Naive Background Style Transfer

Vikram Shenoy
Khoury College of Computer Sciences
Northeastern University
Boston, MA
Email: shenoy.vi@husky.neu.edu

***Abstract*—Image Segmentation and Style Transfer have both been ground-breaking discoveries in the field of computer vision. Image Segmentation involves creation of segmentation maps that creates multiple fragments in the image for better representation of the image. Neural Style Transfer is an algorithm that incorporates the style and texture of one image (style image) onto another image (content image) by reducing their overall loss. This paper proposes a new strategy called Naive Background Style Transfer that merges the ideas of Semantic Segmentation and Neural Style Transfer to generate images with abstract backgrounds while the foreground in the original image remains untouched.**

Keywords: Computer Vision, Semantic Segmentation, DeepLabv3+, Neural Style Transfer

I. Introduction

Computer Vision is a sub-field of Artificial Intelligence that deals with gaining a deeper understanding of digital images. During the last few years theres been a surge in Computer Vision applications mainly due to the increasing research in Convolutional Neural Network architectures and the availability of high processing power. Style Transfer and Image Segmentation are just one of the many branches in this domain. Leon A. Gatys et al. introduced their algorithm, A Neural Algorithm of Artistic Style in 2015 [1] which seamlessly generated a stylized version of the content image based on the style image provided. The algorithm generated a lot of buzz for creating images stylized using famous paintings such as The Starry Night by Vincent Van Gogh and The Scream by Edvard Munch. Research in this field also led to the creation of the unique photo-editing mobile application, Prisma [2], which generates an artistic version of the image provided to it. Image segmentation is another widely used domain of Computer Vision which deals with highlighting objects or segments in the image using Neural Networks. From medical imaging to detect tumours [3] to self-driving cars for detecting pedestrians and objects [4], image segmentation is being deployed at the root of highly complex AI applications. Image Segmentation can be broadly classified into two types: Semantic Segmentation and Instance Segmentation. In this paper, we will be covering a semantic segmentation algorithm known as DeepLabv3+ [5]. Nave background style transfer fuses the ideas of Neural Style Transfer and DeepLabv3+ to create an image with the mentioned style incorporated into the background of the given image.

II. Related Work

In this paper we will discuss two important domains of computer vision: Style Transfer and Semantic Segmentation.

A. Style Transfer

Style Transfer is the process of incorporating the style and texture of one image onto another image so as to generate a new image which appears to be an artistic version of the given content. There have been many developments in Style Transfer, but for the scope of this project we will be focusing on the original neural style transfer algorithm by L. A. Gatys.

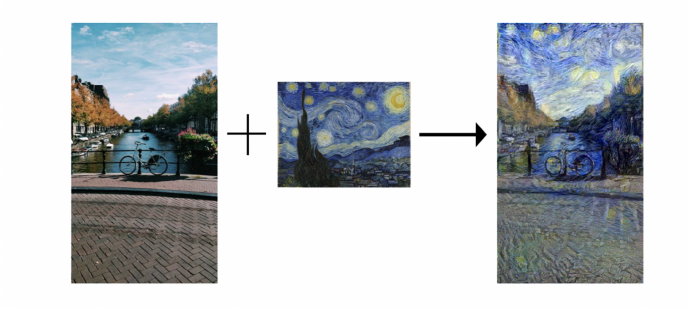

Fig. 1. Example of style transfer using The Starry Night by Vincent Van Gogh

1) Neural Style Transfer: Neural Style Transfer achieves the objectives of style transfer with the help of neural networks. In the original paper, the algorithm is built on top of the VGG19 network that is pretrained on Imagenet weights. The activations of one of the later layers of the VGG19 network are used as content activations whereas multiple layers throughout the network form the style activations. Using these activations the algorithm works towards minimizing the overall loss which is given as,

$$J(G) = J_{content}(C, G) + J_{style}(S, G)$$

Where, $J_{content}(C, G)$ is the content loss that measures how the similarity between the content of the content image and the generated image and $J_{style}(S, G)$ is the style loss which measures the similarity between the style of the style image and the generated image. and are weighting factors for the content and style reconstruction respectively.

Gradient descent minimization is used to minimize $J(G)$ and produce the generated image

$$G := G - \frac{\partial}{\partial G} J(G)$$

Now, lets dive deeper into each of the two loss functions mentioned above.

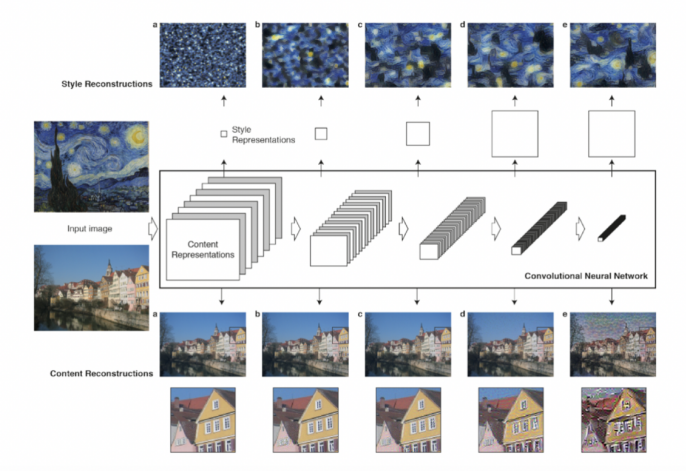

Fig. 2. Neural Style Transfer architecture as seen in the original paper by L. A. Gatys

Consider hidden layer l of the VGG19 Network. The activations of this layer for the content image are denoted as $a^{[l](C)}$ and the activations of this layer for the generated image are denoted as $a^{[l](G)}$. The content loss is given as,

$$J_{content}(C, G) = \frac{1}{2} ||a^{[l](C)} - a^{[l](G)}||^2$$

If $a^{[l](C)}$ and $a^{[l](G)}$ are similar, both images have similar content and the overall loss will be minimal.

Style loss is a slightly more complicated concept which deals with the correlation between different channels of the image. Suppose one of the channels corresponds to the neuron that has the highest activation when a vertical edge is detected and another channel corresponds to the neuron that has the highest activation when a blue hue is detected. If the channels are correlated, the part of the image having the vertical edge will also have the blue hue. If the two channels are uncorrelated the part of the image having a vertical edge will not have a blue hue. This gives us a measure of similarity in the style of the style image as compared to the style of the generated image. These correlations can be captured with the help of a Gram matrix or a Style matrix. Now, $a^{[l]}_{(i,j,k)}$ is the activation at (i, j, k) for hidden layer l where i traverses the height, j traverses the width, and k traverses the channels of the feature map. $G^{[l](S)}$ is a matrix with dimensions as $n_c^{[l]}$ x $n_c^{[l]}$, where $G^{[l](S)}$ is the Gram matrix for the style image at hidden layer l.

The Gram matrix is computed as follows,

$$G^{[l](S)}_{kk'} = \sum_{i=1}^{n_h^{[l]}} \sum_{j=1}^{n_w^{[l]}} a^{[l]}_{(i,j,k)} . a^{[l]}_{(i,j,k')}$$

$$G^{[l](G)}_{kk'} = \sum_{i=1}^{n_h^{[l]}} \sum_{j=1}^{n_w^{[l]}} a^{[l]}_{(i,j,k)} . a^{[l]}_{(i,j,k')}$$

Where, $k = 1, 2, ..., n_c^{[l]}$ and $k' = 1, 2, ..., n_c^{[l]}$

If the channels are correlated, the activations will be large together and hence $G_{kk'}$ will be large.

The style loss for a single hidden layer l is given by,

$$J^{[l]}_{style}(S, G) = \frac{1}{(2 * n_h^{[l]} * n_w^{[l]} * n_c^{[l]})^2} ||G^{[l](S)} - G^{[l](G)}||^2_F$$

Where, F indicates the Frobenius norm. This can be further simplified to,

$$J^{[l]}_{style}(S, G) = \frac{1}{(2 * n_h^{[l]} * n_w^{[l]} * n_c^{[l]})^2} \sum_k \sum_{k'} (G^{[l](S)}_{kk'} - G^{[l](G)}_{kk'})^2$$

Now, the overall style loss over multiple layers is given as,

$$J_{style}(S, G) = \sum_l \lambda^{[l]} * J^{[l]}_{style}(S, G)$$

Where, λ is a hyper-parameter that lets you take multiple layers in the neural network.

B. Semantic Segmentation

Semantic segmentation [6] is a kind of Image segmentation where each pixel in the image is tagged with a category label. Semantic segmentation does not differentiate between instances and cares only about the pixels. Semantic segmentation can be defined as an image classification at the pixel level. This paper talks about the use of the DeepLabv3+ semantic segmentation algorithm for generating binary masks. Before diving into DeepLabv3+, lets take a look at the previous implementations of DeepLab and their disadvantages.

1) DeepLabv1: DeepLabv1 [7] performs semantic segmentation using deep convolutional neural networks with Atrous or dilated convolutions.

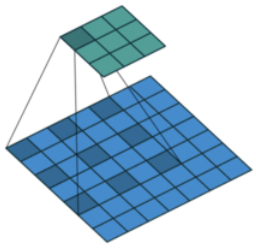

Fig. 3. Working of Atrous Convolution

Atrous [8] convolution requires an additional parameter called the dilation rate which aids in delivering a wider field of view at the same computational cost. In other words, it offers an efficient mechanism to control the field of view of filters to allow larger context assimilation (large field-of-view) or accurate localization (small field-of-view). The resulting feature

map passes through bi-linear interpolation and is the processed using a conditional random field. Conditional Random Fields [9] are used to model connections between different images but here the author uses them to model connections between image pixels.

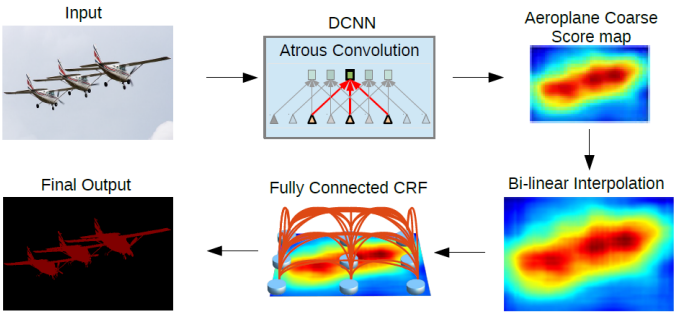

Fig. 4. DeepLabv1 from the Original Paper by L. C. Chen et al.

To overcome the limitations of a short-range CRF, a fully connected CRF model [10] is employed. This model makes use of the energy function,

$$E(x) = \sum_i \theta_i(x_i) + \sum_{i,j} \theta_{i,j}(x_i, x_j)$$

Now, $\theta_i(x_i) = -logP(x_i)$

Where, $P(x_i)$ is the label assignment probability at pixel i as computed by the DCNN.

And,

$$\theta_{i,j}(x_i, x_j) = 1_{x_i \neq x_j} [w_1 \exp\left(-\frac{||P_i - P_j||^2}{2*\sigma_\alpha^2} - \frac{||I_i - I_j||^2}{2*\sigma_\beta^2}\right) + w_2 \exp\left(-\frac{||P_i - P_j||^2}{2*\sigma_\gamma^2}\right)]$$

Bilateral kernel — Smoothness Kernel

Where, $1_{(x_i \neq x_j)}$ serves as a uniform penalty for nearby pixels with different labels. P_i is the position of pixel i and I_i is the intensity (color) vector of pixel i. w_1 and w_2 are the learned parameters and α, β, γ are hyper-parameters that control the scale of the Gaussian Kernels. The Bilateral kernel deals with the pixel positions and their intensity whereas the Smoothness kernel only deals with pixel positions.

However, the fully connected conditional random field is a post-processing process which makes DeepLabv1 not an end-to-end Deep Learning framework.

2) DeepLabv2: DeepLabv2 [11] is very similar to the DeepLabv1 model with the exception of the Deep Convolutional Neural Network. The DCNN in this model employs Atrous Spatial Pyramid Pooling as opposed to the previous Atrous convolutions. ASPP is actually an Atrous version of Spatial Pyramid Pooling, the concept used in SPPNet [12]. In ASPP, parallel Atrous convolution with different dilation rates is applied to the input feature map and fused together. As objects of the same class can have different scales in the image, ASPP helps to account for different object scales which can improve the accuracy.

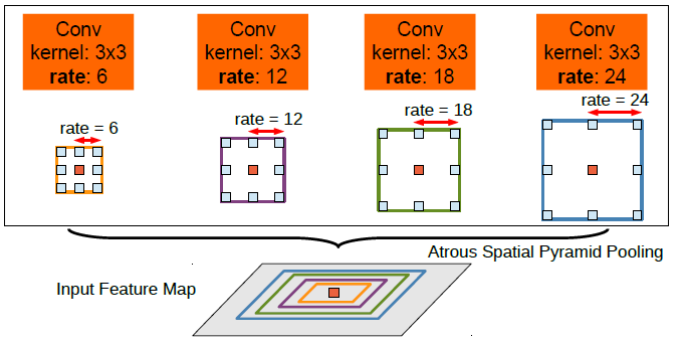

Fig. 5. Atrous Spatial Pyramid Pooling from the DeepLabv2 Paper

However, even in this model a fully connected CRF model is required as a post-processing process and hence is not an end-to-end deep learning framework.

3) DeepLabv3: DeepLabv3 [13] is an updated version of DeepLab that does not use conditional random fields and is an end-to-end deep learning framework. Motivated by multi-grid methods [14] which employ a hierarchy of grids of different sizes to adopt different dilation rates within block 4 to block 7 in the proposed model (ResNet). Now, we perform ASPP with different Atrous rates that effectively captures multi-scale information. However, as the sampling rate becomes larger, the number of valid filter weights becomes smaller. To overcome this problem and incorporate global context information to the model, the author adopts image level features. Specifically, global average pooling [15] is applied on the last feature map of the model, the resulting image-level features is fed to a 1x1 convolution with 256 filters and then the features are bi-linearly upsampled to the desired spatial dimension. Therefore, the improved ASPP consists of : (a) one 1x1 convolution and three 3x3 convolutions with rates = (6, 12, 18) when output stride = 16 (All with 256 filters and batch normalization) and (b) Image-level features.

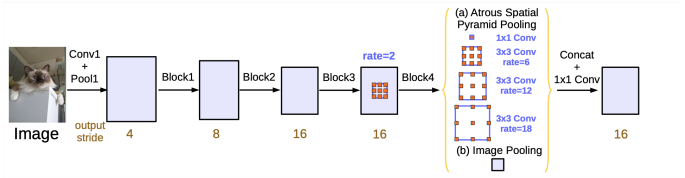

Fig. 6. DeepLabv3 architecture from the Original Paper displaying modified ASPP

The resulting features from all the branches and then concatenated and passed through another 1x1 convolution before the final 1x1 convolution which generates the final logits. In DeepLabv2, the target ground truths are downsampled by 8 during training. In DeepLabv3, it is important to keep the ground truths intact and instead upsample the final logits. DeepLabv3 employs Atrous convolution with upsampled filters to extract dense feature maps and capture long range context.

4) DeepLabv3+: The resulting segmentation map in DeepLabv3 is slightly distorted due to simple bi-linear upsampling. DeepLabv3+ achieves a much more detailed segmentation map by employing an encoder-decoder network architecture. With DeepLabv3+, the DeepLabv3 model is extended by adding a simple yet effective decoder module to refine the segmentation results along the object boundaries. The depth wise separable convolution is applied to both atrous spatial pyramid pooling and decoder modules, resulting in a faster and stronger encoder-decoder network architecture for segmentation.

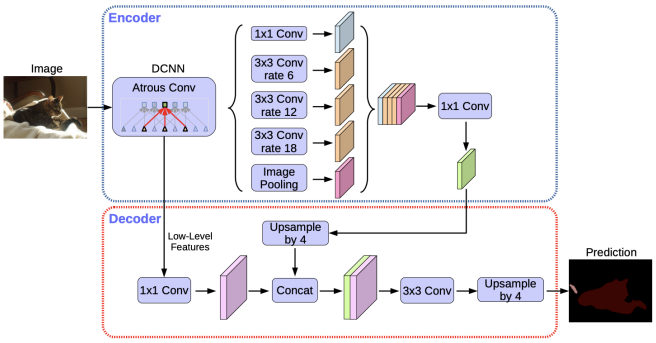

Fig. 7. DeepLabv3+ architecture from the Original Paper

The DeepLabv3 architecture acts as the encoder in this network without the bi-linear upsampling. The encoder features are first bi-linearly upsampled by a factor of 4 and then concatenated with the corresponding low-level features. The low-level features are first passed to a 1x1 convolution to reduce the number of channels. After the concatenation, a few 3x3 convolutions are applied to refine the features followed by another simple bi-linear upsampling by a factor of 4.

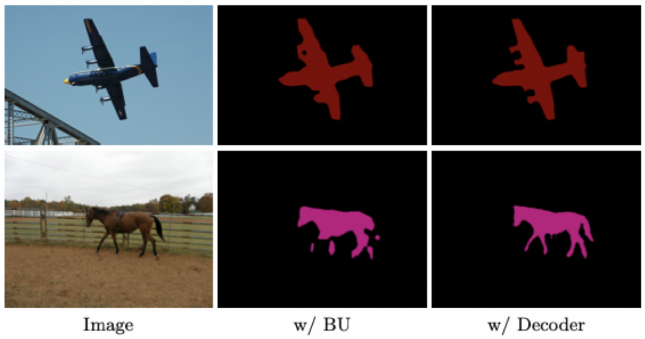

Fig. 8. Comparison between Bi-Linear Upsampling and the Decoder Network

III. Methodology

A. Architecture

The architecture for Nave Background Style Transfer involves the two networks discussed above to generate the resulting stylized image. The input to the architecture is the content image and the style image. The DeepLabv3+ model generates an intermediate binary mask from the content image which is passed to the modified style transfer network.

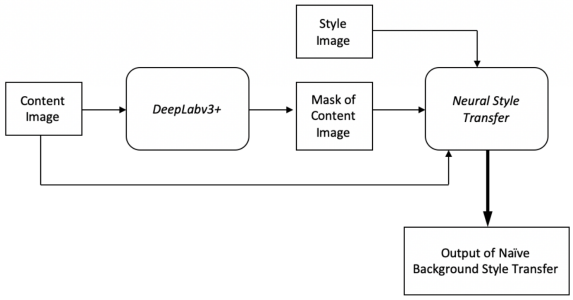

Fig. 9. Naive Background Style Transfer Proposed Architecture

B. Binary Mask Generation

The segmentation map generated by the DeepLabv3+ model is processed internally so as to create a binary mask. The processing works by focusing only on the person class of the COCO dataset. The generated mask is displayed below along with the corresponding content image.

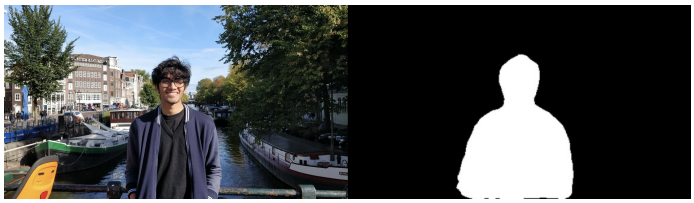

Fig. 10. Generated Binary Mask

C. Mask Adaptive Style Transfer (Nave Approach)

This approach is a nave approach and hence uses the intermediate binary mask to guide the pixels of the content image to the generated image to create the resulting stylized image. This involves first applying style transfer to the content image to create the generated image and then using the binary mask to see which pixels have to be transferred on to the generated image.

IV. Results

A. Background Style Transfer

The results of background style transfer showcase the clear distinction between the foreground and the background of the image. The background incorporates the style of the style image accurately and does not interfere with the foreground of the image. The results are shown in tabular form (as seen in Fig. 11) along with the intermediate binary mask generated from the content image.

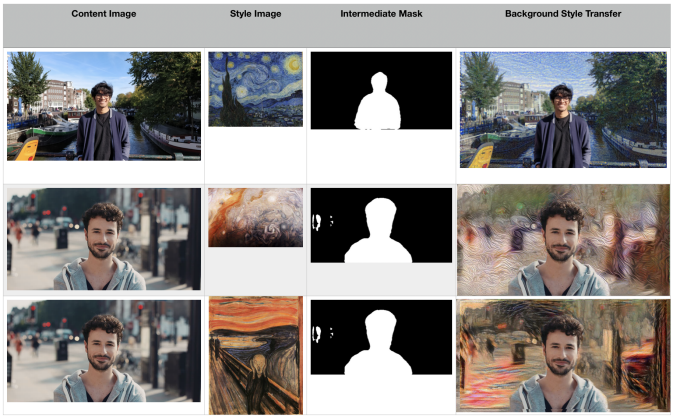

Fig. 11. Results of Naive Background Style Transfer

B. Ablation Study of Hyper-parameters

There are 6 hyper-parameters in this model, however we will only be discussing the effects of 2 of these hyper-parameters i.e. the content weight and the style weight. A variation in these values will affect the styling of the background in the image. A higher content weight as seen in Fig 12. (a)

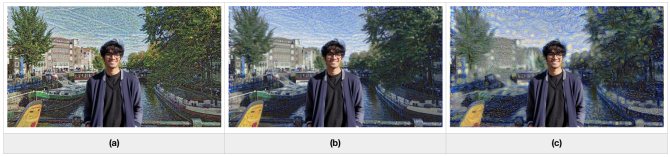

Fig. 12. (a) Content weight set to $1x10^{10}$ (b)Default Parameters (c) Style weight set to $1x10^{6}$

will be robust to the textures of the style image and looks very different from (b) which uses the default parameters. (c) showcases the effects of having extremely high style weights which leads to a heavily stylized background.

V. Future Work

Since this is a naive approach and not an end-to-end network, it could be improved by implementing a new loss function that takes into consideration the mask image and the gradients could be directly propagated to the specified parts of the images. The algorithm can also be made into a mobile application or web application to achieve on-the-go background style transfer. The segmentation map could be processed further to only detect the central entity in the image if there are similar objects in the image.

VI. Conclusion

In this paper, we have seen an application that results from blending two major computer vision algorithms. Even though it is a nave approach, the results are sharp and the transition between the foreground and background is seamless. The user can also control the degree of background style transfer by varying the content weight and style weights which are provided as hyper-parameters. Further development in this algorithm can lead to the creation of new image filters for social media applications like Instagram, Snapchat, and Facebook.

References

[1] L. A. Gatys, A. S. Ecker, and M. Bethge *A Neural Algorithm of Artistic Style*, arXiv preprint arXiv:1508.06576, 2015.

[2] Prisma Labs *Prisma Art Photo Editor with Free Picture Effects and Cool Image Filters for Instagram Pics and Selfies*, Itunes.apple.com. Retrieved 19 July 2016.

[3] H. Dong, G. Yang, F. Liu, Y. Mo, Y. Guo *Automatic brain tumour detection and segmentation using U-Net based fully convolutional neural networks*, arXiv preprint arXiv: 1705.03820, 2017.

[4] M. Treml, J. Arjona-Medina, T. Unterthiner, R. Durgesh, F. Friedmann, P. Schuberth, A. Mayr, M. Heusel, M. Hofmarcher, M. Widrich, B. Nessler, S. Hochreiter *Speeding up semantic segmentation for autonomous driving*, NIPS Workshop (2016).

[5] L. C. Chen, Y. Zhu, G. Papandreou, F. Schroff,and H. Adam *Encoder-decoder with atrous separable convolution for semantic image segmentation*, In ECCV, 2018.

[6] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. *Learning hierarchical features for scene labeling*, Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2013.

[7] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. *Semantic image segmentation with deep convolutional nets and fully connected crfs*, In ICLR, 2015.

[8] S. Mallat *A Wavelet Tour of Signal Processing*, Acad. Press, 2 edition, 1999.

[9] C. Rother, V. Kolmogorov, and A. Blake *Grabcut: Interactive foreground extraction using iterated graph cuts*, In SIGGRAPH, 2004.

[10] P. Krahenbuhl and V. Koltun *Efficient inference in fully connected crfs with gaussian edge potentials*, In NIPS, 2011.

[11] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille *Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs*, TPAMI (2017)

[12] K. He, X. Zhang, S. Ren, and J. Sun. *Spatial pyramid pooling in deep convolutional networks for visual recognition*, arXiv:1406.4729v2, 2014.

[13] L. C. Chen, G. Papandreou, F. Schroff, and H. Adam *Rethinking atrous convolution for semantic image segmentation*, arXiv:1706.05587 (2017)

[14] W. L. Briggs, V. E. Henson, and S. F. McCormick.*A multigrid tutorial*, SIAM, 2000.

[15] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. *Pyramid scene parsing network.*, arXiv:1612.01105, 2016.