Ceng466 Take Home Exam 3

Necla Nur Akalın
2171148

Ayşenur Bülbül
2171403

I. Introduction

In this document, we explained the solutions for the third take home exam of the course Ceng466 Fundamentals of Image Processing. In this homework, we were expected use object counting, detection and image segmentation techniques on given images.

II. Part A

Object counting is one of the most important topics in image processing, in other words, in image analysis and in the first part of our homework, we were required to count objects(jets in our case) on the given images.

There are different approaches to find and count objects on images, in which some of them are deep or machine learning based. However, it is possible to detect object by only using differences in colors. Thus, we decided to use the traditional image processing approach.[2]

A. Detecting Objects

The first step in the traditional approach to detect objects and counting them is to convert RGB images to gray scale images. This step is necessary as it makes it easier to detect objects. To convert images to gray scale, after reading images, we used $rgb2gray$ MatLab function.

The second step is to threshold the image and making it a binary frame, in which 1's stand for the objects and 0's stand for the background. In our case, we had different 6 images to find jets in them. Every image had different color values, so we had to analyze every image one by one and then decide on a threshold value.

For the images $A1$, $A3$, $A5$ and $A6$, we only used one threshold value per image. For example, for image $A1$, if the pixel value is over 70, we labelled it as a background by putting 0 into the binary image to the corresponding pixel. This value is changed to 65 for image $A3$, 85 for image $A5$ and 15 for image $A6$. Cause of this change in the values is that not in every image, jets have the same colors. In image $A5$, for example, jets are brighter when compared with others. Thus, by using the same threshold value, one can't detect jets in $A5$. So, we used different threshold values on different images.

You can see the resulted binary image for $A1$ in $figure1$.

For image $A4$, on the other hand, we used two threshold values. Since jets in the image $A4$ have both very bright and very dark colors, we set the values between 45 and 246 as background and the rest as objects.

Different than others, in the image $A2$, we used a different method as it was not enough to look only one or two threshold values. Since in the half of the image there is a forest with dark colors and in the other half there is a background with light colors, it was necessary to divide image into two and decide on different threshold values for each part. Thus, for the upper part, we set it as a background if the pixel value is over 75 whereas for the lower part we it as a background if the pixel value is between 86 and 89.

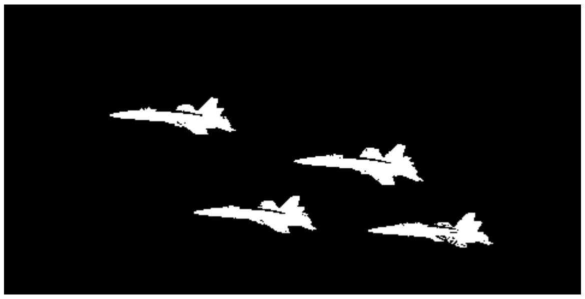
Fig. 1. A1 after thresholding

You can see the resulted binary image for $A2$ below.

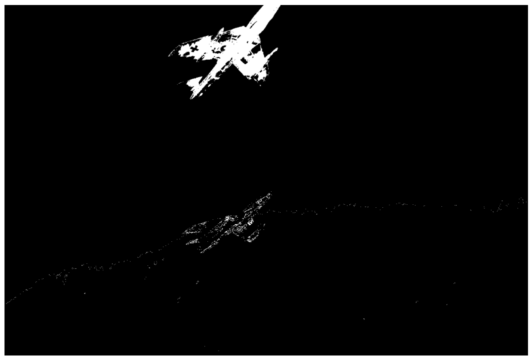
Fig. 2. A2 after thresholding

B. Counting Objects

After the detecting process, we got images with more components than the total jet numbers. This result was expected as there were some pixels on the background with same color values that we used to detect jets. Moreover, on the jets, there were some components which are not connected to each other. To prevent this and get the correct counts, we used 3 different functions.

1) Function imclose(): This function is needed to morphologically close the binary image, in other words, it fills the empty spaces of an object. It takes two inputs, one of them is the image itself and the other one is the pixel size of the largest gap to be filled.[3]

2) Function bwmorph(): This function allows us to implement morphological operations on binary images. We used this function with four different operation types in our scripts. One of them was the $'clean'$ type. This operation removes isolated pixels from the image, in other words, it deletes individual 1's which are are surrounded by 0's. The other one is the $'spur'$ operation which removes spur pixels. The third one is the $'bridge'$ operation, it connects unconnected pixels with a bridge of 1's. Lastly, we used the $'thicken'$ operation to make unconnected components to thicken and be 8-connected in binary images.[4]

3) Function deleteComp(): Different than other two MatLab function, this is the only function we've written to remove the components with a certain index. It takes two inputs, one of them is the image itself and the other one is the index of the component to be removed. To use this function, we first had to give indexes to the components with the MatLab function $bwlabel$.

Since every image was different, we had use these functions in different orders with different indexes and sizes. Except the image $A4$, we got correct counting results even though the shapes were not exactly the same with the shapes of the jets on the original image. In the image $A4$, since the first two jets overlap with each other, there was no way to count them as 2 separate objects, thus, instead of the correct jet number 4, our script gives 3 as the total jet number.

You can see the image $A4$ after the process of detecting and counting objects below.

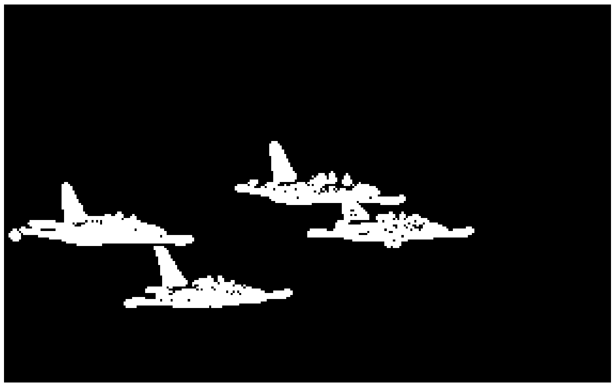

Fig. 3. A4 after counting process

III. PART B

In this part of the homework, we were expected to solve the image segmentation problem on Berkeley Image Segmentation Database. We can summarize image segmentation as the operation of partitioning an image into pieces of connected sets of pixels. There are different types of segmentation techniques based on different features of an image and in this part, we were expected use two of these techniques and compare them.

A. Threshold Based Segmentation

We used threshold based segmentation as our first algorithm. Thresholding is one of the simplest method of segmentation. In this algorithm, firstly we take the image and using $graythresh$ function we compute a global threshold from grayscale of that image using Otsu's method. Otsu's method separates pixels into two classes. These classes are foreground and background. Then it chooses a threshold that minimizes the intraclass variance of the thresholded black and white pixels.[1] We used output of this function to an input a new function: $im2bw$. This special MatLab function convert an image to a binary image based on a threshold. In our case it depends on thresholding done by $graythres$ function which performs Otsu's method.

As a result we get a binary image. To segment it we need to find the connected components in it. Thus, we used $bwconncomp$ function. This function takes a binary image and returns a structure which has fields of $Connectivity$, $ImageSize$, $NumObjects$ and $PixelIdxList$. Then we give this struct as an input to the $labelmatrix$ to create label matrix. After getting the label matrix all we need to do is coloring segments(connected components). For this purpose we used $label2rgb$ function which colors the labeled regions in the matrix. The colors are determined for each object based on the number of objects in the label matrix.

B. K-Means Clustering Based Algorithm

For our second algorithm, we used K-Means clustering based algorithm. We worked on image's gray scale. K-means clustering segments the objects from the background. It partitions the given data into K-clusters. The purpose of K-Means clustering is to minimize the sum of euclidean distances between all points and the cluster center. The steps we followed in this algorithm is firstly depending on the given K selecting K random points, centers for clusters. Then we assign each data to the closest random center points. We computed each distance and find new centers of clusters. Then reassigned data to the new centers. After getting clusters to colorize them we used $colors$ and $reshape$ functions.[2]

For K-means clustering algorithm we needed a parameter which is K. K specifies the number of clusters to detect. Correct choice of K is ambiguous and depends on the input image. Since our code takes images from given folder we need to decide on a generic K that can fit on all image set. Thus we choose K as 4 to divide to images 4 clusters. Given dataset mainly comprise images with less than 4 objects. This was the reason of choosing parameter K as 4.

C. Comparison

To compare the results of two segmentation algorithms, looking at the resulting images, we can say that K-means clustering based segmentation algorithm create more segments than threshold based segmentation algorithm. This may be the result of K-means finding the connected components and clusters better than threshold.

Below, you can see image $100080.jpg$ after two different segmentation methods.

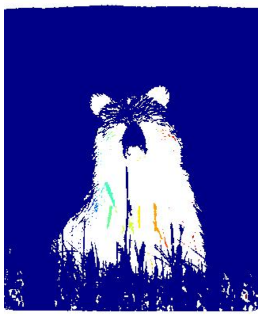
Fig. 4. Threshold based

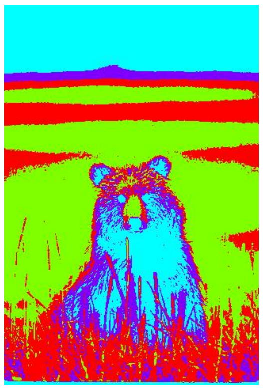
Fig. 5. K-Means clustering based

In figure 4 we can see the resulting image from threshold based algorithm. There is a distinct object segmentation. Bear is segmented fully from background. However the grass didn't get segmented.

In figure 5 we can see the resulting image from K-means clustering based algorithm. In the background the segmentation is unsuccessful. There is unnecessary segmentation(because we choose k as 4 for all inputs). Since original image's background is blurry, segmentation may be affected. However the bear and grass is segmented successfully. Also bear's face get segmented, which means algorithm successfully found connected components on the object also.

Both of these algorithms have their advantages and disadvantages on the given dataset. Threshold based algorithm gives a basic segmentation which can be not enough for some complicated cases. K-means clustering based algorithm gives more detailed segmentation which can be unnecessary for some cases.

IV. PART C

Similar to the first part of the homework, in this part, it was asked to detect certain objects from the given images and create a masked image in which the areas excluding objects are colored in black. Different than the first part, we were expected to detect apples in this part.

Applying the same algorithm for all 5 images and getting good results in which only apples are shown was harder than we thought. We tried different methods to get good results however, still in image $C1$ the stalk of the banana can be seen, in image $C4$, the apple is not seen as a whole and in image $C5$, some parts of pumpkins are still can be seen whereas some parts of apples are not seen.

A. Thresholding

The first solution came into our minds was to threshold the images just like we did in the first part of the homework. However, in the images, there are two different types of apples; green and red apples. Thus, in our script, we gave the range of the color of the apples as input to our helper function which is named $part3\_helper$. For example, if there are both green and red apples in an image, we called the helper function with the input number $'3'$, whereas we are calling the same function with the input number $'1'$ if there are only red apples.

For the images which are only contain red apples, the thresholding process was based on the red, green and blue channels of the original image, green channel of the k-means clustering based segmented image and finally it was based on the hsv of the original image which we get by using the MatLab function $rgb2hsv$. After creating a binary image, by comparing different values of different channels, we managed to set most of the parts outside the apples by 0's and the rest by 1's. However, it was not enough to get good results. There were so many little components that we couldn't get rid of. Thus, we wrote another function named $cleanComp$, which takes the image itself and some pixel number in which it deletes the components whose size is less than the pixel number. This function was necessary to mask little components as background. Other than this function, we once again used the MatLab functions $bwmorph()$, $imclose()$ and $bwlabel()$ to apply morhological operation. To get the final image, we element-wise multiplied the orginal image with the binary image. Below, you can see the image $C3$ after thresholding process.

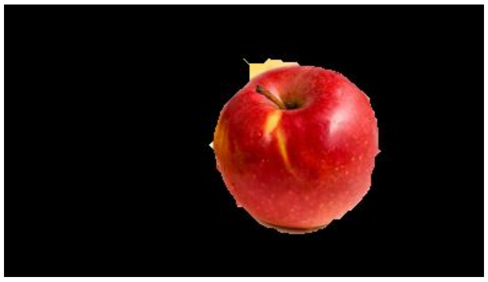
Fig. 6. C3 after thresholding process

In image $C4$, since the color values of the apple and the pear are pretty close to each other, it was harder to detect only the apple with the same algorithm that we used for $C2$ and $C3$. Thus, different than other outputs, in $C4$, the apple is not seen as a whole. Even though the pear is not seen, the quality of the output for the image $C4$ is still pretty low when compared with others. In $figure7$, you can see our output for the image $C4$.

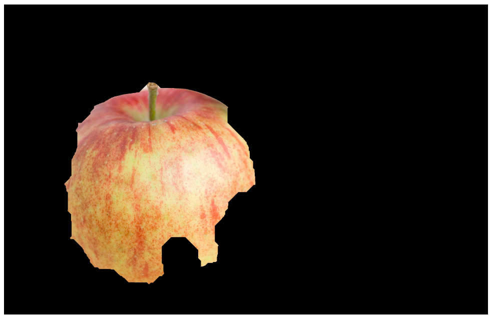
Fig. 7. C4 after thresholding process

For the image $C1$ which contains both red and green apples, we applied a different method. Instead of finding the apples, we focused on deleting the parts outside the apples. In this part, we focused only on the blue channel of the image and the hsv of the image. We set pixels of the binary image to 0 if the blue channel's value is between 85 and 135 and if it's hsv value is between 0.11 and 0.16. However, we still couldn't manage to delete the stalk of the banana.

B. Detecting Apples on Image C5

In all of 5 images, detecting apples in $C5$ was the hardest. The colors of the pumpkins and apples were so close to each other. Thus, with the algorithm that we used for the images $C2$, $C3$ and $C4$, we couldn't get a good result. You can see the corresponding frame in $figure8$.

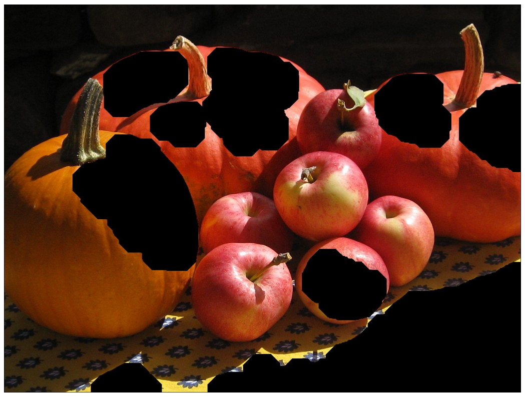

Fig. 8. C5 with old algorithm

In order to get better results, we decided to try different methods specified only for the image $C5$. One of them was doing edge detection, and removing the unnecessary parts from the image. Another method was to find the centers of the apples and drawing circles around those apples. However, in any of those experiments, we couldn't get a result in which pumpkins and the background is in total black and all apples are seen as a whole. Thus, we once again ended up doing thresholding on three channels of the image. You can see our final result below.

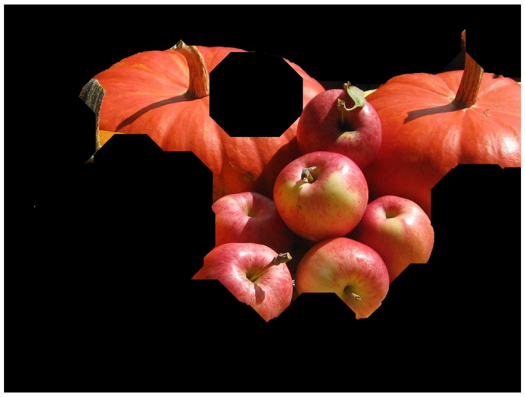

Fig. 9. C5 with new thresholding method

V. Conclusion

In this assignment, we worked on different object detection and counting algorithms and image segmentation techniques.

To do object counting and object detection, we discovered and applied morphological operations on binary images. We used thresholding to detect objects. We learned best ways to separate different objects and how to deal with unwanted components in an image.

In image segmentation part of the homework, we applied two different algorithms. We learned the differences of k-means based segmentation and thresholding based segmentation and why and how to use them.

In the final part of the homework, we improved ourselves in object detection and tried to use only one algorithm to find apples on all of the images. Even though not all the output images were not as good as we expected, we tried to do our best to detect the specified fruit from given images without using any machine learning and deep learning algorithms.

References

[1] Mathworks.com. (2020). Global image threshold using Otsu's method - MATLAB graytresh. [online] Available at: https://www.mathworks.com/help/images/ref/graythresh.html.

[2] Chauhan, Nagesh Singh. "Introduction to Image Segmentation with K-Means Clustering." Medium, Towards Data Science, 30 Aug. 2019, https://towardsdatascience.com/introduction-to-image-segmentation-with-k-means-clustering-83fd0a9e2fc3.