Gebze Technical University
Department of Computer Engineering

CSE 396 - FINAL REPORT

GROUP 2

Group Members:
Ahmed Semih ÖZMEKİK
Ahmet Melih YANALAK
Alparslan KARAGÜNEY
Bengi YÖRÜKOĞLU
Berkay BAKIŞOĞLU
Değer MANDAL
Feyza Nur AKYOL
Mehmet Miraç ATICIOĞLU
Muhammed ÖZKAN
Yusuf PATOĞLU

Advisor:
Prof. Dr. Erkan ZERGEROĞLU

31-08-2020

Contents

1 Definition Of The Project

The aim of our project is to bounce a colored ball left on a transparent tray by using image recognition algorithms by keeping it balanced on the table. In addition to this, there are 3 additional features in our project, the first of which is to keep the ball in balance in the middle of the table without bouncing, secondly, to move the ball by following a square route on the table and the third and lastly, the cross movement by passing the ball through the cross balance points. All these processes are shown in real time both on the mobile and desktop application in 2D, besides, the height of the ball from the table is calculated during the bounce stage.

2 Requirements

In order to run Ball Bouncer without a problem you need either Linux or Windows OS which has at least two available USB ports, minimum 4th generation i5 processor and the computer should have installed and able to run the OpenCV 4.x.x versions without a problem. Also in order to run the mobile application you need a smart phone which has minimum Android 9(pie) OS.

3 Connecting The System To Computer

You need to connect two USB ports from system to computer in order to install required drivers automatically. Serial port communication addresses may be required in the future so setting them up will be helpful to us in first stage. For Windows you can set it through the task manager and for Linux you can set it with sudo setserial -g /dev/ttyUSB[0123] command. The last cable that needs to be connected to electrical outlet which provides the power of system with an adapter.

4 Setting Up The Software

You need to open the "ballbouncer.cpp" source code with any text editor and find that code piece:

```
#ifdefined ( WIN32 ) | | defined ( WIN64 )
#include
#define SERIALPORT "COM4"
#endif
#ifdef linux
#include
#define SERIAL PORT "/dev/ttyUSB0"
#endif
#define CAMERAID 0 // CameraID
int mode = BOUNCER;
```

In this section, depending on the type of OS, the port communication numbers that we set up in earlier stages needs to be replaced in the code. If there are only one camera is connected to system then the CAMERA ID preprocessor macro needs to be set up to zero. If there are another camera is connected we may need to change that macro as one. Lastly, the system will run as default ball bouncing mode. If desired the "mode" integer variable can be changed like: NORMAL BOUNCER, SQUARE CROSS macros. You can also do this when runnning the program with command line parameters as normal, bouncer, cross, square etc. In Linux systems you can build the program with executing make command. Before running program, for Linux systems you need a root privilege with "sudo su" because we'll be reaching to system ports. After getting root privilege you can run the program as ./ballbouncer and drop the orange ping pong ball into the system. In order to build the system in Windows you can use Visual Studio. You can either create new project or import the project that already available in files. Also before building the project you need to set those settings in this link: https://www.deciphertechnic.com/install-opencv-with-visual-studio/ After that the project will be built without any problems.

5 Installing The Mobile App And Connecting To The System

You need to copy the file named with app-release.apk to an Android device. In order to complete installation you need to follow unsigned application installation stages. In the next stage you need to switch off the cellular data of device and connect BALLBOUNCER network wih Wi-Fi. For password you need to enter “12345678” and connect the Access point on the system. After that you can open the app “mobile app” that we’ve installed. When the system starts up, you will be able to follow the x, y and h informations of the ball in real time through this application

6 Settings That May Need To Be Made In the System(Important)

PID values in pidServoX config and pidServoY config files may need to be set up in its context according to processing power for per second. Also when the system is running depending on the light situation you may need to set HighH LowS and LowW variables on Thresholded Image screen. By doing this you will need to find a specific setting that ball itself will be completely white and the other parts will be black.

7 Modules

7.1 Mechanical Design and Implementation

Before transferring the progress and design decisions of the module’s internal studies, you can reach the demo video, which briefly shows the chronological differences and stages in the development process, from the following link: https://youtu.be/fZuPE84c7dk the main structure of our project, a camera capturing a 640x360 330fps image, an ESP8266 based microprocessor, 4 MG996R servo motors and 4 DC-DC converters were used as hardware. The embedded software part is written in C ++. On the computer side, programming was written on C ++ and OPENCV library was used for image recognition. Our system works as follows. The image captured by the camera is firstly detected by the Moment structure of Opencv, and this information is sent to the PID algorithm together with the balance point information required for the ball. As a result of the calculation made here, the error rate to the equilibrium point is calculated. According to the result of this calculation, the information about how many degrees the motors will go to the ESP8266 device is sent via the serial port. The image is captured again and again, the error is calculated. These steps want to keep the margin of error to a minimum. The higher the margin of error of the system, the greater the angle information the system sends to the motors to correct this error. It is possible to adjust the Kp Ki Kd settings of this system from within the project files so that the x and y axis are independent. The mobile connection part, on the other hand, is made available to mobile devices thanks to the WiFi module that comes integrated into ESP8266.

To create the design, we printed 4 arms at first with a 3D printer. After the arms are printed, we wanted to use the servos that we have available, but it caused a lot of vibration because their power was insufficient although raised up to 6 volts also their gears were plastic. Thus, we ordered 4 servos with metal gears whose strength of the one is 10-12 kg/cm and a plexiglass.

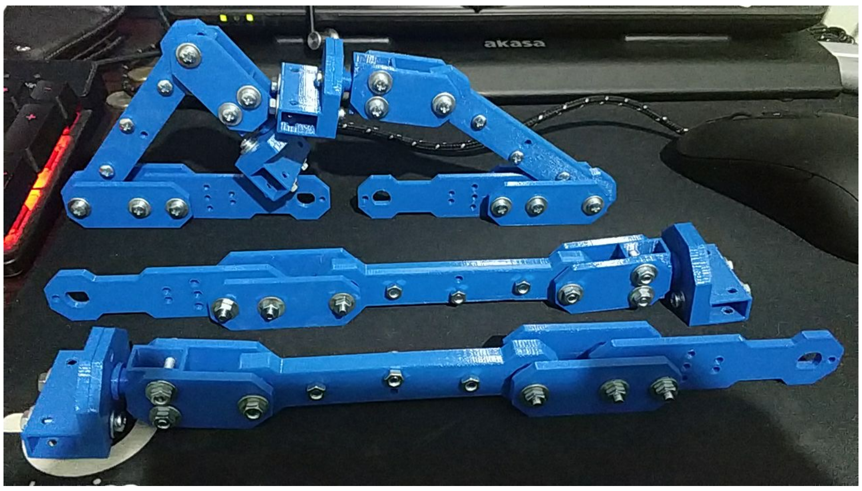


Figure 1.1: Design of the Arms

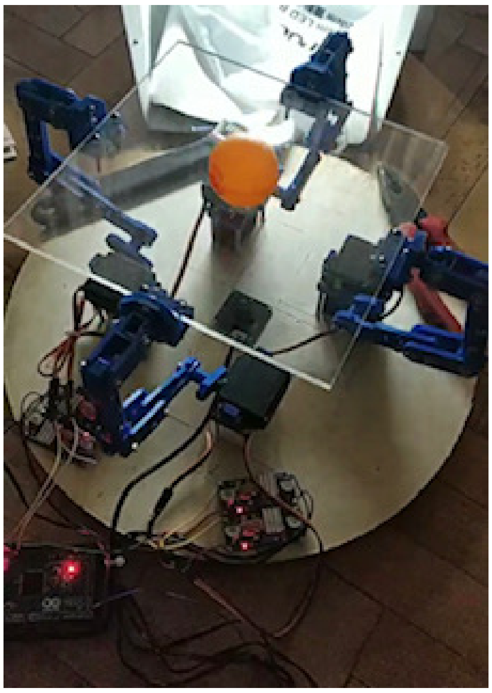


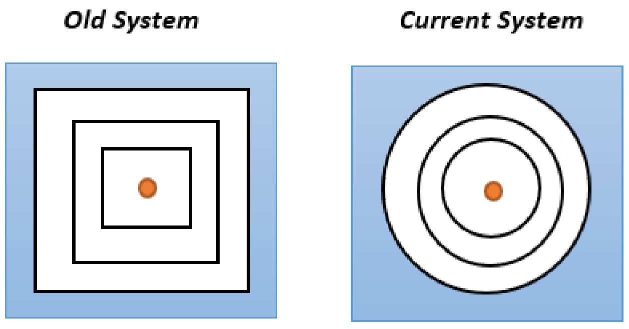
Figure 1.2: Design After Set Up

• How were the PID parameters determined?

It was done by following the steps on this link. `https://otomasyonadair.com/2014/11/27/pid-parametrelerinin-ayarlanmasi/`

• How does the ball bounce?

The first ball balancing process is actively used in this system. In addition, all arms are moved up and down at the same time in proportion to how close the ball is to the center, ie to the balance point. This system was controlled as a virtual square in the first version. However, this system caused the ball to fall from the sides in some cases. In the second version, this is controlled in a circle shape and provides a more stable control. With the classical circle formula, the distance of the ball to the equilibrium point is calculated and the bounce level is changed according to which circle it is in. $(x-a)^2 + (x-b)^2 = R^2$



7.2 NodeMCU v3 Wifi Development Module

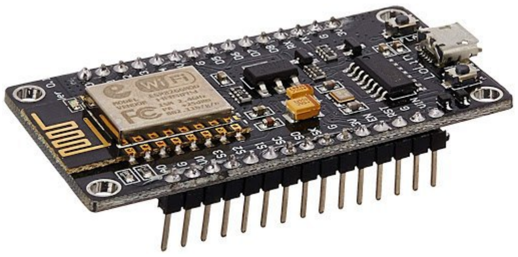


Figure 1.3: NodeMCU Module

We order NodeMCU in order to use to obtain angle values for the motors by calculating the height and coordinate information of the ball with the camera and to provide communication between the mobile and the hardware module. But we have not received the order yet, thus, we are using computer to calulate the required calculations temporarily. We are considering transferring data via wifi provided by NodeMCU.

• Which library was used for serial port communication?
Since our project will be cross platform, it should have a structure that can run on both Linux and Windows.For this, an open source library prepared by Philippe Lucidarme (University of Angers) was used.

7.3 Servo Motor

4 servo motors used to provide the movement. They are moving in a way to keep the ball balanced on the surface according to the specified coordinates with the camera.

According to the data it receives from NodeMCU, it gains an angle and applies force, thus, it gives the plate up and down movement.

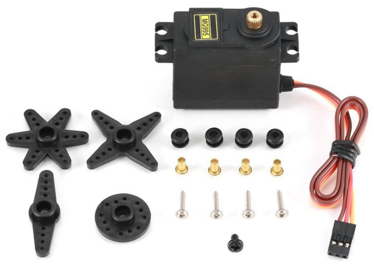


Figure 2.2: Servo Motor

7.4 3D Printed Pushrod

3D printed pushrods applying the motion from the motors to the plate, it gives the plate the desired motion and also keep the plate balance.

7.5 640x360 330 FPS Camera

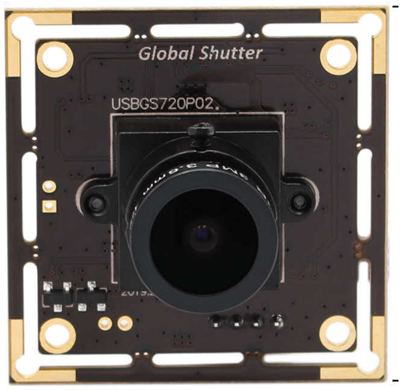
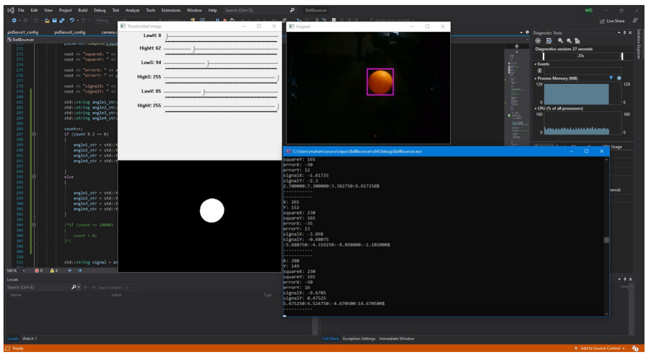


Figure 2.3: 640x360 330fps camera

What and how were the structures used in the project made?

- Why has the 640x360 image taken from the camera been reduced to 490x335? There are two reasons for this. The camera lens used on the system was replaced with another one because the angle of its original lens was too narrow for the system. Therefore, blank black areas appeared on the edges, as the new lens attached offers an angle close to fish eye and is not compatible with the camera's sensor. Since these blank black areas increase the matrix size for unnecessary operations in image recognition at runtime, unnecessary parts for us are cropped.

8 Desktop Application and Visual Recognition



LowH: 0
HighH: 62
LowS: 94
HighS: 255
LowV: 85
HighV: 255

C:\Users\muham\source\repos\BallBouncer\x64\Debug\BallBouncer.exe

```
squareY: 165
errorX: -30
errorY: 12
signalX: -1.61725
signalY: -2.3
2.700000:7.300000:3.382750:6.617250$
-----------
-----------
X: 265
Y: 152
squareX: 230
squareY: 165
errorX: -35
errorY: 13
signalX: -3.898
signalY: -0.68075
-5.680750:-4.319250:-8.898000:-1.102000$
-----------
-----------
X: 280
Y: 149
squareX: 230
squareY: 165
errorX: -50
errorY: 16
signalX: -9.6705
signalY: 0.47525
5.475250:4.524750:-4.670500:14.670500$
-----------
```

Figure 2.5: Screenshot of Ball Recognation Program

In the project, it was necessary to visually identify and position the object on the plate. For this, we used the OpenCV library in C++. In the process of determining the position of the object, we knew that some operations would be applied to its image matrix (such as converting color, blurring, eroding or dilating), and consequently this could hinder the real-time process of the system.

Therefore, in the image processing phase, we needed an algorithm that was optimized as much as possible. And by trying a few algorithms in C++, we started some tests. First of all, we used the Hough transform method, which is used in many fields such as image analysis, computer vision, and digital image processing to detect a single object. Then, we tried these codes on our system where pid operations are not ready yet, but still at the appropriate stage for performing visual recognition tests.

As a result, when we decided that this method caused performance problems in the system, we looked for different algorithms and methods. And we wrote our final implementation with Image Moments. It showed good results in tests.

We provided an input to a PID function with the coordinates we obtained with the image recognition process and the balancing points we determined earlier. We obtained the necessary slope on the surface by turning the motors in the system to their required positions according to the error rates we obtained here. During PID training, we applied the following methods.

We use to tune a PID the following steps:

1. Set all gains to zero.
2. Increase the P gain until the response to a disturbance is steady oscillation.
3. Increase the D gain until the the oscillations go away (i.e. it's critically damped).
4. Repeat steps 2 and 3 until increasing the D gain does not stop the oscillations.
5. Set P and D to the last stable values.

6. Increase the I gain until it brings you to the set point with the number of oscillations desired.

- Why are Moments used instead of Hough Circle Transform

The Hough Circle Transform system operates with sufficient performance when the matis size is small and there is only one round object on the screen, but the performance of the system drops significantly when other round objects enter the screen. In addition, the image of the ball falling on the camera during the bounce stage may lose its round structure due to light reflections, and this caused problems in the recognition process. Moments structure works on recognizing moving objects, not round objects in the image. And it gave a faster and more stable result than the Hough Circle Transform system. We can easily reach the area of the object defined in the moment structure.

- How is the height of the ball from the table surface calculated?

As mentioned before, we can reach the object area defined in the Moments structure. From this point of view, while the object is on the table, the object area is at the maximum value, for example height is 0. As the ball rises from the table, its area gets smaller. On the basis of this information, after finding the radius of the ball from the field, we were able to approximate the height information using an inverse ratio.

9 Mobile Application

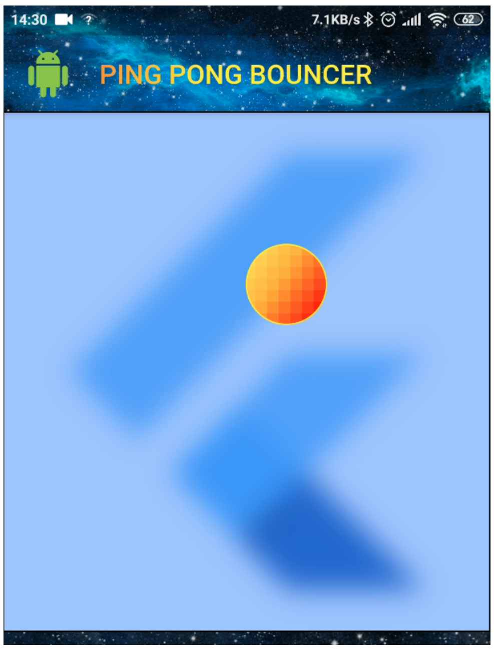


Figure 2.5: Flutter Icon

In order to create a mobile application which will allow the user to watch the current position of ball simultaneously and get the coordinates information,Flutter is used on Android Studio platform.Flutter is an open-source UI software development kit created by Google.
So far,random coordinates are generated each second to simulate the app without getting in touch with the actual hardware.The mobile app is kept as simple as possible in order to integrate new updates.The next goal is to integrate the mobile application with hardware and show the identical position of the ball.

10 Test and Simulation

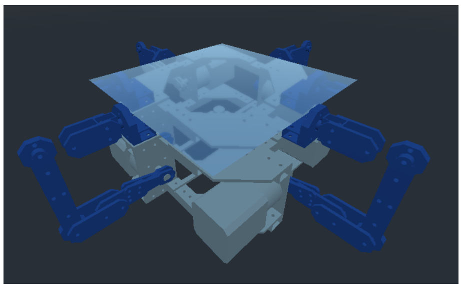


Figure 2.6: 3D Model

We made a model of the mechanical system to use in Godot Engine. The Engine is using its own physics system right now. We plan to use the real time coordinates of the ball to simulate the mechanism for the final project.

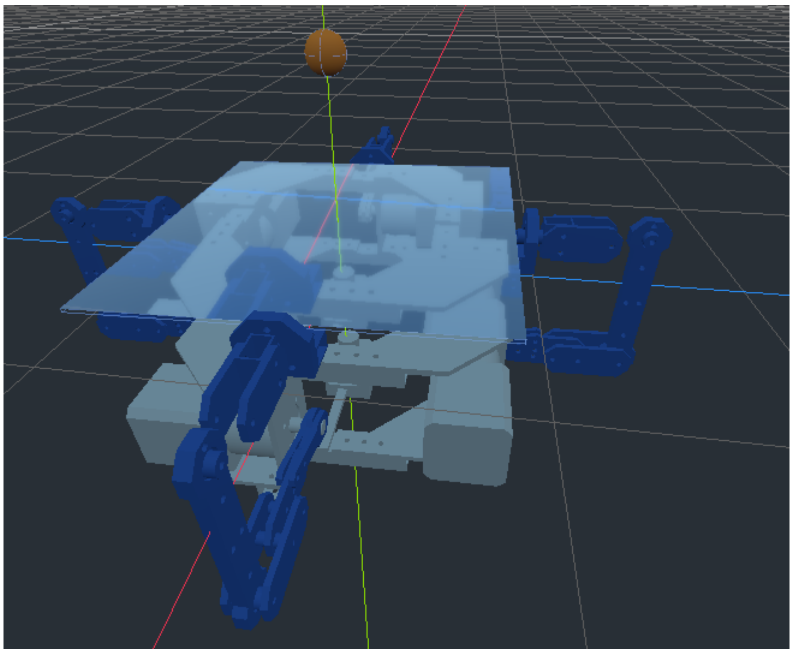


Figure 2.7: Godot model with environment textures.

You can reach the video, which briefly represents how the simulation is worked out, from the link below: `https://youtu.be/HV61CvCdYRc`

11 Workflow Diagram

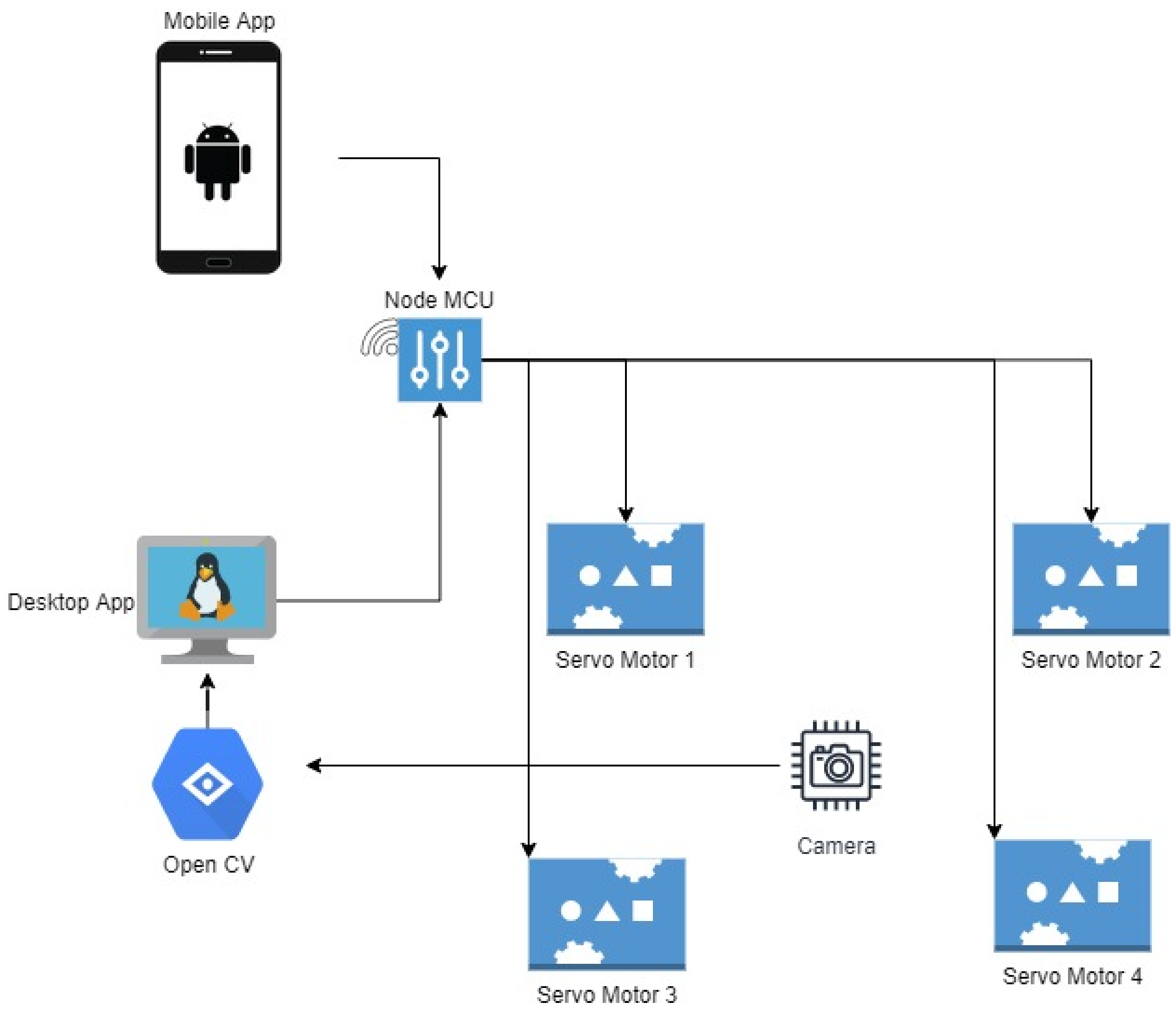


Figure 1: Flow of the General System

12 Project Budget

1. Screws 25 tl
2. Plexiglass 20 tl
3. 4 Motor 130 tl
4. 4 Voltage Regulator 50 tl
5. Node MCU 50 tl
6. Adaptor 50 tl
7. Camera 350 tl
8. 3d Printing Filament 50 tl

TOTAL 725 tl

13 Videos Of The Project

Bouncer Mode https://www.youtube.com/watch?v=rXE_N2s3X5s
Cross Mode https://www.youtube.com/watch?v=VVtiZge8e-Y
Normal Mode https://www.youtube.com/watch?v=BaT8Sq6ADKo
Square Mode https://www.youtube.com/watch?v=NPMFVuwvdVY
Video of simulation: https://youtu.be/HV61CvCdYRc
Video of hardware part:https://www.youtube.com/watch?v=fZuPE84c7dk&feature=youtu.be
Video of ball recognation part: https://www.youtube.com/watch?v=38nhb_p1s1Q
Dektop GUI: https://www.youtube.com/watch?v=b2EuwV7PlsM

14 Resources

https://answers.opencv.org/question/46755/first-example-code-error/
https://forum.qt.io/topic/92848/opencv-error-undefined-reference-to-cv-videocapture-vid
https://linuxize.com/post/how-to-install-opencv-on-ubuntu-20-04/#installing-opencv-from
https://github.com/T-Kuhn/HighPrecisionStepperJuggler https://electrondust.com/2020/03/01/the-octo-bouncer/

15 Work sharing

Names	Modules
Ahmed Semih Özmekik	Mechanical Design And Implement Desktop Application and Visual Recognition
Ahmet Melih Yanalak	Mobile Application Mechanical Design And Implement
Alparslan Karagüney	Mechanical Design And Implement Test and Simulation
Bengi Yörükoğlu	Mobile Application Mechanical Design And Implement
Berkay Bakışoğlu	Mechanical Design And Implement Test And Simulation
Değer Mandal	Desktop Application and Visual Recognition Test And Simulation
Feyza Nur Akyol	Mobile Application Desktop Application and Visual Recognition
Mehmet Miraç Atıcıoğlu	Mobile Application Test And Simulation
Muhammed Özkan	Mechanical Design And Implement Desktop Application And Visual Recognition
Yusuf Patoğlu	Mobile Application Desktop Application And Visual Recognition