Non-parametric Texture Synthesis

I. Basic Algorithm

According to the algorithm introduced in assignment PDF, the pseudo code of the basic algorithm is shown as below.

```
read in Texture sample as matrix
Image;
put Image on the left-up corner
of Large;
for unfilled pixels
find the pixels Pi with maximum
known neighbors;
get known pixels P_known around
Pi within Window_size;
culculate SSD of P_known and
Windows in Image;
if SSD of Window_n is minimum
fill Pi with center of Window_n
N;
end
```

II. Algorithm Implementation

If a n x n sample texture is used to generate a 2n x 2n large texture, every unfilled pixel's neighbors need to be compared with all window pixels in sample texture. So that the time complexity is log (n^4), which is quite huge. Therefore, if fill pixels and compare windows pixel by pixel, it will take too long time synthesizing an image. To decrease time complexity, in MATLAB, the vector operations can be used. If all operations are converted to vector operations, the time complexity will be greatly reduced to log (n^2).

Because the sample texture is on the left-up corner in the large image, the pixels with the most known neighbours are tightly close to the edge of the sample texture at first. Besides, in every iteration the known pixels expand close to the last iteration. Therefore, the close edge of known pixels can be taken as filling pixels every iteration.

By the 'im2col' function in MATLAB, every possible window can be convolution to vectors for vector operation. According to the convolution arrange, the Gaussian parameter for every pixel in one column can be converted to vector as well.

Vector operations greatly reduce the time complexity and increase efficiency of the algorithm.

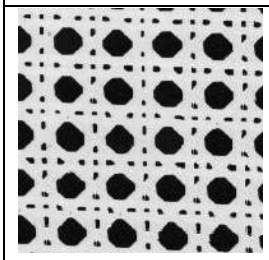
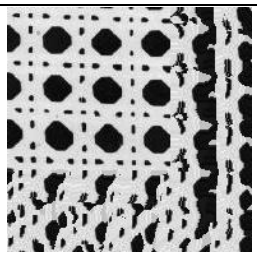
III. Influencing Factors

i. Window size

In addition to the size of the generated image, the window size affects the running speed of the program the most. The smaller the window size, the slower the program running speed. At the same time, window size also greatly affects performance.

When choosing window size, the window size must be large enough to contain regular patterns, and not too large so that the available centre points are limited, and the best fit cannot be achieved.

Below shows result of texture 7 in different window size.

window size=30	window size=55
	

ii. Sample texture feature

When the sample texture graphics are messy, and there are no line segments or slices of graphics that need to be aligned, the algorithm performs better, because alignment is not required, and the result is more tolerant and more natural.

When the regular pattern of the sample texture is large, if the window size is not enough

to contain the regular pattern, the result will be worse.

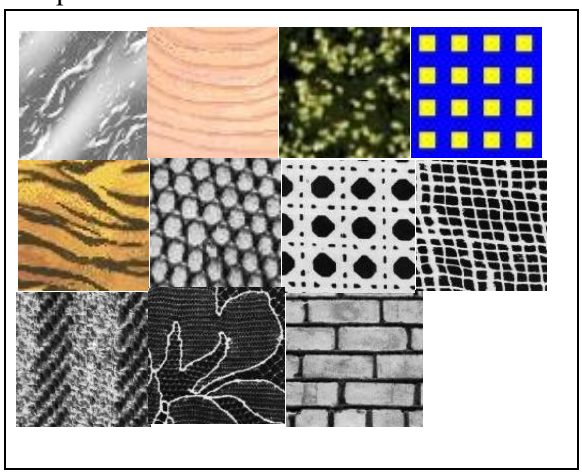
The smaller the picture, the fewer pixels that need to be filled, and the shorter the program running time.

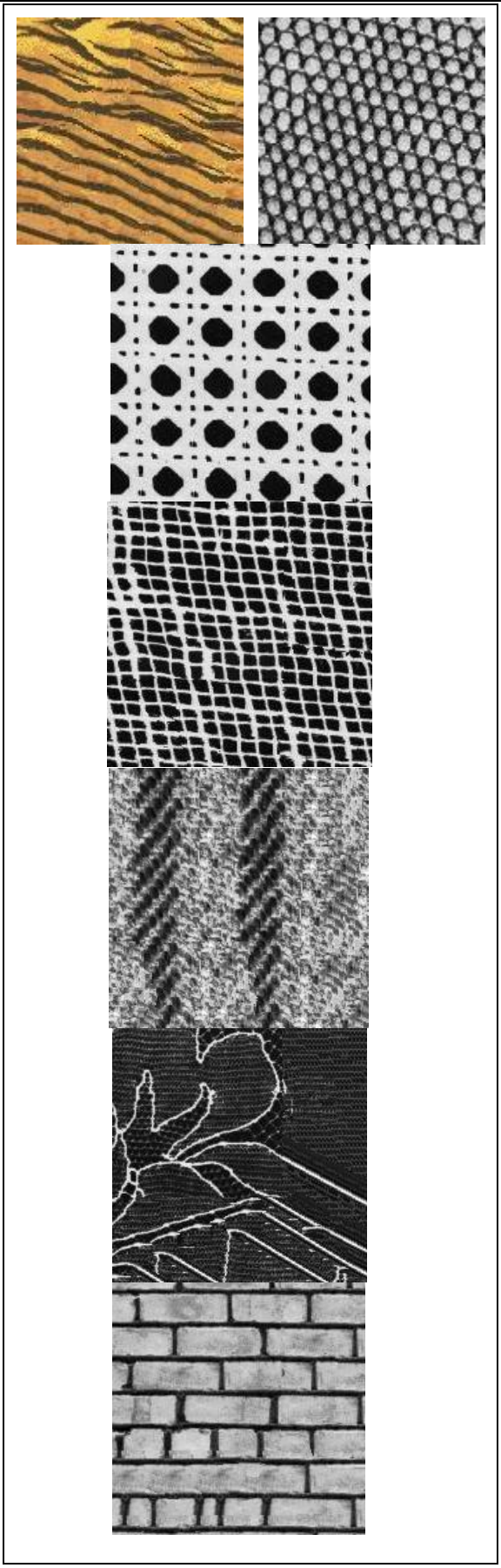
iii. Sigma

For Gaussian weighting, the pixels farther from the centre have a smaller weight. Sigma is an exponent in the Gaussian function. When the sigma is larger, the pixel in the distance has less influence on the result. Therefore, different window sizes should have different sigma. After many experiments, when sigma= window_half / 3, the algorithm has better performance.

IV. Result

Sample textures:





Synthetic textures: