

# HOMework 3 U2

## MORPHOLOGICAL OPERATIONS 3

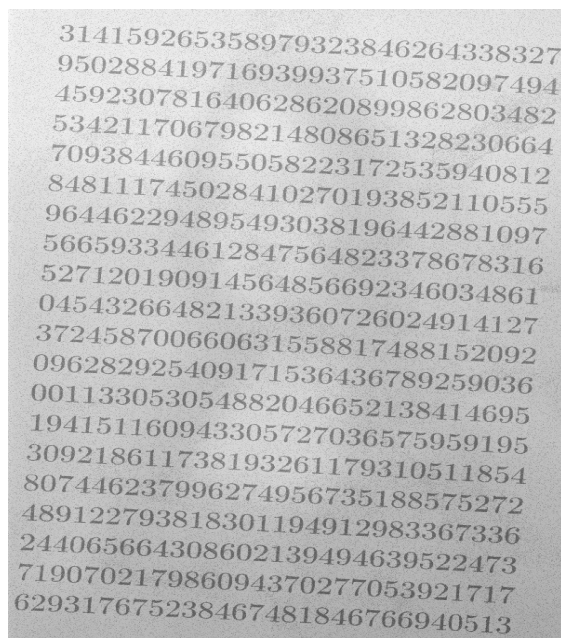
Héctor Mauricio Mendoza Xicoténcatl

**Abstract**—In this report, we documented the last of three homework assignments related to morphological transformations. Particularly, in this activity, we applied the knowledge learned in the previous two activities to find a solution to a problem.

**Index Terms**—OpenCV, Bitwise-or, Morphology, Hit or Miss operation.

### 1 PROBLEM TO SOLVE

From the following image:



- Try to obtain the background image (Mona Lisa) from the foreground
- Try to obtain the foreground from the background (numbers)

Use any algorithm you think suits the best.

### 2 CODING PROCESS

For the problem solving of this activity, the main algorithms we used involved the use of:

- The Morphological Operation Hit or Miss.
- Dilation Algorithm.

In order to detect the numbers on the image, we used the dilation algorithm. In order to detect Mona Lisa silhouette, we used the Hit or Miss operation. Therefore, our following code is a combination of both algorithms. Similar to previous exercises, first we imported the libraries necessary for the program with the commands:

```
import cv2 as cv
import numpy as np
```

In order to implement or morphological operation, we need to be able to read the values of our input image. Thus, we used the following line:

```
img_dimensions = image[ :, :, 2 ]
```

The next step is to create our Kernel matrix to detect Mona Lisa.

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

We apply the Hit or Miss:

```
output_image = cv.morphologyEx(
    img_dimensions, cv.MORPH_HITMISS,
```

```
kernel_matrix, iterations = 11)
```

```
kernel_matrix = (kernel_matrix + 1) *  
127  
kernel_matrix = np.uint8(kernel_matrix)
```

We apply a threshold to detect numbers.

```
ret,img = cv2.threshold(img, 127, 255, 8)  
kel = np.zeros(img.shape, np.uint8)  
kernel = np.zeros((5,5), np.uint8)
```

We apply a Kernel structuring element and dilate our image to detect the numbers:

```
elem = cv2.getStructuringElement(cv2.  
MORPH_CROSS, (3,3))  
dil_img = cv2.dilate(img, kernel, iterations=1)
```

We used a while cycle to find the numbers:

```
while True:  
— open = cv2.morphologyEx(img,  
cv2.MORPH_OPEN, elem)  
— temp = cv2.subtract(img, open)  
— eroded = cv2.erode(img, elem)  
— kel = cv2.bitwise_or(kel,temp)  
— img = eroded.copy()  
— if cv2.countNonZero(img)==0:  
—— break
```

Finally, we print the results:

```
cv2.imshow("Original_Image", img_org)  
cv2.imshow("Letters", letters)  
cv2.imshow("Final_kernel", kernel_matrix)  
cv2.imshow("Result_Hit_or_Miss",  
output_image)
```

### 3 QUESTIONS

#### 3.1 COULD YOU SEGMENT OUT THE NUMBERS FAIRLY WELL?

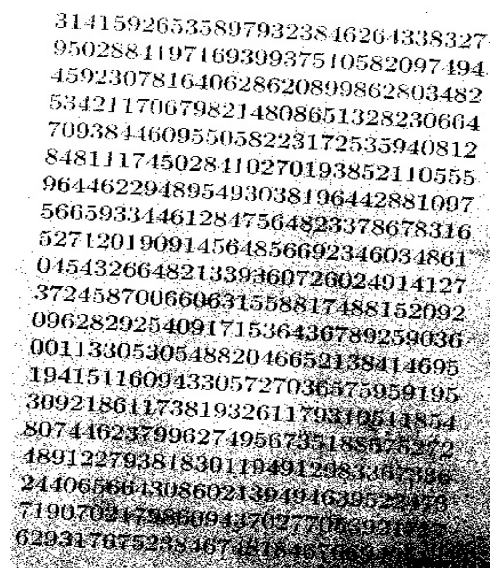
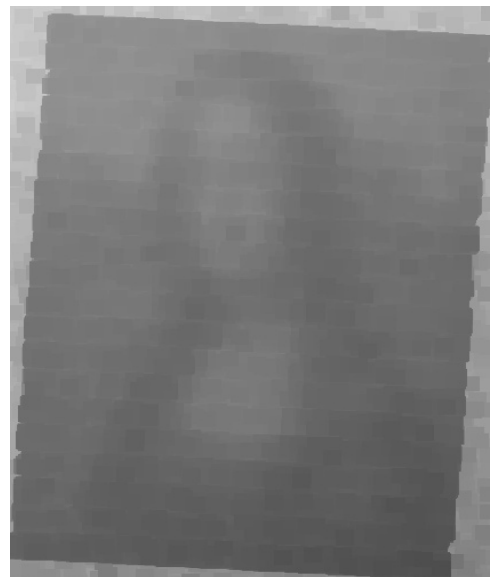
I consider the results for the numbers are good enough to be considered as a fair attempt for improvement. There is some noise on the right bottom side of the image, but I consider it is a fair attempt.

#### 3.2 COULD YOU SEGMENT OUT THE MONA LISA FAIRLY WELL?

In comparison with the numbers in the image, the results to find Mona Lisa are not as smooth as seen with the numbers.

I consider that the operation to find Mona Lisa is not as good as we would like to. But analyzing the problem itself, we have concluded that the expected results cannot be really different from ours. Due to the image complexity, it is understandable that we could not extract the integrity of the picture, not as Leonardo da Vinci would have wanted to.

### 4 RESULTS:



**Héctor Mendoza** Héctor Mauricio Mendoza Xicoténcatl is a Computational Robotics Junior student, currently studying at Universidad Politécnica de Yucatán.