Texture Synthesis and Style Transfer with Sliced Optimal Transport

Bianca Marin Moreno
bianca.marin-moreno@polytechnique.edu

Leonardo Martins Bianco
leonardo.martins_bianco@ens-paris-saclay.fr

Abstract

We explore a method for texture synthesis and style transfer which makes use of histogram matching on the feature maps of a convolutional neural network. We perform numerical experiments to test the algorithm, and identify where it performs well and where it performs poorly. We perform statistical measurements trying to validate some affirmations made in the article studied about this method. Also, we explore a sliced optimal transport approach to do histogram matching. Finally, we discuss possible directions of advancement.

1 Introduction

This work was conducted as a final project for the course of "Imagerie Numérique" in the MVA master at École Normale Supérieure Paris-Saclay. We explored a method for texture synthesis and style transfer using histogram matching, and then implemented sliced optimal transport for this task.

The goal of texture synthesis is to develop a generating process that produces arbitrarily new samples of a given example texture. The criterion used to evaluate these new synthesized textures is usually the visual quality: if a human observer cannot distinguish the synthesized samples from the original one, then we say the generating algorithm performs well. Given a pair of images, a content and a style, the goal of style transfer is to synthesize an image preserving the semantics of the content but with the same visual characteristics as the style. The key challenge of this task is first finding an effective representation of the texture (or style), then matching it in the content image.

A common approach to texture synthesis is to define a parametric model, consisting of a set of statistical measurements taken over some representations of the image. For a model, the texture is uniquely determined by the statistical measurements, and two images with the same measurements are considered to have the same texture. Recently, with the work done by Gatys et al. [2, 3], the focus has been on using a convolutional neural network to encode the image into a feature space, and taking the correlation (Gram matrix) between the features of the encoded image to determine the statistical measurement defining its texture. Thus, the algorithm to perform style transfer consists on matching the correlations (Gram matrices) between the deep features within an iterative optimization framework.

However, one of the major drawbacks of [2, 3] is the inefficiency of the optimization process. In Li et al. [7] they formulate the transfer task as an image reconstruction process. For each intermediate layer of the deep network encoder, they apply a deterministic transform to the extracted content features to make them have the same statistical characteristics as the texture. After transformation, the features are decoded by another neural network. No more iterative optimization is performed.

The global idea of the main article studied in this project [11] is to combine the work of Gatys using iterative optimization with the work of Li et al. [7] using the auto-encoder strategy, to create a new texture synthesis and style transfer algorithm. The use of second-order summary statistics in [2, 3, 7] is heavily criticized in [11], which proposes using first order statistics of the encoding of images as features instead. This is done using histogram matching, inspired by the color transfer literature [10]. This project is about studying this method by understanding what motivates it and how it works, as well as by doing numerical experiments to perform qualitative analysis on it.

Some choices of the main method are poorly motivated in the article. For example, before doing histogram matching the author suggests performing random rotations claiming that by doing so we achieve the same goal as if we were using the Gram/Covariance matrix. However, no proof or explanation is given to validate this affirmation. Therefore, we decided to contest the utility of performing random rotations, and to calculate the Gram matrix error on texture synthesis samples generated with and without the suggested rotations. We found that visually the results seem identical and the Gram matrix error is also similar. The code for this is also provided.

We noticed that, despite the title of the article being "Fast and Robust Texture Synthesis and Style Transfer through Optimal Transport", the method proposed does not seem to use optimal transport explicitly at any moment, but only classical histogram matching. For us, it is not clear that he is trying to minimize the Wasserstein distance, nor its sliced version. Thus, we propose a different version of his method using sliced optimal transport to perform the histogram matching. We provide a code for it, and in this report we analyze some results we obtained.

This work will be divided as follows: first in Section 2 we study in detail the previous literature, as well as the theory of sliced optimal transport. Then, in Section 3 we present in detail the algorithm proposed by [11]. In Section 4 we provide a new method idea using sliced optimal transport. Section 5 is devoted to numerical experiments, where we make a qualitative and quantitative analysis of the proposed method for both texture synthesis and style transfer, and we analyze our new proposed approach. Finally, Section 6 discuss the experiments as well as some contributions and drawbacks of the method.

2 Related Work and Background

2.1 Texture Synthesis and Style Transfer with Convolutional Neural Networks

The work on texture synthesis and style transfer done by Gatys et al. [2, 3] introduces deep learning to the field. As previous works on texture synthesis, their model first aim at extracting features of different sizes from the image. Then to compute a spatial summary statistic on the feature responses to obtain a stationary description of the image. Finally, to find a new image with the same stationary description, performing gradient descent on a white noise image.

The main difference from previous works is that they use a feature space provided by a VGG-19 network trained for object recognition tasks. Each layer in the network defines a non-linear filter bank. These filters are activated in response to an image, forming a set of feature maps. Only one spatial summary statistic is then used: the correlations between the responses of different features. These features correlations are given by the Gram matrix (the correlation matrix). A set of Gram matrices for each layer of the network fully specifies a texture in this model.

Thus, to generate new texture based on a given texture example, we use gradient descent to update the pixels of a white noise image to find another that matches the Gram-matrix representation of the original image. This approach relies on standard error back-propagation, and on some numerical optimization strategy to update the pixels of the random image. The cost of this training framework is indicated in the literature as a major drawback of this method.

This approach can also be used on style transfer. It is sufficient to update the pixels of the white noise image by minimizing a loss between the white noise representation and the content representation in one layer and the style representation defined on a number of layers of the Convolutional Neural Network.

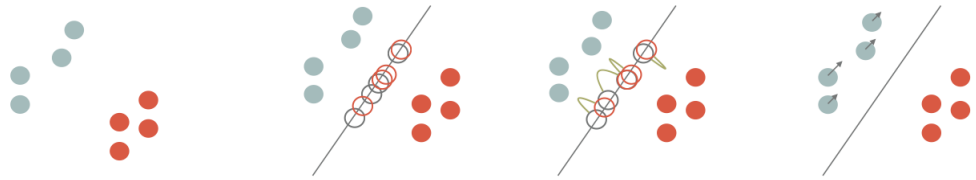


Figure 1: Visual example of sliced Optimal Transport on the discrete case.

2.2 Style Transfer with Whitening and Coloring Transforms

Among the works proposing a solution to the high cost of utilizing back-propagation training, we have Li et al. [7] introducing a simple universal style transfer algorithm based on a decoder strategy. They also employ a VGG-19 network trained for object recognition as the feature extractor (encoder). Additionally, they train a symmetric decoder to invert the VGG-19 features of the original image. Thus, applying directly the encoder and the decoder after training should result in the input image.

To perform style transfer, they apply a transformation called Whitening and Coloring transform (WCT) to the extracted features in each intermediate layer of the encoder, such that they exhibit the same statistical characteristics as the style features. Then the transformed features are sent through the decoder layers to obtain the final stylized image. The whole procedure of the method only requires learning the decoder, with no style involved.

2.3 Sliced Optimal Transport

Let $\alpha = \sum_{i=1}^{n} a_i \delta_{x_i}$, $\beta = \sum_{i=1}^{m} b_i \delta_{y_i}$ be two discrete distributions, described by their supports $(x_i)_{i=1}^{n} \in \mathbb{R}^{n \times p}$ and $(y_i)_{i=1}^{m} \in \mathbb{R}^{m \times p}$, and weight vectors such that $\sum_{i=1}^{n} a_i = \sum_{j=1}^{m} b_j = 1$. Optimal transport compares α and β by considering the most efficient way to transporting the masses a and b onto each-other, according to some cost function. When this cost is quadratic, the Wasserstein distance given by Eq. 1 corresponds to

$$W_2^2(\alpha, \beta) = \min_{P \in U(a,b)} \langle P, M \rangle \tag{1}$$

where $U(a, b) = \{P \in \mathbb{R}^{n \times m} : P1_m = a, P^T 1_n = b\}$, and M is the mean square error matrix between x and y.

There are many ways to solve the Optimal Transport problem. Here we focus on the sliced formulation of OT. The sliced formulation consists in projecting the measures α and β into a 1D line, solving the OT problem on the projections, and approach the full problem by iteratively solving 1D problems on a large number of random directions.

As we are in the discrete case (our measures are sums of Diracs), solving the OT problem in 1D consists on sorting the projections, computing the projection of the difference between points with the same sorting index, then advecting the corresponding point in the source distribution along this projected difference. A good exposition of the method is present in [5]. Numerically, we consider only a finite number of directions that we call *slices*. Figure 1 visually represents sliced Optimal Transport on the discrete case.

3 Article's Method

In this section we explain the main algorithm studied in this project presented in [11] by Risser. This algorithm can be adapted to both texture synthesis and style transfer. In general, it follows the same encoder/decoder approach of the method introduced by [7] explained in 2.2. The main difference being that instead of performing the Whitening and Coloring transform to the extracted features in each intermediate layer, Risser proposes doing random rotations and histogram matching (inspired by [10]) between the representation features of the original image and of the style image at each layer of a VGG-19.

Risser's main argument is that the second order statistics transform done by Li et al. and by Gatys has a lot of instabilities, and that matching the first order statistics by histogram matching is a more robust approach. Furthermore, Risser affirms that performing random rotations before histogram matching achieves the same goal as the Gram matrix, thus leaving only the histogram distance to be minimized.

3.1 Texture generation

Let us describe the algorithm in the case of texture generation. First, let us characterize a given image x as a set of feature maps. We pass x through a VGG (encoder). Each layer in the network can be thought as a non-linear filter bank. The activations in response to the image form a set of filtered images, also called the feature maps. Consider a layer l with N^l feature maps, each of size M_l (if vectorized). We define $F^l \in \mathbb{R}^{N_l \times M_l}$ such that F^l_{jk} is the activation of the j-th filter at position k of layer l.

Let x and $\hat{x}$ be the main texture image and a white noise image respectively, where we want to generate the texture sample. We pass them both through the VGG (encoder). Let $F^l_{\cdot k}, \hat{F}^l_{\cdot k}$ be the activations of position k of the feature maps of layer l for x and $\hat{x}$ respectively. Note that as we have N_l feature maps, $F^l_{\cdot k}, \hat{F}^l_{\cdot k} \in \mathbb{R}^{N_l}$, and k can be M_l different positions.

Our goal is to solve the histogram matching problem between F^l and $\hat{F}^l$ for each layer. However, before doing so the method suggests projecting F^l and $\hat{F}^l$ in a new random basis of $\mathbb{R}^{N_l}$, in a way that this would be equivalent as doing random rotations. Let us call F^l_r and $\hat{F}^l_r$ the activations of each layer after a random rotation.

Thus, the goal is to match the histogram of $\hat{F}^l_r$ with the histogram of F^l_r. From the article it is not clear how we should do the histogram matching. A code we found in [9] proposes doing it by matching the probability distribution functions, and also implement other types of match. However, none of them seemed to be optimal transport to us.

When this matching is done, the features with histograms matched are decoded by the inverse VGG trained to be a decoder. This procedure is repeated a certain number of times n, and for a number of random rotations per layer N_l/n, the number of channels of the layer l over the number of iterations. After all iterations, the final image decoded from the extracted features $\hat{F}^l$ of each layer l is our generated texture.

3.2 Style transfer

We can modify this algorithm to perform style transfer. This task introduces a second exemplar image called the "content image" C. Before passing through the encoder, we align C by subtracting out its mean and adding the mean of the texture image S. Then, using the same scheme as for texture generation, C and S pass through the VGG (encoder). Their features F^l_C and F^l_S are extracted. The histogram matching is now done between the feature map of the random noise image, and the weighted feature map between the "content" and the style given by $F^l_S + \alpha(F^l_C - F^l_S)$. Where α serves for users to control the transfer effect, the closer to one, the bigger is the weight given to the content.

This operation is called content matching in [11]. For this algorithm this is performed only with the features extracted on the 3 final layers of the VGG. The parameter α is divided by 2 in the anti-penultimate layer, and by 2 in the penultimate layer.

3.3 Disclaimer

Although the article [11] shows interesting results with this method, some of its ideas lack proofs or explanations. The principal one is the random rotations before performing histogram matching. Risser claims that this process helps to robustly matches feature independence. Nonetheless, we contest its pertinence as no proof is provided. In Section 5 we perform quantitative and qualitative analysis to explore the relevance of this approach. We test producing textures with and without the random rotations, and we compare the results visually, and by calculating the error between their

Gram matrix and the one of the original texture. We see that rotation does not seem to change the quality of the result.

In addition, it is unclear to us where in the method it is used optimal transport theory, as it seems the algorithm only solves classical histogram matching problems. As the title of the project claims using optimal transport to perform texture synthesis we decided to code a modification of Risser's method using sliced optimal transport to match the histograms. In Section 4 we explain how this approach works, and in 5 we show some visual results.

4 Sliced Optimal Transport Method

In this section we propose a method inspired on Risser's, but using sliced optimal transport. As it is unclear where in the method Risser uses optimal transport, we decided to try ourselves a different version of his method where we propose using directly optimal transport to perform histogram matching.

For texture generation, our method starts equivalent as 3.1, by encoding the images into a feature space of a VGG, until the part where we match the activations of each layer between the generated image $\hat{F}^l$ and the original image ^{l}. Instead of performing random rotations and histogram matching, we suggest using sliced optimal transport as introduced in Section 2.3 to do histogram matching.

In this case, the supports $(x_i)_i$, $(y_i)_i$ of the distribution are given by each point $F^l_{\cdot k}$ and $\hat{F}^l_{\cdot k}$ induced by k, living in $\mathbb{R}^{N_l}$, for image x and $\hat{x}$ respectively. The weight is given by the mean number of times each point appears. The number of slices is chosen to be N_l/n for layer l, the number of feature maps in layer l over the number of iterations of the method.

Finally, after all iterations the features are decoded by the inverse VGG, and the output is the generated texture.

5 Numerical Experiments

5.1 Methodology

For the numerical experiments presented below, we proceed as follow. First, we consider the separation of textures in five classes, as in [8]. These are: irregular, regular, near regular, near stochastic, and stochastic textures. Figure 2, taken from [8], exhibits the visual aspect of each such class. We then sample five textures from each of these classes excluding the stochastic class, as it was not easy to find samples from it.

We start in 5.2.1 and 5.2.2 using the algorithm discussed in Section 3 to perform texture synthesis and mixing. Following, in 5.2.3 we analyze the pertinence of performing random rotations before histogram matching as suggested by the main method described in Section 3. We compare samples generated with and without rotation visually, and quantitatively by analyzing the error between their Gram matrix and the original texture Gram matrix. Finally, in 5.2.4 we use the sliced optimal transport algorithm introduced in Section 4 to perform texture synthesis. Code for both the Gram matrices analysis and the sliced optimal transport synthesis is provided.

5.2 Results

5.2.1 Texture synthesis

The results for texture synthesis with Risser's method (Section 3) are shown in Figures 3 and 4. We emphasize the absence of quantitative metrics in this field, and how the evaluation of results is usually based on the visual perception of similarity between the style texture and the generated one.

In our case, the results seem to be mostly correct and overall satisfying with the exception of the regular textures class. The structure and symmetries of such textures are not respected on the generated samples. This happens because the matching histogram procedure in feature space does not seem take these constraints into account. We will discuss a possible correction to this problem in Section 6.

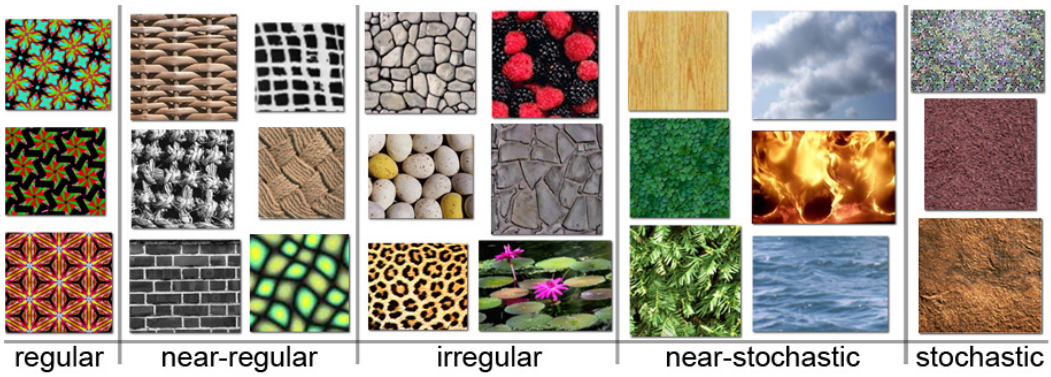


Figure 2: The "texture spectrum" we used to classify our experiments

In the class of irregular textures (Figure 3), we observe that individuality is overall respected: the generated strawberries, skin patches, lily pads, pebbles and mosaic pieces are visually a collection of distinct elements. As we saw in class, this is a problem with some classical approaches. However, some imperfections can be perceived. Most notably some of these generated objects present an unnatural curvature or torsion effect, giving them a visual aspect of being "melting". The border between some distinct objects can also seem blurry or ill-defined, contributing to this effect.

Textures in the near stochastic class are arguably those giving the most satisfying results. The only defects we notice are locally darker patches in the synthesized images for wood and ocean water. For near regular textures, as before, the greater the symmetry present in the style image, the more we notice these being broken in the generated images. Most noticeably, the parallelism of the brick wall is not present in our resulting texture, and the resulting puzzle pieces are merged.

5.2.2 Style transfer

For our experiments on style transfer, we used the images in Figure 7 as contents and styles. The results for style transfer between these are shown in Figure 8. We denote by α the parameter measuring content strength. We used $\alpha = 0.5$ for the styles of more structured textures, i.e., for the transfer of the cybernetic and brick wall styles; we used $\alpha = 0.07$ for the artistic styles and the less structured textures, i.e., for the Van Gogh, J.M. Basquiat and Mosaic styles. This was decided on a visual-quality basis. We also studied the effect of varying the α parameter, as is seen in Figure 9.

5.2.3 Rotation vs. no rotation

Figure 5 shows a visual comparison between a texture synthesis and a style transfer examples done using the method presented on 3 with the random rotations before histogram matching, and without these random rotations. Visually there is no major difference between the images using the rotations and the ones not. Furthermore, table 1 shows the mean square error between the Gram matrices computed at each layer for the original sample and a generated sample. Table 1 shows results for the graffiti image generated with and without random rotations before histogram matching.

5.2.4 Histogram matching with sliced optimal transport

In Figure 6 we show visual results of our suggestion of using sliced optimal transport to perform the histogram matching. We see that the algorithm manages to transform the random noise image into something that follows the same visual characteristics of the original texture. However, we can still see a lot of noise and imprecision's in the generated samples. Also, our code is slower that the one in [11]. We still believe this is an interesting approach that could be further explored.

6 Discussion

The main method of this report has its merits, especially regarding near-stochastic/irregular textures. However, it does not take many experiments to observe that the method does not work at all with any

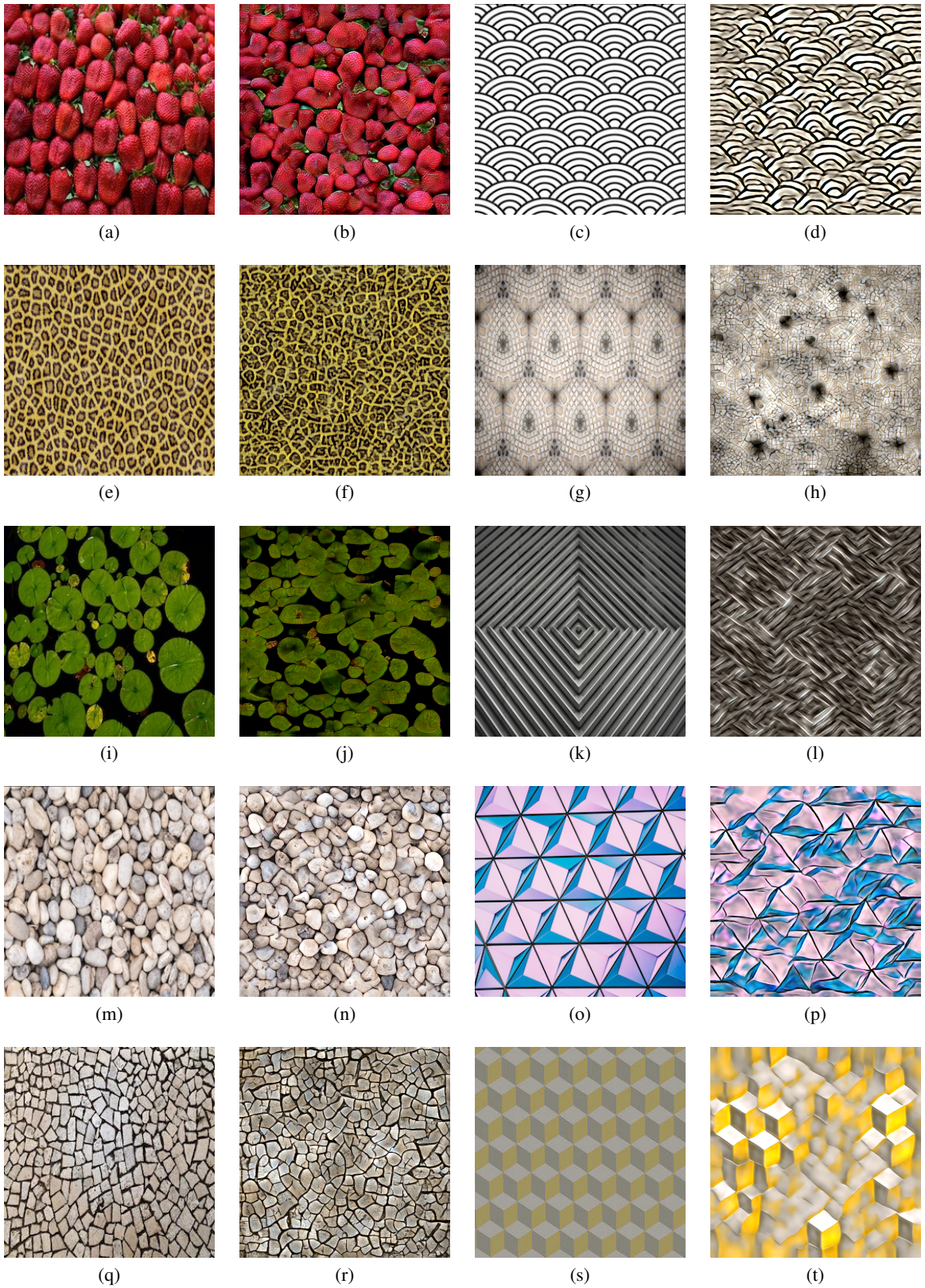


Figure 3: Texture synthesis on irregular and regular textures

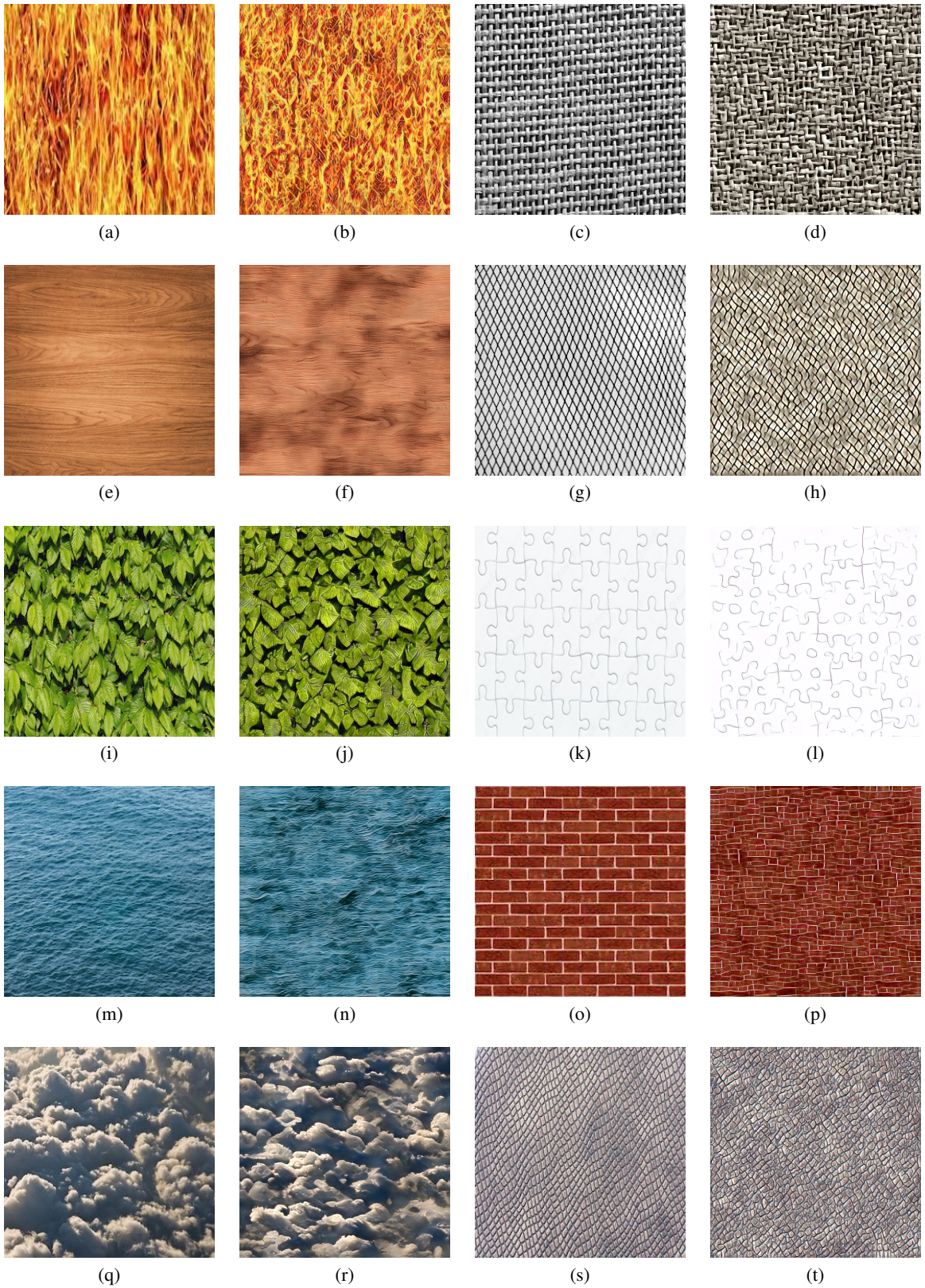
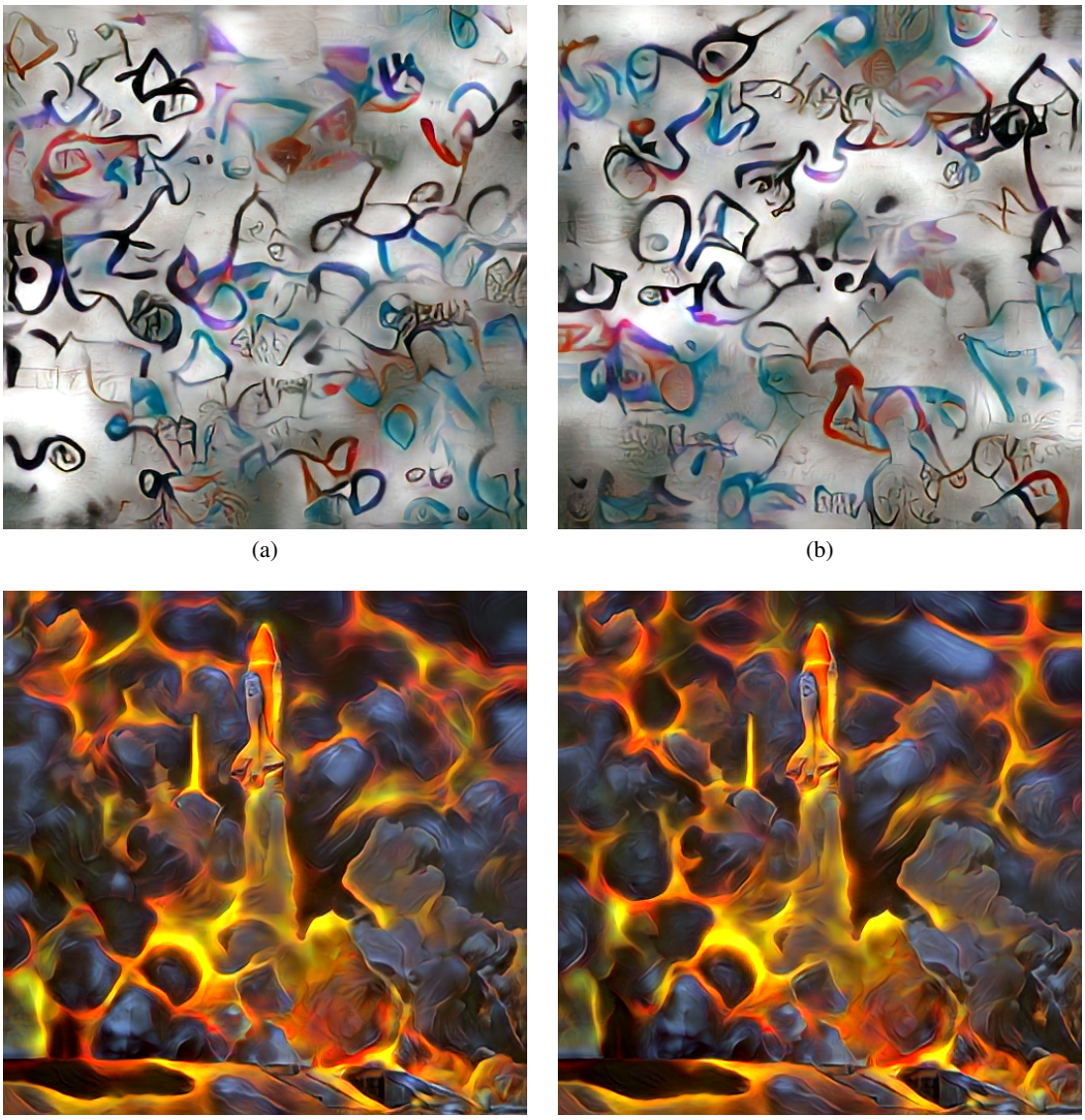


Figure 4: Texture synthesis on near stochastic and near regular textures

Random rotation	No rotation



(a) (b)

(c) (d)

Figure 5: Style transfer with random rotations and without

Layer	Error sample w/ rotations	Error sample without rotations
1	33385.9	32670.5
2	3496.5	3363.6
3	3477.1	3214.5
4	975.1	902.93
5	2588.4	2428.9

Table 1: Mean square error between original and generated image Gram matrices for each layer of the VGG for two samples: one generated with and one without the random rotations before histogram matching. Values correspond to the graffiti image of Figure 5

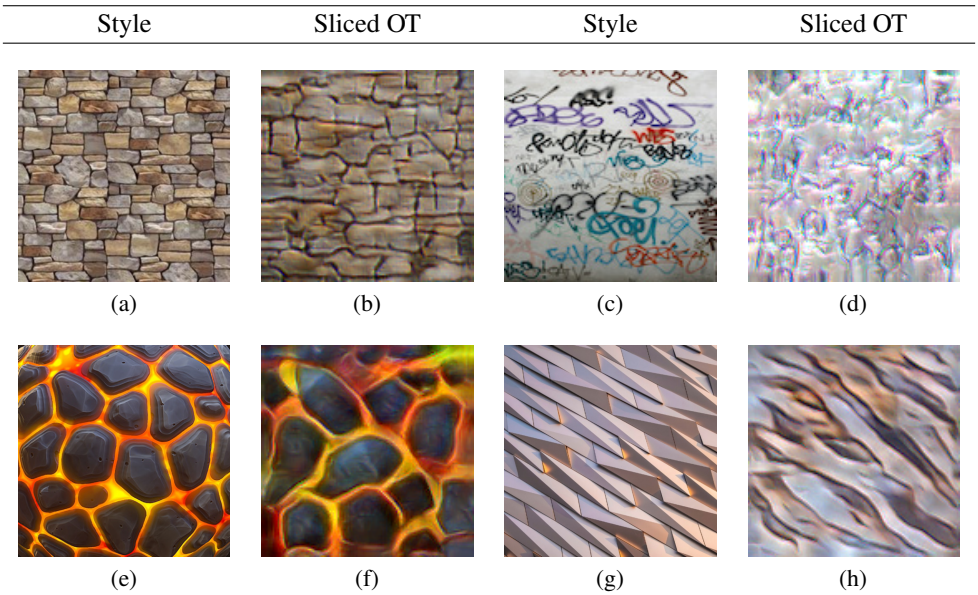


Figure 6: Texture generation with sliced optimal transport

structured texture or pattern. This is completely omitted in [11], which is concerning given that a large class of textures consist of symmetric patterns. We could also notice that the lack of precise metrics of quality in this field creates empty accusations between authors and imprecise notions of improvement of a method with respect to others. For example, [7] criticized the idea of performing histogram matching based on visual aspects; meanwhile, [11] defends it and attacks the use of the whitening and coloring transform by the same arguments.

We now outline a possible line of further research in this subject. We aim at including the constraints of symmetry and structure of textures in the method. There are two steps for achieving this. First, given a texture one must be able to determine what are its symmetries, i.e., to what transformations the texture under study is invariant to. This general theme has been only recently explored in papers such as [1], [6] and [4]. Once you identify the proper symmetry present in the texture, you must use neural networks that are properly designed for preserving them. These are known as "equivariant neural networks" [1]. We believe that performing sliced OT for matching histograms in the feature space of these networks (associated with the correct symmetry group) should be able to give results taking these constraints into account.

7 Conclusion

In this report, we briefly presented a method of texture synthesis and style transfer using histogram matching on the feature maps of a neural network. We explained the related work that motivated the

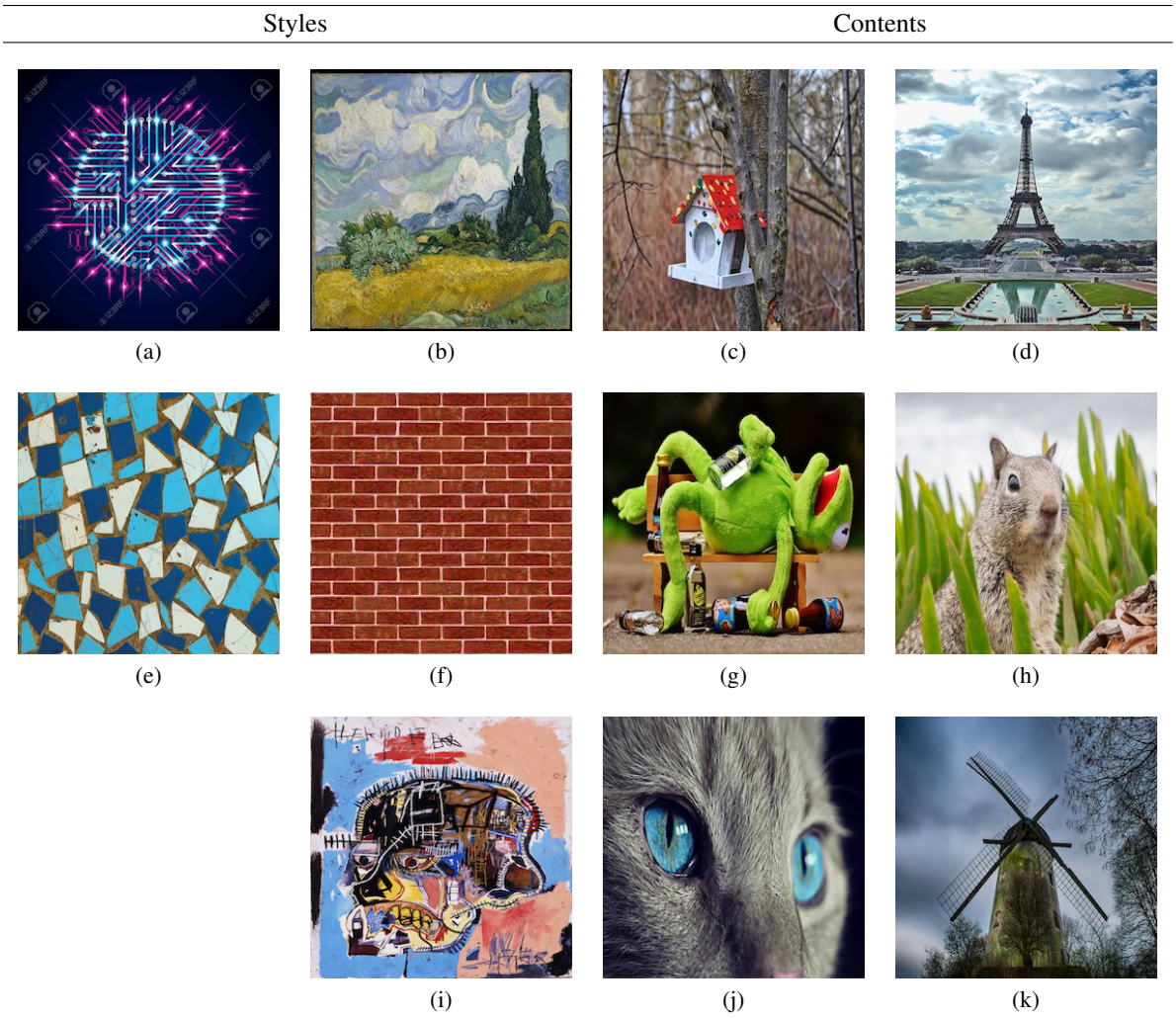


Figure 7: Styles and contents for the experiment on style transfer

creation of the method, and the theory of the main algorithm. We proposed a different approach using sliced optimal transport as it is unclear if the main article really uses optimal transport theory in the method, despite its title.

We performed numerical experiments on texture synthesis and style transfer to verify to what extent the main method works, and found visually satisfying results in cases where not much structure or symmetry is present on the content image. We identified the presence of rigid structures and symmetries as being a weakness to the method, and laid out an idea for future research that could go in the direction of solving these issues. We contested some of the authors choice, such as the random rotations performed before histogram matching, and by quantitative and qualitative analyses we concluded the random rotations do not seem to change the final output. Furthermore, we explored our proposed method with sliced optimal transport, and realized it does succeed on generating images that imitate the original texture, although with more noise. Finally, we also discussed how the lack of quantitative metrics creates difficulties in communication and introduces a considerable degree of subjectiveness in the evaluation and comparison of different algorithms.

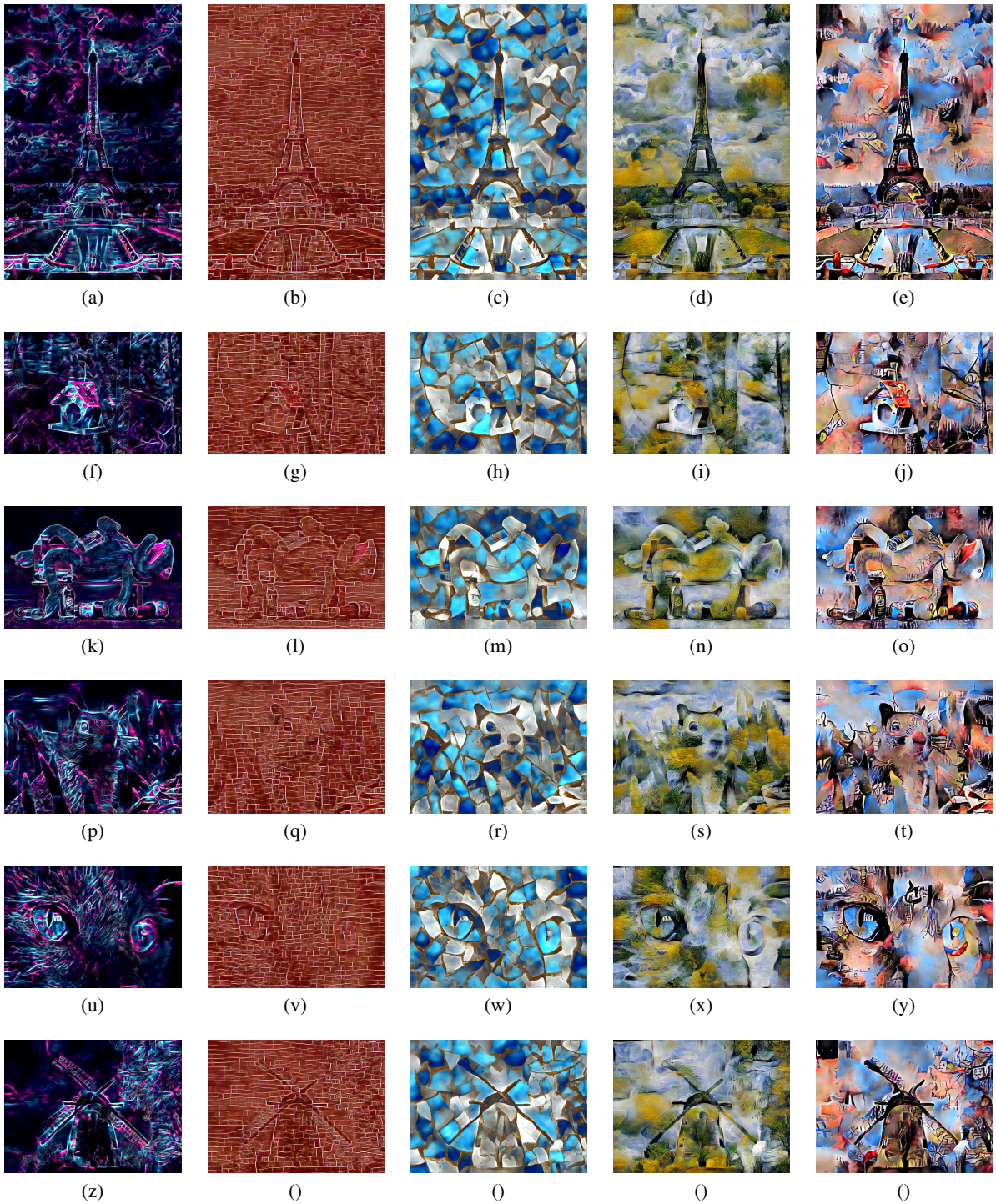
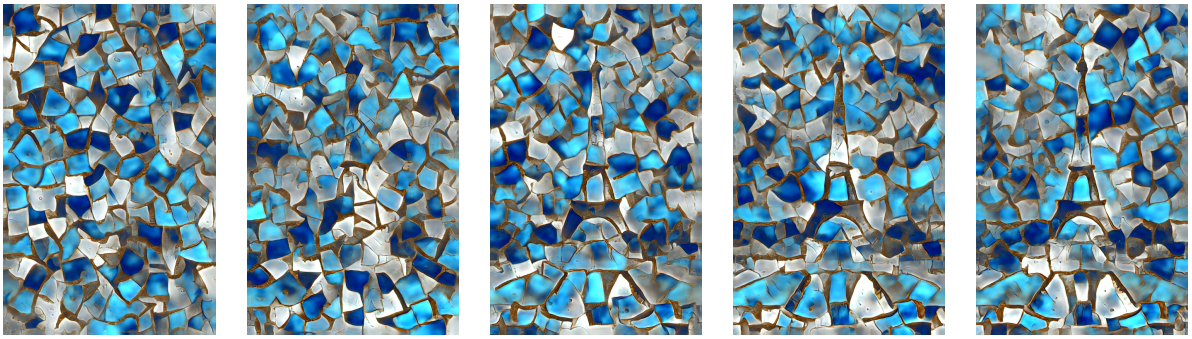
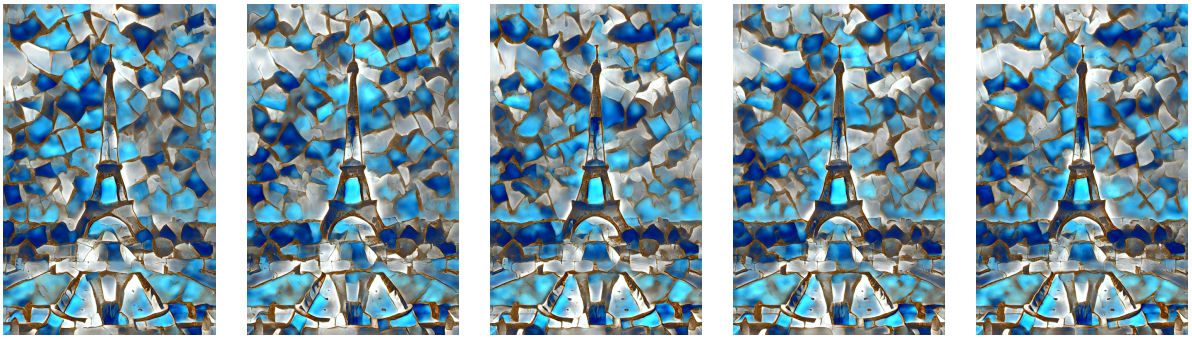


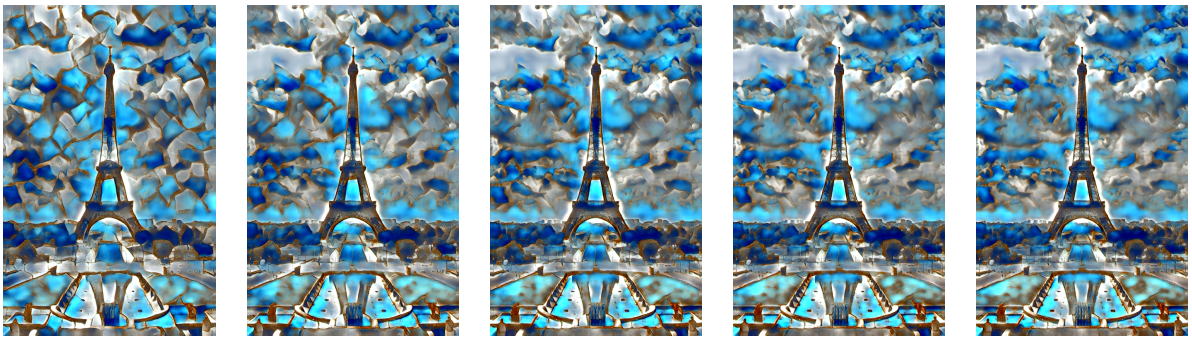
Figure 8: Style transfer for texture and artistic styles



(a) $\alpha = 0.00$ (b) $\alpha = 0.01$ (c) $\alpha = 0.02$ (d) $\alpha = 0.03$ (e) $\alpha = 0.04$



(f) $\alpha = 0.05$ (g) $\alpha = 0.06$ (h) $\alpha = 0.07$ (i) $\alpha = 0.08$ (j) $\alpha = 0.09$



(k) $\alpha = 0.1$ (l) $\alpha = 0.2$ (m) $\alpha = 0.3$ (n) $\alpha = 0.4$ (o) $\alpha = 0.5$

Figure 9: Varying the parameter α regulating content strength

References

[1] Taco S. Cohen and Max Welling. *Group Equivariant Convolutional Networks*. 2016. arXiv: 1602.07576 [cs.LG].

[2] Leon Gatys, Alexander S Ecker, and Matthias Bethge. "Texture Synthesis Using Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/a5e00132373a7031000fd987a3c9f87b-Paper.pdf.

[3] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "Image Style Transfer Using Convolutional Neural Networks". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2414–2423. DOI: 10.1109/CVPR.2016.265.

[4] Jonathan K. George et al. "Symmetry perception with spiking neural networks". In: *Scientific Reports* 11.1 (Mar. 2021). DOI: 10.1038/s41598-021-85232-3. URL: https://doi.org/10.1038/s41598-021-85232-3.

[5] Rabin Julien et al. "Wasserstein Barycenter and its Application to Texture Mixing". In: *SSVM'11*. Israel: Springer, 2011, pp. 435–446. URL: https://hal.archives-ouvertes.fr/hal-00476064.

[6] Sven Krippendorf and Marc Syvaeri. *Detecting Symmetries with Neural Networks*. 2020. arXiv: 2003.13679 [physics.comp-ph].

[7] Yijun Li et al. "Universal Style Transfer via Feature Transforms". In: *Advances in Neural Information Processing Systems*. 2017.

[8] Wen-Chieh Lin et al. "Quantitative Evaluation of Near Regular Texture Synthesis Algorithms". In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 1. 2006, pp. 427–434. DOI: 10.1109/CVPR.2006.233.

[9] *Optimal textures*. https://github.com/JCBrouwer/OptimalTextures.

[10] François Pitié, Anil C. Kokaram, and Rozenn Dahyot. "Automated colour grading using colour distribution transfer". In: *Computer Vision and Image Understanding* 107.1 (2007). Special issue on color image processing, pp. 123–137. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2006.11.011. URL: https://www.sciencedirect.com/science/article/pii/S1077314206002189.

[11] Eric Risser. "Optimal Textures: Fast and Robust Texture Synthesis and Style Transfer through Optimal Transport". In: *CoRR* abs/2010.14702 (2020). arXiv: 2010.14702. URL: https://arxiv.org/abs/2010.14702.