Image Quilting and Texture Synthesis

Huzefa Chasmai - 15D170013
Aniket Shirke - 150100012
Samarjeet Sahoo - 150100017

CS 663 - Digital Image Processing

Problem Statement :

There are two aims of the project :

1) **Texture Synthesis** : Generate a new image by stitching together small patches of existing images maintaining the texture. This process is called Image quilting.
2) **Texture Transfer** : Rendering an object with a texture taken from a different object.

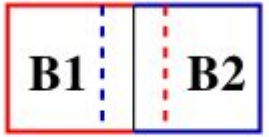
We are essentially implementing the ideas presented in the paper Efros et. al [1].

Algorithms Implemented: Notations

B_i : unit of synthesis (user defined square block)

S_B : set of all such overlapping blocks

B_{ov} : overlap region between two blocks to be B_i/ 6.



neighboring blocks constrained by overlap

We try to place blocks one after the other by placing newer blocks that overlap well with the already present blocks and generate the whole output image in a block by block manner.

Algorithms Implemented: Texture Synthesis

The following outlines the algorithm for image quilting :

Step 1:

We randomly choose the first block B_i which will be the starting block of our new image.

Algorithms Implemented: Texture Synthesis

Step 2:

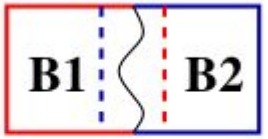
For the given block we choose from S_B all blocks that have the overlap error (defined as the sum of absolute difference between the pixels in the overlap area of two blocks) within a specified error tolerance. We take the error tolerance of both the top and left blocks. From this set we randomly pick a block. This randomness is to ensure certain **stochasticity** in the algorithm or else we get a deterministic result based on the first patch.

Algorithms Implemented: Texture Synthesis

Step 3:

Compute the error surface (the error for each of the pixels in the overlap area). If $\mathbf{B_1}$ and $\mathbf{B_2}$ are the two patches and $\mathbf{B^{ov}_1}$ and $\mathbf{B^{ov}_2}$ be the overlap region, then the error surface is $\mathbf{e = |B_1 - B_2|}$. Find the minimum error boundary along the error surface using an iterative Dynamic Programming approach. We set all pixels to the left and above the boundary of the current block to the values of the already present block. We now paste this new block and repeat the above steps until we acquire the image of desired size.

Minimum Error Boundary Cut



minimum error boundary cut

We calculate the minimum error boundary cut using dynamic programming recursively in the following manner :

1) For the first row we store the minimum error cut boundary to cover the row, i.e. for each col we store the $\mathbf{e_{ij}}$ value.
2) We recursively calculate the minimum error value for i = [2, N] and $\forall$ j as $\mathbf{E_{i,j} = e_{i,j} + \min(E_{i-1,j-1}, E_{i-1,j}, E_{i-1,j+1})}$.
3) The minimum value in the last row in E will indicate the end of the minimal vertical path through the error surface and we backtrack and trace the path of the best cut.
4) A similar approach is followed for finding the minimal horizontal path through the error surface

Algorithms Implemented: Texture Transfer

For texture transfer a correspondence map is defined called ***C*** which maps a particular quantity of the target image (for e.g. the intensity, the luminance or blurred luminance) while also ensuring that the texture matches the source texture file.

Thus we have two constraints:

1) Output should be legitimately synthesized from source texture, that is the neighboring blocks should go well together
2) The correspondence mapping is respected, that is the generated block at a particular position should also go along with the correspondence values at the same position

Algorithms Implemented: Texture Transfer

We modify the error term of the image quilting algorithm to be the weighted sum

:= α x (block overlap matching error) + (1 - α) x (error b/w the correspondence map pixels within the source texture block and those at the current target image position)^2.

And use the same quilting algorithm for texture transfer.

Datasets :

Some of the images we used were from the paper itself. For comparison purposes, we extracted the images from the following resources :

1) https://people.eecs.berkeley.edu/~efros/research/quilting/results.html
2) https://people.eecs.berkeley.edu/~efros/research/quilting/results2.html

We tried our own textures too like this (Google search for textures):

1) Texture
2) Seamless Texture
3) Fruit Texture

Results - Texture Synthesis (Paper Data) :

Results - Texture Synthesis (Paper data):

Paper Results

Texture

Our Results

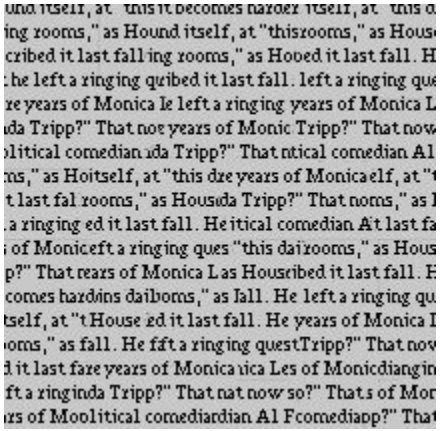
Results - Texture Synthesis (Paper data):

Our Results :
Patch Size = 60

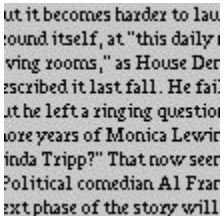
Our Results :
Patch Size = 36

Results - Texture Synthesis (Paper data):

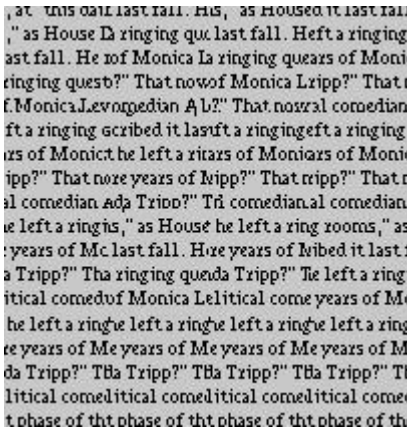
Paper Results



Texture

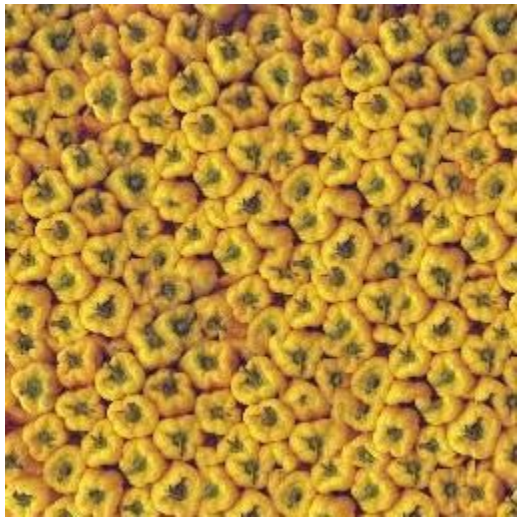


Our Results



Results - Texture Synthesis (Paper data):

Paper Results



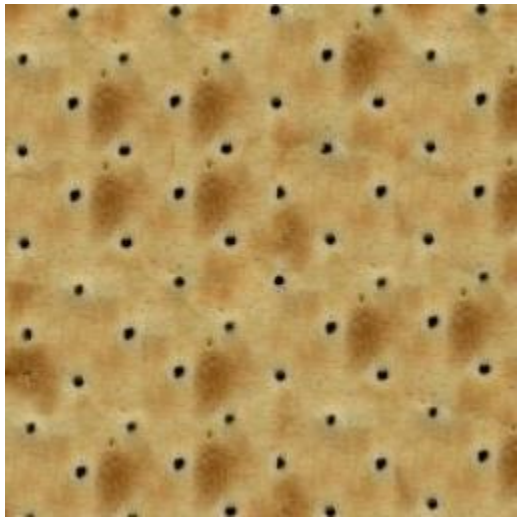
Texture

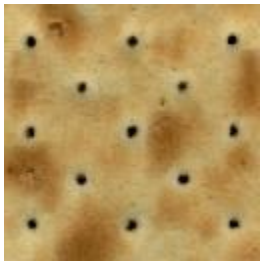
Our Results

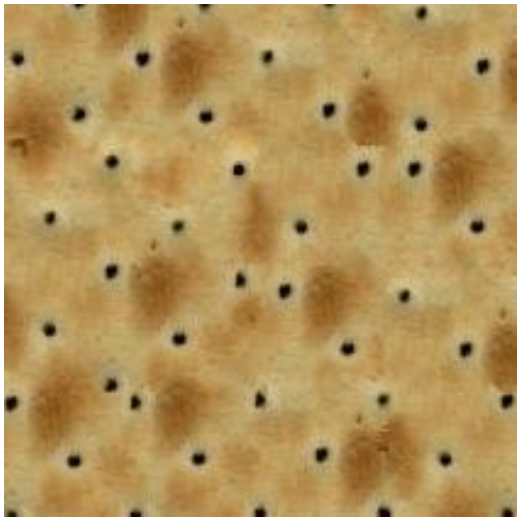
Quiz Time

Results - Texture Synthesis (Paper data):

Results - Texture Synthesis (Paper data):

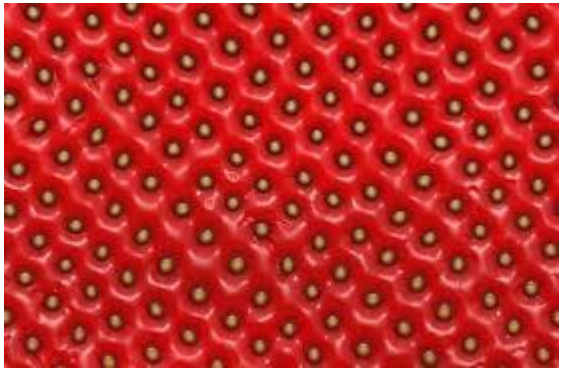


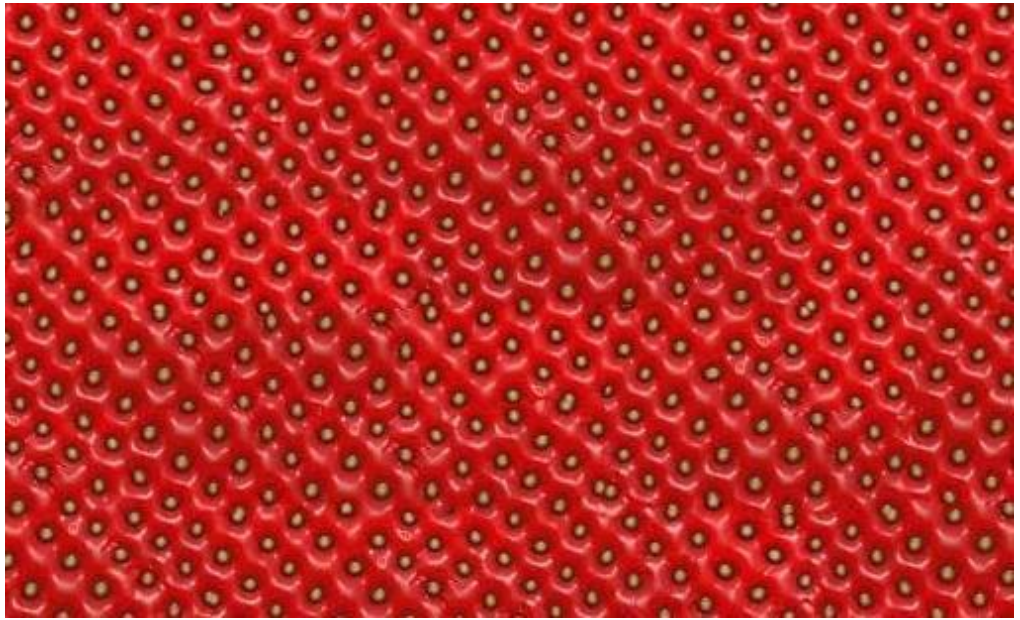




Results - Texture Synthesis (Own Data) :

Results - Texture Synthesis (Own data):





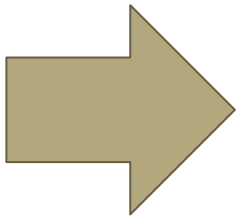
Results - Texture Synthesis (Own data):

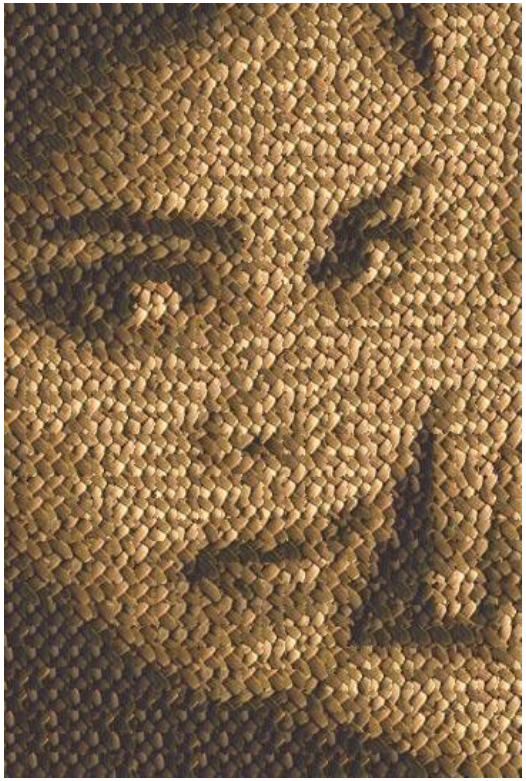
Results - Texture Synthesis (Own data):

Results - Texture Synthesis (Own data):

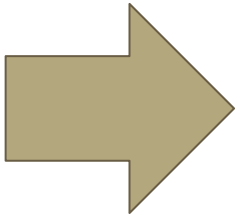
Results - Texture Transfer :

Results - Texture Transfer :

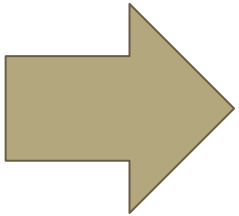




Results - Texture Transfer :



Results - Texture Transfer :



Intensity Transfer

Luminance Transfer

Conclusion :

1) The Image Quilting algorithm extremely well in case of stochastic textures and fairly well in semi-structured textures
2) The time taken to run the algorithm obviously depends heavily on the patch size
3) The image quality depends on the texture and the patch size
4) The algorithm presented is a very simple yet elegant approach towards solving the problem
5) Just like other contemporary texture synthesis algorithm, this is not a perfect algorithm and has some minor drawbacks
6) Our algorithm has a complexity of $(MN)^2 / P^2$

Conclusion :

6) Drawbacks include repeated patterns and minor discontinuity in a few blocks depending on the texture, stochasticity factor and the patch size

7) Texture Transfer takes a along time, as we transfer texture on real portrait images - Images are larger than the texture images and the extra loss factor requires more time to compute as well

8) We have used a single pass for texture transfer. Multiple passes might yield better results with dynamic weighing factor and changing block sizes

Contribution :

We have worked on all the parts of the project together, including coding and report, just as we did for our assignments!

References :

[1] : Efros, Alexei A., and William T. Freeman. "Image quilting for texture synthesis and transfer." Proceedings of the 28th annual conference on Computer graphics and interactive techniques. ACM, 2001.

Thank You

Results - Texture Synthesis (Paper data):

Texture

Our Results