iterations. This allows simply 1 epoch over the training data [11].

Some example images are:

Figure 2: Microsoft COCO sample training datasets examples [11]

5 Experiments and Results

In our experiment, we used the following painting as the target style image. We generated output images with styles that approach this target:

Then, with this target image, we computed the following qualitative baseline of style transfer that did not incorporate depth information shown in Fig 4b, 4e, 4h, 5b, 5e and 5h.

5.1 Style Transfer Quantitative Baseline

Quantitatively, one way to validate that what we are doing is of good importance is by performing an experiment in which we fed a stylized image by Johnson [1] into the depth estimation network by Liana[7] to see whether the resulting depth map presents meaningful depth information.

As an example, we present here two depth maps done on images after style transfer without depth information incorporated. As can be seen from Fig 6c, and Fig 6g, after style transferring, the relative depth relationship of different objects in the image is barely preserved, and the depth map after style transfer

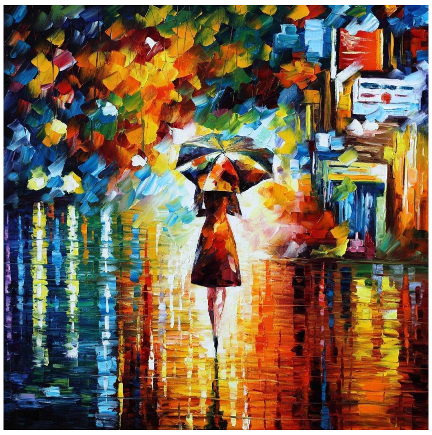
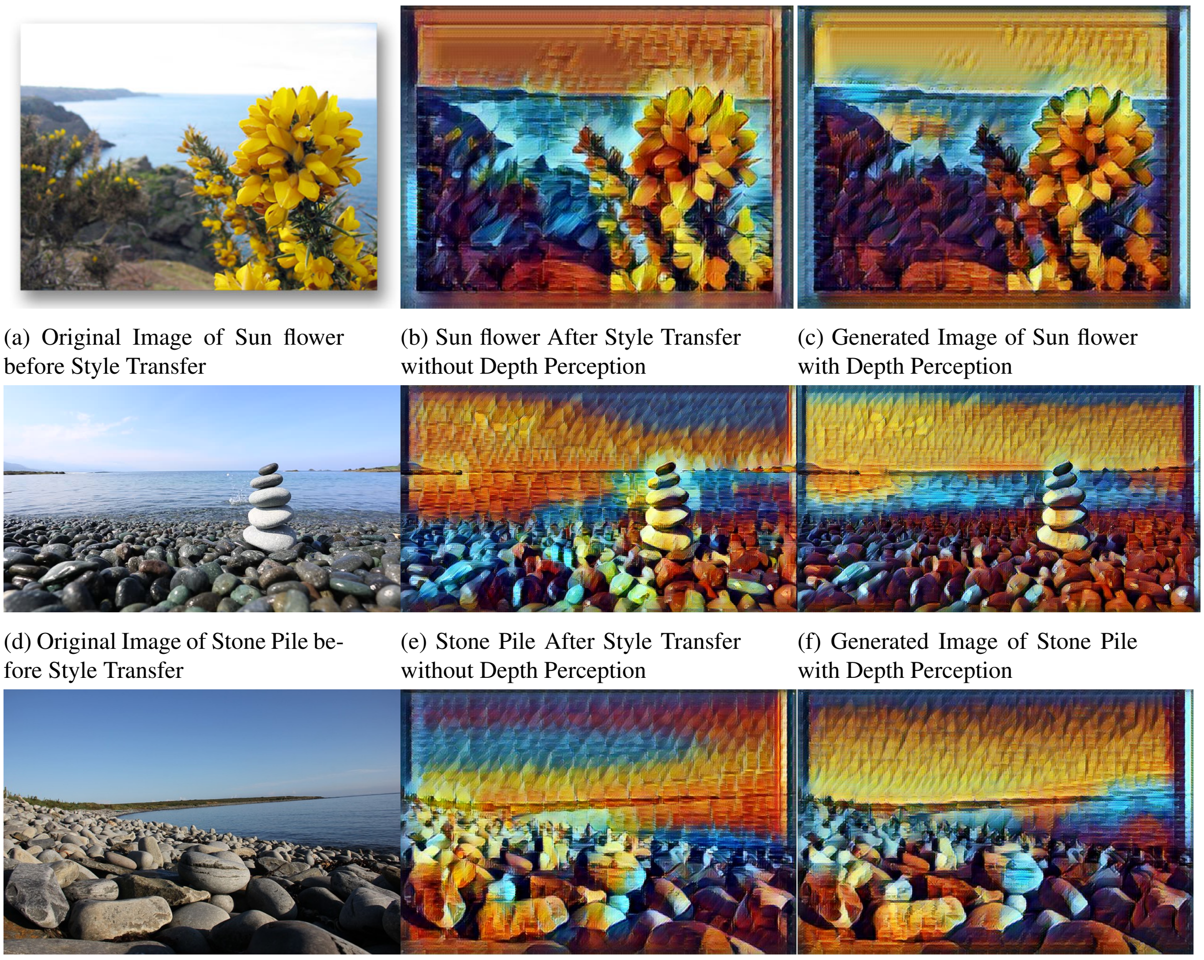


Figure 3: Target style image [18]


(a) Original Image of Sun flower before Style Transfer

(b) Sun flower After Style Transfer without Depth Perception

(c) Generated Image of Sun flower with Depth Perception

(d) Original Image of Stone Pile before Style Transfer

(e) Stone Pile After Style Transfer without Depth Perception

(f) Generated Image of Stone Pile with Depth Perception

(g) Original Image of Beach before Style Transfer

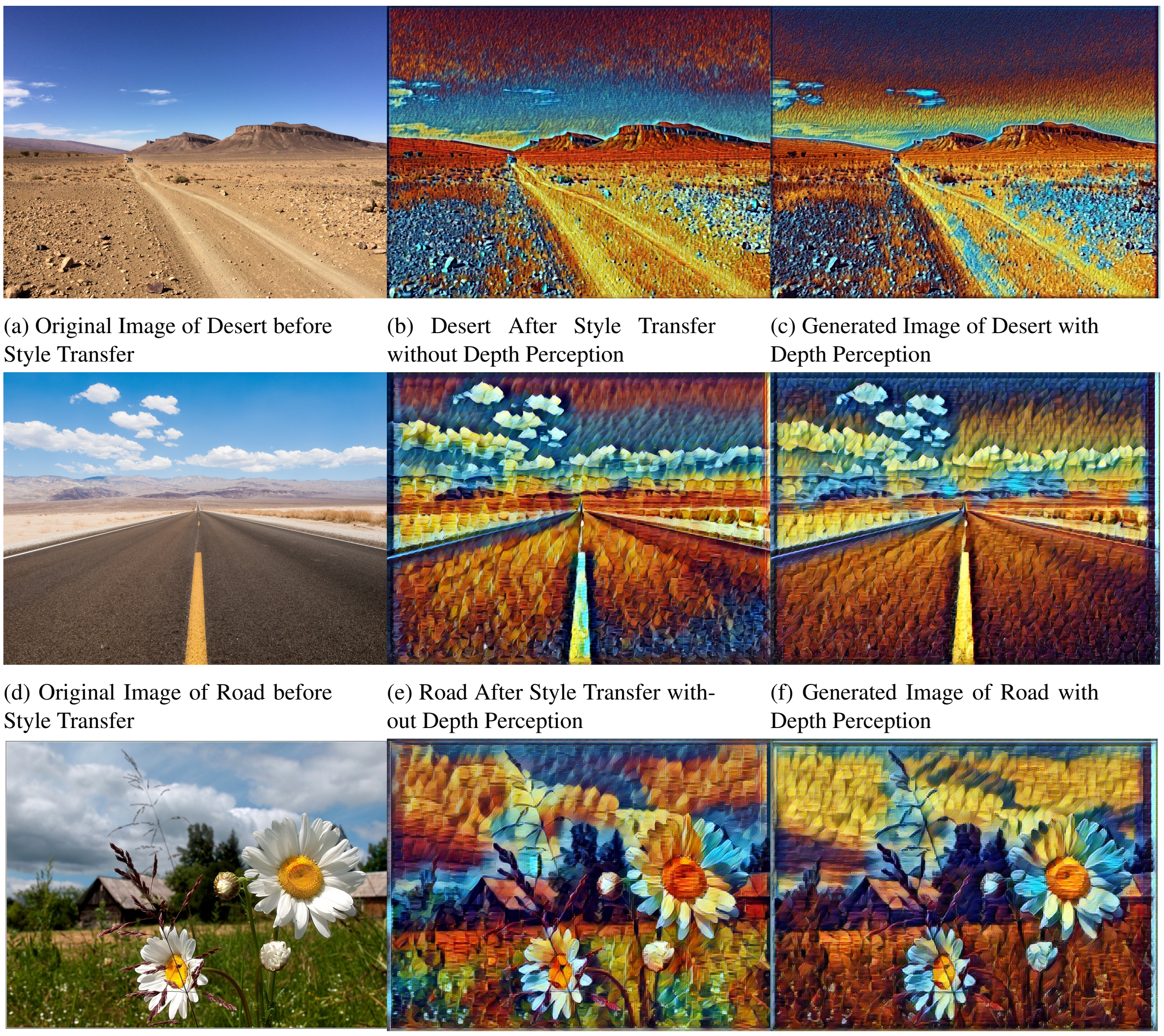
(h) Beach After Style Transfer without Depth Perception

(i) Generated Image of Beach with Depth Perception

Figure 4: Comparison of original images, qualitative baseline, and our qualitative results (1)

becomes blurry and incorrect. It is visually very obvious that in the depth information is heavily distorted.

After this experiment, we arrived at the conclusion: Some style model would greatly distort depth information of the original image. And this can be the cause that some of the stylized images are confusing and visually unpleasant. With this set as our quantitative baseline, we will, with our model, try to produce stylized images with better depth prediction when going through the depth perception network.



(a) Original Image of Desert before Style Transfer

(b) Desert After Style Transfer without Depth Perception

(c) Generated Image of Desert with Depth Perception

(d) Original Image of Road before Style Transfer

(e) Road After Style Transfer without Depth Perception

(f) Generated Image of Road with Depth Perception

(g) Original Image of Flowers before Style Transfer

(h) Flowers After Style Transfer without Depth Perception

(i) Generated Image of Flowers with Depth Perception

Figure 5: Comparison of original images, qualitative baseline, and our qualitative results (2)

5.2 Training Process

We implemented the architecture we presented in the "Method" section. In order to achieve the best performance, we had to fine tune parameters including content_weight=7.5e0, style_weight=1e2, tv_weight=2e2, depth_weight=2e2, and learning_rate(1e-3). Using Microsoft COCO dataset, we trained our model for 2 epochs, which is approximately 40,000 iterations. The training time on Tesla K80 was around 22 hours.

By setting checkpoint every 2000 iterations, we monitored the change of each of the four losses: content_loss, style_loss, tv_loss and depth_loss. The addition of depth_loss to the totoal perceptual loss affected the minimization of the original three losses, which caused the other three losses to fluctuate during minimization. Depth_loss fluctuated during training as well. The reason may be that the depth prediction network we used was not trained on Microsoft COCO dataset, so the accuracy of depth_loss was negatively affected.

5.2.1 Preparing Depth Training Data

In order to train our network, we had to modify the two existing models. We feeded the output from the last layer of depth prediction model into Lengstrom's fast-style-transfer model in Tensorflow, which uses shallower layers of VGG19 instead of VGG16, instance normalization instead of batch normalization. The reason we chose to extend Lengstrom's model instead of Johnson' is that we found that Lengstrom's structure itself can preserves depth better than Johnson's model with similar image quality.

5.3 Final Output

5.3.1 Qualitative

Qualitatively, as we can tell from the images in Fig 4c, 4f, 4i, 5c, 5f, and 5i, our newly generated depth incorporated graphs has a much deeper penetration in depth. Also, as in the shadow and the contrast between light and darkness, our model does a much better job in predicting the color that is supposed to be in style transfer. For example, comparing between Fig 4e and 4f, we can see that the light around the stone piles seems much more natural and smooth in our generated image with depth information. In the baseline picture, the color of the stones on the left lower corner is completely bright blue, which gives less of a contrast and smooth gradient than our predicted image. In our generated image, each stone on the beach is so well processed to the finest details.

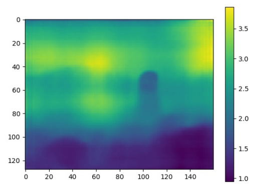
Similarly, as we observe in Fig 4h and 4i, we can see the the depth of the sky is much more wholesomely preserved in our generated image compared with the baseline without the depth information. The contrast between the yellow sky and the blue water makes much more aesthetic sense, and increases the perceived beauty of this landscape tremendously.

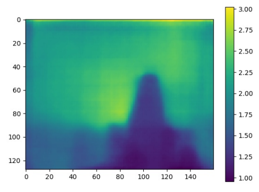
5.3.2 Quantitative

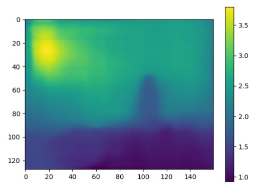
Quantitatively, we produced the depth maps for the two beach pictures we mentioned above by feeding the style transfer results from our newly trained neural network back to depth prediction network [7]. And obtained the following Fig 6d, 6h.

As we can see that on both images, the depth map produced after passing the generated images from

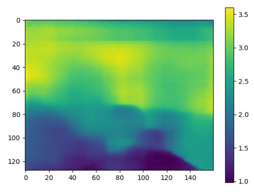
(a) original stone depth map

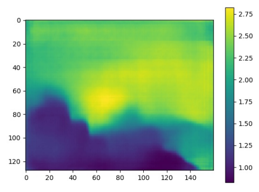

(b) original stone depth map


(c) without depth transfer stone depth map


(d) with depth transfer stone depth map

(e) original stone depth map


(f) original beach depth map


(g) without depth transfer beach depth map

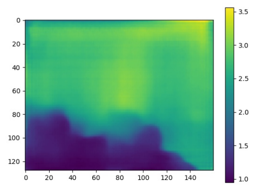

(h) with depth transfer beach depth map

Figure 6: Depth estimation of original image and images stylized by different models

our network resembles the depth map of the original image by a large portion. And, quantitatively, our generated images' depth map has much less loss when fed through the depth prediction network [8], than the performance of the previous model. The L2 difference of the depth map of our generated image and original image is about half of that between the previously generated image and original image. So we can further conclude that our model preserves a reasonably larger amount of depth information than the previous style transfer result. This further explains why our model is better at generating this complexity and penetration effect of images than the previous model.

6 Conclusion and Future Work

In the conclusion, in this project, based on the work by Johnson [1] and Laina [5] , we developed a deep CNN that further improves the style transfer quality by incorporating depth information in the training process of image transformation network. Our result not only shows that this model is implementable, but also demonstrated some high quality style transfer results which produced promising improvements on the network before. This success may inspire future research in improving quality of styles transfers by introducing even more parameters into styles transfers like segmentation information. Or we can even try to incorporate two sets of styles into one. If we had more time, we would try all of those implementations upon our dept perception model.

On the more personal level, in this project, not only did we gain a much deeper understanding of computer vision, convolutional neural network, we also became significantly more familiar with the tools to implement neural networks, like python numpy, and Tensorflow. We give great appreciation to all the TAs who helped us in the process, and especially Justin Johnson who has been a great help in pointing us to the right direction, and guiding us to obtain a clear scope for our project.

References

[1] Johnson, Justin, Alexandre Alahi, and Li Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution." European Conference on Computer Vision. Springer International Publishing, 2016.

[2] Mather, George. "Image blur as a pictorial depth cue." Proceedings of the Royal Society of London B: Biological Sciences 263.1367 (1996): 169-172.

[3] Rheingans, Penny, and David Ebert. "Volume illustration: Nonphotorealistic rendering of volume models." IEEE Transactions on Visualization and Computer Graphics 7.3 (2001): 253-264.

[4] Reichinger, Andreas, Stefan Maierhofer, and Werner Purgathofer. "High-quality tactile paintings." Journal on Computing and Cultural Heritage (JOCCH) 4.2 (2011): 5.

[5] Laina, Iro, et al. "Deeper depth prediction with fully convolutional residual networks." 3D Vision (3DV), 2016 Fourth International Conference on. IEEE, 2016.

[6] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge. "A Neural Algorithm of Artistic Style." arXiv:1508.06576

[7] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, Nassir Navab. "Deeper Depth Prediction with Fully Convolutional Residual Networks." arXiv:1606.00373

[8] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, Nassir Navab. "Deeper Depth Prediction with Fully Convolutional Residual Networks", https://github.com/iro-cp/FCRN-DepthPrediction.

[9] Kaihuchen. "Monocular Depth Perception with CGAN." All about Machine Learning... All about Machine Learning..., 22 Dec. 2016. Web. 12 June 2017.

[10] Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[11] Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Doll´ar, P., Zitnick, C.L.. "Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014." Springer (2014) 740–755

[12] Karen Simonyan, Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." arXiv:1409.1556

[13] Collobert, R., Kavukcuoglu, K., Farabet, C.. "Torch7: A matlab-like environment for machine learning." BigLearn, NIPS Workshop. Number EPFL-CONF-192376 (2011)

[14] Gross, S., Wilber, M.: Training and investigating residual nets. http://torch.ch/blog/2016/02/04/resnets.html (2016)

[15] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In Proceedings of the IEEE International Conference on Computer Vision, pages 2650–2658, 2015.

[16] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385 (2015)

[17] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proceedings of The 32nd International Conference on Machine Learning. (2015) 448–456

[18] Lengstrom. "Lengstrom/fast-style-transfer." GitHub. N.p., 20 Apr. 2017. Web. 12 June 2017, https://github.com/lengstrom/fast-style-transfer.

[19] Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Instance Normalization: The Missing Ingredient for Fast Stylization." arXiv preprint arXiv:1607.08022 (2016).

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012.

[21] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. CoRR, abs/1409.4842, 2014.