Texture Synthesis

Submitted by:

Aayush Goyal (2019CS10452)

Under the supervision of

Prof. Prem Kalra

COD 310, Mini Project

Indian Institute of Technology, Delhi

Abstract

Texture Synthesis is a topic of study in computer graphics and is used in a variety of fields such as fashion, digital image editing, and medicine. We want to expand its application in the medical industry. Our main goal is to generate larger instances that perceptually resemble the smaller input texture exemplar given some small texture of parts of our body. The main idea is to use a patch-based method for generating visual appearance i.e. new image is synthesised by intelligently stitching together small patches of an existing image.

Contents

Abstract **2**

CHAPTER 1: Introduction **1**

1.1 What is Texture Synthesis? 1

1.2 Problem Formulation 2

CHAPTER 2: Proposed Method **3**

2.1 About Patch-Based Method 3

2.2 Using Neighboring Block constrained by Overlap 4

2.3 Minimum Error Boundary Cut method 5

CHAPTER 3: Results and Discussion **6**

3.1 Results 6

3.2 Discussion 8

CHAPTER 4: Scope of Future Work **9**

References **10**

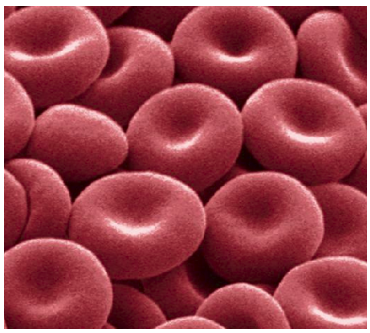
Chapter 1

Introduction

1.1 What is Texture Synthesis?

It is a method for producing arbitrarily large textures from small real-world samples. The idea is that we take an input image of a texture and then look for other images that look similar to that texture. So the assumption here is that somewhere out there, in the real world there exists larger texture and the input image is just a small sample and we want to get just some more samples of that texture. One of the assumptions here is that the input image is large enough to capture the essence of this texture.

The main challenge for texture synthesis is that there are many different textures, such as some that are purely **repeated** (look like somewhat regular patterns), some that are **stochastic** (texture images of stochastic textures look like noise), and most that fall somewhere in between the **mix** of repeated and stochastic.



(a) Repeated (RBCs)

(b) Stochastic (BVs)

(c) Both (Brain)

1.2 Problem Formulation

We want to expand its application in the medical industry. When we provide the algorithm a small sample of the texture of a region of our body, the primary objective is to produce larger examples that visually resemble the smaller input texture exemplar. The user will provide input in the form of an image, and they will be able to select the size of the texture that they wish to generate. The final product will be a newly generated texture that looks very much like the image that was input. The primary concept behind this is the utilisation of a patch-based method for the generation of visual appearance. This means that a new image is created by cleverly piecing together small patches of an already existing image.

Chapter 2

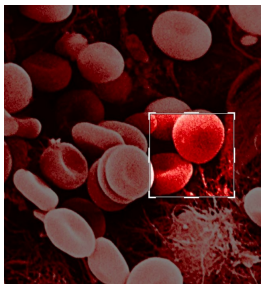
Proposed Method

2.1 About Patch-Based Method

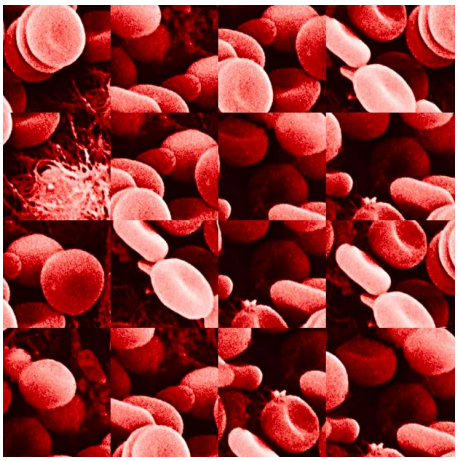
We'll look at a patch-based technique also known as **Image Quilting**. First, we will define the synthesis unit **Bi**, which is a square block whose size is defined by the user. We've assigned set **S** to all of these square blocks, which are overlapping blocks from input image.

The most basic and naive approach is **Random Block Placement**, in which we randomly select patches from the input texture and place them on a larger target image. This can be accomplished by looping from left to right, top to bottom (rendering order), and placing randomly sampled blocks from **S** next to each other.

(a) Input image



(b) Choosing a Patch

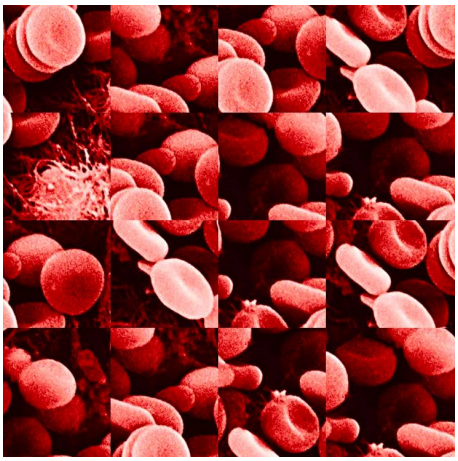


(c) Output by Random Placement

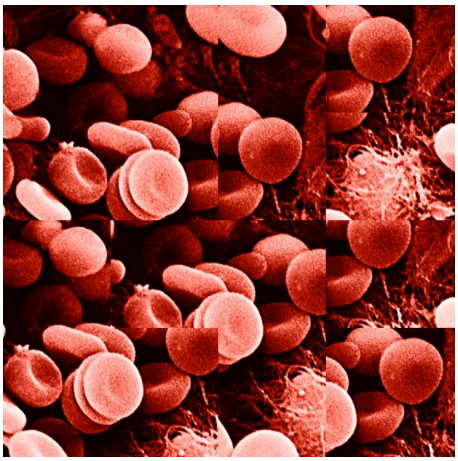
We can see that the blocks do not match, and the output image is not satisfying.

2.2 Using Neighboring Block constrained by Overlap

As we can see in the image output created by selecting random patches from the input texture, the blocks do not match. The output image has visible edges, as we can see. So we must overcome this. We'll accomplish this by using **Neighboring Block constrained by Overlap**, which will cause some overlap in the placement of blocks on the new image. The top left patch is chosen at random, and the next patch from **S** is chosen to minimise the overlap error **L2 norm**. The new block is partially overlaid on the previous block, starting from the centre.



(a) Ouput image formed by placing blocks Randomly

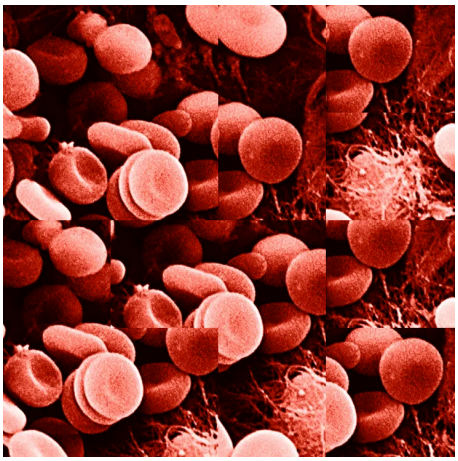


(b) Ouput formed by placing blocks based on Neighboring Block constrained by Overlap

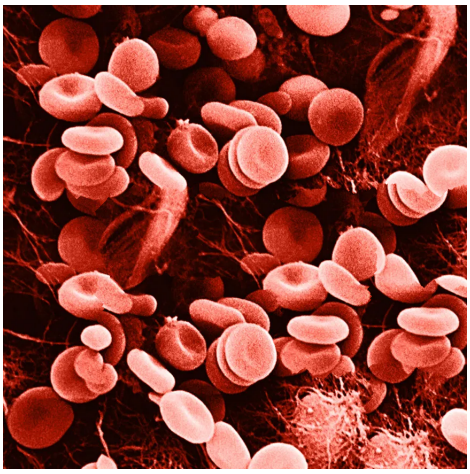
The above image (b) shows an improvement in the target image's structure; the edges have been smoothed compared to (a), but the edges between the blocks are visible.

2.3 Minimum Error Boundary Cut method

Instead of a straight line between the two patches, we can use the Minimum Error Boundary Cut method to remove these visible edges. Before inserting a selected block into the target image, we inspect the overlap region between it and the other blocks for errors. We find the cheapest path through the error surface and declare it the boundary of the new block.



(a) Ouput formed by placing blocks based on neighboring block constrained by overlap



(b) Ouput formed by placing blocks with minimum error boundary cut

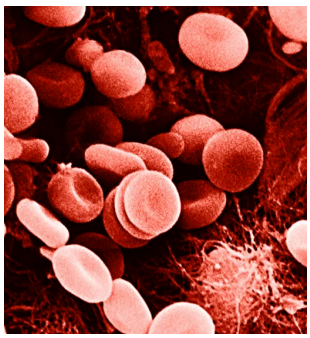
Above image (b) shows that the generated output is satisfying and edges are also smooth.

Chapter 3

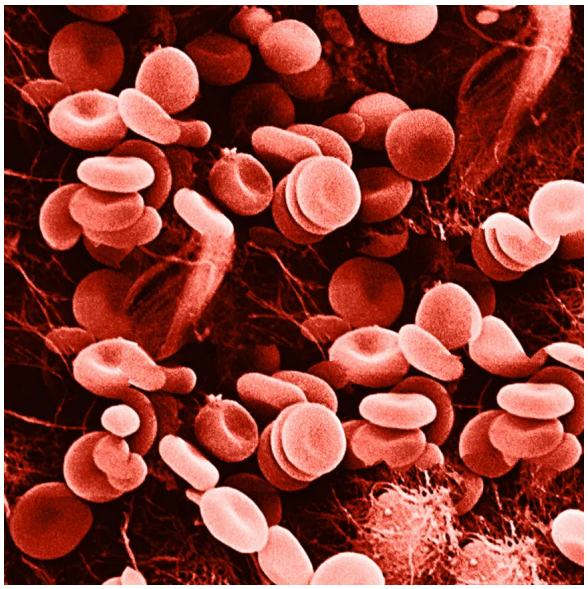
Results and Discussion

3.1 Results

We have introduced image quilting, a method of syn- thesizing a new image by stitching together small patches of existing images. Despite its simplicity, this method works remarkably well when applied to texture synthesis, producing highly stable results. For wide variety of input textures, the result of texture synthesis can be seen below.

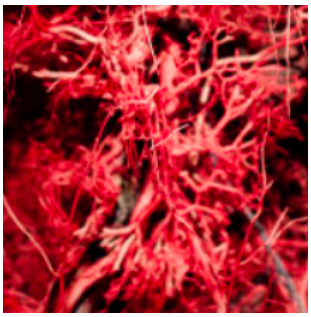


(a) Input image

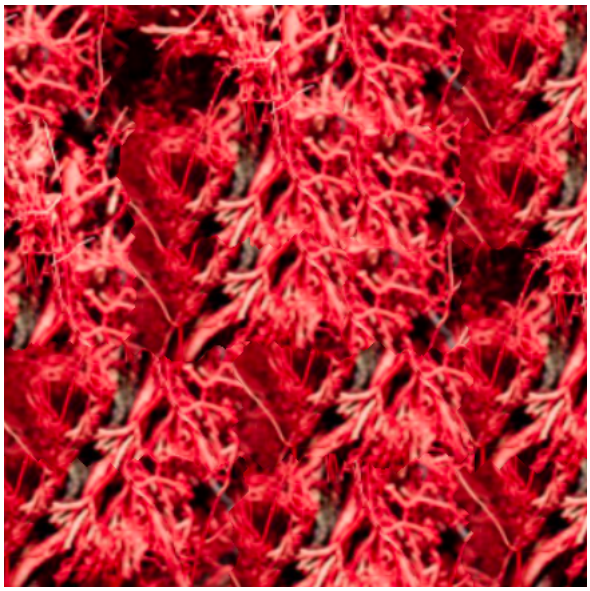


(b) Output Image

Figure 3.1: Input is an image of human Red Blood Cells. The size of input is 516 x 557 pixels. A block size of 200 will be able to cover the components like individual cell and nerves. The size of output image is 868 x 868 pixels.



(a) Input image



(b) Output Image

Figure 3.2: Input is an image of human Blood vessels. Size of input is 254 x 256 pixels. Block size of 100 will be appropriate. The size of output image is 436 x 436 pixels.

(a) Input image

(b) Output Image

Figure 3.3: Input is an image of human Brain. Size of input is 287 x 231 pixels. Block size of 210 will be appropriate, taking it less than that will produce noise in the output. The size of output image is 560 x 560 pixels.

3.2 Discussion

1. The algorithm performs well on stochastic and very regular textures, but user-specified parameters such as block size, number of blocks, and input texture type all had a significant impact on the quality of the results. One of the big assumptions here is that the input image is large enough to capture the essence of this texture. Then we should cleverly choose the block size. The block size should be chosen in such a manner that it is able to capture the different components of the texture. Like in case we have an input image of Red blood cells, an appropriate block size would be something that covers one cell roughly. It should not be too small or large.

2. The time taken to generate the output highly depends on the size of the input image and the block size (Bi) chosen by us. The larger the input size and the lesser the block size, more time is required to generate the output. Further it also increases proportionally with the number of blocks. The more the number of blocks, the bigger the image will be and the more time it will take to generate the output. Let's say the block size is B pixels, number of blocks is N, and the overlap region is V pixels. Then the size of the output is: $NB-(N-1)V$. Currently the value of overlap size is set to $\frac{1}{6}^{th}$ of the Block size.

Chapter 4

Scope of Future Work

One of the major limitations of this algorithm, as discussed above, is that it requires the input image to be large enough to cover the essence of the larger texture that we want to generate. Also, the output image is created by stitching together patches from the input image itself. As a result, there is no possibility of new features being introduced into the larger output image. Deep learning-based methods can be used to overcome this. Not only that, they are effective in limiting the output to specific boundaries. Let's say we have a texture for some part of the brain and want to expand it to the entire sketch of the brain, then it is possible to do this using models trained using these algorithms. This research was published in the paper [2, TextureGAN: Controlling Deep Image Synthesis with Texture Patches]. They have provided examples based on clothing and footwear. Essentially, they enter a small texture as the design for the shoe or cloth, followed by the sketch of the shoe or cloth. The model generates an output in which the texture has been expanded to cover the entire object. However, due to a lack of medical data, we do not currently have enough data to train the model for medical applications. If we can obtain the medical data, we can use this model to generate output for medical applications.

Bibliography

[1] Alexei A. Efros, William T. Freeman. Image Quilting for Texture Synthesis and Transfer. University of California, Berkeley. Mitsubishi Electric Research Laboratories.

[2] Wenqi Xian, Patsorn Sangkloy, Varun Agrawal, Amit Raj, Jingwan Lu, Chen Fang, Fisher Yu, James Hays. TextureGAN: Controlling Deep Image Synthesis with Texture Patches. Georgia Institute of Technology, Adobe Research, UC Berkeley.

[3] Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Dani Lischinski, Hui Huang. Non-Stationary Texture Synthesis by Adversarial Expansion