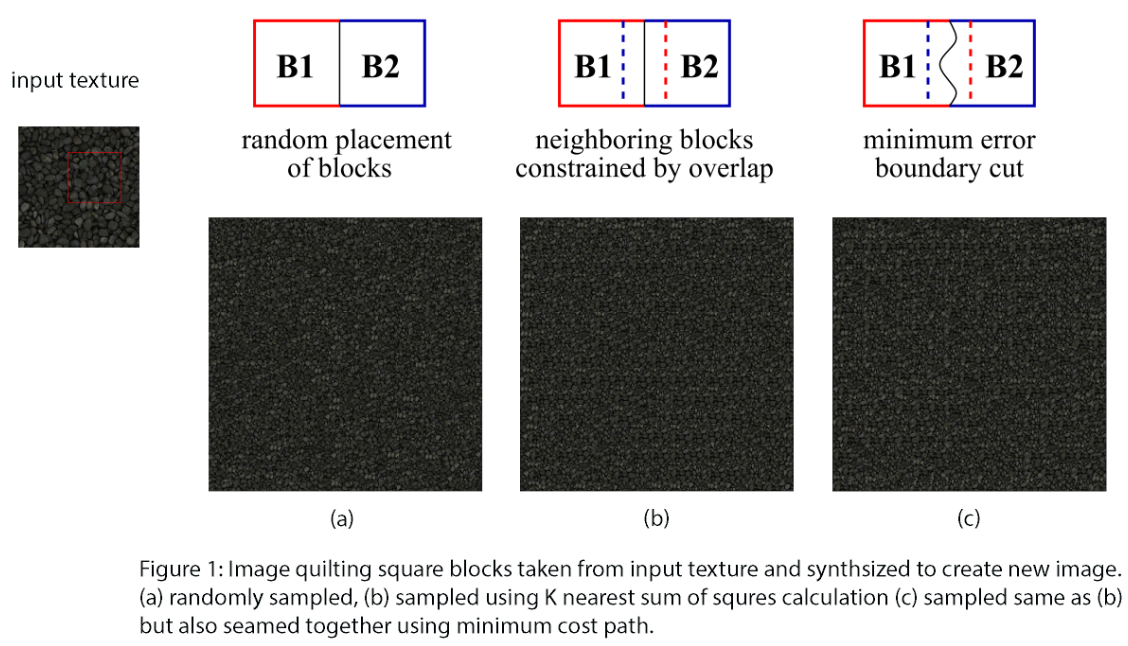
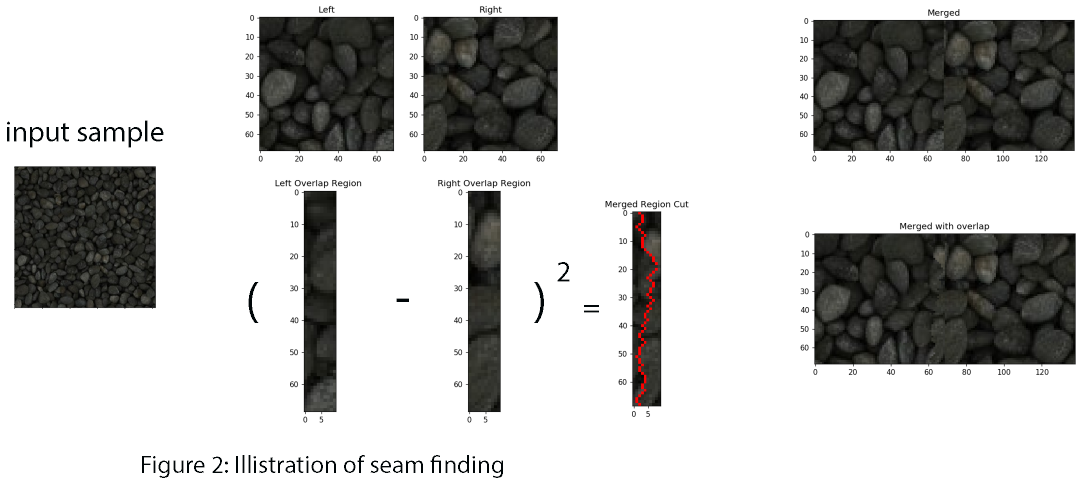
Figure 1 below shows the different methods for creating a large textured image from a small sample. The image chosen to compare the three methods of random, overlapping, and seam-finding, is of small dark colored rocks. In the random sample method a small square sample patch is taken from the image and applied to the output image each time going from left to right up to down. This is shown in figure 1a, where if someone were to zoom in a little more they would be able to see the outline of each patch an determine that in fact the image was randomly chosen.

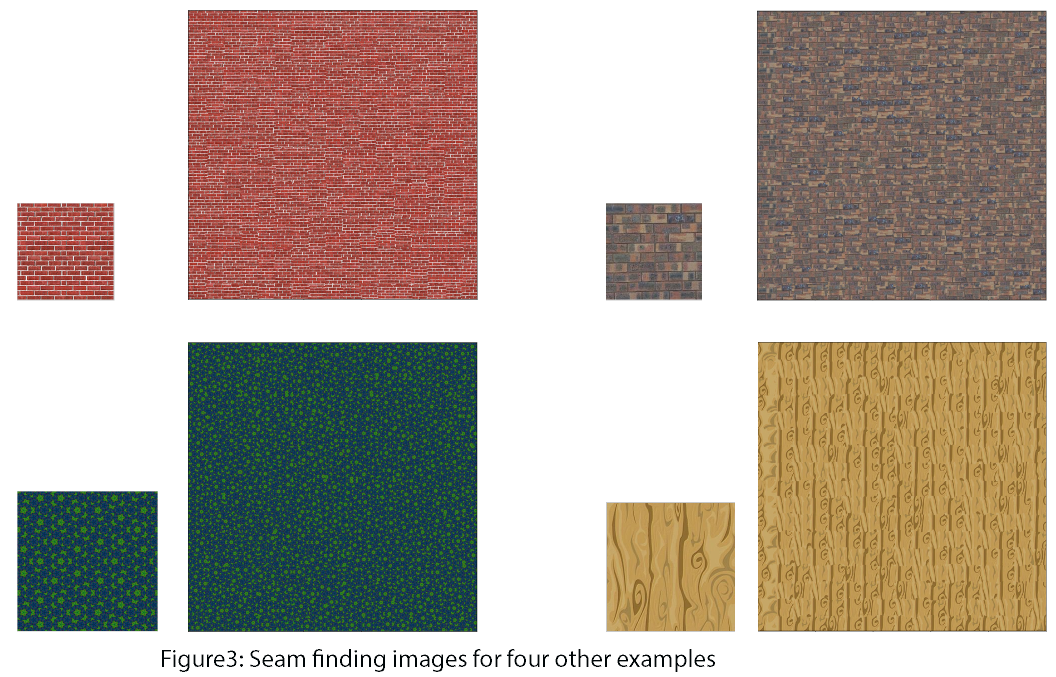
The next figure 1b shows the overlapping patch method by taking the neighboring patch in the output image and finding the sum of squares difference and randomly sampling from the smallest K patches in the sample image. This iteration is done until all the patches are filled. In the code provided a value of K was used as opposed to a tolerance level. The K level is the number of randomly sampled patches that are chosen from, from the lowest k minimum patches.



When comparing the image to 1a, the image is more uniform and now starts to resemble an actual texture image as opposed to a random one. With a lower value of K the image looks like a repeated pattern for each patch. With a very large K the image resembles almost identical to 1a, making it completely random. The parameter K was tweaked to get an ideal ratio, along with the sampled patch size. Too small of a patch size and the image looks unrecognizable and too big is just the sample image overlapping each time.

The final method is the seam-finding method that uses a similar algorithm to find the patch like in the overlapping method. Now the new patch is seemed together with the old patch creating a better transition between the patches in the output image. Figure 1c was the best of the three images, looking more like a full image and very difficult to tell where the actual patches were seemed.

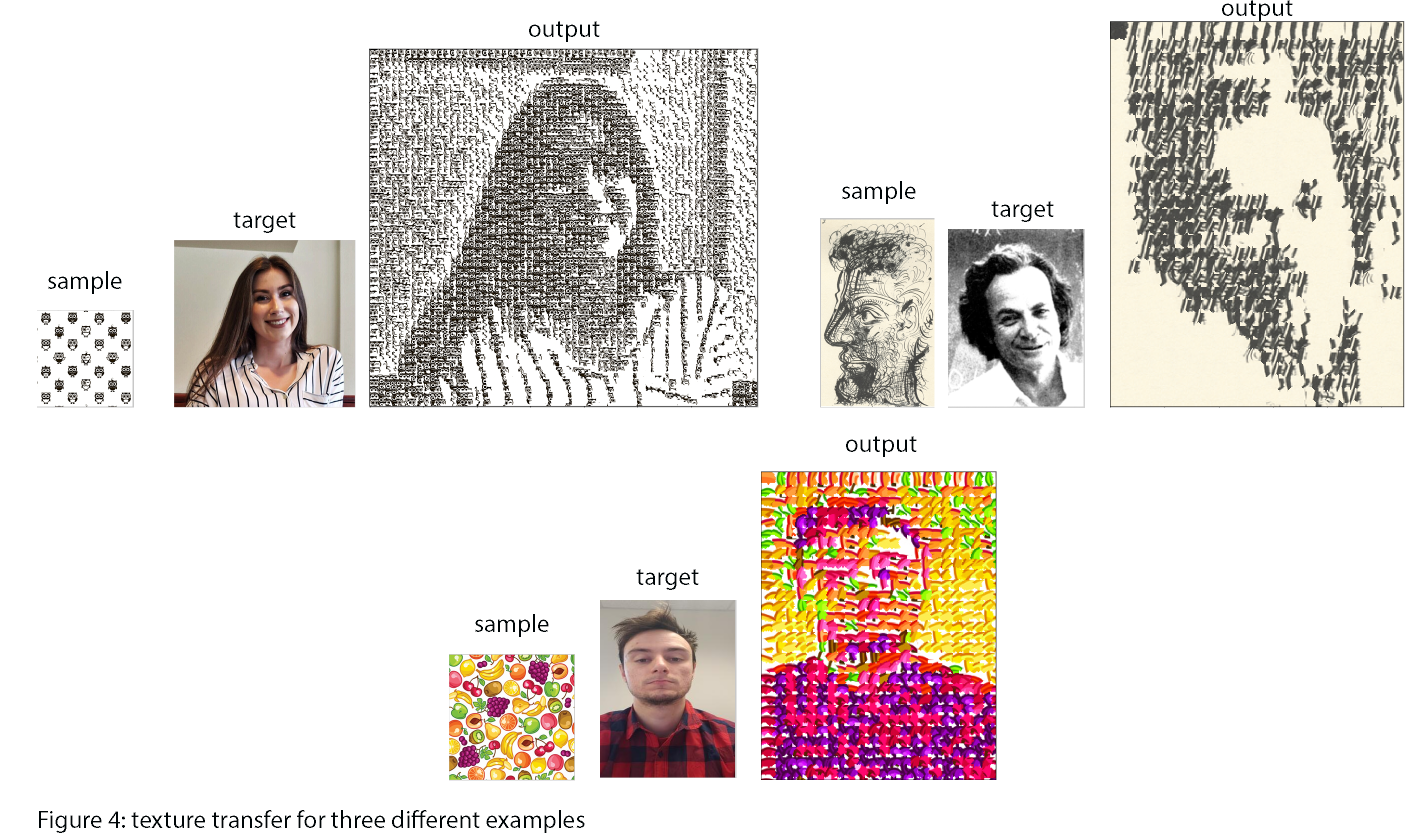


Figure 2 shows a representation of seam finding for one of the vertical cases where a patch is seamed together to its corresponding left patch in the output image. The current patch that we want to fill is calculated by using the sum of squares difference that most closely resembles the patch directly left of the current output from K lowest (SSD\_ values). Once we have that patch we need to calculate the overlapped region of the sum of squares difference to get an error matrix to find the least path from top

to bottom of the overlapped region. Once we have that we can seam the two patches together and add the corresponding cut path in the overlapped region. We will keep doing this iteration until the whole image is filled. The difficult part was the edge cases since the patch sizes were not the same as before. This had to be accounted for throughout the right and bottom edges. We then get a nicely seamed image like in figure 1c.

Figure 3 shows more examples of the seam finding images that were produced and the corresponding input sample patches that they were taken from.

The final task was to create a texture transferred image from a pair of sample and target images. The sampled image would be the texture image and the target image would be where the placement of the texture would go over. The texture transfer uses the same iterative method described in seam finding by finding a patch that has the lowest sum of squares difference to the corresponding point



in the target image. A value of alpha is used to weight which sum of squares difference to us either of the current outputted patch in the output image of the current sum of squares difference in the target image. Again this is our cost function to determine what patch to sample from the sample image. This iteration is done till the whole image is filled. Again edge cases were tricky because of the varying patch sizes. Two examples are show in figure 4 below of the corresponding sample target and output images.

**Sources**

Paper

<https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/papers/efros-siggraph01.pdf>

Lecture Slide

<https://courses.engr.illinois.edu/cs445/fa2019/lectures/Lecture%2007%20-%20Texture%20Synthesis%20-%20CP%20Fall%202019.pdf>

wood

http://www.textures4photoshop.com/tex/wood/free-wood-texture-with-high-resolution.aspx

rocks

https://www.arroway-textures.ch/sites/default/files/styles/square\_teaser/public/gravel-005.jpg?itok=Vn6D4mjI

owl

<https://cdn5.vectorstock.com/i/1000x1000/85/94/pattern-with-black-owls-vector-13748594.jpg>

brick

<https://www.pinterest.com/pin/415808978069274019/>

green pattern

<https://theconversation.com/the-maths-behind-impossible-never-repeating-patterns-63801>

fruit

<https://images.all-free-download.com/images/graphicthumb/vector_fresh_fruit_seamless_pattern_graphics_583573.jpg>

**Points**

**10/10 points** for the random patch texture synthesis with one result.

**30/30 points** for the overlapping patch texture synthesis with one result.

**20/20 points** for the seam finding texture synthesis with five results (including at least two from your own images).

**30/30 points** for texture transfer with at least two results (including at least one from your own images).

**10/10 points** for quality of results (e.g., 0=poor 5=average 10=great)

**100/100 Total**