Assignment 1 – CMP3108M Image Processing

Dylan Samuel Petty

19703357@students.lincoln.ac.uk

Deadline: 19/01/2023

Contents

Task1 – Pre-Processing:

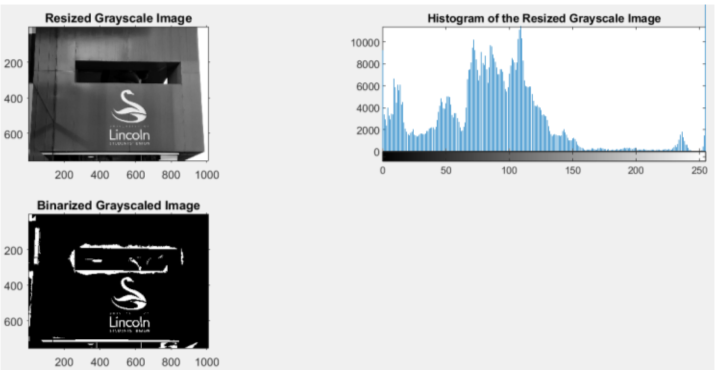


Figure 1. Greyscale Histogram and Binarized image

Based on the histogram that I generated from the resized grayscale image shown in Figure 1, a good value for a binary threshold for detecting the swan based on value would be around 200 because the swan is a slightly faded white colour meaning a high RGB value; shown by the small peak of entries at the far right side of the graph. Separating the swan based on just thresholding alone is quite ineffective in most cases as there are other pixels which share the same RGB value.

Using the imresize function to apply bilinear interpolation to accurately reduce the image in size. This process is a resampling method which gathers the value of the surrounding 4 cells and estimates the value of the selected cell after transformation, repeating this for all cells in the image until the transformation has been complete. (Patel, V. 2013. 129)

Otsu's thresholding method is a staple in thresholding as this has revolutionised the way automatic thresholding procedures work, this is the case because the algorithm "picks the threshold value to minimize the intra-class variance of the thresholded black and white pixels." (bangare et al. 2015. 21777 – 21780) ultimately this method specialises in separating pixels based on intensity into 2separate fields: foreground and background, allowing the images features to be truly picked apart.

Task 2 – Edge Detection

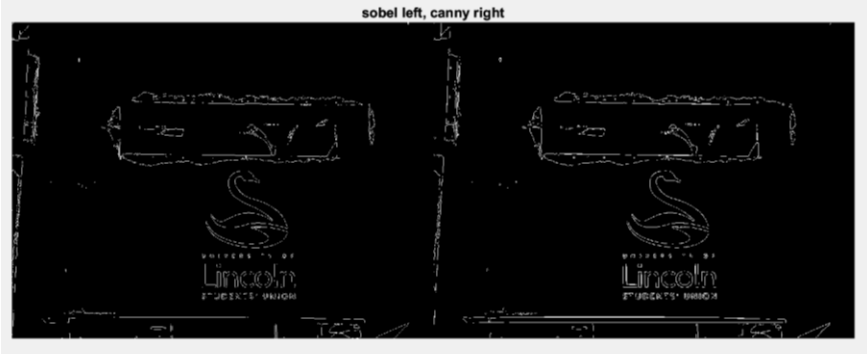


Figure 2. Edge detection methods.

As seen in Figure 2, the Canny edge detection method is slightly better performing in this example as there are less gaps in the overall shape of the swan in comparison to that of the Sobel detection method and the shape is smoother in general. The Sobel method detects fewer extra edges which may even make it easier to cut out excess noise when comparing the source image to a ground truth image. The Sobel also method especially struggles against the Canny method when producing smooth and thin edges, therefore Canny is the better edge detection method out of the two. However, the Prewitt edge detection method will yield different results, this method is outside the scope of this evaluation so its results will not be documented. (Othman et al. 2009. 133-135)

Task 3 – Simple Segmentation

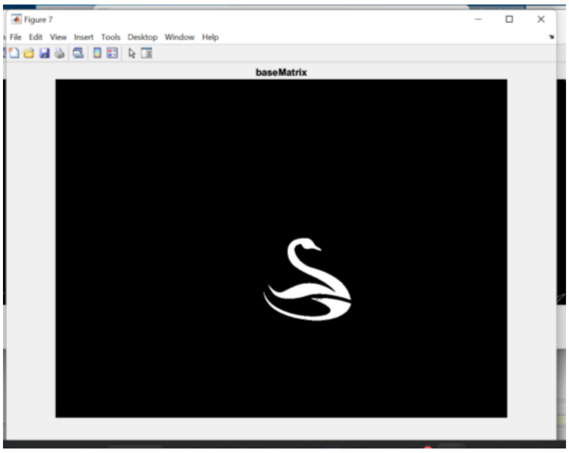


Figure 3. Feature Extraction.

For the simple segmentation implementation I have used the Connected Component analysis function bwconncomp to detect and enumerate the features of the input binarized image and store them in an array. I have then created a blank black matrix which will hold the features of the swan to for displaying output. Finally, I searched through the enumerated list of features until the swan body and tail were found, adding them to the blank matrix with the Boolean value True; Boolean values in a binarized image determine the colour of white(255) or black(0) based on a return of 0 or 1. See Figure 3 for the blank matrix with the swan features adding to it. (Durvasula, R.(n.d.). 1-2)

Task 4 – Swan Recognition

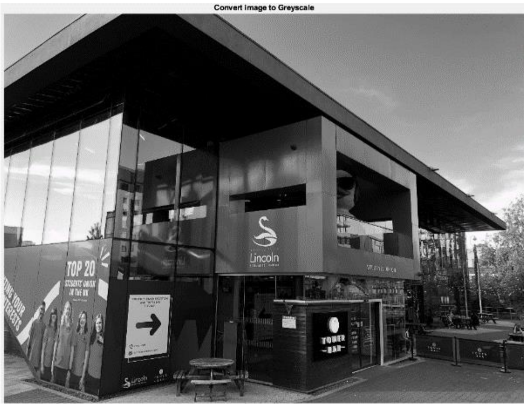


Figure 4. Grayscaled Image

As seen in figure 4, I have turned the image to grayscale so that it can be manipulated in the later stages using imreducehaze to reduce the effects of sunlight on the image and then prepare the image to be binarized later on.

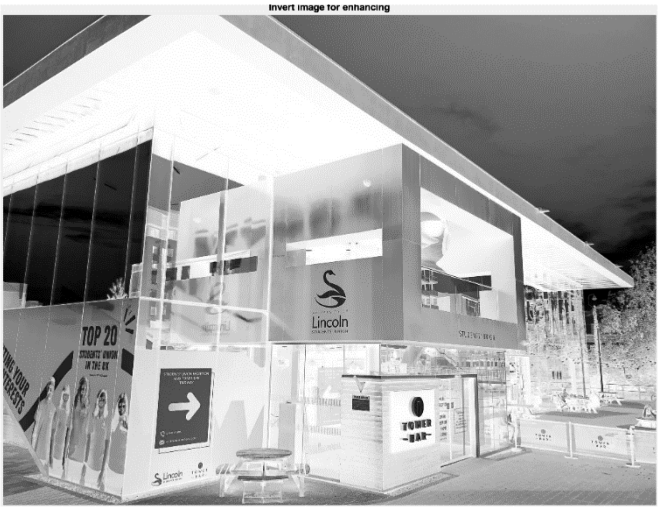


Figure 5. Enhanced image.

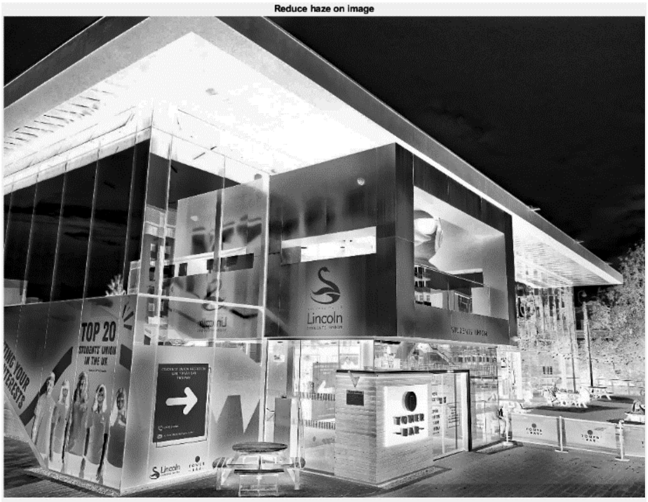


Figure 6. Haze Removed

In Figure 5 you will see that the image has been inverted so that the imreducehaze function can help to remove the effects of lighting glare and reflections on the image at hand. This is a necessary step so that the features can be correctly detected and extracted for a better functioning detection model. As seen in Figure 6, the glare of the light has been greatly dimmed as proof of the above.

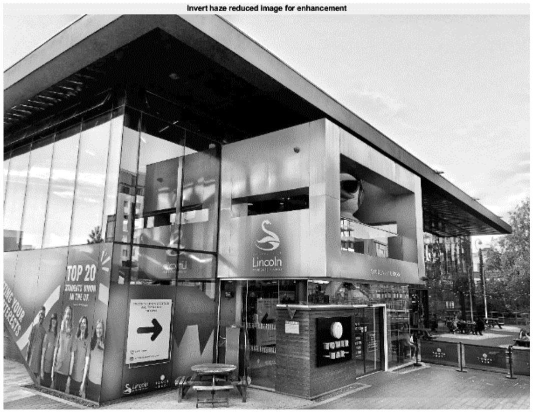


Figure 7. Enhanced image.

In figure 7, the image has been inverted back to its original grey scale after the haze has been reduced, evidently there is less effect of the light on the image in comparison to Figure 1 and the detection methods from further parts are less likely to be thrown off by such factors.

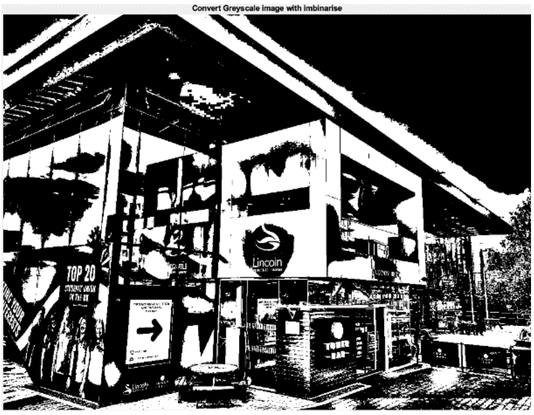


Figure 8. Binarise the image.

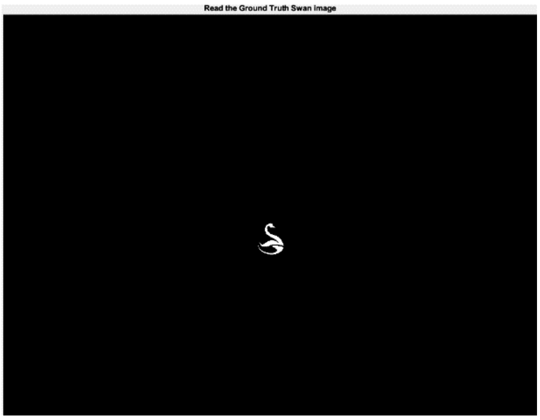


Figure 9. Read ground truth image.

In Figure 8 the image is binarized so that ground truth image, which is of the same type, can be directly compared to the ground truth image. In this exact example there is a lot of noise in the image, which will inevitably skew the accuracy of the detection methods accuracy, but the noise is different through every image, some having less will yield a better evaluation result.

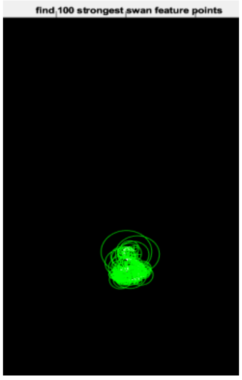


Figure 10.

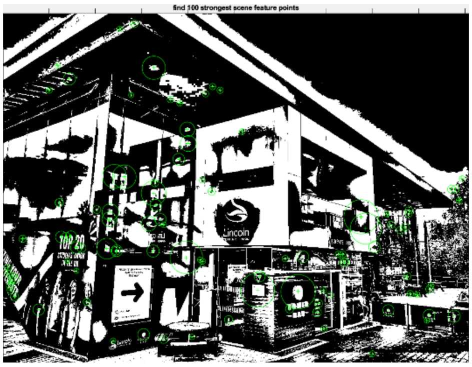


Figure 11.

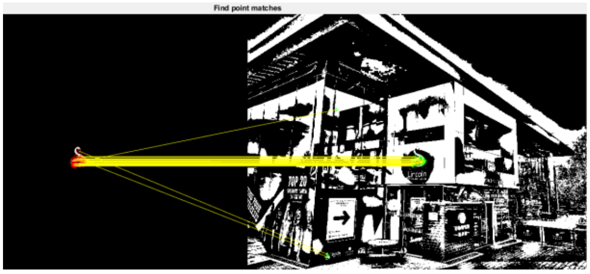


Figure 12.

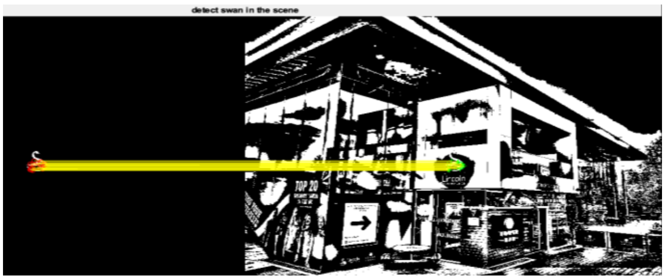
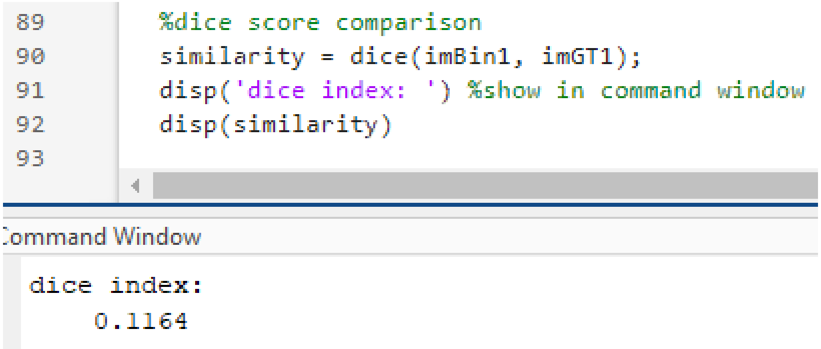


Figure 13.

In figures 10 and 11 you can see the 100 strongest features of each image highlighted with a bright green point, the first image being the ground truth then the other from above. Evidently, the noise from this particular example has affected the detection method so the comparison in Figures 12 and 13 will be affected. These points can be matched against one another as seen in the in the next 2 figures. Using the showMatchedFeatures and estgeotform2d functions I extracted the features and mapped the comparison. The next step is the evaluation of the methods. (Manjunath et al. 1996. 6-7)

Task 5 - Evaluation



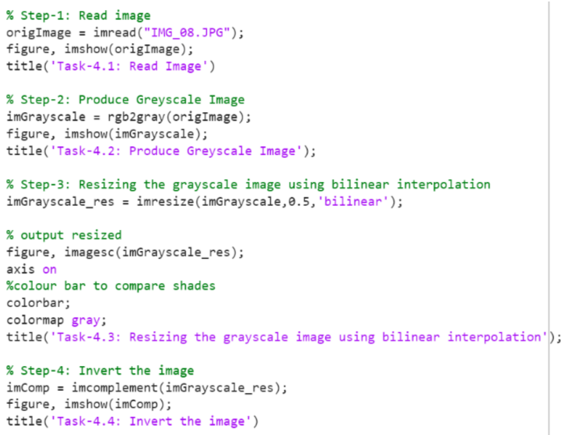
```
%dice score comparison
similarity = dice(imBin1, imGT1);
disp('dice index: ') %show in command window
disp(similarity)
```

Command Window

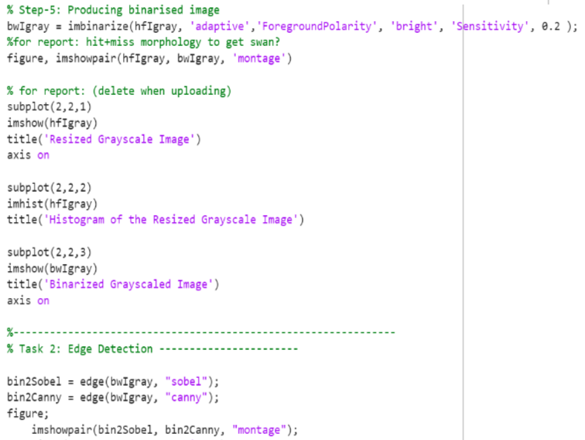
```
dice index:
    0.1164
```

The Dice score is used in this context to provide a pixel-based comparison between images that are input as ground truth and others that are being evaluated. This is a metric we can use to measure the accuracy of our swan recognition system and find out the Dice Score of said evaluation. In this context, I have prepared the images, including both normal and ground truth images, compared them with the dice() function and could have used the output to calculate the mean and standard deviation of this system. (Jha, S. 2019. 763) I have not completely implemented the dice score function as I intended; check the comments in the evaluation section for more on this. From my code, the dice score has been very low due to the incompleteness and lack of noise reduction in my implementation.

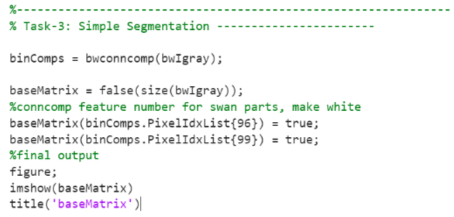
Appendix



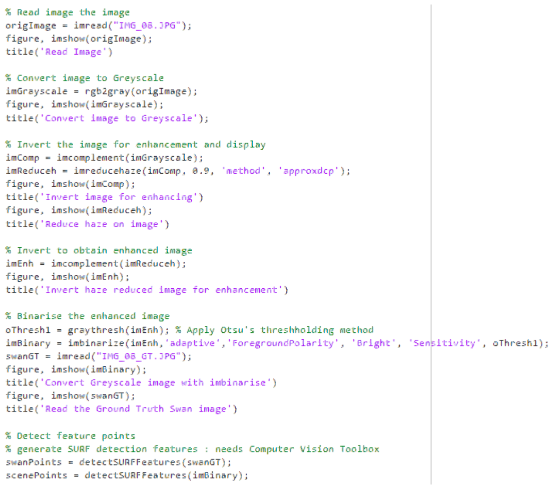
```
% Step-1: Read image
origImage = imread("IMG_08.JPG");
figure, imshow(origImage);
title('Task-4.1: Read Image')

% Step-2: Produce Greyscale Image
imGrayscale = rgb2gray(origImage);
figure, imshow(imGrayscale);
title('Task-4.2: Produce Greyscale Image');

% Step-3: Resizing the grayscale image using bilinear interpolation
imGrayscale_res = imresize(imGrayscale,0.5,'bilinear');

% output resized
figure, imagesc(imGrayscale_res);
axis on
%colour bar to compare shades
colorbar;
colormap gray;
title('Task-4.3: Resizing the grayscale image using bilinear interpolation');

% Step-4: Invert the image
imComp = imcomplement(imGrayscale_res);
figure, imshow(imComp);
title('Task-4.4: Invert the image')
```



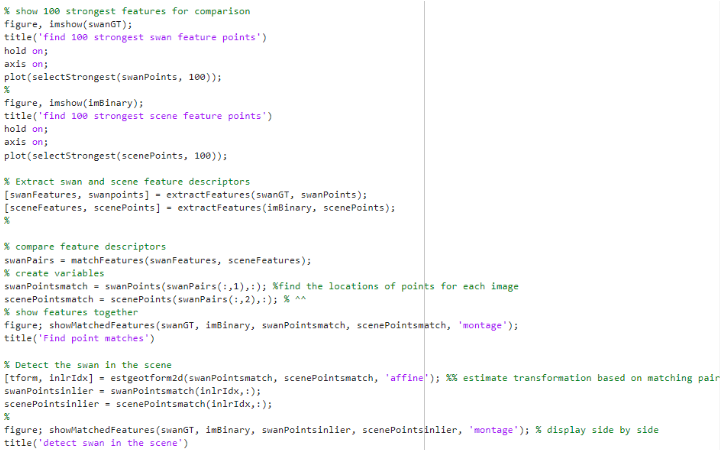
```
% Step-5: Producing binarised image
bwIgray = imbinarize(hfIgray, 'adaptive','ForegroundPolarity', 'bright', 'Sensitivity', 0.2 );
%for report: hit+miss morphology to get swan?
figure, imshowpair(hfIgray, bwIgray, 'montage')

% for report: (delete when uploading)
subplot(2,2,1)
imshow(hfIgray)
title('Resized Grayscale Image')
axis on

subplot(2,2,2)
imhist(hfIgray)
title('Histogram of the Resized Grayscale Image')

subplot(2,2,3)
imshow(bwIgray)
title('Binarized Grayscaled Image')
axis on

%-------------------------------------------------------------
% Task 2: Edge Detection ----------------------

bin2Sobel = edge(bwIgray, "sobel");
bin2Canny = edge(bwIgray, "canny");
figure;
    imshowpair(bin2Sobel, bin2Canny, "montage");
```



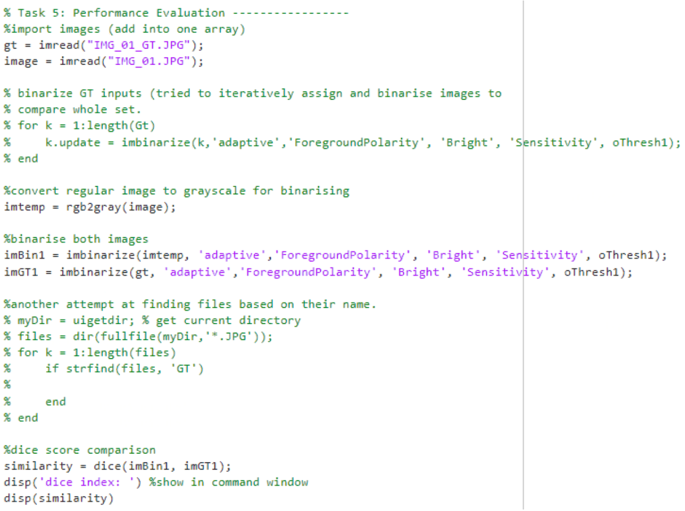
```
%-----------------------------------------------------------------
% Task-3: Simple Segmentation -----------------------

binComps = bwconncomp(bwIgray);

baseMatrix = false(size(bwIgray));
%conncomp feature number for swan parts, make white
baseMatrix(binComps.PixelIdxList{96}) = true;
baseMatrix(binComps.PixelIdxList{99}) = true;
%final output
figure;
imshow(baseMatrix)
title('baseMatrix')
```



```
% Read image the image
origImage = imread("IMG_08.JPG");
figure, imshow(origImage);
title('Read Image')

% Convert image to Greyscale
imGrayscale = rgb2gray(origImage);
figure, imshow(imGrayscale);
title('Convert image to Greyscale');

% Invert the image for enhancement and display
imComp = imcomplement(imGrayscale);
imReduceh = imreducehaze(imComp, 0.9, 'method', 'approxdcp');
figure, imshow(imComp);
title('Invert image for enhancing')
figure, imshow(imReduceh);
title('Reduce haze on image')

% Invert to obtain enhanced image
imEnh = imcomplement(imReduceh);
figure, imshow(imEnh);
title('Invert haze reduced image for enhancement')

% Binarise the enhanced image
oThresh1 = graythresh(imEnh); % Apply Otsu's threshholding method
imBinary = imbinarize(imEnh,'adaptive','ForegroundPolarity', 'Bright', 'Sensitivity', oThresh1);
swanGT = imread("IMG_08_GT.JPG");
figure, imshow(imBinary);
title('Convert Greyscale image with imbinarise')
figure, imshow(swanGT);
title('Read the Ground Truth Swan image')

% Detect feature points
% generate SURF detection features : needs Computer Vision Toolbox
swanPoints = detectSURFFeatures(swanGT);
scenePoints = detectSURFFeatures(imBinary);
```



```
% show 100 strongest features for comparison
figure, imshow(swanGT);
title('find 100 strongest swan feature points')
hold on;
axis on;
plot(selectStrongest(swanPoints, 100));
%
figure, imshow(imBinary);
title('find 100 strongest scene feature points')
hold on;
axis on;
plot(selectStrongest(scenePoints, 100));

% Extract swan and scene feature descriptors
[swanFeatures, swanpoints] = extractFeatures(swanGT, swanPoints);
[sceneFeatures, scenePoints] = extractFeatures(imBinary, scenePoints);
%

% compare feature descriptors
swanPairs = matchFeatures(swanFeatures, sceneFeatures);
% create variables
swanPointsmatch = swanPoints(swanPairs(:,1),:); %find the locations of points for each image
scenePointsmatch = scenePoints(swanPairs(:,2),:); % ^^
% show features together
figure; showMatchedFeatures(swanGT, imBinary, swanPointsmatch, scenePointsmatch, 'montage');
title('Find point matches')

% Detect the swan in the scene
[tform, inlrIdx] = estgeotform2d(swanPointsmatch, scenePointsmatch, 'affine'); %% estimate transformation based on matching pair
swanPointsinlier = swanPointsmatch(inlrIdx,:);
scenePointsinlier = scenePointsmatch(inlrIdx,:);
%
figure; showMatchedFeatures(swanGT, imBinary, swanPointsinlier, scenePointsinlier, 'montage'); % display side by side
title('detect swan in the scene')
```



```
% Task 5: Performance Evaluation -----------------
%import images (add into one array)
gt = imread("IMG_01_GT.JPG");
image = imread("IMG_01.JPG");

% binarize GT inputs (tried to iteratively assign and binarise images to
% compare whole set.
% for k = 1:length(Gt)
%     k.update = imbinarize(k,'adaptive','ForegroundPolarity', 'Bright', 'Sensitivity', oThresh1);
% end

%convert regular image to grayscale for binarising
imtemp = rgb2gray(image);

%binarise both images
imBin1 = imbinarize(imtemp, 'adaptive','ForegroundPolarity', 'Bright', 'Sensitivity', oThresh1);
imGT1 = imbinarize(gt, 'adaptive','ForegroundPolarity', 'Bright', 'Sensitivity', oThresh1);

%another attempt at finding files based on their name.
% myDir = uigetdir; % get current directory
% files = dir(fullfile(myDir,'*.JPG'));
% for k = 1:length(files)
%     if strfind(files, 'GT')
%
%     end
% end

%dice score comparison
similarity = dice(imBin1, imGT1);
disp('dice index: ') %show in command window
disp(similarity)
```

References:

-Bangare, S.L., Dubal, A., Bangare, P.S. and Patil, S.T., 2015. Reviewing Otsu's method for image thresholding. International Journal of Applied Engineering Research, 10(9), pp.21777-21783.

- Durvasula, R., Sai, K.N., Vivek, G.S. and KUMAR, M.K.P., Evaluation of Optical Character Recognition Using Feature Based Extraction Method.

- Jha, S., Kumar, R., Priyadarshini, I., Smarandache, F. and Long, H.V., 2019. Neutrosophic image segmentation with dice coefficients. *Measurement*, *134*, pp.762-772.

- Manjunath, B.S., Shekhar, C. and Chellappa, R., 1996. A new approach to image feature detection with applications. *Pattern Recognition*, *29*(4), pp.627-640.

-Othman, Z., Haron, H., Kadir, M.R.A. and Rafiq, M., 2009. Comparison of Canny and Sobel edge detection in MRI images. Computer Science, Biomechanics & Tissue Engineering Group, and Information System, pp.133-136.

-Patel, V. and Mistree, K., 2013. A review on different image interpolation techniques for image enhancement. International Journal of Emerging Technology and Advanced Engineering, 3(12), pp.129-133.