Homework 2: Texture Synthesis and Image Inpainting
Computer Vision

Xiaoli He

April 8, 2017

1 Texture Synthesis

The synthesized images for T1,T2,..., T5.gif by with WindowSize = 5, 9 ,11, 15,23 pixels are shown in Figure 1. To filled a 200*200 pixel image, the average run time was about 105 s for window size 0f 5 pixels, and the run time doubled as window size increase.

I also tried window size of 15 and 23 pixels, and the effects are different for different examples. For example, 15 pixels worked best for T1, 23 pixels worked best for T5, and for the rest, increasing window size didn't help significantly. In general, if the texture is more complicated, increasing window size helps.

Note: the size of each synthesized images are the same (200 *200 pixels), the appeared sizes are different in Figure 1 due to Latex typesetting...

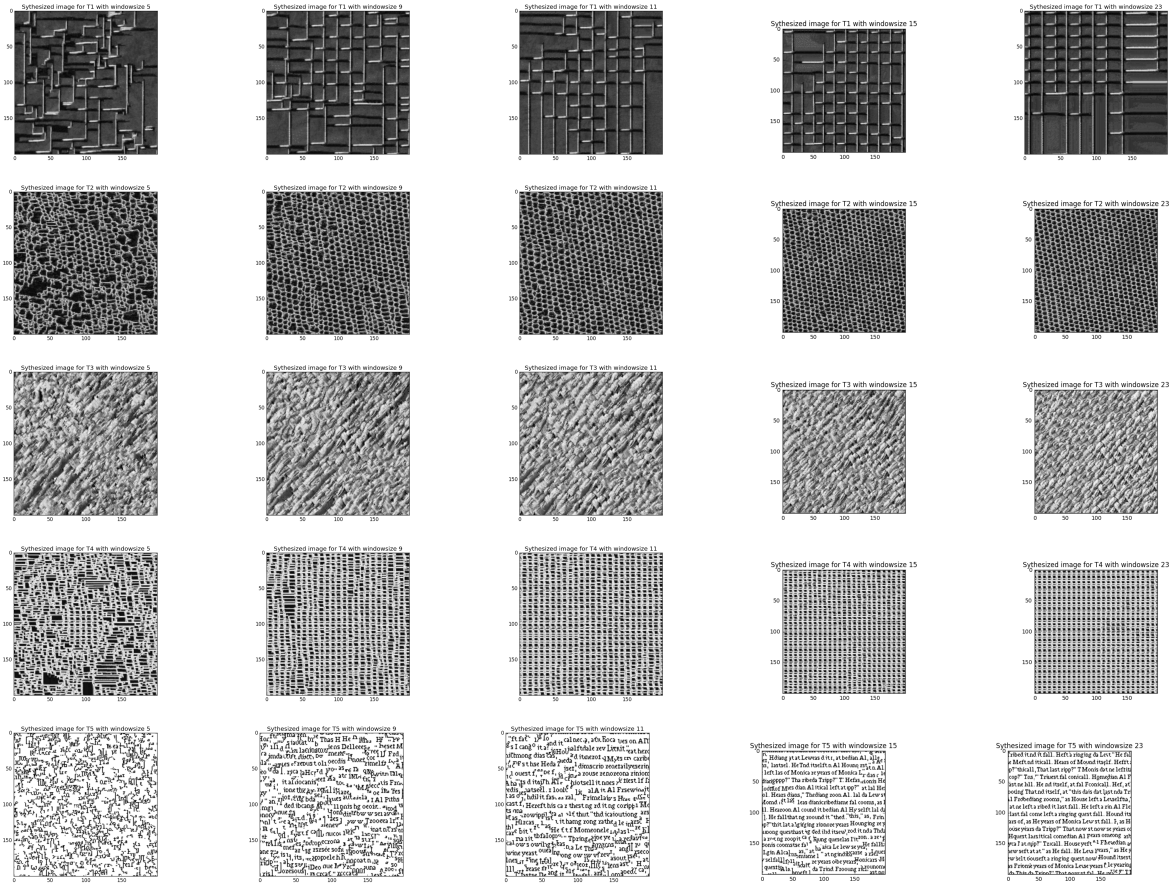

Figure 1: Synthesized images for sample images T1,T2,T3,T4,T5 by with Window Size = 5, 9,11,15,23 pixels

2 Image Inpainting

By analyzing the pattern of the given two test images, I found that the most similar examplar should be near the target point (point to be filled). Therefore, to increase efficiency, I only searched the nearby region of the target point for the best match examplar. This window size is around 200*200 pixels. The program takes about 5 minutes for each case.

The inpainted image for $test\_im1$ and $test\_im2$ are shown in Figure 2. The window size are 5, 9 , 11, 19 pixels. It seems like the large window size(19 pixel) had the best performance.

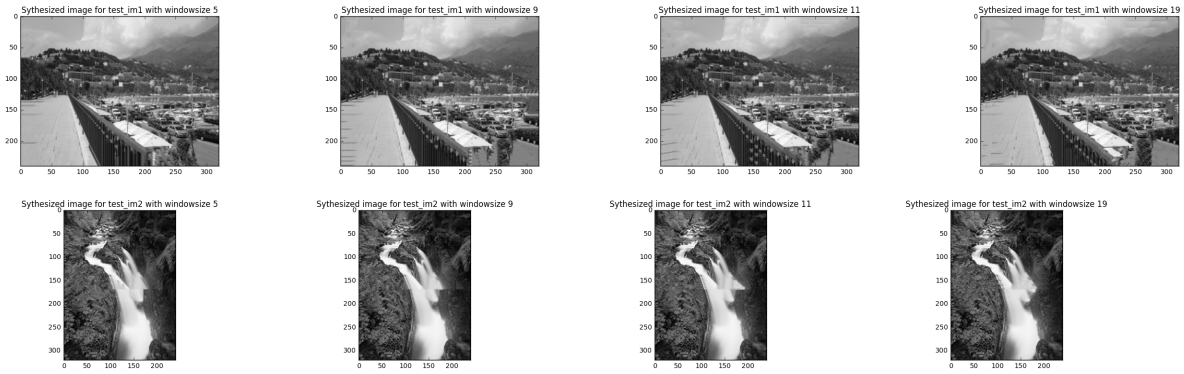

Figure 2: Inpainted images for sample images $test_im1$ and $test_im2$ by with Window Size = 5, 9, 11, 19 pixels

3 Object removal

The result of object removal for *test_im3.jpg* is shown in Figure 3. The first row shows results using Efros and Leung non-parametric synthesis approach (with window size = 9 pixel), and the second row shows results using Criminisi's Examplar-based inpainting method. Each column shows different removal region (from left to right, people, ground, and sign). The window size are 9 pixels for all conditions.

The result shows that for small regions(people and sign) where there are more linear texture, the examplar-based method works better. However, for the large regions (ground), the non-parametric method works better. The reason is that the examplar-based method filled in the whole patch at a time, so each best-match patch plays a larger role, compared to the non-parametric method, where each time only one pixel is filled (you have more chances to correct any error when finding the best-match patches). Another possible reason is that, using SSD to measure similarity will find some patches that are visually different, because SSD doesn't care about the structure information inside the patch. A better way would be to include saliency information inside the patch (such as Hellinger distance) (Roumy, Blanchard, 2012).

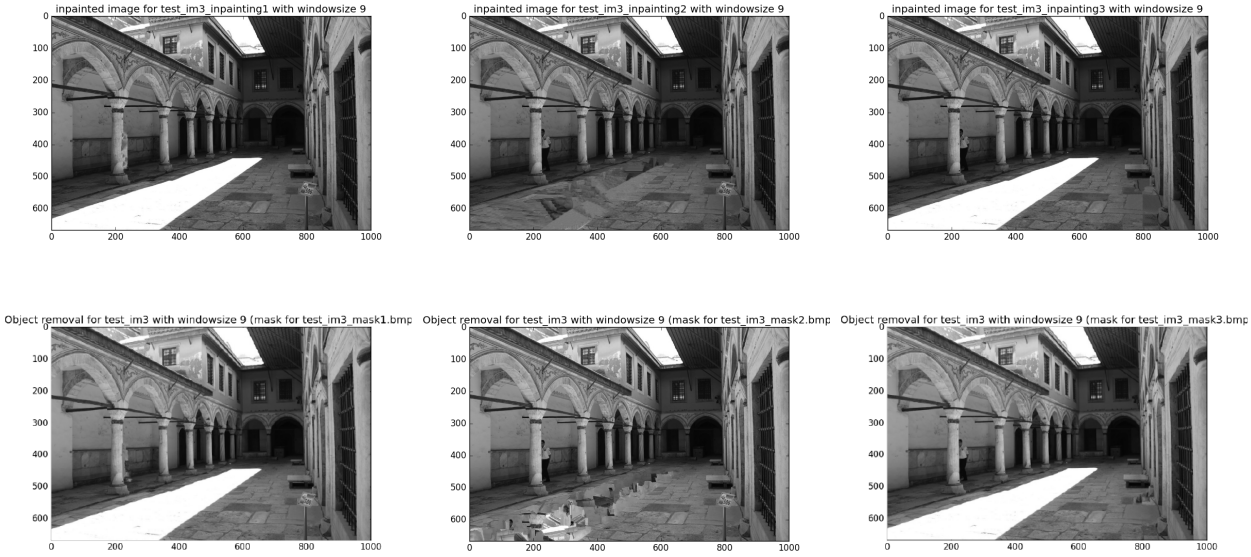

Figure 3: Object removal for test_im3 by Efros and Leung non-parametric synthesis approach (top row) and Criminisi's Examplar-based inpainting method (bottom row). Window Size is 9 pixels.

4 Image quilting

For image quilting, I used an online Matlab code (http://people.csail.mit.edu/thouis/efros-freeman/). The results are shown in Figure 4.The overlap size is 2 pixel.

Similar to non-parametric sampling, quilting is also a patch based texture synthesis method. First, it chooses small patches from sample images randomly, and then it also searches the new block in the sample image. The new block is overlapped with the initial one, and the new block is chosen so that the blocks are similar to the overlapped region. They used ragged edges so that the new block will better approximate the features in the texture.

The difference of quilting method is the way it handles the boundaries of patches. Image quilting exploits a minimum error boundary cut to find an optimal boundary between two patches. The cut defines an irregular path separating overlapping patches, so that each patch provides the synthetic texture only image signals on its side of the path. That's why quilting works better and is more stable for the given images, especially in T1 – it has fewer empty areas because it considers both sides of the overlapped region.

Another difference is that the unit of quilting filling is patch, but the unit for non-parametric sampling is pixel.

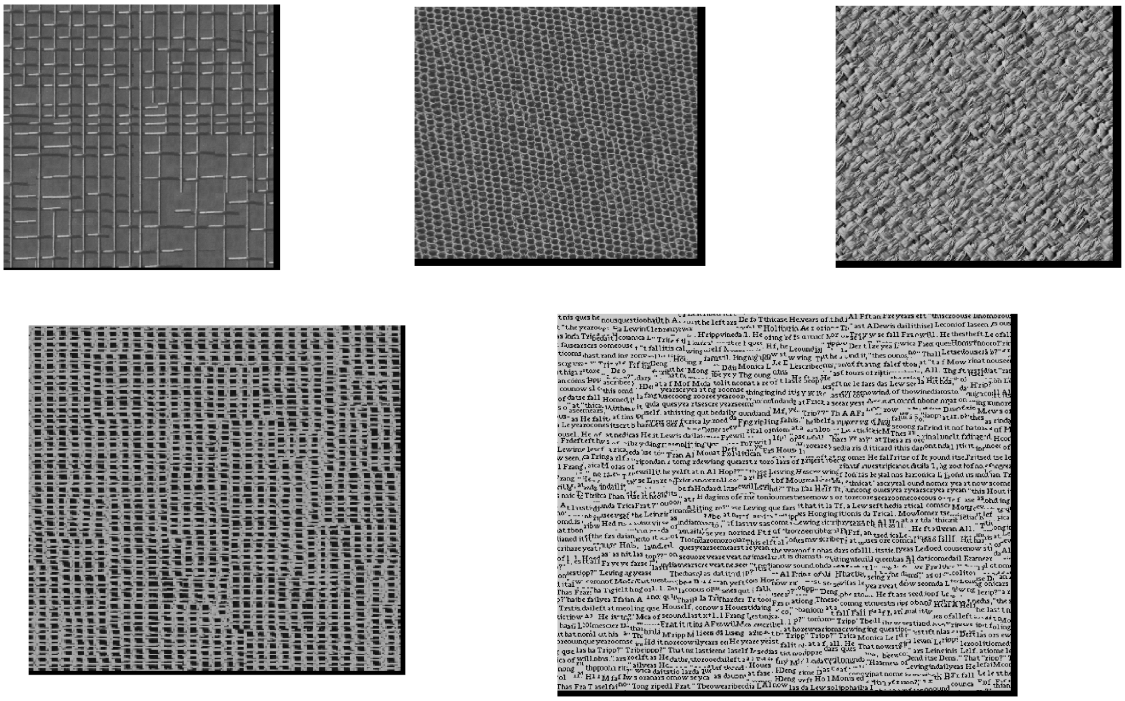

Figure 4: Results of image quilting for T1, T2, T3, T4, T5

5 How to run the code

I ran the code using Pycharm edu on Mac. It works with no bugs.
The main codes for running the given images are in the end of each py file. So just run each py file.
Figures are inside the subfolder 'Assignment-II-images'. Otherwise, you may need to modify directory in the codes.
Each code will save the figures and show them after each run.
The parameters are as followed:

1. 2.2.1: texture_synthesis.py
   synthesize(filename, win_size, shape_newimage = (200,200), shape_seed=(5, 5))
   'filename' doesn't include format.

2. 2.2.2: inpainting.py
   inpainting(filename, filetype, win_size, sample_winsize = 51)
   sample_winsize is the window size of sample image region around the target points.

3. 2.2.3: object_removal.py
   objectremoval(filename = 'test_im3', filetype = '.jpg', maskname = 'test_im3_mask1.bmp', win_size = 9)
   'test_im3_mask1.bmp' is the mask for the people;
   'test_im3_mask2.bmp' is the mask for the ground;
   'test_im3_mask3.bmp' is the mask for the sign.