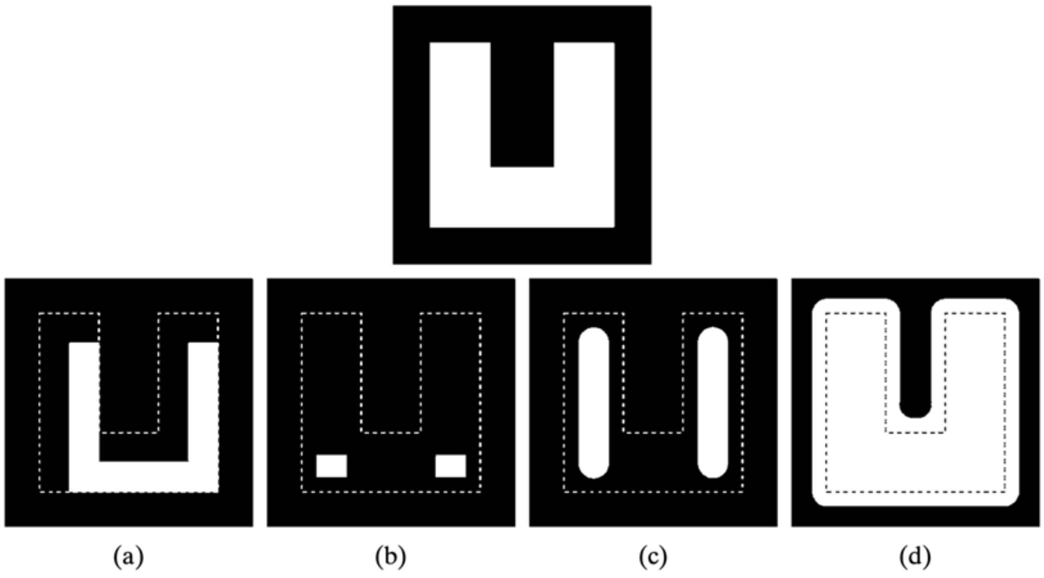
EIE4512 - Digital Image Processing – Assignment #3

Instructor: Zhen Li Due on 2024/07/21 11:59 pm

Part I Written Exercises

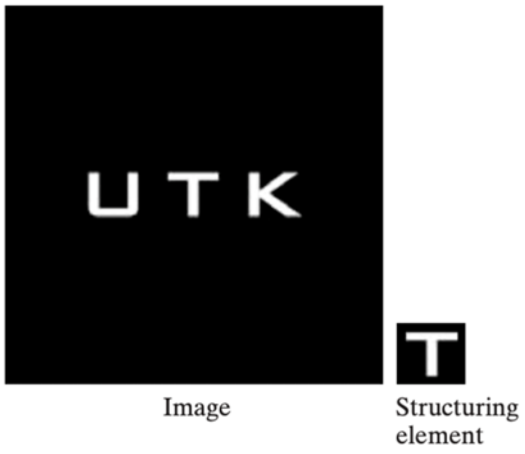
Problem 1 (5 points)

With reference to the image shown, give the structuring element and morphological operation(s) that produced each of the results shown in images (a) through (d). Show the origin of each structuring element clearly. The dashed lines show the boundary of the original set and are included only for reference. Note that in (d) all corners are rounded.



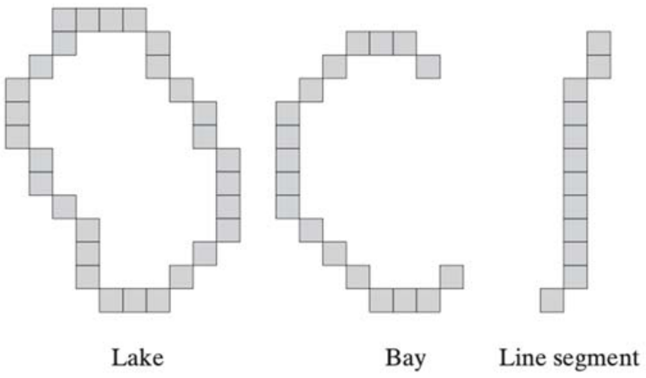
Problem 2 (5 points)

Sketch the result of applying the hit-or-miss transform to the image and structuring element shown. Indicate clearly the origin and border you selected for the structuring element.



Problem 3 (5 points)

Three features (lake, bay, and line segment) useful for differentiating thinned objects in an image are shown below. Develop a morphological/logical algorithm for differentiating among these shapes. The input to your algorithm would be one of these three shapes. The output must be the identity of the input. You may assume that the features are 1 pixel thick and that each is fully connected. However, they can appear in any orientation.



Problem 4 (5 points)

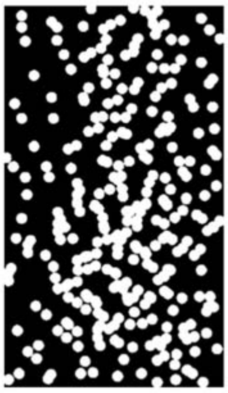
A gray-scale image, $f(x, y)$, is corrupted by nonoverlapping noise spikes that can be modeled as small, cylindrical artifacts of radii $R_{\min} \le r \le R_{\max}$ and amplitude $A_{\min} \le a \le A_{\max}$.

(a) Develop a morphological filtering approach for cleaning up the image.
(b) Repeat (a), but now assume that there is overlapping of, at most, four noise spikes.

Problem 5 (5 points)

A preprocessing step in an application of microscopy is concerned with the issue of isolating individual round particles from similar particles that overlap in groups of two or more particles (see following image). Assuming that all particles are of the same size, propose a morphological algorithm that produces three images consisting respectively of

(a) Only of particles that have merged with the boundary of the image.
(b) Only overlapping particles.
(c) Only nonoverlapping particles.

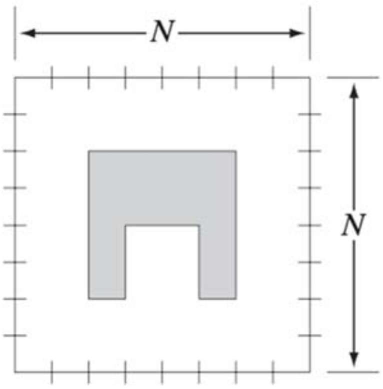


Problem 6 (5 points)

The objects and background in the image shown have a mean intensity of 170 and 60, respectively, on a [0, 255] scale. The image is corrupted by Gaussian noise with 0 mean and a standard deviation of 10 intensity levels. Propose a thresholding method capable of yielding a correct segmentation rate of 90% or higher. (Recall that 99.7% of the area of a Gaussian curve lies in $a \pm 3\sigma$ interval about the mean, where σ is the standard deviation.)

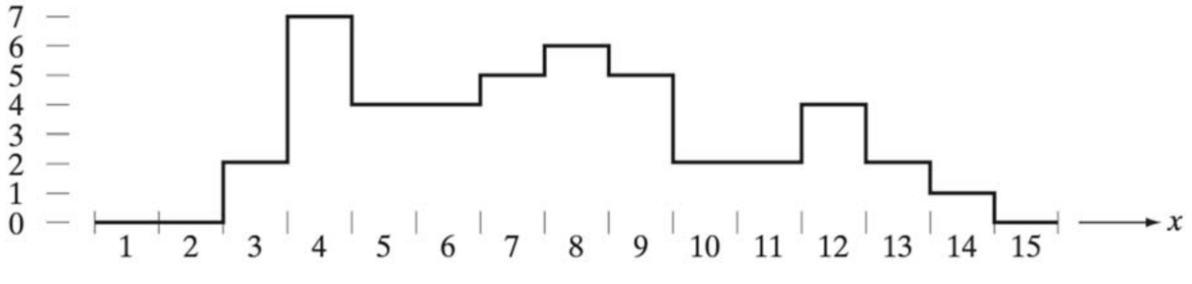
Problem 7 (5 points)

Segment the image shown by using the split and merge procedure. Let $Q(R_i)$ = TRUE if all pixels in R_i have the same intensity. Show the quadtree corresponding to your segmentation.



Problem 8 (5 points)

Give a step-by-step implementation of the dam-building procedure for the one-dimensional intensity cross section shown. Show a drawing of the cross section at each step, showing "water" levels and dams constructed.



Submission List:

- All answers in the file *your_school_id_name.pdf* (*e.g.*, *1190190xx_San_Zhang.pdf*).

Part II Programming Exercises

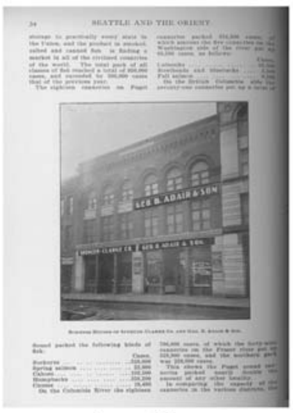
Problem 1 Binarization of Scanned Book Pages (20 points)

Please use “pro1_page1.jpg”, “pro1_page2.jpg” and “pro1_page3.jpg” as the inputs. You can only use the following Python libraries: OpenCV, numpy, math, matplotlib, and scipy.

Submission List:

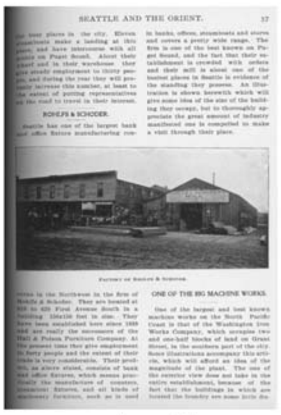
- Codes for part (a) in the file *global_thresh.py*.
- Codes for part (b) in the file *local_thresh.py*.
- Codes for part part (a), (b), and (c) in the file *main.py*.
- All code files in the folder *Problem1*.
- Display the results for part (a), (b), and (c) in the file *your_school_id_name.pdf*. (*e.g.*, *1190190xx_San_Zhang.pdf*).

Services like the Guttenberg Project and Internet Archive have digitized large collections of books. First, the book pages are scanned. Then, the scanned images are binarized and processed through an optical character recognition (OCR) engine. For modern books, the book spine can be removed to separate the book pages for more efficient scanning. For vintage books, however, destroying the original binding of the book is often undesirable. If a book is scanned with its spine left intact, the curvature of the pages causes uneven illumination in the resulting images. This effect can be observed in the images pro1_page1.jpg and pro1_page2.jpg, which are available in the figs folder.


34 SEATTLE AND THE ORIENT

[illegible]

pro1_page1.jpg


SEATTLE AND THE ORIENT. 37

[illegible]

pro1_page2.jpg

(a) For the images pro1_page1.jpg and pro1_page2.jpg, generate a binary image by performing global thresholding and implement the function *globalThresh(img, parameter1, …)*. Use a threshold chosen by Otsu's method (using OpenCV function cv2.threshold). Display the binary image and a histogram of the original image's gray values where the thresholds of Otsu's method are clearly marked. Comment on the quality of the binary image.

(b) For the images pro1_page1.jpg and pro1_page2.jpg, perform locally adaptive thresholding and implement the function *localAdaptThresh(img, parameter1, …)*. Treat uniform and non-uniform regions differently based on the local variance. Show the binary image. Comment on the quality of the binary image compared to the result from part (a).

(c) Apply locally adaptive thresholding1 to binarize the image pro1_page3.png, so that the text appears as black pixels and the blank space on the page appears as white pixels. Report the block size you select for locally adaptive thresholding. Display the binarized image.

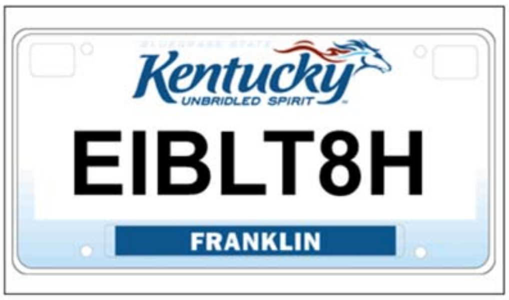
Problem 2 Morphologial Image Processing (25 points)

Please use "pro2_license_plate_clean.png", "pro2_license_plate_noisy.png", and 36 separate images of the alphanumeric characters in "pro2_character_templates.zip" as the inputs. You can only use the following Python libraries: OpenCV, numpy, math, matplotlib, and scipy.

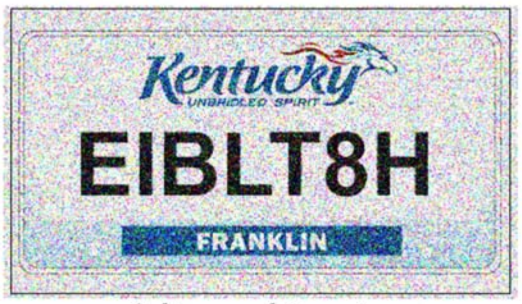
Submission List:

- Codes for part (e) in the file *rank_filter.py*.
- Codes for displaying results in part (a), (b), (c), (d) and (e) in the file *main.py*.
- All code files in the folder *Problem2*.
- Display the results for part (a), (b), (c), (d) and (e) in the *your_school_id_name.pdf*. (*e.g.*, *1190190xx_San_Zhang.pdf*).

In this problem, we study how to recognize the characters on a license plate using morphological image processing. Please use the two images pro2_license_plate_clean.png, pro2_license_plate_noisy.png, and pro2_character_templates.zip, which contains 36 separate images of the alphanumeric characters "ABC…XYZ0123456789" in the same font as the "EIBLT8H" written on the license plate.



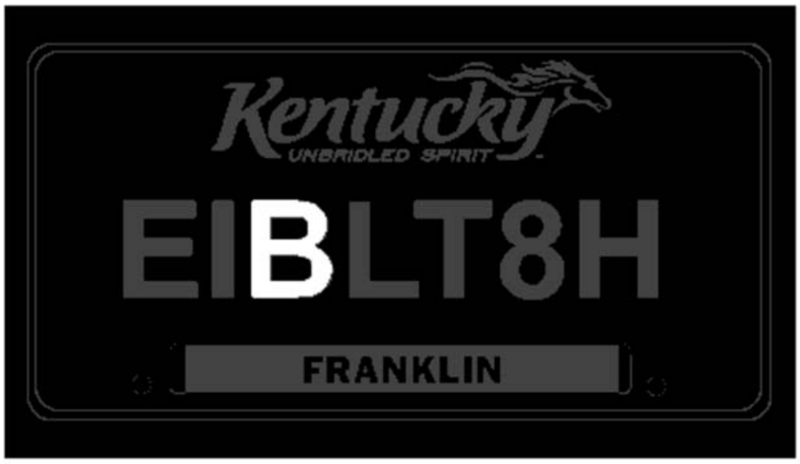
pro2_license_plate_clean.png



pro2_license_plate_noisy.png

(a) Binarize the clean license plate image, the noisy license plate image, and the template images, so that the large characters in the middle of the plate appear as white and the background appears as black. Choose the threshold by applying Otsu's method on the clean license plate image, and use the same threshold for all binarization operations in this problem. Display the binarized clean and noisy license plate images and the binarized template images.

(b) Perform character detection by erosion in the binarized clean license (using OpenCV function cv2.erode). To eliminate the effects of slight mismatches along the character boundaries, compute a slightly eroded template which is a 3 × 3 square, and then use the eroded template as the SE in the erosion detector. For each template that generates a nonzero detection result, dilate the eroded license plate image (using OpenCV function cv2.dilate) by the template for visualization purposes and show the dilation result, like shown in the following example for the template "B". Which characters are detected by this approach? Are there any false positive detections, and if so, why?



(c) Repeat part (b) using a hit-miss filter in place of erosion for the detector (using OpenCV function cv2.morphologyEx). Use the same SE as in part (b) for the foreground SE. For the background SE, use the 5 × 5 square dilate template and 3 × 3 square dilate template to extract a thin outline around the character. Comment on the advantages of the hit-miss detector compared to the erosion detector.

(d) Repeat part (c) replacing the clean license plate image by the noisy license plate image. Comment on how the noise affects the hit-miss detector's accuracy.

(e) Repeat part (c) replacing the clean license plate image by the noisy license plate image and replacing the hit-miss filter by the minimum of two rank filters: *rankFilter1(img1, p1, SE1)* and *rankFilter2(img2, p2, SE2)*. Notice that *img1* is a binary image, *img2* is NOT a binary image, *SE1* and *SE2* are the same foreground and background SEs used in part (c), and *p1* and *p2* indicate the two ranks, respectively. If *p1* and *p2* are 1, the minimum of the two rank filters is the same as the hit-miss filter. Choose and report the ranks *p1* and *p2* that enable the correct detection of the characters in the noisy license plate without false positives.

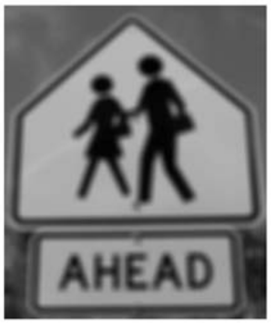
Problem 3 Sharpness Enhancement by Dilation and Erosion (15 points)

Please use “pro3_road_sign_school_blurry.jpg” as the input. You can only use the following Python libraries: OpenCV, numpy, math, matplotlib, and scipy.

Submission List:

- Codes for displaying results in part (a) and (b) in the file *main.py*.
- All code files in the folder *Problem3*.
- Display the results for part (a) and (b) in the *your_school_id_name.pdf*. (*e.g.*, *1190190xx_San_Zhang.pdf*).

In this problem, we use an iterative grayscale morphological image processing algorithm to enhance the sharpness of structures in a blurry image.



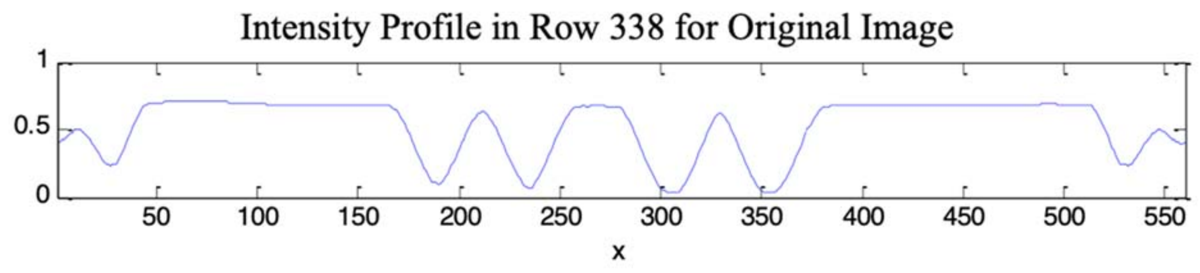
pro3_road_sign_school_blurry.jpg

(a) The image pro3_road_sign_school_blurry.jpg shows a school crossing road sign taken by an out-of-focus camera. Apply 10 iterations of the following algorithm. Design and report your structuring element. Display the resulting image after 10 iterations. Comment on which features in the image have been made sharper. (using OpenCV functions cv2.erode, cv2.dilate, and cv2.getStructuringElement).

```
Im := Input Image
For Iteration = 1:NumIterations
      Im_d = dilate(Im, W)     % Note that this is grayscale dilation
      Im_e = erode(Im, W)      % or erosion with structuring element W
      Im_h = 0.5(Im_d + Im_e)

      % Perform the following test for each pixel
      If Im ≥ Im_h
            Im := Im_d
      Else
            Im := Im_e
      End
End
```

(b) Display plots of the intensity profile in row 338 after 6 iterations. For example, the following figure is a plot of the intensity profile in row 338 in the original image. Comment on how the algorithm iteratively changes the intensity profile.


Intensity Profile in Row 338 for Original Image
1
0.5
0
50
100
150
200
250
300
350
400
450
500
550
x