

Automatic Recognition of Fish Diseases in Fish Farms

By:

Ahmed Waleed

Hadeer Medhat

Mariam Esmail

kareem Osama

Supervised By: DR.Taraggy M Ghani
Eng.Radwa Samy

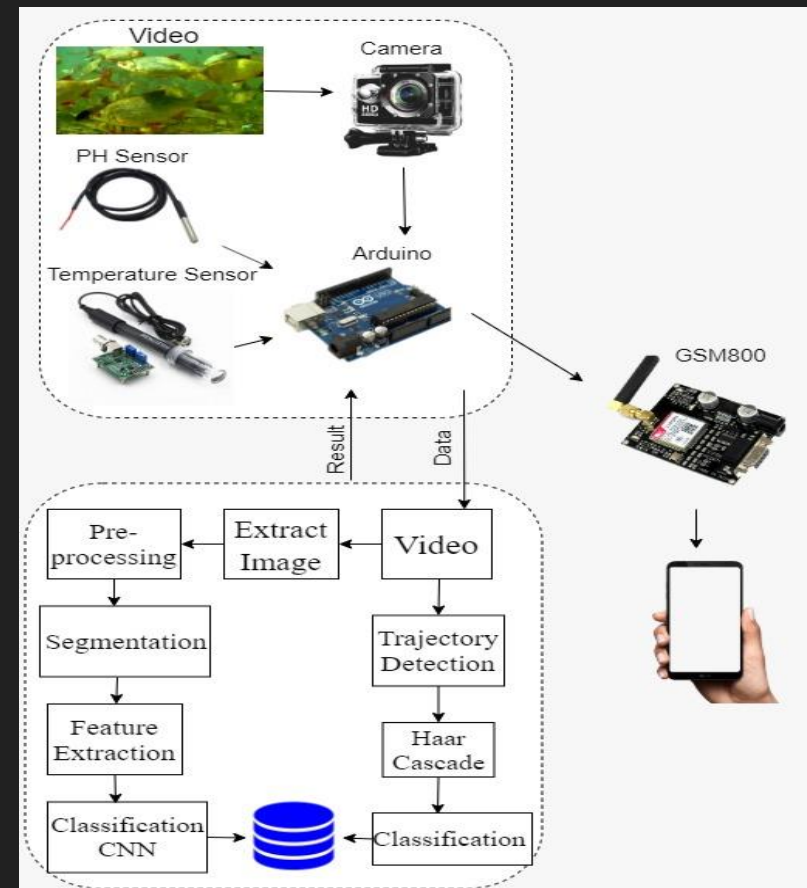
Introduction

- The purpose of this Software design document is to describe the architecture and system design for our Automatic recognition of Fish Diseases in Fish Farms system. Our system mainly detects and diagnose fish diseases in fish farms au-tomatically and examine water quality

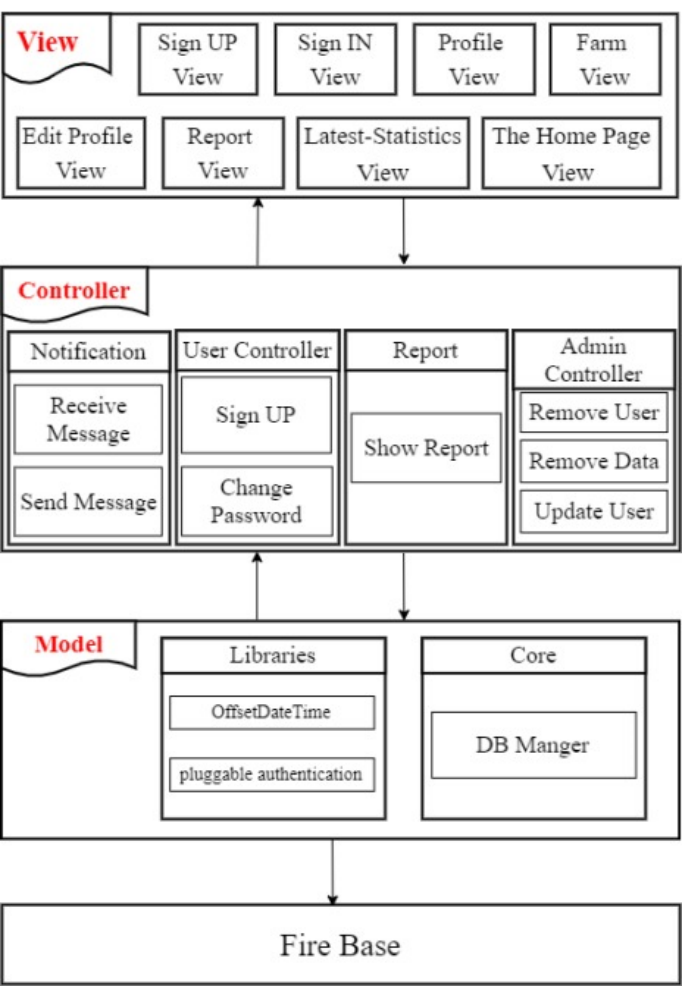
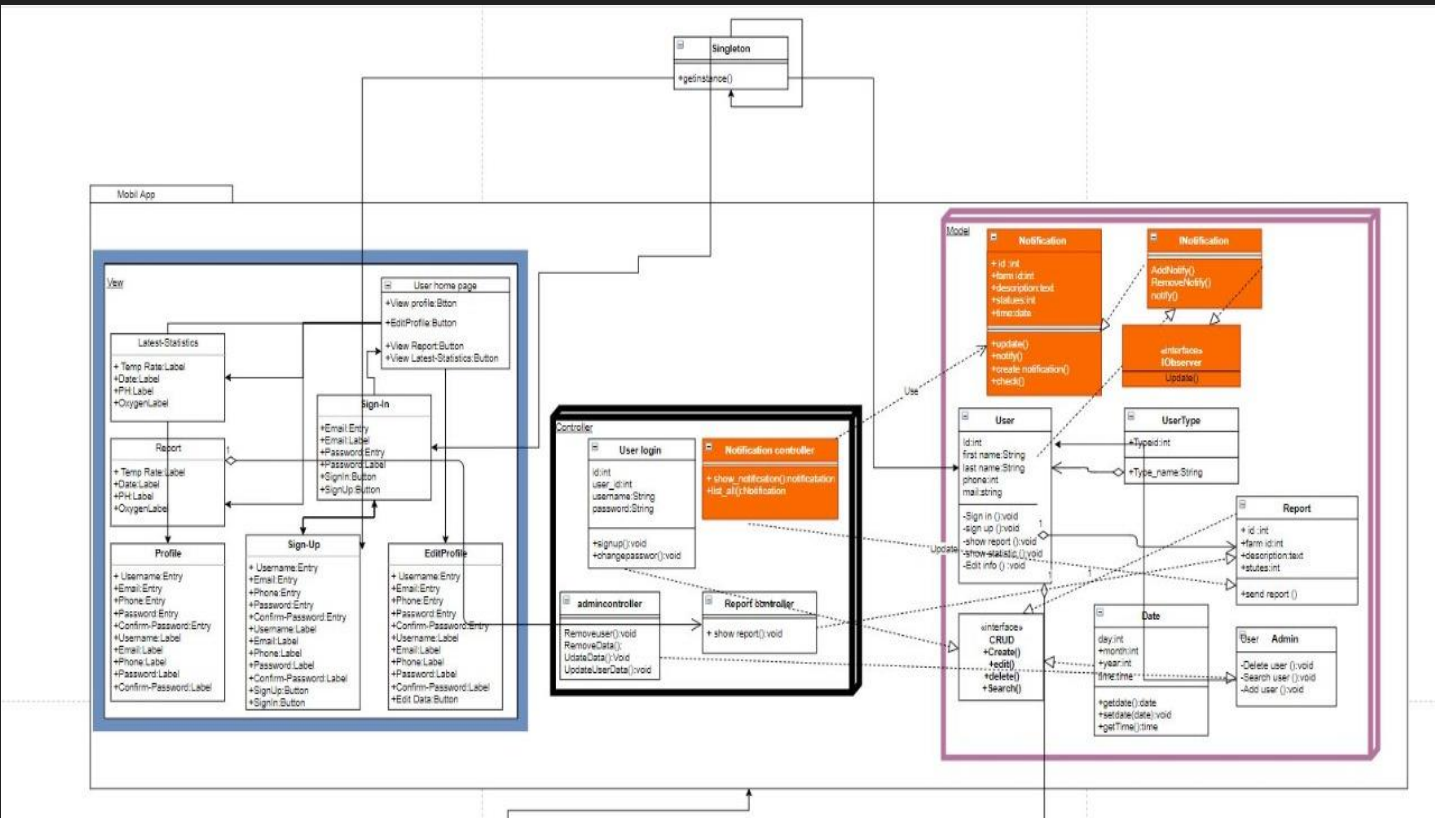


Overview

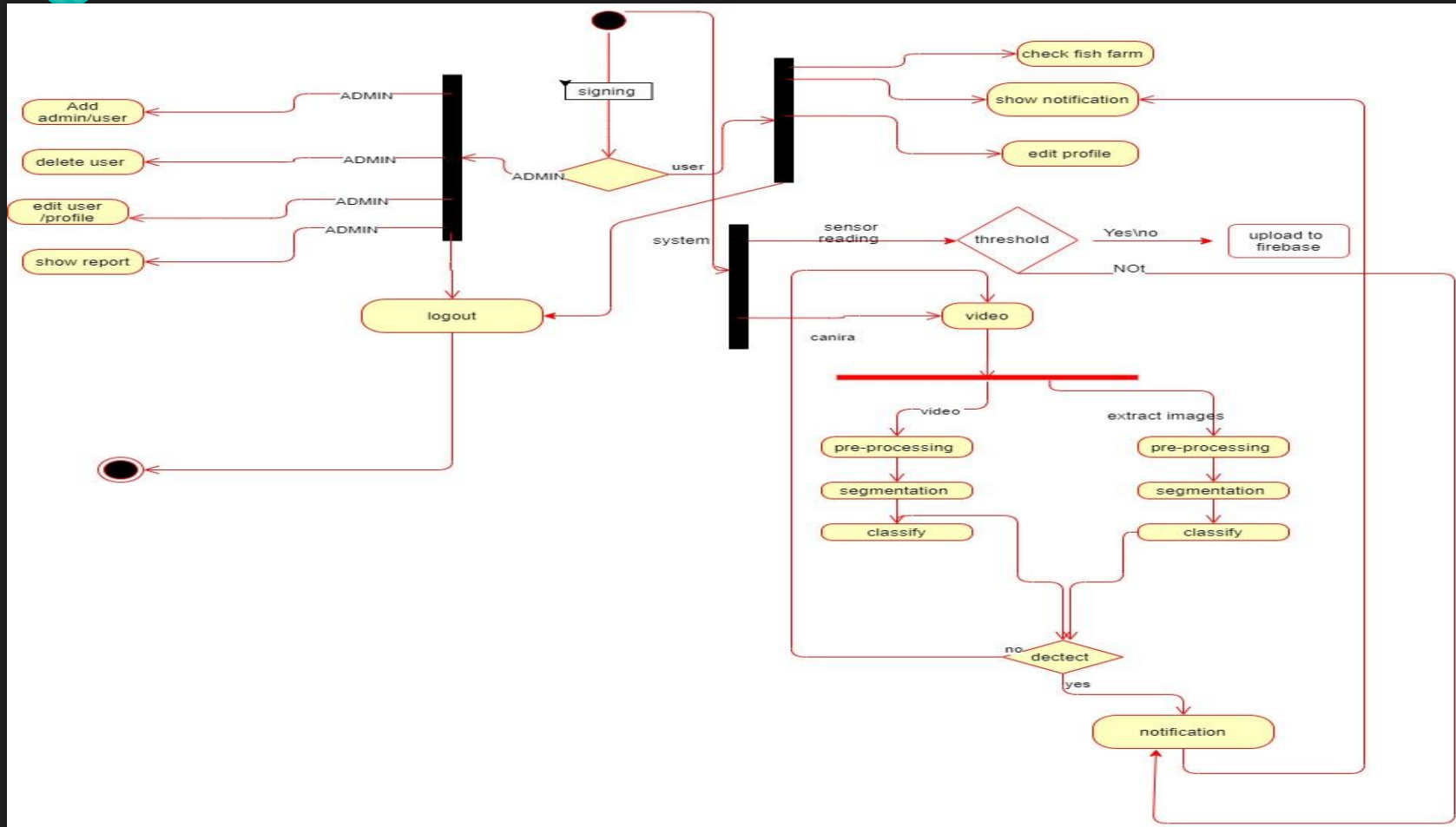
- Our system is created to performs detection and identification of fish diseases and analyze fish abnormal behavior. Its goal is to prevent and control spreading of diseases by early identification and diagnosis



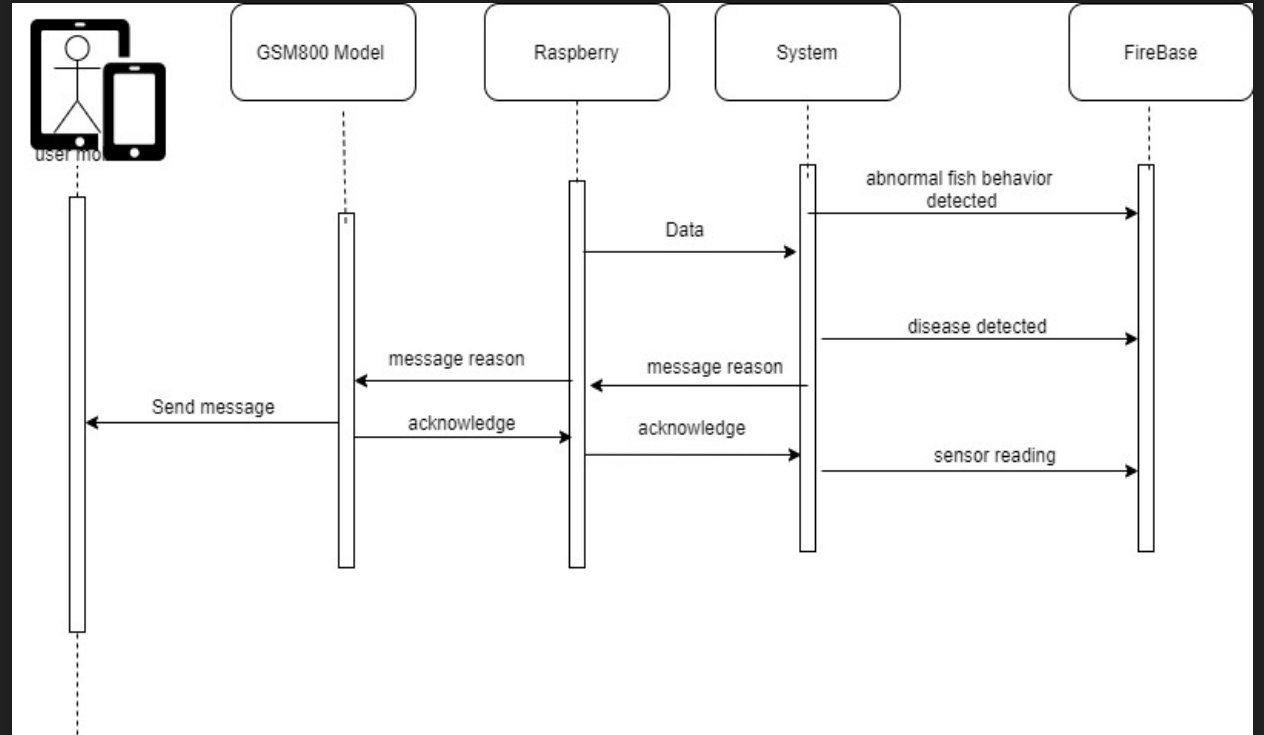
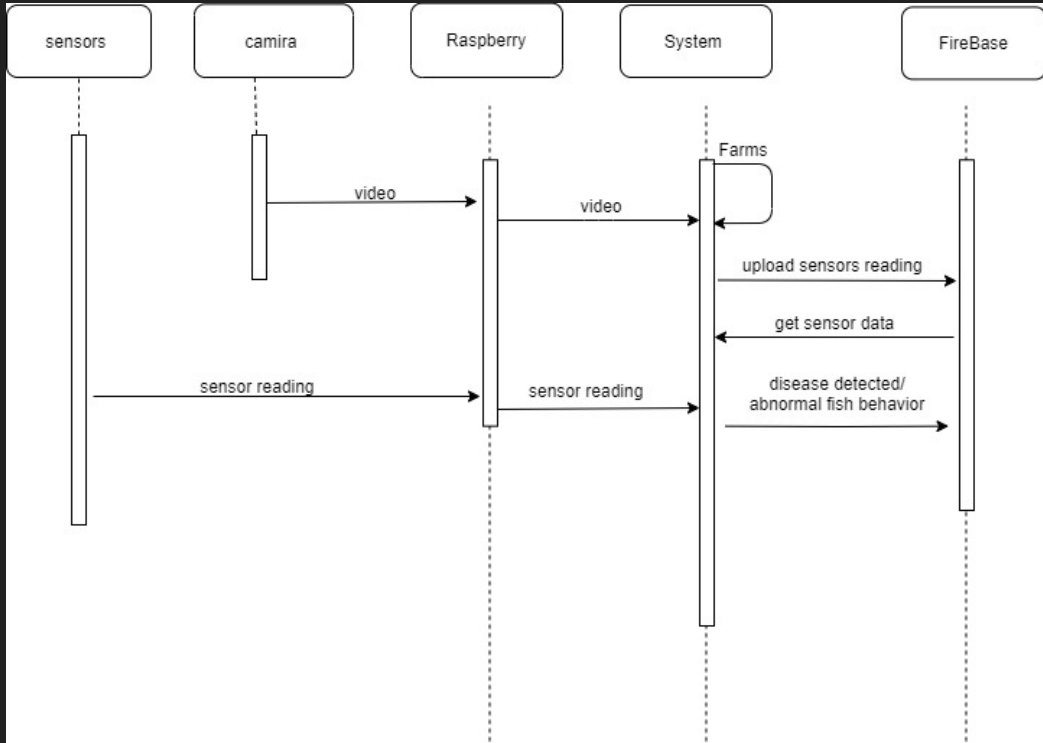
Architectural Design



Activity Diagram



Sequence Diagram



Component Design

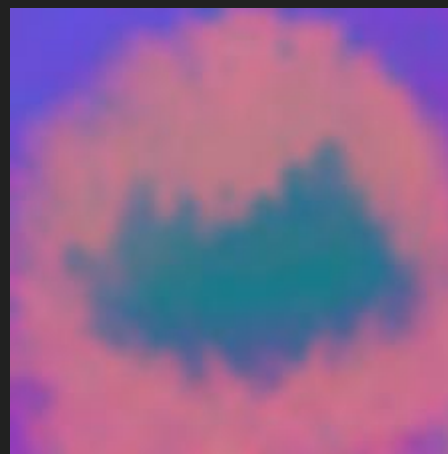
1) Pre-processing



RGB



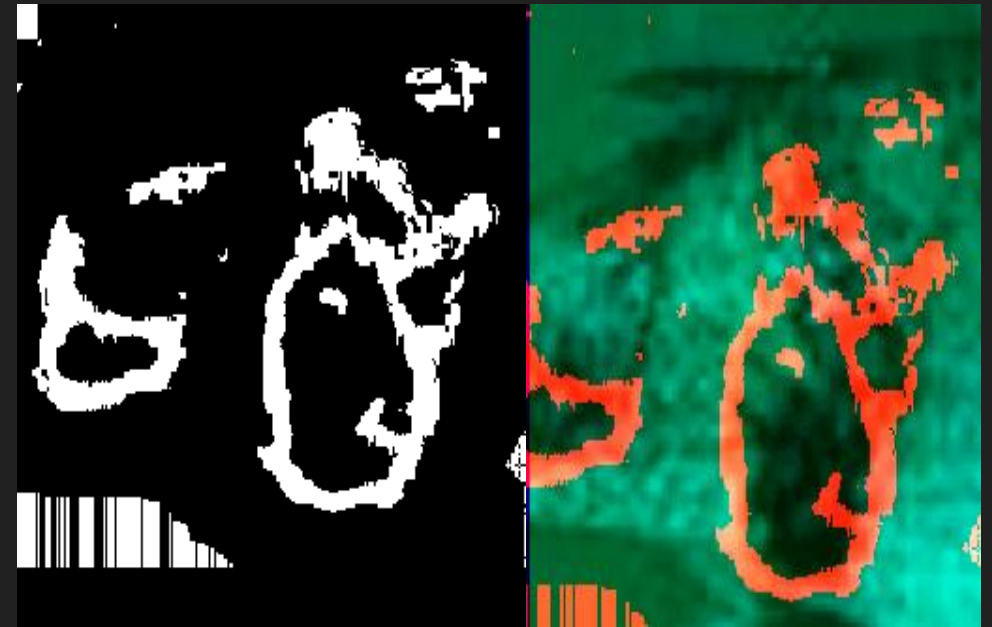
YCbCr



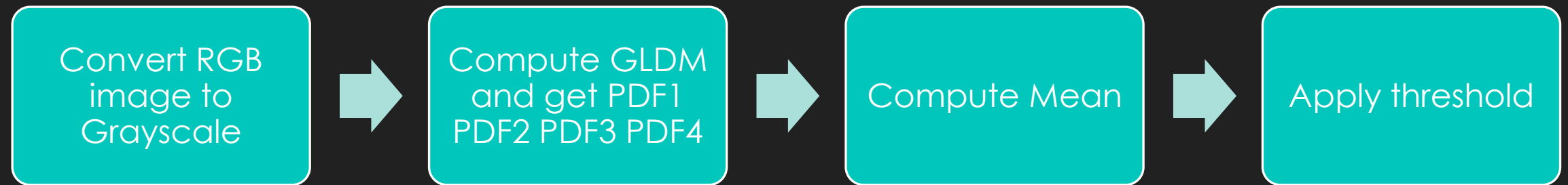
XYZ

Color-Based Gaussian distribution model

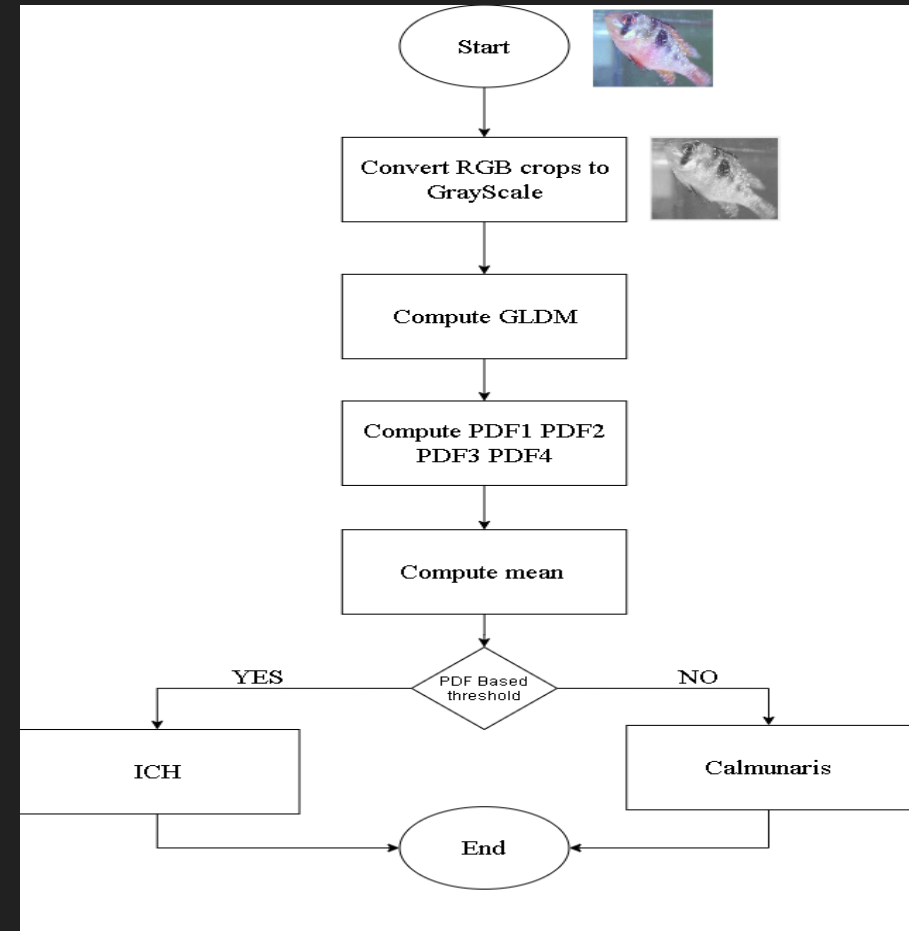
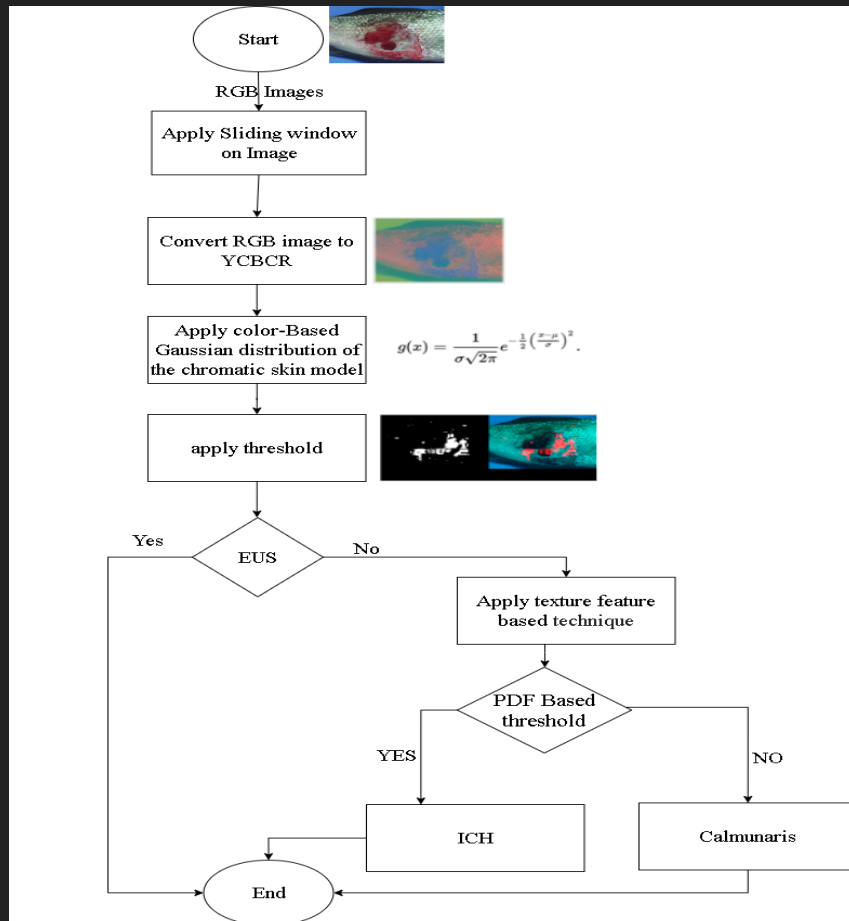
- Convert Image to YCBCR
- Get CB and CR
- Get mean and covariance
- Build gaussian distribution model
- Apply threshold



Texture Feature Based



Multi staging hybrid segmentation technique



CNN Classification

ResNet
 VGG
 Squeeze net
 AlexNet
 Google Net
 MobileNetv2
 Xception
 ShuffleNet
 DenseNet
 Inceptionv3

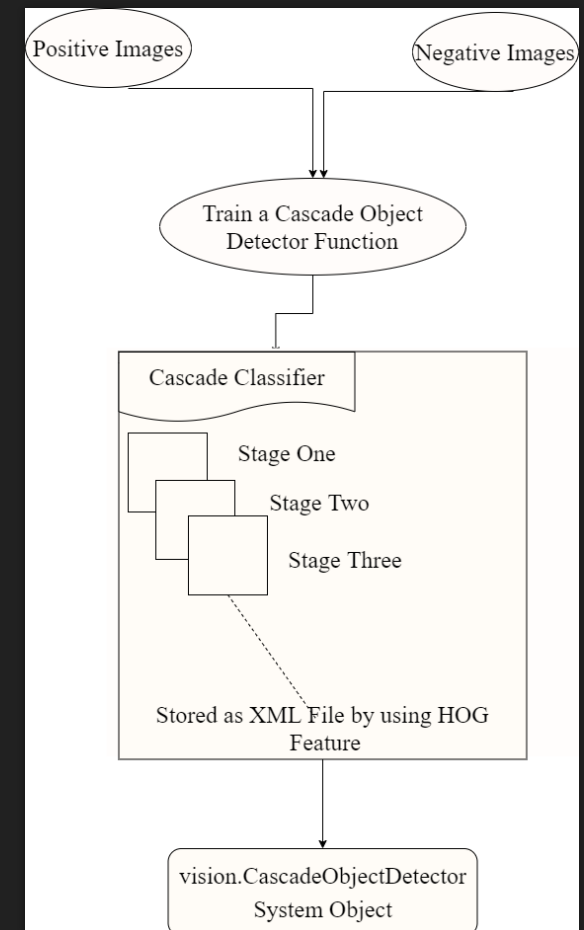
Model	Versions	Layers	Activation function	Input size	Convolution kernel size	Innovative point	Applications
ResNet ??	18 50 101	18 layers 50 Layers 101 Layers	ReLU	224x224x3	ResNet initial convolution: 7x7. Resnet 50 and 101: 1x1, 3x3 and 1x1. ResNet 18: 3x3	ResNet overcomes degradation and vanishing gradient problems residual blocks that increases the number of hidden layers. The core idea of ResNet is introducing "identity shortcut connection" that skips one or more layers	ImageNet ?? 2012 dataset and Automatic Spoofing Detection [1]
VGG [2]	16 19	16 layers 19 layers	ReLU non-linear	224x224x3	VGG: 3x3	This network uses only 3x3 convolutional layers stacked on top of each other in increasing depth. VGG makes improvement over AlexNet by replacing large kernel-sized	ImageNet (ILSVRC-2012) and Classifying Cooking Object's State [3]
SqueezeNet [4]		18 layers	ReLU	227x227x3	SqueezeNet: 1x1 and 3x3	SqueezeNet goal was to create small neural network with fewer parameters. It was able to achieve a 50X reduction in model size compared to AlexNet. SqueezeNet has advantage of fire module, which uses less filters to decrease number of parameters	ImageNet dataset, fine-grained object recognition [5] and Generic Visual Recognition [6]
DenseNet [7]	201	201 layers	ReLU	224x224x3	DenseNet: 7x7	In DenseNet, the feature-maps of all preceding layers are used as inputs, and its own feature-maps are used as inputs into all subsequent layers. DenseNet alleviate the vanishing-gradient problem and reduce the number of parameters	ImageNet (ILSVRC-2012) [8] and optical flow [1]
AlexNet [9]		8 layers	ReLU non-linear	227x227x3	Alexnet: 11x11, 5x5 and 1x1	AlexNet has large number of filters to perform the convolution operation of sizes 11x11, 5x5 and 3x3	ImageNet(ILSVRC-2010) and high Spatial Resolution Remote Sensing [10]
GoogleNet [11]		22 layers	ReLU	224x224x3	GoogleNet: 5x5, 3x3 and 1x1	GoogleNet increases the depth of the network and gain a higher performance level. It is based on the concept of the inception module, it is the collection of convolution and pooling operation performed in a parallel manner so that features can be extracted using different scales	ImageNet(ILSVRC14) and High performance offline handwritten Chinese character recognition [12]
Mobilenetv2 [13]	2	54 layers	ReLU6	224x224x3	Mobilenetv2: 1x1, 3x3	Mobilenetv2 improves the performance of mobile models on multiple tasks. Mobilenetv2 is based on an inverted residual structure	ImageNet, Face Attribute Detection [14]
Xception [15]		71 layers	ReLU	299x299x3	Xception: 1x1,3x3	Xception involves Depthwise Separable Convolutions, it is supposed to be more efficient than classical convolution in terms of computation time. Xception relies on Shortcuts between Convolution blocks	ImageNet dataset [16], Audio Event Detection and Tagging [17]
Inceptionv3 [18]	3	48 layers	ReLU	299x299x3	Inceptionv3: 3x3	In inceptionv3, computational efficiency and fewer parameters are realized	ImageNet(ILSVRC 2012) and flower classification [19]
ShuffleNet [20]			ReLU	224x224x3	Shufflenet: 1x1,3x3	Shufflenet aim to explore a highly efficient architecture specially designed for limited computing ranges. Shufflenet allows more feature map channels and it is especially critical to the performance of very small networks. ShuffleNet achieves 13x actual speedup over AlexNet while maintaining comparable accuracy	ImageNet, Mobile devices [21] .

CNN Classification Result

Cnn Architectures	RGB(0.001)	RGB(0.01)	RGB(0.1)	YCBCR(0.001)	YCBCR(0.01)	YCBCR(0.1)	XYZ(0.001)	XYZ(0.01)	XYZ(0.1)
ResNet18	17,60%	14,50%	24,60%	13,60%	12,50%	12,50%	17,70%	18,50%	15,50%
ResNet50	18,70%	18,60%	15,60%	14,60%	15,50%	15,50%	24,50%	16,50%	16,50%
ResNet101	16,80%	14,%	16,50%	13,50%	14,50%	14,50%	15,50%	16,50%	19,50%
Alex-Net	12,60%	7,50%	9,60%	7,60%	9,50%	9,50%	8,50%	8,50%	8,60%
VGG16	14,50%	14,60%	14,50%	15,50%	18,40%	18,50%	16,60%	16,50%	15,50
VGG19	14,50%	14,60%	15,50%	21,50%	17,60%	14,60%	16,50%	15,50%	15,50%
Mobilenetv2	13,50%	7,60%	8,50%	17,60%	7,40%	7,40%	10,60%	9,60%	9,60%
Xception	12,50%	12,50%	11,50%	11,50%	16,60%	13,50%	13,50%	12,50%	12,60%
Inceptionresnetv2	11,50%	16,60%	12,60%	12,60%	11,50%	11,60%	15,50%	12,50%	12,60%
Shufflenet	7,50%	7,60%	7,50%	7,60%	7,50%	8,50%	21,50%	19,60%	15,60%
Nasnetmobile	9,50%	7,50%	7,60%	7,50%	7,50%	8,60%	15,60%	15,60%	15,60%
Nasnetlarge	13,60%	13,50%	15,50%	14,50%	14,60%	13,60%	14,50%	14,60%	14,60%
Squeezenet	7,50%	7,60%	7,50%	7,50%	7,50%	8,50%	8,60%	11,50%	8,50%
Inceptionv3	15,50%	14,40%	12,50%	15,60%	11,60%	11,50%	14,50%	16,50%	13,50%
Densenet201	6.53,60%	13,50%	7,60%	16,60%	7,50%	10,60%	12,60%	10,50%	7,60%
Googlenet	15,60%	14,50%	14,50%	14,50%	14,50%	16,50%	20,50%	16,50	19,50%

Train Cascade Object Detector

- The `vision.CascadeObjectDetector` System object detects objects in images by sliding a window over the image.
- The detector then uses a cascade classifier to decide whether the window contains the object of interest.

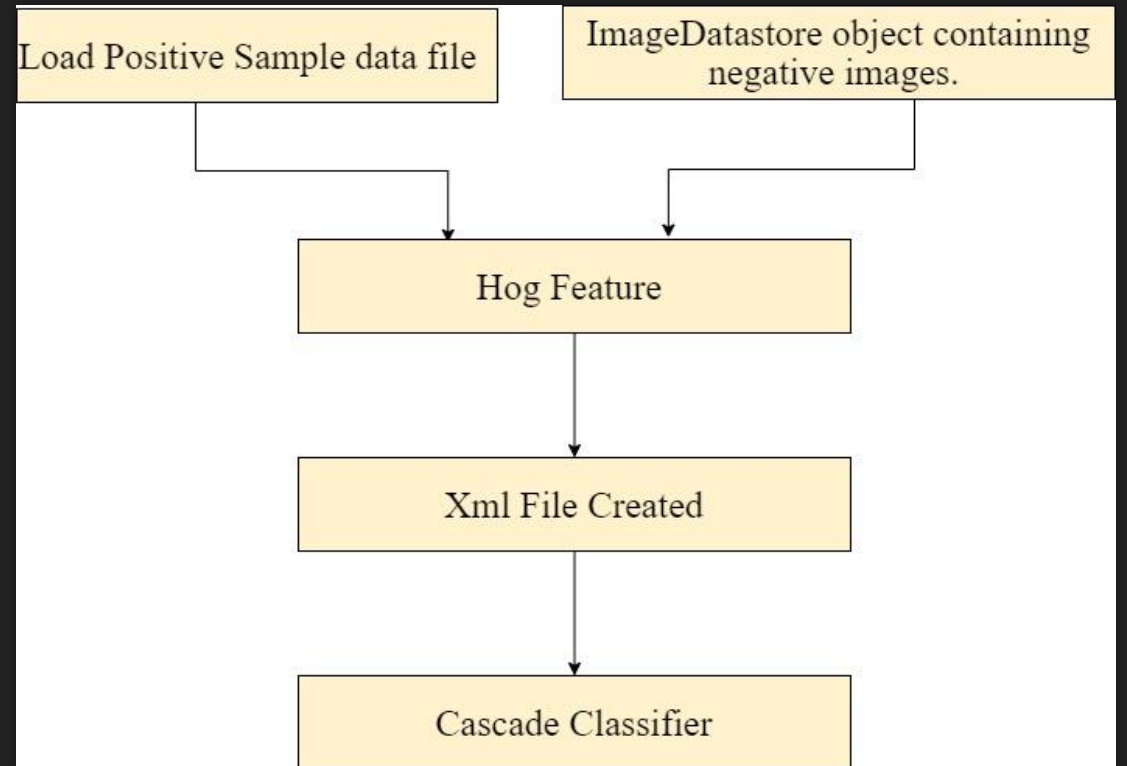


XML Creation

- Load the positive samples data from file.
- Create an imageDatastore object containing negative images.
- Choose the feature that suits the type of object detection.

(The HOG features are often used to detect objects) .

- Then run `trainCascadeObjectDetector` to create xml file



Compare

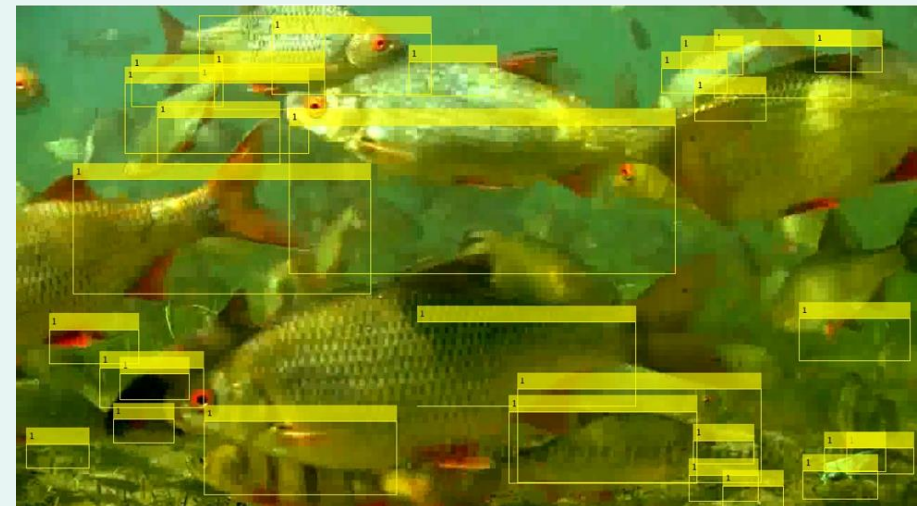
HOG Feature

HOG features are often used to detect objects such as people and cars. They are useful for capturing the overall shape of an object.



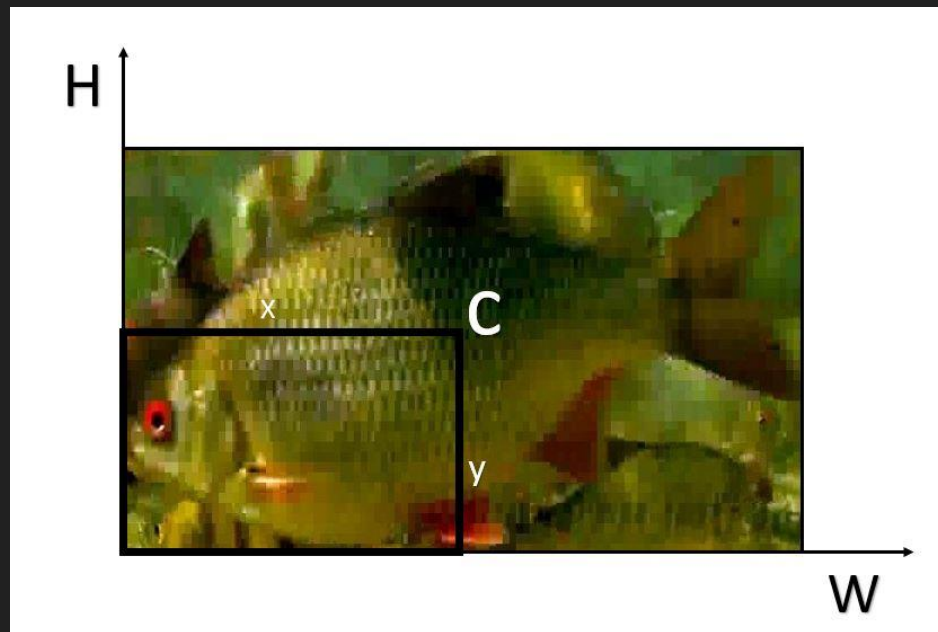
Haar Feature

Haar features are often used to detect faces because they work well for representing fine-scale textures.



Video Trajectory

Matrix Creation



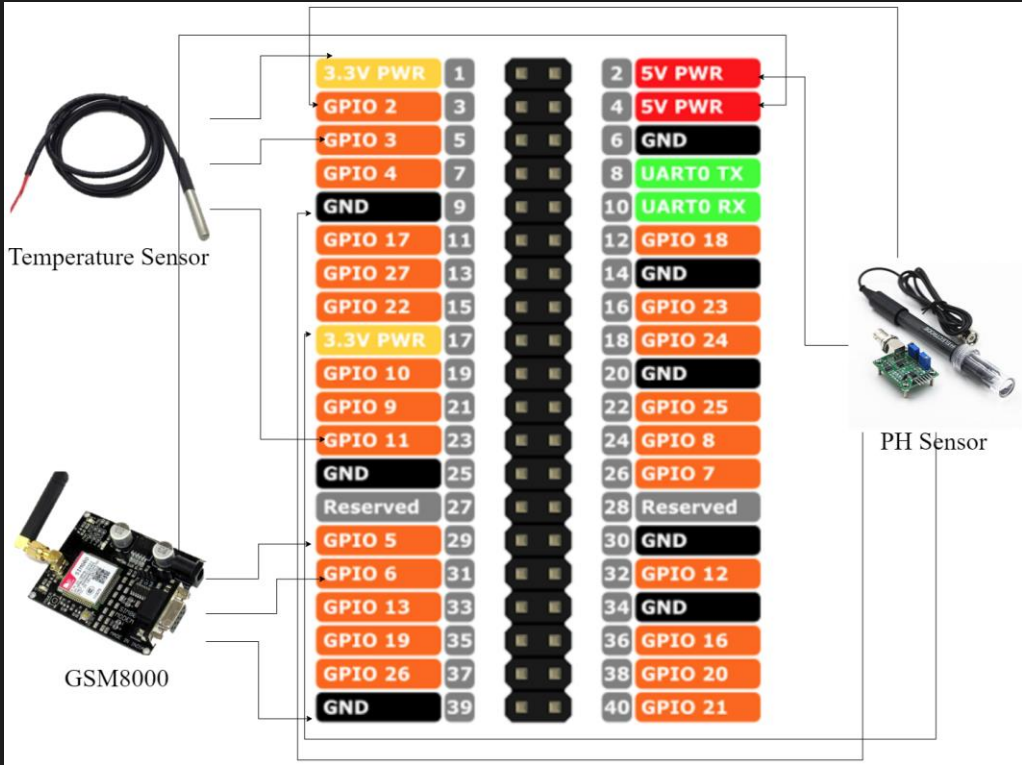
matrix2			
181x2 double			
	1	2	3
1	372	102.5000	
2	1.0745e+03	94	
3	670	199	
4	687	458	
5	458.5000	104.5000	
6	682.5000	197	
7	578	526	
8	288	331.5000	
9	428.5000	105.5000	
10	279	138.5000	
11	719	191	

Tracking-Scenario

- Time corresponding to arrival at each waypoint in seconds, The first elements of Time Of Arrival must be 0.
- Waypoints: calculate the position for each bonding box in the matrix
- Velocity in the navigation coordinate system at each way point in meters per second, specified as an N-by-3 matrix

1	2			3		
Time	Position			Velocity		
0	-10.0800	202.6500	33.9418	-11.2800	8.8100	-0.0588
1.9608	-21.1200	211.2700	33.8846	-11.2500	8.7900	-0.0577
2.9412	-32.1300	219.8800	33.8287	-11.2100	8.7800	-0.0565
3.9216	-43.1000	228.4900	33.7738	-11.1600	8.7800	-0.0554
4.9020	-54	237.1000	33.7201	-11.0900	8.7900	-0.0543
5.8824	-64.8500	245.7300	33.6674	-11.0300	8.8100	-0.0531
6.8627	-75.6200	254.3800	33.6158	-10.9500	8.8400	-0.0520
7.8431	-86.3100	263.0700	33.5654	-10.8600	8.8800	-0.0510
8.8235	-96.9100	271.8000	33.5159	-10.7600	8.9300	-0.0499
9.8039	-107.4100	280.5800	33.4676	-10.6600	8.9900	-0.0488
10.7843	-117.8100	289.4300	33.4202	-10.5500	9.0500	-0.0478
11.7647	-128.0900	298.3400	33.3739	-10.4200	9.1300	-0.0467
12.7451	-138.2500	307.3200	33.3286	-10.2900	9.2100	-0.0457
13.7255	-148.2700	316.3900	33.2843	-10.1500	9.2900	-0.0447
14.7059	-158.1500	325.5400	33.2410	-10	9.3800	-0.0437
15.6863	-167.8800	334.7900	33.1987	-9.8400	9.4800	-0.0427
16.6667	-177.4500	344.1400	33.1574	-9.6700	9.5900	-0.0417
17.6471	-186.8500	353.5900	33.1170	-9.5000	9.6900	-0.0407
18.6275	-196.0600	363.1500	33.0776	-9.3100	9.8100	-0.0397
19.6078	-205.1000	372.8200	33.0391	-9.1100	9.9200	-0.0388
20.5882	-213.9300	382.6100	33.0016	-8.9100	10.0400	-0.0378
21.5686	-222.5500	392.5100	32.9649	-8.6900	10.1600	-0.0369
22.5490	-230.9600	402.5400	32.9292	-8.4600	10.2800	-0.0360
23.5294	-239.1400	412.6800	32.8944	-8.2300	10.4100	-0.0351
24.5098	-247.0900	422.9400	32.8604	-7.9800	10.5300	-0.0342

Hardware



Human Interface Design

Sign Up form

👤 Username:

✉ Email:

☎ Phone:

🔒 Password:

🔒 Confirm Password:

SUBMIT

Already have an account? [Sign in](#)

Home

Profile >

Edit Profile >

Latest Statisitcs >

Report >

Statistics

From: to :

Sunday

Temperature Rate :

50 KL

Oxygen Rate:

5645564

PH Rate:

5645564

Readings

Today :

Sunday

Temperature Rate :

50 KL

Oxygen Rate:

5645564

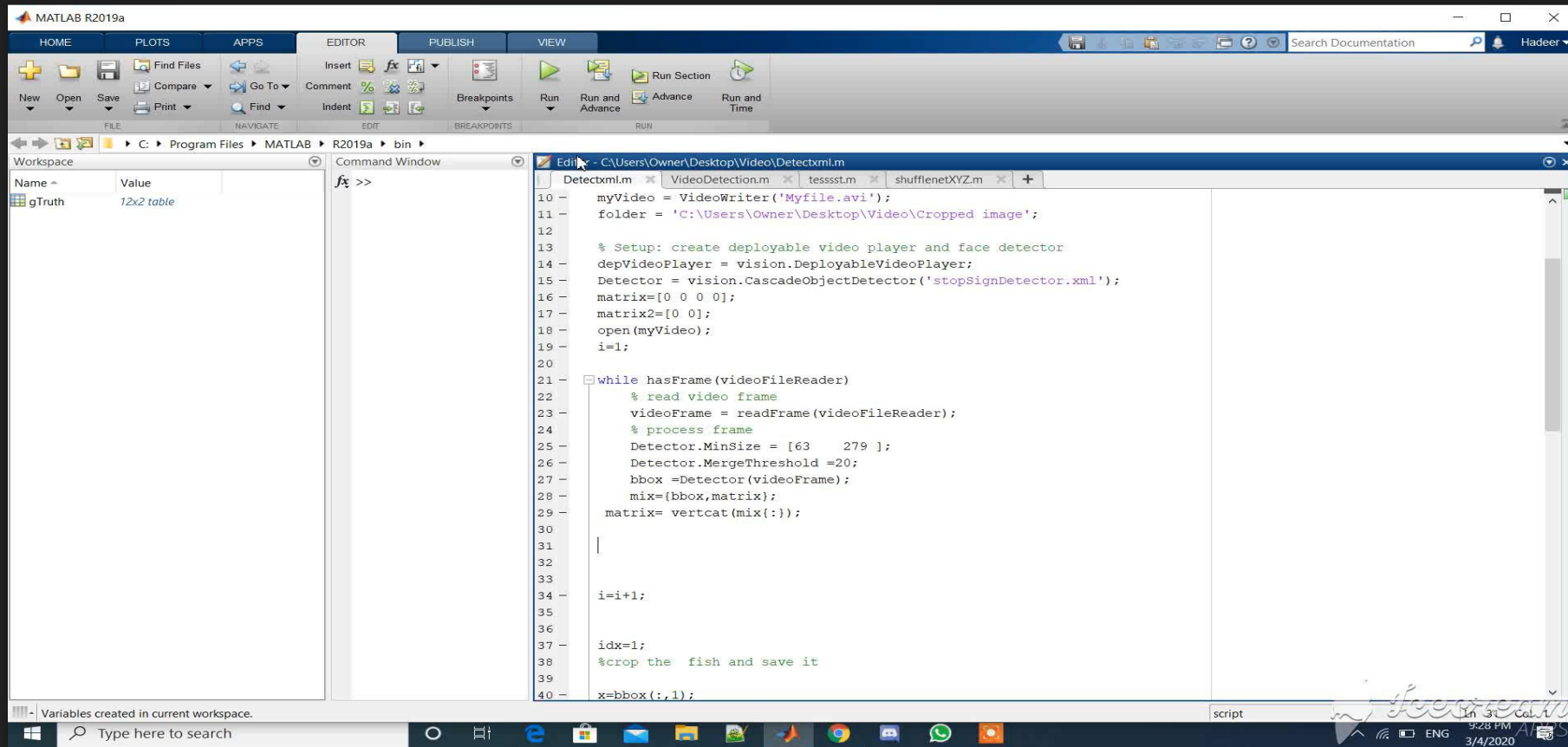
PH Rate:

5645564

Demo



Detect and cropped Fish

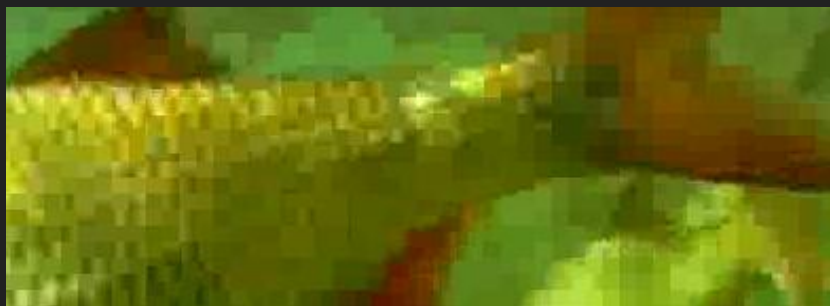


The image shows the MATLAB R2019a software interface. The main window displays a script named 'Detectxml.m' with the following code:

```
10 - myVideo = VideoWriter('Myfile.avi');
11 - folder = 'C:\Users\Owner\Desktop\Video\Cropped image';
12
13 % Setup: create deployable video player and face detector
14 - depVideoPlayer = vision.DeployableVideoPlayer;
15 - Detector = vision.CascadeObjectDetector('stopSignDetector.xml');
16 - matrix=[0 0 0 0];
17 - matrix2=[0 0];
18 - open(myVideo);
19 - i=1;
20
21 - while hasFrame(videoFileReader)
22     % read video frame
23     videoFrame = readFrame(videoFileReader);
24     % process frame
25     Detector.MinSize = [63 279];
26     Detector.MergeThreshold =20;
27     bbox =Detector(videoFrame);
28     mix=(bbox,matrix);
29     matrix= vertcat(mix(:));
30
31
32
33
34 - i=i+1;
35
36
37 - idx=1;
38 %crop the fish and save it
39
40 - x=bbox(:,1);
```

The Workspace window on the left shows a variable named 'gTruth' with a value of '12x2 table'. The Command Window shows the MATLAB prompt 'f>>'. The bottom of the screen shows the Windows taskbar with the search bar and system tray.

Sample of crop Images



Matrix Calculation

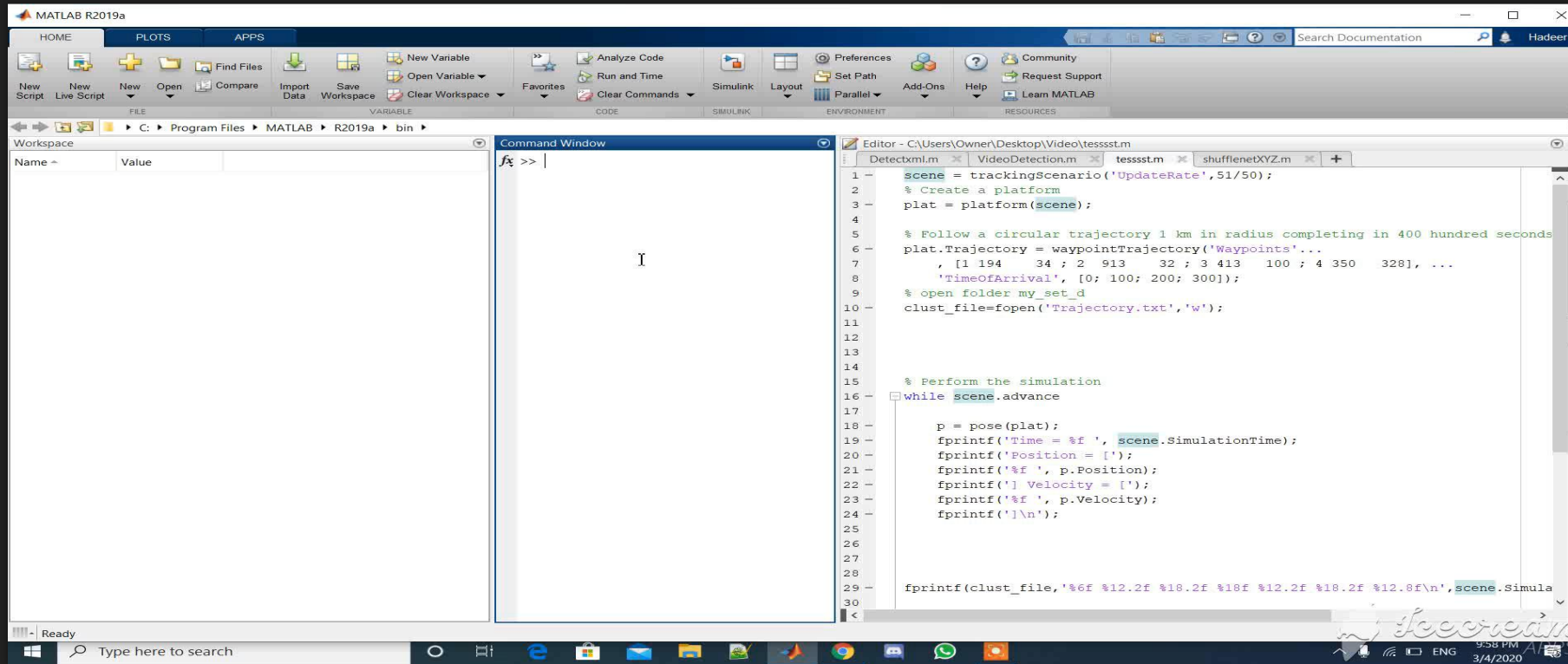
The image displays the MATLAB R2019a environment. The main window shows a script named 'Detectx.m' with the following code:

```
1 fish Detect
2 fish Detect
3 fish Detect
4 fish Detect
5 fish Detect
6 fish Detect
7 fish Detect
8 fish Detect
9 fish Detect
10 fish Detect
11 fish Detect
12 fish Detect
13 fish Detect
14 fish Detect
15 fish Detect
16 fish Detect
17 fish Detect
18 fish Detect
19 fish Detect
20 fish Detect
21 fish Detect
22 fish Detect
23 fish Detect
24 fish Detect
25 fish Detect
26 fish Detect
27 fish Detect
28 fish Detect
29 fish Detect
30 fish Detect
31 fish Detect
32 fish Detect
33 fish Detect
34 fish Detect
35 fish Detect
36 fish Detect
37 fish Detect
38 fish Detect
39 fish Detect
40 fish Detect
41 fish Detect
42 fish Detect
43 fish Detect
44 fish Detect
45 fish Detect
46 fish Detect
47 fish Detect
48 fish Detect
49 fish Detect
50 fish Detect
51 fish Detect
52 fish Detect
53 fish Detect
54 fish Detect
55 FishCrop=imcrop(videoFrame,bbox(j,1:4));
56 baseFileName = sprintf('%d.png', i); % e.g. "1.png"
57 fullFileName = fullfile(folder, baseFileName); % No need to worry about slashes now!
58 imwrite(FishCrop, fullFileName);
59 i=i+1;
60
61 end
62
63
64 %annotatedImage = insertShape(img,"rectangle",bbox,"Fish");
65 %figure; imshow(detectedImg);title('Detect Frame');
66
67 n=size(bbox,1);
68 str_n=num2str(n);
69 str=strcat(str_n,' fish Detect');
70 disp(str);
71
72 % Display video frame to screen
73 %depVideoPlayer(videoFrames);
74 % Write frame to final video file
75 writeVideo(myVideo, videoFrames);
76
77 pause(1/videoFileReader.FrameRate);
78
79 end
80
81 close(myVideo)
82
83
84
85
```

The Command Window shows the output of the script, which is a list of 'fish Detect' messages, one for each frame from 1 to 85. The Workspace window shows various variables defined in the script, such as 'baseFileName', 'bbox', 'depVideoPlay...', 'Detector', 'FishCrop', 'folder', 'fullFileName', 'gTruth', 'H', 'i', 'idx', 'imageLabelin...', 'imDir', 'j', 'matrix', 'matrix2', 'mix', 'mix2', 'mix3', 'myVideo', 'n', 'negativeFolder', 'negativeImag...', 'positiveInstan...', 'str', 'str_n', 'videoFileRea...', 'videoFrame', 'videoFrames', 'w', 'x', 'x2', 'y', and 'y2'.

The bottom of the screen shows the Windows taskbar with the search bar and system tray. The system tray includes the date and time (3/4/2020, 9:45 PM) and the language setting (ENG).

Tracking-Scenario



The image shows the MATLAB R2019a software interface. The main window is titled "Editor - C:\Users\Owner\Desktop\Video\tesssst.m" and contains the following MATLAB code:

```
1 scene = trackingScenario('UpdateRate',51/50);
2 % Create a platform
3 plat = platform(scene);
4
5 % Follow a circular trajectory 1 km in radius completing in 400 hundred seconds
6 plat.Trajectory = waypointTrajectory('Waypoints'...
7 , [1 194 34 ; 2 913 32 ; 3 413 100 ; 4 350 328], ...
8 'TimeOfArrival', [0; 100; 200; 300]);
9 % open folder my_set_d
10 clust_file=fopen('Trajectory.txt','w');
11
12
13
14
15 % Perform the simulation
16 while scene.advance
17
18     p = pose(plat);
19     fprintf('Time = %f ', scene.SimulationTime);
20     fprintf('Position = [');
21     fprintf('%f ', p.Position);
22     fprintf('] Velocity = [');
23     fprintf('%f ', p.Velocity);
24     fprintf(']\n');
25
26
27
28
29 fprintf(clust_file,'%6f %12.2f %18.2f %18f %12.2f %18.2f %12.8f\n',scene.Simula
30
```

The interface also shows the Command Window with the prompt ">>|" and the Workspace window which is currently empty.

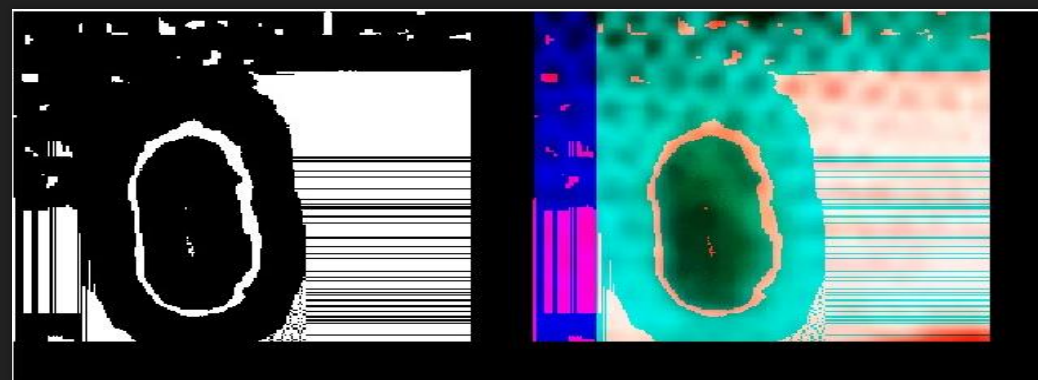
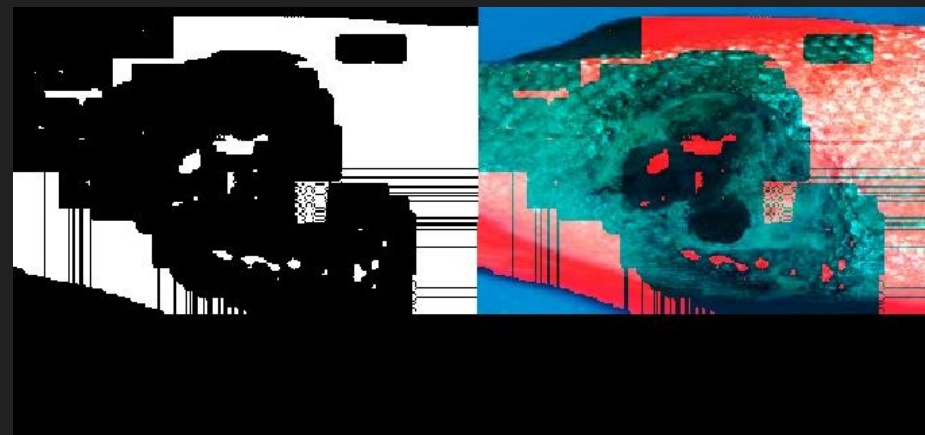
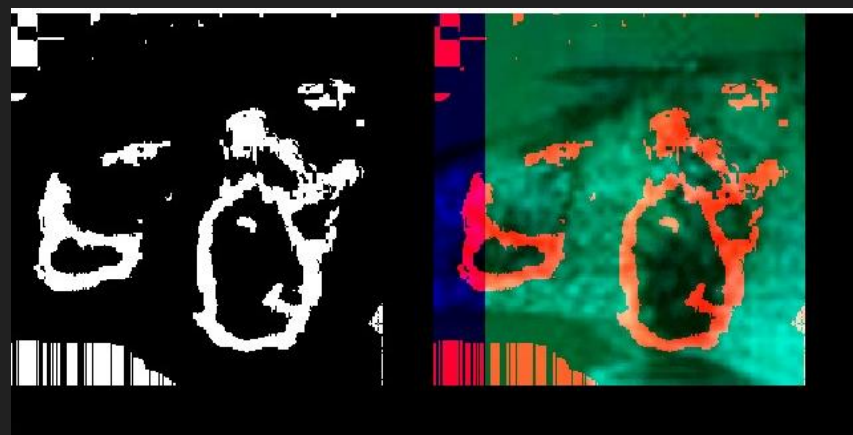
Segmentation

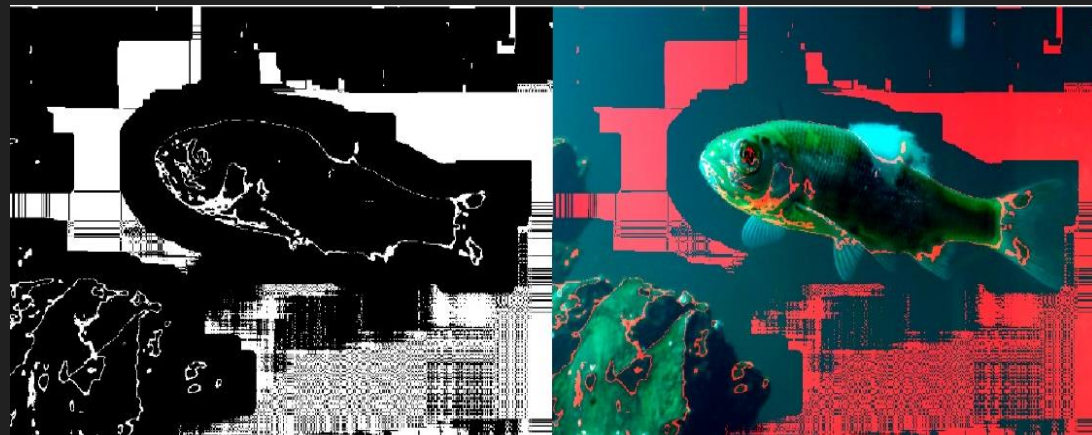
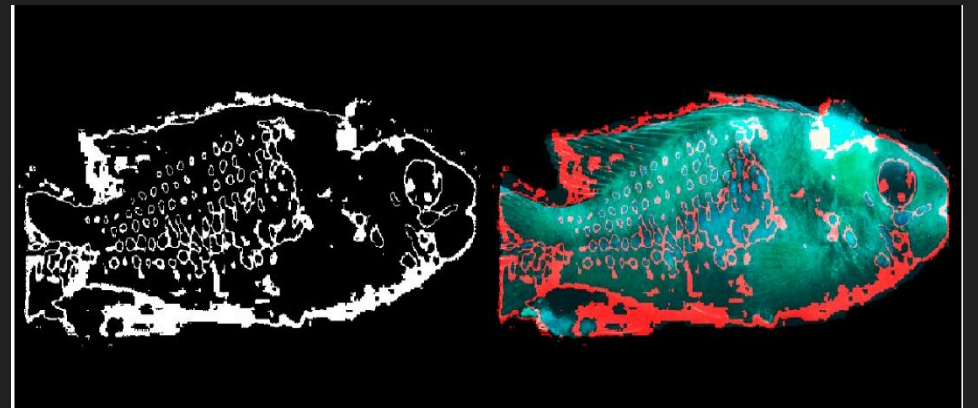
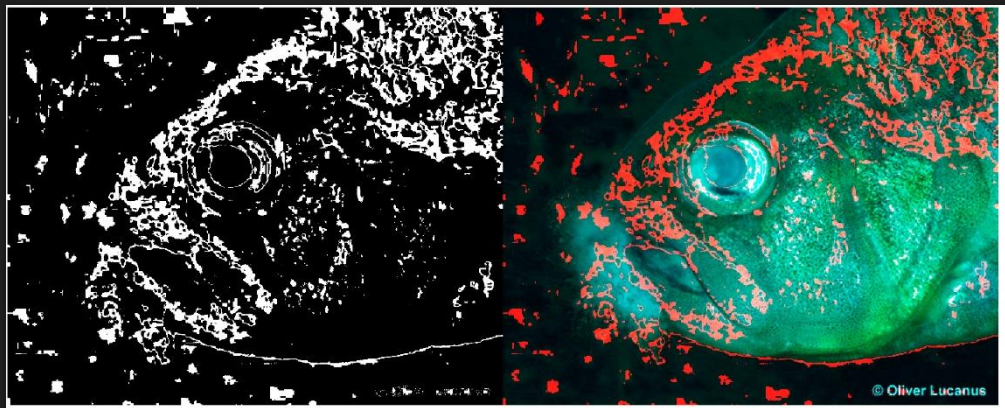
The image displays the MATLAB R2019a software interface. The main window shows a script named 'test.m' in the Editor, which implements a skin color segmentation algorithm. The script reads an image, converts it to YCbCr space, and applies a Gaussian distribution model to the chromatic channels to segment skin pixels. The Workspace window on the right shows the current state of variables.

```
27 - colorchart = zeros(256);
28 - for b = 0:255
29 -     for r = 0:255
30 -         x = [(double(b) - double(bmean)); (double(r) - double(rmean))];
31 -         colorchart(b+1,r+1) = exp(-0.5* double(x)'*invy(double(brcov))*double(x));
32 -     end
33 - end
34
35
36
37 - img = imread('C:\Users\Kareem\Downloads\matlap\segmentation\Dataset\eus\Eus Test\eus5');
38 - ycbcr1 = rgb2ycbcr(img);
39 - cb = ycbcr1(:,:,2);
40 - cb = filter2(lpf, cb);
41 - cb = reshape(cb, 1, prod(size(cb)));
42 - cr = ycbcr1(:,:,3);
43 - cr = filter2(lpf, cr);
44 - cr = reshape(cr, 1, prod(size(cr)));
45
46 % Create a Gaussian distribution of the chromatic skin model.
47 - res = zeros(255,255);
48 - for b1 = 1:size(ycbcr1,1)
49 -     for r1 = 1:size(ycbcr1,2)
50 -         x1 = [double((ycbcr1(b1,r1,2)) - double(bmean)); (double(ycbcr1(b1,r1,3)) - dou
51 -             res(b1,r1) = exp(-0.5* x1'*invy(brcov)*x1);
52 -     end
53 - end
54
55 %imshow(res);
56
```

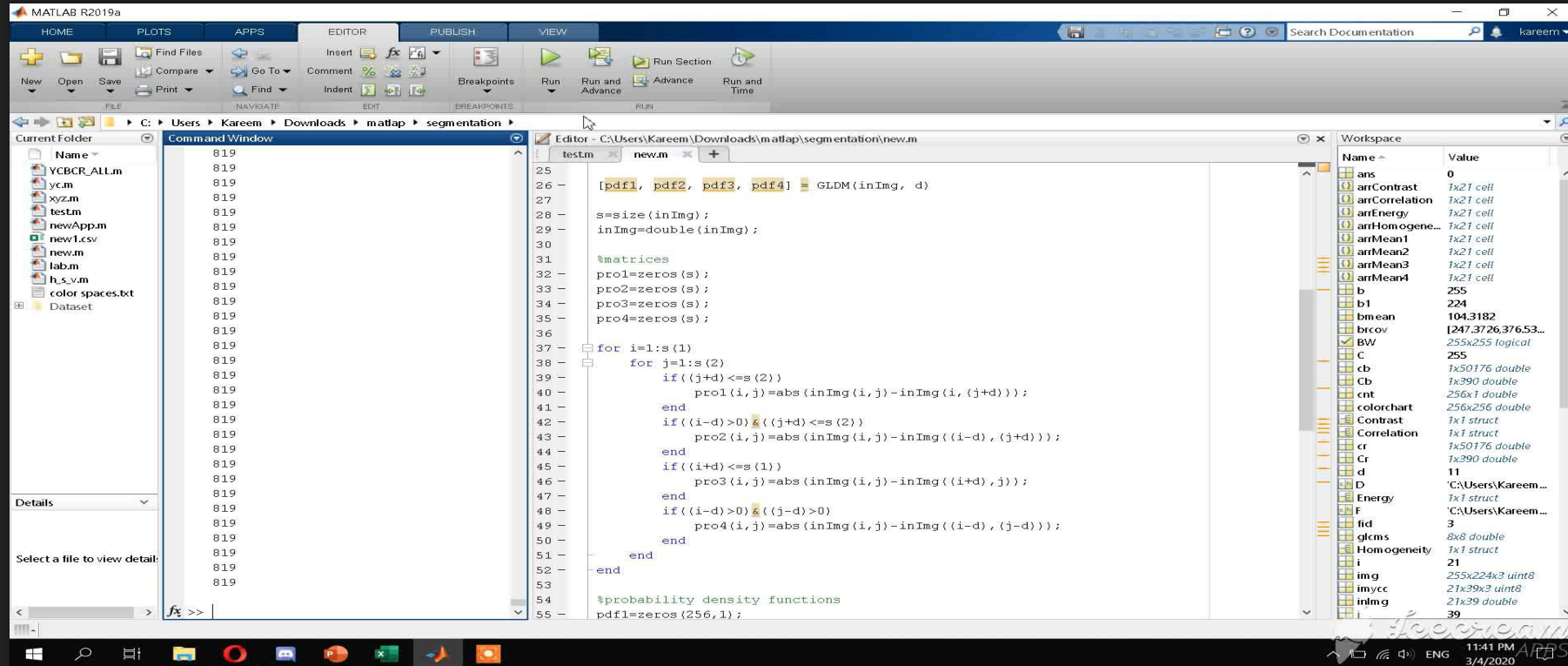
Name	Value
b	255
b1	224
bmean	104.3182
brcov	[247.3726 376.53...
BW	255x255 logical
C	255
cb	1x50176 double
Cb	1x390 double
colorchart	256x256 double
cr	1x50176 double
Cr	1x390 double
D	'C:\Users\Kareem...
F	'C:\Users\Kareem...
img	255x224x3 uint8
imyc	15x26x3 uint8
k	30
lpf	[0.1111,0.1111,0...
npixel	0
pixel	0
R	255
r	255
r1	224
res	255x255 double
rmean	158.5490
S	30x1 struct
T	255x255 double
value	204
x	[150.6818;96.451...
x1	[8;-21.5490]
ycbcr	15x26x3 uint8
ycbcr1	224x224x3 uint8

Sample for Segmentation





Texture Feature



The image displays the MATLAB R2019a environment with a script named 'new.m' open in the Editor. The script performs texture feature extraction using GLDM (Gray Level Dependence Matrices). It starts by reading an image 'inImg' and a distance 'd'. It then generates four matrices (pro1, pro2, pro3, pro4) based on the image and distance. The script uses nested loops to calculate the absolute differences between pixels at distance 'd' in four directions (up, down, left, right). The results are stored in a cell array 'pdf1' through 'pdf4'. Finally, it calculates the probability density functions for these matrices.

```
25  
26 [pdf1, pdf2, pdf3, pdf4] = GLDM(inImg, d)  
27  
28 s=size(inImg);  
29 inImg=double(inImg);  
30  
31 %matrices  
32 pro1=zeros(s);  
33 pro2=zeros(s);  
34 pro3=zeros(s);  
35 pro4=zeros(s);  
36  
37 for i=1:s(1)  
38     for j=1:s(2)  
39         if ((j+d)<=s(2))  
40             pro1(i,j)=abs(inImg(i,j)-inImg(i,(j+d)));  
41         end  
42         if ((i-d)>0) && ((j+d)<=s(2))  
43             pro2(i,j)=abs(inImg(i,j)-inImg((i-d),(j+d)));  
44         end  
45         if ((i+d)<=s(1))  
46             pro3(i,j)=abs(inImg(i,j)-inImg((i+d),j));  
47         end  
48         if ((i-d)>0) && ((j-d)>0)  
49             pro4(i,j)=abs(inImg(i,j)-inImg((i-d),(j-d)));  
50         end  
51     end  
52 end  
53  
54 %probability density functions  
55 pdf1=zeros(256,1);
```

The Workspace window shows the following variables and their values:

Name	Value
ans	0
arrContrast	1x21 cell
arrCorrelation	1x21 cell
arrEnergy	1x21 cell
arrHomogene...	1x21 cell
arrMean1	1x21 cell
arrMean2	1x21 cell
arrMean3	1x21 cell
arrMean4	1x21 cell
b	255
b1	224
bmean	104.3182
brcov	[247.3726,376.53...
BW	255x255 logical
C	255
cb	1x50176 double
Cb	1x390 double
cnt	256x1 double
colorchart	256x256 double
Contrast	1x1 struct
Correlation	1x1 struct
cr	1x50176 double
Cr	1x390 double
d	11
D	'C:\Users\Kareem ...
Energy	1x1 struct
F	'C:\Users\Kareem ...
fid	3
glcms	8x8 double
Homogeneity	1x1 struct
i	21
img	255x224x3 uint8
imyc	21x39x3 uint8
inImg	21x39 double
i	39



Thanks!

