Fruit Recognition Using Color Analysis

Nehemiae George Roy EVD18I018

Sunny Kumar EVD18I025

Ayush Ranjan EVD18I006

Umesh Kumar Yadav EV18I027

June 26, 2020

1 Abstract

Computer guided classification of objects is a difficult task as every object has its own unique features. Yet, the recognition of objects, in general, have very good scope in several fields of science and engineering. Hence, using statistical and textural features, we have tried to make a solution for the classification of fruits.

2 Introduction

Visual object recognition is an interesting, unique and sometimes extremely difficult computational problem. The core problem is that each object in the world can cast an infinite number of different 2-D images onto the retina as the object's position, pose, lighting, and background vary relative to the viewer. Yet the brain solves this problem effortlessly. But it is a great challenge for computers with the ultimate goal of being able to achieve near human levels of recognition.

The fruit recognition system can be used in grocery stores to insert the fruit names automatically into the bill chosen by the customer. This solution can be used for classification of fruits in the supermarkets. This solution with some modifications can also be used for food grading or for separation of rotten fruits and ripened fruits which will reduce a lot of manual labour. This project with required modifications for the medical field can be used to identify cysts and tumours which can then be used to prevent dangerous cancers with proper medications. The Gray Level Co-occurrence Matrix (GLCM) we used in our solution is also being used in

almost all the medical imaging techniques such as X-rays, ultrasounds and many types of radiotherapies too.

The food industry uses color, size, shape, texture features for its applications but the performance of the methods proposed can be increased by increasing the number of features used. Both surface information (color and texture) and geometry information (size and shape) of food products in images plays a significant role in defect detection and class discrimination. Therefore, to capture more information about the quality of food from images, multiple kinds of features corresponding to the grading system of the food products should be proposed. However, in our proposed solution, we have stuck to only color analysis, and hence deal with the color and texture properties. For accurate recognition of fruit images from cameras, a number of challenges can be encountered. One type of fruit generally has significant variation in color and texture, depending on how ripe they are. For example, bananas range from being uniformly green, to yellow, to patchy and brown. Thus, analysing the color values alone, is insufficient. Color and texture of an image are the fundamental characteristics of the image and are widely used for visual categorization.

The process of classification based on color and texture involves extraction of useful information about the spectral properties of object surfaces and discovering the best match from a set of known descriptions or class models to implement the recognition task. This paper presents the design of a system, that can detect the

given image of a fruit, with minimal user input. In Section III, we give a brief literature surevey of the exisiting models, followed by the methodology research paper followed by us in Section IV. Subsequently, the block diagram, execution procedure, code and discussion are provided, concluding with our conclusions in Section XI.

3 Literature

Several approaches have been followed in order to employ computer vision in recognizing fruits, a few of which are mentioned below.

A method that combines four features analysis method shape, size and color, texture based method to increase accuracy of recognition, has been proposed[6]. The proposed method usees nearest neighbor classification algorithm (KNN). These methods classify and recognize the fruit images from the nearest training fruit example. Another paper[2] purpose a fruit classification system based on the color, shape and texture of the fruits. A dataset containing five different fruits was constructed using an ordinary camera. Many fruits have same color and shape but the chances of have all similar features are negligible. Different classifiers were experimented with to find the classifier which gives best results. All the fruits were analyzed on the basis of their color (RGB space), shape and texture and then classified using different classifiers to find the classifier that gives the best accuracy. Gray Level Co-occurance Matrix (GLCM) is used to calculate texture features. Best accuracy was achieved when support vector machines (SVM), was used.

Another project[5] is able to recognize the fruit based on the features like shape, colour, and texture. For this methodology, image segmentation is used to detect particular fruit. The project is implemented for both Real time and Non-Real

time. The proposed method has four stages: First is Pre-Processing and second is Feature Extraction and third is Segmentation and fourth Recognition. In case of NonReal time, the first stage is used to browse the image, while the second stage is extraction of the features from images using Grey Level Co-occurrence Matrix (GLCM), RGB and Color Histogram. System will convert the image from RGB to grayscale image for further processing. The color histogram represents the distribution of colors in an image. Since image is captured under different illumination condition. In the third stage, the three extracted image is obtained in the form of red, green and blue. In the fourth stage, the extracted features are used as input to Support Vector Machine (SVM) classifier. Then name of the fruit is output is obtained.

4 Methodology

In our project, we have followed the proposed paper[7]. Unlike other papers, we do not include shape analysis, and have yet modelled a statiscally viable system. The fundamental principle in the fruit recognition process, is to extract features, that are highly differentiable between different fruits, and then use these parameters to compare with an input image. We shall now discuss the methodology of our system. This paper mainly focuses on extracting various color and texture fearures of given fruit and collecting the extracted information in a database. Texture features are computed from the luminance channel 'V', and color features are computed from the chrominance channels 'H' and 'S' . It is mainly divided into three parts: Pre-processing, Feature extraction and Classification.

4.1 Pre-processing

One of the most imperative aspects of image processing, is background removal. When we are dealing with a system, that is highly dependent on some statistical properties, then large sample values in the population that are out of error, will cause a huge bias in the data, and lead to a wrong output. Hence, we need to efficiently and accurately remove the background pixels, for every fruit that is used for training the model, as well as for the input images. While, RGB images work well for display purposes they do not work efficiently enough, in image processing, because of the high correlation between the components R, G and B[3]. It becomes impossible, to segement out the background, since all three

channels will contribute to both the foreground and background. So, instead we turn to the HSV colour model.

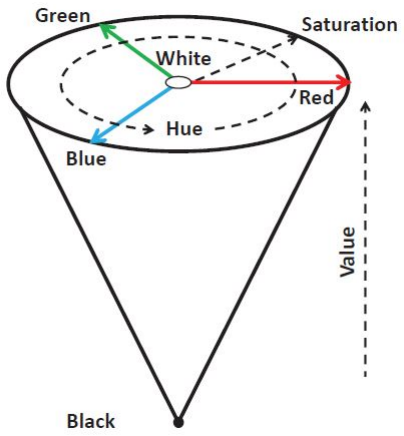


Figure 1: HSV Colour Model

In the HSV space the color is represented with the components hue (H), saturation (S) and value (V). Hue is the chromatic feature that describes a pure color; for instance, yellow, orange, red, etc. Saturation is a measure of how the hue is diluted in white light; value is the intensity or brightness of the color[3]. The following two properties make the HSV model very valuable in image processing:

1. The intensity component, value, is decoupled from the hue data - thus every lighting variation in the image is stored in the V channel alone.

2. The hue and saturation components emulate the human perception of color. With these features the HSV space becomes a useful tool to develop image processing algorithms based on some properties of the human color perception.

Since the H and S channel are least susceptible to lighting variations[7], we can threshold these channels to remove the background. Since, we are not using any machine learning algorithm, we require the user to use MATLAB's Colour Thresholder App, and threshold out the background. This is the only user assistance, required in this procedure.

4.2 Feature Extraction

Color is considered as a significant feature for image representation due to fact that color is invariant with respect to image translation, scaling, and rotation. Therefore, the first feature extraction method uses color characteristics to generate the feature vector for each fruit image in the dataset. Color parameters basically are extracted from H and S channel of the HSV color space. Here parameters like mean, standard deviation, skewness and kurtosis are derived from the H and S Channel. The advantage of using the HSV color space for calculating all these parameters, is that the H and S channel remains unaffected by lighting variation and hence it will be good for comparision purpose. As mentioned in the previous section, we are prevented from using the RGB Color space because all these parameters such as mean, standard deviation, skewness and kurtosis will differ greatly. For example if we have a bright apple with low shade and a apple with dark shade then the values will differ a lot for the same fruit making it difficult for comparision purpose. Therefore, the HSV color space is more suitable than RGB color space.

Skewness is a measure of symmetry, or more precisely, the lack of symmetry. A distribution, or data set, is symmetric if it looks the same to the left and right of the center point (mean). It measures the lack of symmetry in data distribution. The skewness for a normal distribution is zero, and any symmetric data should have a skewness near zero. Negative values for the skewness indicate data that are skewed left and positive values for the skewness indicate data that are skewed right. By skewed left, we mean that the left tail is long relative to the right tail. Similarly, skewed right means that the right tail is long relative to the left tail[1].

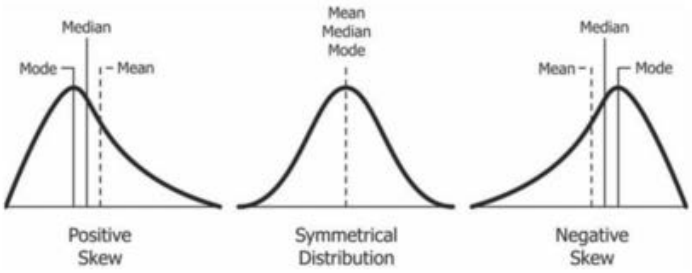


Figure 2: Distribution of Skewed Data

The formula for Skewness is given by:

$$Skewness = \frac{E(x-\mu)^3}{\sigma^3} \tag{1}$$

Kurtosis is a measure of whether the data are heavy-tailed or light-tailed relative to a normal distribution. That is, data sets with high kurtosis tend to have heavy tails. Data sets with low kurtosis tend to have light tails. A uniform distribution would be the extreme case[1]. The kurtosis of a normal distribution is 3, and if the tails are heavy the kurtosis is more than 3, and if the tails are light, then the

kurtosis is less than 3.

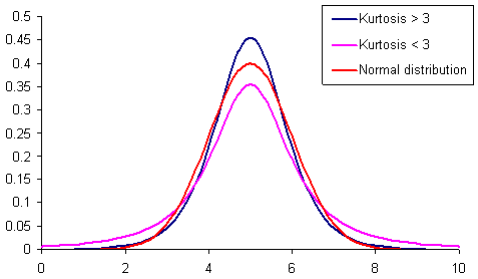


Figure 3: Kurtosis of a Dataset

The formula for Kurtosis is given by:

$$Kurtosis = \frac{E(x-\mu)^4}{\sigma^4} \tag{2}$$

Next are the texture features. Texture is a repeating pattern of local variations in image intensity. In statistical texture analysis, texture features are computed from the statistical distribution of observed combinations of intensities at specified positions relative to each other in the image. According to the number of intensity points (pixels) in each combination, statistics are classified into first-order, second order and higher-order statistics. The Gray Level Coocurrence Matrix (GLCM) method is a way of extracting second order statistical texture features[4]. GLCM is created in four directions with the distance between pixels as one. Texture features are extracted from the statistics of this matrix. A normalized GLCM is composed of probability values, and expresses the probability of the pairs of pixels in direction having a particalar gray level. The GLCM we employ is a symmetric matrix and its size is determined by the image gray-level. In this system, four

texture features are considered. They include contrast, energy, cluster shade and cluster prominence.

Contrast tells whether there are more adjacent pixels with the same grey level (less contrast) or adjacent pixels with highly different grey levels (more contrast). We know that the elements along the diagonal (left diagonal) of GLCM, correspond to the pairs of pixels with the same grey level. As we move away from the diagonal (either to the top right or bottom left), we are dealing with pixels with increasing difference in grey level. So somehow in the summation we must include a 'weight' that increases as we move away from the diagonal - hence we multiply by $(i-j)^2$.

$$Contrast = \sum_{i,j=0}^{N-1} P_{ij}(i-j)^2 \tag{3}$$

In energy, the weight used is the term P_{ij} itself. If there are any common combinations of pixel grey levels (more orderliness), then energy will be high.

$$Energy = \sum_{i,j=0}^{N-1} P_{ij}^2 \tag{4}$$

Cluster shade is a measure of the skewness and uniformity of the GLCM. A higher cluster shade implies greater asymmetry about the mean. This is purely a statistical measure applied onto the GLCM.

$$Shade = \sum_{i,j=0}^{N-1} (i+j-2\mu)^3 P_{ij} \tag{5}$$

Cluster prominence is a measure of the skewness and asymmetry of the GLCM. A higher value implies more asymmetry about the mean while a lower value indicates a peak near the mean value and less variation about the mean.

$$Prominence = \sum_{i,j=0}^{N-1} (i + j - 2\mu)^4 P_{ij} \tag{6}$$

4.3 Classification

In the classification phase, for the test fruit image, color and texture features are derived as that of the training phase and compared with corresponding feature values, stored in the feature library. The classification is done using the Minimum Distance Criterion (KNN method). The image from the training set which has the minimum distance when compared with the test image says that the test image belongs to the category of that training image[7].

$$Distance = \sqrt{(prop1_{input} - prop1_{dataset})^2 + ... + (prop_{n_{input}} - prop_{n_{dataset}})^2} \tag{7}$$

To summarize, we will use a total of 12 properties (4 from H channel, 4 from S channel and 4 from V channel) from each input image, to compare against a developed dataset.

5 Work Done

1. Theoretical Research

    - Background Removal - Nehemiae G. Roy
    - Viability of Area Based Classification - Umesh Kumar
    - Statistical Properties - Sunny Kumar
    - Textural Properties - Umesh Kumar, Nehemiae G. Roy

2. Code Development

    - Colour and Textural Parameters Calculator - Sunny Kumar
    - Data Set Preparation - Ayush Ranjan
    - Training Model Framework - Ayush Ranjan
    - Testing and Training Model Development - Nehemiae G. Roy

3. Database Training

    - Image Collection - Ayush Ranjan
    - Training - Nehemiae G. Roy

4. Report

    - Abstract and Introduction - Ayush Ranjan
    - Literature Research - Umesh Kumar
    - Methodology - Sunny Kumar

- Block Diagram, Procedure, Code and Conclusions - Nehemiae G. Roy
- Report Compiliation - Nehemiae G. Roy

5. Presentation - Umesh Kumar, Sunny Kumar

6. Video - Nehemiae G. Roy

6 Block Diagram

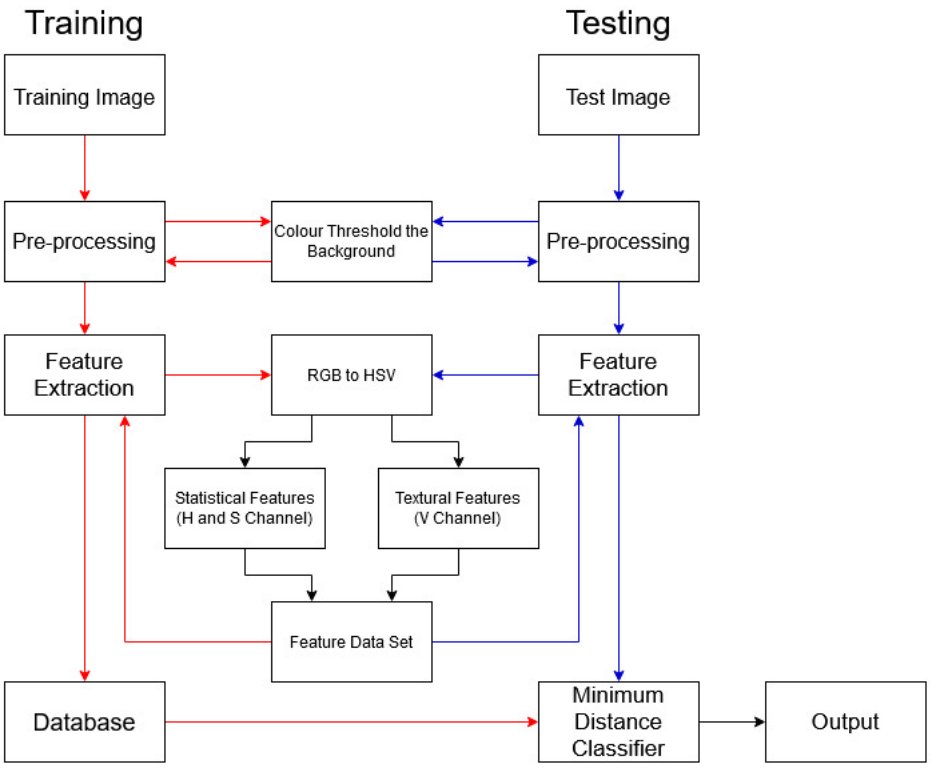


Figure 4: Block Diagram Model of the Project

The system depicted above, consists of 2 phases a training phase and a testing phase. The former is used to build the dataset that will then be used for the classification of the latter. For training the system we have chosen 4 fruits - Apple, Banana, Guava and Strawberry. A total of 33 images were used for training and a total of 33 images were used for testing the system. The flow of code is now described through the block diagram.

In the training phase, we take in an input image, and threshold out the background using the HSV color scheme. The user must perform this operation using MATLAB's Colour Thresholder App. Once this RGB image is sent back to the system, the extracted foreground is sent for feature extraction. The image is first converted to the HSV color scheme, and then stastistical properties of Mean, Standard Deviation, Skewness and Kurtosis are calculated from the the H and S channel. From the V channel, the GLCM is calculated and then textural features of Contrast, Energy, Cluster Shade and Cluster Prominence are calculated. All these features are then sent back to the system, where they stored to a table (written to a .txt file).

An input image to be tested, is essentially processed the same way. The user removes the background and the system calculates the necessay properties. These are then sent to the minimum distance classifier function, which calculates the Euclidean distance of the properties of the input image with the properties of each image stored in the table. The data entry for which the distance is minimum, must be fruit that the input image is - and hence this name is written to the terminal.

7 Procedure

The following steps are to correctly detect one particular fruit beloning to Apple, Banana, Guava and Strawberry.

Step 1: Run the FruitMain.m file and fill in the file location of the required image to be tested.

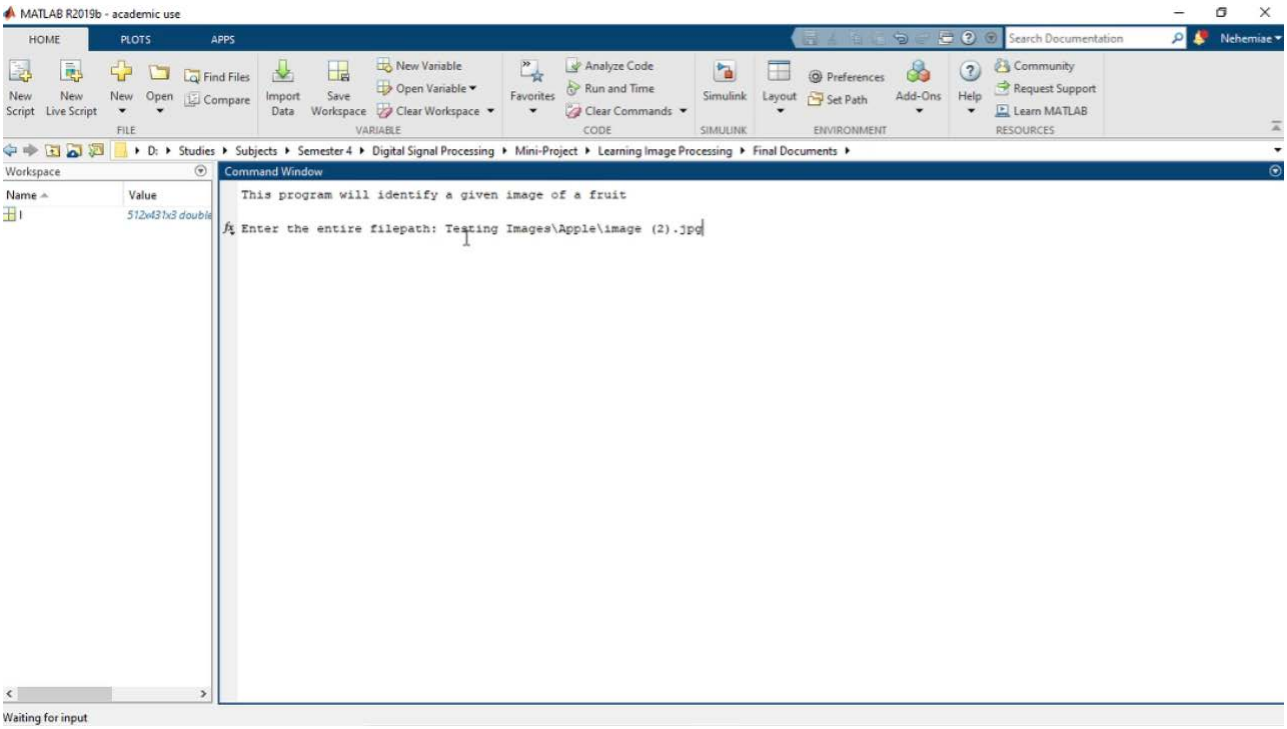


Figure 5: Read the input image

Step 2: Read the instructions carefully and then enter 'Y' at the prompt.

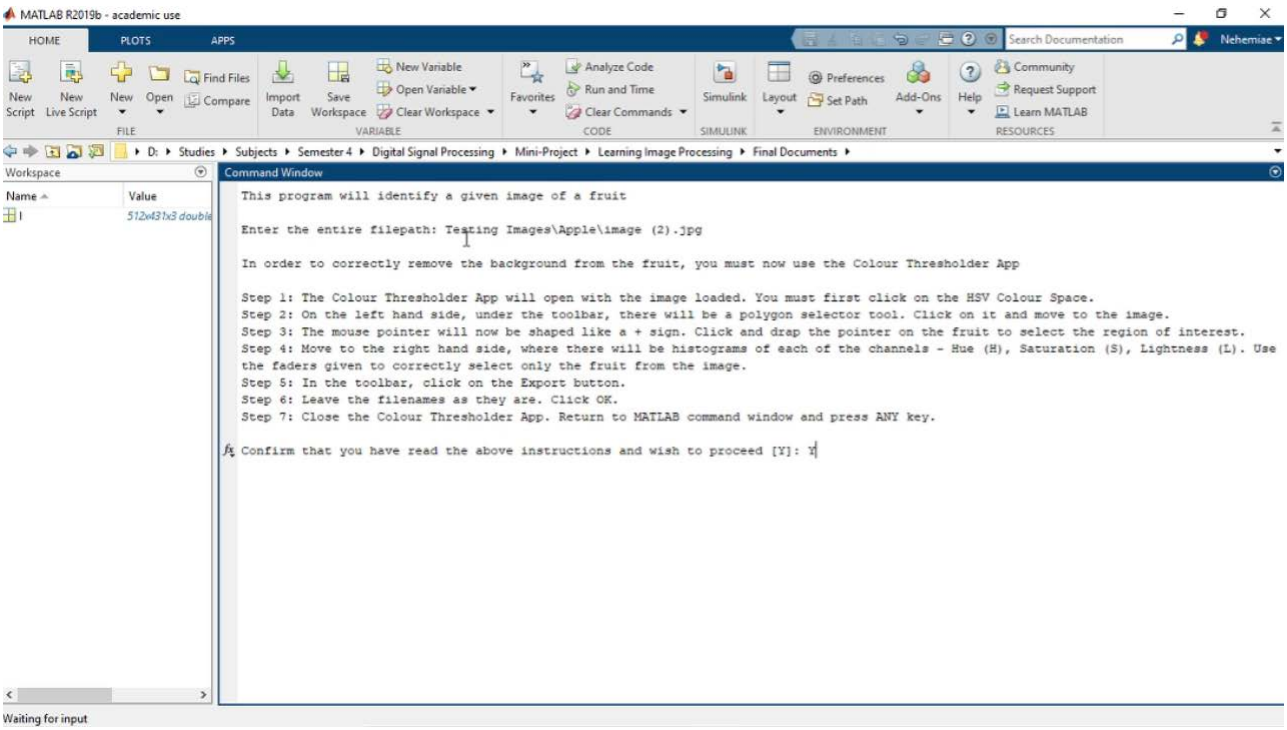


Figure 6: Read the instructions

Step 3: Once the Colour Thresholder App has opened, select the HSV Colour Scheme.

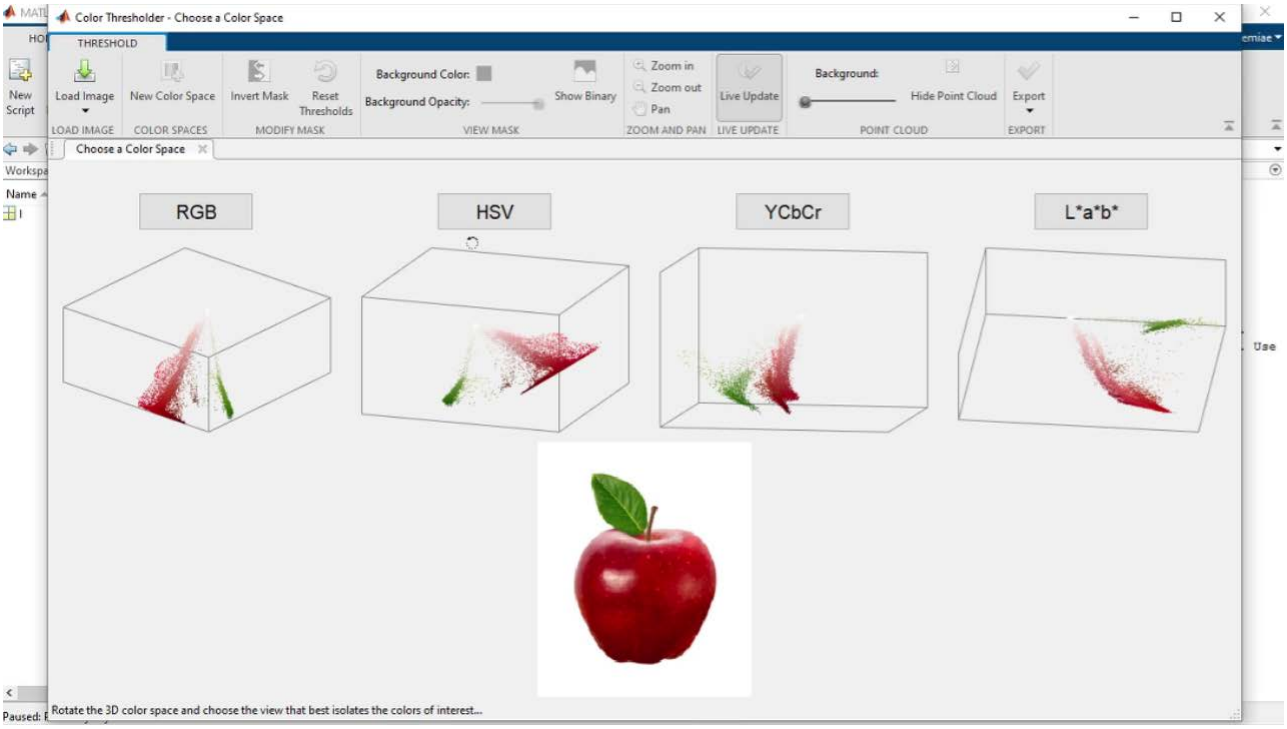


Figure 7: Select HSV Colour Scheme

Step 4: You are now in the HSV thresholder of the Colour Thresholder App. There is a polygon tool at the top left corner under the toolbar that can be used to draw any sahpe on the rgion of interest (ROI), and MATLAB will automatically set the required thresholds.

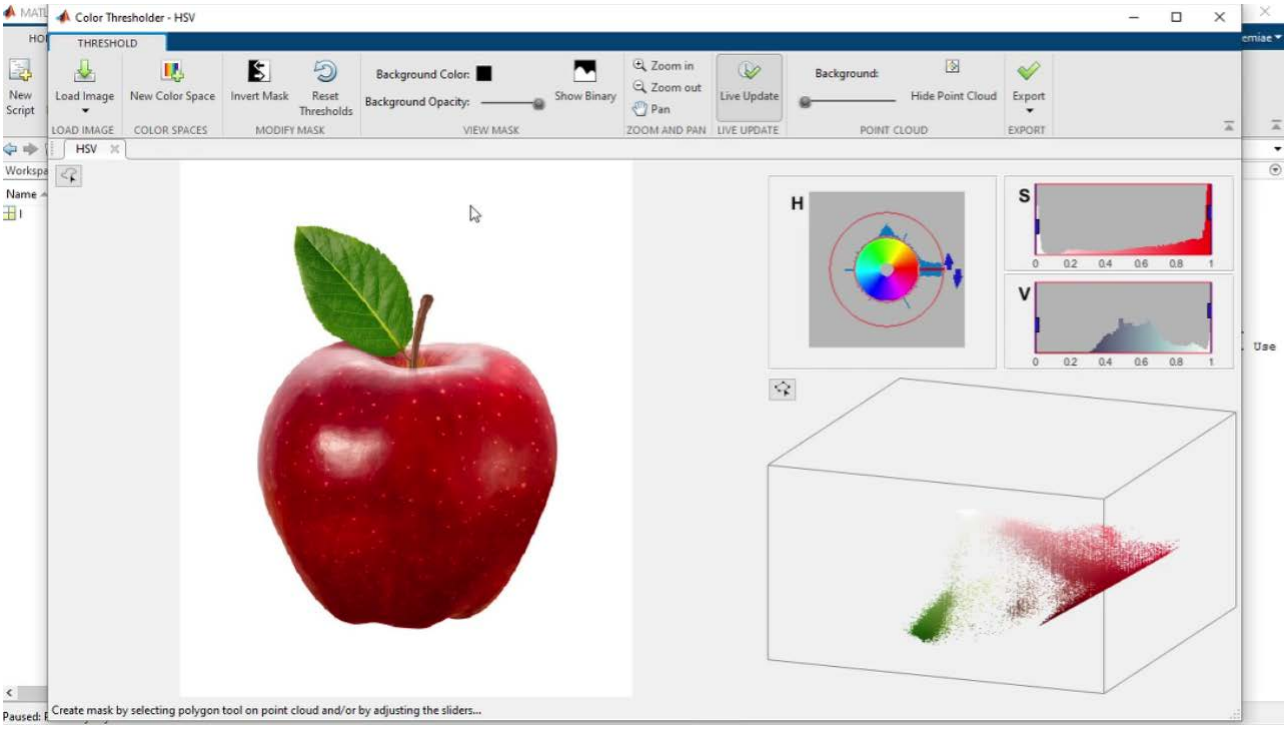


Figure 8: Colour Thresholder App

Step 5: Using the polygon tool, select the fruit alone, and ensure that the background turns to black. You can refine the selection using the sliders for each channel on the right.

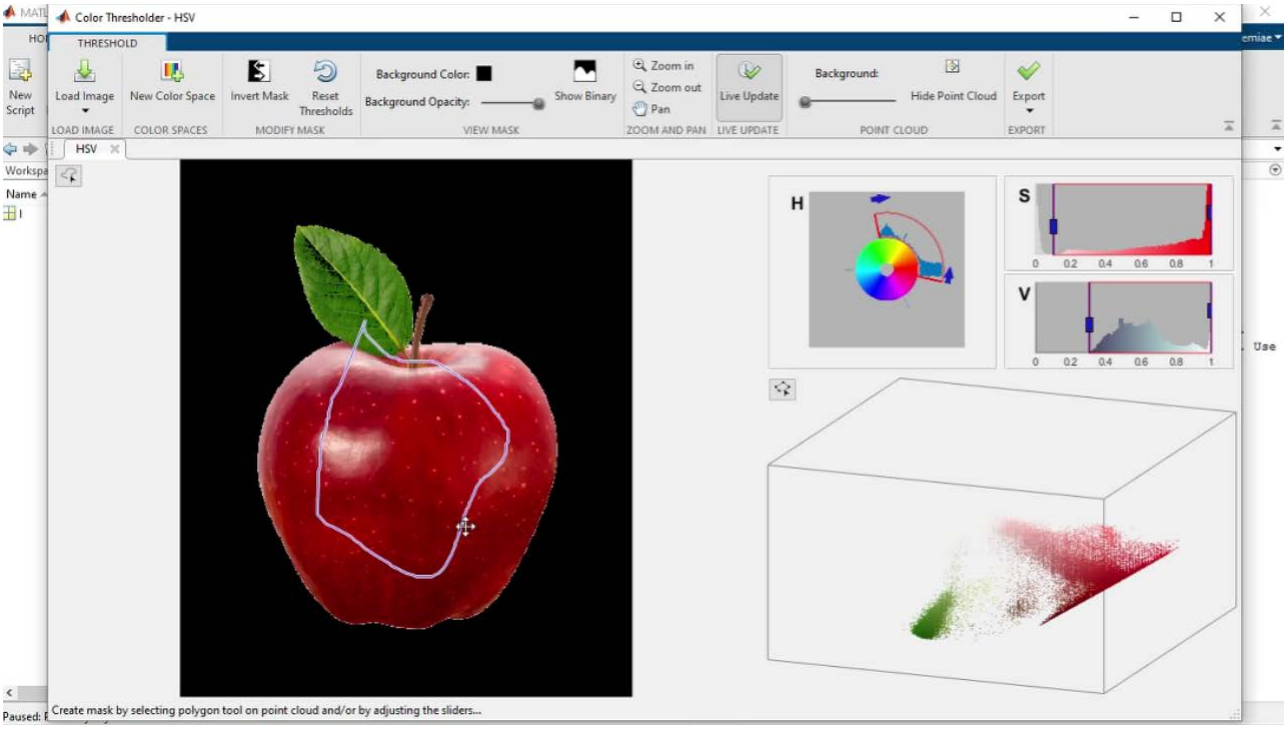


Figure 9: Threshold out the Image

Step 6: Click on Export in the toolbar. Leave the filenames as they are and return to the MATLAB terminal.

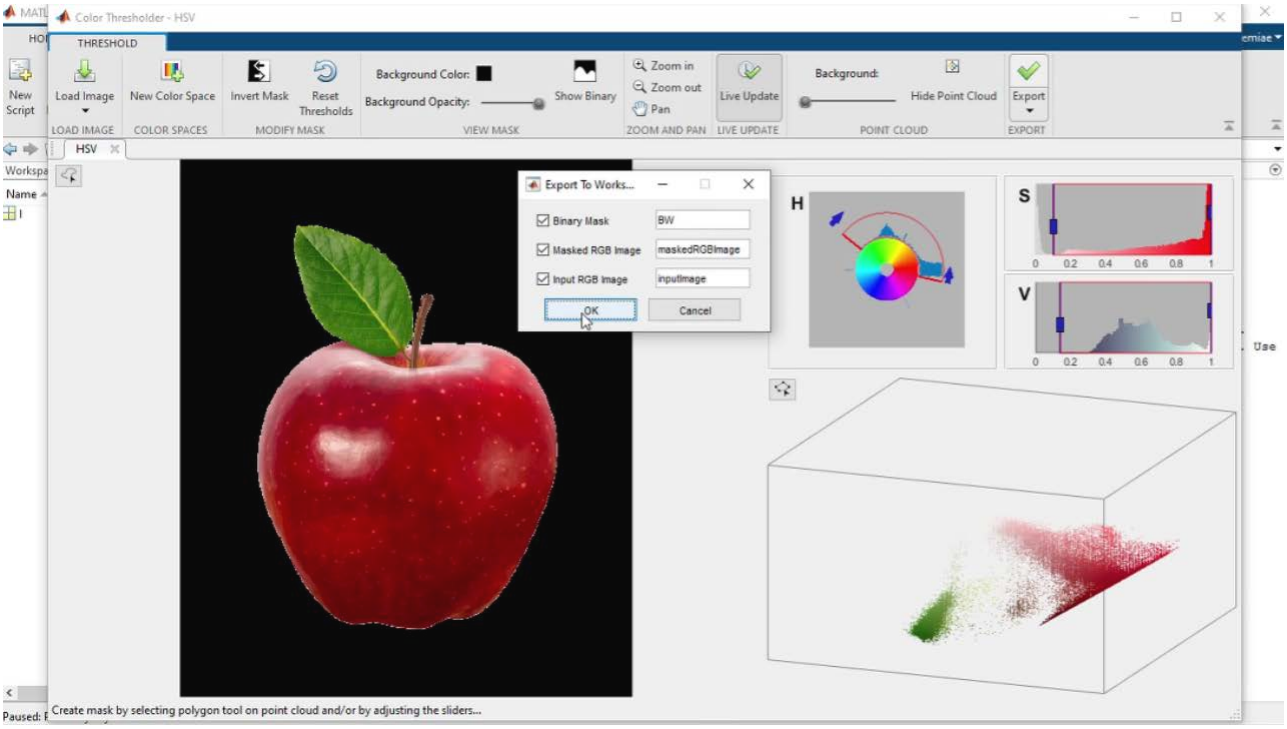


Figure 10: Threshold out the Image

Step 7: Press ANY key while at the terminal, to awaken MATLAB to start processing the image.

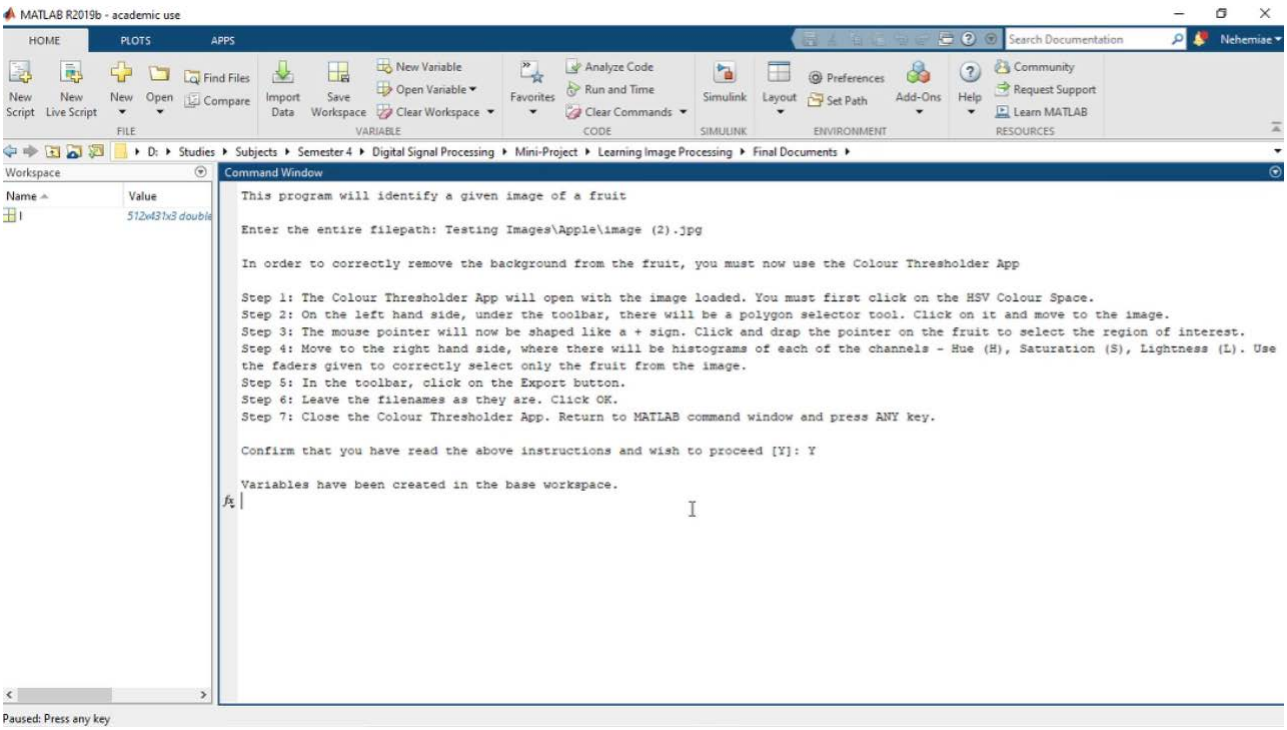


Figure 11: Press ANY key to awaken MATLAB

Step 8: The output image showing the mask used to threshold the image, along with the thresholded image are displayed.

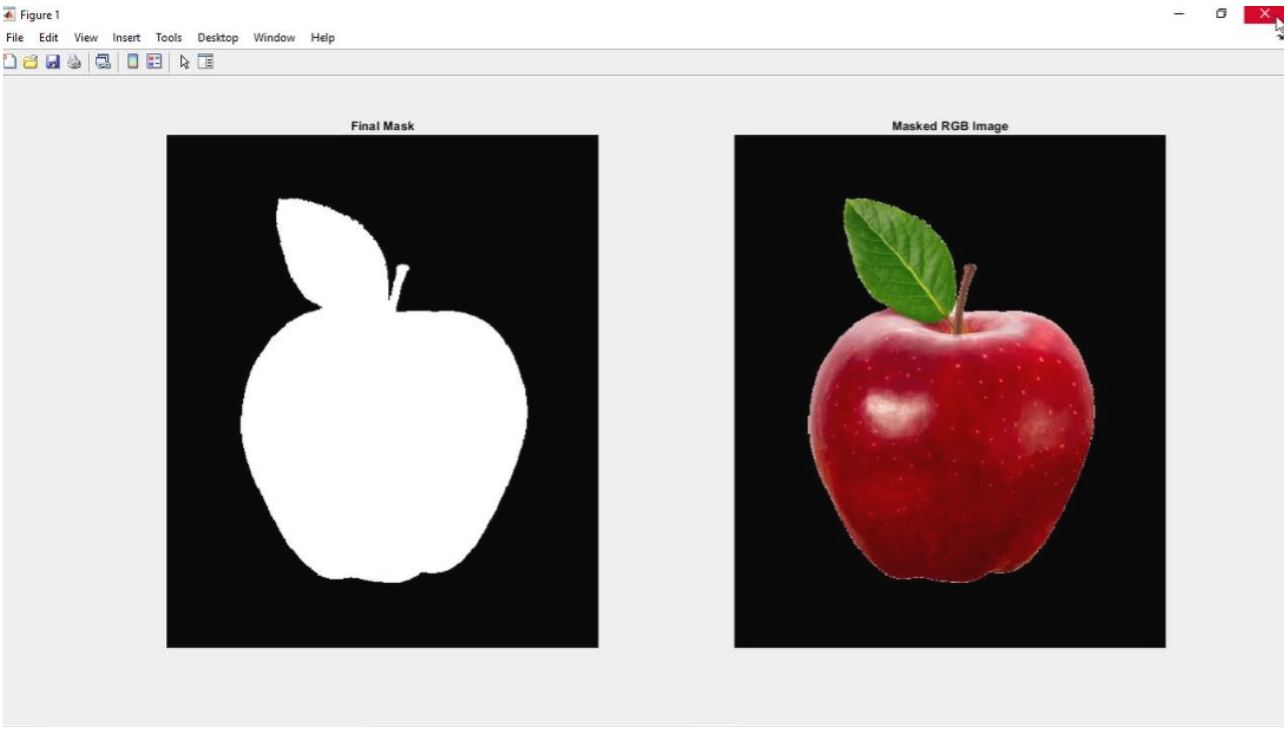


Figure 12: Output Images

Step 9: The result is written in the terminal.

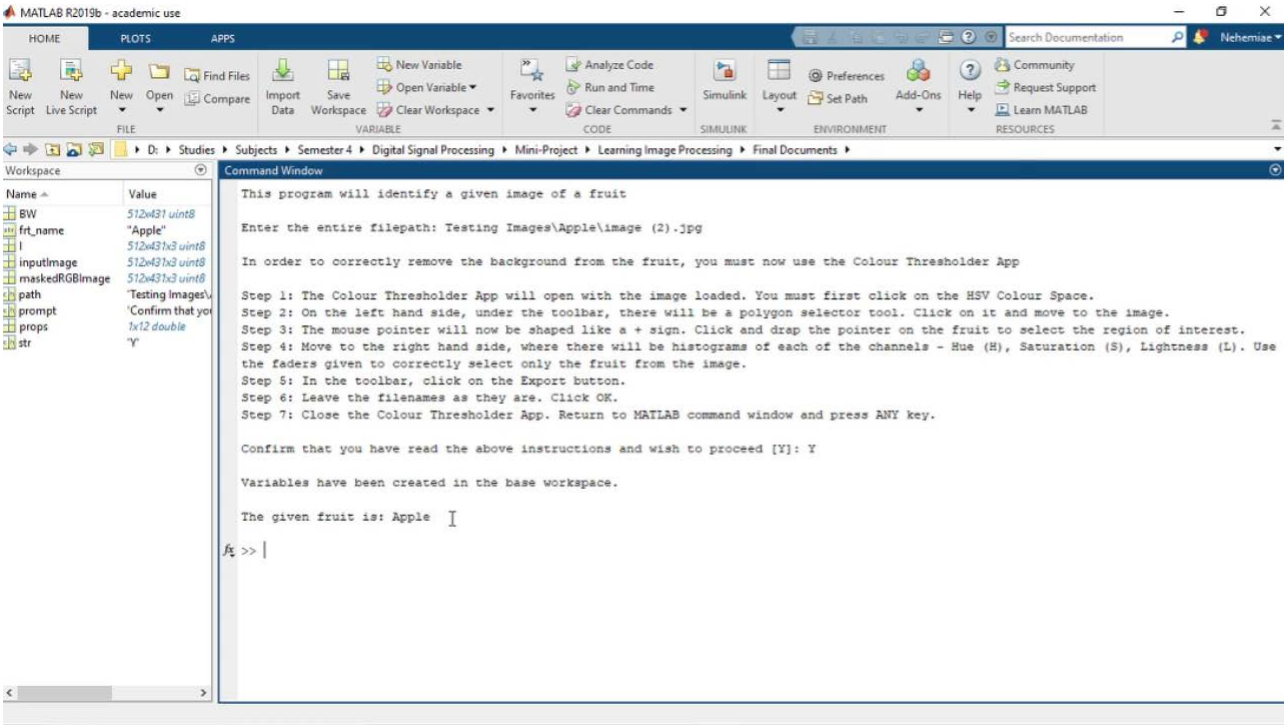


Figure 13: Output Result

8 Code

The code included in this section, are the only m files needed for running a simulation. The codes used for database training are included in the Appendix.

The file FruitMain.m is the main file in the program and all calculations go through it. The background extraction and final output also takes place here.

```
%This program is used to identify a given image of a
   fruit (Apple, Banana,
%Guava or Strawberry), by calculating certain
   statistical and textural
%properties on the color channels of the image, and
   then comparing them
%with the mean values from a table.

clear;
clc;

disp('This program will identify a given image of a
   fruit');
disp(' ');
path = input('Enter the entire filepath: ', 's');
I = imread(path);
```

```
disp(' ');

disp('In order to correctly remove the background from
    the fruit, you must now use the Colour Thresholder
    App');
disp(' ');
disp('Step 1: The Colour Thresholder App will open
   with the image loaded. You must first click on the
   HSV Colour Space.');
disp('Step 2: On the left hand side, under the toolbar
   , there will be a polygon selector tool. Click on
   it and move to the image.');
disp('Step 3: The mouse pointer will now be shaped
   like a + sign. Click and drap the pointer on the
   fruit to select the region of interest.');
disp('Step 4: Move to the right hand side, where there
    will be histograms of each of the channels - Hue (
   H), Saturation (S), Lightness (L). Use the faders
   given to correctly select only the fruit from the
   image.');
disp('Step 5: In the toolbar, click on the Export
   button.');
disp('Step 6: Leave the filenames as they are. Click
```

```
   OK.');
disp('Step 7: Close the Colour Thresholder App. Return
    to MATLAB command window and press ANY key.');
disp(' ');
prompt = 'Confirm that you have read the above
   instructions and wish to proceed [Y]: ';
str = input(prompt,'s');
while(str ~= 'Y')
    str = input(prompt,'s');
end
disp(' ');

%Background Removal
colorThresholder(I);
pause; %Wait until the user presses any key
BW = imfill(BW, 'holes'); %Fills up any holes in the
   mask
BW = bwareafilt(BW, 1); %Get the largest object
figure;
subplot(1,2,1);
imshow(BW);
title('Final Mask', 'FontSize', 10);
BW = cast(BW, 'like', I); %Convert the mask into an
```

```
   image with 3 channels - just like the input image
maskedRGBImage = bsxfun(@times, I, BW); %Multiply our
   input image and the mask, to get the fruit alone
subplot(1,2,2);
imshow(maskedRGBImage);
title('Masked RGB Image', 'FontSize', 10);

%Calculating the Properties
props = propcalc(maskedRGBImage);

%Identifying the Fruit
frt_name = mindistcalc(props);

fprintf('\nThe given fruit is: %s\n\n', frt_name);
```

The function propcalc.m calculates the 12 properties of every image passed to it. It returns a vector with the mentioned properties.

```
%This function will calculate certain statistical and
   textural properties
%of an input image, and return them.

function props = propcalc(RGB)
HSV = rgb2hsv(RGB);
```

```
props = zeros(1,12);

%H and S Channel:
k = 1;
for i = 1:2
    [~,~,foregnd_pxls] = find(HSV(:,:,i)); %Get the
       foreground pixels of H or S channel
    props(k) = mean(foregnd_pxls);
    props(k+1) = std(foregnd_pxls);
    props(k+2) = skewness(foregnd_pxls,0);
    props(k+3) = kurtosis(foregnd_pxls,0);
    k = k + 4;
end

%V Channel:
GRAY = rgb2gray(RGB);
glcm = zeros(8,8,4);
glcm(:,:,1) = graycomatrix(GRAY,'Offset',[0 1],'
   Symmetric',true); %For horizontal 0 degree
glcm(:,:,2) = graycomatrix(GRAY,'Offset',[-1 -1],'
   Symmetric',true); % For 135 degree left diagonal
glcm(:,:,3) = graycomatrix(GRAY,'Offset',[-1 0],'
   Symmetric',true); % For vertical 90 degree
```

```
glcm(:,:,4) = graycomatrix(GRAY,'Offset',[-1 1],'
   Symmetric',true); % For 45 degree right diagonal

%Remove all counts from each GLCM, that calculate
   adjacency with background
%pixels
for k = 1:4
    glcm(1,:,k)=0;
    glcm(:,1,k)=0;
end

%Get the Contrast, Correlation, Energy and Homogeneity
    properties. We make
%sure that the GLCM calculations are rotationally
   invariant.
glcm1_props = graycoprops(glcm(:,:,1));
glcm2_props = graycoprops(glcm(:,:,2));
glcm3_props = graycoprops(glcm(:,:,3));
glcm4_props = graycoprops(glcm(:,:,4));
props(9) = (glcm1_props.Contrast + glcm2_props.
   Contrast + glcm3_props.Contrast + glcm4_props.
   Contrast)/4;
props(10) = (glcm1_props.Energy + glcm2_props.Energy +
```

```
    glcm3_props.Energy + glcm4_props.Energy)/4;

%Normalize all the GLCMs
for k = 1:4
    glcm(:,:,k) = glcm(:,:,k)./sum(glcm(:,:,k),'all');
end

%Get the Cluster Shade and Cluster Prominence
   propoerties
glcm_mean = 0;
for k = 1:4
    for i = 1:8
        for j = 1:8
            glcm_mean = glcm_mean + (i)*glcm(i,j,k);
        end
    end
end
glcm_mean = glcm_mean/4;

for k = 1:4
    for i = 1:8
        for j = 1:8
            props(11) = props(11) + (((i + j - 2*
```

```
                glcm_mean)^3).*glcm(i,j,k));
            props(12) = props(12) + (((i + j - 2*
                glcm_mean)^4).*glcm(i,j,k));
        end
    end
end
props(11) = props(11)/4;
props(12) = props(12)/4;

end
```

The function mindistcalc.m calculates the which data entry from the database has the least distance (in terms of calculated parameters) from the input image. It returns a string with the name of the image.

```
function frt_name = mindistcalc(props)
table = readtable('PropTable.txt');
dist = zeros(1,max(table.S_No));

for i = 1:max(table.S_No)
    index = i+1;
    dist(i) = sqrt(((props(1)-table.H_Mean(index))^2)
       + (props(2)-table.H_SD(index))^2 + (props(3)-
       table.H_Skewness(index))^2 + (props(4)-table.
```

```
        H_Kurtosis(index))^2 + (props(5)-table.S_Mean(
        index))^2 + (props(6)-table.S_SD(index))^2 + (
        props(7)-table.S_Skewness(index))^2 + (props(8)
        -table.S_Kurtosis(index))^2 + (props(9)-table.
        V_Contrast(index))^2 + (props(10)-table.
        V_Energy(index))^2 + (props(11)-table.V_Shade(
        index))^2 + (props(12)-table.V_Prominence(index
        ))^2);
end

[~,pos] = min(dist);
frt_name = string(table.Fruits(pos+1));
end
```

9 Results

The output image showing the mask used to threshold the image, along with the thresholded image are shown below along with a screenshot of the terminal with the output.

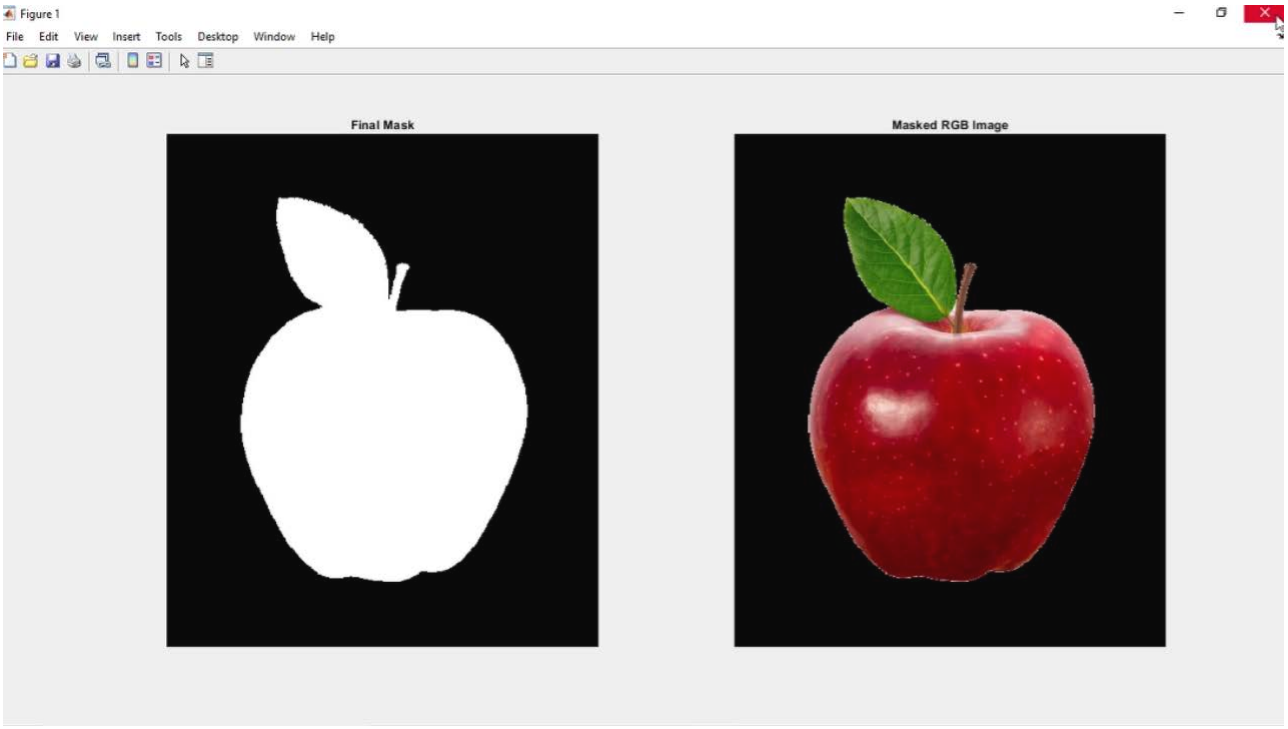


Figure 14: Output Images

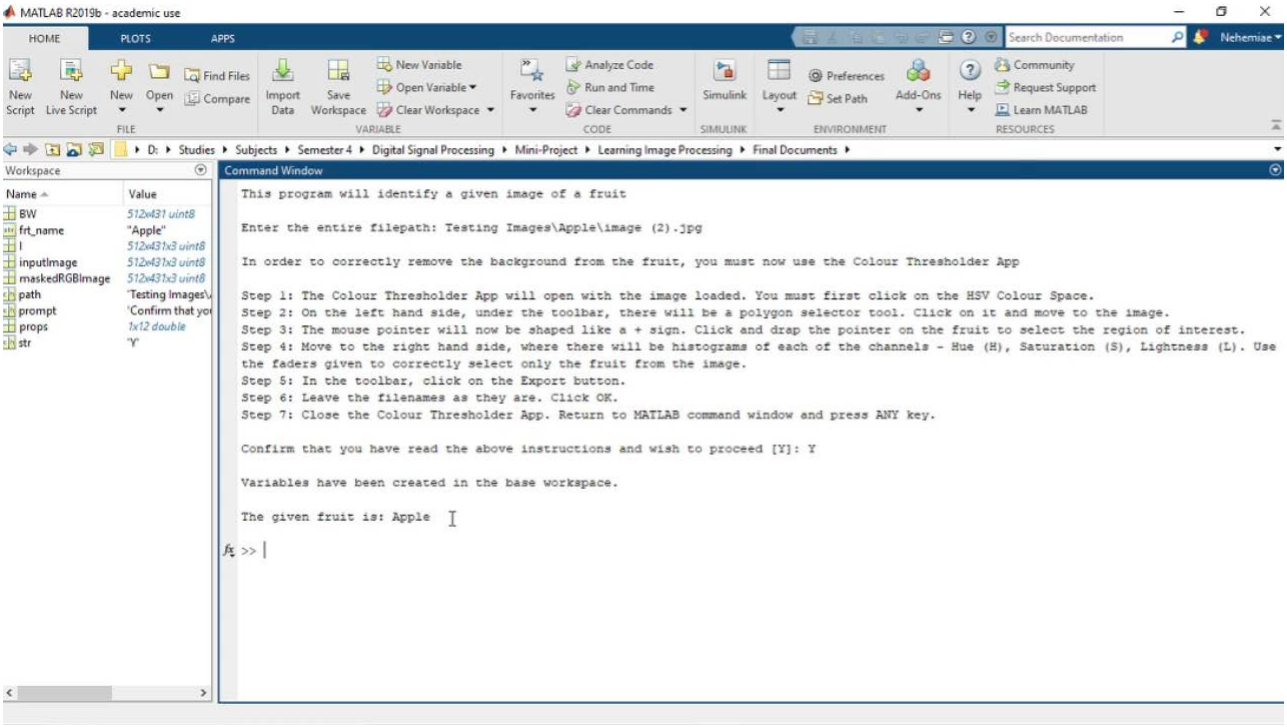


Figure 15: Output Result

The system was trained with 33 images and tested with 33 images. The following table summarizes the output results.

S.No.	Fruit	Training Images	Test Images	Hits	Misses
1	Apple	10	7	5	2
2	Banana	10	8	7	1
3	Guava	5	10	9	1
4	Strawberry	8	8	6	2

10 Conclusion

The statistical viabliltiy of the project can be improved by taking a larger dataset. One of the biggest hindrances encountered while collecting quality images, is the lack of uniformity for a particular fruit. This will greatly affect the validity of the system. This issue can resolved, by going to a local distribution center or APMC market and taking photos of the fruits over there. This will ensure in uniformity of a dataset. This was method implemented by the project[7], and they achieved an accurancy of greater than 90%. In addition, the inclusion of machine learning algorithms for removing backgrounds and finiding similarities between photos, can completely eliminate the need of human interaction with the system.

References

[1] Measures of skewness and kurtosis.

[2] Ruaa Adeeb Abdulmunem Al-Falluji. *Color, Shape and Texture based Fruit Recognition System*.

[3] Asdrubal Lopez-Chau Farid García-Lamont. *Segmentation of Images by Color Features: A Survey*.

[4] Mryka Hall-Beyer. *GLCM TEXTURE: A TUTORIAL*.

[5] Ms. Dhanusha Ms. Anisha M Nayak, Mr. Manjesh R. *Fruit Recognition using Image Processing*.

[6] Mrs. Shikha Pandey Pragati Ninawe. *A Completion on Fruit Recognition System Using K-Nearest Neighbors Algorithm*.

[7] R. Newlin Shebiah S. Arivazhagan. *Fruit Recognition using Color and Texture Features*.

11 Appendix

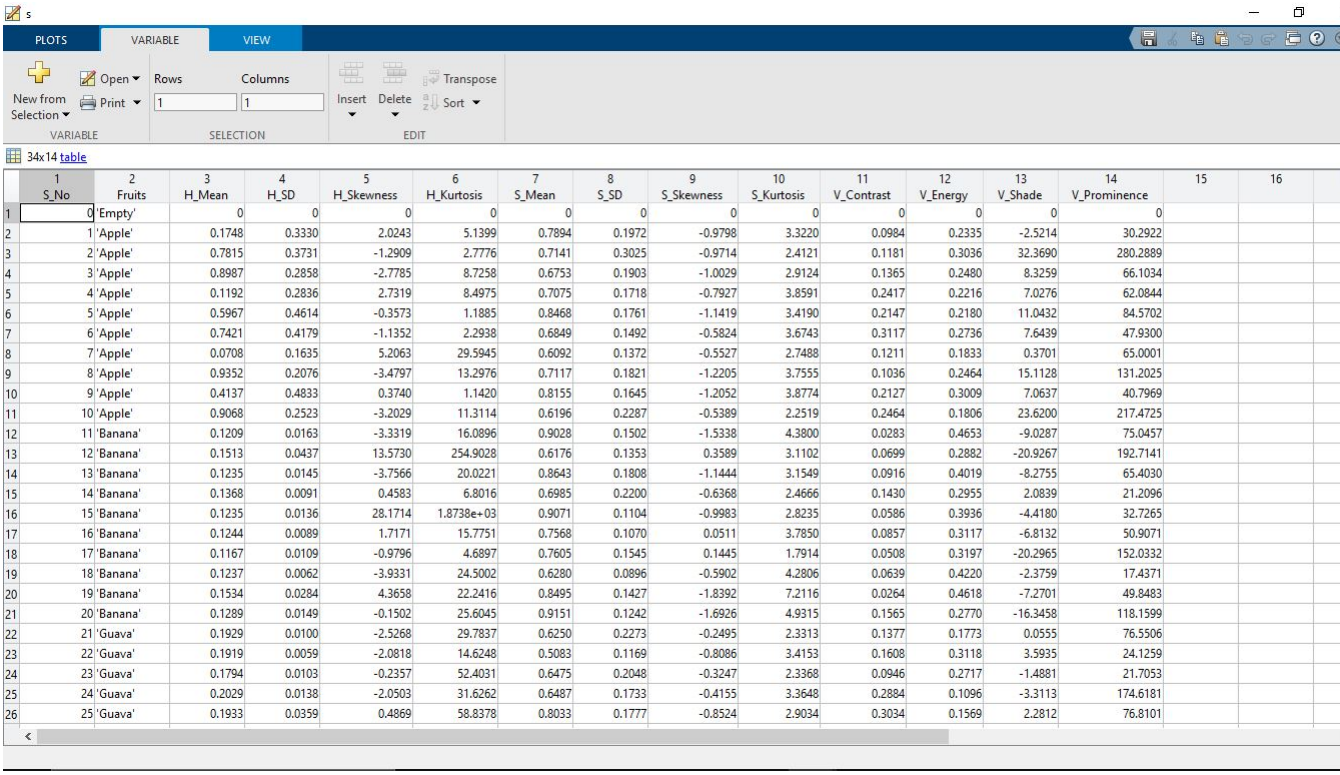
```
%This program is used to prepare a datasheet with mean
    values of several
%properties of 4 fruits - Apple, Banana, Guava and
   Stawberry. This data can
%be used to identify a given fruit.

clear;
clc;

myDir = 'Training Images\Apple'; %gets directory
myFiles = dir(fullfile(myDir,'*.jpg')); %gets all jpg
   files in struct
for img_index = 1:length(myFiles)
    baseFileName = myFiles(img_index).name;
    fullFileName = fullfile(myDir, baseFileName);
```

```
fprintf('Now reading %s\n', fullFileName);

I = imread(fullFileName);

fprintf('##Background Removal... \nPress any key \
   n');
colorThresholder(I);
pause; %Wait until the user presses any key
BW = imfill(BW, 'holes'); %Fills up any holes in
   the mask
BW = bwareafilt(BW, 1); %Get the largest object

fprintf('##Property Calculation... \n');
props = propcalc(maskedRGBImage);

prompt = '\nEnter the name of the fruit: ';
frt_name = input(prompt,'s');

fprintf('\n##Updating File... \n');
updatefile(frt_name,props);

fprintf('##Process Over... \n');
clear
```

```
    myDir = 'Trial'; %gets directory
    myFiles = dir(fullfile(myDir,'*.jpg')); %gets all
       jpg files in struct
end

%This function updates the table containing the
   properties of various
%fruits.

function updatefile(frt_name, props)
t = readtable('PropTable.txt');

index = max(t.S_No) + 1;

x = table();
x.S_No = index;
x.Fruits = frt_name;
x.H_Mean = props(1);
x.H_SD = props(2);
x.H_Skewness = props(3);
x.H_Kurtosis = props(4);
x.S_Mean = props(5);
x.S_SD = props(6);
```

```
x.S_Skewness = props(7);
x.S_Kurtosis = props(8);
x.V_Contrast = props(9);
x.V_Energy = props(10);
x.V_Shade = props(11);
x.V_Prominence = props(12);

t = [t; x];
writetable(t, 'PropTable.txt');

%This script is used to initialize the table used in
   the recognition of
%fruits, that holds the various properties of color.

S_No = 0;
Fruits = "Empty";
H_Mean = 0;
H_SD = 0;
H_Skewness = 0;
H_Kurtosis = 0;
S_Mean = 0;
S_SD = 0;
S_Skewness = 0;
```

```
S_Kurtosis = 0;
V_Contrast = 0;
V_Energy = 0;
V_Shade = 0;
V_Prominence = 0;


t = table(S_No, Fruits, H_Mean, H_SD, H_Skewness,
   H_Kurtosis, S_Mean, S_SD, S_Skewness, S_Kurtosis,
   V_Contrast, V_Energy, V_Shade, V_Prominence);
writetable(t, 'PropTable.txt');
```



	1 S_No	2 Fruits	3 H_Mean	4 H_SD	5 H_Skewness	6 H_Kurtosis	7 S_Mean	8 S_SD	9 S_Skewness	10 S_Kurtosis	11 V_Contrast	12 V_Energy	13 V_Shade	14 V_Prominence	15	16
1	0	'Empty'	0	0	0	0	0	0	0	0	0	0	0	0		
2	1	'Apple'	0.1748	0.3330	2.0243	5.1399	0.7894	0.1972	-0.9798	3.3220	0.0984	0.2335	-2.5214	30.2922		
3	2	'Apple'	0.7815	0.3731	-1.2909	2.7776	0.7141	0.3025	-0.9714	2.4121	0.1181	0.3036	32.3690	280.2889		
4	3	'Apple'	0.8987	0.2858	-2.7785	8.7258	0.6753	0.1903	-1.0029	2.9124	0.1365	0.2480	8.3259	66.1034		
5	4	'Apple'	0.1192	0.2836	2.7319	8.4975	0.7075	0.1718	-0.7927	3.8591	0.2417	0.2216	7.0276	62.0844		
6	5	'Apple'	0.5967	0.4614	-0.3573	1.1885	0.8468	0.1761	-1.1419	3.4190	0.2147	0.2180	11.0432	84.5702		
7	6	'Apple'	0.7421	0.4179	-1.1352	2.2938	0.6849	0.1492	-0.5824	3.6743	0.3117	0.2736	7.6439	47.9300		
8	7	'Apple'	0.0708	0.1635	5.2063	29.5945	0.6092	0.1372	-0.5527	2.7488	0.1211	0.1833	0.3701	65.0001		
9	8	'Apple'	0.9352	0.2076	-3.4797	13.2976	0.7117	0.1821	-1.2205	3.7555	0.1036	0.2464	15.1128	131.2025		
10	9	'Apple'	0.4137	0.4833	0.3740	1.1420	0.8155	0.1645	-1.2052	3.8774	0.2127	0.3009	7.0637	40.7969		
11	10	'Apple'	0.9068	0.2523	-3.2029	11.3114	0.6196	0.2287	-0.5389	2.2519	0.2464	0.1806	23.6200	217.4725		
12	11	'Banana'	0.1209	0.0163	-3.3319	16.0896	0.9028	0.1502	-1.5338	4.3800	0.0283	0.4653	-9.0287	75.0457		
13	12	'Banana'	0.1513	0.0437	13.5730	254.9028	0.6176	0.1353	0.3589	3.1102	0.0699	0.2882	-20.9267	192.7141		
14	13	'Banana'	0.1235	0.0145	-3.7566	20.0221	0.8643	0.1808	-1.1444	3.1549	0.0916	0.4019	-8.2755	65.4030		
15	14	'Banana'	0.1368	0.0091	0.4583	6.8016	0.6985	0.2200	-0.6368	2.4666	0.1430	0.2955	2.0839	21.2096		
16	15	'Banana'	0.1235	0.0136	28.1714	1.8738e+03	0.9071	0.1104	-0.9983	2.8235	0.0586	0.3936	-4.4180	32.7265		
17	16	'Banana'	0.1244	0.0089	1.7171	15.7751	0.7568	0.1070	0.0511	3.7850	0.0857	0.3117	-6.8132	50.9071		
18	17	'Banana'	0.1167	0.0109	-0.9796	4.6897	0.7605	0.1545	0.1445	1.7914	0.0508	0.3197	-20.2965	152.0332		
19	18	'Banana'	0.1237	0.0062	-3.9331	24.5002	0.6280	0.0896	-0.5902	4.2806	0.0639	0.4220	-2.3759	17.4371		
20	19	'Banana'	0.1534	0.0284	4.3658	22.2416	0.8495	0.1427	-1.8392	7.2116	0.0264	0.4618	-7.2701	49.8483		
21	20	'Banana'	0.1289	0.0149	-0.1502	25.6045	0.9151	0.1242	-1.6926	4.9315	0.1565	0.2770	-16.3458	118.1599		
22	21	'Guava'	0.1929	0.0100	-2.5268	29.7837	0.6250	0.2273	-0.2495	2.3313	0.1377	0.1773	0.0555	76.5506		
23	22	'Guava'	0.1919	0.0059	-2.0818	14.6248	0.5083	0.1169	-0.8086	3.4153	0.1608	0.3118	3.5935	24.1259		
24	23	'Guava'	0.1794	0.0103	-0.2357	52.4031	0.6475	0.2048	-0.3247	2.3368	0.0946	0.2717	-1.4881	21.7053		
25	24	'Guava'	0.2029	0.0138	-2.0503	31.6262	0.6487	0.1733	-0.4155	3.3648	0.2884	0.1096	-3.3113	174.6181		
26	25	'Guava'	0.1933	0.0359	0.4869	58.8378	0.8033	0.1777	-0.8524	2.9034	0.3034	0.1569	2.2812	76.8101		

Figure 16: Table of Stored Properties