Name: Emerson Raja Jose Peeterson

Matric no. : A0195021N/E0383686

Module : EE4212, part 2, Assignment 1

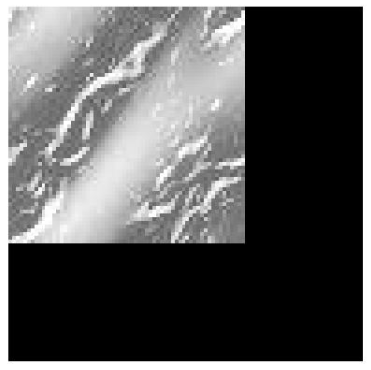
**Assignment 1: Non-parametric sampling**

**Part 1) : Basic Synthesis algorithm**

Result of each texture sample

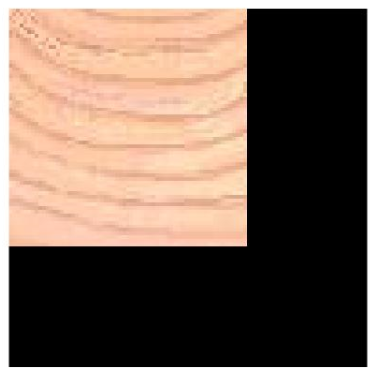
1. neighbour size = 21

Original texture1 : 64x64 Enlarged 1.5x texture1: 96 x 96

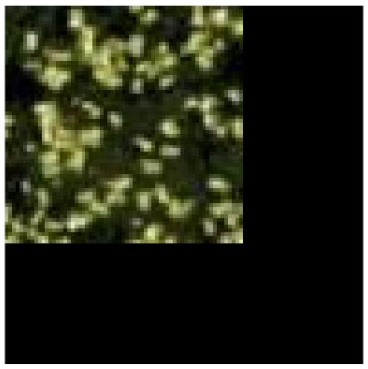
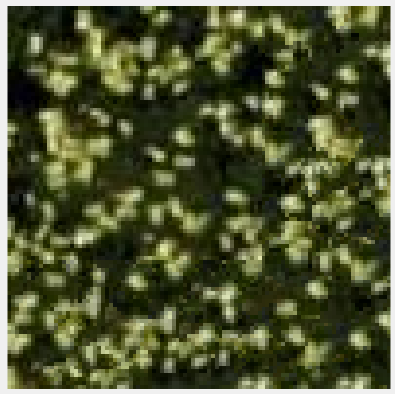
1. neighbour size = 9

Original texture2 : 65x65 Enlarged 1.5x texture2: 98 x 98

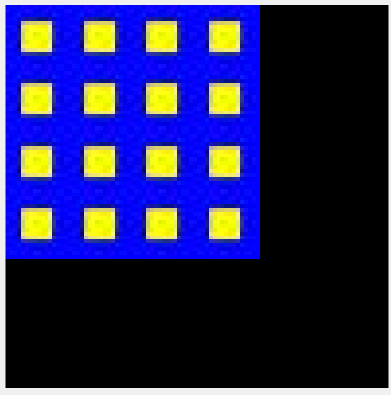
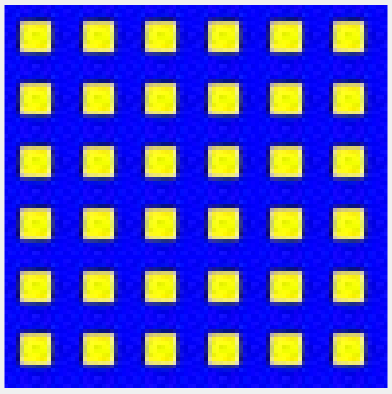
1. neighbour size = 17

Original texture3 : 65x65 Enlarged 1.5x texture3: 98 x 98

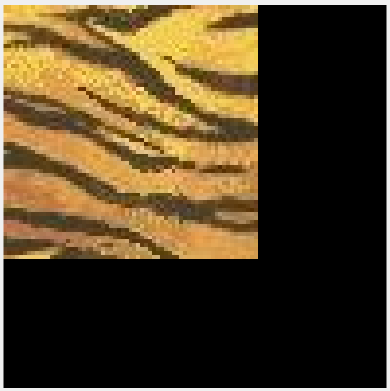
1. neighbour size = 31

Original texture4 : 65x65 Enlarged 1.5x texture4: 98 x 98

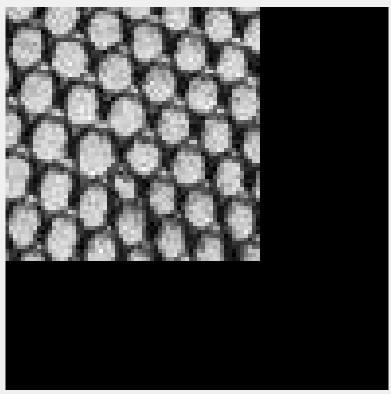
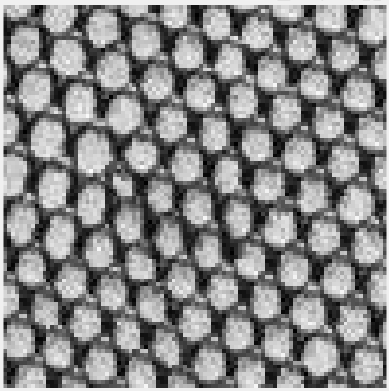
1. neighbour size = 21

Original texture5 : 65x65 Enlarged 1.5x texture5: 98 x 98

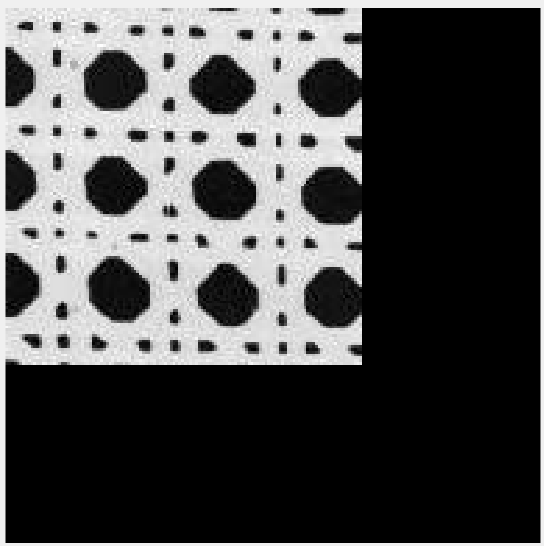
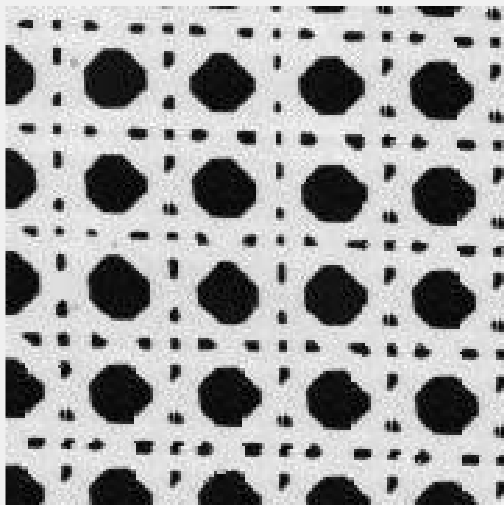
1. neighbour size = 27

Original texture6 : 65x65 Enlarged 1.5x texture6: 98 x 98

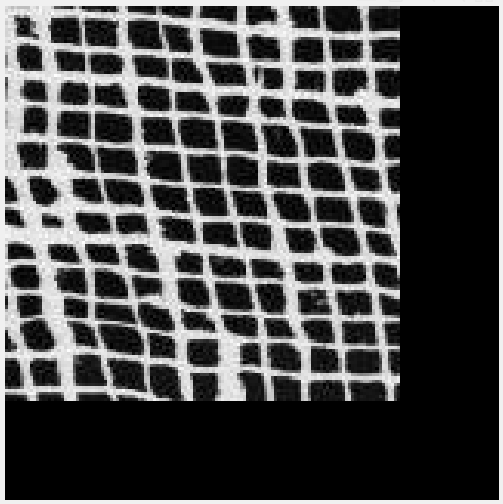
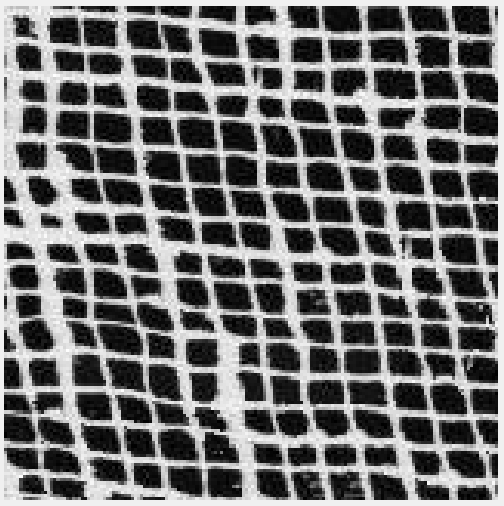
1. neighbour size = 91

Original texture7 : 128x128 Enlarged 1.25x texture7: 160x 160

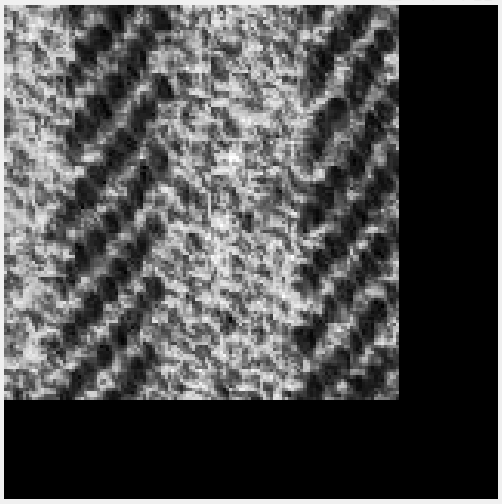
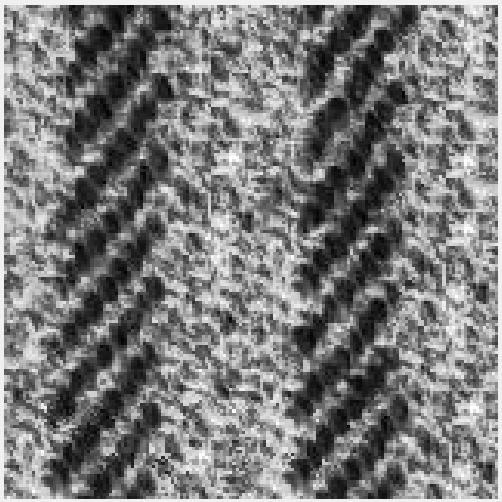
1. neighbour size = 31

Original texture8 : 128x128 Enlarged 1.25x texture8: 160x 160

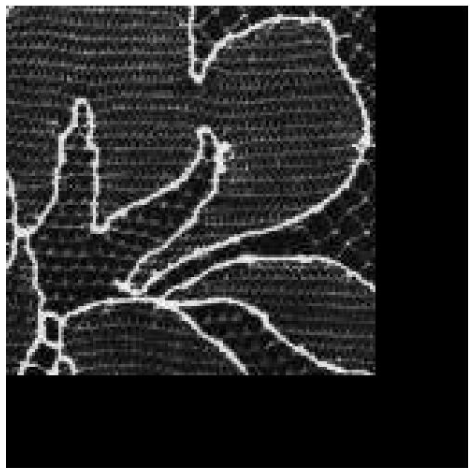
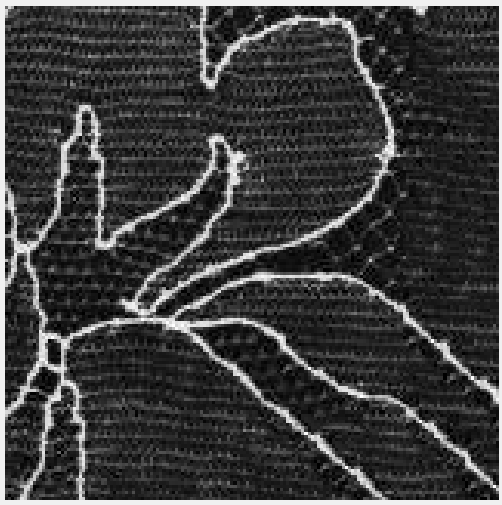
1. neighbour size = 31

Original texture9 : 128x128 Enlarged 1.25x texture9: 160x 160

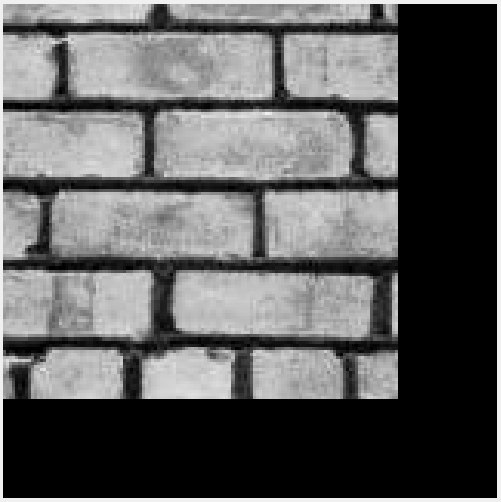
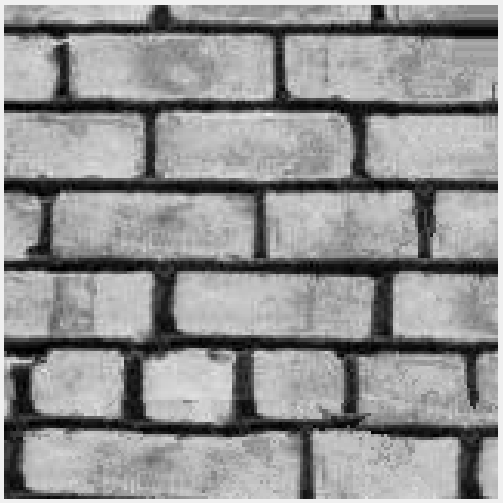
1. neighbour size = 17

Original texture10 : 128x128 Enlarged 1.25x texture10: 160x 160

1. neighbour size = 31

Original texture11 : 128x128 Enlarged 1.25x texture11: 160x 160

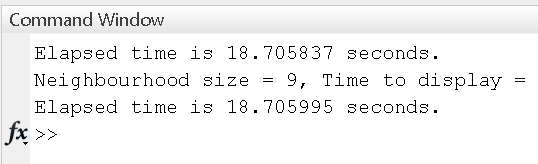
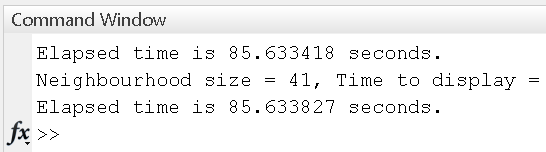
1. How neighbourhood **size** affects synthesis **results** and **speed**?

For texture sample without any discernible pattern like texture 1, texture 5 and texture 10 a medium sized neighbourhood size results in better results because large features in the image will begin to dominate and the small details will not be captured for large neighbourhood size. When neighbourhood size is too small distance relations between patterns at a larger pixel distance is missed. An example as in texture 1 is shown below. A similar comparison can be made for texture 5.

neighbour size = 9 neighbour size = 21 neighbour size = 41

Texture synthesis is faster when neighbour size is small. For example: for Texture 1 with neighbourhood size of 9, it takes 18.7 seconds to create larger texture. For texture 1 with neighbourhood size of 41, it takes 85.6 seconds to create larger texture.

1. What kind of **data** works best/worst?

Data with patterns that are regular and repeated in the texture sample like texture 4 work best. Data with irregular pattern and without contrast like grayscale images work worst. Texture 1 gives the worst results depending on the choice of neighbourhood as shown above in 1).

1. How the implemented **algorithm** can be **improved** in terms of **result quality** and **efficiency**?

At the moment, the neighbourhood size, sigma of gaussian (spread), eps (0.1, error threshold to ensure at least one match) are arbitrarily chosen by the user. Therefore, they cannot be optimal. We can first process the texture sample to find repeating patterns in the images and define the neighbourhood size and sigma accordingly.

If there are too many zeros or close to zeros in the grayscale image like in texture 10 we can first scan for these zero pixels and avoid them in the calculation so that processing can be sped up.

Matlab code is given as Texture\_synthesis.m . Fill in the User Defined Parameters to run code.

**Part 2) : PatchMatch**

Result for Water Lily

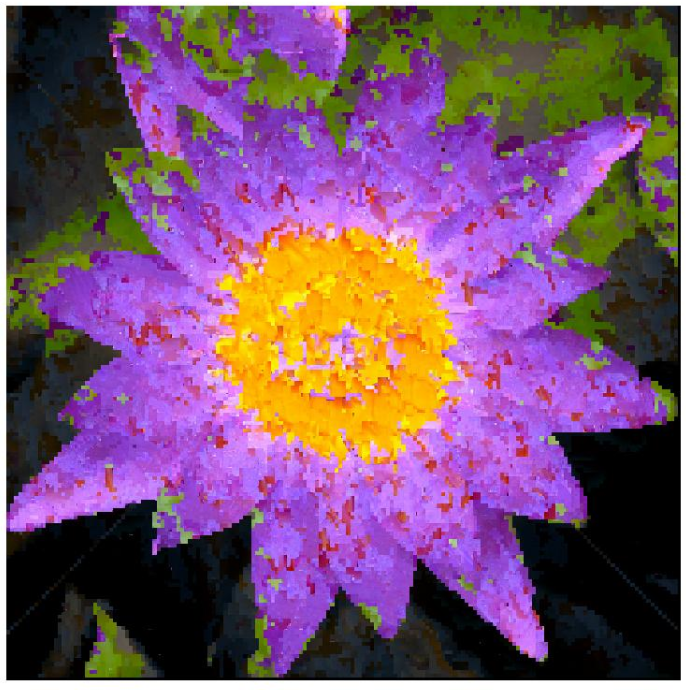
**Image A**



**Image B**



**Reconstructed Image A**



Matlab code can be found at PatchMatch. Fill in the user defined parameters to run code.

\*Please use images with at least 512x512 resolution to get good results.