Computer Vision and Imaging (extended) [06-30241]
Assignment 1

9am, Monday, 8 March 2021

Submission deadline:	**9am (GMT), Monday, 15 March 2021**
Instructor:	Dr. Hyung Jin Chang Dr. Yixing Gao Dr. Mohan Sridharan Dr. Masoumeh Mansouri
Total marks:	100
Contribution to overall module mark:	25%
Submission Method:	This assignment must be submitted through Canvas.

Name:	Zhuang Ma
Username:	ZXM097
Study program:	MSc Advanced Computer Science

Part 1

Question 1.1 *What affect does this kernel have? (Consider both this image and larger example images)*

After the convolution operation, the image matrix of the central dotted area is as follows:

$$BeforeConvolution: \begin{bmatrix} 8 & 11 & 112 \\ 9 & 13 & 131 \\ 9 & 18 & 141 \end{bmatrix} \qquad AfterConvolution: \begin{bmatrix} -15 & -326 & 92 \\ -10 & -335 & 243 \\ -14 & -309 & 321 \end{bmatrix}$$

It can be seen from the above results, after the image is convoluted using this convolution kernel, *pixels with smaller values in the image will be filtered out, and pixels with larger values will be retained.*

In addition, the weight of this convolution kernel is *zero.* After using this convolution kernel to convoluted the actual image, *the effect of edge detection can be achieved.* The result *(Figure 1) is to convert the edge information of the image into white, while other information is black.*

Question 1.2 *What are their responses to uniform brightness? Are the filters isotropic or anisotropic and explain the terms. Which filter would you use for finding edges? What kind of edges would you use it to find?*

1. The first thing to be sure is that after the two convolution checks are convoluted on the image. The *uniform brightness will decrease.* Because in the convolution operation, the convolution kernel is multiplied by the value of the corresponding pixel, and then the sum is finally summed. Since the weights of the A and B convolution kernels are both zero, there may be pixels with the same value under the area of the convolution kernel size. Eventually, the value of the pixel is filtered to zero. At the same time, there may

be another situation for the pixels corresponding to the convolution kernel. That is, the value of pixels outside the center of the convolution kernel is greater than the pixel value of the center point. This may cause negative numbers in the result of convolution. *Therefore, the two convolution checks of A and B have a tendency to weaken the uniform brightness of the image.*

2. *The A filter kernel is isotropic, and the B filter kernel is anisotropic.* According to the definition of anisotropy. It refers to the property that all or part of the physical and chemical properties of an object varies with the direction, and isotropy is the opposite. Because the shape of the filtering kernel A is similar to the Gaussian kernel, but the difference is that the weight of the Gaussian kernel is one, and the weight of A is zero. Then A can be defined as a detector, which detects shapes similar to circles or spots. So no matter how the direction changes, *the object detected by A is always a circle or similar shape and will not change.* However, the filter kernel B and A are different. The B filter detects the edges in the vertical direction of the image. *If the direction changes, the edges detected by filter B will no longer be vertical.*

3. For finding the edge, it is more appropriate to use the *B filter kernel.* What this filter looks for is *the edge of the image in the Y direction.* Usually, we use the current pixel value to compare with the neighbouring pixel value. If the difference between the two is large, the pixel with the larger value is defined as the edge. The B filter is very suitable for this type of detection. *(See Figure 2)*

Question 1.3 *Is it possible to use the intrinsic and extrinsic camera properties to transform from a point in the camera 2D pixel coordinate system to a point in the world 3D coordinate system, and why?*

Usually in the case of a monocular camera, this kind of mapping does not exist. Because the result of Euclidean coordinates to homogeneous coordinates is not unique. According to the conversion relationship between pixel points and world coordinates, it can be seen that (the formula is as follows) where the parameter Z cannot be obtained. Therefore, the point in the world coordinate system cannot be obtained.

$$Z\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{d_x} & 0 & u_0 \\ 0 & \frac{1}{d_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_1 1 & r_1 2 & r_1 3 & t_1 \\ r_2 1 & r_2 2 & r_2 3 & t_2 \\ r_3 1 & r_3 2 & r_3 3 & t_3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{1}$$

Simplify formula (1):

$$Z\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f}{d_x} & 0 & u_0 & 0 \\ 0 & \frac{f}{d_y} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_1 1 & r_1 2 & r_1 3 & t_1 \\ r_2 1 & r_2 2 & r_2 3 & t_2 \\ r_3 1 & r_3 2 & r_3 3 & t_3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2}$$

In the above formula, d_x and d_y indicate the length of a pixel, that is, divide the size of the sensor by the number of pixels. For example, the sensor resolution of $2928.384um \times 2205.216um$ is 2592×1944, and the size of each pixel is about $1.12um$. f represents the focal length. According to similar triangles, the points in the world coordinate system and the points in the imaging plane coordinate system have the following relationship:

$$\frac{X_c}{x} = \frac{Y_c}{y} = \frac{Z_c}{f}$$

1. Transform the pixel coordinates to the camera coordinate system:

$$Z\begin{bmatrix}u\\v\\1\end{bmatrix}=\begin{bmatrix}\frac{f}{d_x}&0&u_0&0\\0&\frac{f}{d_y}&v_0&0\\0&0&1&0\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix}=K\begin{bmatrix}x\\y\\1\end{bmatrix} \tag{3}$$

After multiplying both sides by the inverse of K, it is derived:

$$\begin{bmatrix}x\\y\\z\end{bmatrix}=K^{-1}\begin{bmatrix}u\\v\\1\end{bmatrix} \tag{4}$$

2. Transform from the camera coordinate system to the world coordinate system:

$$\begin{bmatrix}X_c\\Y_c\\Z_c\end{bmatrix}=R\begin{bmatrix}X\\Y\\Z\end{bmatrix}+t \tag{5}$$

Multiplying the equation by R^{-1} :

$$\begin{bmatrix}X\\Y\\Z\end{bmatrix}=\begin{bmatrix}X_c\\Y_c\\Z_c\end{bmatrix}R^{-1}-tR^{-1}=Z\begin{bmatrix}x\\y\\1\end{bmatrix}R^{-1}-tR^{-1} \tag{6}$$

From the above derivation, it is possible to find an accurate point in the world coordinate system when the value in the camera coordinate system is equal to Z_c and the plane coordinate Z.

Question 1.4 *Taking a 1cm^2 6 MegaPixel imaging array as an example, what is the limiting factor to increasing the number of sensors to increase the resolution of the image captured? The size of the pixel is its limiting factor.* The resolution of an image is composed of the number of pixels in the horizontal and vertical directions. Therefore, the essence of improving image resolution is to increase the number of pixels in the horizontal and vertical directions of the image. As for the sensor, the larger the area, the larger the image. Under the same conditions, it can record more image details, the interference between pixels is also small, and the image quality is better. In this question, if you increase the number of sensors and let them capture the same image, the distance between the sensors must be the same as the distance between the pixels inside the sensor, which is about 4 microns. Such conditions are obviously not easy to achieve, so adding sensors in the same area is a solution. If the number of sensors is increased in the same area, the surface base of the sensors will be reduced. In other words, the size of the pixel will be reduced. The more the sensor increases, the smaller the pixel size. The size of pixels cannot be reduced forever, so this is a limiting factor.

Part 2

For this task, you will have to submit the following file:

- code **username_assignment1_part2.m**

Question 2.1.1 *Discuss your observations.*
See detail in Figure 3

1. Gaussian noise
   Gaussian noise is the result of adding a Gaussian to each pixel of the original image. In this topic, it is obtained by adding a Gaussian function with a mean value of 0.3 and a variance of 0.5 to each pixel of the original image. The result is obvious, the influence of noise on the original image is great.

2. Salt and pepper noise
   Different from Gaussian noise, salt and pepper noise changes the pixel value of the original image to 0 (black) or 255 (white) to achieve the purpose of noise. The noise density used in this question is 0.075, which means that the pixels affected by the original image account for approximately 7.5 percent of the total pixels. The result is shown in Figure 3. In this problem, the salt and pepper noise has a small effect on the original image.
3. Speckle noise
   Similar to Gaussian noise, all pixels of the original image are also randomly sampled. But the difference is that Gaussian noise uses a normal distribution, while speckle-noise samples a uniform distribution. In this question, uniform distribution with a mean value of 0 and a variance of 0.05 is used to add noise. The result is shown in Figure 3. The value of some pixels is reduced, but it has little effect on the artificial recognition of the original image.

Question 2.1.2 *How well did they work and why? Discuss your observations.*
See detail in Figure 4

1. Minimum Filter
   First, determine the size of the convolution kernel, and then use a 5×5 convolution kernel in this question. The value of each element of the convolution kernel should be 1. After determining the convolution kernel, use the convolution kernel to perform convolution operations on the image. Unlike ordinary convolution operations, the pixel value of the pixel at the current position is no longer the result of the convolution weighted summation, but the minimum value of the convolution result at the current position is selected as the new pixel value of the pixel of the current original image.

   (a) For Gaussian noise
   Because most of the images after adding noise are noise points. Therefore, when the minimum filter is used, most of the new pixel value of the current pixel will be black. The filtering effect is not ideal.

   (b) For Salt and Pepper noise
   After adding salt and pepper noise to the original image, the range of noise influence is not large. Therefore, when the minimum filter is used, the filter will expand the range where the noise value is 0 (black). The expanded range is the size of the filter kernel, which is 5×5.

2. Maximum Filter
   For the maximum filter, the process is the same as that of the minimum value filter. But the difference is that the maximum value of the pixel value selected by the maximum filter is the maximum value in the current filter kernel convolution result as the new pixel value of the current pixel.

   (a) For Gaussian noise
   Contrary to the result of minimal filtering. Since Gaussian noise is white noise, the pixel value of the noise is relatively large, and the noise almost covers the entire image. Therefore, the result of using maximum filtering is very unsatisfactory.

   (b) For Salt and Pepper noise
   Contrary to the result of minimal filtering. Maximum filtering will expand the range of the noise pixel value of 255 (white).

3. Mean Filter
   For mean filtering, the convolution kernel used is the same as the maximum/minimum

filtering. The value of each element of the filter kernel is 1. In the convolution process, the value of the pixel corresponding to the convolution kernel needs to be obtained first. Then the value of the entire convolution kernel is averaged. Finally, the average is used as the new pixel value of the current pixel of the original image.

(a) For Gaussian noise
Mean filtering is not very effective for removing Gaussian noise. Since noise almost covers the entire image, the size of the mean is easily affected by noise. For example, the value of a certain pixel in the convolution kernel is much larger than other pixels. Then the final value of the pixel will also be affected.

(b) For Salt and Pepper noise
For salt and pepper noise, average filtering is relatively effective. But there are still problems. The disadvantage is that the two pixels with larger pixel values in the original image are averaged. This makes the filtered image becomes blurred, and noise removal is not completed.

4. Median Filter
For median filtering, its convolution operation is similar to mean filtering. But the difference is that the new pixel value of the current pixel of the original image is not the average value of the convolution result. Instead, sort all the results of the current point convolution in ascending order. Select the value of the center position of the array after sorting (if the number of the ascending array is an even number, select the average value of the center position of the array) as the new pixel value of the current pixel of the original image.

(a) For Salt and Pepper noise
Using a median filter to filter salt and pepper noise is very effective and effective. This is because the nature of the median filtering does not generate new pixel values. Its essence is to fill the noisy pixels with the pixel values of other positions in the image. Thus, the purpose of filtering is achieved.

(b) For Gaussian noise
For Gaussian noise, even though the nature of median filtering does not generate new pixel values, it fills the noise pixels with pixel values from other locations in the image. However, due to Gaussian noise, almost the entire image is covered. The pixel value obtained by the filter is very likely to be a noise point, which ultimately leads to an unsatisfactory filtering effect.

Question 2.2 *Discuss your observations.*
See detail in Figure 5
Except for the noise points, all the feature points in the noise image are matched with the feature points of the denoised image.

Question 2.3 *Perform image registration and display your result.*

See detail in Figure 6

Part 3

For this task, you will have to submit the following file:

- code **username_assignment1_part3.m**

Question 3.1 *Compare your results to the results of using the OTSU's method and comment on the reasons for the observed differences in results. If other methods produce worse results than the default OTSU's method, then please explain why you think this is.*

Please see Table 1 for specific operations and parameters.
In this question, I used the OTSU algorithm as the basis, and added pre-processing and post-processing. The result is shown in Figure 7. It can be clearly seen from the results that the default OTSU algorithm is noisy when segmenting images. And some of the adherent cells were not separated.

First of all, I considered how to remove noise. I chose among the commonly used filters, and the final result shows that the median filter is more effective. This is taking into account that the color of the cells marked with GFP is not uniform in the picture. If Gaussian filtering or mean filtering is used, it may cause some cells with lighter colors to be filtered out, which may eventually lead to under-segmentation problem.

Secondly, because the filter is used to remove noise, no matter what kind of filter is used, it may filter out non-noise points and noise points at the same time. In order to solve such problems, I consider using morphological operations to restore the problem of filtering out non-noise points. In this question, the closed operation is mainly used, that is, first expansion and then corrosion.

Finally, I considered the problems of occlusion and adhesion before the cells. Such problems can usually be solved using segmentation algorithms. Considering the complexity and performance of the algorithm implementation, I finally chose to use the watershed algorithm as a segmentation method for segmenting adherent cells. The reason is that this algorithm is easy to implement and runs faster, but its accuracy is lower than other segmentation algorithms, and it is more dependent on the previous pre-processing.

Question 3.2 *Explain the purpose and the outcome(s) of any operation that you have used.*

In this question, I implemented two versions of the split method, and the result is shown in Figure 8.

The two versions of segmentation are based on threshold segmentation, using filters to remove noise, using edge detection algorithms to extract image edges, using morphological operations to restore non-noise points, and finally using watershed algorithm for image segmentation. The difference between the two versions lies in the use of filters and the choice of different threshold processing methods.

1. For version 1
   After reading the image, extract all the values of the green channel. Since the object to be segmented is green, and other colors may affect the effect of segmentation, the green channel is selected to provide data for pre-processing, instead of providing data for pre-processing by converting RGB images to grayscale images.

   The green channel of the image is filtered, and the median filter is used in version 1. Because the color of the cells is not uniform, the use of other filters may remove a large number of non-noise points, so the median filter is used.

   The image data is pre-segmented by means of threshold segmentation. Threshold segmentation was performed twice in succession in version 1. The threshold segmentation in this version is different from the traditional one. After calculating the threshold, the image is thresholded. The pixel value less than or equal to the threshold value is set to a value equal to the threshold value. In this way, two consecutive threshold processing will easily separate the edge points of the image. After the first threshold processing,

the background and foreground of the image will be separated, and the overall grayscale of the image will increase. After the second threshold processing is continued, the gap between the cells will also become the same color as the background, in other words, the adjacent cells will be separated. After the above methods are processed, the edge of each cell will be found very accurately when performing edge extraction.

Use the canny edge detection algorithm to extract the edge of the processed image. But often it is not possible to completely extract all the edges of each cell, so I use morphological means to patch the edge images. Some edges that cannot form a closed circle will be stitched. This operation is handled by dilation. First, use line-shaped filters to create two direction filter kernel, one is horizontal and the other is vertical. These two filtering kernels are used to perform dilation operations on the edge image respectively. This is because the edge image extracted by canny is often composed of a one-pixel connection. If other shapes of filter kernel are used, noise points will be generated.

After the expansion process, the edge image has basically formed a closed circle. Then you can use the filling method to change the edge image into an image similar to that processed by the OTSU algorithm. But the difference is that in the filled edge image, the boundary of each cell is more obvious.

Although the boundary of the filled edge image is more obvious, there are still some cases where the edge is blurred or the edge between two cells overlaps. When the overlapping parts of the edges are not serious, the division between cells can be achieved by using the watershed algorithm.

2. For version 2
   The edge detection and segmentation algorithms of version 2 and version 1 are the same, the difference lies in filters, threshold processing and morphological operations.

   For the filter, version 2 is different from version 1. In this version, the mean filter is used. Also consider the uneven color of the cells, and our goal is to separate each cell. Therefore, the average filter is used to make the color of the gap between the cells the same as the background color as much as possible, that is, to minimize the difference in pixel value between the two.

   When thresholding the filtered image, version 2 is different from version 1. In this version, the value is thresholded once. Similar to version 1, the threshold processing method is different from the traditional one. This threshold processing is to subtract the threshold from the entire image to achieve a similar effect to version 1. After this threshold processing, the edge of each cell in the image becomes obvious, and the color of the gap between the cells is the same as the background.

   In terms of morphological operations, version 2 is also different from version 1. Compared with version 1, this version performs morphological operations again before performing the watershed algorithm. Since the first operation is to expand the image, the effect of the expansion is too obvious. Therefore, before segmentation, the boundary of each cell should be as clear as possible. Therefore, it is necessary to use corrosion operations to achieve this purpose.

Question 3.3.1 *Area (in pixels) of each cell*
Please take a look at the each_cells_area.mat file in the output folder in the compressed package.

Question 3.3.2 *Mean brightness (in green channel) of each cell*

Please take a look at the each_brightness_mean.mat file in the output folder in the compressed package.

Question 3.3.3 *Mean and standard deviation for the area and brightness for all the cells in the image*
Place take a look at the mean_area_brightness.mat file and std_area_brightness.mat file in the output folder in the compressed package. Among them, mean_area_brightness stores the average value of area and brightness, and std_area_brightness stores the standard deviation of area and brightness.

Question 3.4 *Repeat Questions 3.1, 3.2, and 3.3 on the image of E-coli.*
Your answer

1. Question 3.4.1

   In this question, the same method as 3.1 is used to segment the image, but the effect is not ideal. It is slightly worse than the default OTSU algorithm. The main reasons are as follows:

   (a) Filter:
   Because only GFP-labeled cells are divided, and the cell size is relatively small. At the same time, there are many target cells tightly connected to each other. This is very unfavorable for using filters to filter noise. The filter can easily merge several cells into one, or filter small or closely connected cells. Therefore, the segmentation task cannot be realized.

   (b) Morphological operation:
   No matter what kind of kernel is used, no matter the size of the kernel. Using the corrosion operation will corrode most of the cells, while using the expansion operation will merge several cells together. However, if the open or close operation is used, the result is quite different from the default OTSU algorithm. Sometimes the default OTSU can separate connected cells, and the open or close operation does not achieve the desired effect.

2. Question 3.4.2
   This question is the same as in question 3.2 version 1. The difference is that the steps of using filters to denoise the image are cancelled in this question. The reason is the same as the above question. It is easy to filter out cells with smaller sizes and smaller gradients using filters. And it will merge several closely connected cells into one large cell. Such large cells cannot be separated by morphological manipulation and other means. The watershed algorithm has better segmentation for slightly connected objects. But for closely connected cells, it is very difficult to separate them using the watershed algorithm.

3. Question 3.4.3

   (a) Question 3.4.3.1

   Please take a look at the Ecoli_each_cells_area.mat file in the output folder in the compressed package.

   (b) Question 3.4.3.2

   Please take a look at the Ecoli_each_brightness_mean.mat file in the output folder in the compressed package.

(c) Question 3.4.3.3

Place take a look at the Ecoli_mean_area_brightness.mat file and Ecoli_std_area_brightness.mat file in the output folder in the compressed package. Among them, mean_area_brightness stores the average value of area and brightness, and std_area_brightness stores the standard deviation of area and brightness.

Appendix

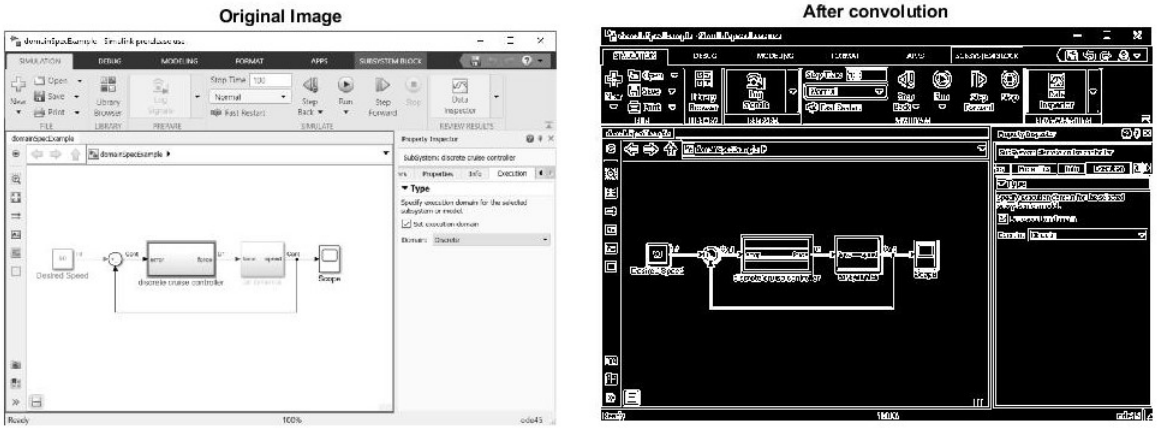


Figure 1: Example of Question 1.1

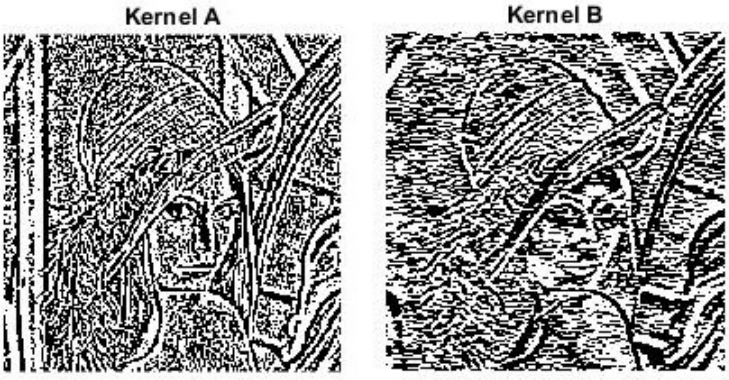


Figure 2: Example of Question 1.2

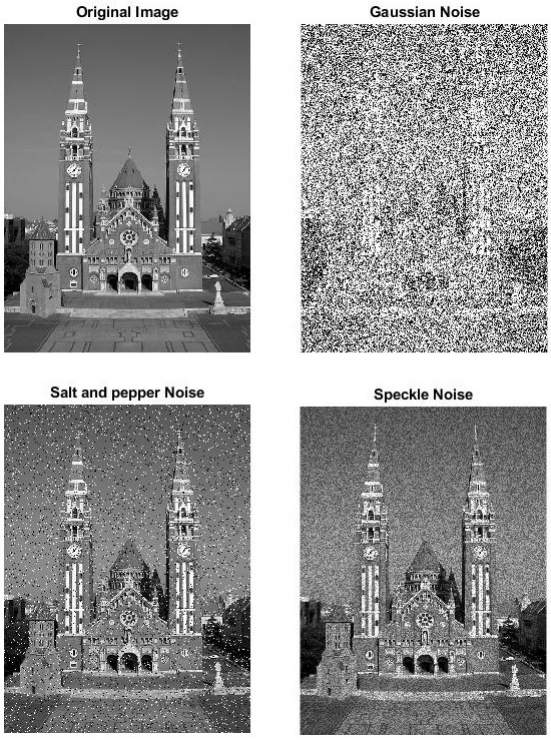


Figure 3: Result of 2.1.1

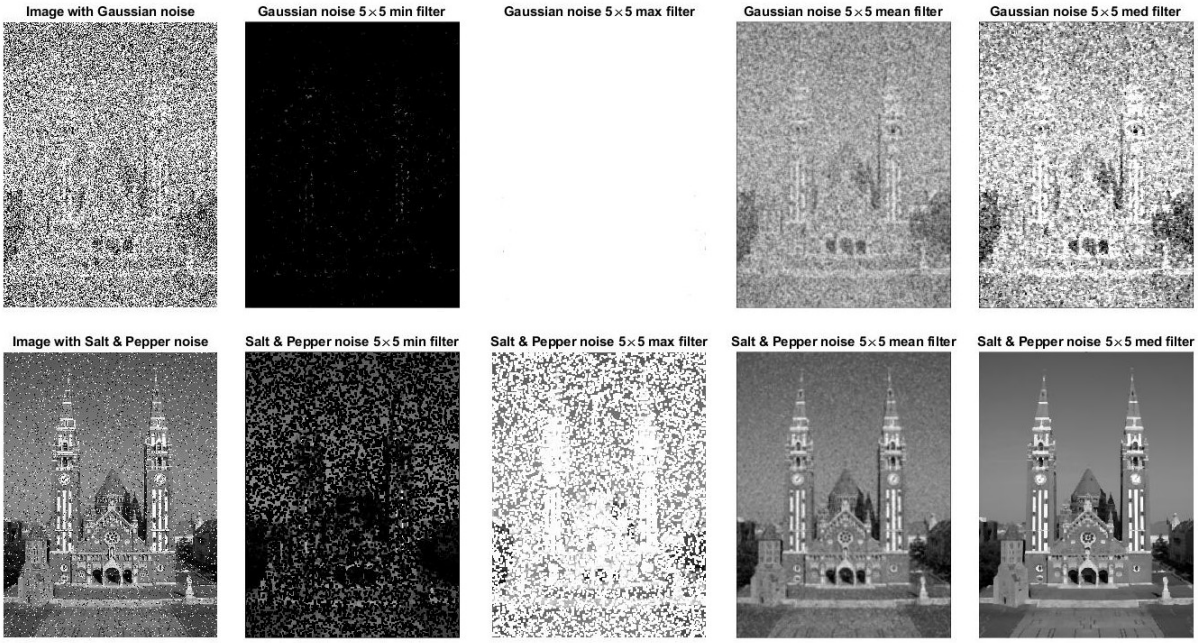


Figure 4: Result of 2.1.2

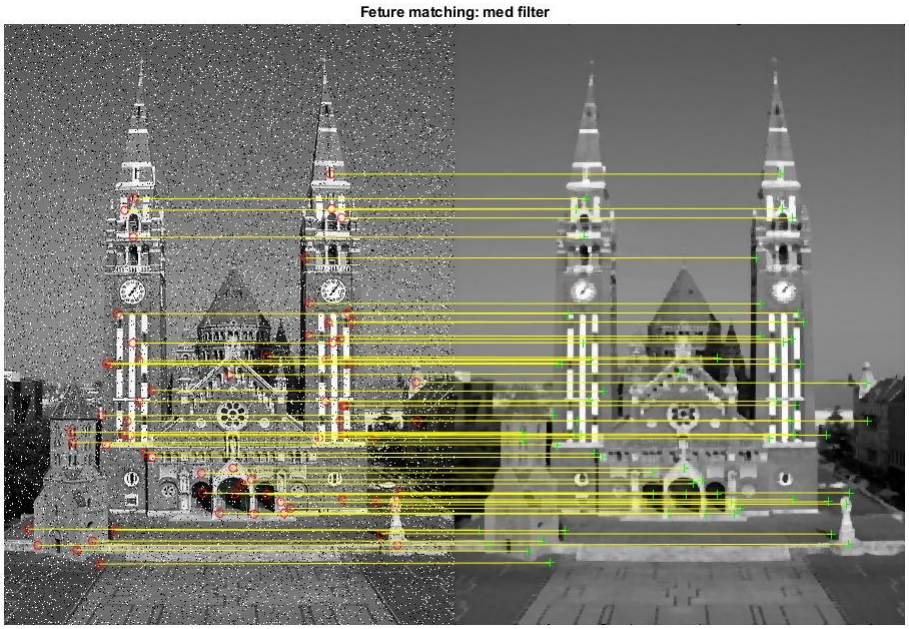
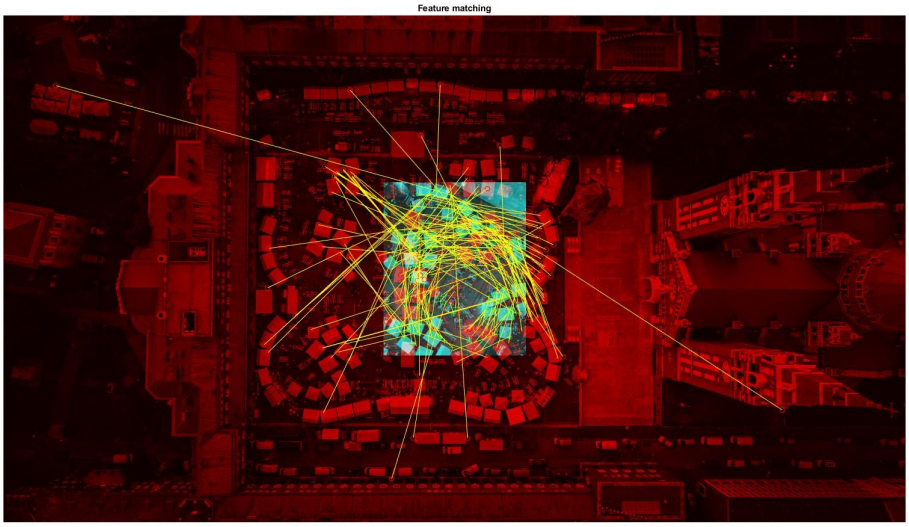


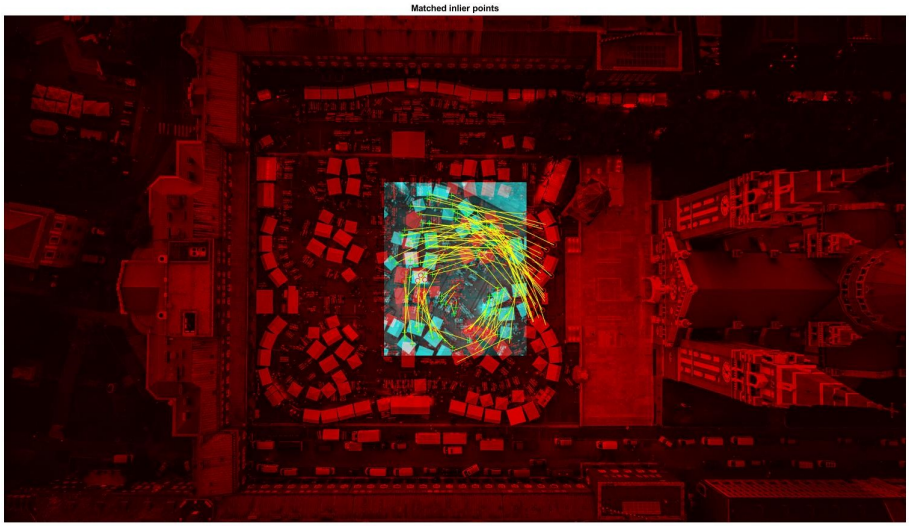
Figure 5: Result of 2.2

Table 1: Table of 3.1

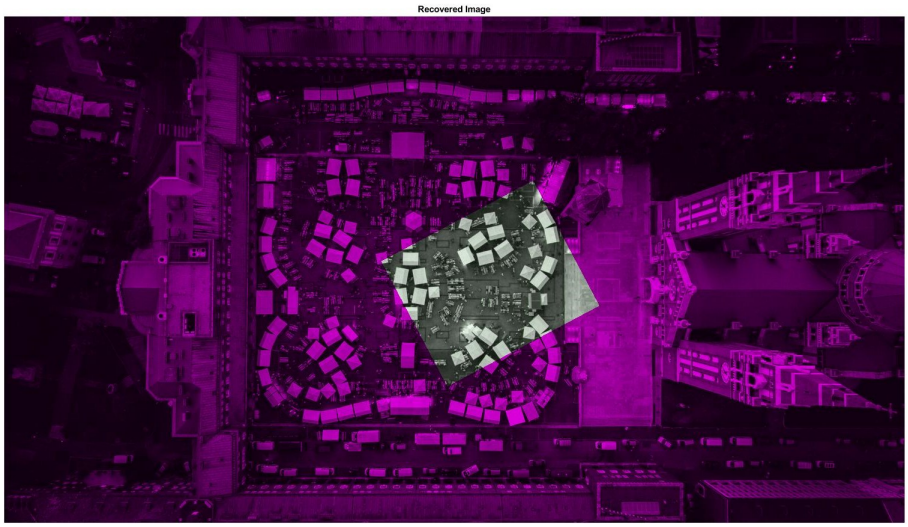
Operating	Type	Parameter	Effect
Median filter	Per-processing	Filter kernel size (5×5)	Filter out image noise points
Close operation	Post-processing	Filter kernel shape (disk), Filter kernel size (5×5)	Reduce the influence of median filtering on non-noise points
Watershed	Algorithm	Extended minimum transformation (1)	Cells separated together



(a) All of the matches including outliers



(b) Inlier matches



(c) Recovered image

Figure 6: Result of 2.3

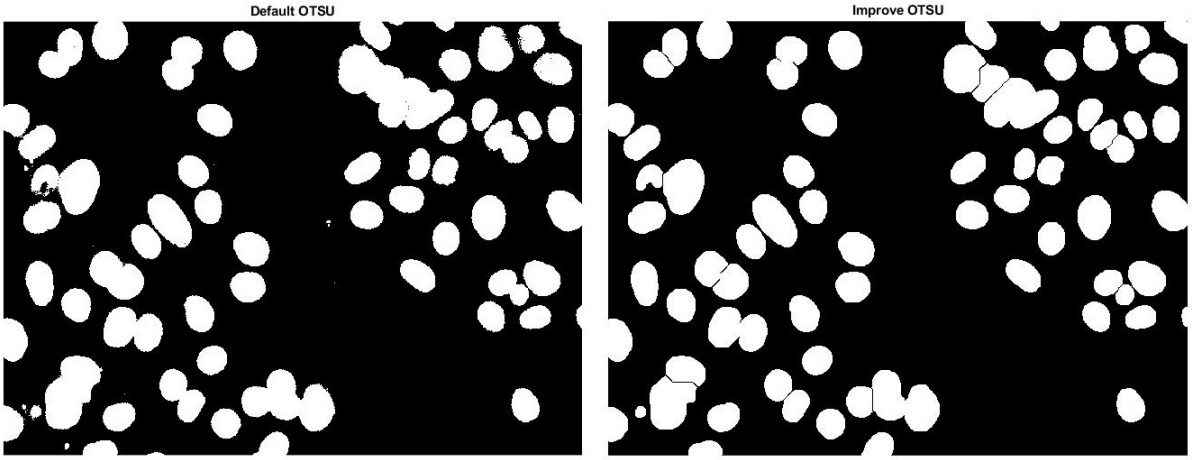
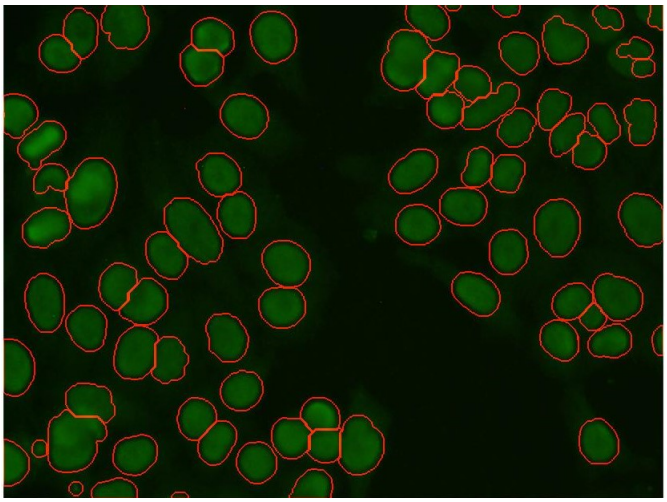
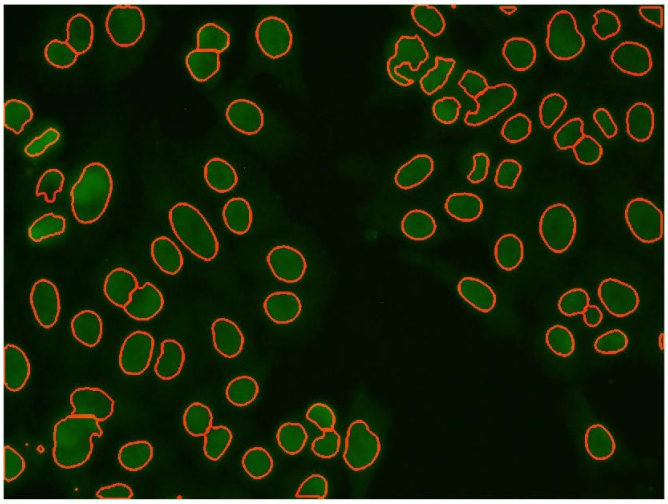


Figure 7: Result of 3.1



(a) Version 1



(b) Version 2

Figure 8: Result of 3.2