

GNG1103
Design Project User and Product Manual

PBATS
Padel Ball Analysis and Tracking System

Submitted by:

PBATS Group 10

Maria van Scherrenburg, 300297943

Nwike Osiagwu, 300241642

Vlad Romanenko, 300282038

December 7th, 2022

University of Ottawa

Table of Contents

Table of Contents	ii
List of Figures	v
List of Tables	vii
List of Acronyms and Glossary	viii
Credits	ix
1 Introduction.....	1
2 Overview.....	2
2.1 Conventions.....	4
2.2 Cautions & Warnings	4
3 Getting Started	5
3.1 Configuration Considerations	5
3.2 User Access Considerations	5
3.3 Accessing/Setting-up the System	5
3.3.1 Files.....	5
3.3.2 Python	5
3.3.3 Libraries	7
3.3.4 Visual Studio Code	8
3.3.5 Unity	9
3.3.6 Phone Access	10
3.4 System Organization & Navigation	11
3.4.1 Python Code.....	12

3.4.2	Camera Access	12
3.4.3	Unity	12
3.4.4	Excel	12
3.5	Exiting the System	12
4	Using the System	13
4.1	Code Inputs	13
4.1.1	Camera and Video.....	13
4.1.2	Ball Colour.....	14
4.1.3	Excel	16
4.2	Unity.....	16
5	Troubleshooting & Support	17
5.1	Error Messages or Behaviors	18
5.1.1	Code Errors	18
5.1.2	Unity Errors	18
5.2	Maintenance	19
5.3	Support	19
6	Product Documentation	20
6.1	Coding	20
6.1.1	BOM (Bill of Materials) and Equipment List.....	20
6.1.2	Instructions.....	20
6.2	Unity.....	20
6.2.1	BOM (Bill of Materials) and Equipment List.....	20
6.2.2	Instructions.....	20

6.3	Complete Bill of Materials	21
6.4	Testing & Validation	21
7	Conclusions and Recommendations for Future Work	23
8	Bibliography	24
	APPENDICES	25
9	APPENDIX I: Design Files	25

List of Figures

Figure 1: Excel Outputs for the PBATS System	2
Figure 2: Unity Output for the PBATS System	3
Figure 3: Code Output for the PBATS System.....	3
Figure 4: Block Diagram of the PBATS System	4
Figure 5: Python System Download	6
Figure 6: Confirming Python is Installed.....	6
Figure 7: Edit Environment Variables	7
Figure 8: Edit Environment Variables (Cont.).....	8
Figure 9: Visual Studio Code Python Extension	9
Figure 10: Python Option Select Visual Studio Code	9
Figure 11: Installing Unity in the Unity Hub.....	10
Figure 12: Adding Build Support in Unity	10
Figure 13: EpocCam Download Selection.....	11
Figure 14: Video Input Options for the Python Code.....	13
Figure 15: Video File Scale Options for the Python Code	13
Figure 16: Ball Colour Sliders (Unselected Colour).....	14
Figure 17: Ball Colour Sliders (Colour Selected).....	14
Figure 18: Colour Value Inputs for the Python Code	15
Figure 19: Code Outputs With Colour Selected	15
Figure 20: Selecting Number of Unwanted Pixels	16
Figure 21: Number of Pixels Output to the Terminal	16

Figure 22: Excel Inputs for the Python Code..... 16

Figure 23: Zip File Containing Unity Environment 17

Figure 24: Unity Environment for Design Day 17

Figure 25: Converting Indentation to Spaces/Tabs..... 18

List of Tables

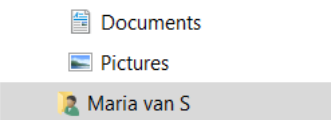
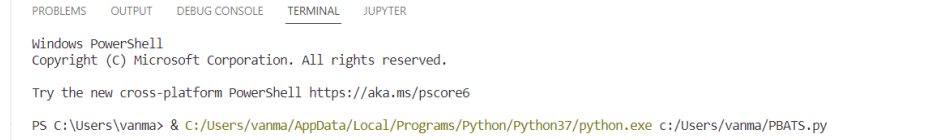
Table 1. Acronyms.....	viii
Table 2. Glossary	viii
Table 3: Support Links for Various PBATS Components.....	19
Table 4: Complete Bill of Materials for Final Solution.....	21
Table 5: Ball Tracking Software Functionality Test Results and Comments.....	21
Table 6. Referenced Documents.....	25

List of Acronyms and Glossary

Table 1. Acronyms

No acronyms were used in this document.

Table 2. Glossary

Term	Acronym	Definition
User Profile	N/A	In the computer files, it is the section that contains all the user's files. It appears as the following icon: 
Terminal	N/A	The code's outputted values appear in this window. This is found in the PBATS.py file and looks like the following: 

Credits

We would like to thank the following for their indirect help in contributing to the code mentioned in this document. Firstly, we would like to thank Murtaza's Workshop on youtube for his work on the implementation of the yolo object tracking system [1]. Secondly, we would like to thank the creator of the yolo object tracking algorithm, Joseph Redmond [2]. Thirdly, we would like to thank asi55 on stackoverflow for his help with the colour detection system [3]. Lastly, we would like to thank the TA's who helped create the colour contour code that was used. It couldn't have been done without the help of all those mentioned above.

1 Introduction

This user manual is designed for the client or owner of this software to be able to become fully educated on the design, usage and capabilities of the product. An in depth overview of each of the components is provided along with detailed documentation of how each component was developed, tested and implemented in the final product the user owns now.

This manual assumes that the reader is at least moderately well versed on coding using Python and C Sharp, as well interactions between these languages and the 3D environment software Unity. In depth explanations of the source code is provided for those interested, however more general descriptions are given in the documentation for those simply interested in operation and basic understanding.

Methods for fixing errors through troubleshooting are also listed as well as recommendations for how the product was intended to be used.

This manual is organized as follows:

- Section 1: Introduction
- Section 2: Overview – incentive for creating such a product as well as general team philosophies on the development process
- Section 3: Getting Started – what to download to make sure the system works
- Section 4: Using the System – how to setup, run and configure the product to work in the most optimal way, as well as an in depth analysis of how the software operates
- Section 5: Troubleshooting & Support – how to solve errors and problems that may occur during operation of the software
- Section 6: Product Documentation – development history and rationale for specific aspects of the design
- Section 7: Conclusions and Recommendations for Future Work – possible areas of development and conclusions
- Section 8: Bibliography

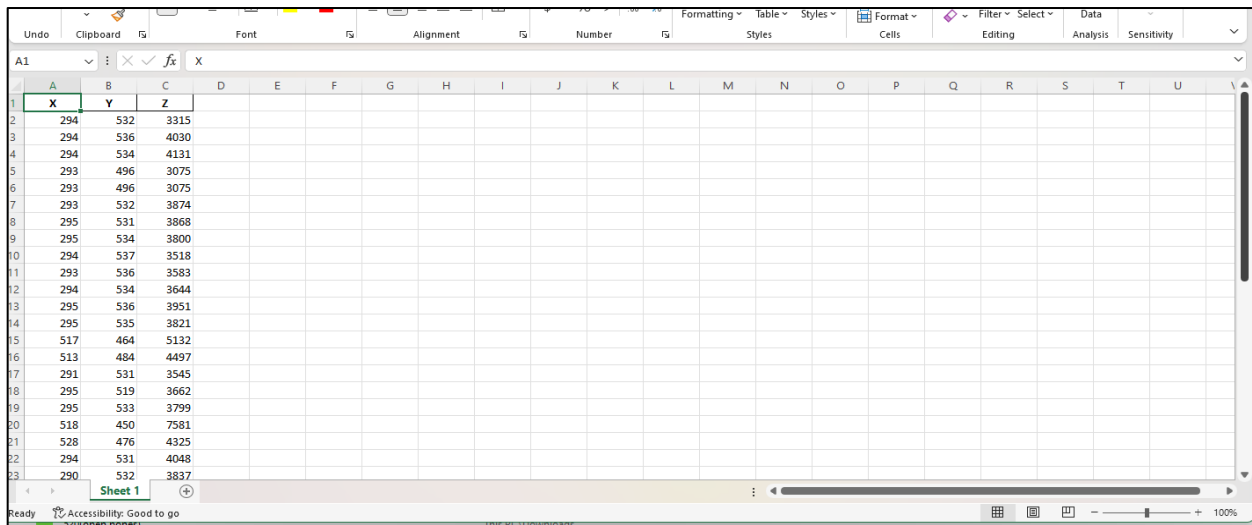
Design files and other software download links can be found at the end of the document in APPENDIX I: Design Files.

2 Overview

As the sport of Padel ball grows in Canada, a need exists for a system that can track and analyze the kinematic properties of a Padel ball for athletes to improve their performance. This solution must use cameras to track the ball into a 3D environment.

After extensive meetings with and questioning of the client, the fundamental needs of the user were identified and specified as ball-tracking with multiple cameras, trackable position and kinematic properties of the ball, storage of all information tracked, and a high-quality 3 dimensional environment.

The PBAT system's defining quality is its ability to track the ball. The ball recognition system uses a three-step process to verify the location of the ball. It uses color recognition, shape recognition, and object recognition in order to most accurately determine the position of the ball.



	X	Y	Z
2	294	532	3315
3	294	536	4030
4	294	534	4131
5	293	496	3075
6	293	496	3075
7	293	532	3874
8	295	531	3868
9	295	534	3800
10	294	537	3518
11	293	536	3583
12	294	534	3644
13	295	536	3951
14	295	535	3821
15	517	464	5132
16	513	484	4497
17	291	531	3545
18	295	519	3662
19	295	533	3799
20	518	450	7581
21	528	476	4325
22	294	531	4048
23	290	532	3837

Figure 1: Excel Outputs for the PBATS System

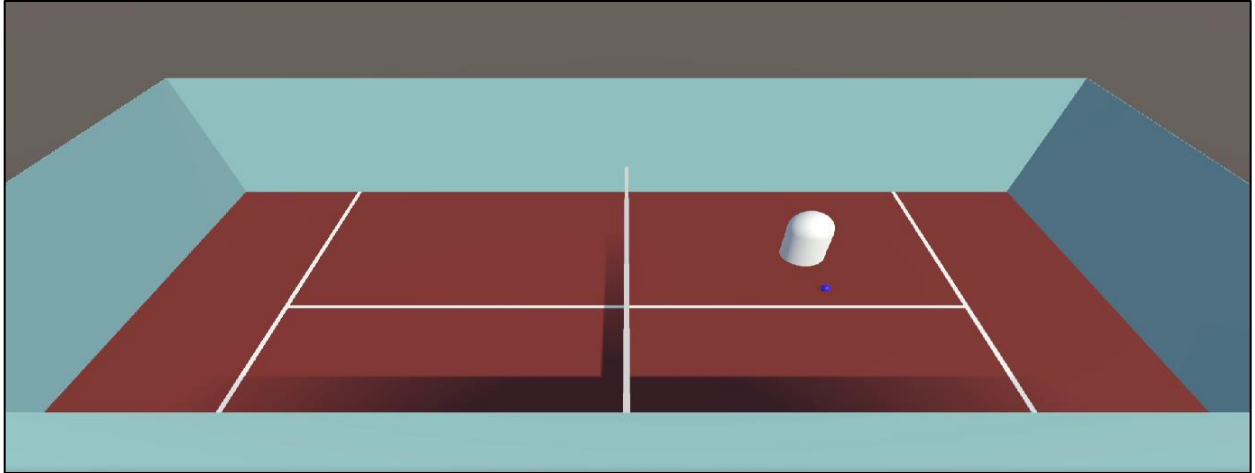


Figure 2: Unity Output for the PBATS System

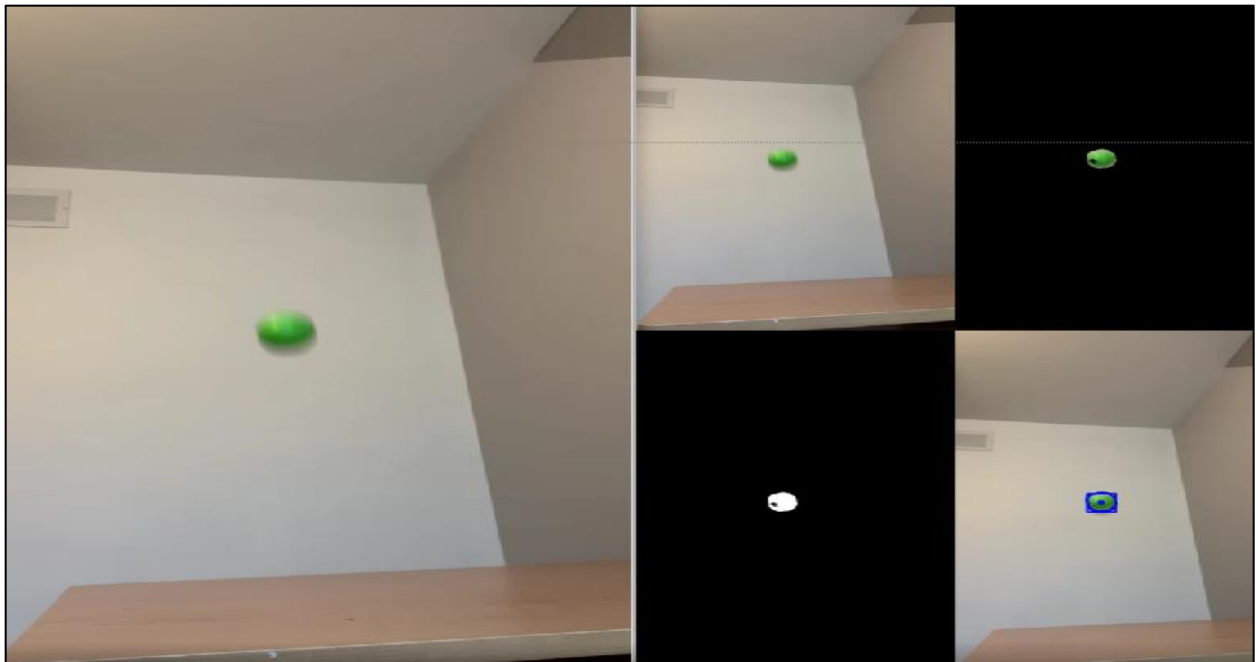


Figure 3: Code Output for the PBATS System

The foregoing images display the three essential systems of the PBATS, the data storage system, the VR environment, and the ball tracking system. These systems all work independently of each other. The following is a block diagram showing the basic concept of our final system:

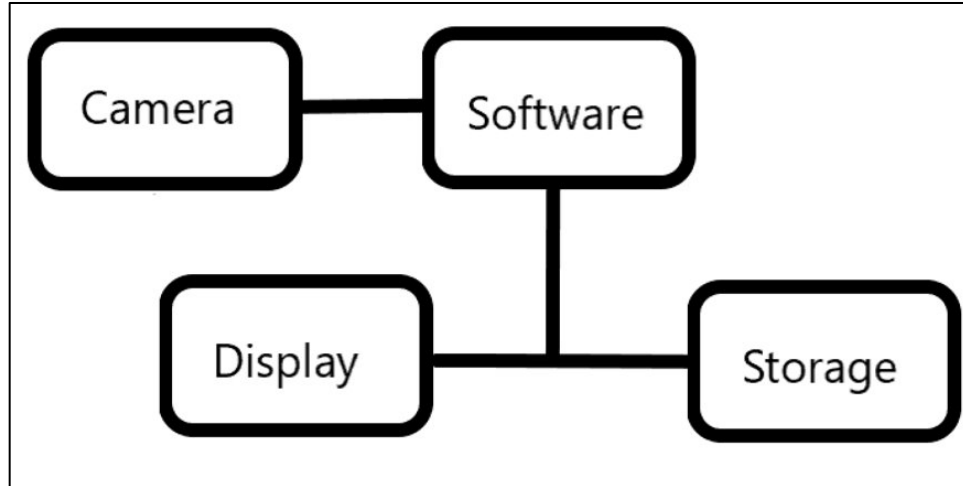


Figure 4: Block Diagram of the PBATS System

The system is a virtual tracking system that only requires a working camera and a mid-high computer. All that is required for set up is proper camera positioning, and multiple cameras can be added for increased accuracy.

2.1 Conventions

This manual is put in the most simple terms possible, all instructions are put into layman's terms. There are no specific conventions, nor special syntax found in this text.

2.2 Cautions & Warnings

The system requires a decent amount of processing power to run with one camera. The average laptop can run the system with one camera very smoothly; however, as cameras get added the amount of processing power required increases greatly. As cameras get added, be weary of the increasing load on the device running the system.

3 Getting Started

To get started, a few programs must first be downloaded. These programs include: Python, Visual Studio Code, Unity, and either EpocCam or DroidCam, depending on the type of cellphone you own. Links can be found in APPENDIX I: Design Files, but are also mentioned in the instructions listed below. This section describes how to download all the necessary files and applications to allow the system to run. The next section titled Using the System describes how to run the code, phone access, and Unity programs.

3.1 Configuration Considerations

To configure the PBATS system, files and programs will first need to be downloaded. The process to do so and the list of files can be found in later sections titled Accessing/Setting-up the System and Using the System. The inputs for the system include a camera, and outputs the ball's coordinates and data into the python code, the 3D Unity Environment, and an Excel spreadsheet.

3.2 User Access Considerations

The intended users for this system are Padel Ball players and coaches, however this can be applied to other sports where tracking and ball analysis could be used. Restrictions for this are the smartphones or computer's owned by the users. If the phone cannot reliably be connected or the user does not own a laptop that can be used on site, then the footage can be recorded to be inputted into a computer later.

3.3 Accessing/Setting-up the System

3.3.1 Files

Download the necessary files from APPENDIX I: Design Files, including:

- PBATS.py
- Yolov3.cfg
- Yolov3.weights
- Ball Recognition v5.zip

Make sure that these files are located in the User Profile section of your computer's files. The location of your User Profile can be found in the Glossary.

3.3.2 Python

Firstly, download python version 3.7.0 at the following link: <https://www.python.org/downloads/release/python-370/>. To find the correct version for your computer, scroll down to the section labeled *Files*. For Windows computers, select the option that includes "executable installer".

Files					
Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		41b6595deb4147a1ed517a7d9a580271	22745726	SIG
XZ compressed source tarball	Source release		eb8c2a6b1447d50813c02714af4681f3	16922100	SIG
macOS 64-bit/32-bit installer	macOS	for Mac OS X 10.6 and later	ca3eb84092d0ff6d02e42f63a734338e	34274481	SIG
macOS 64-bit installer	macOS	for OS X 10.9 and later	ae0717a02efea3b0eb34aad680dc498	27651276	SIG
Windows help file	Windows		46562af86c2049dd0cc7680348180dca	8547689	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	cb8b4f0d979a36258f73ed541def10a5	6946082	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	531c3fc821ce0a4107b6d2c6a129be3e	26262280	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	3cfdaf4c8d3b0475aaec12ba402d04d2	1327160	SIG
Windows x86 embeddable zip file	Windows		ed9a1c028c1e99f5323b9c20723d7d6f	6395982	SIG
Windows x86 executable installer	Windows		ebb6444c284c1447e902e87381afeff0	25506832	SIG
Windows x86 web-based installer	Windows		779c4085464eb3ee5b1a4fffd0eabca4	1298280	SIG

Figure 5: Python System Download

Follow the prompts from the python installer once it has been downloaded. When the installer appears, make sure to check the box labeled “Add to Path”. To ensure that python has correctly been installed, open the command prompt. On Windows computers, this can be done by clicking the windows button on the bottom left of the screen and typing “Command Prompt”. Once in the command prompt, type in “python” and press enter. A message like the one shown below should appear.

```

Command Prompt - python
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\vanma>python
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _

```

Figure 6: Confirming Python is Installed

To exit, type “exit()” and press enter. The command prompt should stay open. Do not X it out yet.

3.3.3 Libraries

For certain things in the python code to work, we need to download libraries. While still in the command prompt from the previous section, type in one of the following commands and press enter before proceeding to typing the next:

- pip install cvzone
- pip install numpy
- pip install opencv-python
- pip install opencv-contrib-python
- pip install pandas
- pip install openpyxl

It might ask for you to update the library. The command to type is shown in the command prompt and should look something like: “pip install –upgrade pip”.

If the libraries are giving **errors**, it might be that Python was not correctly added to the Path when installed. To manually do so, follow the steps outlined below:

- By clicking the windows button on the bottom left of the screen, type in: “Edit Environment Variables” and click the option that appears like the one below:
- Another screen will appear. Click the “Environment Variables” button at the bottom of the screen.
- When a new window appears, double click on PATH and another window should appear:

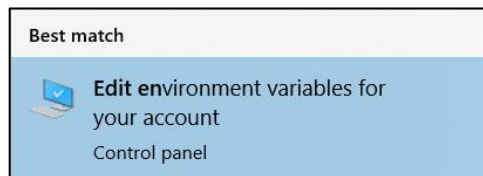


Figure 7: Edit Environment Variables

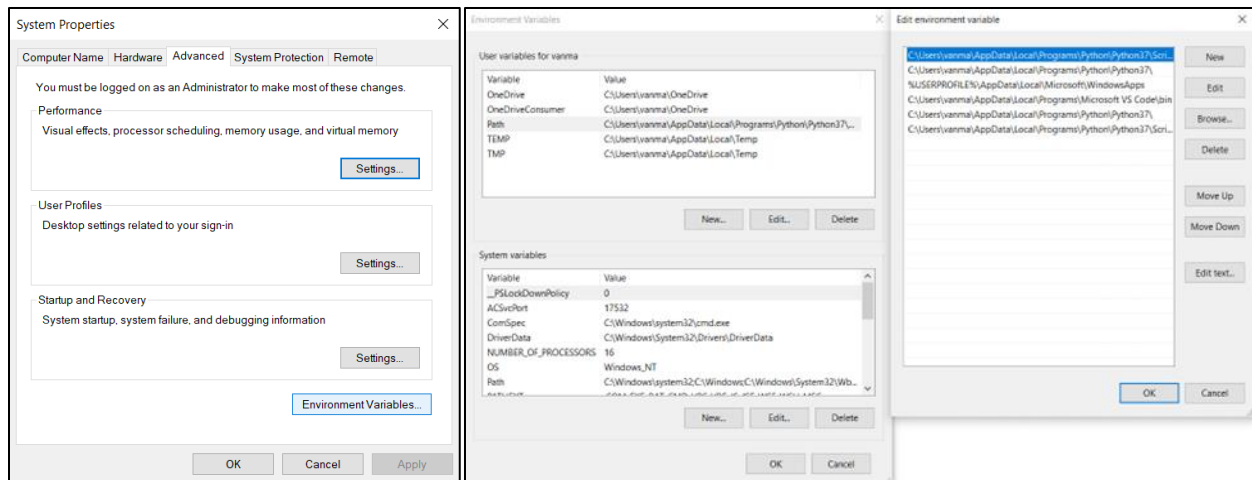


Figure 8: Edit Environment Variables (Cont.)

- If there are no environment variables that have the form *C:\Users\user_name\AppData\Local\Programs\Python\Python37* then a new path must be created. Press the “New” button on the right side of the screen. Then paste in the following, making sure to edit in your unique user profile name.
 - o C:\Users\user_name\AppData\Local\Programs\Python\Python37
 - o C:\Users\user_name\AppData\Local\Programs\Python\Python37\Scripts

Now that the path is added successfully, return to the Libraries **Error! Reference source not found.** section and try installing the libraries.

3.3.4 Visual Studio Code

Next, to download the system in which the code will run, go to the following link: <https://code.visualstudio.com/download> and download the appropriate version for your computer. Install like you would any software, by following the prompts. If it asks to add python to path, check the box to select YES. When Visual Studio Code opens, it is time to download the Python extension. Under the extensions tab on the left-hand side, type in “Python” and install.

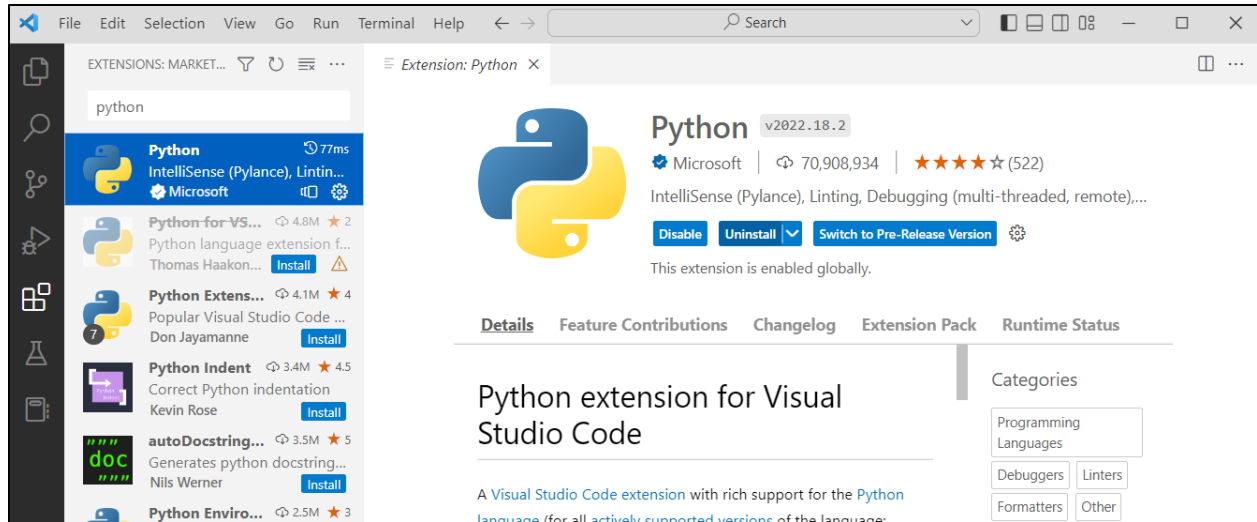


Figure 9: Visual Studio Code Python Extension

If the option appears, select “Select a Python Interpreter” and choose “Python 3.7.0”. If the option does not appear or the option to choose “Python 3.7.0” does not appear, then click File > New File > Python File or type in a name ending with .py, such as “test.py”. Once the new file is created, the option to select “Python 3.7.0” should show up in the bottom right of the screen.



Figure 10: Python Option Select Visual Studio Code

3.3.5 Unity

Finally, we will download Unity to access the 3D environment in which the ball will be shown. Go to the link: <https://unity.com/download> and download the appropriate version for your computer. Follow the download instructions on the Unity Hub Installer. Create a Unity ID if prompted. Now to download the actual Unity Software. In the Unity Hub, go to the Installs tab on the left side of the screen and download the latest version (Unity 2019.4.18f1 (LTS)).

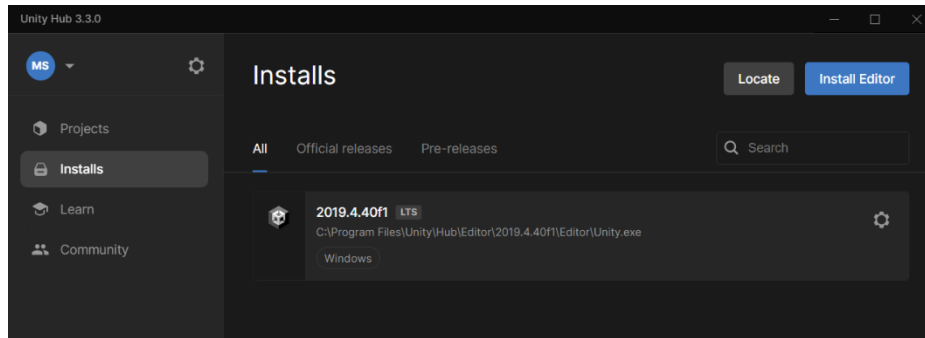


Figure 11: Installing Unity in the Unity Hub

When it prompts you to add modules to the install, be sure to select the module titled Microsoft Visual Studio Community 2019 and either Mac Build Support or Windows Build Support.

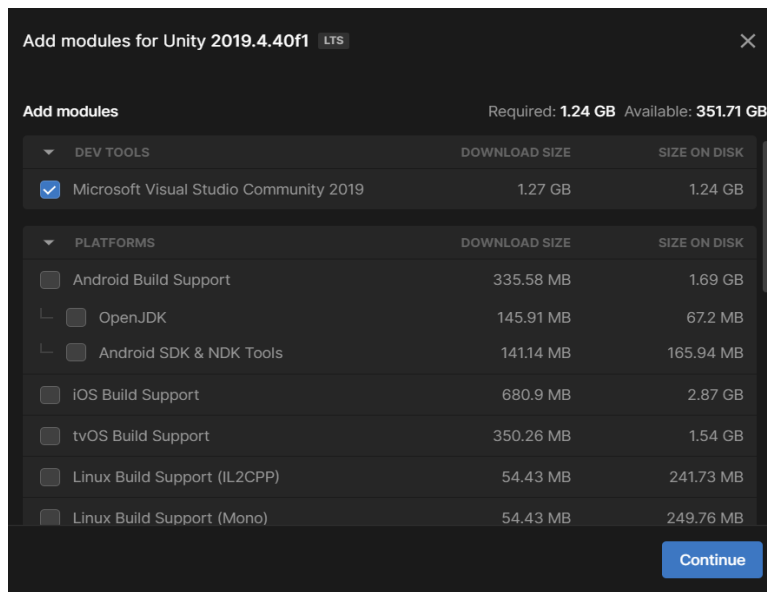


Figure 12: Adding Build Support in Unity

3.3.6 Phone Access

To allow access for your computer to link to an external phone camera, you must download two apps, one on your computer and one on your phone.

3.3.6.1 iPhone

For iPhone, download EpocCam from the App Store. When downloaded, follow the prompts. On your computer, download Elgato Camera Hub from the following link: <https://www.elgato.com/en/downloads>.

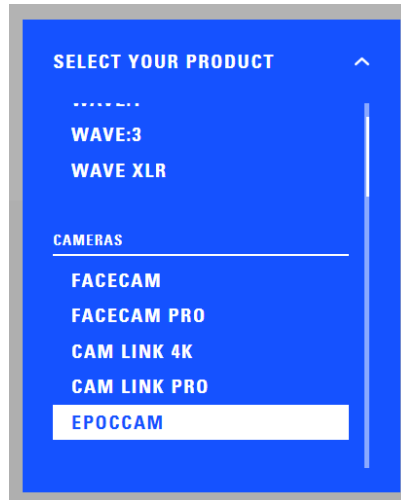


Figure 13: EpocCam Download Selection

Note that **iTunes must be downloaded** on your computer as well. When downloaded, open up both the app on your phone and on the computer and follow the instructions. Plugging in your phone into your computer via a charging cable yields the easiest and best results. Do not minimize out of the EpocCam app on your computer as the code will no longer register the camera input. Instead, just click on a new window and leave the EpocCam app run in the background.

3.3.6.2 Android

To connect your android phone to your laptop to be used as a camera input, then following the instructions at the provided link: <https://www.dev47apps.com/>. This link will instruct you how to set up the system. First begin by downloading the appropriate app on your windows or mac computer, then the respective app on your phone, as mentioned at the link provided.

3.4 System Organization & Navigation

The PBATS system revolves around the PBATS.py python code. When you open the python file, then it links to the Unity file, camera app, and to an excel file to store the data.

In this section, the steps on how to turn on the system is described. The next section titled Using the System describes in detail how to change the inputs needed for the PBATS system.

3.4.1 Python Code

To start the python code, open up Visual Studio Code. Once that has been opened, then navigate to the top left of the screen and choose Files > Open File and select the PBATS.py file from the User Profile section. Once the code has been downloaded, it can be run using the play button in the top right of the screen.

3.4.2 Camera Access

To start the camera connection to your computer, follow the instructions on the app that was provided in 25APPENDIX I: Design Files. Note that for EpocCam and iPhones, iTunes must also be downloaded on your computer. When the code is run and the camera is successfully connected, do not minimize the EpocCam or DroidCam app. Instead, just navigate to another window.

3.4.3 Unity

To access the Unity system, ensure that the zip file was extracted in the User Profile section of your files. In the extracted zip file, navigate to Lab 5 > Ball_Recognition_Isaac.exe. The Unity environment should appear. For windows computers, you can get out of the Unity environment by selecting Windows + Tab to change tabs.

3.4.4 Excel

If you have sent the data to Excel, then the Excel sheet should appear in your User Profile section. You can access the file by opening it as you would a regular excel file.

3.5 Exiting the System

To exit the system, all open tabs and windows can be X out. These include the PBATS.py file, the Unity file, the camera app, and the excel file if it was opened.

4 Using the System

The following sub-sections provide detailed, step-by-step instructions on how to use the various functions or features of the PBATS system. This includes inputs needed for the code and using the Unity environment. To operate the system, certain files must be downloaded and opened. As mentioned in Accessing/Setting-up the System, the files to download can be found in APPENDIX I: Design Files, which include: yolov3.cfg, yolov3.weights, PBATS.py, and Ball_Recognition.zip. The files to open include the PBATS.py file and the Ball_Recognition_Isaac.exe file from the Ball_Recognition.zip file, as found in 9: Appendix I.

4.1 Code Inputs

The python code that was downloaded requires a set of inputs that need to be adjusted based on individual scenarios. In essence, the code requires input for video, the colour of the ball that is to be tracked, if the data is to be sent to excel, and more advanced commands to optimize the system, such as finding the unwanted number of pixels in the frame.

4.1.1 Camera and Video

For the Camera portion, the code can get inputs from 'phone', 'computer' or an external file. In the line named *inputvideo*, the argument can be changed to either of the 3 video input options. If you want to **add an external file** such as a recording, then replace the *inputvideo* with an argument such as 'ball.mp4'. The files **MUST** be named in lowercase and **MUST** be located in the User Profile section where the code was downloaded and moved to.

```
1 #options for video input include: 'phone', 'computer', or a video file name such as 'ball.mp4'  
2 inputvideo = 'phone'
```

Figure 14: Video Input Options for the Python Code

For external files, they might have to be scaled. This can be played with in the next line by toggling the scale size in percentage. For a full scale, set scale = 100.

```
4 #when a file is uploaded, it might have to be scaled. input value to find appropriate scale  
5 #iphone files are about 20%  
6 scale = 100
```

Figure 15: Video File Scale Options for the Python Code

4.1.1.1 Camera Set-Up and Placement

The camera that will be used to record the ball's position can be set up in various ways. For a small scale court like the one that was provided during Design Day, then the camera can reliably be placed anywhere since it can see the entire court. For a larger scale court, it will be difficult to mount cameras at the side of the courts as the camera's field of view cannot see the

whole court. Mounting the camera to see above the court might be beneficial if the option is presented, however, phone mounts do not usually extend that high up.

4.1.2 Ball Colour

To allow the code to recognize the ball colour, then the findcolour must be selected to True. When the code is run using the play button in the top right of the screen, 2 windows will appear and look like this:

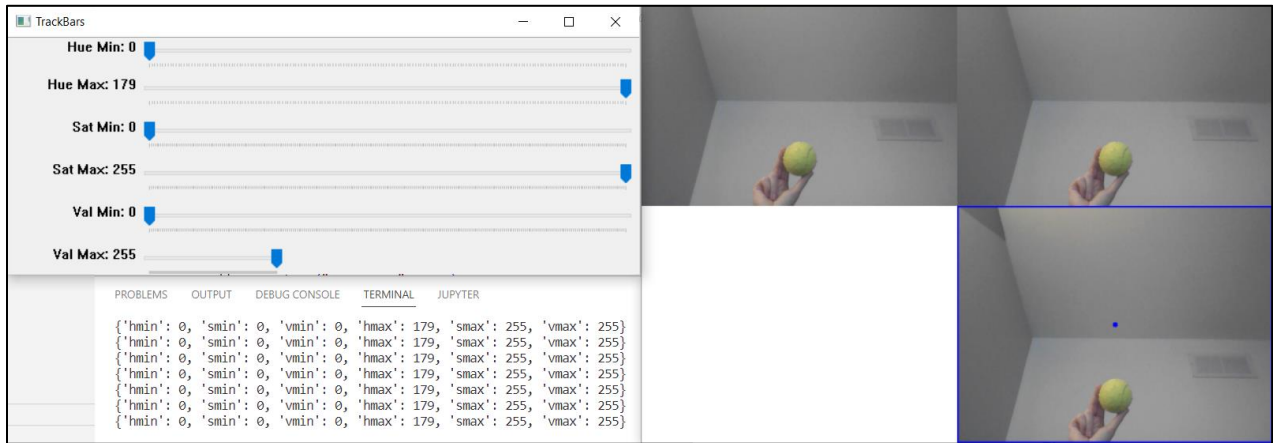


Figure 16: Ball Colour Sliders (Unselected Colour)

To isolate the colour of the ball, move the sliders with the mouse until you isolate the ball like in below:

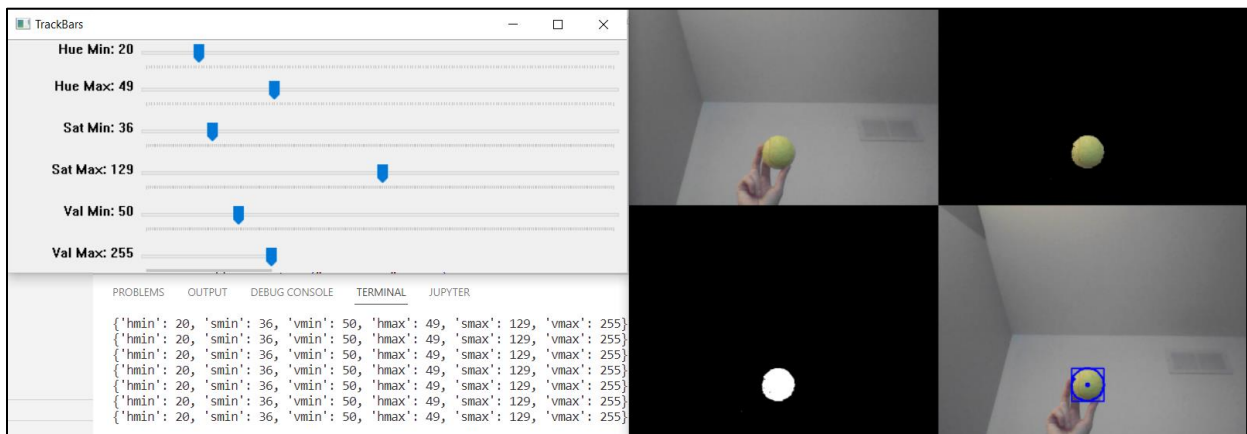


Figure 17: Ball Colour Sliders (Colour Selected)

Before stopping the code with the trash can icon in the bottom right, minimize the two windows that appeared. Note the values in the Terminal. These will be used to make sure the code can recognize the ball. Replace the numbers at the beginning of the code with those in the Terminal:

```
8 #if you want to bring up the colour slider, input True, and to stop the slider, input False
9 findcolour = True
10
11 #replace colour values here:
12 hsvVals = {'hmin': 20, 'smin': 36, 'vmin': 50, 'hmax': 49, 'smax': 129, 'vmax': 255}
```

Figure 18: Colour Value Inputs for the Python Code

Once the values have been found, you can stop the code with the trash can icon in the bottom right and then change *findcolour* to equal “False”. When *findcolour* is set to False and the code is run, then you should see that two windows appear and that the coordinates of the ball are displayed in the Terminal.

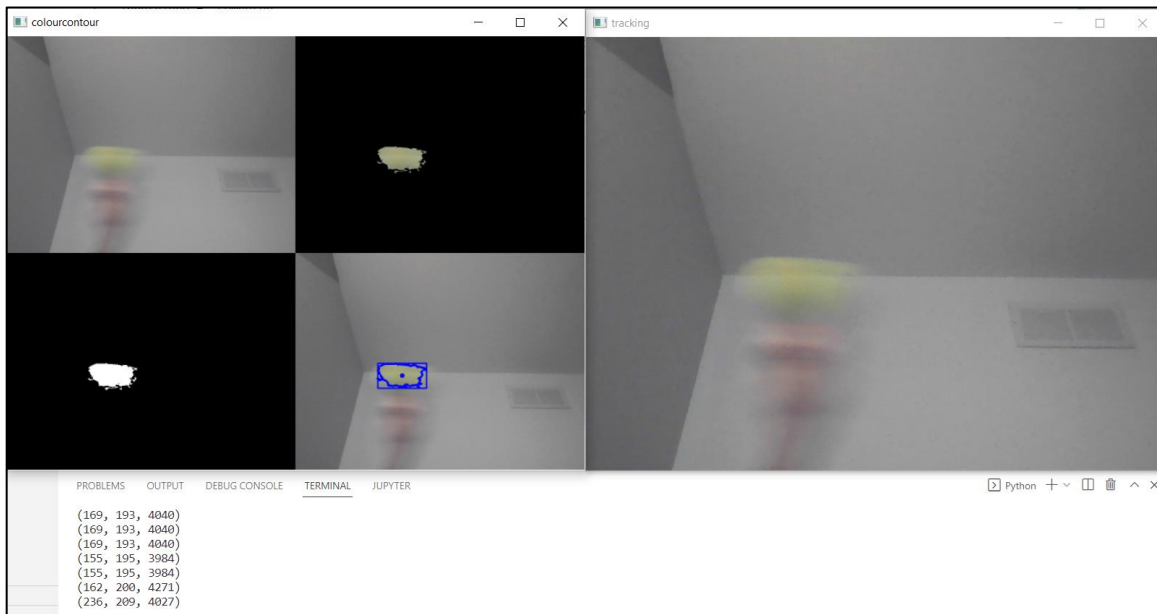


Figure 19: Code Outputs With Colour Selected

4.1.2.1 Advanced Commands

If the coordinates are still being shown while the ball is not in frame, then the number of pixels appearing on the screen must be found. Under certain lighting conditions, the colour of the ball can be similar to other coloured objects and make the code confused. By setting *findpixels* to True, you can find the maximum of “unwanted” pixels.


```
18 #ADVANCED COMMANDS
19 #if you wish to find the number of "unwanted" pixels, set findpixels to True
20 findpixels = True
21 numberofpixels = 10000
```

Figure 20: Selecting Number of Unwanted Pixels

By running the code when *findpixels* is set to true, do NOT put the ball in frame. The number of unwanted pixels will display in the terminal. Find the maximum number of unwanted pixels and put that value into the *numberofpixels*.



The screenshot shows a terminal window with a tabbed interface. The tabs are labeled 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', and 'JUPYTER'. The 'TERMINAL' tab is active and displays a list of numbers: 2771, 1488, 1343, 817, None, 8084, and 6191.

Figure 21: Number of Pixels Output to the Terminal

In this case, the number of unwanted pixels was rounded up to 10,000. It is ok to go with a number of pixels higher. You can then reset *findpixels* to False and run the code as normal.

4.1.3 Excel

To send the received coordinates to Excel to be stored, change the *excel* input to True. If you don't want to do this, then set it to False. To name the excel sheet, change the name within the quote marks.

```
14 #if you want to send the data to an excel file, input True, else, input False
15 excel = False
16 excelname = 'testproject1.xlsx'
```

Figure 22: Excel Inputs for the Python Code

The excel data will appear in the User Profile section where the code is stored. To access the User Profile section, refer to the section in the User Guide titled **Error! Reference source not found..** It will have the (x,y,z) coordinates of the ball including the speed and acceleration at each time point.

4.2 Unity

To use Unity, make sure that the Unity zip file is downloaded. Extract the file into the User Profile section on your computer and open up the .exe file.

Ball_Recognition_Isaac_Data	2022-11-30 7:29 PM
MonoBleedingEdge	2022-11-30 7:29 PM
Ball.py	2022-11-30 7:29 PM
Ball_Recognition.py	2022-11-30 7:29 PM
Ball_Recognition_Isaac.exe	2022-11-30 7:29 PM
UnityCrashHandler64.exe	2022-11-30 7:29 PM
UnityPlayer.dll	2022-11-30 7:29 PM

Figure 23: Zip File Containing Unity Environment

The Unity environment should appear as seen below. To exit the Unity Screen, press the Windows + Tab key to switch windows on your computer.

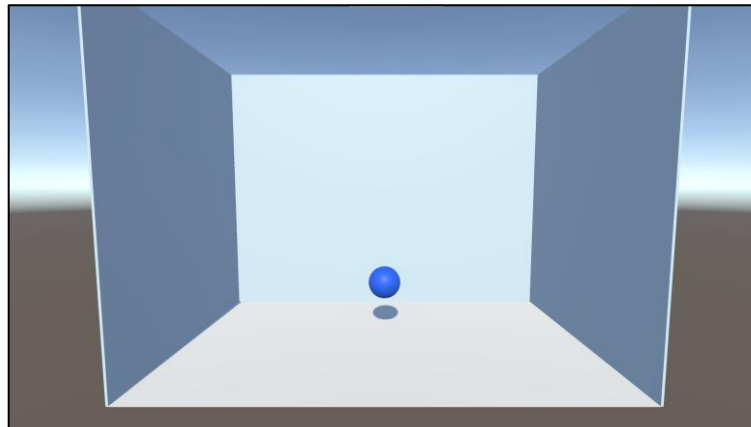


Figure 24: Unity Environment for Design Day

Run the code using the play button at the top right of the screen. The ball should move in the Unity environment.

5 Troubleshooting & Support

Common errors that were encountered while operating the system are listed below. Their recovery and correction procedures are mentioned below, and points of support are listed for the various components used in the PBATS system.

5.1 Error Messages or Behaviors

5.1.1 Code Errors

5.1.1.1 Indentation Error

One common error when running the code is the line indicating “indentation errors”. This occurs when the code is copied over improperly. This can be fixed by changing the file’s default indentation. At the bottom right of the code file, you can change the indentation by selecting the Spaces or Tabs:



Figure 25: Converting Indentation to Spaces/Tabs

At the top of the screen, it will give you options to *Convert Indentation to Spaces* or *Convert Indentation to Tabs*. Selecting either option often fixes the problem.

5.1.1.2 Camera [\[WARN:0@0.688\]](#) Error

If the camera is already opened either in another application or the wrong input is added to the camera input, then it is possible to get an error that begins with: *WARN:0@0.688*. Another message that will appear at the end of the message is: *AttributeError: 'NoneType' object has no attribute 'shape'*. Some examples of this happening is trying to run the program while the camera is on in another application such as a video meeting on teams or zoom. In this case, turn off your camera on teams/zoom and run the code as normal. This message can also appear if the ‘phone’ option is selected as an input but no phone camera is provided.

5.1.1.3 Module Not Found Error

If the proper libraries mentioned in Libraries are not downloaded, then an error message similar to: *Exception has occurred: Module Not Found* or *No module named*. Open the CMD as mentioned in Python and retry to install the libraries. This might also require an update, as listed in Python and Libraries.

5.1.2 Unity Errors

5.1.2.1 Ball not moving

If the code is running but the ball is not moving in Unity, then it is possible that the necessary files to send the data to Unity were not downloaded or are in the wrong place. Download the BallMovement.cs and UDPReceive.cs files from APPENDIX I: Design Files and add them in the User Profile section of the computer's files.

5.2 Maintenance

The python libraries, python system, 3rd party apps, or Unity may require updates in the future. If these updates appear, ignore them if possible as the code and version of Unity run on older versions. If the updates are necessary, then following the provided instruction for the system on how to update the product. Note that updates may lead to the system not functioning as intended.

5.3 Support

All the components used have some sort of online support systems for users to troubleshoot issues. The links to each system's support page can be found below. If the support pages cannot offer help, you may contact the teaching team of the University of Ottawa GNG 1103 course who can get in contact with our group to further assist. Errors can also be noted to the teaching team which will be forwarded to our group. The teaching team can be contacted through the following email: mmajeed@uottawa.ca or by phone: 613-562-5800 Ext 6163.

Table 3: Support Links for Various PBATS Components

Component	Support Link
VS Code	https://code.visualstudio.com/docs/supporting/FAQ
Elgato/EpocCam	https://help.elgato.com/hc/en-us/sections/360013950972-Elgato-Camera-Hub-Software
Unity	https://unity.com/support-services

If other errors are encountered with the code, then the given errors can be pasted into google. Chances are others have encountered the same problem and that a solution has been found. Often these solutions are posted to StackOverflow.com.

6 Product Documentation

6.1 Coding

The software/coding aspect of the final solution was chosen based on the information given to us through this course. The instructions to build the code and software's to download such as Visual Studio Code and Unity were given to this group through the labs that were offered with the GNG 1103 course. For this reason, that was the program that was chosen to build the final solution.

Throughout the process, many other object tracking options were considered. Through considerable research, the ball tracking options that were chosen for the final solution were those that had various examples online of how to set up the code and that had the best combination of range and accuracy.

6.1.1 BOM (Bill of Materials) and Equipment List

All of the materials that were needed to build this subsystem were available online for free; these included software platforms such as Python and Visual Studio Code. The full list of all libraries can be found in Libraries from Accessing/Setting-up the System, and all software that was used is listed below in Complete Bill of Materials.

6.1.2 Instructions

No physical building of this system is required. All of the code is completed in the PBATS.py file that was downloaded with the PBATS system. This subsystem however was built like any other code, through a trial-and-error process. The code was built piece by piece and was run many times to make sure each piece worked as it should before any other components were added.

6.2 Unity

6.2.1 BOM (Bill of Materials) and Equipment List

To build the Unity environment, the only equipment that was needed was the Unity Software.

6.2.2 Instructions

No physical building of this system is required. All of the work to create a Unity environment is completed in the Ball Recognition v5 zip file that was downloaded with the PBATS system. This subsystem however was built based on Unity tutorials online and was catered to resemble the mini Padel Ball court that was provided during Design Day. Using Unity tutorials online can allow for the creation of more complicated environment and more accurate looking courts.

6.3 Complete Bill of Materials

Table 4: Complete Bill of Materials for Final Solution

Item #	Item	Description	Quantity	Link/Source
1	Unity	3D environment software	1	https://unity3d.com/get-unity/download
2	Python	Ball tracking software	1	https://www.python.org/downloads/
3	Visual Studio Code	Coding software	1	https://code.visualstudio.com/download
4	Yolo Files	ball object tracking system	1	https://pjreddie.com/darknet/yolo/
5	Python Libraries	pip installs	6	Python Pip Installs
6	Phone Mount	phone tripod to hold camera	1 to 3	Provided for Design Day
7	PBATS Files	all files needed for the PBATS system	1	Provided with System
8	Excel	Used for data storage	1	Microsoft Office
9	Smartphone Camera	camera	1 to 3	User
10	Smartphone USB Cable	cable to connect phone to laptop	1 to 3	User
11	Laptop	computer	1	User
12	Power Cable	power cable for laptop	1	User
13	Extension Cord	power supplies to reach outlet	1	User
14	USB Drive	to store excel sheets (if wanted)	1	User

6.4 Testing & Validation

The ball tracking software underwent various tests as the last solution was being created to determine the maximum range and accuracy of the system. These results are listed below. The table also indicates where the system struggled and improvements that were implemented.

Table 5: Ball Tracking Software Functionality Test Results and Comments

Method	Colour Contour	Colour Track	Object Detection
Description	Code given to us in the first lab. Locates the desired colour and draws a mask.	Finds all the coloured pixels in the frame and averages them to a single point.	Uses a prefabricated neural network to detect objects.
Range	This system struggles when the ball is further than about 5ft.	This system can track balls that are at a considerable distance, provided no other similar coloured pixels are in the frame.	This system can track stationary balls at up to about 8ft.
Speed	This system could track the ball at slow speeds (rolled across a floor) and at medium speeds (passed between two people) This system runs smoothly and does not slow down the framerate.	This system could track the ball at slow speeds (rolled across a floor) and at medium speeds (passed between two people) This system runs smoothly but can slightly slow down the system compared to the contour method.	This system could track the ball at slow speeds (rolled across a floor) but occasionally at faster speeds. This system considerably slows down the system.

Accuracy	When the ball is within the speed and distance ranges, it can accurately track the ball about 90% of the time.	When there are no other similar pixels in the screen, this system can accurately track the ball nearly 100% of the time. When other similar pixels are present, then the accuracy decreases. If there are a few similar pixels, the accuracy is about 70-80% when the ball is near and about 20-30% when the ball is further.	At slower speeds, the system can track the ball about 80-90% of the time. At faster speeds, this reduces to 50% at times.
Comments	This system is reliable for closer balls. It can accurately pinpoint the ball's position at slower and faster speeds and when other coloured pixels are present in the frame. For padel applications, this system would not be beneficial at a distance.	This method struggles when similar coloured pixels are in the frame. When the ball is far away, the ball only registers about 10 pixels. This can easily confuse the system when there are many pixels scattered in the frame. To try and improve the system, the coordinates were only taken when the total pixels in the frame were greater than a selected number.	This method struggles when the ball smudged in the frame. Since the system is trained to detect round balls, when the ball is blurred the system cannot recognize it. This system also slows down the framerate quite a bit. This system is thus used as a last resort.

Once the ball tracking software was completed, then it was tested with the completed Unity environment to ensure the ball's movement was accurately recorded. The ball's coordinates were also sent to Excel and a verification of the data was completed to ensure the values that were recorded were reasonable and values that were expected.

7 Conclusions and Recommendations for Future Work

Throughout this project, lessons were learned in coding and 3D environment building. It was a great idea for the group to split up the work into parts that can be done individually. The hard thing with coding as a group is that it is hard to divide in a way that allows parts of the code to run individually.

For future work, the biggest improvement is related to the ball tracking. The methods used for the ball tracking, including the colour contour, the colour track, and the object detection only worked well if the code was tailored for success. The colour tracking struggled anytime similar coloured pixels were in the frame. The object detection had difficulty tracking a ball that had motion blur and was overall very slow. Improving the code to make it more accurate and faster would be the best next steps if we had more time to work on this project.

If the option presents to try the system with a faster computer or better cameras, this should be explored to see if the code can detect the ball better with a higher frame rate. Multiple cameras can also be tested simultaneously by running two or more instances of the same code separately. Our group was able to run the code with a laptop webcam and a phone camera at the same time but it slowed down the system considerably. If the group had more time, we would love to work on improving the framerate of the system when multiple cameras are in use.

8 Bibliography

- [1] M. W. -. R. a. A. AI, "YOLO v3 EASY METHOD | OpenCV Python," 26 June 2020. [Online]. Available: https://www.youtube.com/watch?v=Sp9mEGubBJs&list=PLMoSUbG1Q_r8nz4C5Yvd17KaXy8p0ufPH&index=5.
- [2] J. Redmond, "YOLO: Real-Time Object Detection," [Online]. Available: <https://pjreddie.com/darknet/yolo/>. [Accessed 10 November 2022].
- [3] asi55, "How to get screen coordinates from color detection resultant area obtained from openCV with python?," stackoverflow.com, November 2017. [Online]. Available: <https://stackoverflow.com/questions/48145249/how-to-get-screen-coordinates-from-color-detection-resultant-area-obtained-from>. [Accessed 10 November 2022].

APPENDICES

9 APPENDIX I: Design Files

Table 6. Referenced Documents

Document Name	Document Location and/or URL	Issuance Date
Downloads		
Python 3.7.0	https://www.python.org/downloads/release/python-370/	
Visual Studio Code	https://code.visualstudio.com/download	
Unity	https://unity.com/download	
EpocCam	https://www.elgato.com/en/downloads	
DroidCam	https://www.dev47apps.com/	
Files		
MakerRepo	https://makerepo.com/mariavans/1267.gng1103g10pbats	
PBATS.py	MakerRepo link	
yolov3.cfg	MakerRepo link	
yolov3.weights	https://pjreddie.com/darknet/yolo/ (this file was too large to upload to MakerRepo)	
BallMovement.cs	MakerRepo link	
UDPReceieve.cs	MakerRepo link	
Ball Recognition v5.zip	MakerRepo link	
Ball_Recognition_Isaac.exe	Ball Recognition v5.zip	