Texture Synthesis by Non-Parametric Sampling

Spencer McMurray & Mohan Vashist

1 Abstract

This paper outlines the techniques of texture synthesis by non parametric sampling. Textures synthesis process begins at an three by three seed and expands outward from there, iterating through all unfilled pixels which neighbour a filled pixel, determining the best match along the way. This results in outputs which strive to maintain local structure while still being general enough to provide accurate results for a wide array of textures.

2 Introduction

Texture synthesis is the act of expanding or filling in an image to extend what it looks like seamlessly. Texture synthesis proposes many challenges including from determining the nature and quality of which defines a texture, to figuring out how to classify the image.

Texture synthesis has many application which include but are not limited to foreground removal, occlusion fill-in, lossy image and video compression (Efros and Leung,1999). The goal of our approach is being able to synthesize a wide array of different types of textures, while being able to preserve the local structure. With this in mind we choose to follow Efros and Leung approach of texture synthesis by non parametric sampling since it is able to accomplish our goals with a high degree of success. Nevertheless, images with lots of detail won't produce great results, as the image is expanded based on what the rest of the image looks like. Knowing this, images with repeating or similar elements are prime candidates for texture synthesis.

3 Methodology, Results, and Experiments

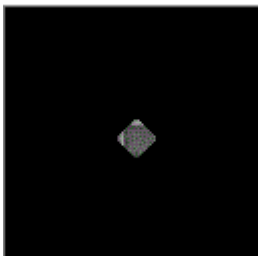
To begin with, non parametric sampling refers to a type of statistics which does not require the population to be analyzed or meet certain assumptions or parameters. Instead what we can do is make some assumptions about our data and use statistical methods such as the ANOVA, or t-test to provide importation about the data being analyzed (Kenton,2020). For our approach we assume a Markov Random Field, which is the probability density function of intensity for a pixel given the intensity values of its spatial neighborhood is independent of the rest of the image. Given this, instead of searching for a model which represents the conditional probability distribution of the current pixel under observation, given the neighbourhood window, we search through the given texture for neighbourhoods to produce a histogram for the pixel under consideration. However, we are faced with an issue that an exact neighbourhood may not be present in the source image for the synthesized pixel in question. Thus, we use

find the best match for the synthesized pixel by taking all samples within the distance of the Gaussian weighted SSD match.

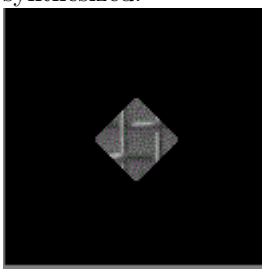
Algorithm Explanation:

The algorithm takes in a base texture image, the shape of the new image, and a window size. First, we seed a new image of the given size with a random 3x3 section of the base texture. Then until each pixel in the new image is filled, we will go through all of the unfilled pixels that neighbour a filled pixel and determine it's best match in the base texture by calculating the Gaussian-Weighted SSD of a window around the pixel. If the match's error is within a threshold, it will be filled in otherwise the threshold will be increased. This allows us to control the degree of randomness. Once this loop is completed, the synthesized image is complete and returned.

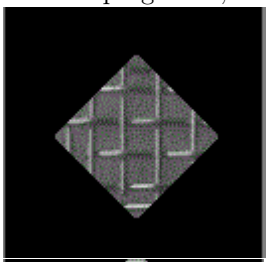
We can see the results as time progresses:

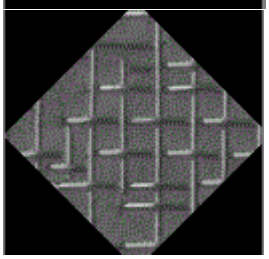


In the above image the texture synthesis process has just begun, the seed is present and few pixels have been synthesized.



As time progresses, more pixels are synthesized.

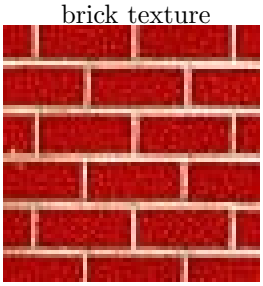




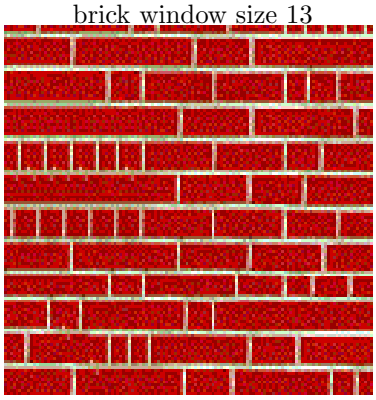
The result of 60% synthesis

Results:

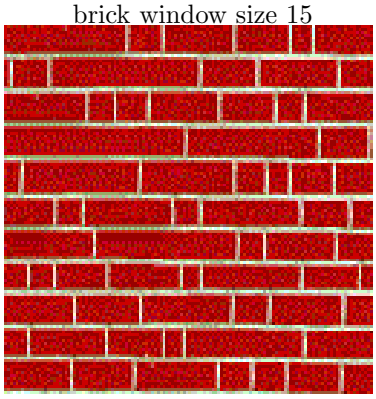
brick texture



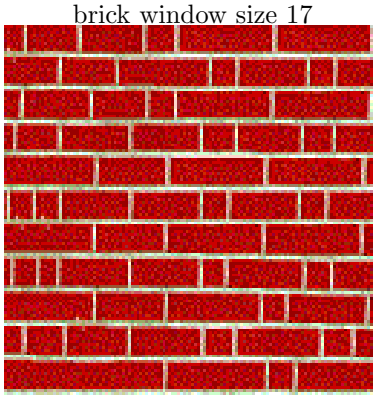
brick window size 13



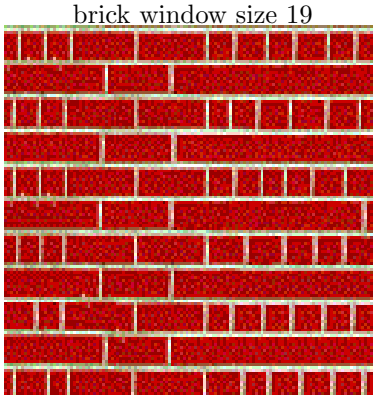
brick window size 15



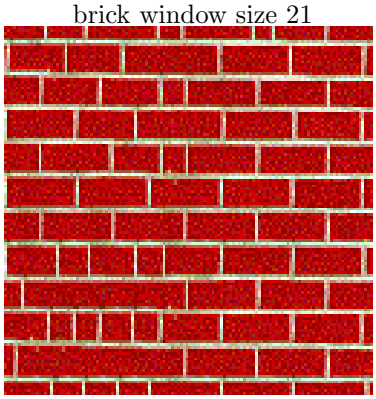
brick window size 17



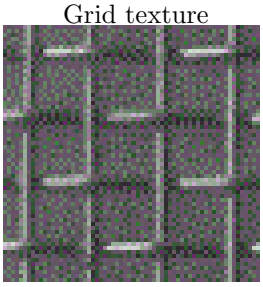
brick window size 19



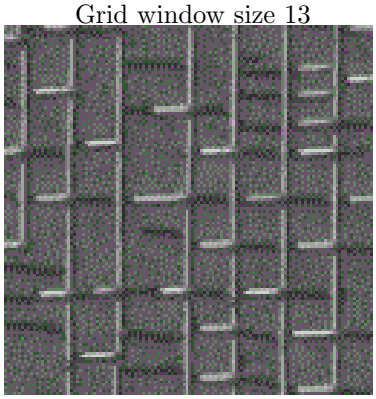
brick window size 21



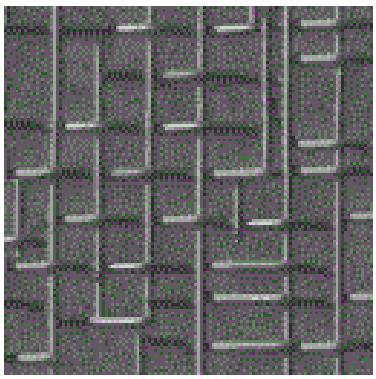
Grid texture



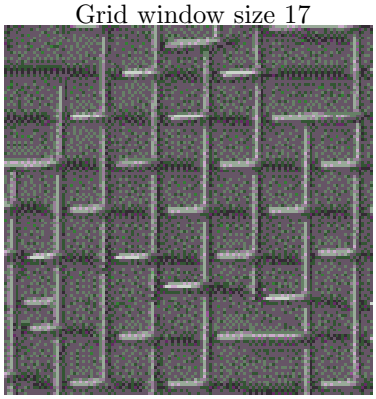
Grid window size 13



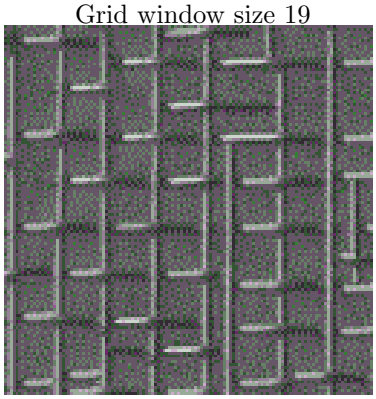
Grid window size 15



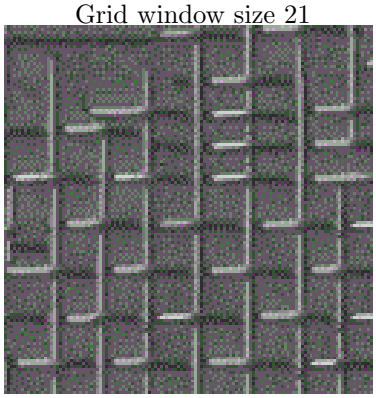
Grid window size 17



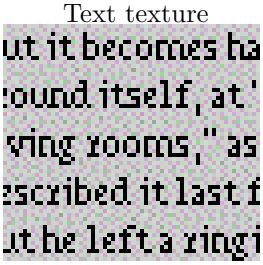
Grid window size 19



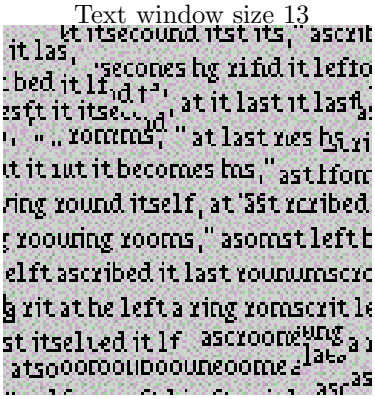
Grid window size 21



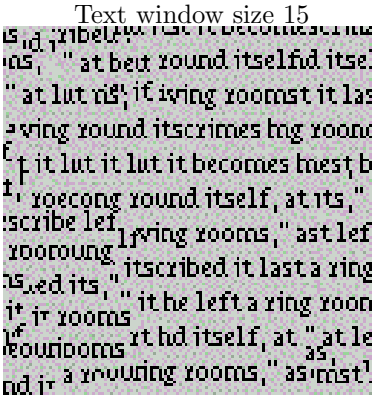
Text texture



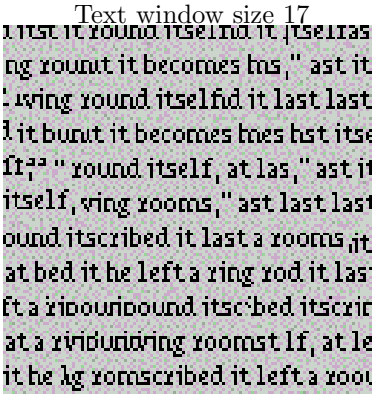
Text window size 13



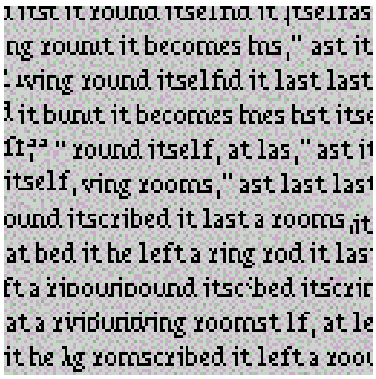
Text window size 15



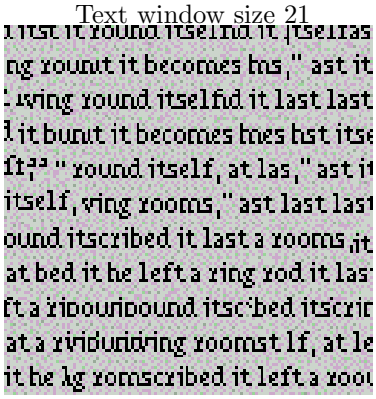
Text window size 17



Text window size 19



Text window size 21



The algorithm produces accurate results for a wide array of textures given that the window size is relatively large. In all cases a window size of 19 or greater is ideal since with a larger window size you are calculating the SSD of a larger area, thus, resulting in a better match for the synthesis of the pixel. Also, with the larger window sizes you can see that the image is more clear with a better synthesized result. However, this does not mean that we would recommend increasing the window size too much since it impacts performance.

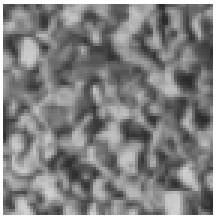
Alternative approaches:

An approach proposed by Zhu, Wu and Mumford describes using Gibbs samplings for synthesis, However, Gibbs samplings is slow to converge and can be very difficult to determine when it has converged.

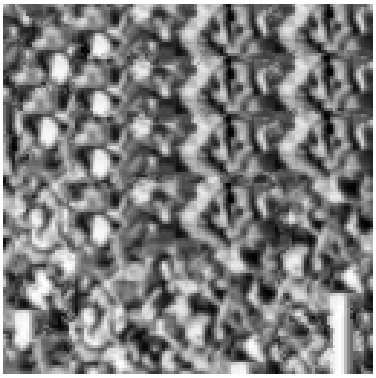
Another approach proposed by J.De Bonet from Massachusetts Institute of Technology aims for "treating input texture images as probability density estimators from which new textures, with similar appearance and structural properties, can be sampled." J.De Bonet's approach is a two phase approach, the first phase "the input texture is first analyzed by measuring the joint occurrence of texture discrimination features at multiple resolutions". For the second phase "a new texture is synthesized by sampling successive spatial frequency bands from the input texture, conditioned on the similar joint occurrence of features at lower spatial frequencies." This approach is more successful than the first approach and is able to synthesises a wide array of different textures. However, with this approach the input textures is replicated first to fill the desired dimensions before synthesis occurs, which assumes that all textures are tillable which is not correct for all textures.

Poor results:

For the given texture:



the output:



The synthesis for this example is poor. The output has large portions which are repeated and do not blend well with the rest of the image. Also, the image has multiple different intensities. This is because our algorithm continues to search the wrong space and continues growing from the poor position.

Future Improvements:

We are looking at ways to improve the speed of the algorithm. We could achieve this in many different ways. One way could be to use multiprocessing. However, due to the nature of Python being a high level language as well as the restrictions of a users machine this would be difficult to implement, and perhaps even be worse for performance since you would have critical pieces of code waiting for non crictical pieces of code to finish.

4 Conclusion

In conclusion, the results of our implementation of texture synthesis by non parametric Sampling adapted from Efros and Leung were overall positive, however there are limitations to our approach. Nevertheless, the applications of texture synthesis are vast, from foreground removal, occlusion fill-in, lossy image and video compression. Thus, texture synthesis stands to be a vital field in computer vision.

5 Authors' Contributions

Spencer McMurray: Algorithm Implementation and Project Report (Worked on the Introduction and Methodology, Results, and Experiments section).

Mohan Vashist: Video script and code quality improvements and Project Report (Worked on Abstract, Introduction and Methodology, Results, and Experiments sections)

Please Note that you can see a full commit history on our Github: *github.com/SpencerMcMurray/Texture-Synthesis*

6 References

Efros, A., amp; Leung, T. (1999). Texture synthesis by non-parametric sampling. Proceedings of the Seventh IEEE International Conference on Computer Vision. doi:10.1109/iccv.1999.790383}

Kenton, W. (2020, June 06). Nonparametric Method. Retrieved August 02, 2020, from https://www.investopedia.com/terms/n/nonparametric-method.asp

J. S. D. Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In SIGGRAPH '97, pages 361–368, 1997.

S. C. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (frame). International Journal of Computer Vision, 27(2):1–20, March/April 1998.