Presentation in "Robotics" Automatic Control Department

Author: Fitim Halimi
Professor : Henryk Palus

Brief information before presentation

- Presentation itself will be explaining how the tasks have been solved and give general information about image processing and methods used.

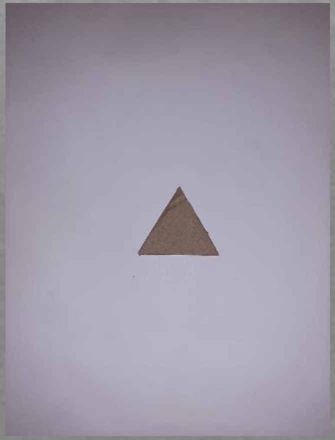
- **Task A, subtask 1:**
  Obtain the images of 10 flat objects of different shapes from different distances and orientations. For each object, acquire minimum 5 images. Binarize the acquired images

- So first thing first, it was necessary creating a database with the images requested, each image had to be different from the same shape that it was taken, it had to be tilted, taken closer or further etc.

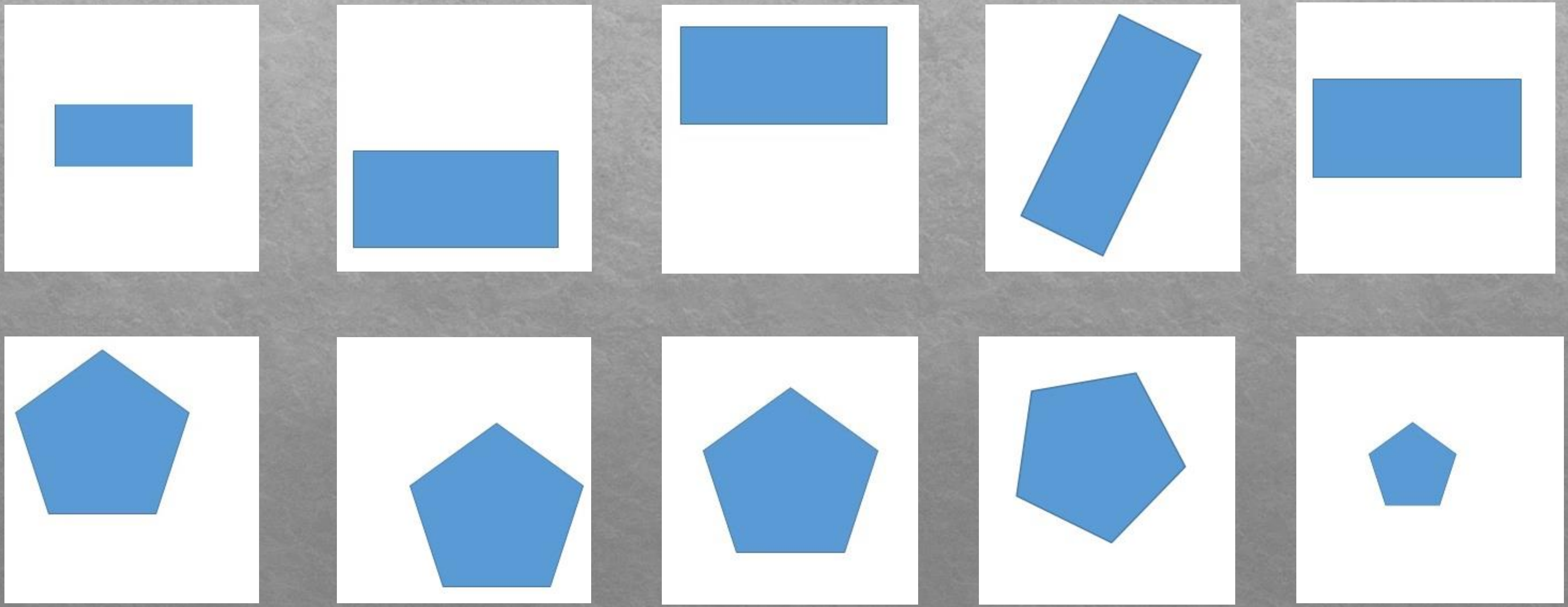
Obtaining images

- Each image from the same shape has to be different from each other so that the program with be efficient, and also able to detect the shape in any orientation, not only in the normal angle.

- The images below are from a triangle, took five pictures in different orientation, it is necessary to have a clear background as later on you will have to filter the noise in the image.



Other images obtained

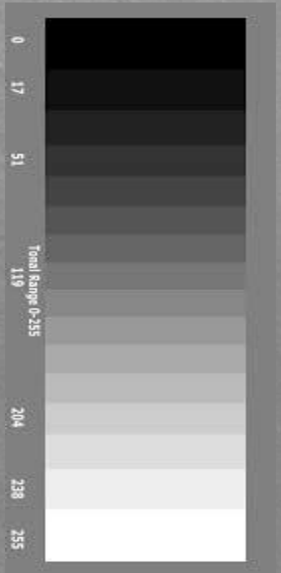


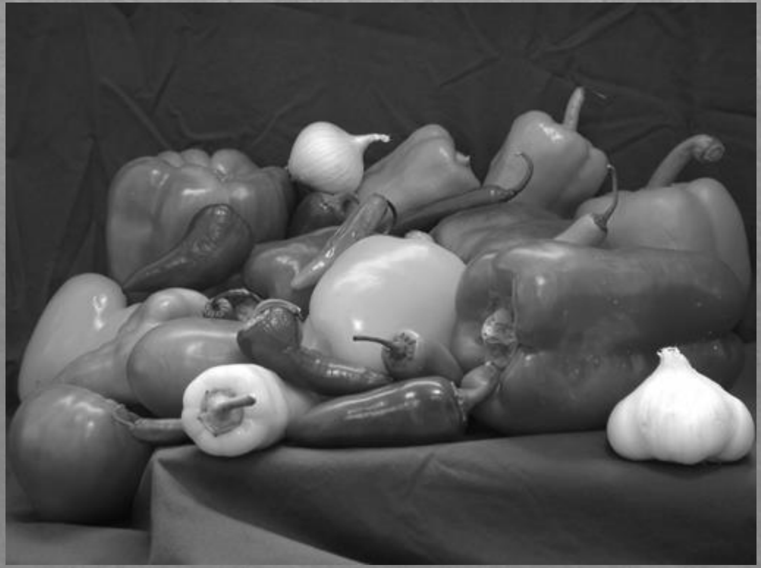
Binarizing images (grayscale)

- A grayscale or greyscale image is one in which the value of each pixel is a single sample representing only an amount of light, that is, it carries only intensity information
- To be able to binarize the image, first the RGB image has to be converted to grayscale through the command in Matlab:

```
RGB = imread('peppers.png')
imshow (RGB)
```

```
I = rgb2gray(RGB);
figure
imshow(I)
```



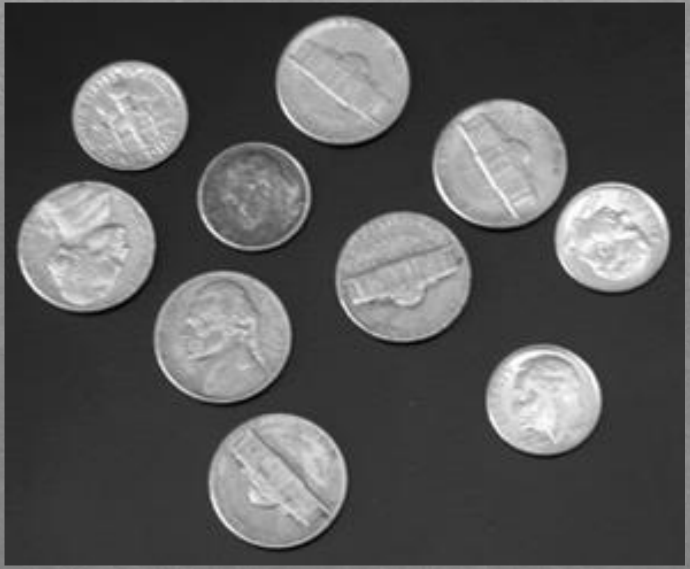


Binarizing images

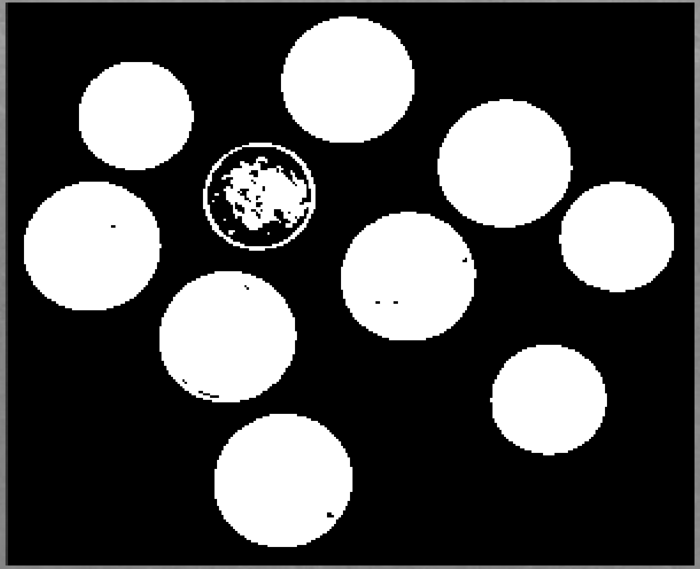
First lets explain what is image binarization.

Image binarization is the process of taking a grayscale **image** and converting it to black-and-white, essentially reducing the information contained within the **image** from 256 shades of gray to 2: black and white, a binary **image**

```
RGB = imread ('coins.png')
I = rgb2gray (RGB)
imshow (I)
```



```
I = imread ('coins.png')
BW = imbinarize (I)
Imshow (I,'montage')
```

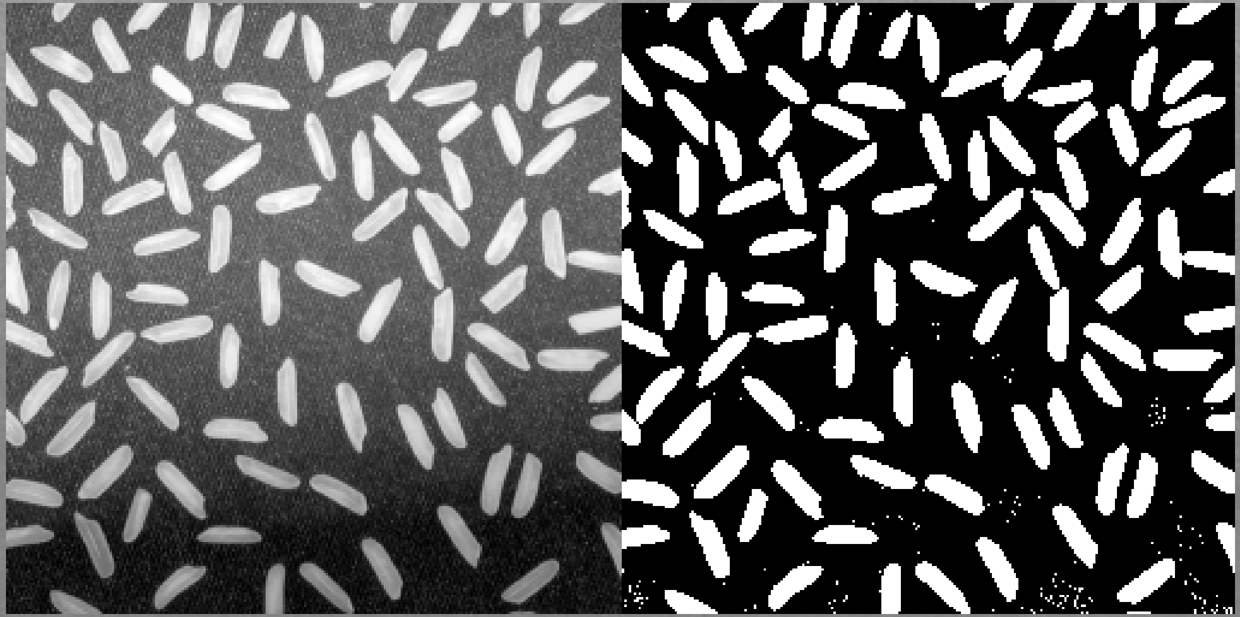


Binarizing images

The usage of these algorithms these days for image processing has been increasing and making the industrialization process much easier and faster.

As by there machines that can process the images, the process of classification has been improved in series production of different shaped objects (for ex, bottles)

Ex. It can classify empty crates, and depending on which color the crates are they can be classified, also depending on the size or shape.



Shape factors

Task A, subtask 2: Implement few shape factors that will be the following as features useful in the process of classification of objects contained in the images acquired in pt.1.

To understand better I will let you watch the following video about Object Detection

```
FACTORS%
[row, col] = find(L==n);
Edgesobel=edge((L==n), 'sobel');   %edge detect each shape again
BinaryX1=imfill((L==n),'holes');   %fill in the holes in each binary component

%To find Bounding Box
sx=min(col)-0.5;                                       %starting position for column
sy=min(row)-0.5;                                       %starting position for row
breadth=max(col)-min(col)+1;                           %gives width of the box
length=max(row)-min(row)+1;                            %gives length of the box
BBox=[sx sy breadth length];                           %Bounding Box value for the range of n
```

Following, a video to explain why image processing is useful for.

Shape factors

```
%Find Perimeter
BW1=bwboundaries(BinaryX1);                    %Find the boundary of the labeled components
c=cell2mat(BW1(1));                            %converts the cell to a matrix
Perimeter=0;                                   %declaring a start at zero for sum
for i=1:size(c,1)-1                            %creates a loop that extracts perimeter

Perimeter=Perimeter+sqrt((c(i,1)-c(i+1,1)).^2+(c(i,2)-c(i+1,2)).^2);
end
display(Perimeter);                                           %display perimeter

%Find Roundness
Roundness=(4*area1*pi)\Perimeter.^2;                           %defines how close the binary
component is to a circle
display(Roundness);                                            %display roundness factor
```

Shape factors

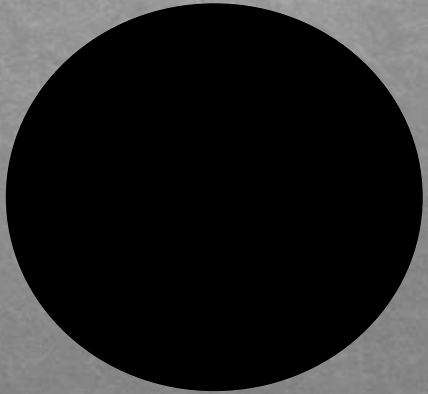
```
%To find Ratio
area1=bwarea(BinaryX1);          %calculates the area of each binary component
x1=round(area1);                 %rounding off calculated area
d=x1 * 0.001;                    %converting from mm to m
a=roundn(d,-3);                  %rounding off to 3 decimal places
area2=(length*breadth);          %calculates the area of each bounding box
ratio = area2/area1;             %calculates the ratio of area's
Ratio=roundn(ratio,-2);          %rounds the ratio to two decimal places

%Edge Detection
Mx=corner(Edgesobel);            %corner detection applied to each component
 x1 = Mx(2,1);                   %each corner has an x and y co-ordinate.
 x2 = Mx(3,1); x3 = Mx(1,1);
 y1 = Mx(2,2); y2 = Mx(3,2);y3 = Mx(1,2);

%Find Centroid
X=mean(col);                     %calculates average of columns
Y=mean(row);                     %calculates average of rows
Centroid=[X Y];                  %matrix of centroid dimensions
display(Centroid);               %display centroid
```

Circle classification

```
%SHAPE CLASSIFICATION CIRCLE
    if(Ratio>=1.30 && Ratio<=1.32 && Roundness>=1 && Roundness<=1.13);
%compares factors
    disp('circle');
%display result
    figure(10+n),imshow(L==n);title('Circle Image');
%shows result in figure
    hold on
%holds figure
    text(Centroid(1),Centroid(2), 'circle');
%places text
    hold off
%goes to next command
    img_wk_bw_L=(L==n);
    end
```

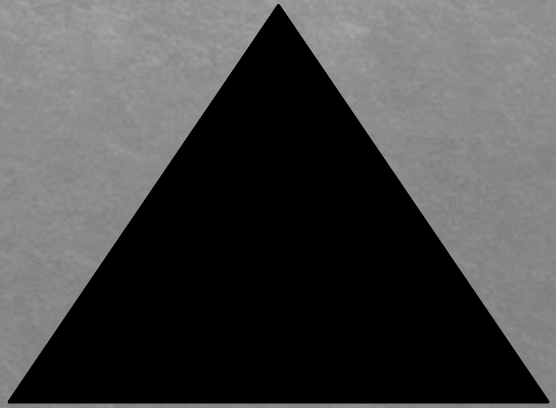


Square classification

```
%SHAPE CLASSIFICATION SQUARE
    if (Result==0 && Ratio==1 )
%compares factors
    disp('square');
%display result
    figure(10+n), imshow(L==n);title('Square Image');
%shows result in figure
    hold on
%holds figure
    text(Centroid(1),Centroid(2), num2str(2));
%places text
    hold off
%goes to next command
    img_wk_bw_L=(L==n);
    end
```

Triangle classification

```
%SHAPE CLASSIFICATION TRIANGLE
    if (Ratio>=1.6 && Ratio<=1.90 &&  Roundness>=1.500 && Roundness<=1.800);
%compares factors
    disp('triangle');
%display result
    figure(10+n),imshow(L==n);title('Triangle Image');
%shows result in figure
    hold on
%holds figure
    text(Centroid(1),Centroid(2), num2str(3));
%places text
    hold off
%goes to next command
    img_wk_bw_L=(L==n);
    end
```



Building the classifier

Task A, subtask 3: Build a classifier that classifies as accurately as possible the objects contained in 50 binary images into 10 classes.

So after resolving this task in the code, we needed all the images classified in 10 different folders, so each folder after the software runs has to contain 5 images of the same shape in one folder. In total 50 images will be processed.

Starting a part of the code below:

```
int main(void)
{
  char * filepath = new char[100];
  char * out_filepath = new char[100];

  for(int img = 0; img < NUM_IMAGES ; img++) // for all images
  {
    sprintf(filepath,
"/home/fitimhalimi/ws/prototype_ws/ShapeDetector/test/img_%i.png", img); //
changing the path
    DetectShapes shapes(cv::imread(filepath));
```

Classifying the shapes

```
 switch(shapes.getShape())
    {
      case DetectShapes::Shapes::Triangle:
      {
        sprintf(out_filepath,
"/home/fitimhalimi/ws/prototype_ws/ShapeDetector/result/Triangle/img_%i.png",
img); // changing the path
        cv::imwrite(out_filepath, shapes.getShapesImage());
        break;
      }
      case DetectShapes::Shapes::Square:
      {
        sprintf(out_filepath,
"/home/fitimhalimi/ws/prototype_ws/ShapeDetector/result/Square/img_%i.png",
img); // changing the path
        cv::imwrite(out_filepath, shapes.getShapesImage());
        break;
      }
```

Fitim Halimi
Applied digital sign...
New Microsof...
LibreOffice 6.3
article
SG90-Micr...
Reports
Robot Vision
System Dynamic...
computer programm...
SaA
Thesis
Description
Visa
Courses to be recog...
Courses to follow
ADSP_Final
Project
This PC
Matlab files DSRr
New folder (2)
71RF2BSP...
$_10
AC
ApowerREC
Recycle Bin
Desktop
Arduino
Arduino_U...
afterburner...
336-336137...
New Microsof...
Google Chrome
InstallSetapp
Aplikimi per burse
arduinona...
IMG_2222
IMG_2225
Fitim
EBank-1
11
arduino-un...
IMG_2221
IMG_2226
Application 2
Untitled4
Mathpix Snippi...
Untitled
IMG_2223
Oracle VM VirtualBox
Application 1
Aplikimi per burse
La.La.Land....
485
IMG_2224
ShapeDete...
00:00:00
ENG
5:29 PM
1/7/2020

Questions, Suggestions?

Note : there are only parts of code explained in the presentation.