Assignment 2 - Image transformations

Start Assignment

Due Sep 14 by 11:59pm **Points** 100 **Submitting** a file upload **File Types** zip
Available Sep 7 at 12am - Sep 17 at 11:59pm 11 days

Task 1 [50 Points]:

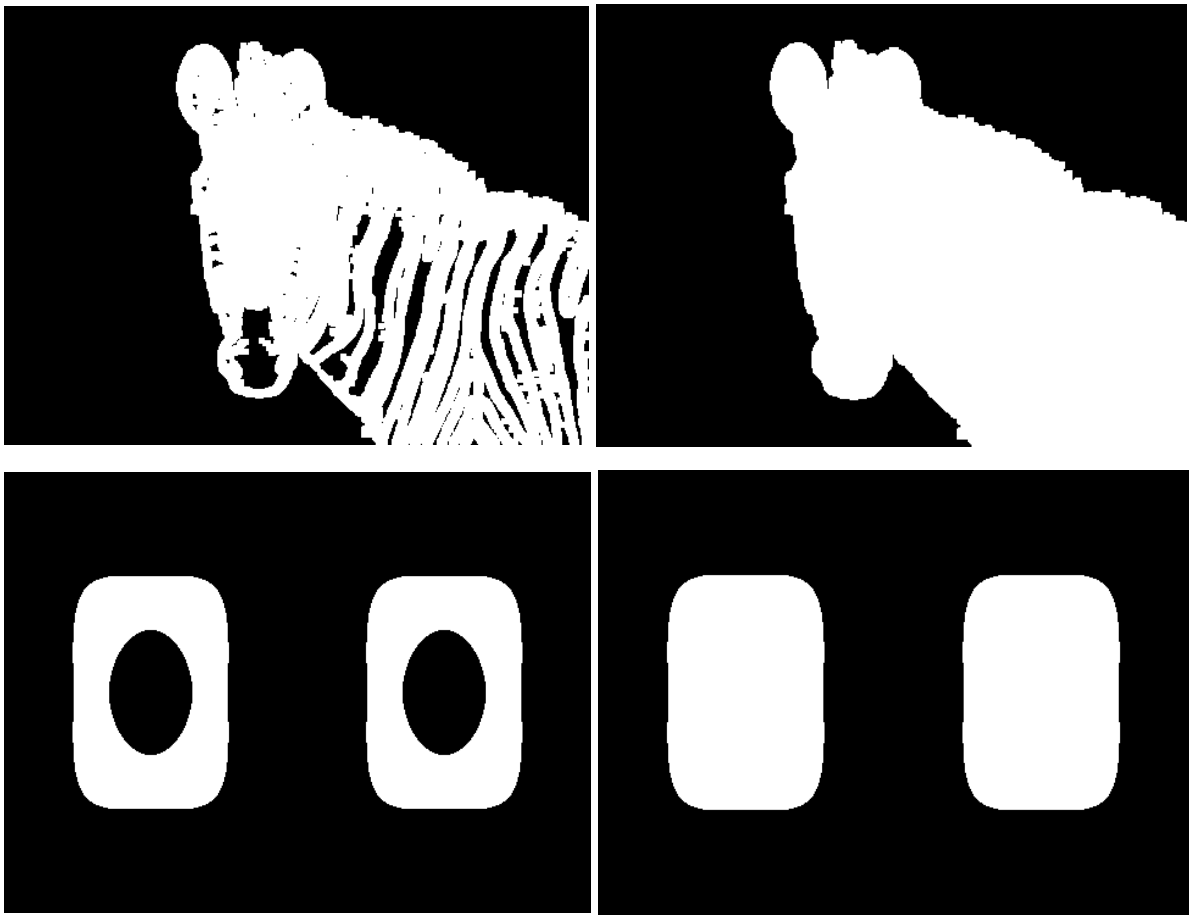


Figure 1: Example inputs and outputs of the remove_holes function.

Write a Matlab function that takes in as argument a binary image **A** *(not a filename*), and returns another binary image **B**, which is the result of removing all holes from A. We will use this (somewhat artificial) definition: A hole is a black 4-connected component that does not include pixel (1,1). If pixel (1,1) in A is black, then we use the term ``background'' for the connected component of all black pixels to which pixel (1, 1) belongs. If pixel (1,1) is white, then the result of your function should be an all-white image.

Your function should be named `remove_holes`, and should take a single argument, i.e., the binary image that you want to process.

```
>> B = remove_holes(A);
```

The key idea that makes it easy to identify holes is this: if C is the negation of A (`C = ~A`, in Matlab), a hole in A is a white connected component in C. The background in A also becomes a white connected component in C, but the label of the background (as returned by `bwlabel(C, 4)`) will be different than the label of any hole.

Your function should work for any image, not just the examples provided above. My solution for this task is 12 lines of code. Shorter solutions are also possible.

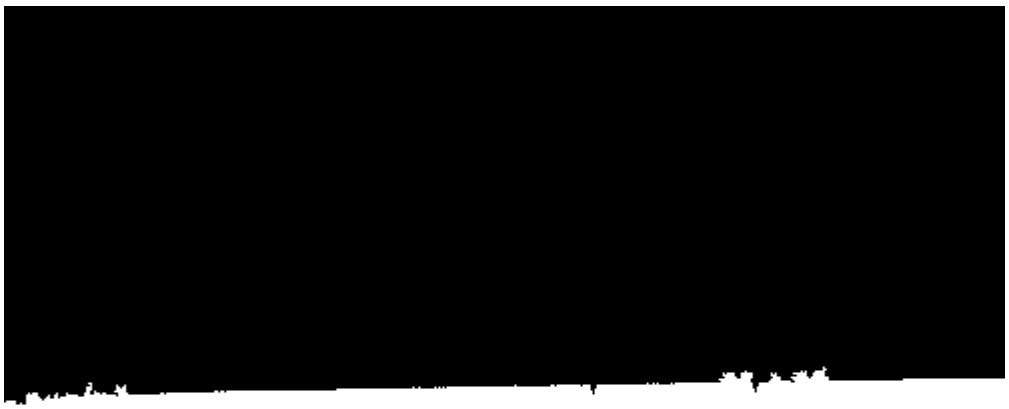
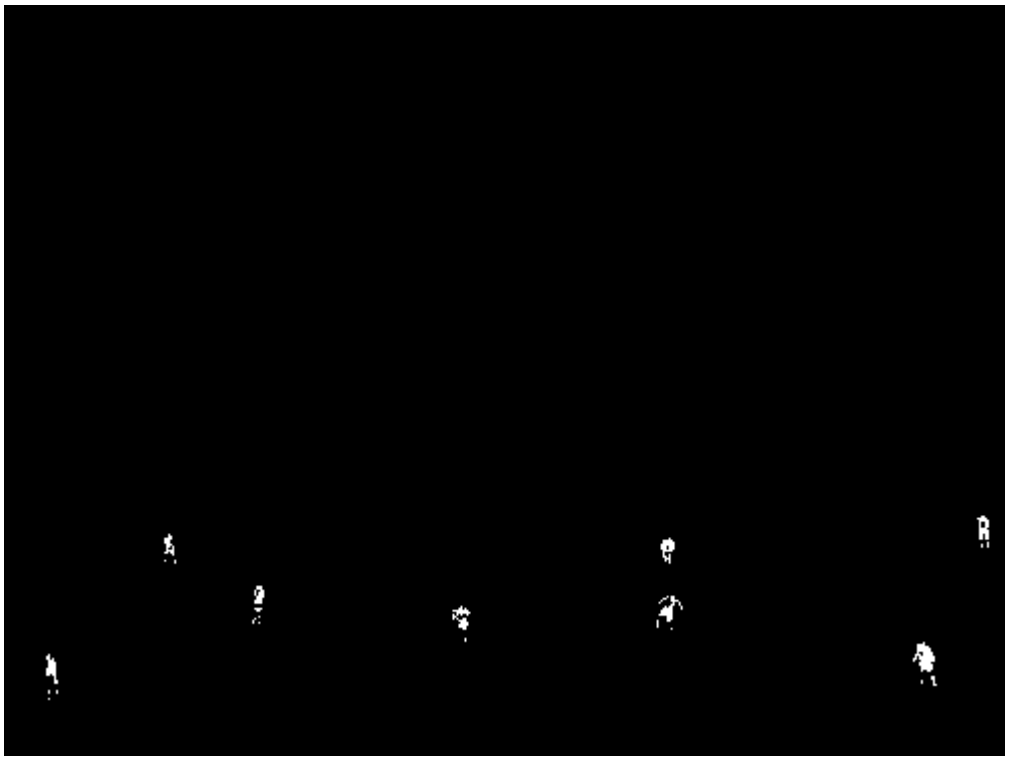
Note: to read a binary image for testing you may use the imread function with the files given, e.g.

```
>> my_image = imread('simple_with_holes.gif')

>> result = remove_holes(my_image);
```

Task 2 [50 Points]:

original image	
image showing area of soccer field	

	
image showing locations of red team players	
image showing locations of blue team players	

Write a Matlab script that reads as input, from a file called 'soccer_field4.jpg' the original image shown above, and produces three images (as shown in the figure above), that illustrate:

- In Figure 1: the area of the soccer field.
- In Figure 2: the locations of the red players.
- In Figure 3: the locations of the blue players.

To generate multiple figures you can use something like:

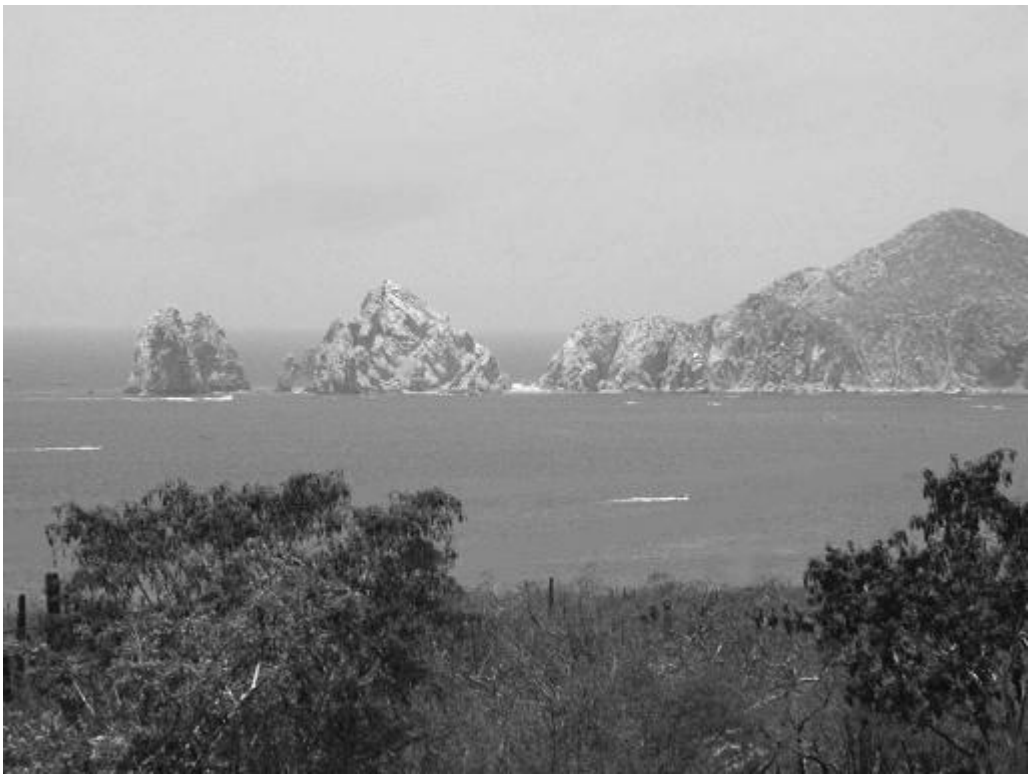
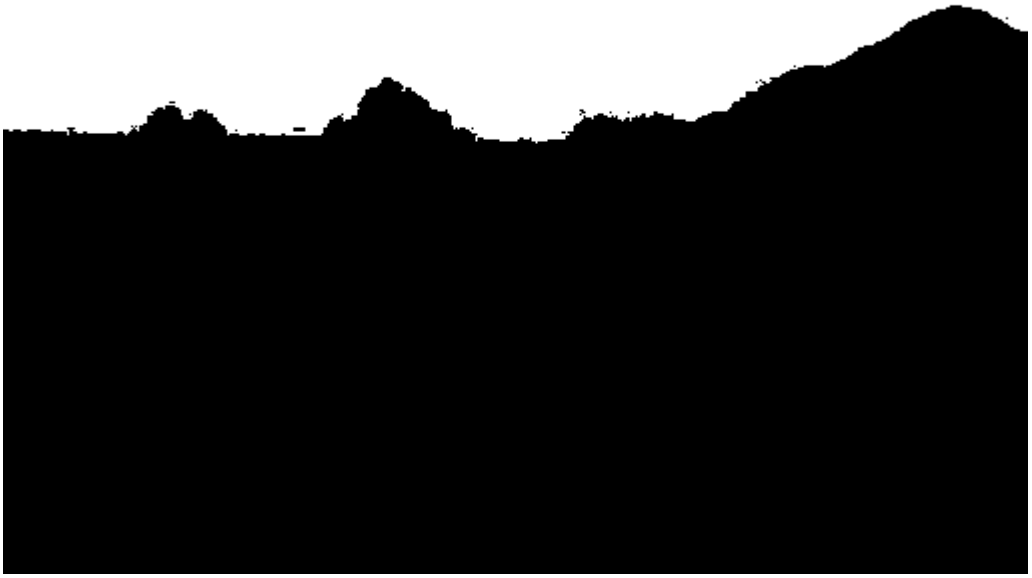
```
figure(1); imshow(field);
figure(2); imshow(red_players);
figure(3); imshow(green_players);
```

Your solution should just consist of images like the ones displayed above. You do not need to output anything else (such as number of players, bounding boxes, or anything like that). You're free to use any thresholds you like, and your solution only needs to work on this image.

My solution for this task (including reading in the original image and generating the three result images) was 16 lines of code (in addition to code written for task 1).

Note: A Matlab script is just a sequence of statements saved in a ".m" file. It does not accept or return any values.

Task 3 [optional: +20 points extra credit]:

original image	
image showing the sky area	
image showing the ocean area	

Write a Matlab script that reads as input, from a file called 'ocean2.jpg' the original image shown above, and produces two figures (as shown above): one showing the sky area, and one showing the ocean area. As in task 2, you only need to produce the figures, nothing else.

My solution for this task (including reading in the original image and generating the two result images) was 14 lines of code (in addition to code written for task 1).

Note: You may have to apply multiple tricks to achieve a result similar to the above, e.g. blurring, erosion, connected components, etc. Try to make your output image look as similar to the above as possible.

Grading

The assignment will be graded out of 100 points. All mandatory tasks (tasks 1 & 2) are worth the same number of points (50 points each). Task 3 is worth 20 extra points.

Things to note

- Input images for tasks 2 and 3, and example input and output images for all tasks can be found at the attached Data.zip file.
- For task 1, the correct solution should be a function. For the other tasks, the correct solutions can be scripts.
- There is no single unique answer for tasks 2 and 3. Just make sure your solution looks reasonable.
- For task 1, the solution should work for any binary image.
- For tasks 2 and 3, your solution only needs to work for the input image specified for each task.

- For tasks 2 and 3, feel free to use any morphological operation, filter, threshold, or any other computer vision-based technique you can think of to obtain the desired result.

How to submit

Submissions are only accepted via Canvas. The submission should include a zip file containing your Matlab code and a README.txt file. The submission should include, as an attachment, a zip file containing your Matlab code and a README.txt file. The zip file should be named NetId_lastname_firstname.zip. Your solution should definitely contain:

- A Matlab file called remove_holes.m, that implements the solution for task 1.
- A Matlab file called task2_solution.m, that implements the solution for task 2. THE SCRIPT SHOULD ASSUME THAT THE INPUT IMAGE IS LOCATED IN THE CURRENT DIRECTORY AND IS CALLED 'soccer_field4.jpg'.
- A Matlab file called task3_solution.m, that implements the solution for task 3. THE SCRIPT SHOULD ASSUME THAT THE INPUT IMAGE IS LOCATED IN THE CURRENT DIRECTORY AND IS CALLED ocean2.jpg'.
- Whatever other Matlab files your solution uses, regardless of whether you wrote them or you downloaded them from the course website. The README file should contain the name and student ID of the student, in addition to any additional comments/instructions useful for running the code and understanding the underlying ideas.

We try to automate the grading process as much as possible. Not complying precisely with the above instructions causes a significant waste of time during grading, and thus points will be taken off for failure to comply, and/or you may receive a request to resubmit.

Submission checklist

- Was the submitted zipped file called `NetId_lastname_firstname.zip`?
- Did you include a README.txt file, as specified?
- Was the submission zipped? We will not accept .rar, .tar, .gz, or any other filetypes, and we will not accept submissions where multiple files are submitted separately.
- Did you make sure that the submission includes only code and text files, and NO IMAGE files?
- For tasks 2 and 3, did you make sure that the filenames from which you read the input images are 'soccer_field4.jpg' and 'ocean2.jpg' respectively, and that your script assumes that these files are located at the current directory?

Additional resources for assignment:

Data.zip (https://canvas.txstate.edu/courses/1798576/files/200877813/download?download_frd=1)