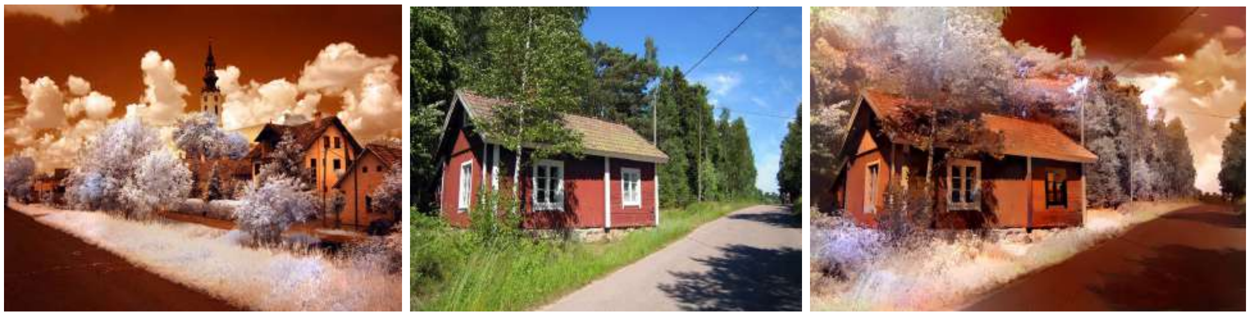
Photorealistic Style Transfer via Wavelet Transforms



04301032019 - Aditi Tiwari
05701032019 - Rhythm

Project: Photorealistic Style Transfer
https://arxiv.org/pdf/1903.09760.pdf
https://arxiv.org/abs/1703.07511
https://arxiv.org/abs/1703.06868
https://arxiv.org/abs/1705.08086

Github repo:
https://github.com/Adititiwari02/Photorealistic-Style-Transfer-AI-College
Input dataset:
https://github.com/luanfujun/deep-photo-styletransfer/tree/master/examples/input
Style dataset:
https://github.com/luanfujun/deep-photo-styletransfer/tree/master/examples/style
Link to colab notebook for dataset analysis:
https://colab.research.google.com/drive/1UOX6sF3-xh9ql4VasTSSup-s1Yn3wHYR?usp=sharing

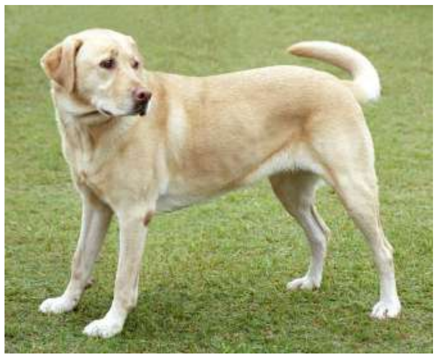
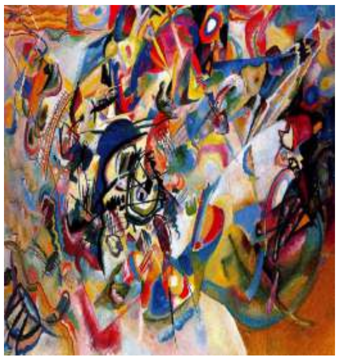
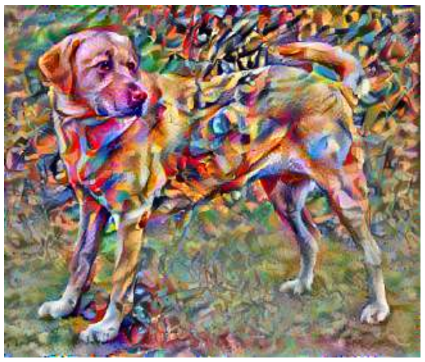
Introduction:

What is photorealism?
Photorealism was an American art movement in which artists attempted to recreate the image in a photo using a different artistic medium such as drawing, pastels, painting, charcoal, etc. The primary goal of a photorealist was to capture the essence of the photo on canvas. We are trying to implement the same via a model which applies a particular reference style on the original image while preserving its details.

Style Transfer: Neural style transfer is an optimization technique used to take two images—a content image and a style reference image. It blends them together so the output image looks like the content image, but “painted” in the style of the style reference image.
This is implemented by optimizing the output image to match the content statistics of the content image and the style statistics of the style reference image. These statistics are extracted from the images using a convolutional network.

Example:

Content Image (C) **Style Image (S)** **Output Image**

Objective:

1. Given a pair of images - S : Style image and C : Content image, perform photorealistic style transfer
2. Leverage wavelet transform to overcome limitations of spatial distortions and introduction of unrealistic artifacts in the final image.
3. Perform end-to-end photorealistic style transfer model that allows to remove the additional post-processing steps.
4. The model provides stable video stylization without temporal constraints.

Work Done:

1. Code written for:
    a. Haar wavelet transform
    b. Encoder-decoder architecture
    c. WCT
2. Integration of the individual components of the architecture.
3. Writing a wrapper function to form end to end style transfer pipeline
4. Performing experiments using several small datasets of content and style images.

What is a wavelet transform?

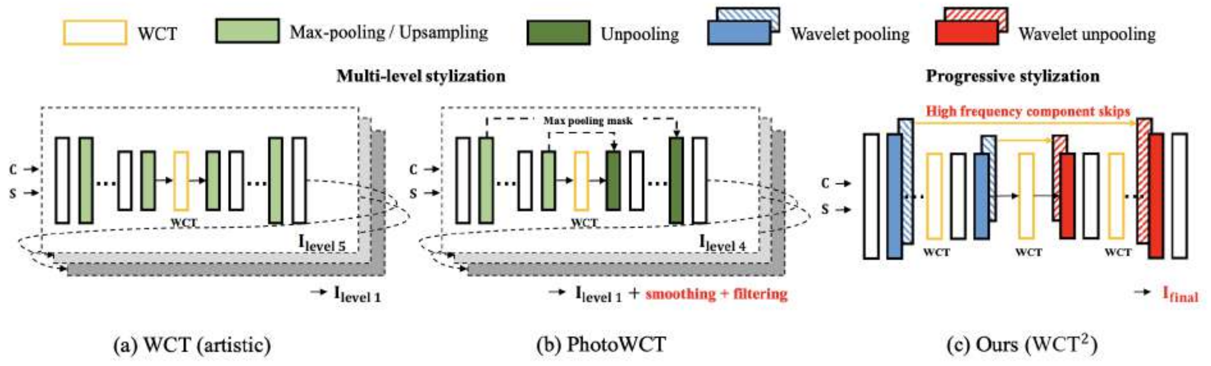
A wavelet is a mathematical function used to divide a given function or continuous-time signal into different scale components. Usually one can assign a frequency range to each scale component. Each scale component can then be studied with a resolution that matches its scale. A wavelet transform is the representation of a function by wavelets.

- We propose a wavelet corrected transfer based on whitening and coloring transforms (WCT^2) that allows features to preserve their structural information and statistical properties of VGG feature space during stylization.
- This is the first and the only end-to-end model that can stylize a 1024*1024 resolution image in 4.7 seconds, which is 830 times faster than the state-of-the-art models,giving a pleasing and photorealistic quality without any post processing or spatial distortions.

Related Work:

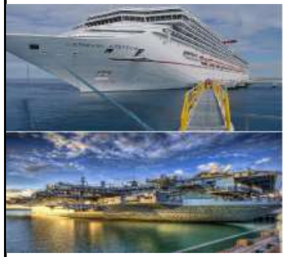
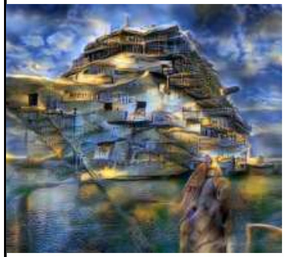
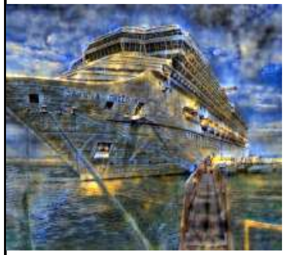
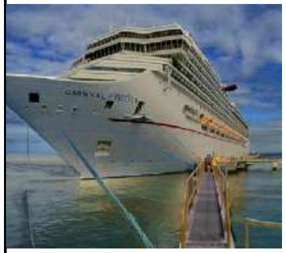
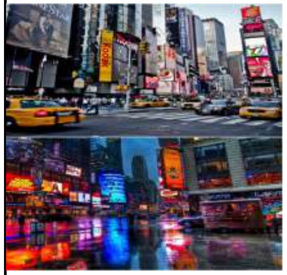
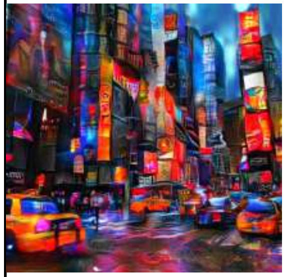
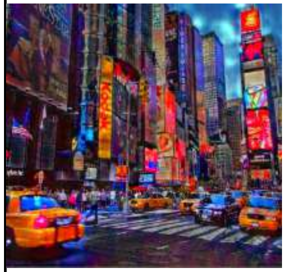
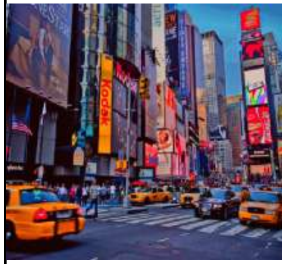
Starting from the seminal work of Gatys et al, many artistic style transfer studies have been proposed to synthesize stylized images through either iterative optimization or manipulating features in pre-trained networks. However, due to their powerful ability to abstract the features, they cannot be used in the photorealistic style transfer as they are.
Compared to artistic style transfer, photorealistic transfer has been overlooked. Classical methods mostly match the color and tone of the images, which are restricted to specific usage.
Luan et al. proposed deep photo style transfer (DPST), which augments the neural style algorithm with an additional photorealism regularization term and a semantic segmentation mask. However, DPST requires heavy computation to solve the regularized optimization problem.
Recently, Li et al. proposed a photorealistic variant of WCT (PhotoWCT), which replaces the upsampling of the VGG decoder with unpooling. PhotoWCT showed that the spatial distortion could be relaxed by providing max-pooling masks to the decoder.
Because the visual quality of the raw outputs of PhotoWCT was not satisfactory, the authors had to employ additional post-processing, such as smoothing and filtering. However, not only do these increase runtime exponentially to the image resolution, but blur final outputs.
Different from the existing methods, our method can preserve the fine structures of an image with little spatial distortion in an end-to-end manner, and thus removes the necessity of additional post-processing steps.

Our approach augments wavelets as a component of the network architecture and provides an interpretable module that can enhance the photorealism of a style transfer model.



WCT2 replaces lossy operations with wavelet pooling and unpooling, and employs the progressive stylization strategy in a single pass. Another advantage of WCT2 over PhotoWCT is that the latter's output needs further post-processing steps.

WCT (Whitening and colouring Transforms)	**PhotoWCT**	**WCT^2 (Our Model)**
Jses Upsampling	Jses max pooling	Jses wavelet pooling
lultilevel Stylization	lultilevel Stylization	Progressive Stylization (single pass method)
akes considerable amount of time	akes considerable amount of time	akes least amount of time
Recursively encoding and lecoding the signal with he lossy VGG networks, rtifacts are amplified luring the multi-level tylization.	PhotoWCT is designed to compensate for information oss during the encoding tep and suppress the patial distortion. These post processing steps equire cumbersome computation and time but hey entail another unfavorable blurry artifact and hyper-parameters to nanually set.	Because of wavelet operations and progressive stylization nvolving a single encoder lecoder system, our model loes not have information loss or suffer from spatial listortions.
Rudimentary outcome is obtained without any steps nvolved to improve image esolution.	akes two more post processing steps(smoothing and filtering) in order to get inal results.	No post processing required and pleasing results obtained

Content and Style Image	WCT	PhotoWCT	WCT^2
			
			

Overview of Method:

➔ Use the VGG-19 model's feature extraction layers as the encoder and its corresponding mirror as the decoder.
➔ Replace the max pooling and unpooling layers with Haar Wavelet Pooling.
➔ Apply WCT after each scale (conv1_X, conv2_X, conv3_X and conv4_X). This is referred to as progressive stylisation.

VGG-19

VGG-19 is a convolutional neural network that is 19-layers deep. There are 16 convolution layers, 3 fully connected layers, 5 MaxPool layers and 1 SoftMax layer. One of the ways in which VGG-19 has been used is content and style superposition using the network.

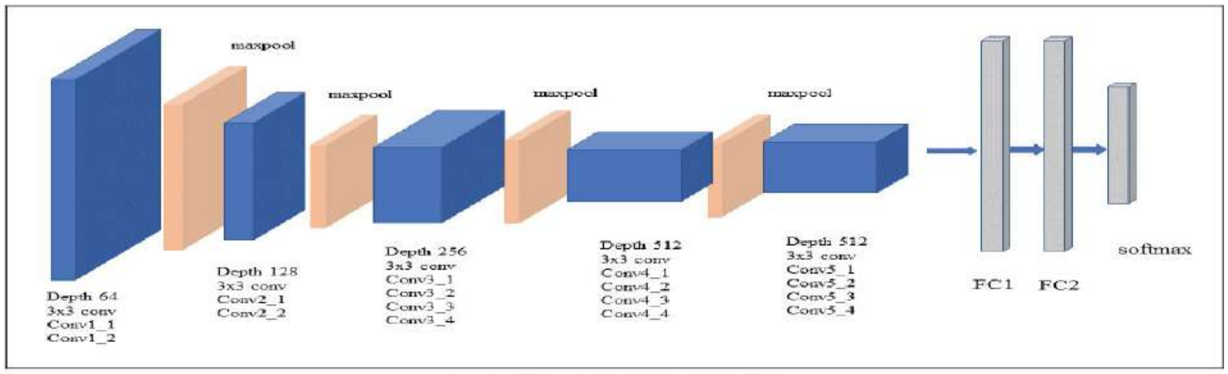


Fig. 3. VGG-19 network architecture

For our use case, we use the layers upto conv4_1 for our encoder. We use the mirror of this model as our decoder.

WCT Procedure:

- Whitening and Colouring Transform(WCT) is basically a feature transformation. It can match the information and data contained in a style image and the content image.
- We first extract the features for both the content (c) and style (s) images. Let us denote this as f_c and f_s .
- We now apply WCT (Whitening and Coloring Transforms).
- We take the output (denoted by f_{cs}) of WCT and pass it to the next layer of the architecture.

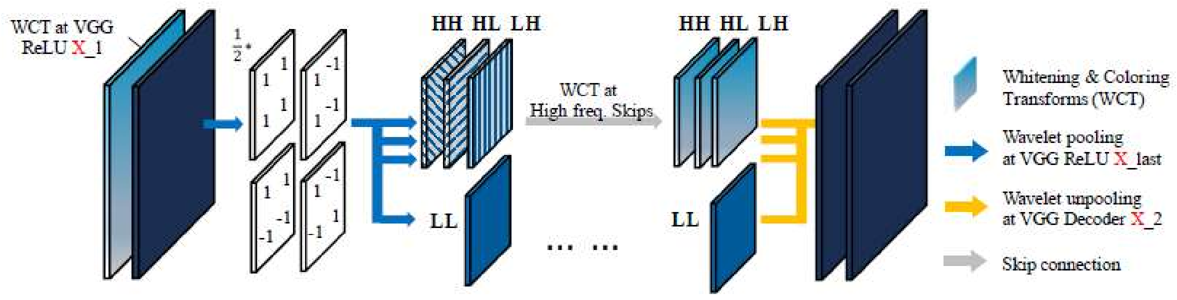
Haar Wavelet Pooling

- Haar wavelet pooling has four kernels, { LL^T LH^T HL^T HH^T }
- The low (L) and high (H) pass filters are:

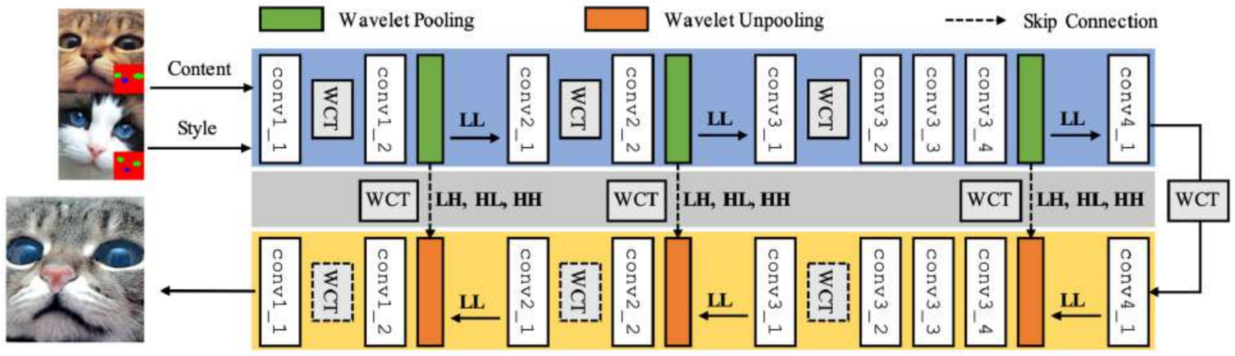
$$L^\top = \frac{1}{\sqrt{2}}\left[\begin{array}{cc} 1 & 1 \end{array}\right], \quad H^\top = \frac{1}{\sqrt{2}}\left[\begin{array}{cc} -1 & 1 \end{array}\right]$$

- Here, the low-pass filter captures smooth surface and texture while the highpass filters extract vertical, horizontal, and diagonal edgelike information
- One important property of the wavelet pooling is that the original signal can be exactly reconstructed by mirroring its operation.
- With this favorable property, our proposed model can stylize an image with minimal information loss and noise amplification.

Proposed Network Architecture


WCT at VGG ReLU X_1
HH HL LH
LL
WCT at High freq. Skips
HH HL LH
LL
Whitening & Coloring Transforms (WCT)
Wavelet pooling at VGG ReLU X_last
Wavelet unpooling at VGG Decoder X_2
Skip connection

Schematic illustration of the wavelet module

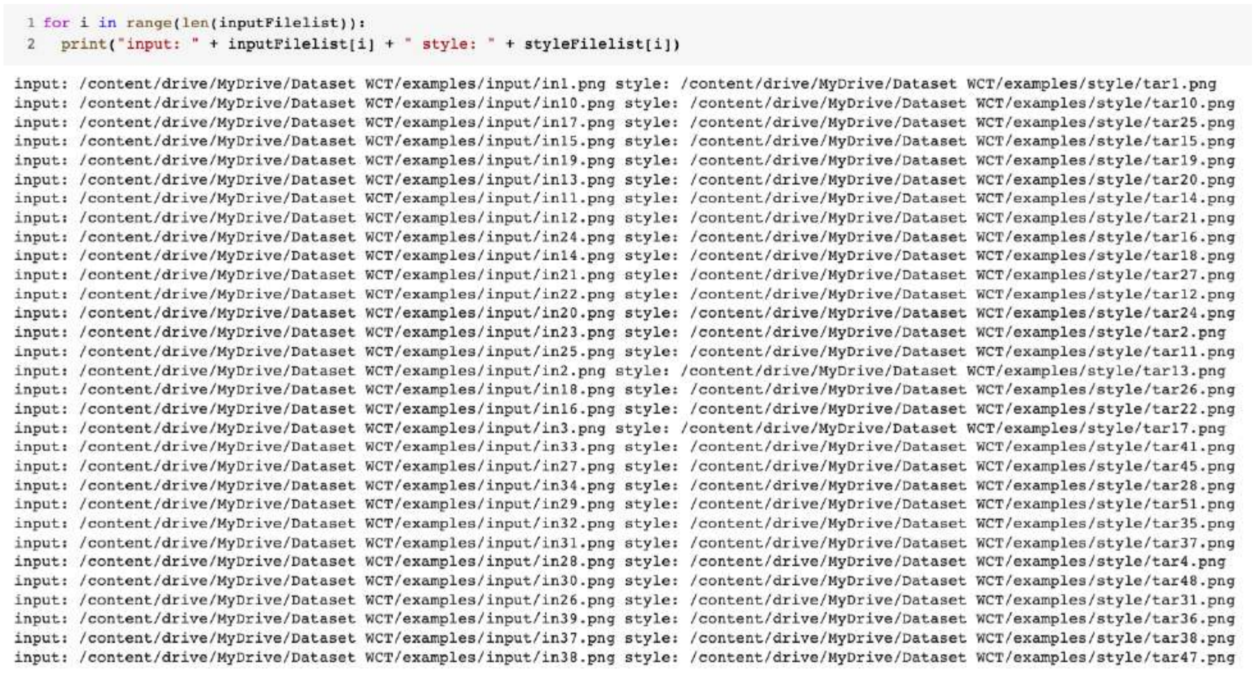


Overview of the proposed progressive stylization. For the encoder, we perform WCT on the output of convX_1 layer and skip connections. For the decoder, we apply WCT on the output of convX_2 layer, which is optional.

Dataset Analysis

Our dataset consists of two subsets:
The first aspect of it is the CONTENT IMAGES. This is the main frame of reference and the result has to be in complete sync with it.
The second aspect is the STYLE IMAGES. This is the secondary frame of reference and the result, even though based primarily on the content image attributes, is modified artistically depending on what the style image looks like.
Therefore, for one result image there need to be two inputs in the form of a content and a style image.

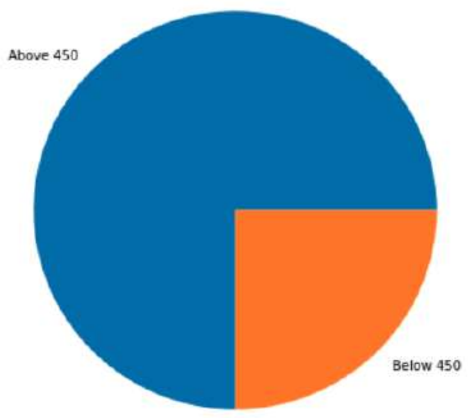
To achieve photorealism, a model should recover the structural information of a given content image while it stylizes the image faithfully at the same time.



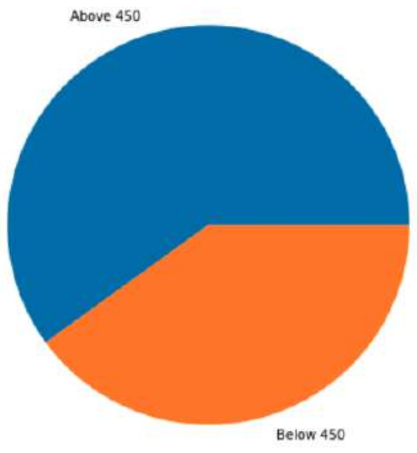
```
for i in range(len(inputFilelist)):
  print("input: " + inputFilelist[i] + " style: " + styleFilelist[i])
```

```
input: /content/drive/MyDrive/Dataset WCT/examples/input/in1.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar1.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in10.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar10.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in17.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar25.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in15.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar15.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in19.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar19.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in13.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar20.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in11.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar14.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in12.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar21.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in24.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar16.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in14.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar18.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in21.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar27.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in22.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar12.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in20.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar24.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in23.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar2.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in25.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar11.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in2.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar13.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in18.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar26.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in16.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar22.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in3.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar17.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in33.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar41.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in27.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar45.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in34.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar28.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in29.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar51.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in32.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar35.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in31.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar37.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in28.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar4.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in30.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar48.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in26.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar31.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in39.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar36.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in37.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar38.png
input: /content/drive/MyDrive/Dataset WCT/examples/input/in38.png style: /content/drive/MyDrive/Dataset WCT/examples/style/tar47.png
```

Pie Chart



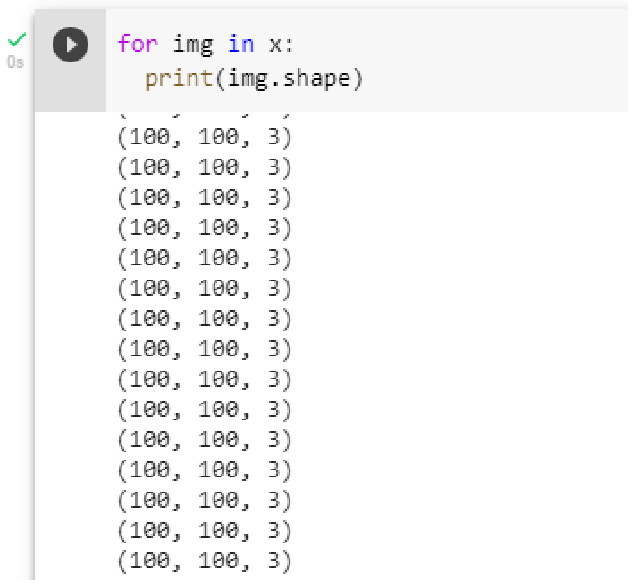
Input images having size greater than 450x700



Style images having size greater than 450x700

Resizing the images in the dataset

After this we resize the images in the dataset to 100x100 using the resize function of opencv to have images of the same size.

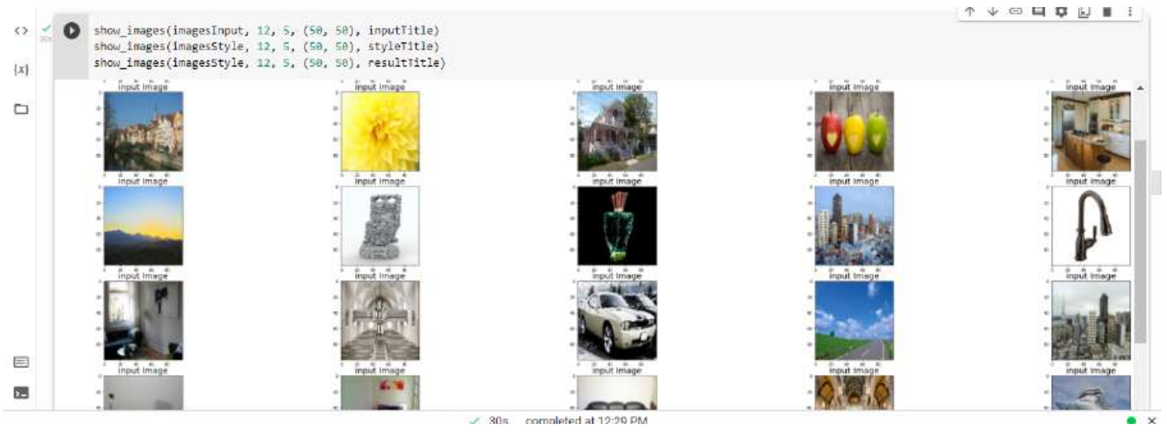


```
for img in x:
  print(img.shape)
```

```
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
(100, 100, 3)
```

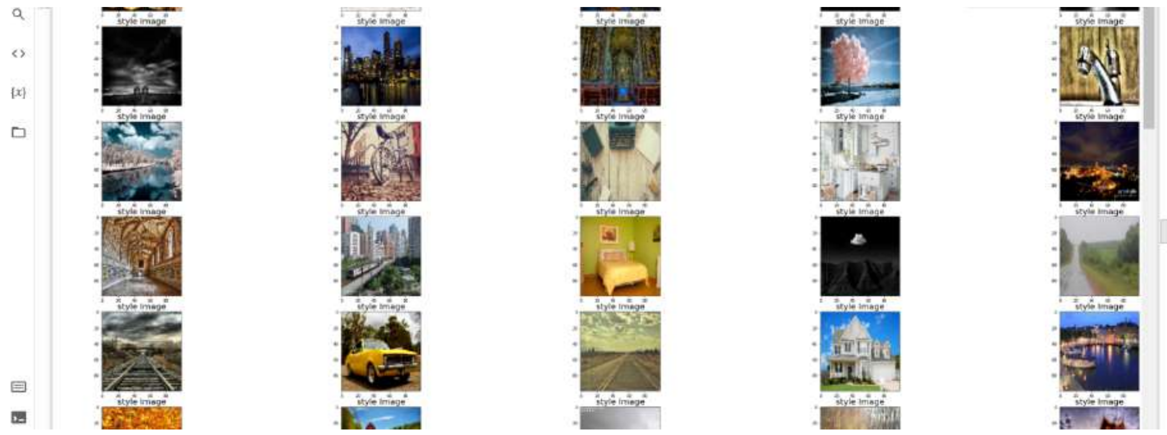
Display the images:

- Images in the input dataset:

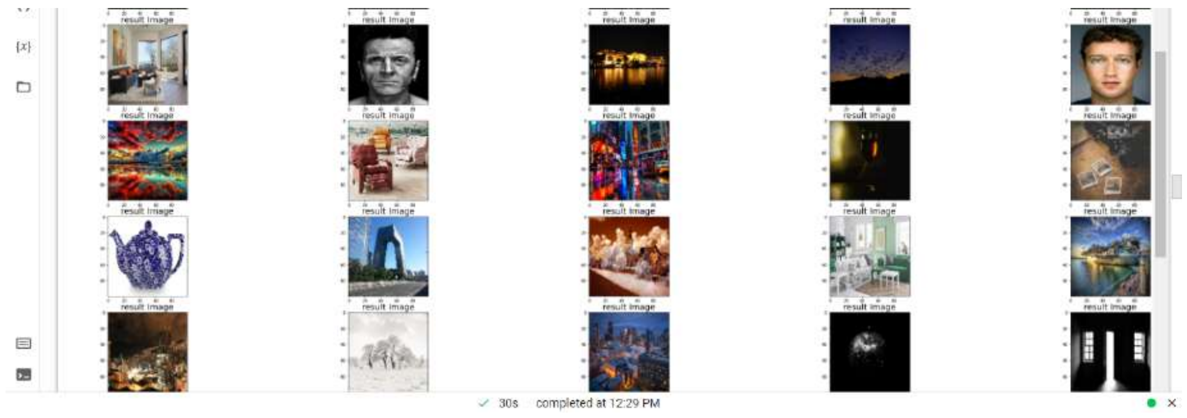


```
show_images(imagesInput, 12, 5, (50, 50), inputTitle)
show_images(imagesStyle, 12, 5, (50, 50), styleTitle)
show_images(imagesStyle, 12, 5, (50, 50), resultTitle)
```

- Images in the style dataset:



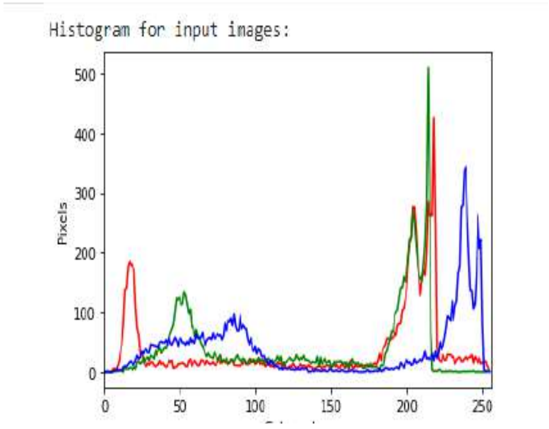
- Images of the result after applying photorealistic style transfer:

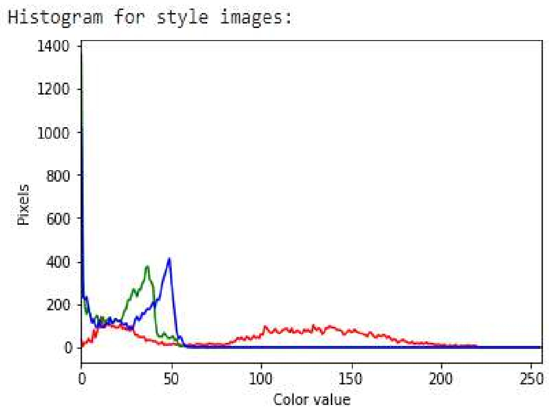


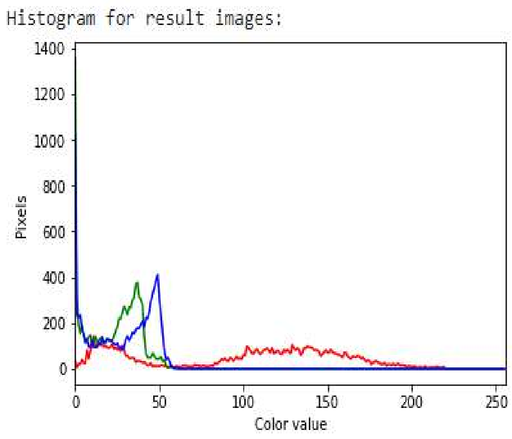
Color Histogram of the images
The image colour histograms show the number of pixels which are plotted against certain colour values. It can be clearly observed on comparing the pictures with their respective histograms that the dominant colour intensity is the one which is possessed by the most number of pixels. Conventionally the colour values range from (0,255) with 0 being dark and 255 being white.

By photorealistic style transfer we aim at preserving the structural details of the input image while applying the textures and gradient of the second subset of the dataset.

As can be seen in the result of the histogram, the result image histogram is very similar to the style image. This is because the texture and color of the style images were taken but the structure of the input image remained intact.







Intermediate and Final results

Intermediate Results

Haar Wavelet Filter

● To test the working of the Haar wavelet convolution filters, we generate the 4 kernels
LL, LH, HL, HH as mentioned in the paper.

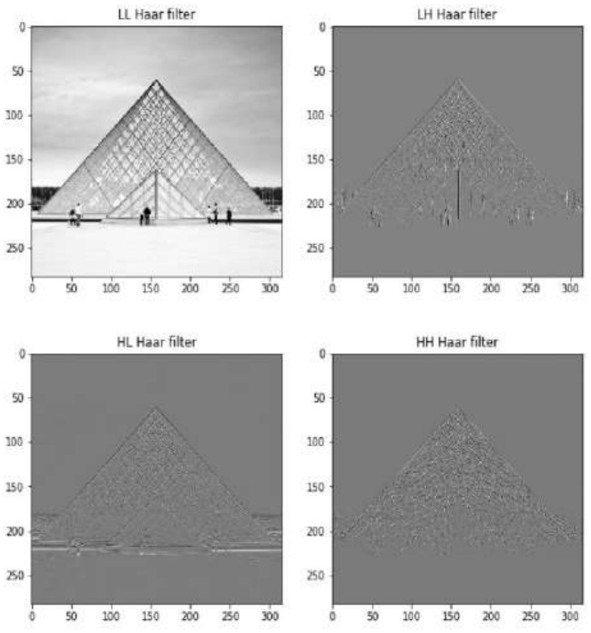
● We apply the wavelet filters to two experiment images and show our observations.

● It is observed that :

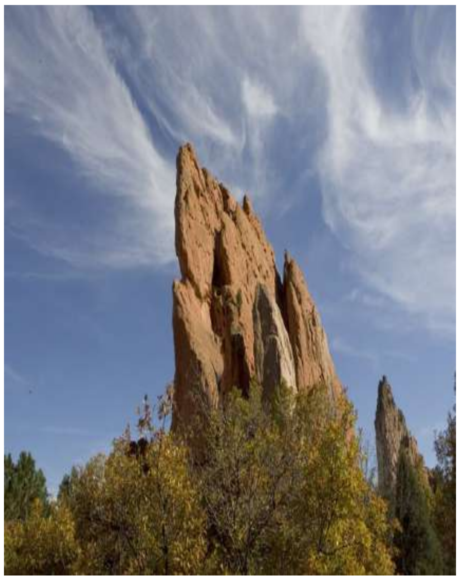
○ The LL filter convolution has little to no change on the image. This is because it
is a low pass filter and captures only the lower frequencies in the image

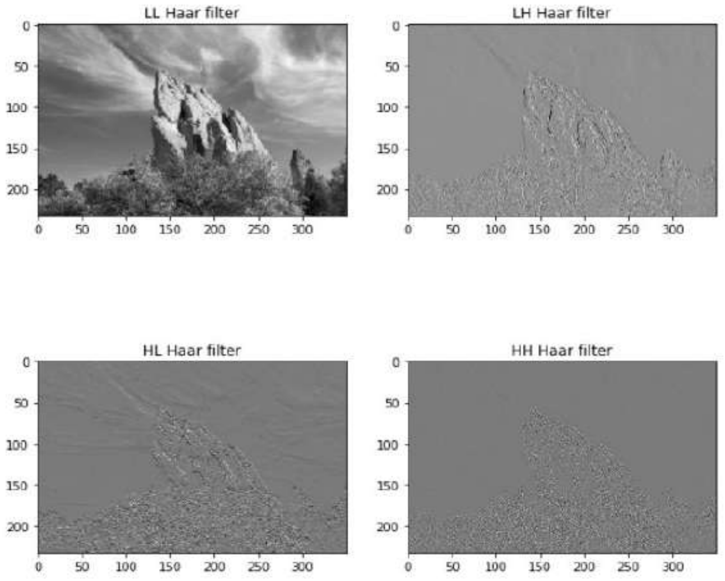
○ The HH filter on the other hand captures the higher frequency, intricate edges of
the image as we can see in the case of the Louvre museum image's glass edges.

Experiment 1:



Experiment 2:





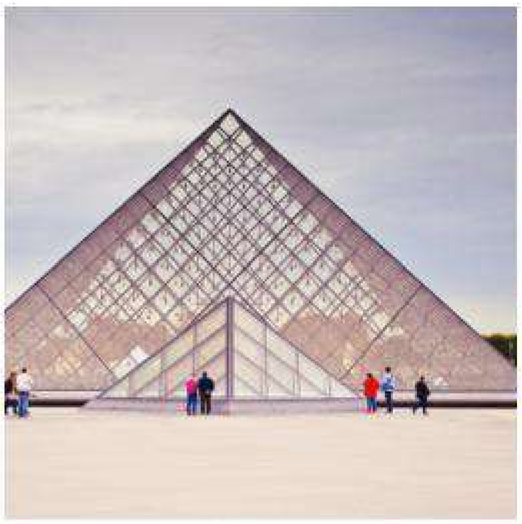
Testing Encoder Decoder with Image Reconstruction

- For the encoder, we have used the feature extraction layers (upto conv4_x) as the encoder and its corresponding mirror architecture as the decoder.
- This architecture had been trained for image reconstruction, i.e, given an image as the input, the encoder should encode the image and the decoder should output the original image.
- In order to test if our implementation of the encoder-decoder is working, we shall see if the above case is working.
- We will also see how the architecture performs if we remove certain layers.

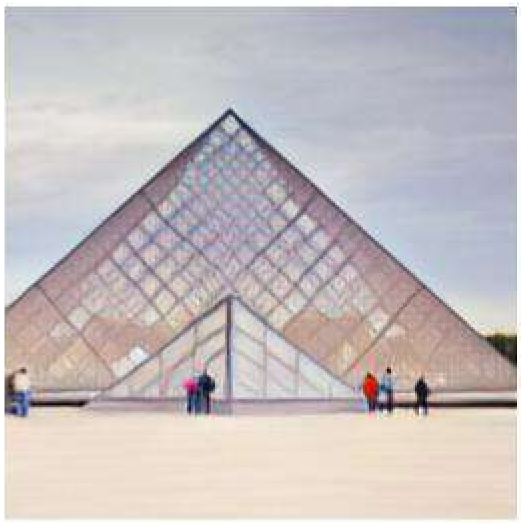
All Layers included (conv1_x, conv2_x, conv3_x, conv4_x)

Experiment Image 1:

Input Image



Output Image

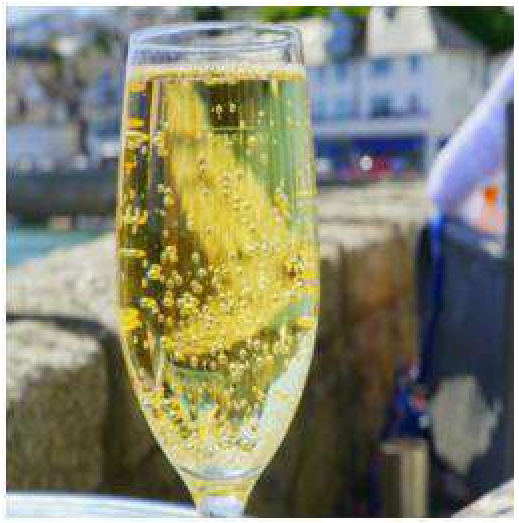


We observe that the reconstructed image is very similar to the input image but is slightly blurrier.

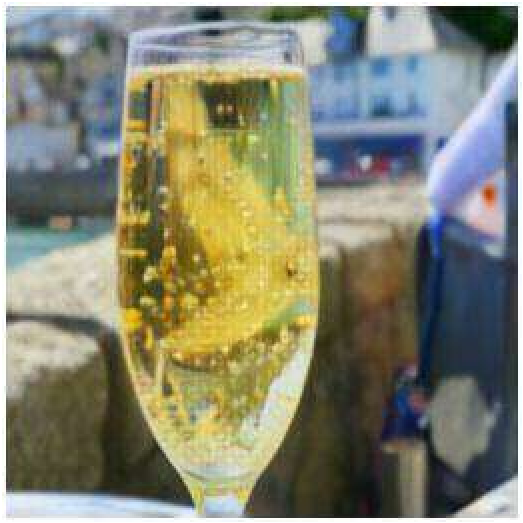
All Layers included (conv1_x, conv2_x, conv3_x, conv4_x)

Experiment Image 1:

Input Image



Output Image



We observe that the reconstructed image is very similar to the input image but is slightly blurrier.

Last layer omitted (conv1_x, conv2_x, conv3_x)

Experiment Image : 1

Input Image

Output Image

We observe that the reconstructed image is mostly gray, however some resemblance of the original image is seen.

Last layer omitted (conv1_x, conv2_x, conv3_x)

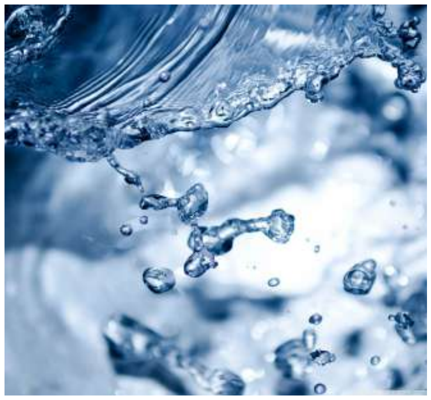
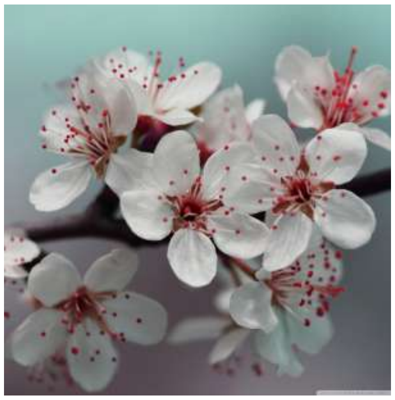
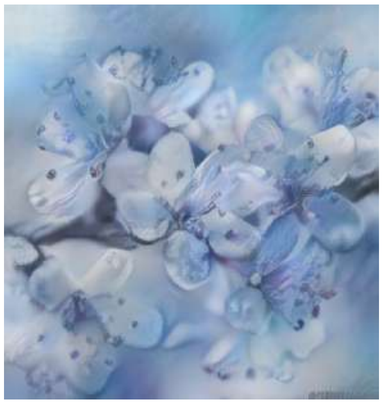
Experiment Image : 2

We observe that the reconstructed image is mostly gray, however some resemblance of the original image is seen.

Single-level stylization:
With both VGG and Decoder fixed, and given the content image C and style image S, our method performs the style transfer through whitening and coloring transforms.

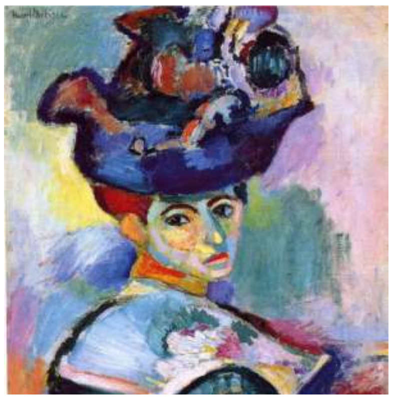
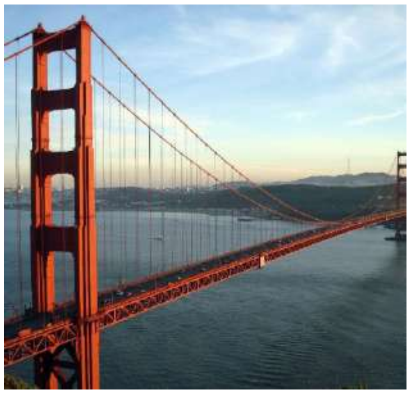
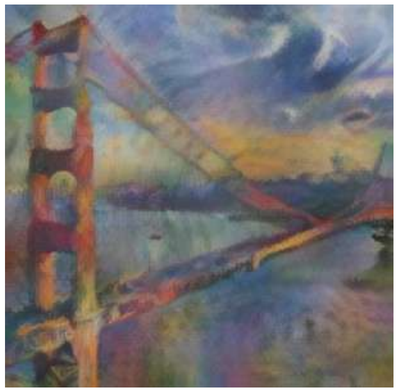
EXPERIMENT 1:

Style Content Single-level Stylization

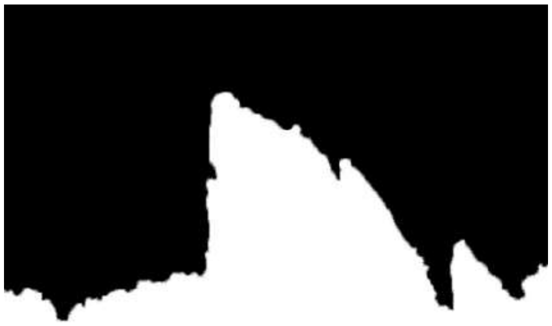




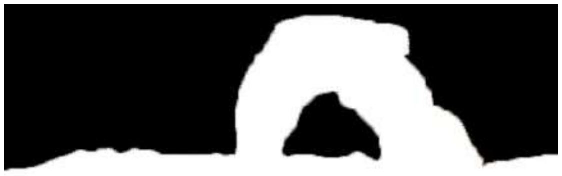
EXPERIMENT 2:

Style Content Single-level Stylization

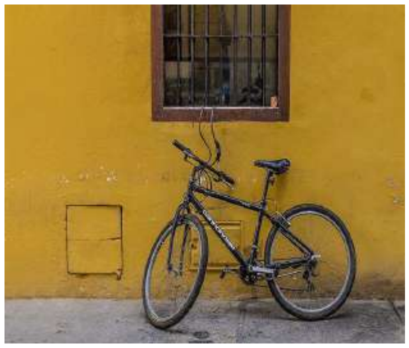
WCT Core and WCT Core Segment
In WCT Core, we apply WCT, take the output and pass it to the next layer of the architecture without using a semantic segmentation mask. In WCT Core Segment, we use semantic segmentation of the inputs to avoid content-mismatch problems (difference in content between input and reference images could result in undesirable transfers between unrelated content), which improves the photorealism of the results.



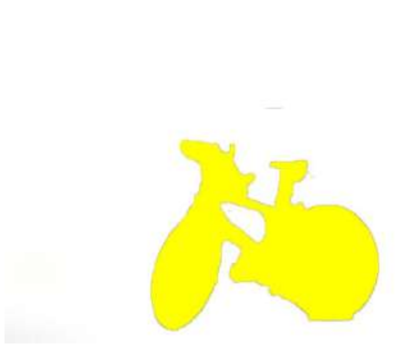


WCT with Semantic Segmentation
In WCT2, to apply semantic segmentation masks to the artistic style transfer methods, we followed the spatial control techniques proposed by the authors. Artistic style transfer methods generate undesired distortions and artifacts and often fail to maintain the structural information despite the spatial control with segmentation maps. In comparison, because of wavelet corrected transfer, WCT2 prevents unrealistic artifacts and preserves structure information such as edges.

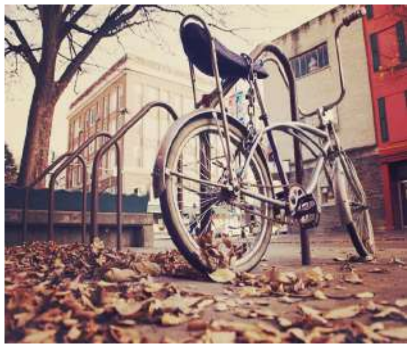
Content Image



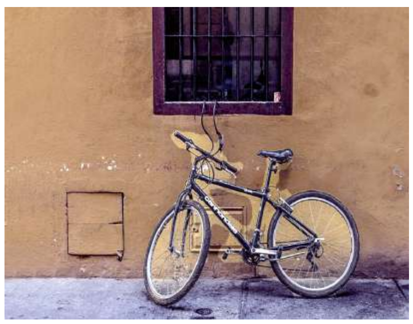
Content Segmentation



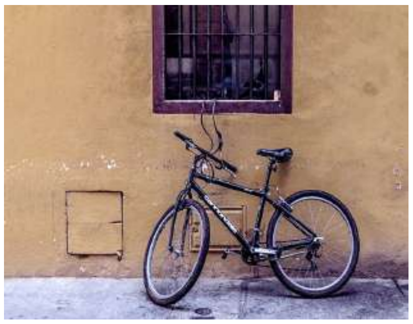
Style Image



Style Image with Segmentation



Style Image without Segmentation

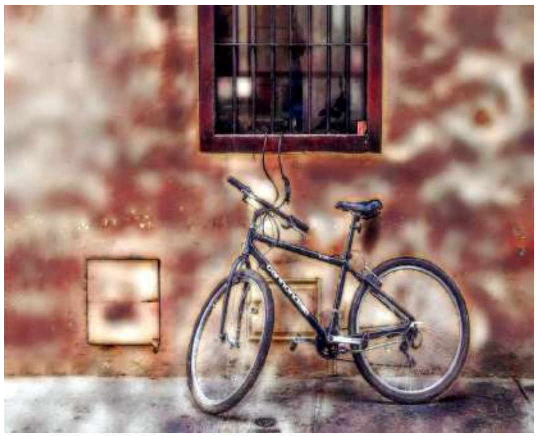


Unpooling Options

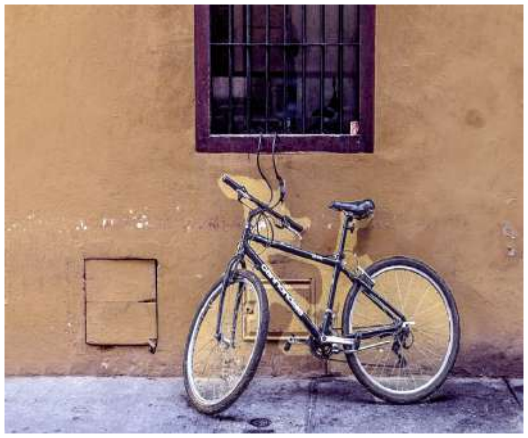
The paper suggests different unpooling options in the model architecture, such as - summation, split pooling, learnable, and concatenation based pooling. Further, the summation and concatenation based methods are analysed in the paper :

- Concatenation (CAT5) pooling outputs look much better than summation based
- CAT5 unpooling performs channel-wise concatenation of the four feature components from the corresponding scale plus feature output before the wavelet pooling
- Summation based is the conventional transposed convolution and summation
- CAT5 enables the network to learn the weighted sum of components

Summation Pooling

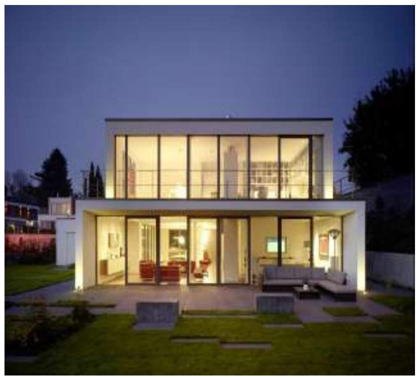
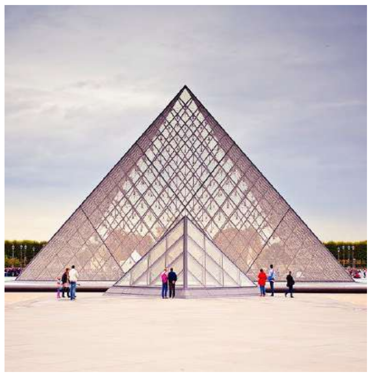
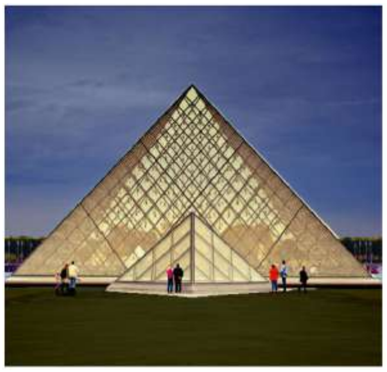


Concatenation Pooling



Final Results

Style Content Results

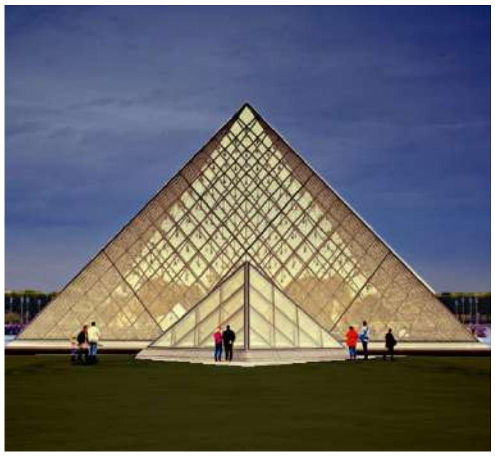
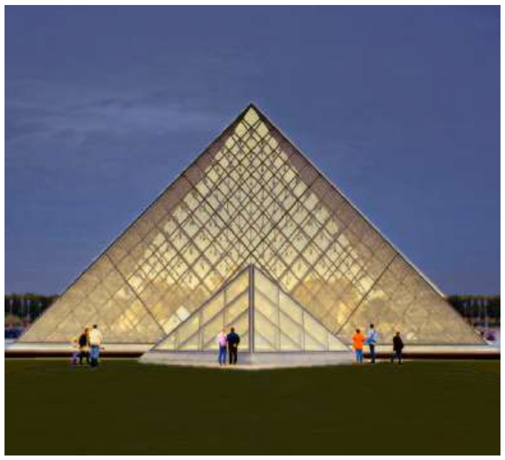




Effect of Skip Connections

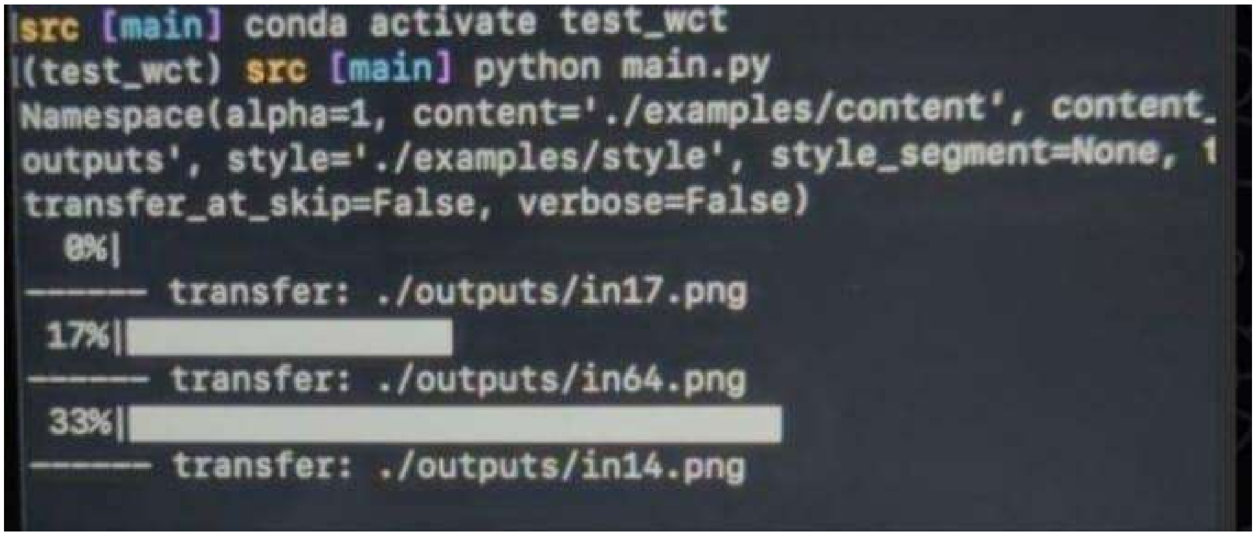
● The low frequency component captures smooth surface and texture while the high frequency components detect edges.

● More specifically, it implies that applying WCT to LL of the encoder affects overall texture or surface while applying WCT to the high frequency components (i.e., LH, HL, HH) stylize edges.

WCT(with skip) WCT(LL only)

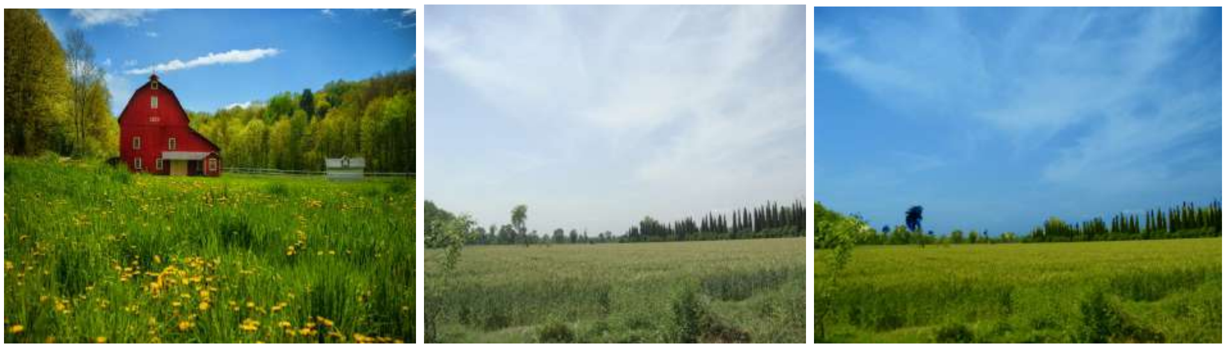



Outputs

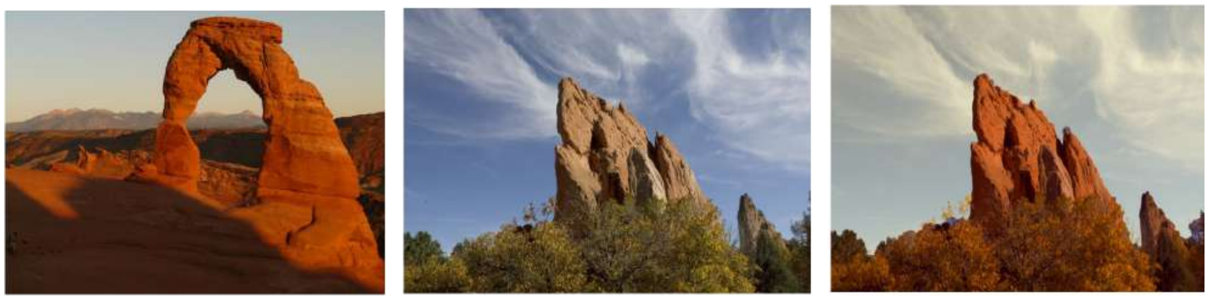


```
src [main] conda activate test_wct
(test_wct) src [main] python main.py
Namespace(alpha=1, content='./examples/content', content_
outputs', style='./examples/style', style_segment=None, t
transfer_at_skip=False, verbose=False)
  0%|
------- transfer: ./outputs/in17.png
 17%|███████████████
------- transfer: ./outputs/in64.png
 33%|██████████████████████████████
------- transfer: ./outputs/in14.png
```

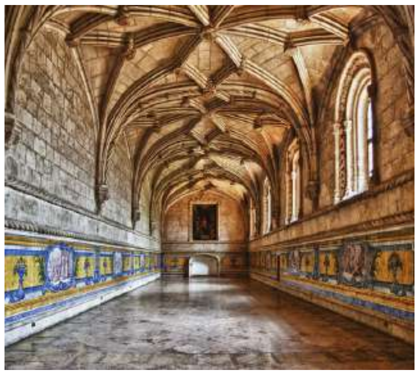
Style Content Final



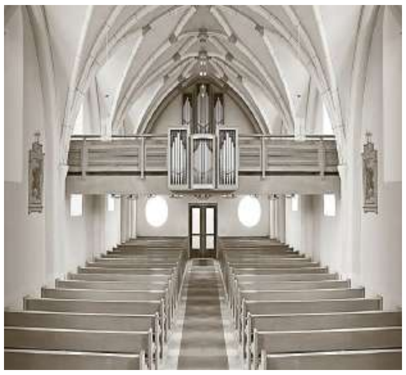
Style Content Final



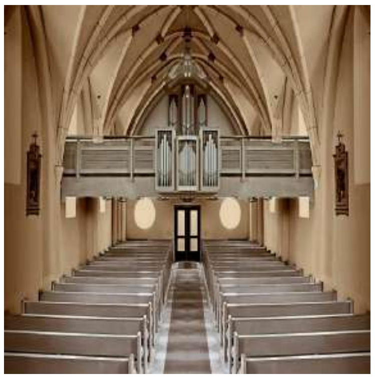
Style



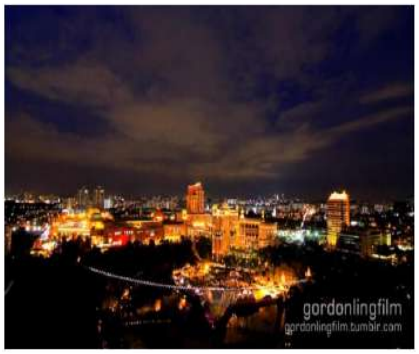
Content



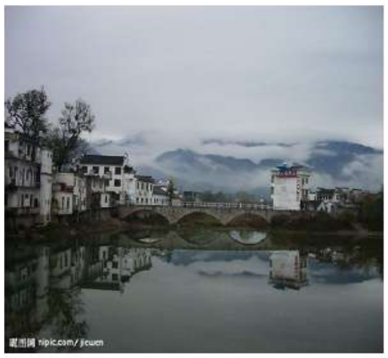
Final



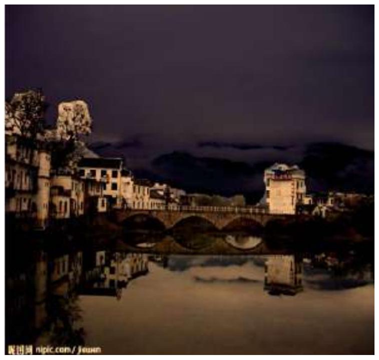
Style



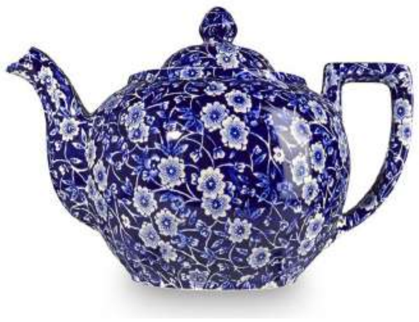
Content



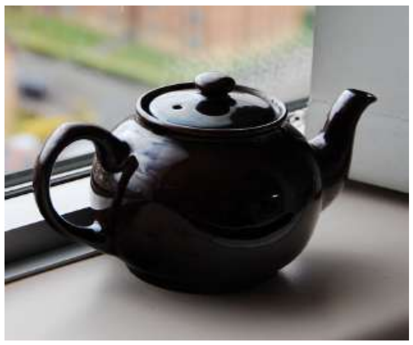
Final



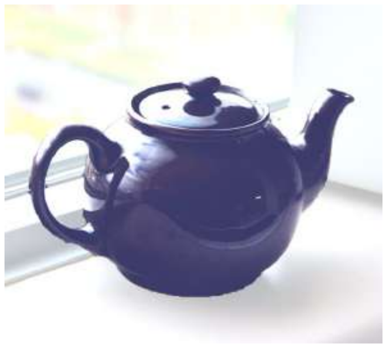
Style



Content



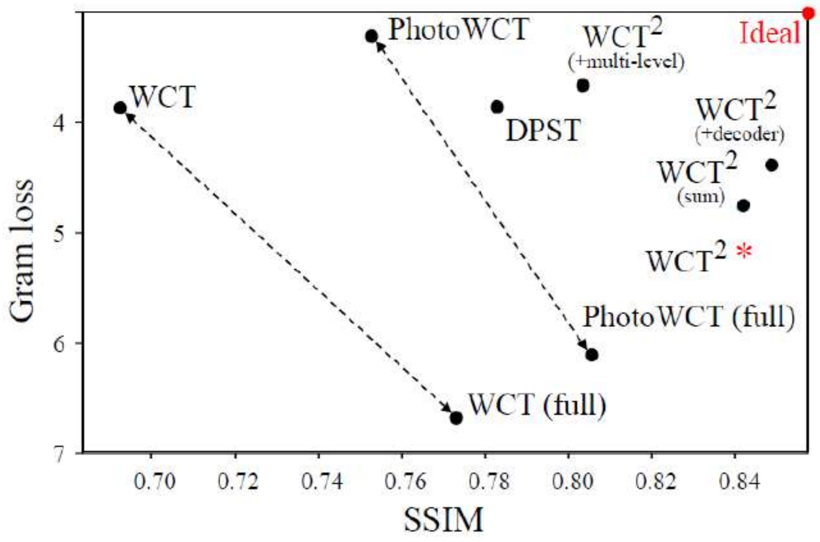
Final



Quantitative evaluation

To measure photorealism, we employ two surrogate metrics for spatial recovery and stylization. We calculate the **structural similarity (SSIM) index** between edge responses of original contents and stylized images. Following WCT, we report the covariance matrix difference (VGG style loss) between the style image and the output of the model.

The figure shows SSIM (X-axis) against style loss (Y-axis). WCT^2 model remarkably outperforms other methods.



Qualitative evaluation

Other models such as DPST often generate “staircasing” or “cartoon” artifacts with an unrealistic color transfer, which severely hurts photorealism. PhotoWCT better reconstructs the details of the content image, while it shows spotty artifacts over entire images. Such artifacts can be removed by employing additional post processing steps. It has three disadvantages: 1) optimization is slow , 2) hyper-parameters should be carefully tuned to trade-off between smoothness and fine details and 3) the final image becomes blurry at the expense of removed artifacts.

Image Size	DPST	PhotoWCT (full) (WCT + post)	Ours
256 × 256	306.9	3.2 + 9.2	**3.2**
512 × 512	1020.7	3.6 + 40.2	**3.8**
896 × 896	2988.6	3.8 + OOM	**4.4**
1024 × 1024	3887.8	3.9 + OOM	**4.7**

Conclusion

Based on the theoretical analysis, we specifically designed our model to satisfy the reconstruction condition. The exact recovery of the wavelet transforms allows our model to preserve structural information while providing stable stylization without any constraints. By employing progressive stylization, we achieved better results with less noise amplification. Compared to the other state-of-the-arts, our analysis and experimental results showed that WCT2 is scalable, lighter, faster and achieves better photorealism quantitatively and qualitatively.