Name (netid): Xuheng Duan (xuhengd2)
CS 445 - Project 2: Image Quilting

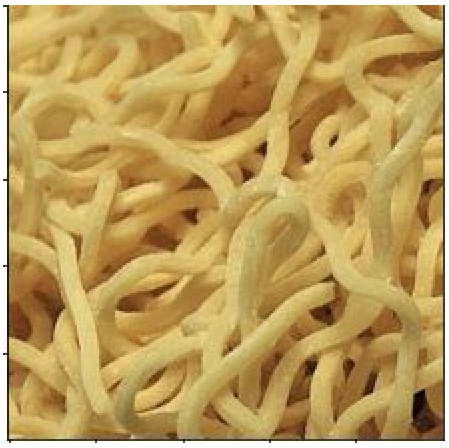
Complete the claimed points and sections below.

Total Points Claimed	**[ ] / 175**
Core	
1. Randomly Sampled Texture	[ ] / 10
2. Overlapping Patches	[ ] / 20
3. Seam Finding	[ ] / 20
4. Additional Quilting Results	[ ] / 10
5. Texture Transfer	[ ] / 30
6. Quality of results / report	[ ] / 10
B&W	
7. Iterative Texture Transfer	[ ] / 15
8. Face-in-Toast Image	[ ] / 20
9. Hole filling w/ priority function	[ ] / 40

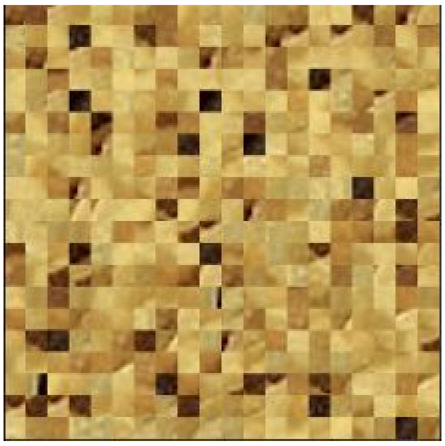
1. Randomly Sampled Texture

Include

- Sample and output images
- Parameters: patch size, output size



SAMPLE



OUTPUT

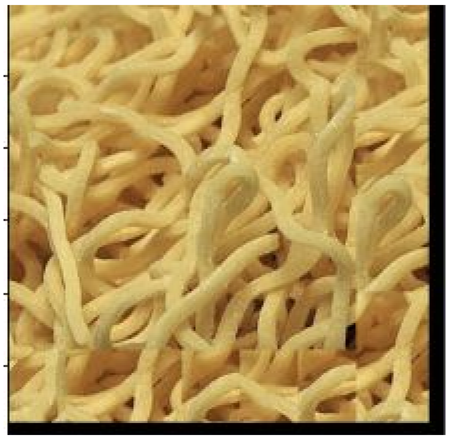
out_size = 200

patch_size = 10

2. Overlapping Patches

Include

- Output image for same sample as part 1
- Parameters: patch size, overlap size, tolerance



OUTPUT

out_size = 1000 # change these parameters as needed

patch_size = 300

overlap = 200

tol = 3

3. Seam Finding

Include

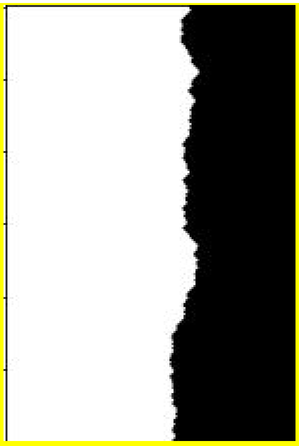
- Output image for same sample as part 1

out_size = 1000 patch_size = 300

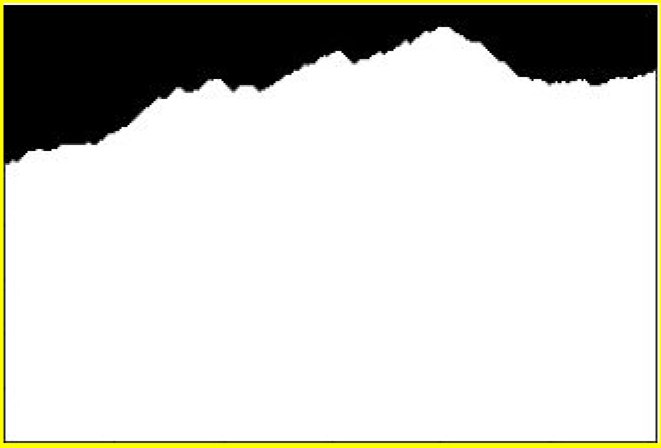
overlap = 200

tol = 3

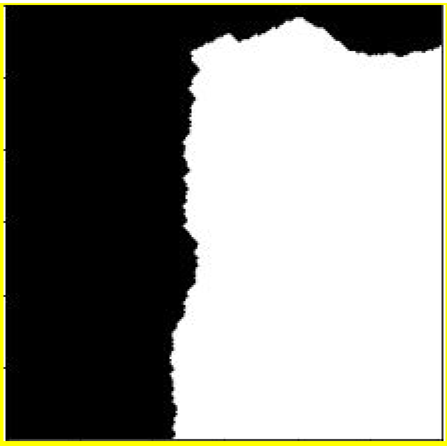
- Illustration: for a selected patch, display (a) the two overlapping portions; (b) pixelwise SSD cost; (c) horizontal mask; (d) vertical mask; (e) combination mask. The mask is binary and tells which pixels come from which patch.



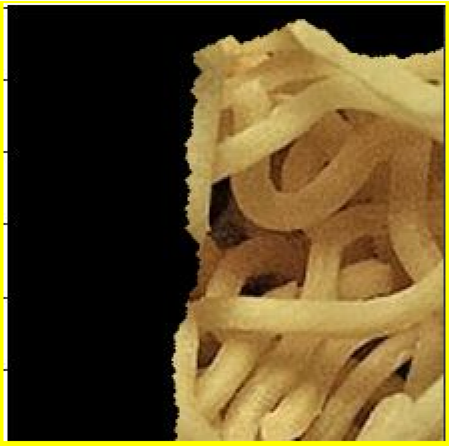
vertical mask



horizental mask



combination of both masks

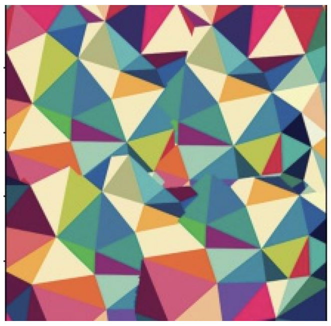


apply combination mask to the randomly picked picture

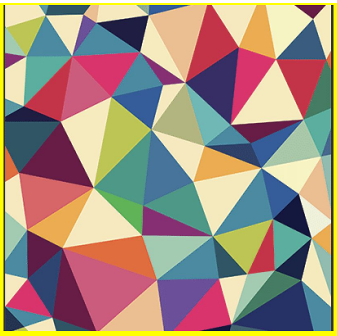
4. Additional Quilting Results

Include

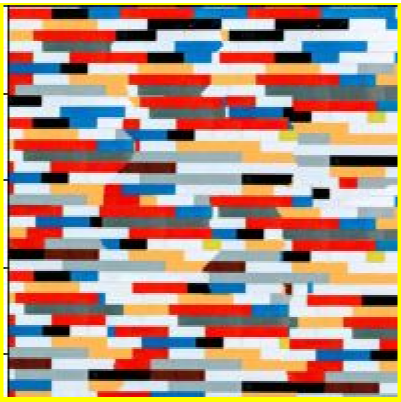
- At least two quilting results on your own images (excluding provided samples). Each result should show input texture image and output, and output should be more pixels than input.



output



Original sample


Output

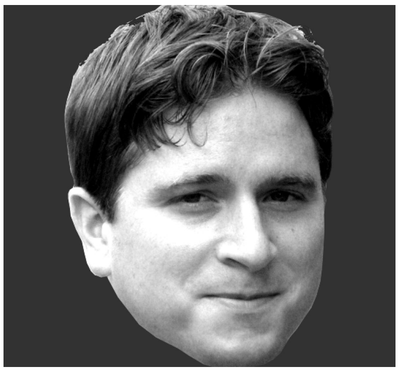
Original sample

5. Texture Transfer

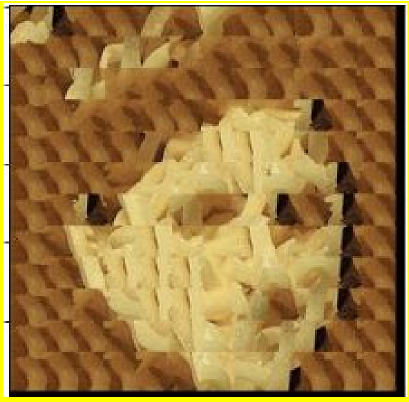
Include

- Brief description of texture transfer method and parameters
  In texture synthesis(part2 and part3), I find the desired patch by comparing the SSD between the template(in output) and the sample. To be more specific, I maintained a priority queue, and I sort the x, y position of the "desired patches" in the PQ by their cost of SSD. In part4, I modify the cost a little bit. The new cost is calculated by the formula,
  cost_SSD = alpha * cost_SSD_overlap+(1-alpha) * cost_SSD_transfer.
  cost_SSD_overlap is the old SSD I calculated in part2 and part3, while cost_SSD_transfer is the additional cost I use to determine the relationship between the guidance graph and sample texture picture. Then I balance both costs by ALPHA and use the result as an index to sort my PQ. Keeping everything other steps the same, I only need to pick the random "tol" patches from the PQ, and build the new output.
- At least two texture transfer results (one result can use provided samples). Include the input texture and target images and the output (output should be same size as target image)

sample picture

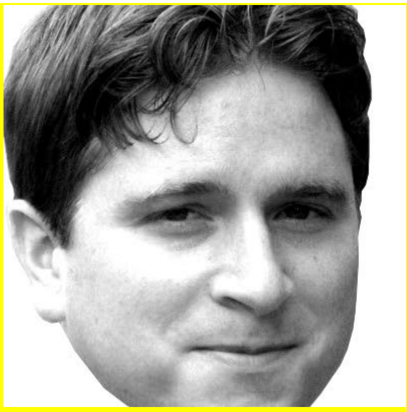


Guidance Picture

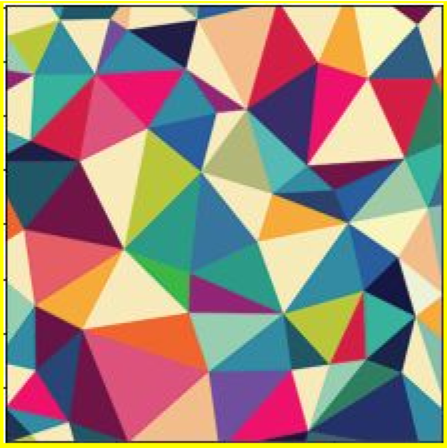


Result Picture

2.



Guidance Picture



Sample Picture

Result Picture

6. Quality of results / report

Nothing extra to include (scoring: 0=poor 5=average 10=great).

7. Iterative Texture Transfer (B&W)

Include

- Describe method
- Results on same images as shown for texture transfer.

8. Face-in-Toast Image (B&W)

Include

- Describe method
- Show input face image, toast image, and final result

9. Hole filling w/ priority function (B&W)

Include

- Describe method
- Show result on at least two images (show input with hole and output)

Acknowledgments / Attribution

List any sources for code or images from outside sources