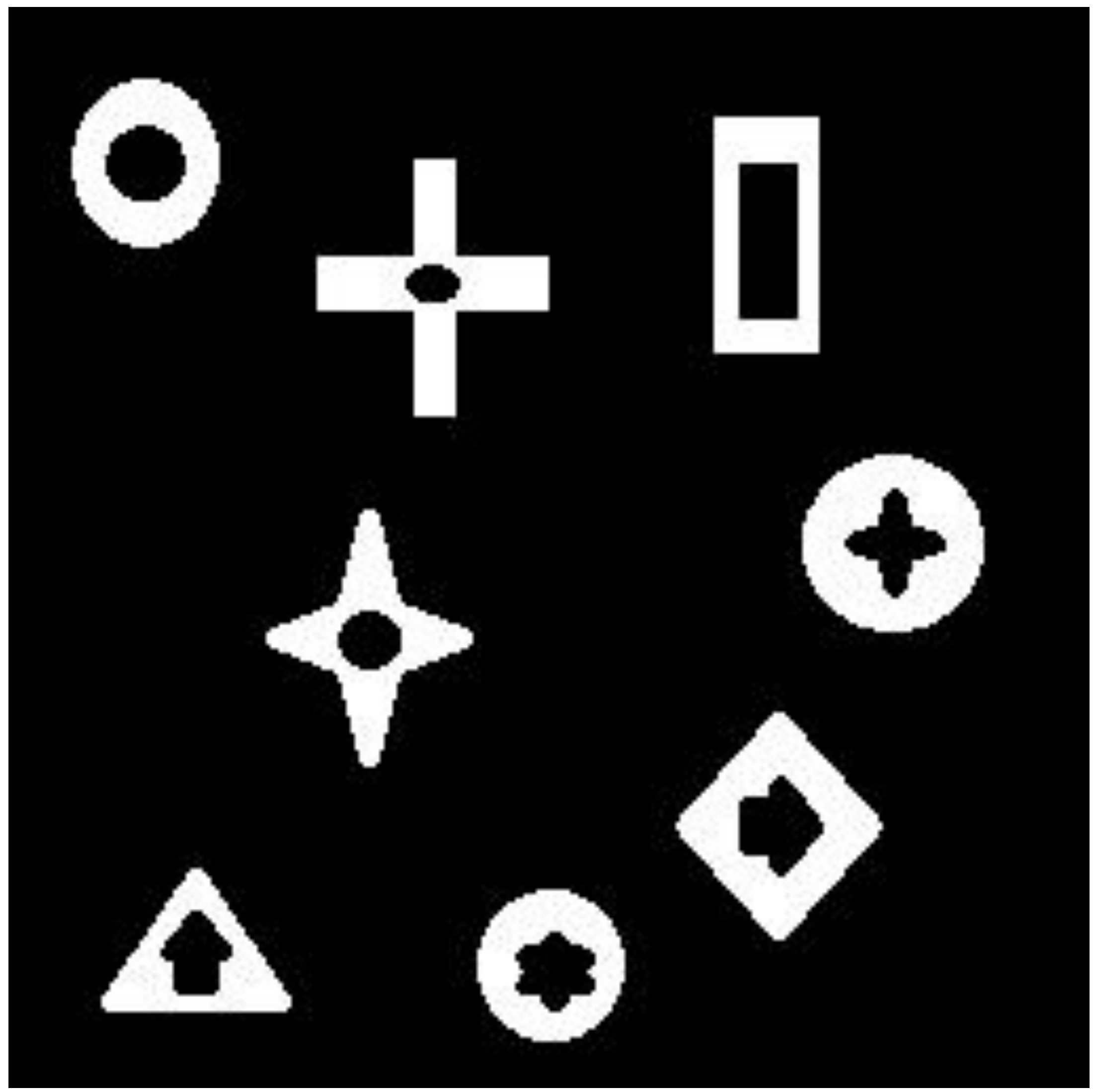
DIP Homework #3

呂翊愷

Problem 1(a) - Boundary Extraction

- Perform boundary extraction on I1 to extract the objects' boundaries.

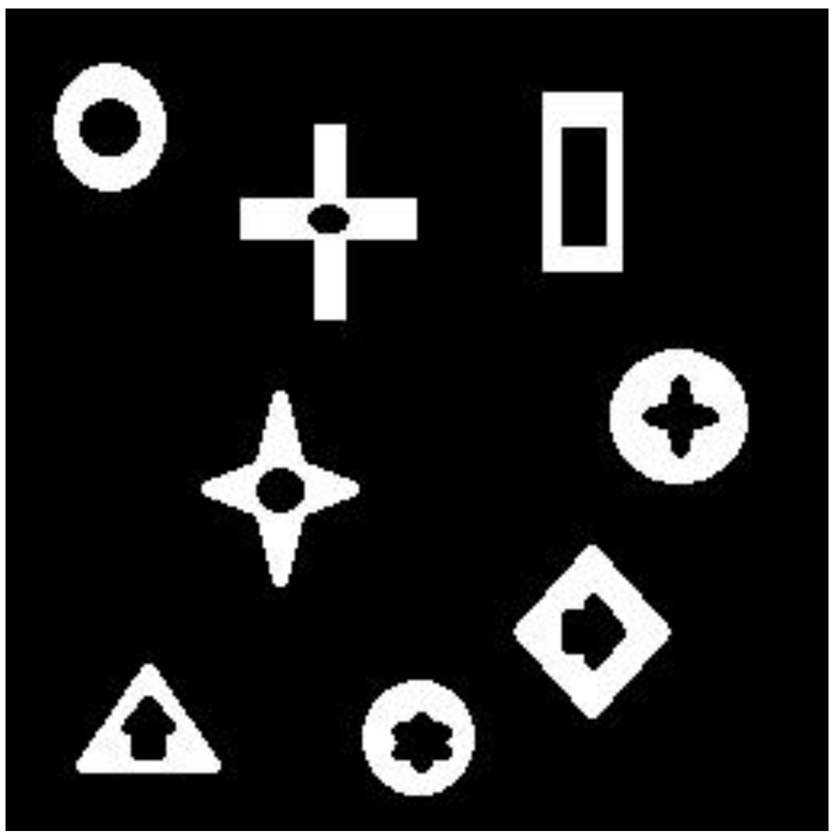


I1

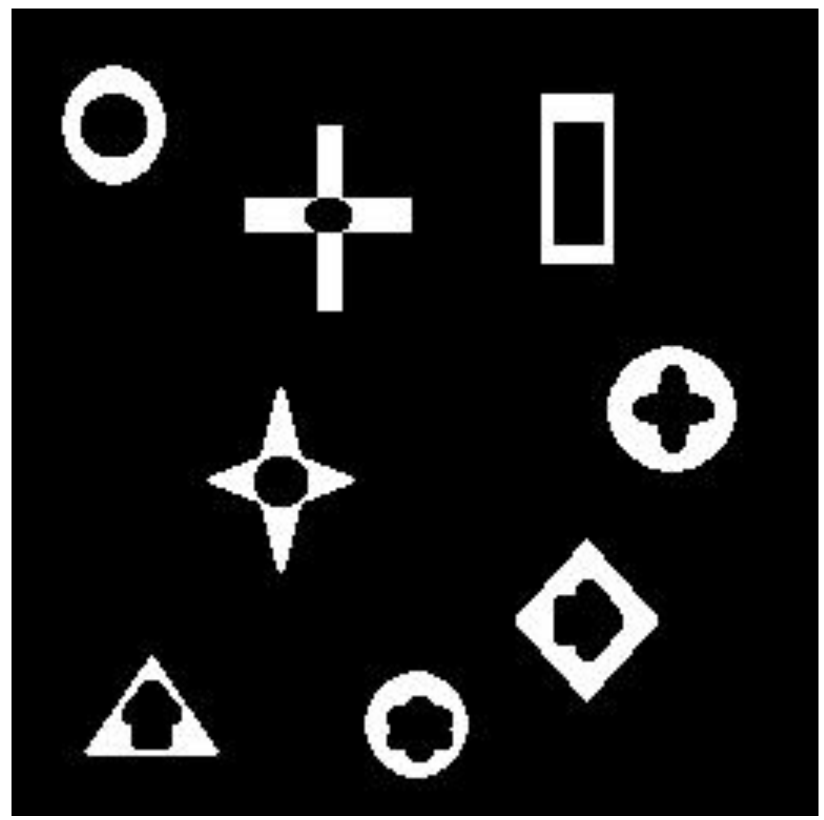
Boundary Extraction

- $B(i,j) = I_1(i,j) - (I_1(i,j) \ominus H(i,j))$

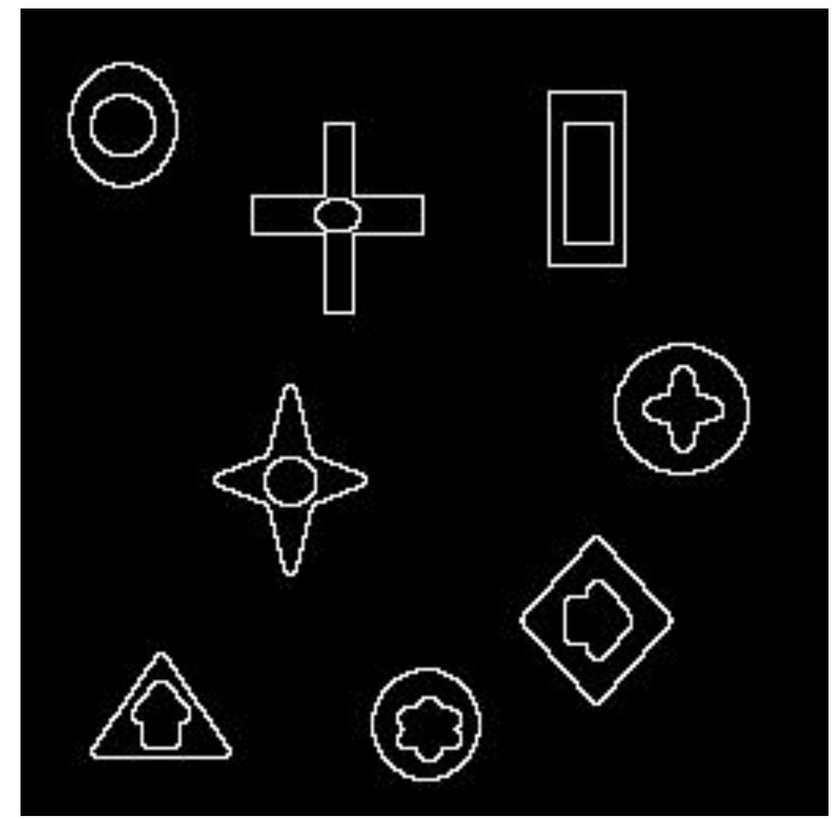
$$H = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



I1



erosion of I1



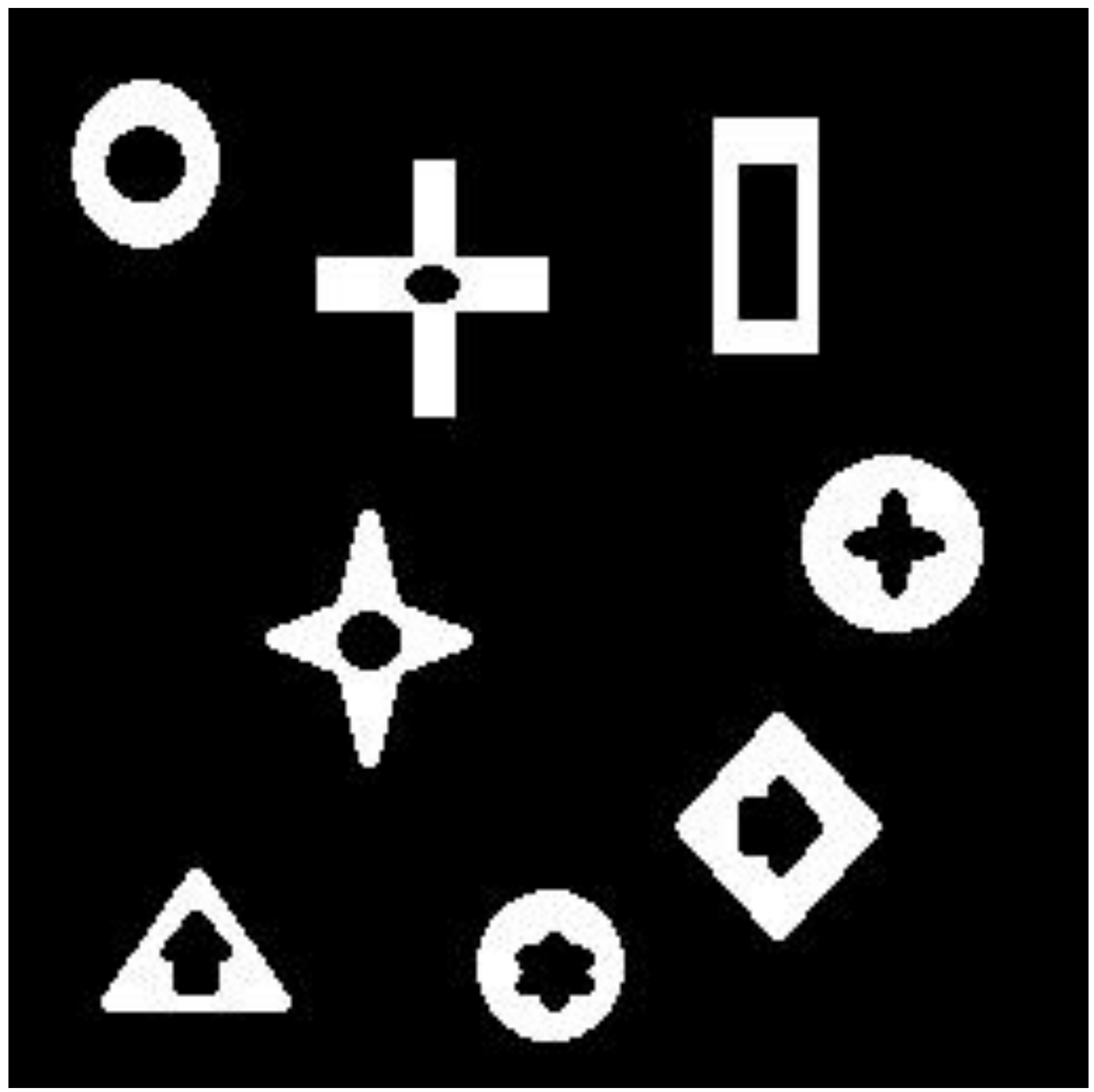
B

Conclusion

- straight forward
- easy to complement
- fast
- good performance

Problem 1(b) - Object Counting

- Please design an algorithm to count the number of objects in I1.



I1

Two-pass Connected-Component Labeling

- First pass

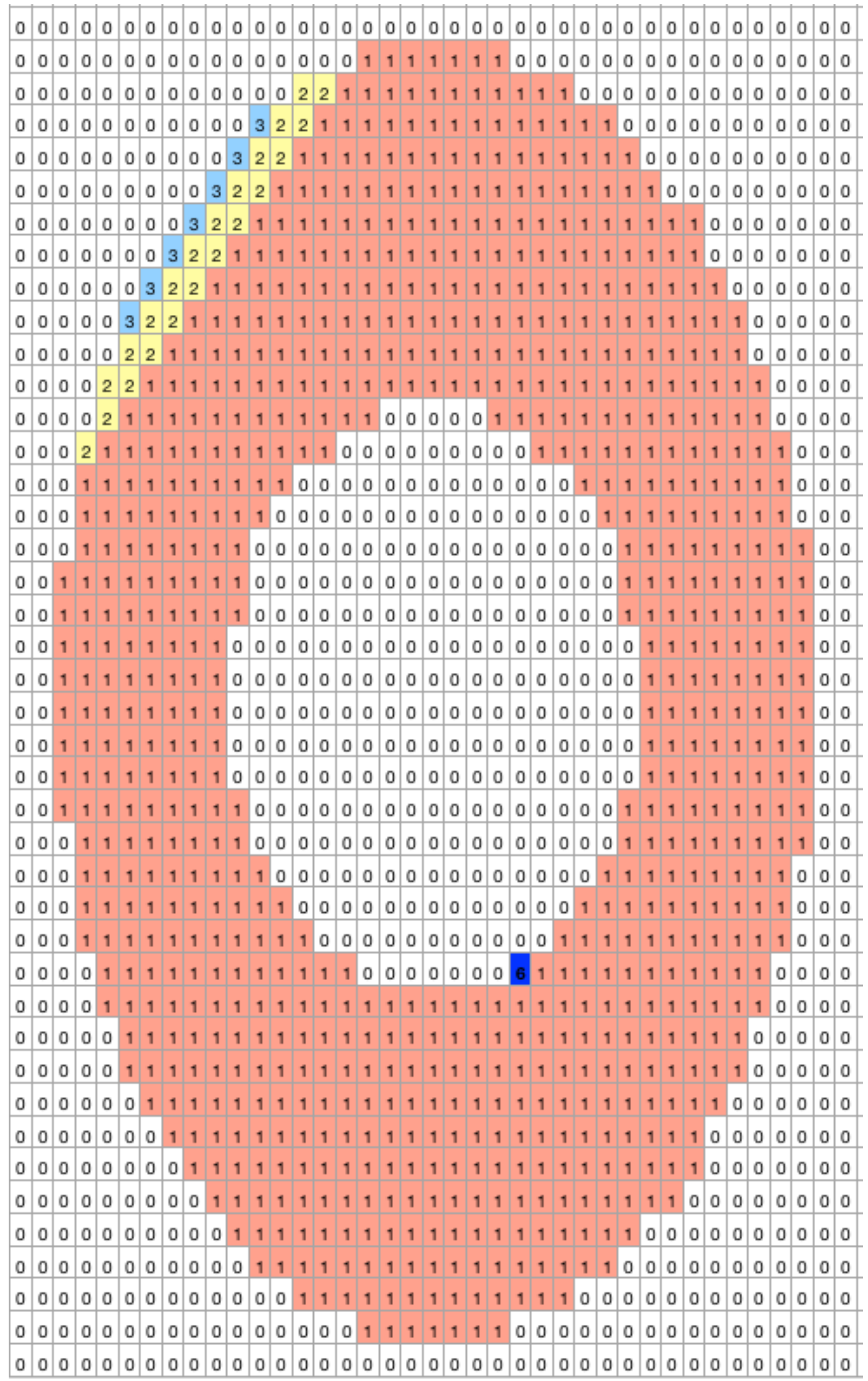
1. Iterate through each pixel of the image by column, then by row
2. If the pixel is not the background
   A. If there are no neighbors, uniquely label the current pixel
   B. Otherwise, find the neighbor with the smallest label and assign it to the current pixel, and store the equivalence between neighboring labels

- Second pass

1. Iterate through each pixel of the image by column, then by row
2. If the pixel is not the background
   A. Relabel the pixel with the lowest equivalent label

Two-pass Connected-Component Labeling

- First pass



Two-pass Connected-Component Labeling

- Second pass

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Result

Input:

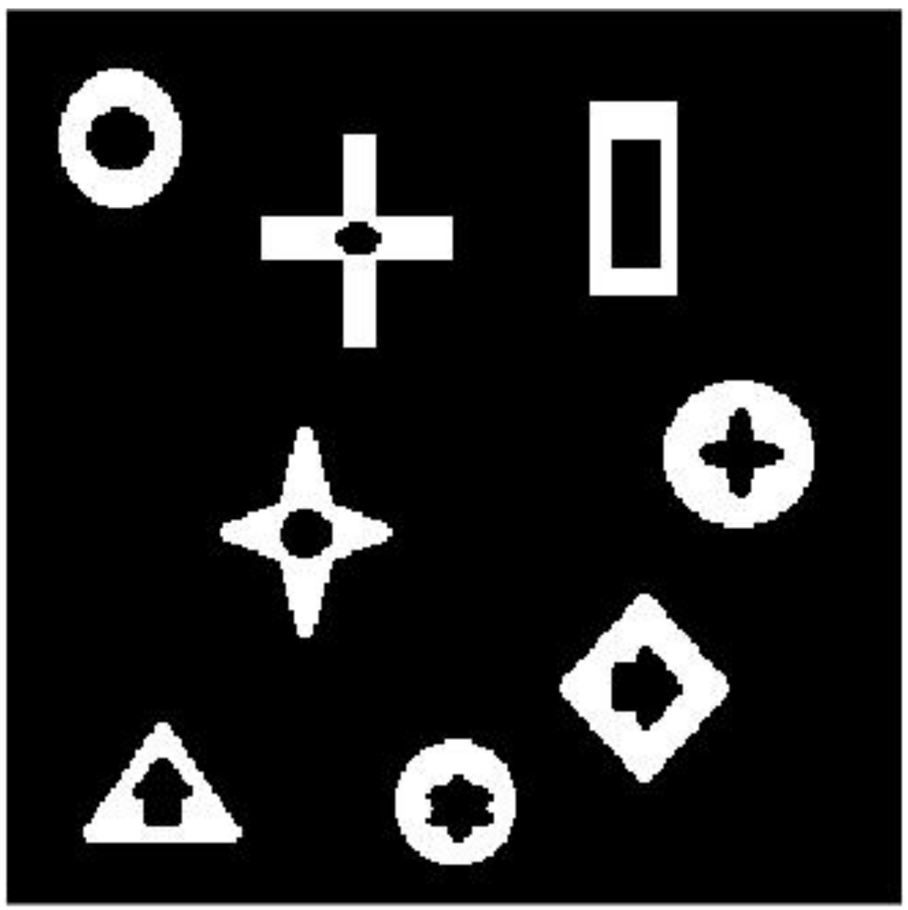


image I1

Output:

```
>> main
        Retrieving Image sample1.raw ...
        Retrieving Image sample2.raw ...
     8
```

Conclusion

- straight forward
- fast
- good performance

Problem 1(c) - Skeleton

- Perform skeletonizing on I1 and output the result as image S.

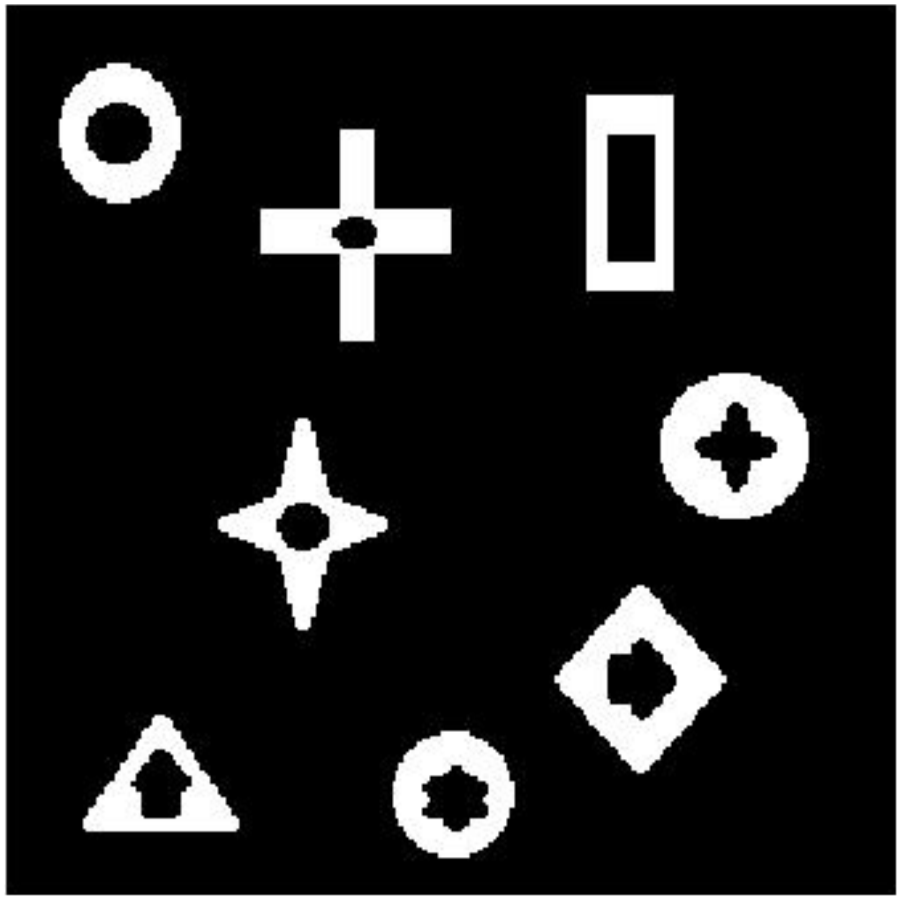


image I1

Skeleton

- Step I: Generate the conditional mask M(i ,j)
  - If M(i, j) == 1, it means (i, j) is a candidate for erase

TABLE 14.3-1. Shrink, Thin and Skeletonize Conditional Mark Patterns [$M = 1$ if hit]

Table	Bond	Pattern							
		0 0 1	1 0 0	0 0 0	0 0 0				
S	1	0 1 0	0 1 0	0 1 0	0 1 0				
		0 0 0	0 0 0	1 0 0	0 0 1				
		0 0 0	0 1 0	0 0 0	0 0 0				
S	2	0 1 1	0 1 0	1 1 0	0 1 0				
		0 0 0	0 0 0	0 0 0	0 1 0				
		0 0 1	0 1 1	1 1 0	1 0 0	0 0 0	0 0 0	0 0 0	0 0 0
S	3	0 1 1	0 1 0	0 1 0	1 1 0	1 1 0	0 1 0	0 1 0	0 1 1
		0 0 0	0 0 0	0 0 0	0 0 0	1 0 0	1 1 0	0 1 1	0 0 1
		0 1 0	0 1 0	0 0 0	0 0 0				
TK	4	0 1 1	1 1 0	1 1 0	0 1 1				
		0 0 0	0 0 0	0 1 0	0 1 0				
		0 0 1	1 1 1	1 0 0	0 0 0				
STK	4	0 1 1	0 1 0	1 1 0	0 1 0				
		0 0 1	0 0 0	1 0 0	1 1 1				

Bond: classification, narrow down the search space
Pattern: coded as an 8-bit symbol for a filter

$$\begin{bmatrix} X_3 & X_2 & X_1 \\ X_4 & X & X_0 \\ X_5 & X_6 & X_7 \end{bmatrix} \otimes \begin{bmatrix} 2^{-4} & 2^{-3} & 2^{-2} \\ 2^{-5} & 2^{0} & 2^{-1} \\ 2^{-6} & 2^{-7} & 2^{-8} \end{bmatrix}$$

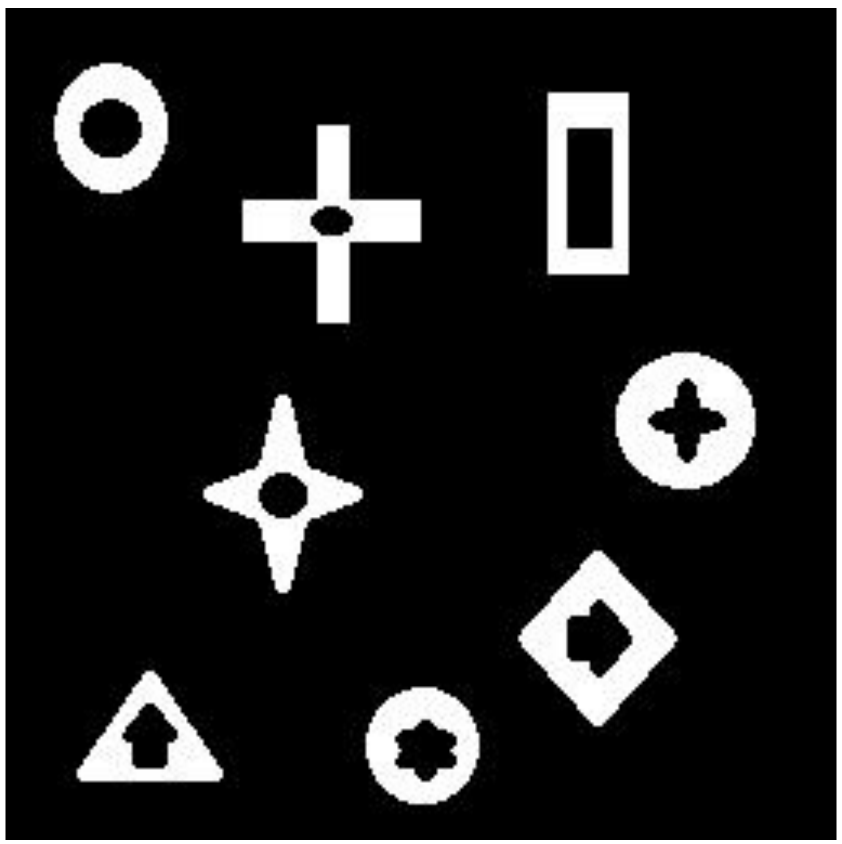
Skeleton

- Step II: Based on the conditional array, we decide whether to erase the candidate or not
  - If there's a hit, do nothing
  - Otherwise, erase it

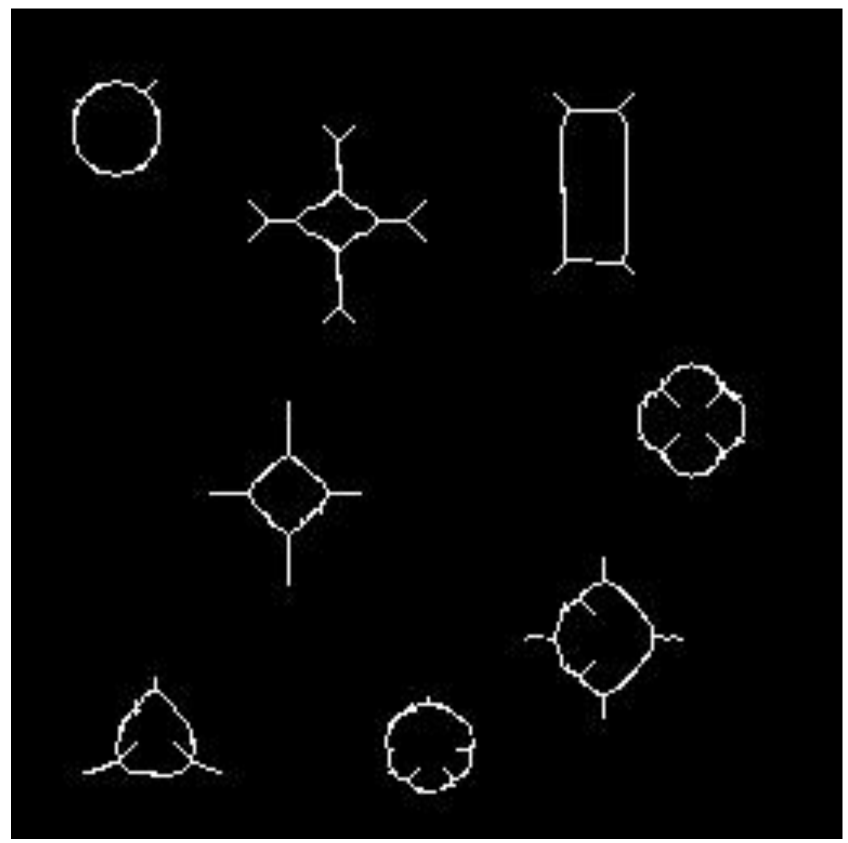
TABLE 14.3-3. Skeletonize Unconditional Mark Patterns
[$P(M, M_0, M_1, M_2, M_3, M_4, M_5, M_6, M_7) = 1$ if hit][a] $A \cup B \cup C = 1, \quad D = 0 \cup 1$

Pattern											
Spur											
0	0	0	0	0	0	0	0	*M*	*M*	0	0
0	*M*	0	0	*M*	0	0	*M*	0	0	*M*	0
0	0	*M*	*M*	0	0	0	0	0	0	0	0
Single 4-connection											
0	0	0	0	0	0	0	0	0	0	*M*	0
0	*M*	0	0	*M*	*M*	*M*	*M*	0	0	*M*	0
0	*M*	0	0	0	0	0	0	0	0	0	0
L corner											
0	*M*	0	0	*M*	0	0	0	0	0	0	0
0	*M*	*M*	*M*	*M*	0	0	*M*	*M*	*M*	*M*	0
0	0	0	0	0	0	0	M	0	0	*M*	0

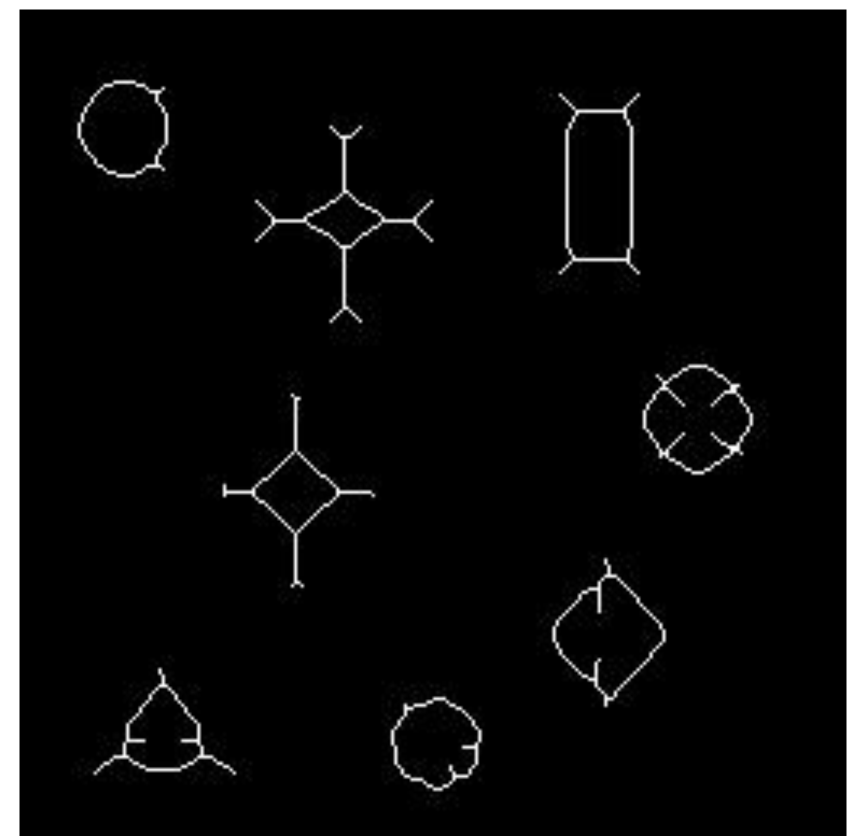
Result



input



output



output by matlab function

Conclusion

- straight forward
- hard to complement
- fast
- good performance

Problem 2(a) - Texture Analysis

- Perform Law’s method on I2 to segment the image into 3 different texture groups. Label the pixels of the same texture group with same intensity values.

image I2

Law's method

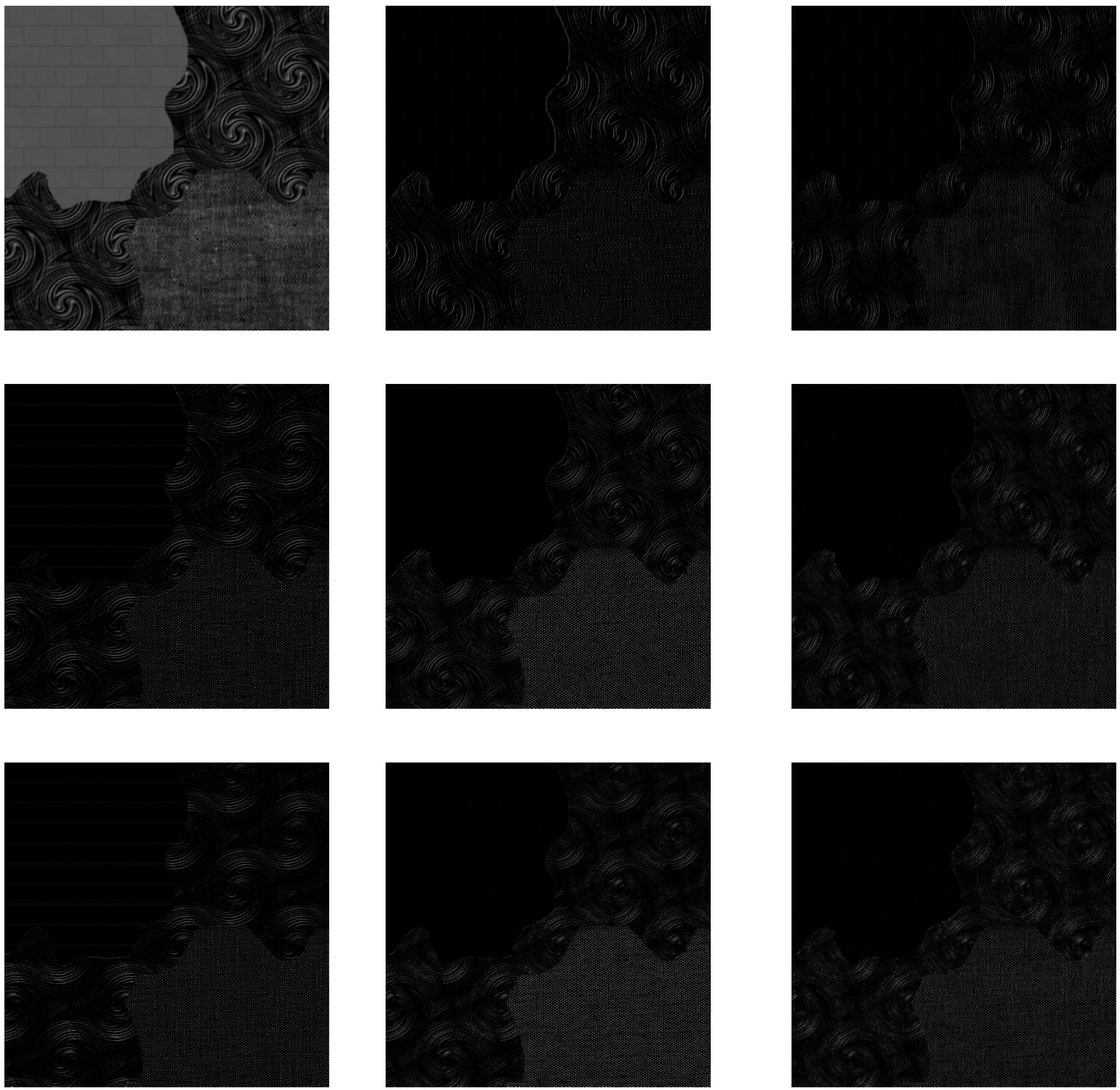
- Step I: Convolution
  - $M_k(i,j) = F(i,j) \otimes H_k(i,j)$

$$H_1(i,j) = \frac{1}{36}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad H_2(i,j) = \frac{1}{12}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad H_3(i,j) = \frac{1}{12}\begin{bmatrix} -1 & 2 & -1 \\ -2 & 4 & -2 \\ -1 & 2 & -1 \end{bmatrix}$$

$$H_4(i,j) = \frac{1}{12}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad H_5(i,j) = \frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \qquad H_6(i,j) = \frac{1}{4}\begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & -2 & 1 \end{bmatrix}$$

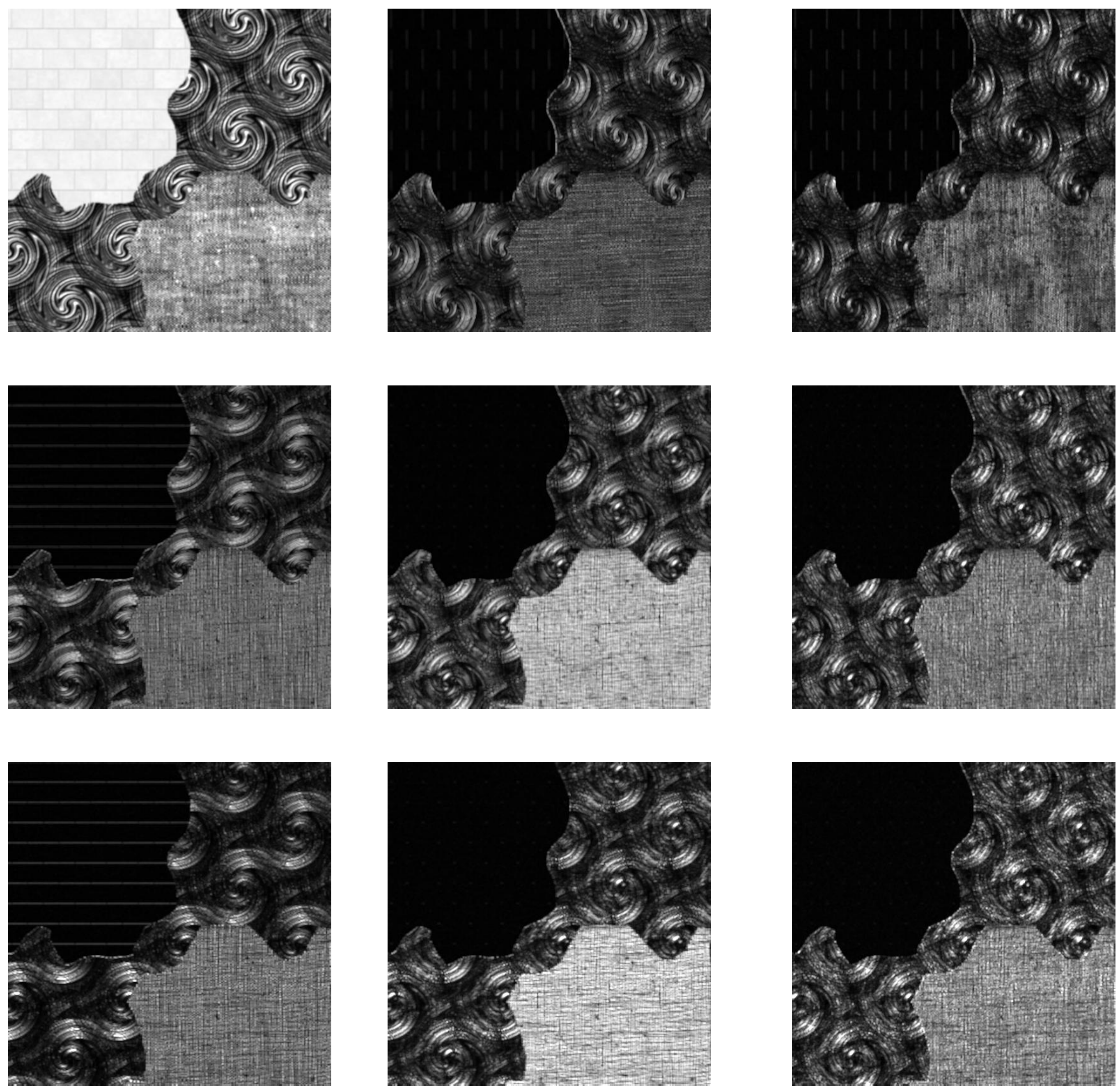
$$H_7(i,j) = \frac{1}{12}\begin{bmatrix} -1 & -2 & -1 \\ 2 & 4 & 2 \\ -1 & -2 & -1 \end{bmatrix} \qquad H_8(i,j) = \frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ 2 & 0 & -2 \\ -1 & 0 & 1 \end{bmatrix} \qquad H_9(i,j) = \frac{1}{4}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Law’s method

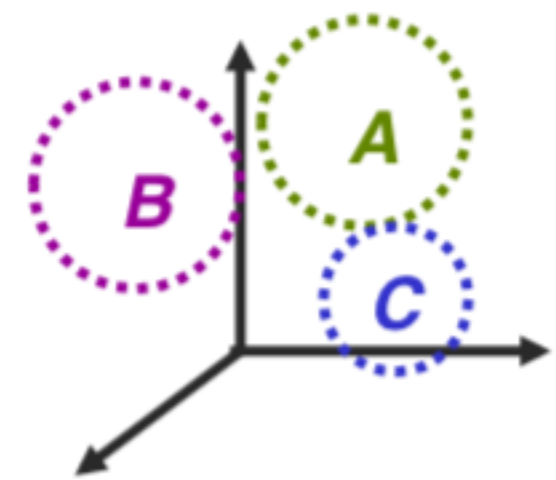


Law's method

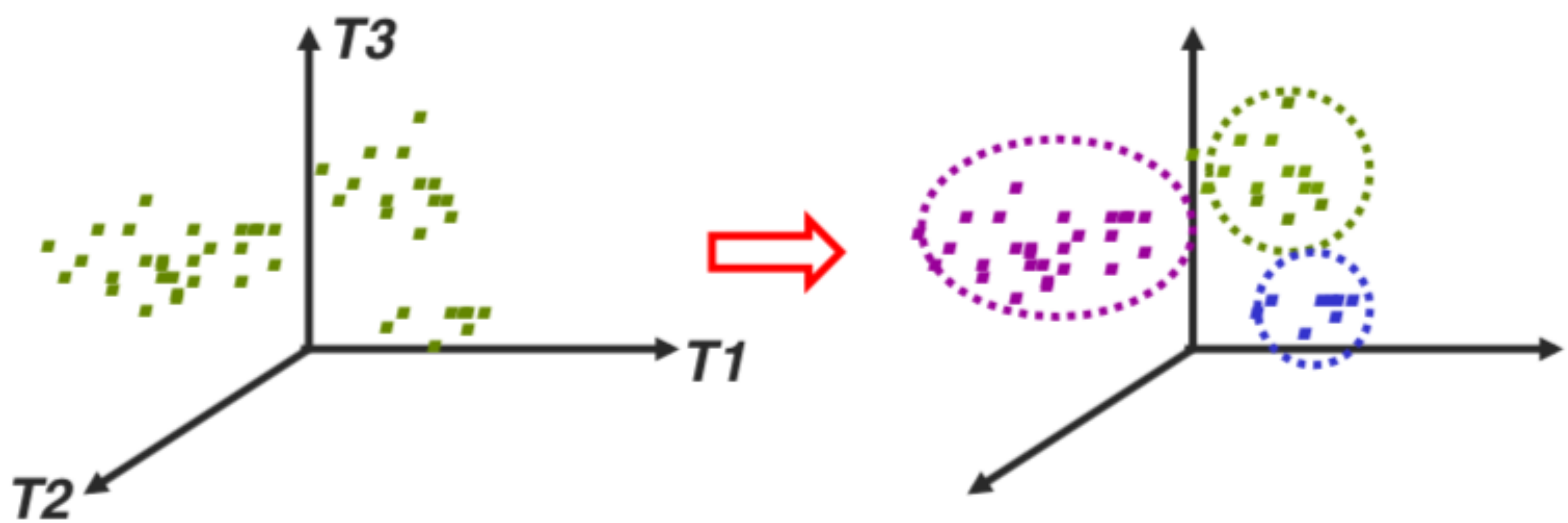
- Step II: Energy Computation
  - $T_k(i,j) = \sum \sum |M_k(i+m, j+n)|^2, (m,n) \in w$



Law’s method



- Given 9 feature sets, T1, T2, T3, …, T9
- Texture space -> 9 dimensional
- Use K-means algorithm to handle unsupervised classification problem



K-means algorithm

- Initialization
  - Select k vectors as the initial centroids
- Do the following iterations
  - Step I: Form k clusters using the NN rule
  - Step II: re-compute the centroid of each cluster

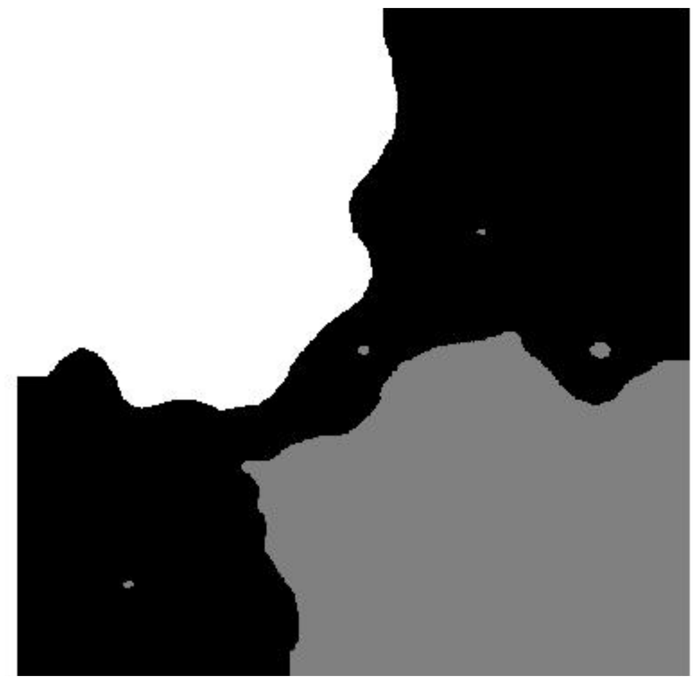
Experiment

- window size = 11


- window size = 31


Experiment

- window size = 41



- window size = 51

Conclusion

- not straight forward
- hard to complement
- time consuming

Problem 2(b) - Texture synthesis

- Generate another texture image by exchange the types of different texture patterns.

Image quilting by Efros & Freeman

- Step I: Tile the new texture image with blocks taken randomly from input texture.

Image quilting by Efros & Freeman

- Step II: For every location, search the input texture for the block that have the least SSD(Sum of Squared Difference) for overlap region and input texture.

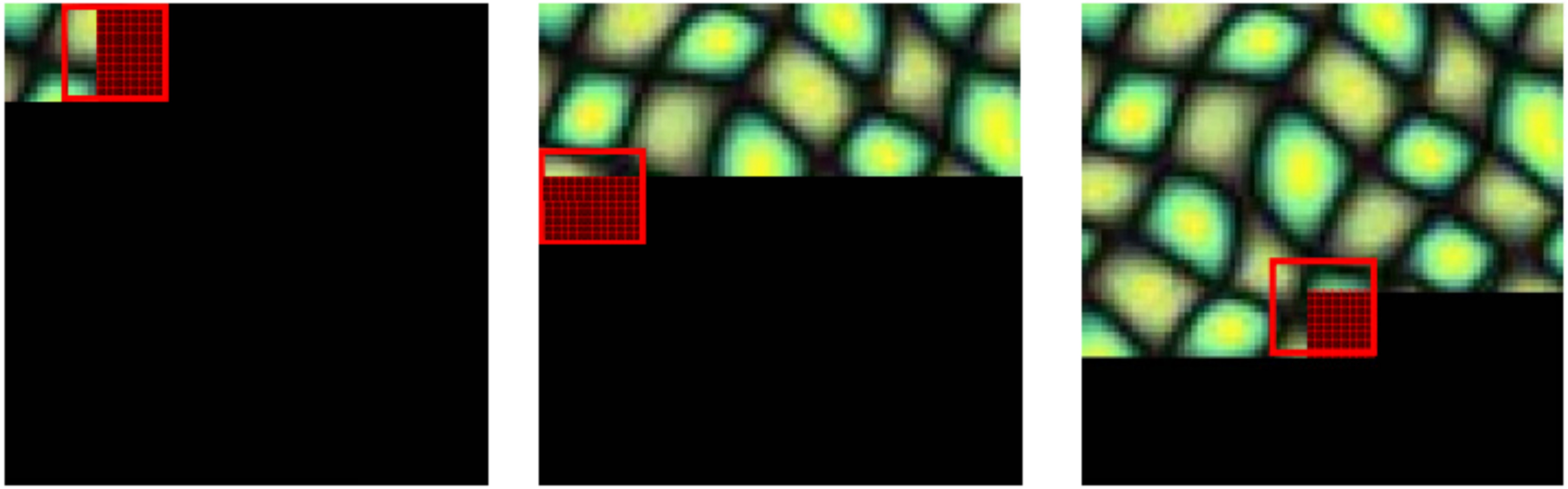
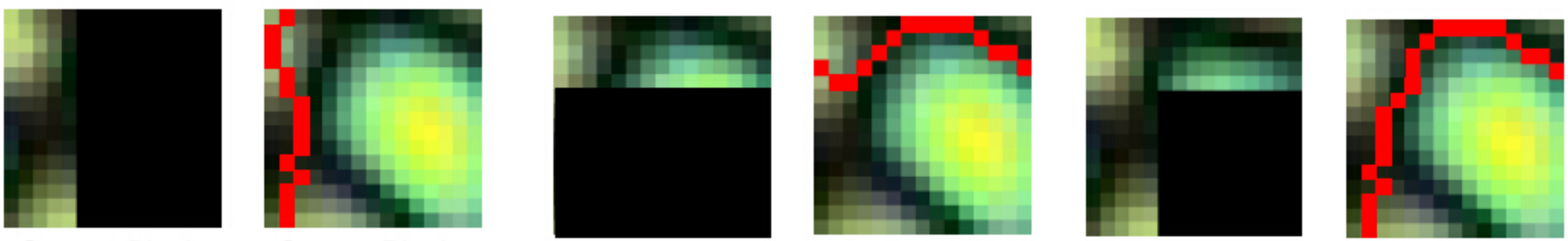


Image quilting by Efros & Freeman

- Step III: Compute the error surface between the newly chosen block and the old blocks at the overlap region. Find the minimum cost path along this surface and make that the boundary of the new block. Paste the block onto the texture.

$$E_{i,j} = e_{i,j} + min(E_{i-1,j-1}, E_{i-1,j}, E_{i-1,j+1})$$



Current Block   Source Block   Current Block   Source Block   Current Block   Source Block

Image quilting by Efros & Freeman

- Texture 1

input image

size: 212 * 212
window size: 21
overlap size: 5

synthesis image

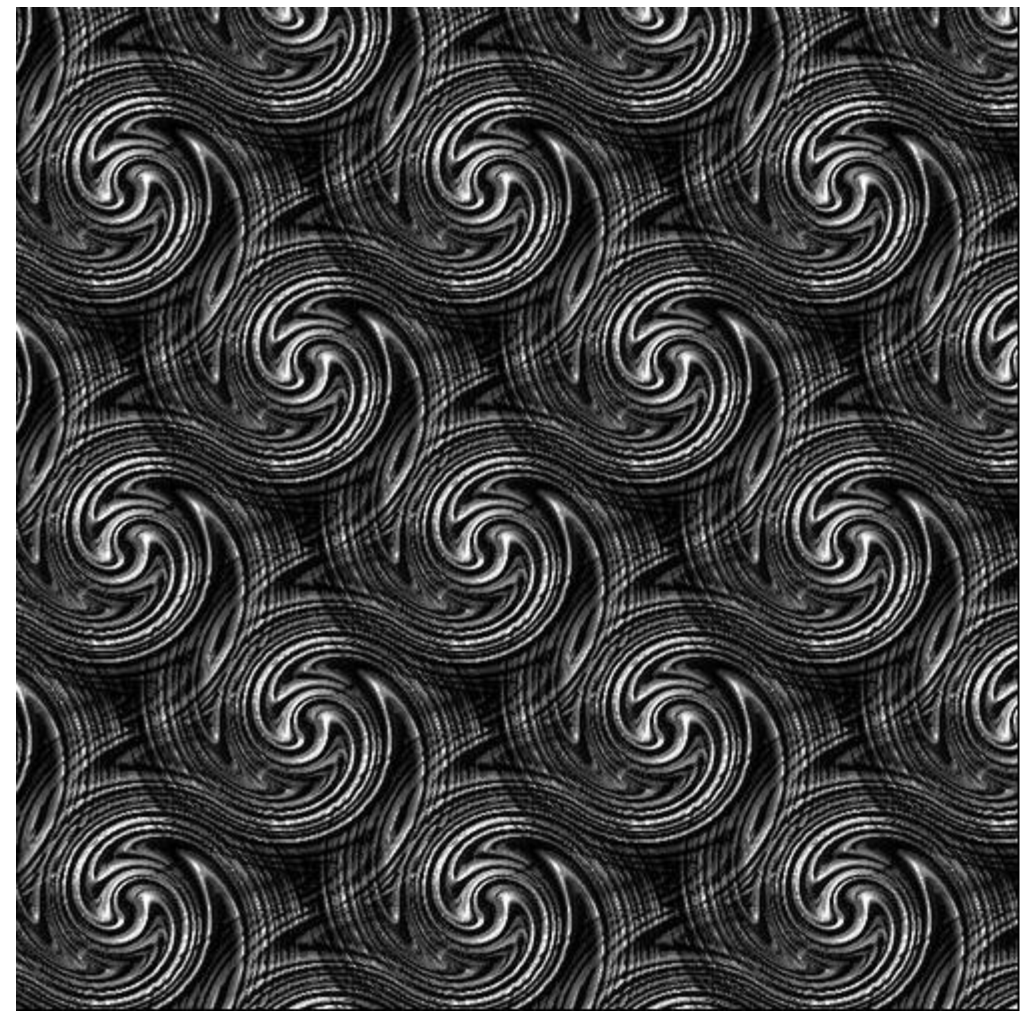


Image quilting by Efros & Freeman

- Texture 1

input image

size: 212 * 212
window size: 3
overlap size: 3

synthesis image

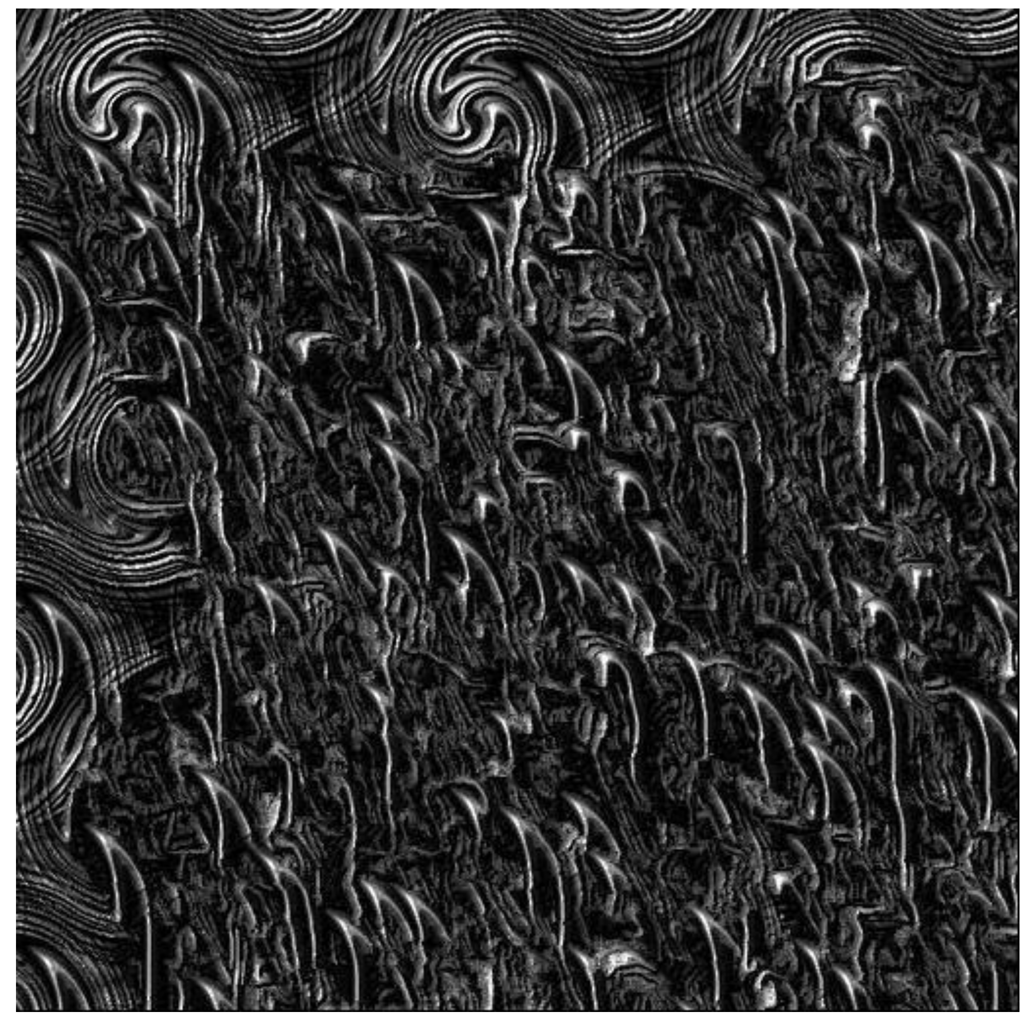
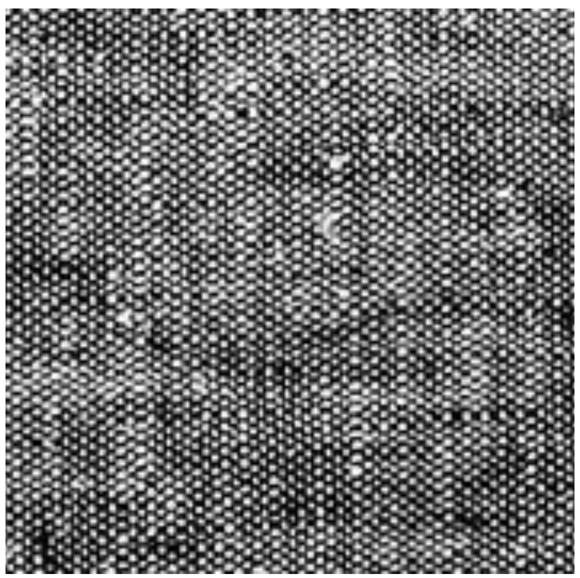


Image quilting by Efros & Freeman

- Texture 2

input image



size: 201 * 177
window size: 31
overlap size: 11

synthesis image

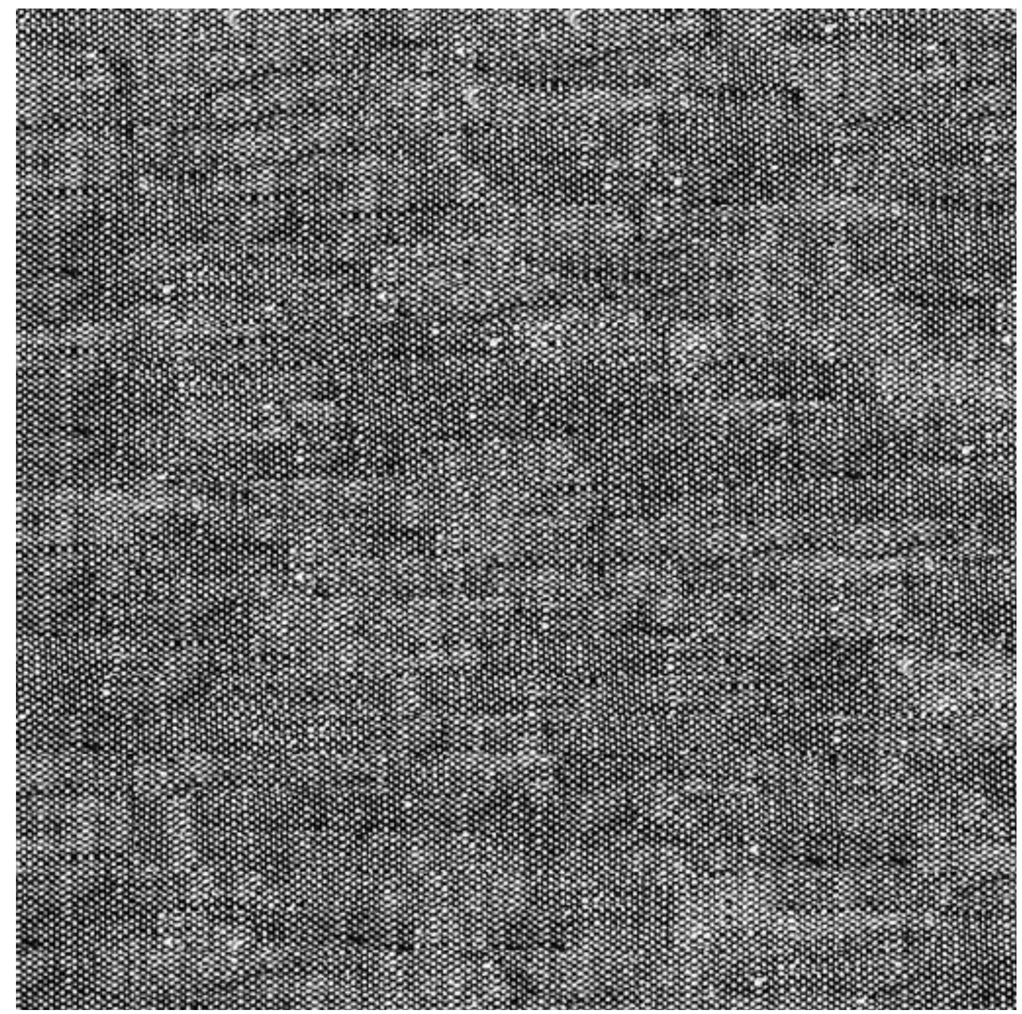
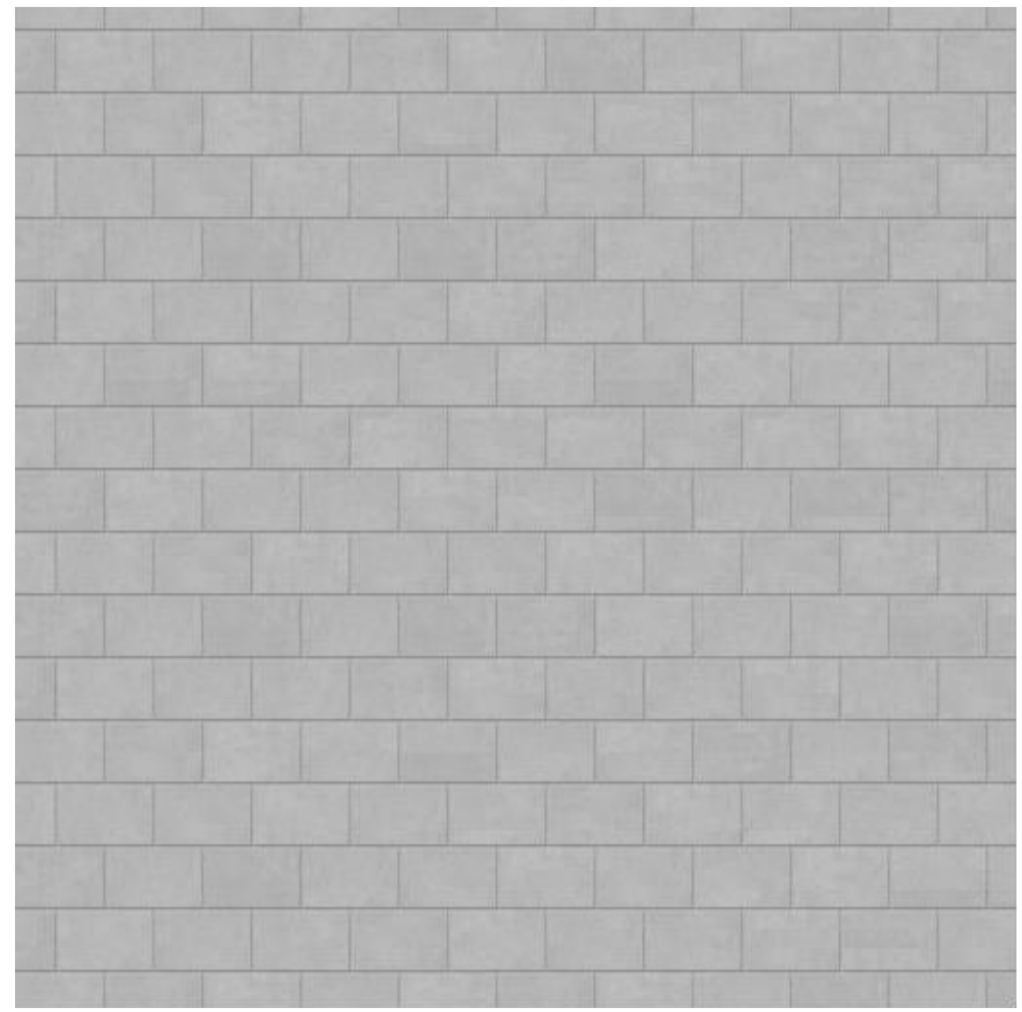


Image quilting by Efros & Freeman

- Texture 3

synthesis image



input image

size: 200 * 200
window size: 31
overlap size: 21

Exchange the type of different textures

input

output

Conclusion

- straight forward
- hard to implement
- time consuming
- good performance

Bonus - Dilation and Erosion

- Produce an image in Fig. 4 by adopting appropriate morphological processing

Fig. 3: sample 3.raw

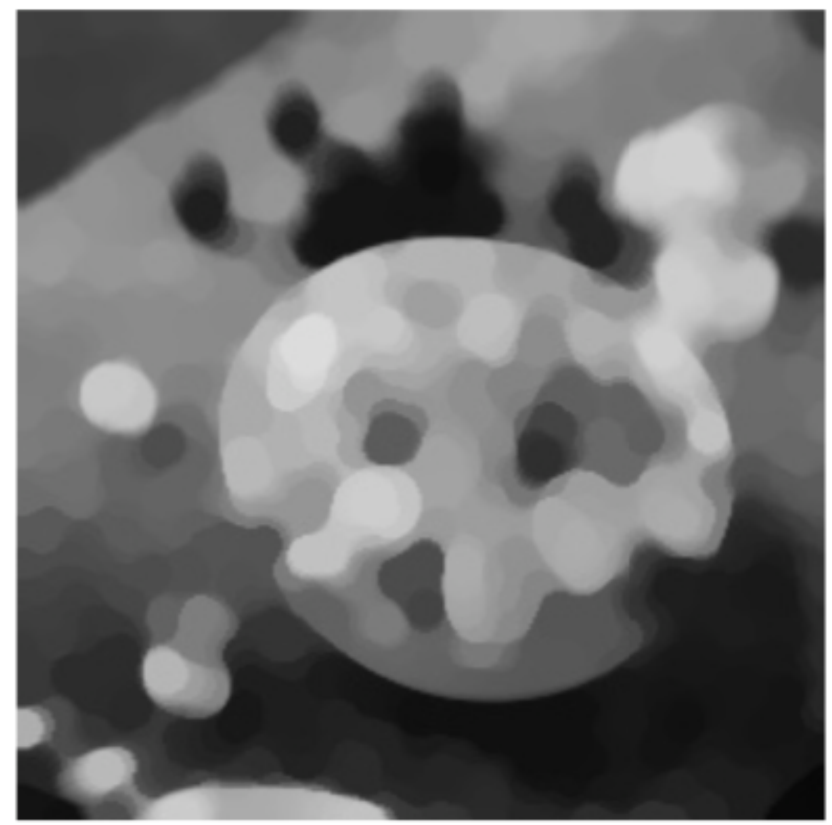


Fig. 4: The desired image

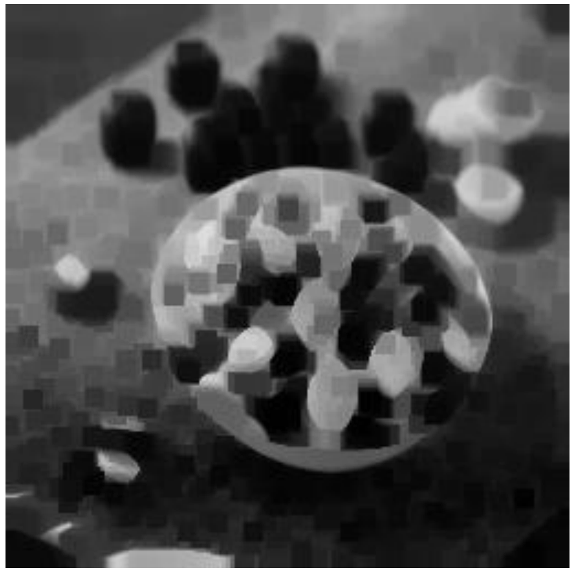
Dilation & Erosion

- Dilation
  - Maximum filter
- Erosion
  - Minimum filter

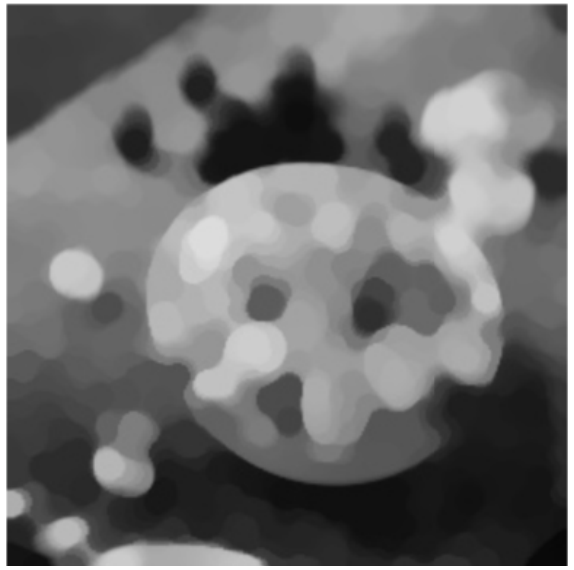
Experiment

- Step I: Erosion (window size = 9)

input

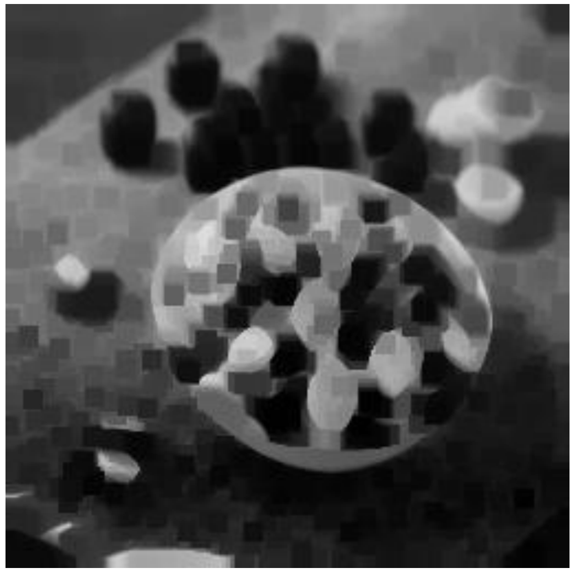


output



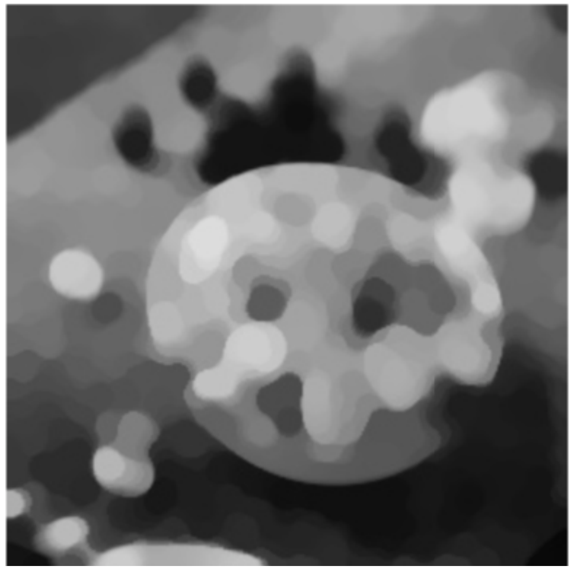
desired image

- Step II: Dilation (window size = 11)



input

output



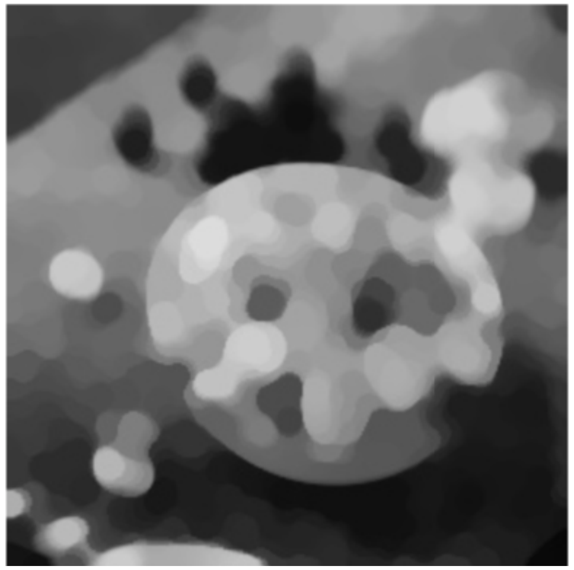
desired image

Experiment

- Step III: Median Filter (window size = 7)

input

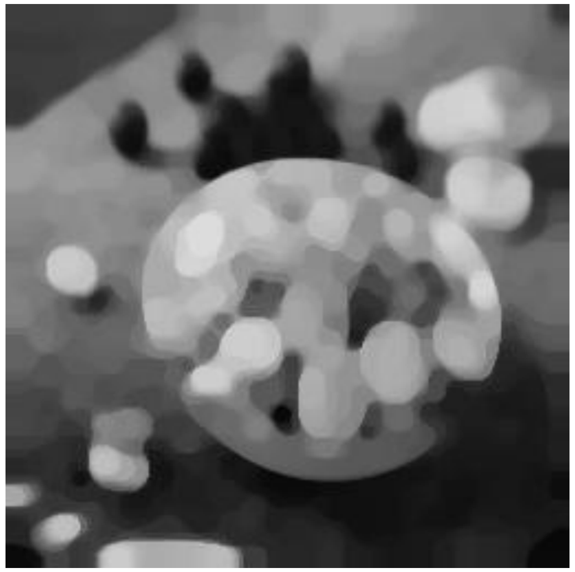
output



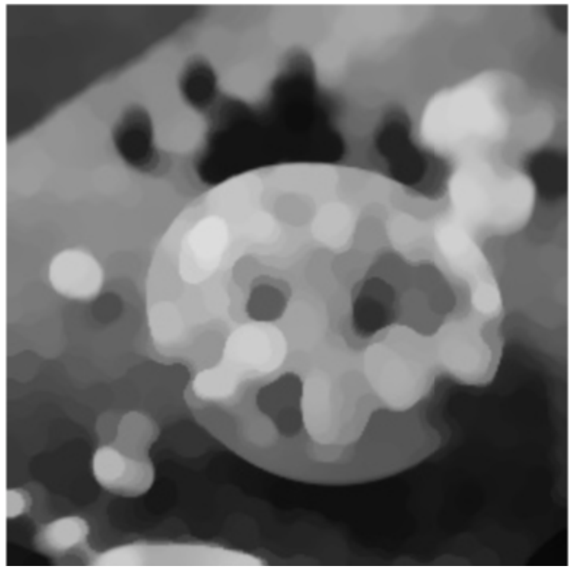
desired image

- Step IV: Dilation (window size = 3)

input



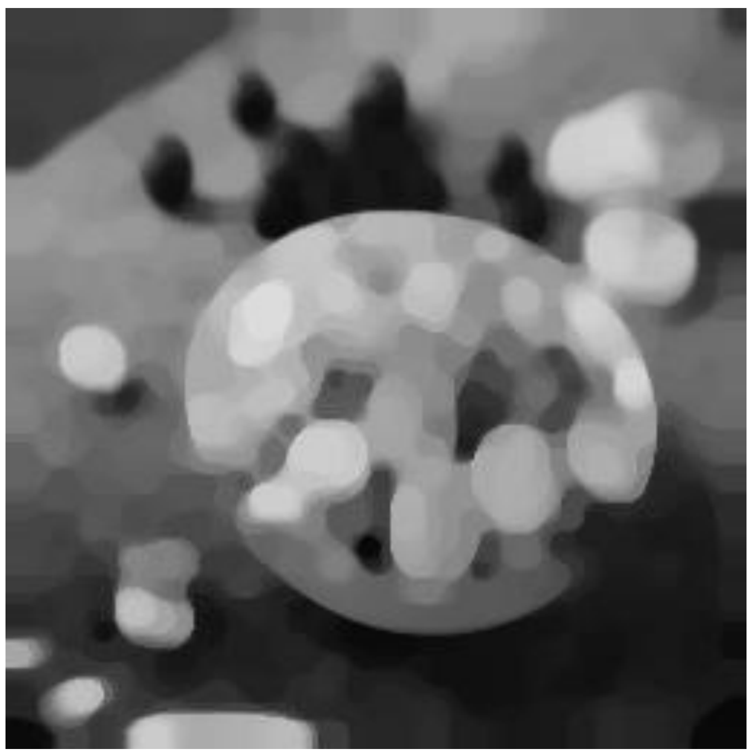
output



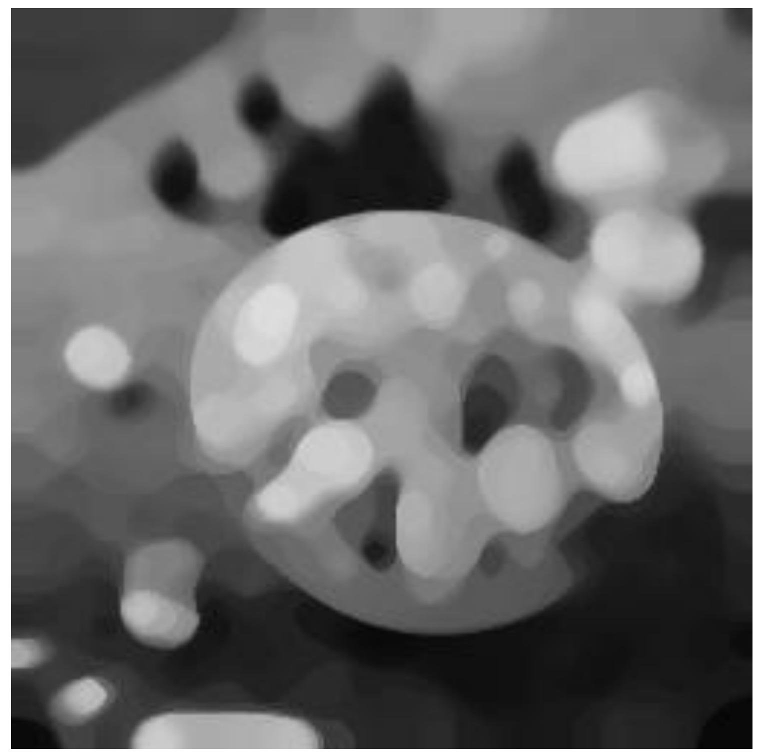
desired image

Experiment

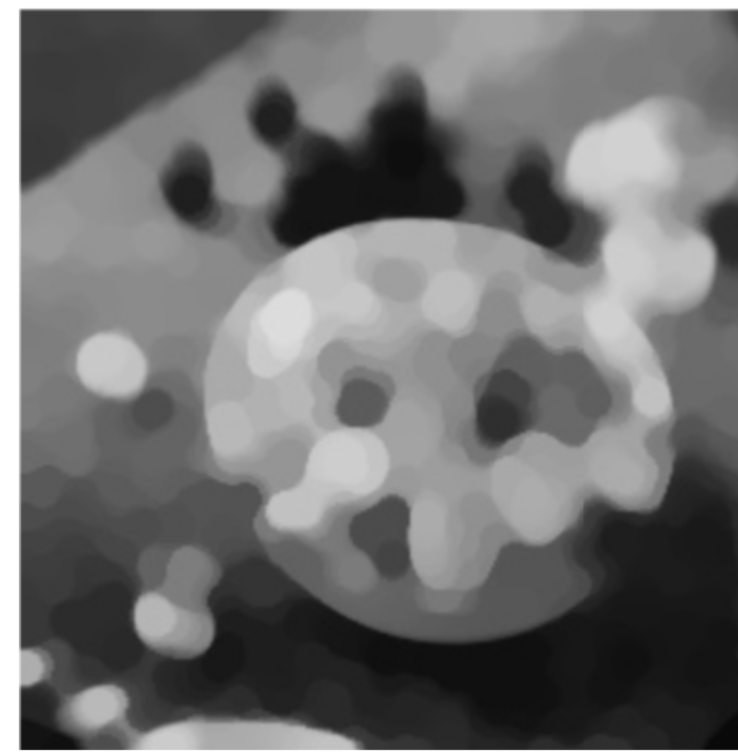
- Step V: Median Filter (window size = 11)



input



output



desired image

Conclusion

- not straight forward
- fast
- easy to complement