Morphological Image Processing for detecting objects in binary images

Honore MBAYA
h.mbaya@stud.uis.no

Ziad RAZANI
z.razani@uis.no

ABSTRACT

Extracting low-level features of objects in an image to perform high level tasks like detection, classification and segmentation is not a trial task. For some applications requiring a clean up image with perfect rendering of the structure of objects in it, morphological operations tend to be a powerful alternative tool. In this work, we explore the theory behind some morphological image processing operations for binary images. We attempt some experiments on tasks like hand digit recognition, character extraction and extraction of connected components using basic morphological operations. We find that, while morphological operations can perform well when dealing with objects in images with precise and distinct shape attributes, they are in most cases computational demanding and thus inefficient for high level computer vision task. However, integrating them as part of the preprocessing or post processing stage, as demonstrated in the literature found, usually improve the accuracy of the results.

KEYWORDS

Mathematical Morphology, Image processing, character detection

1 INTRODUCTION

Morphological Image Processing is based on the theory of Mathematical Morphology where we consider the set of pixels in a binary image and proceed to apply operations in order to extract features from that set.

Mathematical Morphological operation are frequently used in binary image processing for a variety of tasks, including noise reduction, contour detection, and shape regularization. Available literature reveals a wide range of domains where morpholocical operations were successfully applied. [3] proposed a method for character extraction from cover images. Based on basic morphological operations, this method allows the identification of thin and long lines as key features for character extraction from cover images. This method has been improved later on by [8] They separated the input images into various clusters based on the length of the texts to address the single threshold limitation in Gu's method. [7] used mathematical morphology to recognize a collection of handwritten digits. Their method rely on the extraction of some properties that can be used to clearly separate one digit from the other. A rule based decision tree is then implemented to perform the classification. In image forensics [2] suggested a method for binary image morphological filtering forensic detection. The method mainly exploits erosion and dilation operations to detect filter application and estimate the shape of the structing element used. Given a binary image that is unknown, the task consists of evaluating whether it has undergone morphological erosion, morphological dilation, or neither. In medical image segmetation, [4] leverage grayscale morphological and filtering operations as a post-processing in tooth instance segmentation on panoramic dental radiographs. Their experiments show applying morphological post-processing stage to the sigmoid output of U-Net Network significantly reduce the mean error of tooth count. In the same line [6] used morphological operations such as opening, closing in their preprocessing steps for analyzing liver cancer detection.

Inspired by these applications, we will be diving deep into the theory behind morphological operations while trying to demonstrate its effectiveness through some experiments.

2 PROBLEM STATEMENT

The problem can be summarized through the following question: Can morphological image processing be used to detect shapes in a binary image? Precisely, symbols such as numbers and letters that are both printed and hand-drawn.

3 THEORY

Morphological image processing for binary images relies on a set of mathematical morphology operations. These operations allow us to determine the shape of the pixel regions in an image. There are two fundamental morphological operations from which others are derived. We briefly describe these operations and some of the derived ones used for the experiments.

Before presenting them, we will introduce some mathematical notation we will be using. Consider the sets B and C:

$$(B)_x = \{w | w = b + x, \forall b \in B\}$$

$$\hat{C} = \{w | w = -c, \forall c \in C\}$$

Respectively the translation and reflection of sets B and C [5]

3.1 Dilation

In a typical binary image with 0's representing the background and 1's representing the foreground, the dilation operation expands the connected set of 1's of binary image. It is mostly used for growing features and filling holes and gaps, among other applications. Dilation corresponds to Minkowski addition and its formal equation is represented as follows for the dilation of A by kernel K:

$$A \oplus K = \{z | (\hat{K})_z \cap A \neq \emptyset\}$$

[5]

3.2 Erosion

The Erosion operation shrinks the connected sets of 1's of binary image. It is commonly used for shrinking features, removing bridges, protrusions, branches, etc. Erosion is equivalent to the Minkowski subtraction after reflecting the second operand.

$$A \ominus K = \{z | (\hat{K})_z \subset A\}$$

[5]

3.3 Structuring element

It is a shape kernel (or mask) of any size digitally representable used in basic morphology operations. It has a defined origin usually in the middle. Examples of kernel sizes are presented below :

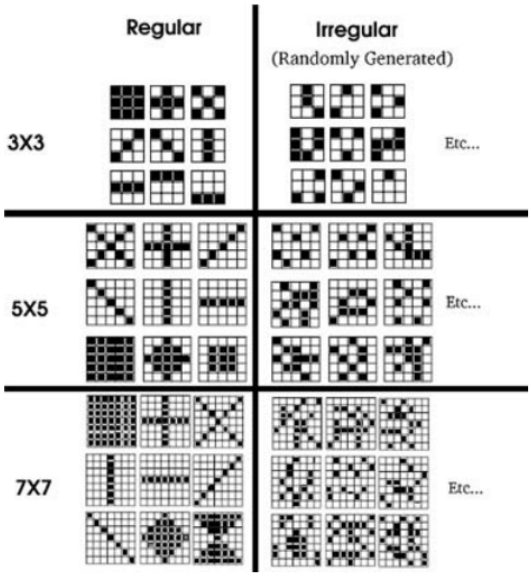

Figure 1: Example of different sized kernels [5].

3.4 Dilation and Erosion algorithms

Hence, given a Binary image and a Structuring element, the dilation and erosion algorithms consists of sliding the structuring element over the binary image, placing the origin at each pixel and computing the binary OR, in case of dilation, and the binary AND in case of erosion, of the corresponding element in binary image.

As mentioned above, from the two basic operations we can get many other composite relations. The most used are closing and opening operations. Both of them have a idempotent property, meaning that repeated operation has no further effects.

3.5 Opening

It is a compound operation of erosion followed by dilation using the same structuring element. Applying opening on an image by a given structuring element corresponds to the union of all translations of the structuring element that fit entirely within the image.

$$AoB = (A\check{\ominus}B) \oplus B$$

3.6 Closing

Likewise, Closing is a compound operation of dilation followed by erosion with the same structuring element. The closing of an image by a structuring element corresponds to the complement of union of all translations of the structuring element that do not intersect the image.

$$A \bullet B = (A \oplus B)\check{\ominus}B$$

3.7 Connected Components

Connected components extraction, also known as connected component labelling, is created by giving each pixel in an image a special label that shows which connected components it belongs to [1, p. 157] . Specifically, it consists of identifying and analyzing each connected set of pixels, either in 4 way or in 8 way connected, from a binary image. The algorithm is formally presented in Figure…

4 EXPERIMENTS

4.1 Exploration of digit detection

Using Python's cv2 implementation of the erosion and dilation operations, we started by attempting to implement a digit recognition algorithm that is described in [7]. This algorithm relies on feature extraction, by targeting specific features that classify the digit into different groups. We rely mainly on the number of "blobs" *Holesinthedigit*, the number of vertical, horizontal and inclined lines, as well as other characteristics that we did not implement *Suchasconcavity*.... For instance, detecting two blobs leads us to conclude the digit is indeed the number "8", when detecting one blob can only lead us to conclude that the digit falls into the group "0,6,9". We present the decision tree from [7]:

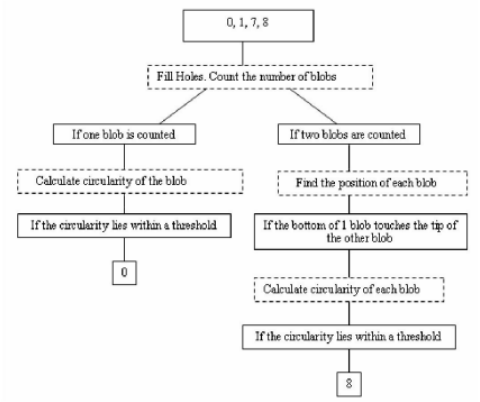

Figure 2: First half of the decision tree.

Our implementation can be found in morpho.py. We can summarize it in the following steps for blob detection:

(1) Loading and Thresholding: We turn the image into a binary image.
(2) Erosion: We erode the image using a $2, 2$ kernel to avoid modifying the digit's shape.
(3) Flood filling: Supposing that the digit will be centered, the background will start from the $0, 0$ pixel, therefore a flood fill using that as a starting position will lead to the "deletion" of the borders of the digit, isolating the blobs.
(4) Dilation: We use a $6, 6$ kernel.
(5) Contouring: To detect the number of blobs in the image, we chose to use the findContours method from cv2 (Other implementations are possible). This allows us to count the number of contours and simply return the result which is the number of detected blobs.

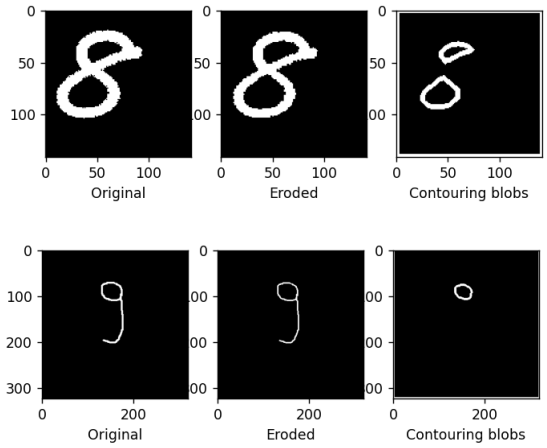

Figure 3: Applying the function to the number 8 and 9

4.2 Results of the exploration of digit detection

Our main conclusion following our experiment is that morphological image processing can be used for the detection of **characters with varying sizes but fixed characteristics**, printed digits or letters for example, since even if the size of their components might vary from an image to another, other parameters such as the inclination of lines and eccentricity remain the same. When we attempt to apply these algorithms to **hand drawn** symbols, we run into much more difficulties since our brain can recognize that a line is horizontal or vertical despite its slight inclination by extracting information from the rest of the image, which is the weakness of the algorithmic approach and therefore necessitates treating all these cases separately (Becoming a less efficient method when comparing it to machine learning or DL models).

4.3 Letter extraction

In this experiment, instead of trying to isolate every characteristic of the symbol we want to recognize we chose a different approach: Eroding the image using as a kernel the character itself. The result of this operation is a remaining pixel at the center of every corresponding symbol in the image, we then locate that pixel and extract the whole character using binary AND operation on the original image. In a sense, the erosion operation acts as a locator of the whole symbol. Here is the algorithm's description:

(1) Dilation: We use a (2,2) kernel to dilate the original image.
(2) Erosion: We erode the image using the target symbol as the kernel.
(3) Binary AND: We loop over the image, locating the remaining "1"s in the image and proceed to a binary AND operation between the block centered on the located "1" and the target symbol kernel.

4.4 Lines and words detection (counting) in a document image

To perform the counting of lines and words in a test document image, we first threshold the image, then w use erosion to shrink

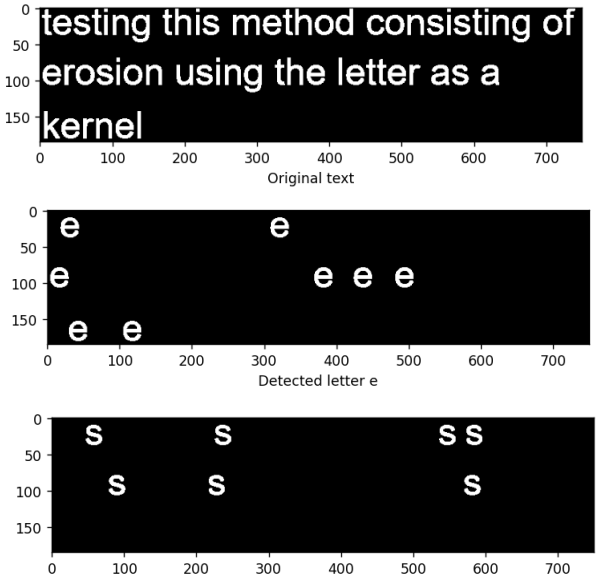

Figure 4: Applying the algorithm on the letters "e" and "s"

spaces between words and, finally, perform Connected Component labelling. The results are shown in the figure below.

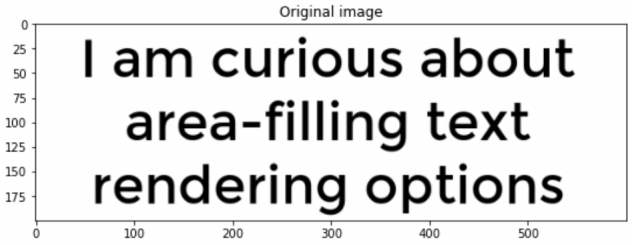

Figure 5: Original document image

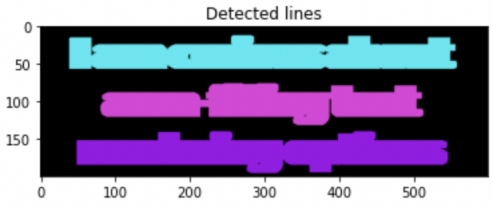

Figure 6: Detected lines highlighted by different colors

Although the above results look somehow accurate, application of Connected components to document image highly depends on the input image as well as the Structuring Element. This means it is difficult to find a standard kernel applicable to any input image.

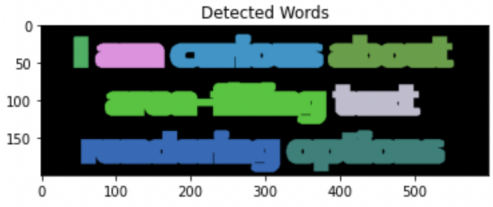

Figure 7: Detected words are labelled with different colors

5 DISCUSSION AND CONCLUSIONS

Morphological Image Processing is a powerful tool when treating objects in images with precise and distinct morphological attributes. Using a combination of Erosion and Dilation operations in parallel with adequate kernel shapes can isolate these attributes, making it an effective way of detecting symbol patterns. That being said, mathematical morphology has its limits and fails where other approaches such as deep learning approaches excel, and that is because it remains an algorithmic approach that increases in complexity when dealing with more complex images (such as hand-drawn symbols).

A CODE FOR THE EXPERIMENTS

All the code can be found on our git repository on MorphologicalImageProc. We can find two branches, one containing blob detection and digit detection decision tree as well as letter detection, and another one containing the connected components experiment.

To run the code, comments were added to guide the user, we can uncomment the desired lines for the procedure we want to execute. We can also add images into the images folder to test the procedures on them.

```
#Loading and Thresholding
txt = cv2.imread("./images/test_text.png",cv2.IMREAD_GRAYSCALE)
letter = cv2.imread("./images/letter_e.png",cv2.IMREAD_GRAYSCALE)
(thresh, txt) = cv2.threshold(txt, 127, 255, cv2.THRESH_BINARY)
threshold = 0
txt[txt>threshold]=1
txt = 1-txt
(thresh, letter) = cv2.threshold(letter, 127, 255, cv2.THRESH_BINARY)
letter[letter>threshold]=1
letter = 1-letter
#Dilation
txt = cv2.dilate(txt,np.ones((2,2),np.uint8))
#Erosion
blank = cv2.erode(txt,letter)
#Location + Logical AND
res = np.zeros(txt.shape,np.uint8)
h,w = letter.shape
for i in range(blank.shape[0]):
    for j in range(blank.shape[1]):
        if blank[i,j] >0:
            block = txt[i-h//2:i+h//2+1,j-w//2:j+w//2+1]
            res[i-h//2:i+h//2+1,j-w//2:j+w//2+1] = np.logical_and(block,letter)
```

Figure 8: Code of the letter detection procedure

It is important to note that letter detection algorithm is valid for any "shape", we can extract that shape from any image as long as we provide it as the kernel. Therefore it can be used beyond symbol detection.

REFERENCES

[1] Stan Birchfield. 2018. *Image Processing and Analysis.* CENGAGE Learning, Boston.

[2] Boato G. De Natale, F. G. B. [n.d.]. Detecting Morphological Filtering of Binary Images. *IEEE Transactions on Information Forensics and Security, 12(5), 1207–1217* ([n. d.]).

[3] Tanaka N. Kaneko T. Haralick R. M Gu, L. 1998. The extraction of characters from cover images using mathematical morphology. *Systems and Computers in Japan, 29(4), 33–42.* (1998), 33–42.

[4] Hamamci A. Helli, S. S. 2021. Tooth Instance Segmentation on Panoramic Dental Radiographs Using U-Nets and Morphological Processing. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi.* (2021), 39–50.

[5] Riccardo Poli Marcos I. Quintana and Ela Claridge. 2006. Morphological algorithm design for binary images using genetic programming. (2006), 2.

[6] Taghavirashidizadeh A. Moghimhanjani, M. 2022. Analysis of liver cancer detection based on image processing. *arXiv:2207.08032* (2022), 1–10.

[7] 'Fatimah Mohammad and S.A. Husain. 2007. Character Recognition Using Mathematical Morphology. (2007), 1–6.

[8] Sultana M. Rahman T. Busra Shorif Uddin, M. 2012. Extraction of texts from a scene Image using morphology based approach. *2012 International Conference on Informatics, Electronics Vision (ICIEV), 876–880.* (2012), 876–880.