534

Xinyang Wang

189001002

HW2

1. **texture Synthesis**

In this part, we use Efros algorithm to synthesize new images according to the given sample images. The basic ideas of Efros is to generate a seed first, and then along the margin of the seed, for each pixel A, we find similar neighbors in the sample images, and random pick a pixel from the neighbors to be the value of pixel A.

Below is the output of the synthesized images under different window size and corresponding running time.

And we can see that, window size of 5 fails to synthesize a good images. And as the window size increases (until 15), the results are improving.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  |  |  |  |
| Window size = 5 | Window size = 9 | Window size = 11 | Window size = 15 |
| Running time:192.533 | Running time:221.761 | Running time:391.642 | Running time:431.328 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  |  |  |  |
| Window size = 5 | Window size = 9 | Window size = 11 | Window size = 15 |
| Running time:102.277 | Running time:172.114 | Running time:183.256 | Running time:361.926 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  |  |  |  |
| Window size = 5 | Window size = 9 | Window size = 11 | Window size = 15 |
| Running time:127.582 | Running time:143.708 | Running time:167.912 | Running time:236.573 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  |  |  |  |
| Window size = 5 | Window size = 9 | Window size = 11 | Window size = 15 |
| Running time:61.6848 | Running time:93.8901 | Running time:116.473 | Running time:154.862 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  |  |  |  |
| Window size = 5 | Window size = 9 | Window size = 11 | Window size = 15 |
| Running time:351.466 | Running time:721.034 | Running time:1241.25 | Running time:1581.56 |

**2. Image Inpainting**

In this part, we use Efros algorithm to inpaint the blank parts (or holes) in the given images. Cause in this case, the sample images are larger, the running time of Efros are quite long.

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
| Window size = 5 | Window size = 9 | Window size = 11 |
| Running time:1389.52792001 | Running time: 1969.31244302 | Running time: 2133.69361687 |

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
| Window size = 5 | Window size = 9 | Window size = 11 |
| Running time:743.741944075 | Running time:1048.46696997 | Running time:1445.42782402 |

**3. Object Removal**

In this part, we first remove target area be setting that area to zero and then using Efros algo and CPT(Criminisi, Perez and Toyama) algo to inpaint the target area.

The difference in CPT algo is that CPT algo sets the order in which pixels are synthesized by computing the **priority**. Thus it gives out a better synthesis of images. Also because CPT synthesizes one block at a time, it is much faster than the speed of Efros algo.

Below are the compare of results of Efros and CPT algo, which endorses the conclusion.

|  |  |
| --- | --- |
| Remove person | |
| Efros algo | CPT algo |
|  |  |
| Window size = 9 | Window size = 9 |
| Running time: 5357.91663599 | Running time: 264.031646967 |

|  |  |
| --- | --- |
| Remove sign | |
| Efros algo | CPT algo |
|  |  |
| Window size = 9 | Window size = 9 |
| Running time: 6485.74055321 | Running time: 301.238740921 |

|  |  |
| --- | --- |
| Remove ground | |
| Efros algo | CPT algo |
| Ran it for more than 12 hours, still did not get the result... |  |
| Window size = 9 | Window size = 9 |
| Running time: | Running time: 5255.68351889 |

**4.Image Quilting**

In this part, we use ‘Image Quilting for Texture Synthesis and Transfer’ whose basic idea is that instead of synthesizing one pixel at a time, we actually take the whole similar neighbor, which obviously increase the speed of synthesis.

Of course if we want to make synthesized images to be consistent, patches need to be overlapped. So we use the minimum error path to decide how to overlap. Below are some results of this algo on example images.

As for **quality**, image quilting produced a little better synthesis in some aspect compared to Efros algo implemented in the part 1. As we can see in the three examples(T1.gif,T2.gif,T3,gif), under the same window size, there are hardly any missing lines when implementing image quilting. But a defect exists, which is the discontinuous lines occurring where the patches overlap.

Due to the limitation of time, I have not done too many tests. But according to my guess, image quilting will do better if we carefully choose *OverlapWidth and InitialThresConstant.*

|  |  |
| --- | --- |
|  | |
| Efros algo | Image quilting |
|  |  |
| Window size = 15 | Window size = 15 |
| Running time: 431.328315973 | Running time: 148.345655434 |

|  |  |
| --- | --- |
|  | |
| Efros algo | Image quilting |
|  |  |
| Window size = 15 | Window size = 15 |
| Running time: 361.926251888 | Running time: 126.34454326 |

|  |  |
| --- | --- |
|  | |
| Efros algo | Image quilting |
|  |  |
| Window size = 15 | Window size = 15 |
| Running time: 236.573104143 | Running time: 97.9874993345 |