DHBW
Duale Hochschule
Baden-Württemberg

IR CAMERA HUMAN DETECTION

Batu Kaan Özen

Hande Yıldırım

SUPERVISOR: Prof.Till Hänisch

Contents

1. Introduction

Thermal imaging is a very powerful remote sensing technique for a number of reasons. Some of them are their efficiently, success and also not being like normal camera. People can watch other people with normal camera but they can not do it with normal Camera. It is complicated situation. Because of that this kind of cameras are common, where normal camera is not permitted, we can apply Thermal Camera Image processing to detect how many people in Room or what are these people doing etc. In our project, we tried to detect how many people in one room and as a result, our success is about %75.

2. System Hardware Design

In our project, we selected Teensy 3.6 as our Micro-controller because of its ARM cortex processer (32bit 180MHZ). It is very fast, and in this project we are going to make a lot of mathematical calculation with big sized arrays. For this reason, Teensy 3.6 is perfect selection for us.

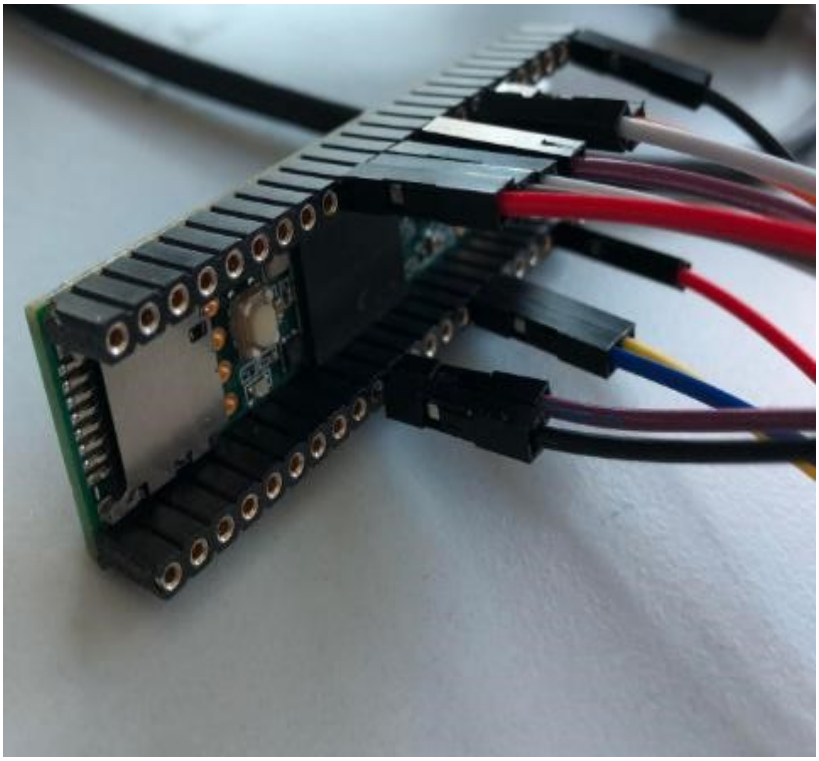


Figure 1 : Teensy 3.6 connected with other modules

Sparkfun Qwiic IR Array MLX90640 is selected as IR camera in our project because of its wide angle detecting (110-70 degrees). It is wide area and it is perfect selection for our project.

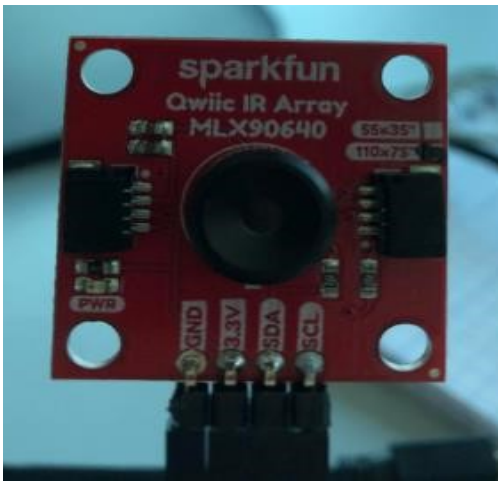


Figure 2: Sprakfun Qwiic IR Array MLX90640 (110x75 degrees)

Lastly, we needed to save our data and we found best way as wireless connection with one server and send this data to this server. Because of that, we need one wireless module and an Adafruit Airlift Wifi is selected in our project. It has a very nice pre-written library.

Figure 3: Adafruit Airlift Wifi

After that we needed a box to put our microcontroller and camera. The box is designed by using Fusion360 and printed with 3D printer.

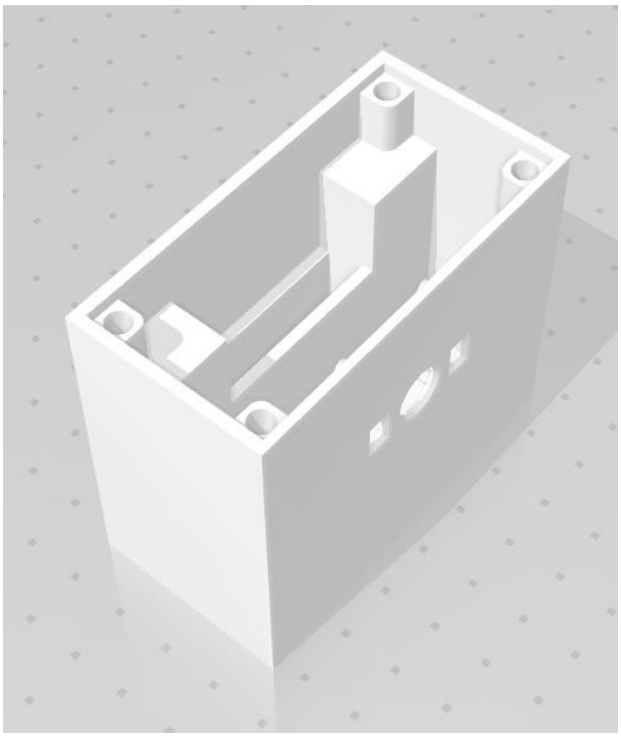


Figure 4: Design of box

You can see the printed version of the box below. It is easy to attach with band to wall (you can see below).

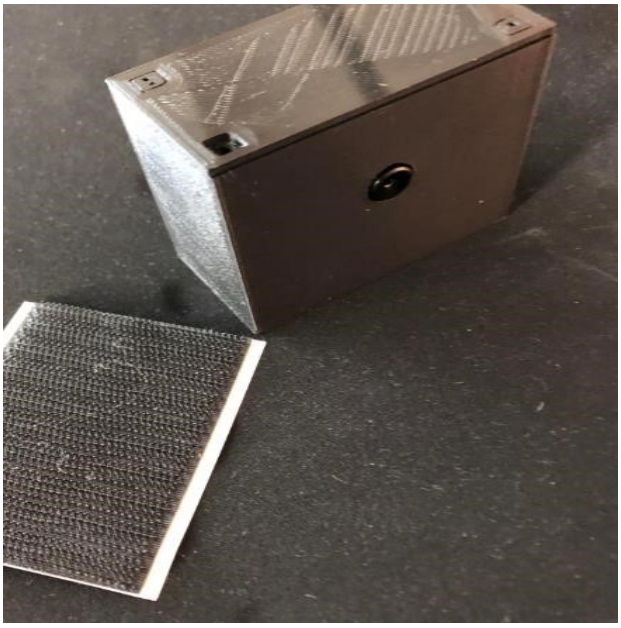


Figure 5: Band and 3D printed box

We made SPI communication between Teensy with Airlift-WİFİ. ADAFRUİT AİRLİFT WİFİ is connected to Teensy 3.6 for SPI communication like that: 13.Teensy pin to SCK , 12 Teensy pin to MISO 11.Teensy pin to MOSI 5.Teensy pin to CS 9.Teensy pin to BUSY 6.Teensy pin to REST and ground to ground, Vin to 3.3V. We used prewritten code for this communication.

We also made I2C communication between Teensy with MLX90640 SPARKFUN IR CAMERA. MLX90640 SPARKFUN IR CAMERA is connected like that: 18. Teensy pin to SDA and 19. Teensy pin to SCL and ground to ground, 3.3V to 3.3V. We used also prewritten code for this communication.

3. System Software Design

In this part, I am going to explain Mathematical logic and also algorithm in my code: Below you can see the system's work principle.

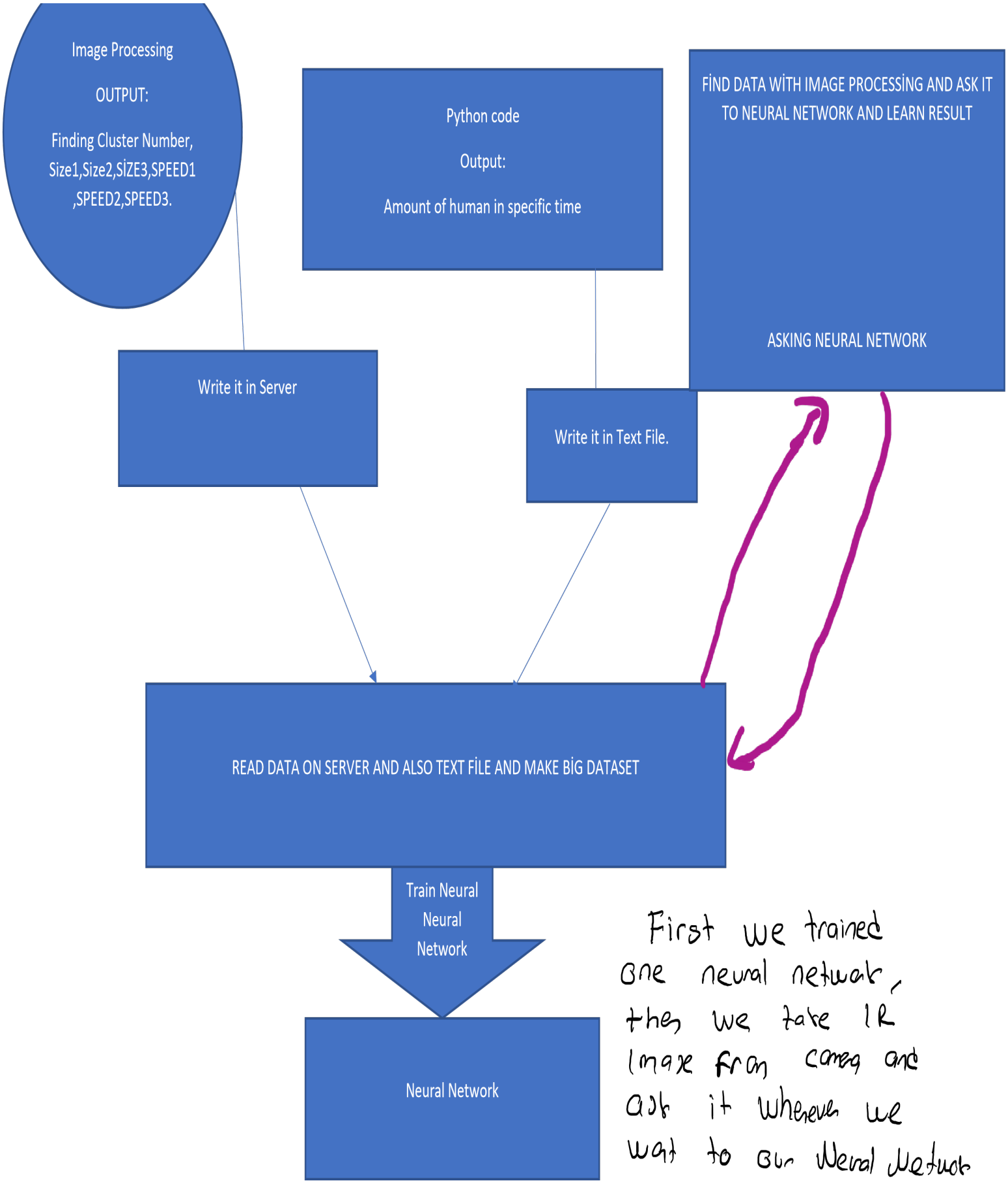


Image Processing
OUTPUT:
Finding Cluster Number, Size1,Size2,SİZE3,SPEED1 ,SPEED2,SPEED3.
Python code
Output:
Amount of human in specific time
FİND DATA WİTH IMAGE PROCESSİNG AND ASK IT TO NEURAL NETWORK AND LEARN RESULT
ASKING NEURAL NETWORK
Write it in Server
Write it in Text File.
READ DATA ON SERVER AND ALSO TEXT FİLE AND MAKE BİG DATASET
Train Neural Neural Network
Neural Network
First we trained one neural netwok, then we take IR Image from camera and ask it whenever we wat to our Neral Netwok

As you see above, my system work like that we have Teensy 3.6 board with Camera and this system taking IR Image and finding on them Features and then It will send it to server, same time there is also one python. It writes time and amount of human in room by hand in text file and then we save it after. We have one code which read data on server and also data on text file and make one dataset after we apply this neural network, we have trained neural network after that we read data from camera via Server and send it to Neural Network and we have the result, How many are people in room.

3.1. Finding Feature On IR Image

This part is mainly about image processing and feature finding on IR Image. Firstly, the image was taken via pre-written Teensy 3.5 MLX90640 code and it was like in Figure 6:

```
24,24,24,24,24,24,23,23,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,
24,24,24,25,24,24,24,24,24,24,23,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,24,
24,24,23,24,24,24,24,24,24,24,23,24,24,25,24,26,26,26,25,24,25,24,24,24,24,24,24,24,24,23,24,24,
24,24,24,24,24,24,23,24,24,24,24,24,25,24,28,25,28,28,26,27,24,25,24,24,24,24,24,24,23,24,24,24,
24,24,24,23,24,24,23,24,24,24,24,25,25,28,27,31,30,29,30,29,29,24,25,24,24,23,24,24,24,24,24,24,
24,25,24,23,24,24,24,24,24,24,26,24,29,27,30,30,30,30,30,30,26,29,23,25,24,24,24,24,23,25,24,24,
24,24,23,24,24,23,24,24,24,25,24,30,28,31,31,30,30,31,30,30,30,26,27,24,24,24,23,24,24,23,24,24,
24,24,24,23,23,23,24,23,24,24,29,25,31,30,32,31,32,31,31,31,28,30,24,27,23,25,24,24,24,24,24,24,
24,24,23,24,24,24,23,24,23,24,24,31,29,32,32,34,33,34,34,32,33,28,29,24,24,24,24,24,24,24,24,24,
24,23,23,24,23,24,24,24,24,24,29,25,33,31,33,33,34,34,33,34,30,32,24,27,24,25,24,24,24,24,24,24,
24,24,24,24,24,23,23,23,24,24,24,32,29,33,33,34,34,34,34,33,33,29,30,24,25,24,24,24,24,24,24,24,
24,24,24,24,24,24,24,24,23,24,28,24,33,30,33,33,34,33,34,34,31,33,25,28,23,24,24,24,24,24,24,24,
24,24,23,23,24,24,23,24,24,24,23,29,27,33,33,33,34,33,34,33,34,28,30,24,25,24,24,24,24,24,24,25,
24,24,23,24,23,23,23,24,24,24,25,24,32,28,34,33,33,34,33,33,29,32,24,28,23,24,23,24,24,25,24,24,
24,24,24,24,24,24,24,24,24,23,24,26,25,33,32,33,34,33,33,33,33,25,27,24,24,23,24,24,24,24,24,25,
24,24,24,24,24,24,23,24,24,24,24,24,31,27,34,33,32,33,33,33,28,32,24,25,23,24,24,24,24,24,24,24,
24,24,23,24,24,24,24,24,25,27,27,28,29,32,31,33,33,33,33,33,33,30,29,26,25,24,24,24,24,24,24,24,
24,25,24,24,24,24,24,24,29,28,28,29,32,31,31,31,33,32,33,33,33,34,28,29,25,26,24,25,24,24,24,24,
25,24,24,24,24,25,27,30,30,30,29,29,30,31,31,33,32,33,33,34,34,32,32,28,28,28,27,27,24,25,25,25,
24,24,25,24,26,26,30,30,29,30,29,30,31,30,31,31,33,32,34,34,34,34,28,31,29,28,30,28,26,25,25,24,
25,25,25,28,29,30,30,30,29,29,29,29,29,31,31,31,32,32,32,34,34,30,32,29,29,30,30,30,30,27,26,25,
25,25,29,29,30,30,30,31,29,30,30,29,28,29,31,31,31,31,33,32,31,33,28,29,30,30,30,30,29,30,26,26,
28,30,31,31,31,32,32,30,30,29,29,30,30,28,29,31,31,31,31,31,31,29,29,30,30,29,29,30,30,30,28,25,
31,31,31,31,31,31,31,31,30,30,29,29,29,30,30,29,30,30,31,31,29,30,29,29,29,30,29,30,30,30,27,28,
```

Figure 6

After that, I planned to apply one adaptive system which is going to calculate average of heat values on Images in a specific time without beginning from the first. Then it calculates IR camera heats value minus average IR camera value. Then this system going to apply 3degree threshold to result and we receive our result. You can see average calculation code in Figure 7 and threshold code in Figure 8. Following these steps, I can detect human heat pixels. The result of this system is like Figure 9.

Attention: One problem about this system is that, we need to start it in environment without human because of its adaptive working principles.

```
void FINDaverage(){
  for (int i=0;i<24;i++){
    for(int j=0;j<32;j++){
      float ara =average[j][i];
      average[j][i]=((float)ara*(float)averagecounter+(float)Image[j][i])/((float)averagecounter+1);
    }
  }
  int solver=0;
  solver = averagecounter;
  averagecounter=solver+1;


  if (averagecounter==30000){
    averagecounter=15000;
  }
  delay(1000);
```

Figure 7: Coding part I

```
void noiseReduction(){
  int oneStepBeforeImage[32][24];
  for (int i=0;i<24;i++){
    for(int j=0;j<32;j++){
        oneStepBeforeImage[j][i]=Image[j][i];
      }
    }
    for(int i=0;i<24;i++){
      for(int j=0;j<32;j++){
        Image[j][i]=oneStepBeforeImage[j][i]-average[j][i];
      }
    }
  for (int i=0;i<24;i++){
    for(int j=0;j<32;j++){
        oneStepBeforeImage[j][i]=Image[j][i]>3;
      }
    }
   for (int i=0;i<24;i++){
    for(int j=0;j<32;j++){
        Image[j][i]=oneStepBeforeImage[j][i];
      }
    }
```

Figure 8: Coding Part II

```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,0,0,0,
0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,0,0,
0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,0,0,
0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,0,0,
0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,1,1,1,0,0,1,1,1,1,1,1,1,1,1,0,0,
0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,1,1,0,0,0,0,1,1,1,1,1,1,1,1,1,0,
0,0,0,1,1,1,1,1,1,1,1,0,0,1,1,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,
0,0,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,
```

Figure 9: (It is me sitting across IR camera)

After that, I continued with Feature Detection Part. In this IR camera data, I can see 3 important features. They are amount of feature on Image, size of these features and amount of their movement. Firstly, I started with amounts of these features and when I was finding these amounts, I separated each cluster in a new array to find their size and their movement. You can see my code in Figure 10 and Figure 11 and see my result in Figure 12:

```
void findDatasonImage(){
  boolean ImageControl;
  numberofCluster=0;
  for (int i=0;i<24;i++){
      for(int j=0;j<32;j++){
         ImageControl = beforedetected(j,i);
         if ( ImageControl){
            continue;
         }
         clusterAlgorithm(j,i,numberofCluster);
         //sendDetectionMatrix();
         numberofCluster = numberofCluster +1;
       // Serial.print("number of clusters");
       // Serial.println(numberofCluster);
      }
  }
}


boolean beforedetected(int x, int y){
  if (detectionMatrix[x][y]==0 and Image[x][y]==1){
    return false;
  }
  return true;
}
```

Figure 10: Coding Part III

```
void clusterAlgorithm(int x, int j, int numberofCluster ){

   object[numberofCluster][x][j]=1;
   detectionMatrix[x][j] =1;

  if((x-1>=0) and (j-1>=0) and (detectionMatrix[x-1][j-1]==0) and (Image[x-1][j-1]==1) ){

    clusterAlgorithm(x-1,j-1,numberofCluster);
  }

  if((j-1>=0) and (detectionMatrix[x][j-1]==0) and  (Image[x][j-1]==1) ){

    clusterAlgorithm(x,j-1,numberofCluster);
  }

  if((x+1<=32) and (j-1>=0) and (detectionMatrix[x+1][j-1]==0) and (Image[x+1][j-1]==1 )){

    clusterAlgorithm(x+1,j-1,numberofCluster);
  }

  if((x+1<=32) and (detectionMatrix[x+1][j]==0) and (Image[x+1][j]==1 ) ){

    clusterAlgorithm(x+1,j,numberofCluster);
  }

  if((x+1<=32) and (j+1<=24) and (detectionMatrix[x+1][j+1]==0) and (Image[x+1][j+1]==1 )){

    clusterAlgorithm(x+1,j+1,numberofCluster);
  }

  if((j+1<=31) and (detectionMatrix[x][j+1]==0)  and (Image[x][j+1]==1)){

    clusterAlgorithm(x,j+1,numberofCluster);
  }

  if((x-1>=0) and (j+1<=24) and (detectionMatrix[x-1][j+1]==0) and (Image[x-1][j+1]==1)){

    clusterAlgorithm(x-1,j+1,numberofCluster);
  }

  if((x-1>=0) and (detectionMatrix[x-1][j]==0) and  (Image[x-1][j]==1 )){

    clusterAlgorithm(x-1,j,numberofCluster);
  }


}
```

Figure 11: Coding Part IV

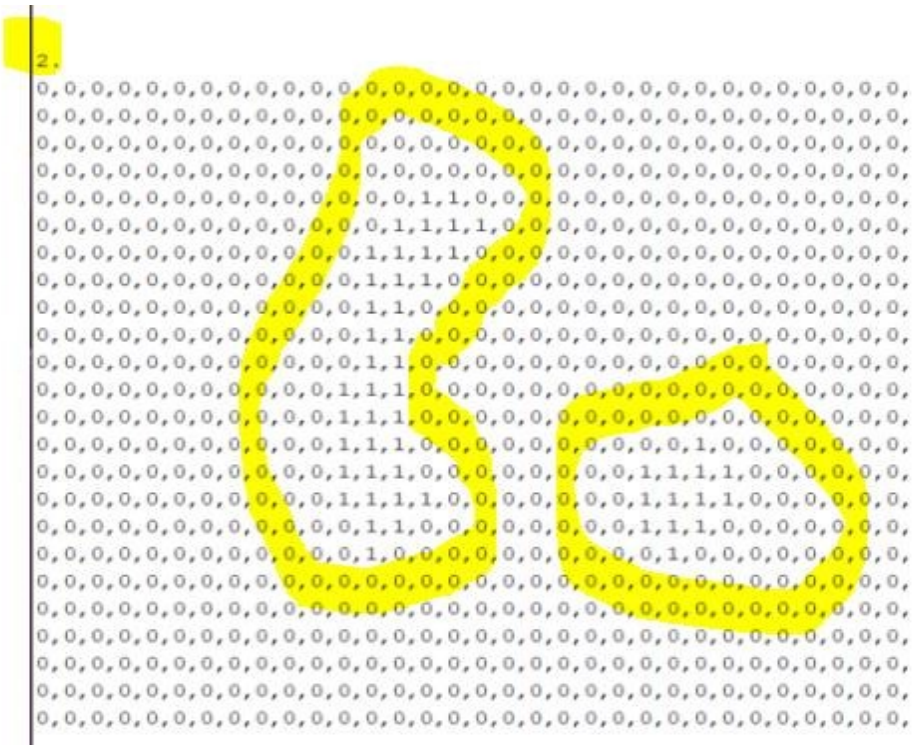


Figure 12: Amount of cluster on this Image

After completing finding amount of cluster and also separating them into different arrays, I continued my code detecting their sizes. For detecting size, I wrote the code in Figure 13:

```
void putMassofcluster(){
for(int i=0;i<10;i++){
  data[i+10]=calculateMass(i);
}
}

int calculateMass(int i){
  int Mass =0;
    for (int k=0 ; k < 24 ;k++){
      for (int j=0; j<32; j++){
        Mass = Mass+ object[i][j][k];
      }
    }
return Mass;
}
```

Figure 13: Coding Part V

Then, I found size of cluster in Figure 14:

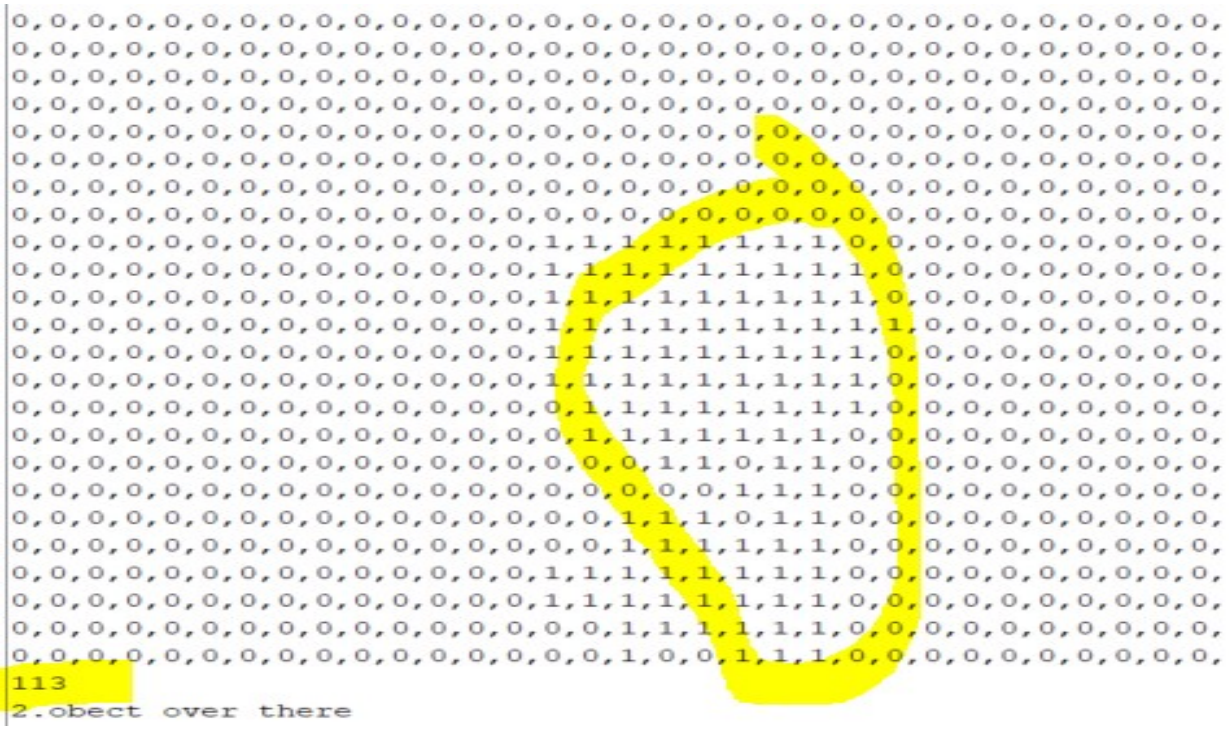


Figure 14

Lastly, I found the movement of each cluster. In this part, I took one IR Image from the camera and calculated each cluster center and then took another IR Image from camera and calculated its center and found minimum distance of each cluster center to another image cluster centers and assign this minimum distance for each cluster as a their movement. You can see this code in Figure 15 (It is like all operation) and see my center finding code in Figure 16:

MLX90640_I2C_Driver.h | Example2_OutputToProcessing - Shortcu

```
void makeDataset(){
  //We are taking first Image
  takeImageFromIRcamera();
  // We are making preprocessing
  FINDaverage();
  noiseReduction();
  //sendImageFromSerial();
  matrixInitiliz();
  findDatasonImage();
  putAmoutofcluster();
  putMassofcluster();
  //FINDİNG FIRST CENTER
  initilizeCenters();
  putAmoutofcluster();
  //TAKİNG ANOTHER IMAGE
  takeImageFromIRcamera();,
  //Preprocessing another image
  FINDaverage();
  noiseReduction();
  //Serial.println("2.");
  findDatasonImage();
  //sendImageFromSerial();
  //FINDING CENTER OF IMAGE
  initilizeSecondCenters();
  //outPuttwoImageCenter();
  //CALCULATING THEIR MOVEMENT
  putMovement();
  sendDatasetFromSerial();

}
```

Figure 15: Coding Part VI

```
void initilizeCenters(){
  amountOfCluster[0] = numberofCluster;
  for(int i=0;i<numberofCluster;i++){
    center[0][0][i]= centerLocationOnX(i);
    center[0][1][i]= centerLocationOnY(i);
  }
  //for (int i=0; i<10;i++){
  //  Serial.print(center[0][0][i]);
  //  Serial.print("X");
  //}
  //Serial.println();
  //for (int i=0; i<10;i++){
  //  Serial.print(center[0][1][i]);
   // Serial.print("Y");
  //}
}

float centerLocationOnX(int i){
  int massofObject = 0;
  massofObject = calculateMass(i);
  int moment = 0;
  for(int t=0;t<24;t++){
    for(int j=0;j<32;j++){
        moment = moment + object[i][j][
    }
  }
  int average= 0;
```

```
//Serial.println("selam");
amountOfCluster[1] = numberofCluster;
for(int i=0;i<numberofCluster;i++){
  center[1][0][i]= centerLocationOnX(i);
  center[1][1][i]= centerLocationOnY(i);
}
//Serial.println();
//for (int i=0; i<10;i++){
//  Serial.print(center[1][0][i]);

float centerLocationOnY(int i){
  int massofObject = 0;
  massofObject = calculateMass(i);
  int moment = 0;
  for(int t=0;t<24;t++){
    for(int j=0;j<32;j++){
        moment = moment + object[i][j][t]*t;
    }
  }
  int average = 0;
  average = moment/massofObject;
  //Serial.println(average);
  return average;
}
```

Figure 16: Coding Part VII

After writing these codes, I received this results in Figure 17 (They are center coordinate of clusters) and I found closest distance of each cluster.

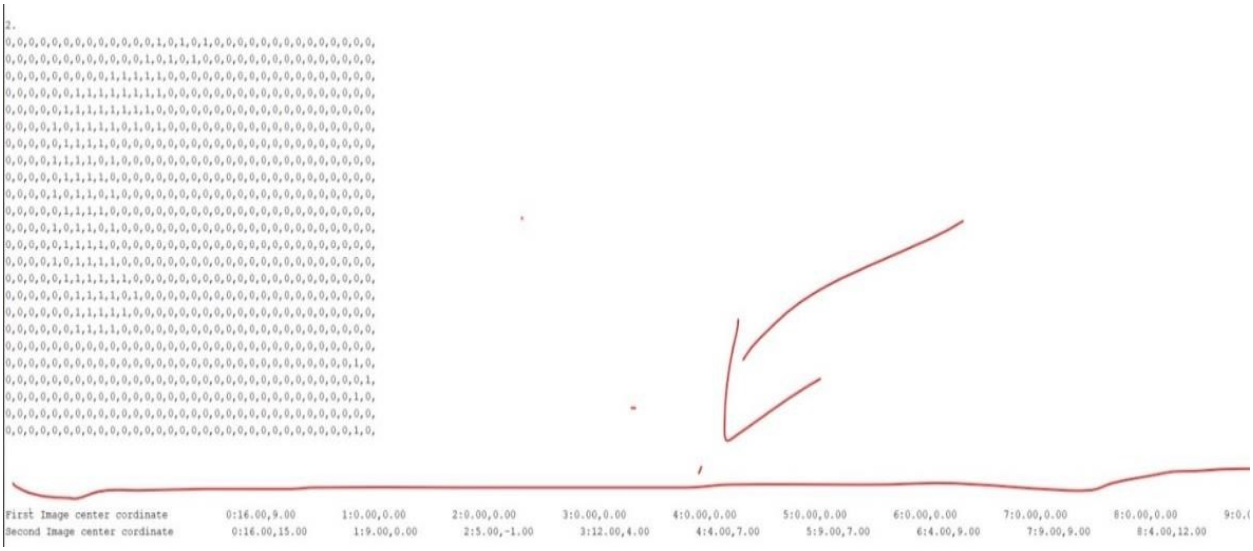


Figure 17

Attention: In this part, I assumed some situations:

First Situation:

Cluster amount of first Image > Cluster amount of second Image

I assign the minimal distance to each cluster center on first Image. In a case of empty second Image, I assigned 0.

Second Situation:

Cluster amount of first Image = Cluster amount of second Image:

I assigned the minimal distance to each cluster center on first Image.

Third Situation:

Cluster amount of first Image < Cluster amount of second Image:

I assigned the minimal distance to each cluster center on first Image.

And I did not take the root of distance because I am sending int value type.

AFTER THAT I SEND DATA SERVER AS:

AMOUNT OF CLUSTER – SİZE OF CLUSTER 1 – SİZE OF CLUSTER 2 -SİZE OF CLUSTER 3 – SPEED OF CLUSTER 1- SPEED OF CLUSTER 2 – SPEED OF CLUSTER 3

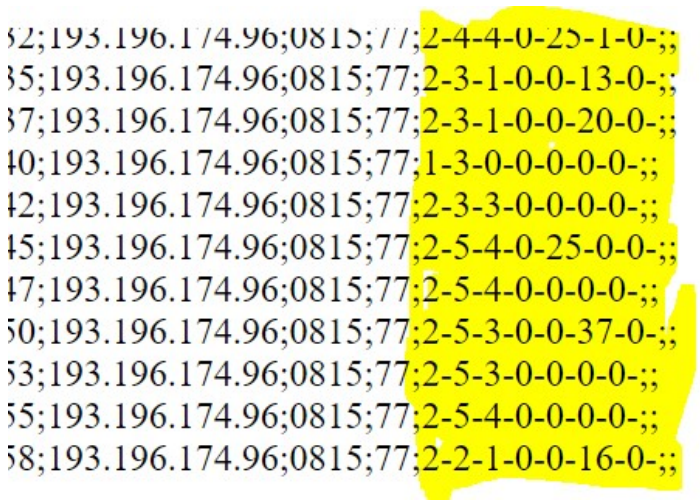


Figure 18: Data in Server

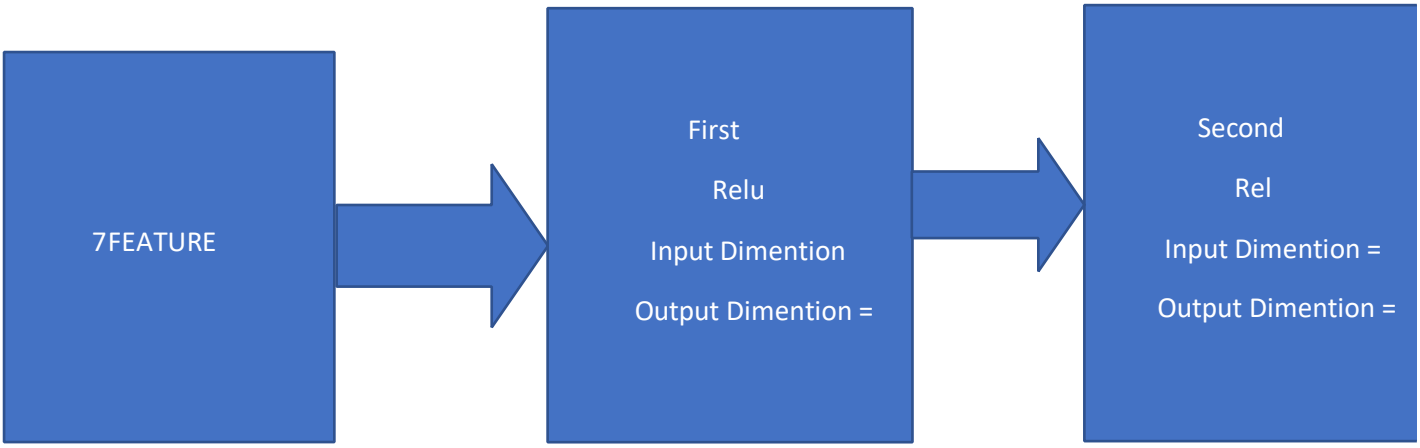
We also limited our feature set because we want to increase our score. Because of this situation we can detect maximal 3 people in room.

3.2. COLLECTING DATASET FOR MACHINE LEARNING PART

After the features are founded, we needed to collect them. Because of this situation, we found solution like that, Firstly, we took Image and then we found feature and then we send our feature in server and in this php server we save. This server easy to implement.

The link, which is below, can easily change in my code
http://193.196.175.153/es_lebt_out.php (PHP SERVER to check data)
http://193.196.175.153/es_lebt_in.php?DEV=0815&STATE=0&KEY=1234

PHP SERVER TO SEND DATA you can change state and key.



We have also one python code for entering amount of people in room and this code saves time and also amount of people like below text file.

```
0---2019-11-08 11:41:21.441902
1---2019-11-08 11:41:22.524785
2---2019-11-08 11:42:01.656909
1---2019-11-08 11:42:04.351245
3---2019-11-08 11:42:35.480150
2---2019-11-08 11:44:26.586869
3---2019-11-08 11:44:34.062277
2---2019-11-08 11:44:48.363388
3---2019-11-08 11:52:19.686792
```

Figure 19

And we have also another code to make dataset for our neural Network. Because of lack of time, we could not collect very big dataset.

3.3. APPLYING NEURAL NETWORK

Our Neural network construction is like that:

Our overall system score is : 0.7648578811369509

Our confusion matrix like below:

```
margins=True)
Out[27]:
Predicted    0  1    2   3  All
True
0.0        186  0    0   0  186
1.0         19  6    9   6   40
2.0          5  0   74  10   89
3.0         15  3   24  30   72
All        225  9  107  46  387
```

Figure 20

From this score we can say that we have 225 sample of zeros and 186 of them are true for 0, 19 of them false for 1, 5 of them are false for 2 and 15 of them are false 3.

From this score we can say that we have 9 sample of ones and 0 of them are false for 0, 6 of them true for 1, 0 of them are false for 2 and 3 of them are false 3.

From this score we can say that we have 107 sample of twos and 0 of them are false for 0, 9 of them false for 1, 74 of them are true for 2 and 24 of them are false 3.

From this score we can say that we have 46 sample of threes and 0 of them are false for 0, 6 of them false for 1, 10 of them are false for 2 and 30 of them are true for 3.

4. Codes Description

Main Function:

Below you can see my main Teensy codes, I will explain my functions.

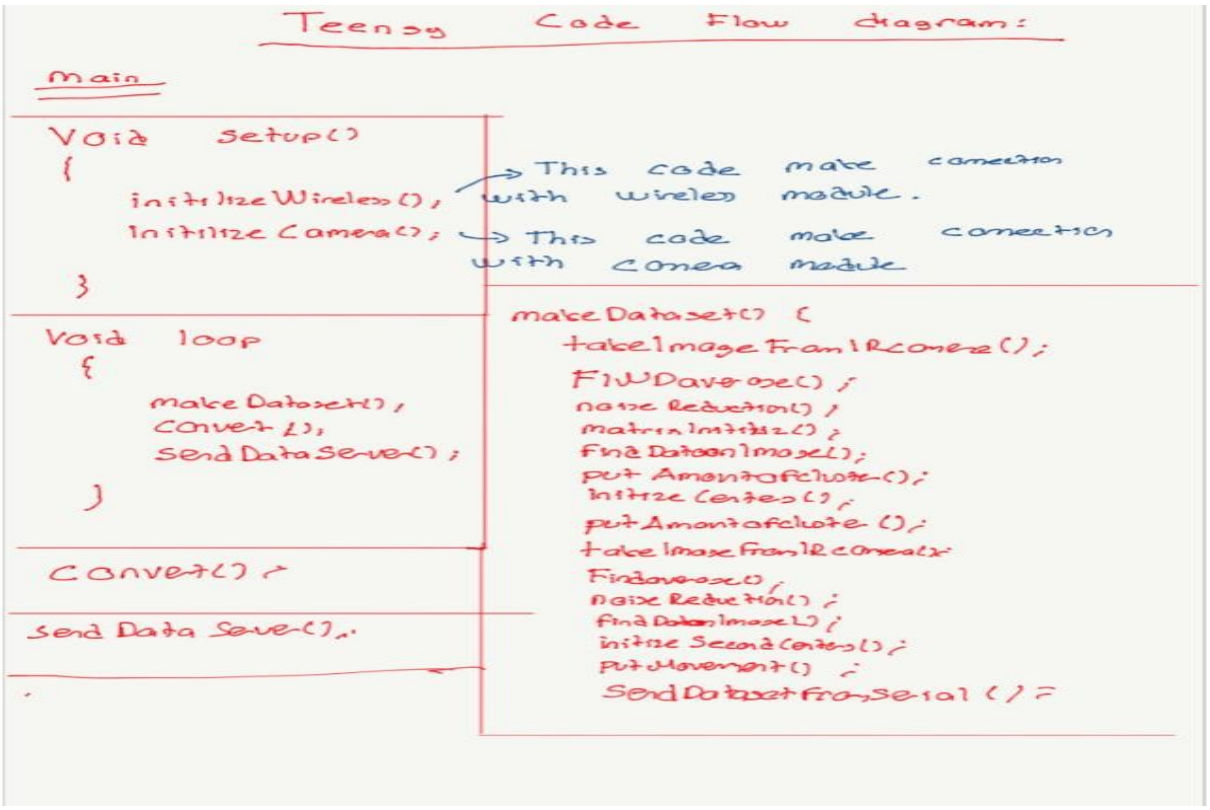


Figure 21: Coding Part VIII

Void Setup Codes (Running only one time):

initilizeWireless(): Aim of this code is initializing connection between wireless module and microcontroller and if it is not successful, it is going to inform us.

initilizeCamera(): Aim of this code is initilializing connection between IR Camera module and microcontroller and if it is not successful, it is going to inform us.

Void Loop (They run in infinitive loops):

makeDataset: This code is about making one feature set from IR Image

Working principle of makeDataset:

In this function, we have a lot of sub function. Their main aim is that producing one feature set like: Amount of cluster-Size of first cluster-Size of second cluster- Size of third cluster-Movement of first cluster-Movement of second cluster-movement of third cluster

takeImageFromIrCamera: This code took IR Image from IR camera and put variable named as Image.

FindAverage() : This code is problematic system in my code. Because of this code, my system needs to work in empty area first. As you know, our filter has main principle about averaging and averaging we need to collect a lot of example and then we can take average. Every loop it saves the average of system.

noiseRecution(): This function subtract IR Image from Average IR Image and then It apply 3 degree threshold to detect human.

matrixInitiliz(): This function initialize 0 valued matrix for next steps.

findDatasonImage(): This function search for cluster on image and save them in array, which initialized in matrixInıtlize() function.

putAmountofcluster(): This function count the amount of clusters and save them in array, which initialized in matrixInıtlize() function.

putMassofcluster(): This function calculate the size of cluster and save them in array, which initialized in matrixInılize() function.

initilizeCenters(): This function initialize the center of clusters from first IR Image.

initilizeSecondCenters(): This function initialize the second center of clusters from second IR Image.

putMovement(): This code calculate the movement and initialize it an array ,which initialized in matrixInitilize() function.

sendDataFromSerial(): This function just send feature, which founded in operation to serial port just for checking.

Convert():

We found makeDataset 30 features cluster set first 10 lines to show how many clusters we found, second 10 lines to show how big these clusters are and last 10 is for their movement but It is really complicated to apply these features Machine Learning because of this reason, we needed to decrease the number of feature and we wrote this code this produce new dataset. It produces

one feature set like: Number of cluster- How big first cluster- how big second cluster – how big third cluster- first movement- second movement- third movement.

SendDataServer():

This code send data to server via wireless module.

MLX90640 and MLX90640_I2C_DRIVER code:

This code is about I2C communication and they are prewritten.

Arduino_secrets:

This code is for Wifi name, Wifi password and also SERVER name but if you want to change the server, where you save data you need to also change SendDataServer() code. You need to change server name also there.

TimeandAmountSaver.py:

This code save the amount of people, from keyboard entered, and also time in text file(“information.txt”).

ConstructDataset.py:

This code construct data set first it read data from url http://193.196.175.153/es_lebt_out.php and also read the time from f = open ("information.txt","r") and construct new data set from then save it as success.csv but when it is doing that, it eliminate some data from server.

Allmodel.py:

This code first read data and shuffle it after that apply StandardScaler after that it produce result of KNN, Linear Regression, decision tree score and big neural network score. It uses sklearn and also tensorflow libraries.

NeuralLast.py:

This code applies our data one basic neural network.

NeuralNetworkbetter.py:

This code finds the best size for neural network (3 layers).

5. Conclusion

In this project, we had about %76 success to detect how many people in room. Because of lack of time, we could not collect more data. The system can be better with more complicated algorithm machine learning algorithm and This system can be trained better with more data If you have any question, please do not hesitate to me ask me.

Batu Kaan Özen
e-mail: ozenbatukaan@gmail.com
mobile phone: +90 531 742 68 18