ANALYZING STYLE TRANSFER ALGORITHMS FOR SEGMENTED IMAGES

By

Seyed Hadi Seyed

December, 2024

Director of Thesis: David Hart, PhD

Major Department: Computer Science

ABSTRACT

The recently developed Segment Anything Model has made grabbing semantically meaningful regions of an image easier than before. This will allow for new applications that build on this approach that weren't previously possible. This thesis investigates integrating the Segment Anything Model with style transfer. Specifically, it proposes Partial Convolution as a way to improve style transfer for segmented regions. Additionally, it investigates how different style transfer techniques are affected by different mask sizes, image statistics, etc.

ANALYZING STYLE TRANSFER ALGORITHMS FOR SEGMENTED IMAGES

A Thesis

Presented to The Faculty of the Department of Computer Science

East Carolina University

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Data Science

By

Seyed Hadi Seyed

December, 2024

Director of Thesis: David Hart, PhD

Thesis Committee Members:

Nic Herndon, PhD

Rui Wu, PhD

DEDICATION

This thesis is dedicated to my family, whose love and support have been the foundation of my achievements. To my parents, thank you for inspiring my passion for learning.

ACKNOWLEDGEMENTS

I sincerely thank everyone who supported me during this journey. I am especially grateful to my advisor, Dr. David Hart, for his invaluable guidance, encouragement, and insight and to the respectful committee for their constructive and valuable feedback. To my family and friends, your unwavering belief in me gave me the strength to complete this work. I would like to thank my parents for providing me with the opportunity to gain this tremendous education and the necessary tools to succeed in life.

Table of Contents

LIST OF FIGURES

Chapter 1

Introduction

In 2023, Meta AI released the Segment Anything Model [8], a new approach for automatically detecting and outlining meaningful regions of an image. In this thesis, I will combine this technique with Style Transfer [2], an approach for making an image look like it was painted or styled in a particular way. I will now describe both of these fundamental pieces in more detail.

1.1 What is the Segment Anything Model?

The Segment Anything Model (SAM) is a powerful AI model developed by Meta AI that can automatically identify and segment objects in images with minimal human input. Previous approaches relied on class labels to segment objects [17, 4] whereas SAM was trained without labels. SAM is designed to work with a wide variety of objects and environments, even those it hasn't seen before, making it highly versatile for various applications in computer vision.

The output of SAM is a list of *masks* for the objects in the scene. Each mask indicates the pixels that are a part of the object. Masks can overlap and the set of masks does not necessarily need to cover every pixel in the image. An example output from SAM is illustrated in Figure 1.1.

SAM has key features that will continue to make it more accessible to public and mainstream in the coming years. Those features include:

- **Zero-shot Segmentation**: SAM can segment objects in an image without requiring

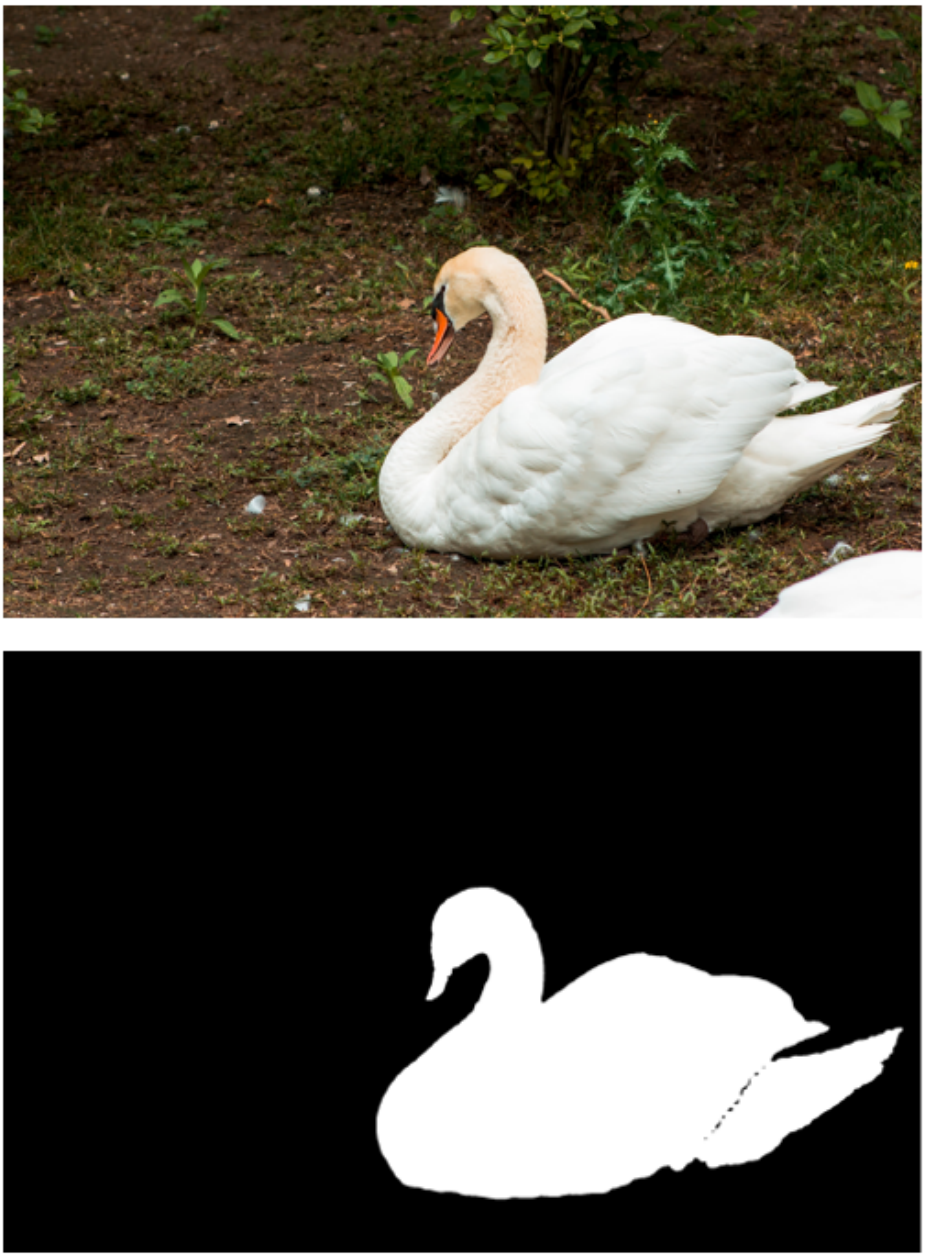

Figure 1.1: Example output using the Segment Anything Model. Given an input image (top) and a point from the user, the model can determine all pixels associated with the object closest surrounding that point, providing a mask of the object (bottom).

task-specific training. It can generalize well across different objects, including unfamiliar ones.

- **Interactive Segmentation**: The model allows users to interact with the segmentation process by providing various inputs like points, bounding boxes, or text prompts.
- **General-purpose**: Unlike traditional segmentation models that are trained to identify specific objects or categories, SAM can segment "anything." It is not restricted to a predefined set of object categories and can work in a wide range of domains.
- **Pre-trained on Massive Data**: SAM is trained on a massive dataset, which gives it a broad understanding of object shapes, boundaries, and contexts. This pre-training allows it to generalize across new images and unseen categories. SAM leverages transformers, a neural network architecture commonly used in natural language processing, for its ability to capture global context in images. It also uses mask generation to highlight different areas of interest within an image, offering flexibility in how objects are segmented.

Clearly, the Segment Anything Model represents a leap in general-purpose segmentation, making it easier to use AI in areas where accurate object identification is beneficial. It will have broad applications in the fields of image editing, medical imaging, robotics, and augmented/virtual reality.

1.2 What is Style Transfer?

Style Transfer is a technique in computer vision and deep learning that involves applying the visual style featuers onto an image. The technique was popularized by the paper "A Neural Algorithm of Artistic Style" by Leon Gatys and colleagues in 2015 [2]. It creates a new image that retains the content from one image (content image) and adopts the artistic or visual style of another image (style image). The process usually uses neural networks, especially Convolutional Neural Networks (CNNs), which are good at recognizing spatial patterns like textures, shapes, and edges.

In particular, this work uses the style transfer approach presented by Li *et al.* [13]. In this approach, a content image is provided which contains the objects, shapes, and structure that want to be preserved. A style image is also provided which contains the artistic style (brushstrokes, color patterns, textures) that want to be applied to the content image. Both of

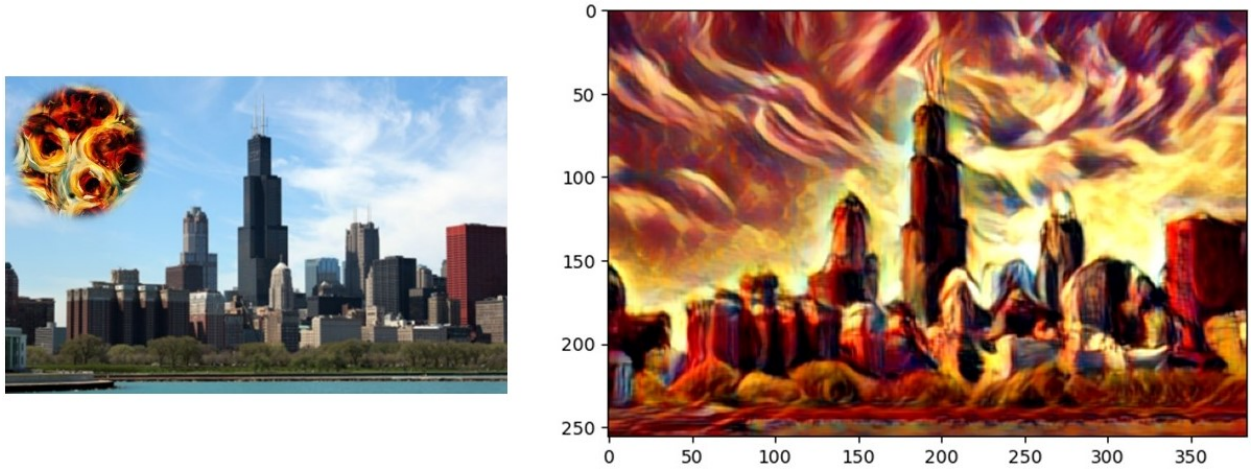

Figure 1.2: Example output using the Linear Style Transfer algorithm for a full image. A content and style image (left) are fed into the style transfer network, resulting in stylized version of the content image (right).

these are fed into a pre-trained CNN and there representative features are blended together to create the output. An example of the style transfer output is shown in Figure 1.2.

Style Transfer continues to be actively researched and have wide use of applications in art creation, image editing, and film/game design. However, almost all approaches have focused on whole-image stylization without addressing how these approaches behave when working on individual regions of an image. This is the focus of my work.

1.3 Combining Style Transfer with the Segment Anything Model

The goal of this work is to combine Style Transfer with the Segment Anything Model. This can result in powerful and creative applications that allow users to selectively apply artistic styles to specific objects or regions within an image.

The naive approach would be to simply apply style transfer to the whole image, and then to mask the image after the stylization. However, I show that this method sometimes leads to *under stylization*, where the features of the style image are not well represented in the stylized content image.

To resolve this issue, I implement Partial Convolution [16] inside the style transfer CNN.

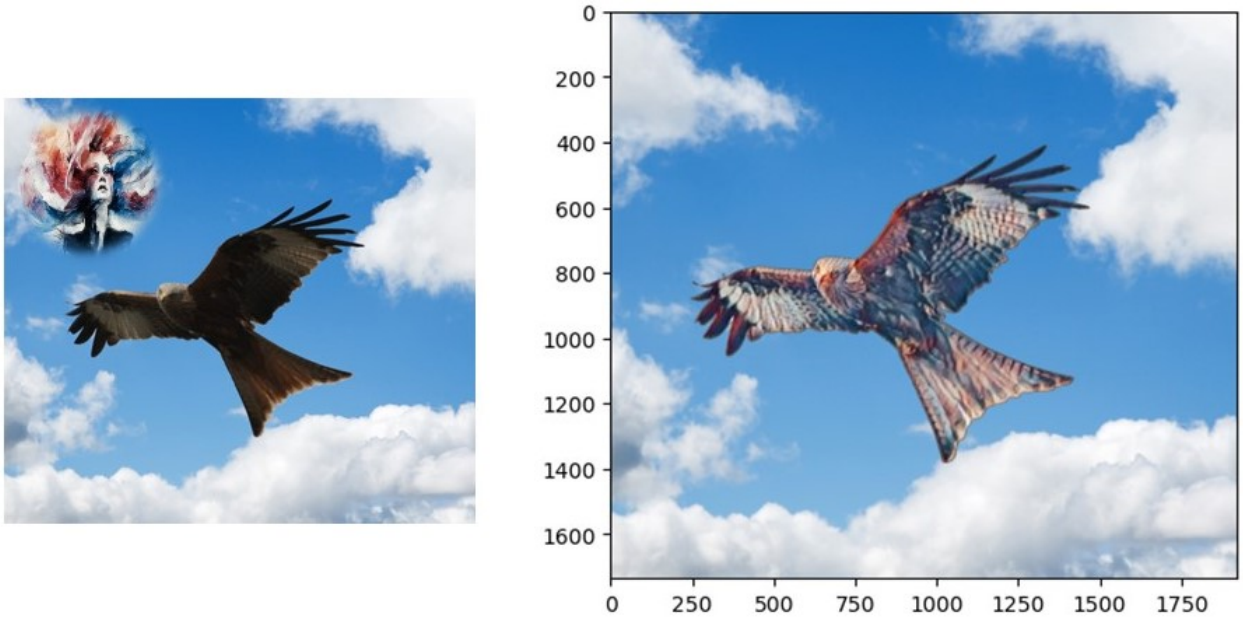

Figure 1.3: Example output using a Partial Convolution algorithm for style transfer. A content, style image (left) and a given masked object (the bird), the partial convolution style transfer algorithm can apply the stylization to the masked region exclusively (right).

Partial convolution is a convolution technique that can properly convolve across masked regions. Partial Convolution was selected for its inherent robustness for mask-aware processing and seamless integration into previously trained CNNs. By using this at each layer of the network and handling the resized mask at each step, the style transfer can be more accurate to the original style. An example output of the partial-convolution-based approach is shown in Figure 1.3.

To verify the approach, the results are analyzed both qualitatively and quantitatively. Partial convolution results will be compared to naive alternatives. Additionally, the results are numerically analyzed to determine why and when partial convolution does better than alternatives. Specifically, I find that the level of contrast in the image versus the masked region can indicate the degree to which partial convolution will improve the stylization. Lastly, this work also provides the insight into why users think a stylization is good or bad based on the level of contrast within the stylization.

In summary, the contributions of this thesis are as follows

- A robust style transfer network that uses partial convolution in each layer.
- Qualitative results showing improvement of stylization in segmented images.
- A quantitative analysis and comparison of style transfer results, providing insights into when partial convolution perform betters and what causes users to prefer one stylization over another.

Chapter 2

Background

2.1 Style Transfer

Modern style transfer techniques begin with the groundbreaking work of Gatys et al. [2] who presented the first CNN-based style transfer algorithm. This approach was optimization-based and operated on a single image and single style at a time. Others works improved the speed and control of this approach [3, 9, 20].

Later, a feed-forward method was presented by Johnson et al. [6] that trained a network for a single style image. After this training process, any content image could be fed into the network and stylized in a quick feed-forward pass. Others also improved this approach [10, 21, 23].

In 2017, Li et al. [14] reformulated style transfer as a modified image reconstruction process. After a standard image reconstruction network is trained (usually an autoencoder), the content image is fed into the network and the intermediate representation is altered based on the style statistics. This approach is considered state-of-the-art and has CNN [13], vision transformer [1], and even diffusion network [24] implementations.

For this thesis, I use the implementation of "Learning Linear Transformations for Fast Image and Video Style Transfer" [13]. This network is used because it is a fully convolutional neural network, making it easy to modify the convolution layers to be partial convolutions.

2.2 Segment Anything Models

Segmentation is a common computer vision task. Before SAM, modern AI-based approaches were class-based, relying on class labels to complete the segmentation [17]. Approaches included ResNet [4] and Detectron [22]. The Segment Anything Model [8] is unique because it was trained in an iterative fashion that does not require class labels, thus allowing it to segment objects, even if they have not been seen by the network before or have no semantic meaning. Many works have built on this approach [18, 26].

2.3 Segmented Style Transfer

Not many works attempt to complete style transfer on segmented regions. Other approaches use segmentation to influence style transfer [25, 15] or use class-based labels [12, 11]. Recently, a work titled SAMStyler [19] has combined the Segment-Anything Model with Style Transfer techniques, but relies on the slow optimization-based approach of the original Gatys implementation. In this work, I will present an approach that completes classless style transfer on segmented regions in a fast feed-forward way.

Chapter 3

Methodology

In this section, I explain the original network used for Style Transfer, how the network is modified using Partial Convolution, and the qualitative and quantitative metrics I will use to evaluate this new network.

3.1 Style Transfer by Linear Transformation

For this work, I use the Linear Style Transfer network model proposed by Li *et al.* [13]. I use this model because it has state of the art results and is able to integrate partial convolution. The model contains two feed-forward networks, a symmetric encoder-decoder image reconstruction module and a transformation learning module, as shown in Figure 3.1. The encoder-decoder is trained to reconstruct any input image faithfully. It is then fixed and serves as a base network in the remaining training procedures for stylization. The transformation learning module contains two small CNNs which takes features of the content and style images from the encoder, and outputs a transformation matrix T. The image style is transferred through linear multiplication between the content features and the transformation matrix T in the same layer. A pre-trained and fixed VGG-19 network is used to compute style losses at multiple levels and one content loss in a way similar to the prior work [7, 5]. The model is a pure feed-forward convolutional neural network, which is able to transfer arbitrary styles efficiently (140 fps).

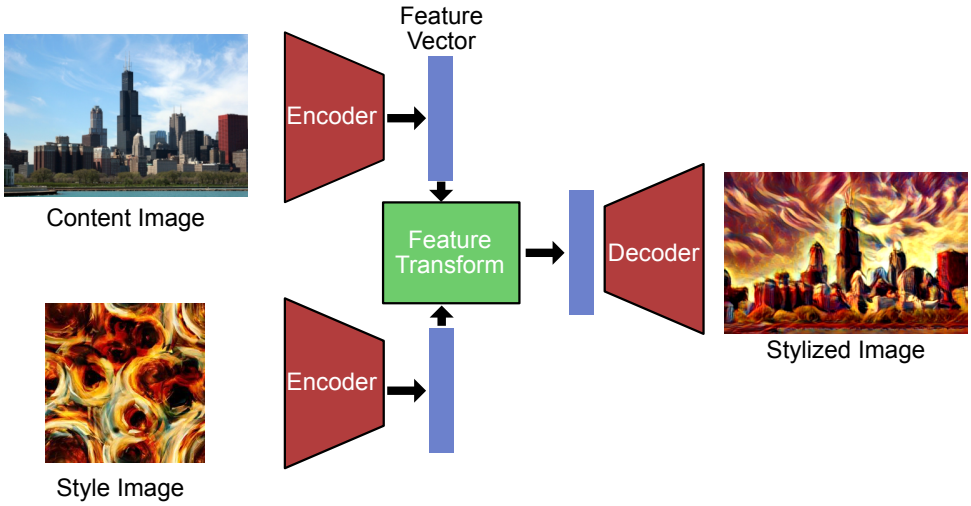

Figure 3.1: Overview of the original style transfer network. The model contains a pre-trained encoder, decoder, and a transformation module. Only the transformation module is learnable during the style transfer training, while all the others are fixed. The transformation module takes the content and style features and outputs a learned feature transform T. The content features are multiplied by T and then fed into the decoder to generate the final output.

3.2 Integrating Partial Convolution into the Style Transfer Network

Partial convolution was first proposed by Liu *et al.* as way of improving inpainting techniques [16]. The partial convolution not only acts on an image, but is provided a mask that indicates which pixels to include and which to ignore. Only pixels within the mask are multiplied by the learned weights and affect the convolution output. A visual of this method is provided in Figure 3.2.

In addition to inpainting, partial convolution can also be integrated into the style transfer task since it can replace any standard convolution layer. Partial convolution was also selected for its natural mask-aware processing. This processing is robust and flexible to any mask and image content and seamlessly handle boundaries in the mask during the convolution operation.

In the original inpainting work, partial convolution layers replaced regular convolution layers during the training process. In comparison, the style transfer network was not trained with partial convolutions. However, partial convolutions can replace the regular convolution

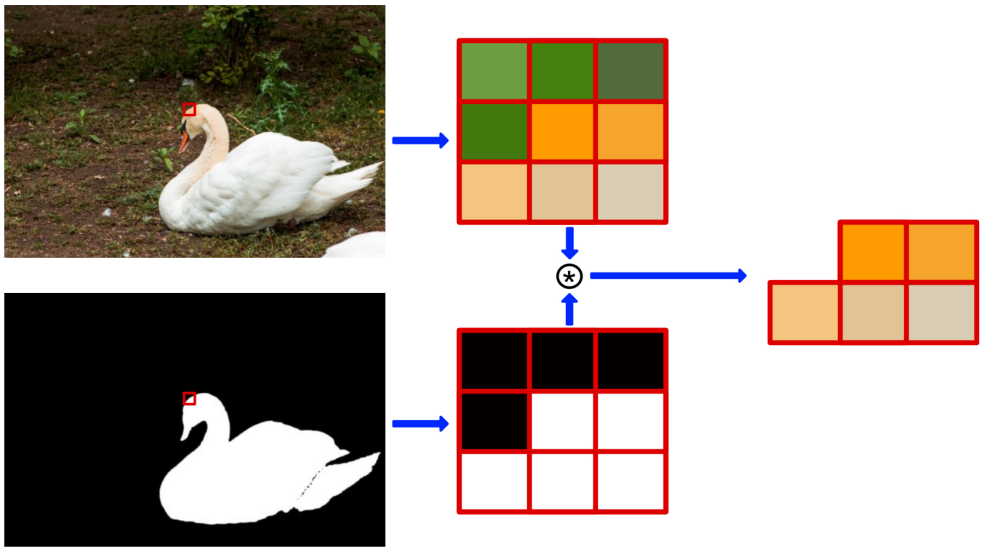

Figure 3.2: Visualization of the Partial Convolution layer. As the kernel for the CNN moves across the image, it is multiplied by the mask for the image, causing aggregation to only occur across pixels within the mask. This can effectively separate foreground information from background information.

in the style transfer network as long as the same convolution weights are used. No additional fine-tuning is needed.

To set up the partial convolution style transfer network, every convolution layer in the encoder, decoder, and transformation blocks is replaced with partial convolutions, making sure to transfer over the same weights. When providing an input image, the input mask is also provided. In the encoder, the mask goes through each padding and pooling layer that the image goes through to guarantee they stay the same size. On the decoder, bilinear interpolation of the original mask is used to guarantee the input mask is the correct size at each layer. Finally, the matrix multiplication that performs the style transformation is also masked at the feature level. The final output is alpha blended with the original image to place the stylized region back onto the background. All of these steps modify the style transfer network to only stylize the region inside the masked area without influencing the stylization with image values outside the masked area. The code for this new network is provided in a GitHub respository at https://github.com/HadiSeyed/analyzing-segmented-style-transfer.

3.3 Segmented Style Transfer Techniques

The partial convolution network can perform style transfer on segmented images. I compare it to two naive approaches for segmented/masked style transfer. This gives three total segmented style transfer approaches:

- **Style-then-mask**: Stylize the whole image, then mask the stylized output.
- **Mask-then-style**: Mask the image, then stylize the modified.
- **Partial Convolution**: Provide the mask in the style network using partial convolution.

I will describe each of these methods in detail.

Style-then-Mask

This method applies style transfer to the entire image first and then applies a mask to keep or remove parts of the stylized output. The process is as follows:

1. Apply Style Transfer Globally: The entire content image undergoes a style transfer.

2. Apply Mask After Styling: After the stylization is complete, a binary mask is applied to the result. The mask dictates which parts of the stylized image are retained, and which parts will revert back to the original content after an alpha blending.

This pipeline is illustrated in Fig. 3.3. Visually, I find that this tends to lead to *under stylized* results. Since the whole image affected the stylization, the stylization in the region of interest does not usually capture all of the style details that were present in the style image. For most styles, this leads to darker colors in the stylizations than are present in the overall style image. An example output is shown in Fig. 3.4.

Mask-then-Style

In this method, the mask is applied first, removing all other content from the image. The process is as follows:

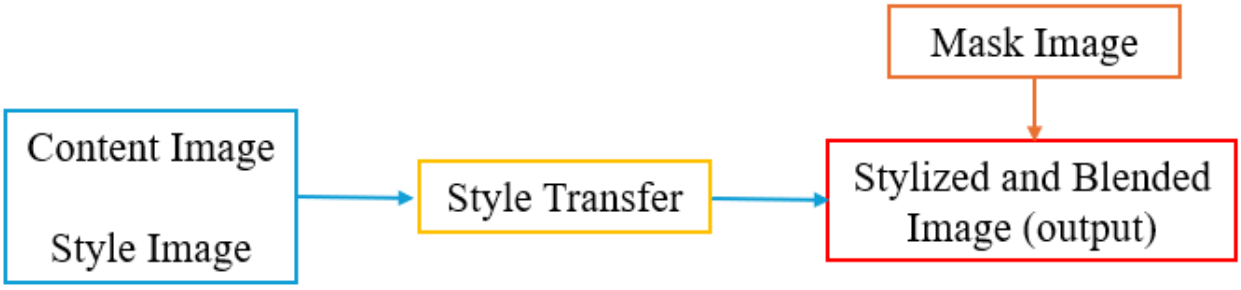

Figure 3.3: Pipeline for the Style-then-Mask algorithm. The mask for the content image is only applied during the blending stage.

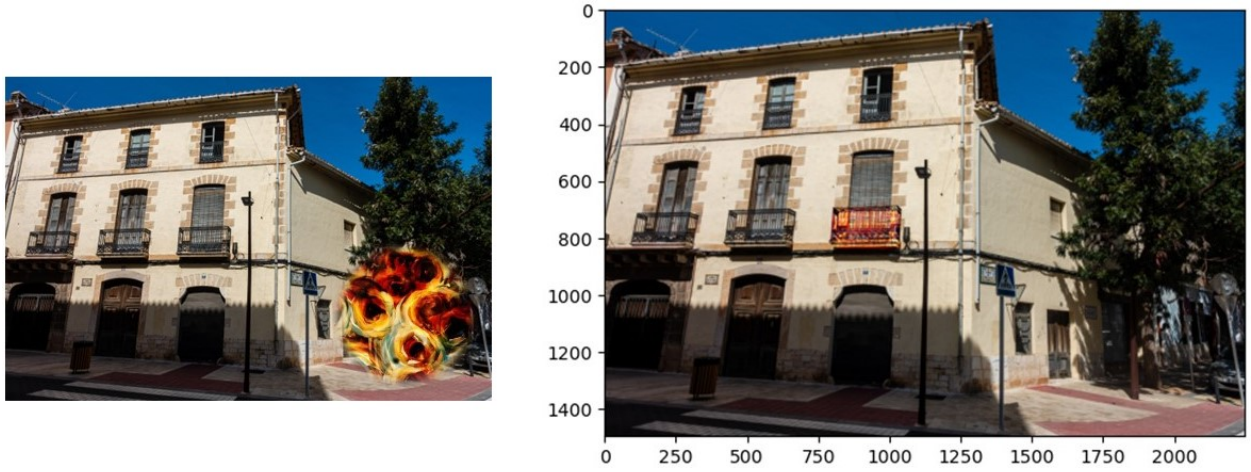

Figure 3.4: Example output using the Style-then-Mask algorithm. A content image and style image (left) give the following stylized output for the masked region (right). The result stylization tends to be darker than the input style features.

1. Apply Mask to the Content Image: A binary mask is applied to the content image to select specific regions where the style transfer will occur. The pixels surrounding the region of interest become black.

2. Apply Style Transfer to the Modified Image: The whole modified image is stylized. The output is alpha blended using the mask to place the stylized output onto the original content image.

This pipeline is illustrated in Fig. 3.5. Visually, I find that this tends to lead to poor results. When masking beforehand, the region of interest is surrounded by constant dark pixels. This makes the region of interest appear as a very bright object against a dark background. Thus, the stylization tends to make very bright outputs when using this technique and these outputs tend to have poor quality and do not match the style features. An example output is given in Fig. 3.6.

Partial Convolution

This method applies style transfer to the unmodified image, but the mask is provided the style transfer network with partial convolutions replacing regular convolution layers as described in Section 3.2. The process is as follows:

1. Modify the Style Transfer Network: Modify the style transfer network to use partial convolution while maintaining the original weights.

2. Apply the Modified Style Transfer Network: Input content, style, and mask into the modified style transfer network.

3. Apply Blending After Stylization: Since partial convolution was used in the neural network, the output is already masked. The output is alpha blended with the original content based on the mask.

This pipeline is illustrated in Fig. 3.7. Visually, I find that partial convolution better matches the style image statistics since no stylized pixels are removed from the final output during

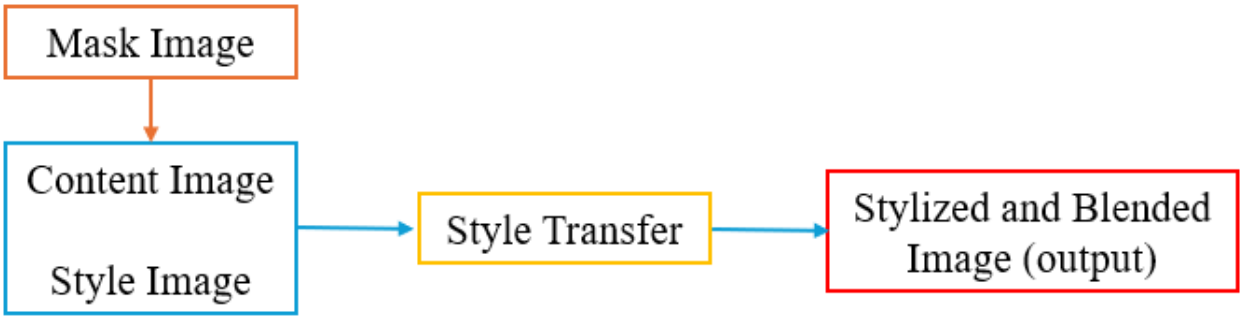

Figure 3.5: Pipeline for the Mask-then-Style algorithm. The mask for the content image is applied before the image is fed into the style transfer network.

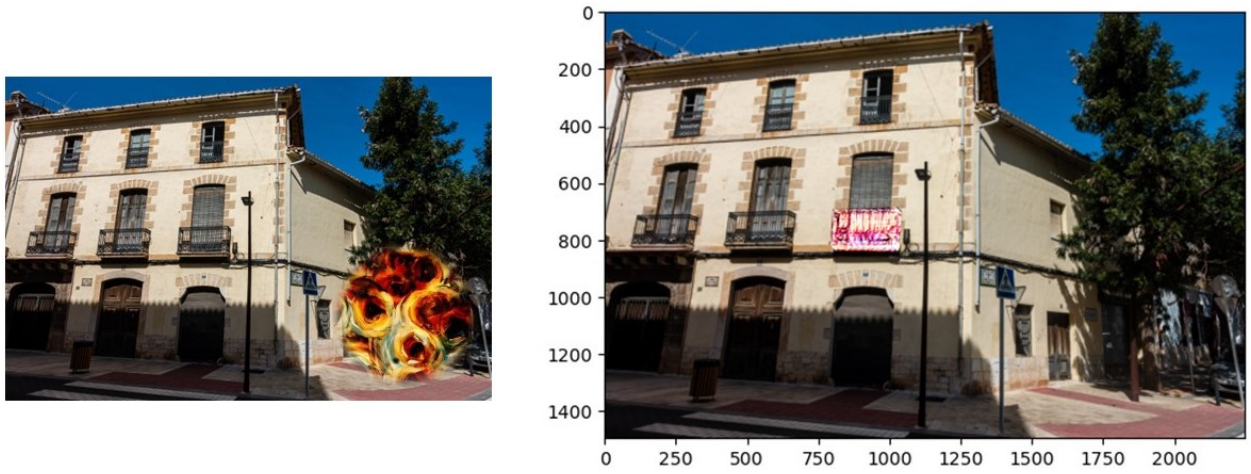

Figure 3.6: Example output using the Mask-then-Style algorithm. A content image and style image (left) give the following stylized output for the masked region (right). The result stylization tends to be much brighter than the input style features.

the alpha blending. General style transfer focuses on globally blending content and style features across the entire image while partial convolution aims at selectively applying style only in regions where data is missing or masked. Also, in style transfer, the style is applied globally, regardless of whether parts of the image are masked, but in partial convolution, convolutions are applied only to valid regions, resulting in more controlled style application in selected areas. An example output is given in Fig. 3.8.

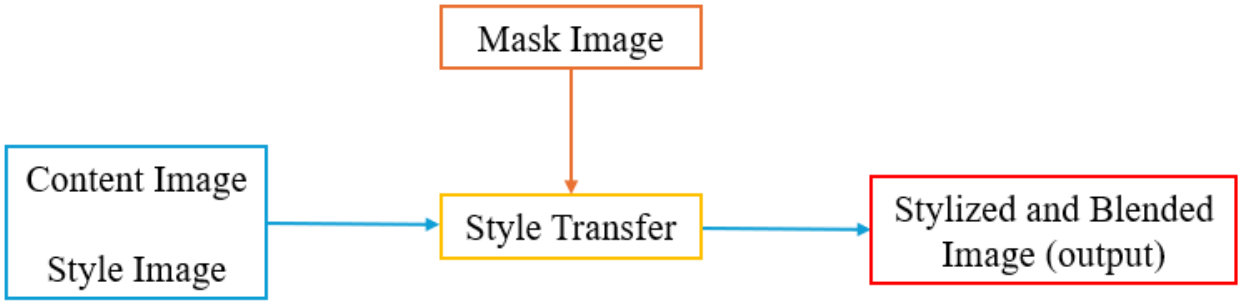

Figure 3.7: Pipeline for the Partial Convolution algorithm. The mask for the content image is applied and the partial convolution layer used at each stage of the style transfer network.

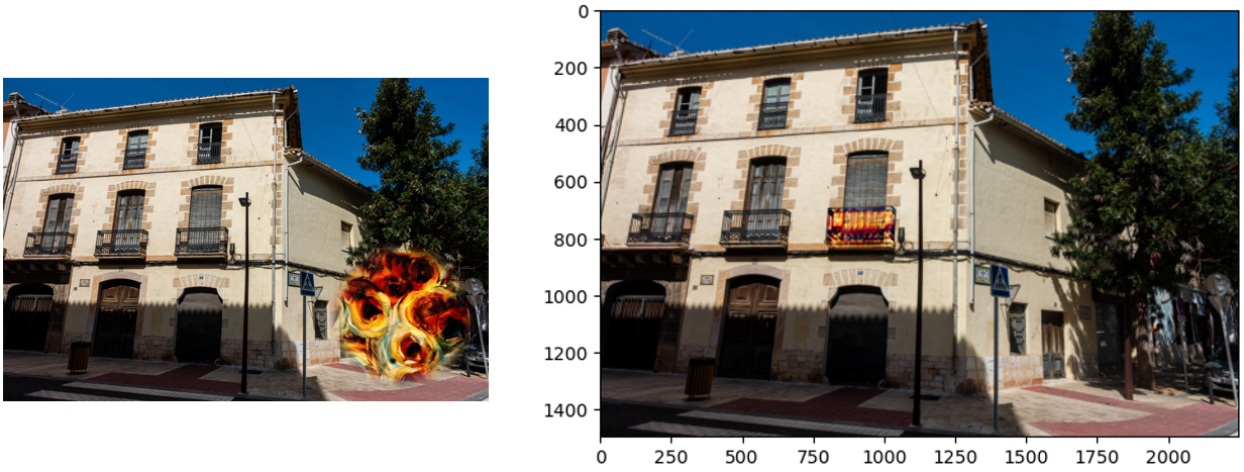

Figure 3.8: Example output using the Partial Convolution algorithm. A content image and style image (left) give the following stylized output for the masked region (right). The result stylization tends to be closer to the input style features than the other two approaches.

3.4 Predicting Which Technique Will Perform Best

In Chapter 4, the outputs of all three segmented style transfer techniques are visually compared side-by-side. This qualitative comparison is ultimately the best comparison for a task that is visual and subjective in nature. What I found was the partial convolution usually stylizes the best, but sometimes style-then-mask performs similarly to partial convolution, and sometimes even better. Mask-then-style appears to never stylize as well as the other two techniques.

Since partial convolution is not the best technique in every case, the last part of this work is to understand what conditions in the initial image leads to a particular method doing better than another. A quantitative analysis between the inputs and outputs of the techniques is conducted. Specifically, I analyze the statistics on the following:

Data

- Original Image
- Masked Region
- Style Image
- Style-then-Mask Output
- Mask-then-Style Output
- Partial Convolution Output

Statistics

- Mean RGB Color Value
- Mean Grayscale Value
- Standard Deviation of RGB Color Value
- Standard Deviation of Grayscale Value
- Contrast in RGB Color (Max Value - Min Value)
- Contrast in Grayscale (Max Value - Min Value)

Additionally, distributions in RGB and grayscale color values can be compared for each kind of data. The findings of this analysis are presented in Chapter 4.

Chapter 4

Results

In this chapter, I present qualitative comparisons between the different style transfer techniques, as well as quantitative analyses of the inputs and results.

4.1 Qualitative Comparisons

For understanding which method works best, I compare all of them side by side for different masks sizes such as large, medium, small, and very small masks. The following process was used to generate results.

First, the content image was loaded and fed into the Segment Anything Model (SAM). SAM then generates and saves masks for each distinct region of interest in the image. Next, the original content image is displayed. An interactive visualization tool was developed using OpenCV where masks from SAM are highlighted and overlayed on the image. Masks are selected interactively with one click to choose only the regions of interest. Here, all masks combine into a single mask that will be applied uniformly across the image. An alpha channel is created from the combined mask to manage transparency and layering, and then the image is normalized so that the values fall within the desired range for processing if needed. Then, the content, style, and mask images are processed as tensors to be fed into the style transfer techniques. The three style transfer techniques are implemented in PyTorch. Finally, the three outputs are alpha blended with the original image to map the stylized regions back onto the background.

Figure 4.1: Example outputs from the three techniques - part 1: style-then-mask (left), mask-then-style (middle), partial convolution (right). content images and style images give the following stylized output for the masked region.

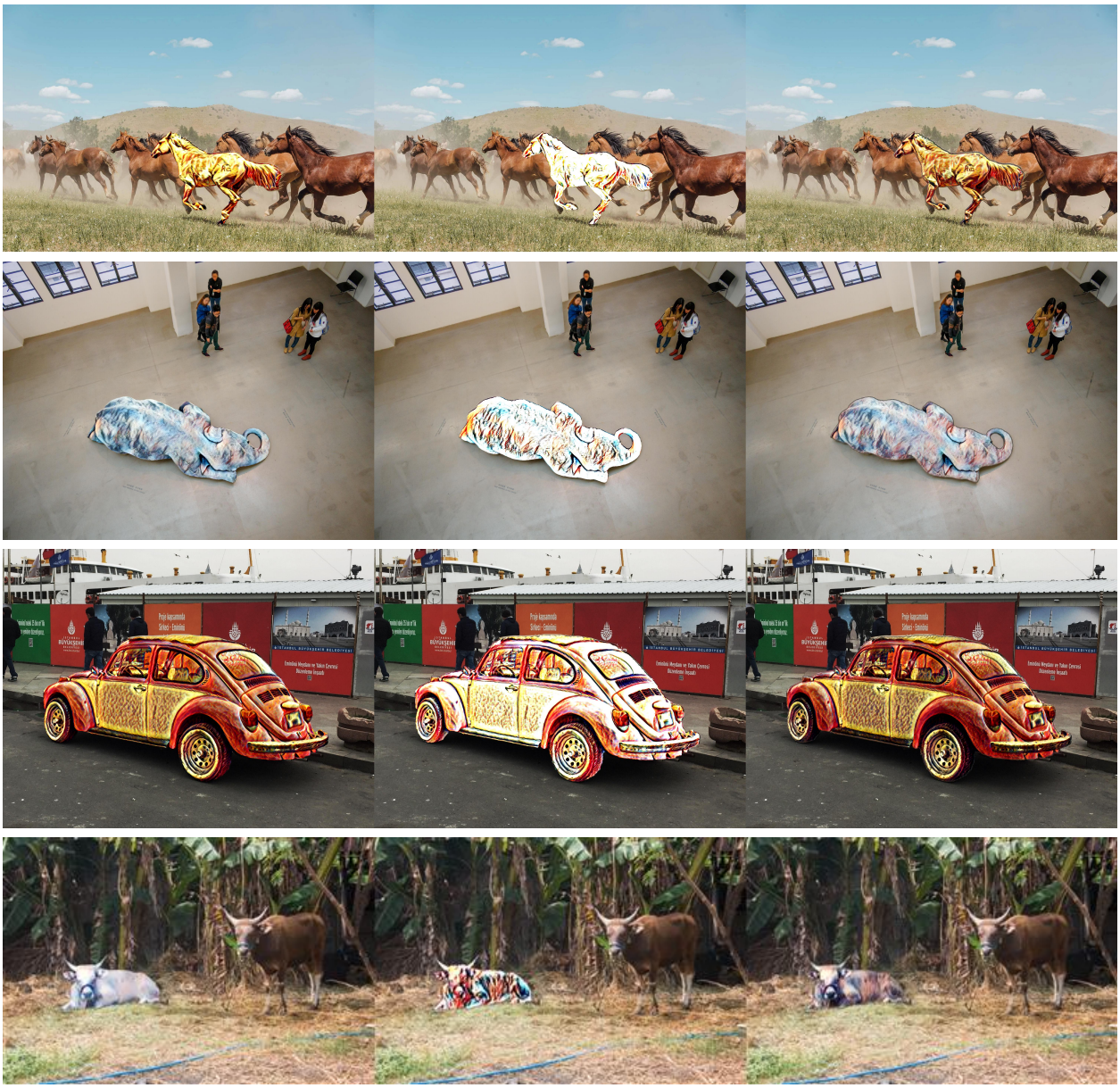

Figure 4.2: Example outputs from the three techniques - part 2: style-then-mask (left), mask-then-style (middle), partial convolution (right). content images and style images give the following stylized output for the masked region.

Figure 4.3: Comparison of the three techniques - part 1: style-then-mask (left), mask-then-style (middle), partial convolution (right) - partial convolution is better. A content image and style image (top) give the following stylized output for the masked region (bottom). The result stylization from partial convolution tends to be closer to the input style features than the other two approaches.

Figure 4.4: Comparison of the three techniques - part 2: style-then-mask (left), mask-then-style (middle), partial convolution (right) - style-then-mask is better. A content image and style image (top) give the following stylized output for the masked region (bottom). The style-then-mask approach matches closer to the content and partial convolution matches closer to the style.

Many visual comparisons are provided for different content images, style images, and masks sizes in Figures 4.1 and 4.2. From these visuals, some observations can be made.

First, the mask-then-style method has brighter outputs then the other two and usually has poor results. In general, mask-then-style is not a good approach for segmented stylization.

Second, partial convolution does better than the other two in most cases. This is because the partial convolution approach can better match the statistics within the style for the masked region, with style-then-mask usually being too dark and mask-then-style being too bright. A clear visual of this behavior is shown in Figure 4.3.

Third, in some cases, style-then-mask and partial convolution results look the same, usually in large masks. For small masks, style-then-mask can sometimes do better. Also, visual quality is ultimately a subjective matter and it can be unclear which one did better. Style-then-mask tends to hold to the content better and partial convolution holds to the style features better. A good example of this is shown in Figure 4.4.

4.2 Quantitative Analysis

In this section, the aim is to understand why partial convolution does better on some images but not on others. Ultimately, that difference has to depend on the content, style, and mask inputs. During the stylization process, statistics about the content image, the masked region of the content image, and the style image were saved to a JSON file for each experiment. By analyzing these saved statistics, the effects they have on the final stylized outputs can be better understood for each of the three methods.

Overall, I present the following hypothesis: partial convolution stylizes the best when the distribution of colors in the masked region is different than the distribution for the whole image. Specifically, partial convolution stylizes better when the contrast in the masked region is smaller than the contrast in the whole image. This conclusion is supported with two experiments as presented in the following subsections:

4.2.1 Comparing Distributions

First, a visual comparison between the original and mask images is provided. The plot-channel-histogram function is defined to create a histogram plot for the image channels. The histogram is normalized by dividing by the total number of pixels in the image or mask to ensure accurate relative frequency representation. The red, green, blue, and grayscale historgrams are plotted separately.

The histograms for the bird image in 4.3 are plotted in Figure 4.5. It can be seen that the distributions for the full content image versus the masked region are very different. This difference leads to the better visual output for partial convolution.

4.2.2 Analyzing Contrast

In order to accurately analyze the input image, it is first necessary to categorize the output into three categories:

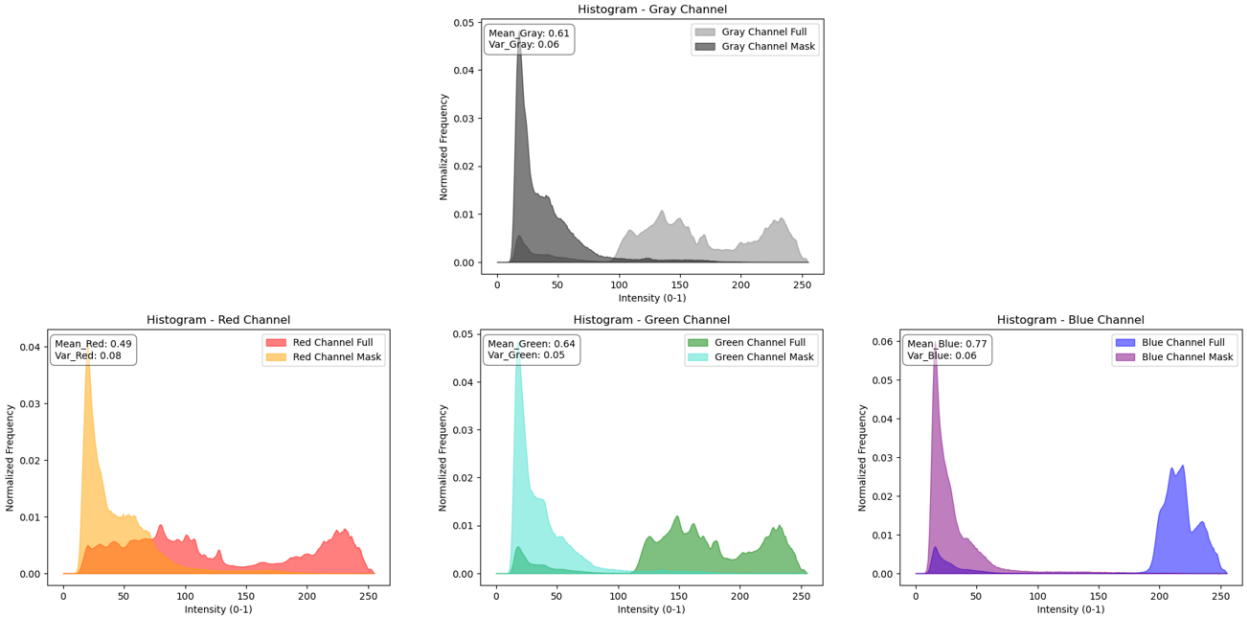

Figure 4.5: The histograms of bird image in red, green, blue channels and grayscale. The statistics for original image and masked region are overlayed on each other. The histograms show that the distribution of colors for the full image and masked region are disparate. This leads to partial convolution outperforming the other two methods.

- **partialconv-better**: When partial convolution had better stylization than style-then-mask
- **same**: When partial convolution had similar stylization than style-then-mask
- **stylethenmask-better**: When style-then-mask had better stylization than partial convolution

Mask-then-style was not included because there were no cases when it performed better. For each input image and mask, the outputs were visually analyzed to determine which method performed best. Then, the associated JSON statistics files were separated into different folders based on that categorization. Finally, I gathered data for different categories, loaded and processed data from each category by iterating through partialconv-better, same, and stylethenmask-better directories to generate a scatter plot for all JSON files. This allows for a visual comparison of the extracted metrics across different categories. The figures were saved to review results and retain a copy for further analysis.

For this experiment, 36 different images were loaded and their outputs categorized. The

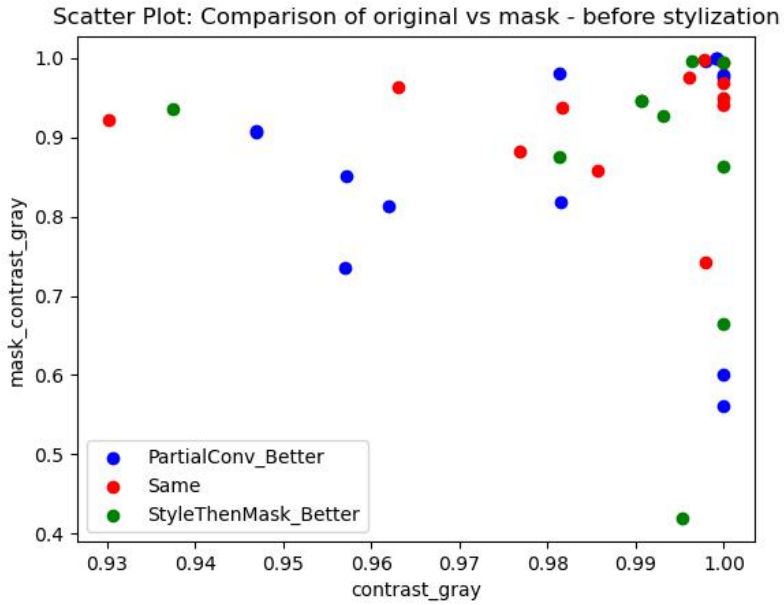

Figure 4.6: Contrast scatter plot in the gray channel in the three categories: partialconv-better, same, stylethenmask-better. Statistics for 36 images are plotted to show the partial convolution method does better where the content image contrast in grayscale is high, but masked region contrast is low.

JSON files were read independently for each category and the statistics were plotted for comparison. The content image, masked region of the content image, and the style image statistics were computed. Mean, standard deviation, and maximum contrast for both RGB and grayscale were analyzed. All possible pairings of statistics and data were considered across all color channels.

After completing this thorough analysis, two statistics seemed to be most closely correlated with when partial convolution does better. Specifically, the contrast in the gray channel appears to be the key factor. Partial convolution does better when the content image contrast is high but masked region contrast is low. The scatter plot showing this pattern is given in Figure 4.6. Intuitively, this makes sense since partial convolution will do a better job of spreading the style across the masked region statistics. An examples of this phenomenon are shown in Figure 4.7.

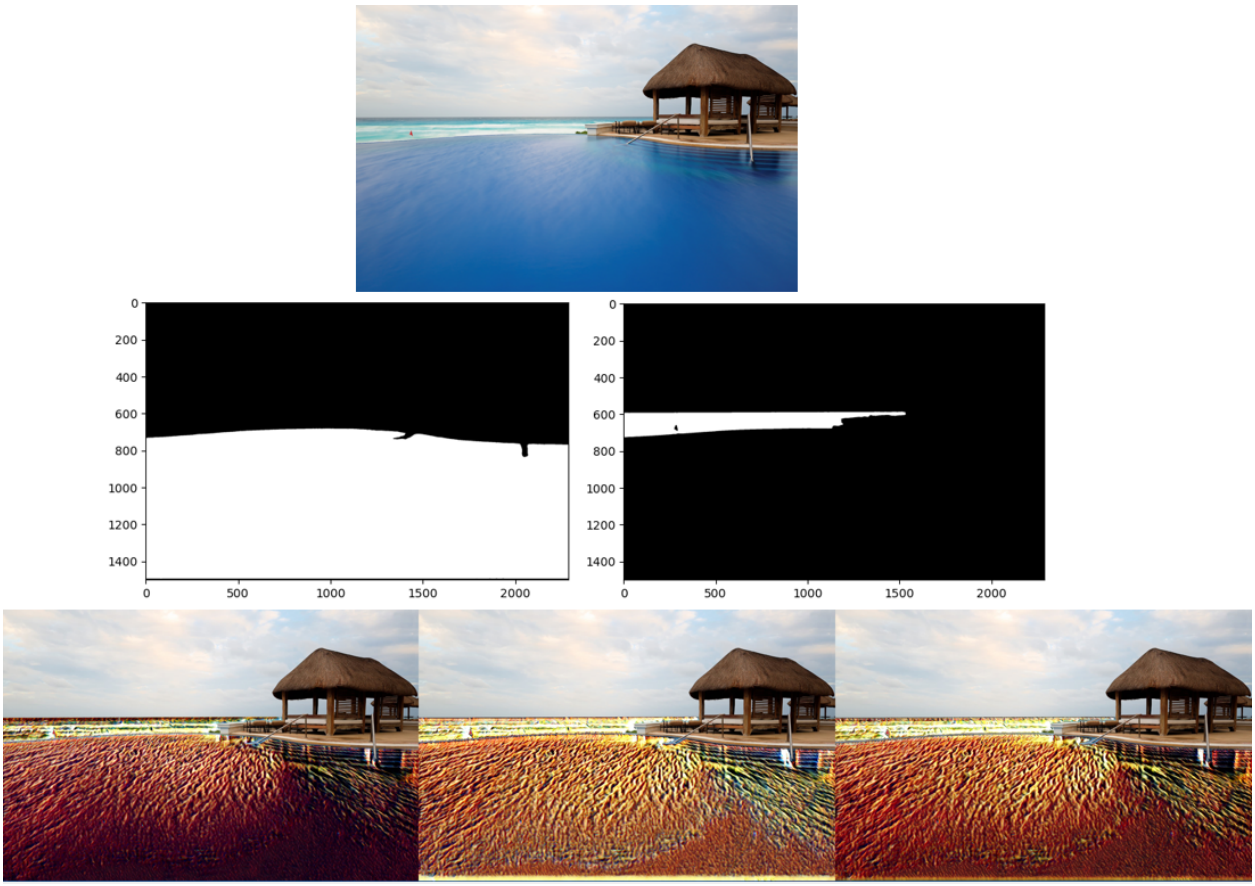

Figure 4.7: Example with different contrast in the contrast image vs the masked region,before and after stylizations. A content image, style image, and mask images give the following stylized output for the masked region (bottom). The result stylization shows the partial convolution will do a better job of spreading the style across the masked region statistics than style-then-mask method due to high contrast.

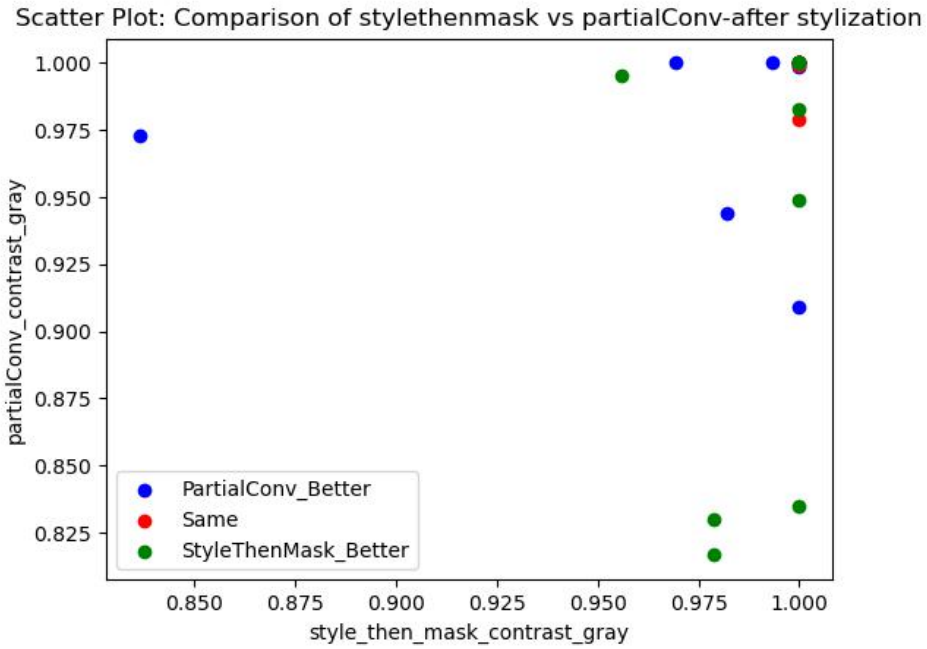

Figure 4.8: Visualization of grayscale contrast values for output stylizations. This plot leads to the insight that contrast in the output may be linked to the selection of the best style method. For example, when the style-then-mask had high contrast, but partial conv did not, style-then-mask was selected as the better method.

4.2.3 Insight on Stylization Preferences

In addition to the content, masked region, and style statistics. The statistics on the three output images were computed. As part of this analysis, an interesting insight was discovered. In terms of categorizing which method performed best, it appears to be highly correlated with the contrast of the output stylization. The stylization that has the highest contrast tends to be selected as the best method. Similar contrasts in the output were often marked as performing the same. This might indicate that contrast can be used to understand user preferences in stylization. A visualization of this pattern is given in Figure 4.8. Two examples illustrating contrasts effect on preference are given in Figures 4.9 and 4.10.

Figure 4.9: Example 1 output of contrast insight on stylization preferences. Style-then-mask (left), mask-then-style (center), and partial convolution (right) style outputs. The result stylization shows the partial convolution contrast is high, but style-then-mask contrast is low, so partial convolution is more visually appealing since it does a better job of spreading the style across the masked region statistics than style-then-mask method.

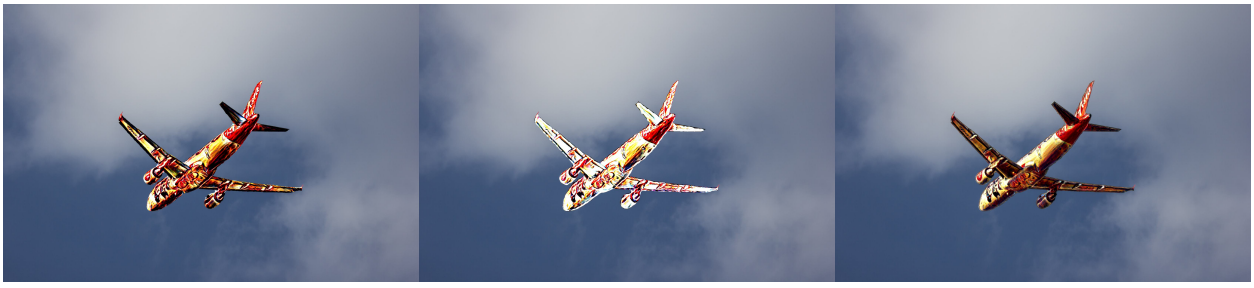

Figure 4.10: Example 1 output of contrast insight on stylization preferences. Style-then-mask (left), mask-then-style (center), and partial convolution (right) style outputs. The result stylization shows that style-then-mask and partial convolution both have high contrast, so both would have similar user preferences.

Chapter 5

Conclusion

This work presented an effective technique for combining style transfer with the segment anything model. It showed that introducing partial convolution into the style transfer network created stylization that more closely matched the style image statistics. It verified this evaluation through both quantitative and qualitative means. It compared multiple segmented stylization techniques and analyzed them through side-by-side visuals. Last of all, statistics were calculated in the masked and unmasked region and shown to play a significant role in the output stylization. Specifically, the contrast is a key factor for determining the effectiveness of stylization with partial convolution.

5.1 Limitations

Combining Style Transfer with the Segment Anything Model (SAM) offers creative control over selective areas of an image, but there are some limitations to be aware of:

- Boundary Precision: SAM segments may not perfectly align with edges in complex images, which can lead to blending artifacts when style transfer is applied only to selected regions. This can make the styled area look unnatural or mismatched with the surrounding content.
- Consistency Across Frames: In videos or multiple-image sequences, SAM may produce slightly different masks for each frame, making it challenging to achieve consistent style transfer across frames, resulting in flickering or temporal distortion.
- Detail Loss: Applying style transfer only to specific regions can sometimes blur or alter fine details within those areas, especially if the style heavily emphasizes textures or colors, potentially reducing important visual features within the segmented areas.

- Performance Overhead: Combining SAM with style transfer requires multiple processing steps, such as segmenting, masking, and transferring styles, which can be computationally intensive, especially for high-resolution images or real-time applications.
- Limited Adaptability to Complex Textures: SAM's segmentation may struggle with highly intricate or overlapping textures, which can hinder precise masking and reduce control over where the style is applied. This can be especially challenging when using complex or textured styles that require accurate segmentation.
- Fixed Semantic Segmentation: SAM does not inherently prioritize semantic information (e.g., distinguishing background from foreground in every context), so certain style transfers may apply to unintended areas if the segmentation is not carefully refined.

These limitations highlight the need for careful mask selection and tuning when integrating SAM with style transfer to achieve a natural, cohesive look in stylized images.

5.2 Future Work

As future work, the following directions could be explored:

- Implementing additional distribution metrics, as well as comparing stylized results with perceptual loss common to the literature.
- Completing an ablation study across multiple configurations of the network.
- Using machine learning algorithms to develop a more precise mathematical model for how image and mask values affect the performance of partial convolution.
- Comparing techniques in the context of stylizing multiple regions at a time, each with its own style image.
- Doing a user study to see if more people agree on the best stylization result for each image.
- Computing the statistics with more images. The prediction scatter plot was generated with only 36 images. Lots of additional images would further solidify the findings.

5.3 Potential Applications

The fusion of segmentation with style transfer opens up new possibilities in both artistic and practical domains by enhancing control over how and where style transfer is applied. SAM's ability to segment specific objects or regions in an image will allow for more precise control

over which parts of an image are stylized and which remain in their original form, unlike traditional style transfer, which affects the entire image. Additionally, artists will be able to use SAM to segment different objects or regions interactively and apply different styles to each. Flexible tools that can be created to improve control of both the mask and stylization to give graphic designers, photographers, and other digital artists more expressive control and while removing difficult or time-consuming barriers.

BIBLIOGRAPHY

[1] Deng, Y., Tang, F., Dong, W., Ma, C., Pan, X., Wang, L., and Xu, C. Stytr2: Image style transfer with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2022), pp. 11326–11336.

[2] Gatys, L. A., Ecker, A. S., and Bethge, M. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 2414–2423.

[3] Gatys, L. A., Ecker, A. S., Bethge, M., Hertzmann, A., and Shechtman, E. Controlling perceptual factors in neural style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 3985–3993.

[4] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778.

[5] Huang, X., and Belongie, S. J. Arbitrary style transfer in real-time with adaptive instance normalization. *CoRR abs/1703.06868* (2017).

[6] Johnson, J., Alahi, A., and Fei-Fei, L. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14* (2016), Springer, pp. 694–711.

[7] Johnson, J., Alahi, A., and Fei-Fei, L. Perceptual losses for real-time style transfer and super-resolution. *CoRR abs/1603.08155* (2016).

[8] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2023), pp. 4015–4026.

[9] Kolkin, N., Salavon, J., and Shakhnarovich, G. Style transfer by relaxed optimal transport and self-similarity. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019).

[10] Kotovenko, D., Sanakoyeu, A., Ma, P., Lang, S., and Ommer, B. A content transformation block for image style transfer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019).

[11] Kulkarni, H., Khare, O., Barve, N., and Mane, S. Improved object-based style transfer with single deep network, 2024.

[12] Kurzman, L., Vazquez, D., and Laradji, I. Class-based styling: Real-time localized style transfer with semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision Workshops* (2019), pp. 0–0.

[13] Li, X., Liu, S., Kautz, J., and Yang, M.-H. Learning linear transformations for fast image and video style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 3809–3817.

[14] Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., and Yang, M.-H. Universal style transfer via feature transforms. *Advances in neural information processing systems 30* (2017).

[15] Lin, Z., Wang, Z., Chen, H., Ma, X., Xie, C., Xing, W., Zhao, L., and Song, W. Image style transfer algorithm based on semantic segmentation. *IEEE Access 9* (2021), 54518–54529.

[16] Liu, G., Reda, F. A., Shih, K. J., Wang, T.-C., Tao, A., and Catanzaro, B. Image inpainting for irregular holes using partial convolutions. In *The European Conference on Computer Vision (ECCV)* (2018).

[17] Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 3431–3440.

[18] Ma, J., He, Y., Li, F., Han, L., You, C., and Wang, B. Segment anything in medical images. *Nature Communications 15*, 1 (2024), 654.

[19] Psychogyios, K., Leligou, H. C., Melissari, F., Bourou, S., Anastasakis, Z., and Zahariadis, T. B. Samstyler: Enhancing visual creativity with neural style transfer and segment anything model (sam). *IEEE Access 11* (2023), 100256–100267.

[20] Puy, G., and Perez, P. A flexible convolutional solver for fast style transfers. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019).

[21] Ulyanov, D., Vedaldi, A., and Lempitsky, V. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017).

[22] Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., and Girshick, R. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[23] Yanai, K., and Tanno, R. Conditional fast style transfer network. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval* (New York, NY, USA, 2017), ICMR '17, ACM, pp. 434–437.

[24] Zhang, Y., Huang, N., Tang, F., Huang, H., Ma, C., Dong, W., and Xu, C. Inversion-based style transfer with diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2023), pp. 10146–10156.

[25] Zhao, H.-H., Rosin, P. L., Lai, Y.-K., and Wang, Y.-N. Automatic semantic style transfer using deep convolutional neural networks and soft masks. *The Visual Computer 36*, 7 (2020), 1307–1324.

[26] Zou, X., Yang, J., Zhang, H., Li, F., Li, L., Wang, J., Wang, L., Gao, J., and Lee, Y. J. Segment everything everywhere all at once. *Advances in Neural Information Processing Systems 36* (2024).