

Verification & Validation v0.5.1

Group 4: Autonomous Foosball Table

|  |  |
| --- | --- |
| Ryan Ma | 1147335 |
| Chenhe Li | 1158035 |
| Taha Hussain | 1200349 |
| Viktor Smirnov | 1154392 |
| Roland Zhou | 1216063 |
| Alvin Li | 0957972 |

# **1 REVISIONS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| 0 | January 10, 2016 | Ryan Ma,  Chenhe Li | Initial layout |
| 0.1 | January 24, 2016 | Ryan Ma,  Chenhe Li | Added introduction, test plans and testing schedule |
| 0.2 | February 1, 2016 | Chenhe Li,  Taha Hussain, | Added validation and adding traceability matrix |
| 0.3 | February 16, 2016 | Ryan Ma,  Chenhe Li,  Taha Hussain,  Viktor Smirnov,  Roland Zhou,  Alvin Li | Added hardware and software testing and verification |
| 0.4 | February 22, 2016 | Ryan Ma,  Chenhe Li,  Taha Hussain,  Viktor Smirnov,  Roland Zhou,  Alvin Li | Added Integration testing verification |
| 0.5 | February 29, 2016 | Ryan Ma | Updated project cost |
| 0.5.1 | February 29, 2016 | Chenhe Li | Fixed table links |

**Table 1. Revisions**

# **2 TABLE OF CONTENTS**

|  |  |
| --- | --- |
| [**1 REVISIONS**](#h.p33sj5s1v39c) | **1** |
| [**2 TABLE OF CONTENTS**](#h.e2033r3dyj3m) | **2** |
| [**3 LIST OF TABLES**](#h.1k7ln9izft1h) | **4** |
| [**4 INTRODUCTION**](#h.qflu4yvdxrbo) | **6** |
| [**4.1 Purpose**](#h.5wtuaaiv4amb) | **6** |
| [**4.2 Scope**](#h.8qk8zia4sf2q) | **6** |
| [**4.3 Background**](#h.40npxd36buvk) | **6** |
| [**4.4 Roadmap**](#h.4l8zi3lcvfrd) | **7** |
| [**4.5 Definitions and Abbreviations**](#h.54q4bse9dns5) | **7** |
| [**4.6 References**](#h.dlq60ulbi7h7) | **8** |
| [**5 VALIDATION**](#h.xvgbbve8s5qx) | **9** |
| [**5.1 Project Goals**](#h.ii3b0653u8mw) | **9** |
| [**5.2 Functional Validation**](#h.n8ybhvv2g9ti) | **9** |
| [**5.3 Behavior Validation**](#h.oa8p4yo8erpd) | **10** |
| [**6 VERIFICATION**](#h.uy8dmyc8ch8t) | **11** |
| [**6.1 Testing Schedule**](#h.90x0bw6ugq9r) | **11** |
| [**6.2 Traceability Matrix**](#h.uszujfuurhbb) | **12** |
| [**6.3 Testing Plan**](#h.46isy6jn3ll7) | **12** |
| [**6.3.1 Hardware Testing**](#h.nvwkle4hwcg0) | **12** |
| [**6.3.1.1 Arduino Mega with Ramps 1.4 Motor Shield**](#h.vwrmssev0act) | **13** |
| [**6.3.1.2 Emergency Stop and Reset button**](#h.dz67gjei5mi0) | **14** |
| [**6.3.1.3 Micro Limit Switches**](#h.8l55shvkm4bm) | **15** |
| [**6.3.1.4 LCD Controller and Display**](#h.pr420xdm6m9f) | **16** |
| [**6.3.1.5 A4988 Stepper driver and NEMA 17 Stepper motor**](#h.qsilyvy9htqo) | **17** |
| [**6.3.1.6 A4988 Stepper driver and NEMA 16 Stepper motor**](#h.f61w853u368u) | **18** |
| [**6.3.1.7 PlayStation Eye Camera**](#h.mw4yx84e1ipr) | **19** |
| [**6.3.1.8 Power supply**](#h.7aleoxfl0wb8) | **20** |
| [**6.3.2 Software Testing**](#h.etey6ow0wrw3) | **21** |
| [**6.3.2.1 Communication**](#h.d5e0s0ae43me) | **21** |
| [**6.3.2.2 Image Processing**](#h.9ztn2aoaon88) | **22** |
| [**6.3.2.3 Ball Detection**](#h.n9r81gxwgxoq) | **23** |
| [**6.3.2.4 Unknown Object Detection**](#h.txu7ggdyqnm3) | **24** |
| [**6.3.2.5 Foosman Detection**](#h.hvuwvrfqh3e7) | **25** |
| [**6.3.2.6 Goal Detection**](#h.jhmiqwnoke1) | **26** |
| [**6.3.2.7 Trajectory Prediction**](#h.ppf876hwev91) | **27** |
| [**6.3.2.8 Motor Controller**](#h.nd64m1wlqzug) | **28** |
| [**6.3.3 Integration Testing**](#h.2tfgu8do196c) | **29** |
| [**6.3.3.1 PC control of motors**](#h.dd15yj23bh8q) | **30** |
| [**6.3.3.2 Motors with mechanical linear and rotational system**](#h.i8y32pkpt1fh) | **31** |
| [**6.3.3.3 Image processing with AI**](#h.l3gg7ax5r7zt) | **32** |
| [**6.3.3.4 AI with full control of linear system**](#h.oaj1z3968efc) | **33** |
| [**6.3.4 Full system**](#h.o0m8xt2xnsg0) | **34** |
| [**7 APPENDIX A. TESTING TOOLS**](#h.pp1qe0c43ulm) | **35** |
| [**8 APPENDIX B. HARDWARE LIST**](#h.axmm9x5bv2ya) | **36** |

# **3 LIST OF TABLES**

[Table 1. Revisions](#id.415a8hz1gg3b)

[Table 2. Project Roadmap](#id.tszjntwns39a)

[Table 3. Definitions and Abbreviations](#id.hauvktld3wqy)

[Table 4. Function Validation](#id.6hz3cdncakgu)

[Table 5.Testing Schedule](#id.4f2gh8gzqstz)

[Table 6. Used hardware and its description](#id.u05qbza42vxw)

[Table 7. Arduino Mega with Ramps testing summary](#id.lqe3g55j8x07)

[Table 8. Button testing summary](#id.adw6yqmcuep3)

[Table 9. Limit switches testing summary](#id.2dmes5dj2nuj)

[Table 10. LCD testing summary](#id.h9v9u0fcdohe)

[Table 11. NEMA 17 stepper motor and motor driver testing summary](#id.2tn2q06vuajb)

[Table 12. NEMA 16 Stepper motor and motor driver testing summary](#id.2c2vr9ie2jpn)

[Table 13. Camera testing summary](#id.vhtq6w1rb9na)

[Table 14. Power supply testing summary](#id.wvnomvjz8vcb)

[Table 15. Communication testing summary](#id.awhu3irr8zcj)

[Table 16. Image processing testing summary](#id.y1noi63j0b9o)

[Table 17. Ball detection testing summary](#id.p4rq43shllzs)

[Table 18. Unknown object detection testing summary](#id.xsg9e39ckh0i)

[Table 19. Foosmen detection testing summary](#id.q0f20e32jqfi)

[Table 20. Goal detection testing summary](#id.8z65jzbe44xy)

[Table 21. Trajectory Prediction testing summary](#kix.gqq2uvev855j)

[Table 22. Motor Controller Testing](#id.o0bqmmnyd4fz)

[Table 23. Integration Testing](#kix.vhpne4g38pqx)

[Table 24. PC controls of motor testing](#id.3vufsh6f9yoz)

[Table 25. Mechanical linear and rotational system testing](#id.u4r7r73sl11l)

[Table 26. Image processing with AI testing](#id.ik7xqm39abbz)

[Table 27. AI with full control of linear system](#id.kd9am95sna7y)

[Table 28. Full system testing](#id.emrbsi5g8mm3)

[Table 29. Hardware List](#id.hgyy94605rtp)

# **4 INTRODUCTION**

## **4.1 Purpose**

The goal of the project is to design and build an autonomous arcade-style foosball table for a single user. This document provides required verification results meant to make sure that the product has met the requirements[4] (5.5 [References](#h.dlq60ulbi7h7)). Verification will be achieved by several testing cases, specifically designed to test a certain parameter or a feature of the design.

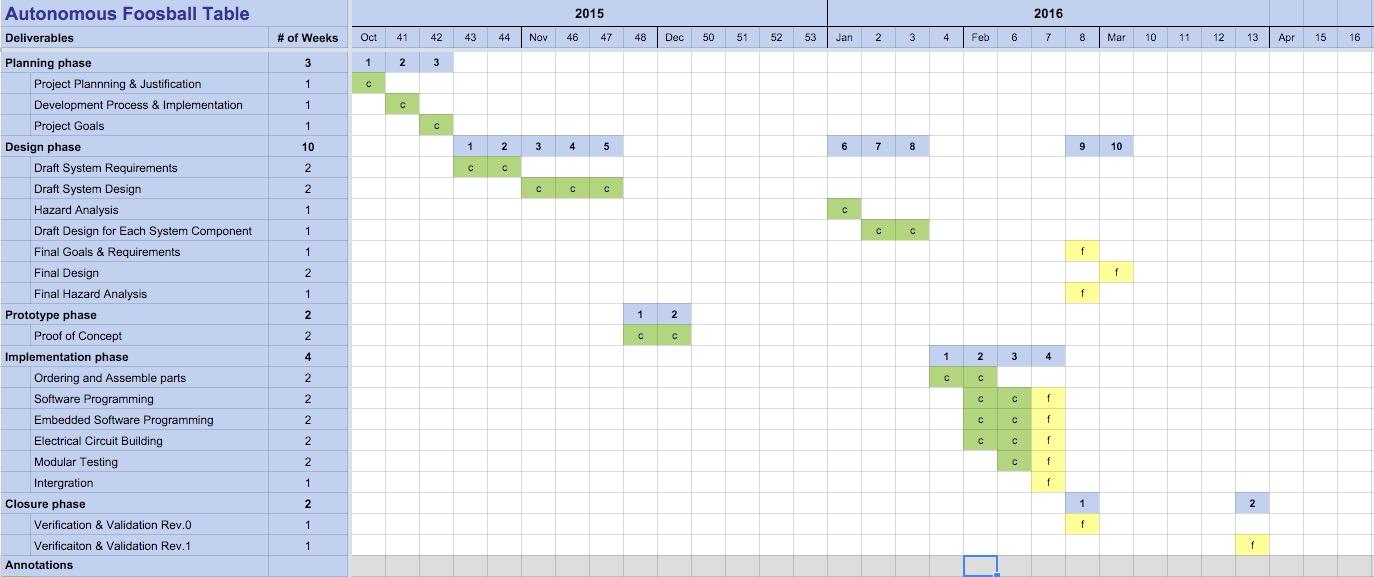
## **4.2 Scope**

The focus of this validation and verification document is to proof that each part of our implementation complies with its requirements[4]. Since the final product is not yet assembled, it will be difficult to completely validate the system, as the hardware and software components are not yet interfaced. Instead we will validate parts of the system which can be directly linked to the validation of the entire system as required by this document.

## **4.3 Background**

The autonomous foosball table is an arcade-style foosball table that offers a challenging opponent to the user. It allows one to play foosball with an AI opponent, at various levels of difficulty, without the need for another human player.

## **4.4 Roadmap**



**Table 2. Project Roadmap (More detailed version available here:** [**link**](https://docs.google.com/spreadsheets/d/1D8vNFgQwGkfuoN6klKOyD6agbPdgBtAi7UDL9EPomso/edit?usp=sharing)**)**

## **4.5 Definitions and Abbreviations**

|  |  |
| --- | --- |
| **Term** | **Definition** |
| foosball table | The autonomous mini foosball table |
| foosman | The players in the foosball table |
| ball | The ball used in the game |
| point | A float tuple of x and y position |
| webcam | Sony PLAYSTATION Eye™ camera uses as input hardware device in computer vision. The captured frames are in RGB color space. It can achieve up to 75 fps at 640x480 pixels, or 120 fps at 320x240 pixels. |
| RGB | RGB color space constructs all the colors from the combination of the Red,Green and Blue colors. The red, green and blue use 8 bits each, which have integer values from 0 to 255. Detail explanation is in [15.1](https://docs.google.com/document/d/1dFD0laP8AWmf8O3S8VwII7dKFG9aCEEmOUwyb67VPE8/edit#heading=h.uai5n5r12nau). |
| HSV | HSV color space constructs all the color based on three components: hue, saturation and value. Detail explanation is in [15.2](https://docs.google.com/document/d/1dFD0laP8AWmf8O3S8VwII7dKFG9aCEEmOUwyb67VPE8/edit#heading=h.oia9fy115igl). |
| Requirements | requirements document mentioned in the document refers to System requirements v1.0; more information, please see the [reference](#h.dlq60ulbi7h7) below. |
| System Design | System Design document mentioned in the document refers to System Design v0.1; more information, please see the [reference](#h.dlq60ulbi7h7) below. |
| Goal | Project Goal document mentioned in the document refers to Project Goal v1.0; more information, please see the [reference](#h.dlq60ulbi7h7) below. |

**Table 3. Definitions and Abbreviations**

## **4.6 References**

[1] “Foosball Rules” Internet: <http://www.foosball.com/learn/rules/>

[2] IEEE Std 829-2008 <https://standards.ieee.org/findstds/standard/829-2008.html>

[3] OLE lab rules <https://www.cas.mcmaster.ca/olelab/>

[4] requirements v1.0 <https://docs.google.com/a/mcmaster.ca/document/d/1KLijYmAGySJdBUqk-_72h1H353zOqcEP-3-h_0saIUA/edit?usp=sharing>

[5] Design v0.2.2 <https://docs.google.com/a/mcmaster.ca/document/d/1dFD0laP8AWmf8O3S8VwII7dKFG9aCEEmOUwyb67VPE8/edit?usp=sharing>

[6] Project Goals v1.0

<https://docs.google.com/a/mcmaster.ca/document/d/10P7kMsps4Be8Bb1oQaBf5ovAyHaff8p_agIm3RywDH8/edit?usp=sharing>

[7] Traceability Matrix

<https://docs.google.com/spreadsheets/d/1ThC3LChBBVlU2AGbgoVK_FY-nrwnhTIHwQQRt85p6UY/edit?usp=sharing>

# 

# 

# **5 VALIDATION**

## **5.1 Project Goals**

The goal of our project is to design a foosball table addon that adds a competitive robotic opponent. The target market of our table is owners of public places for gatherings, like bars and arcades that are willing to update their existing foosball tables to offer extra challenge to their customers.

The autonomous foosball table addon should adhere to all the rules of a foosball game with each module being designed to follow those rules as close as possible. The addon is meant to be fun to play against, which means it must provide enough competition to entertain the user. The price of the future product will be one of the features that differentiate our product with the product in the market right now. However, while we were aiming to set the maximum price at 300$, we soon realized that we have to raise the original budget to $600 in order to complete the product with the desired functionality but this is due to the fact that we ordered custom parts.

## **5.2 Functional Validation**

The system features has several operations in order to achieve the desired behavioral functionality. The system will capture the position of the ball, predict the trajectory of the ball and use a suitable strategy to arrange the positions and rotational movement of the controlled foosman.

|  |  |
| --- | --- |
| **Goal** | **Characteristics and Functions**  **of Current Implementation** |
| The system shall follow the basic rules of foosball | * **Micro Limit Switches** are added to limit the maximum angular displacement of the robot foosmen. |
| The system shall consist of three difficulty levels for different player skills | * **Greedy Algorithm** will be applied to act as the easiest form of AI opponent. * **Path Finding Algorithm** will be applied to act as the intermediate form of AI opponent * **Actively Learning Algorithm** will be applied to act as the advanced form of AI opponent |
| The system shall score at least 50% of the shots taken during perfect conditions (unopposed) | * **Path Finding Algorithm** will be used to find the shortest path from the foosball’s current position to the opponent’s goal line without getting blocked by the opponent foosmen |
| The system shall block at least 70% of human shots in highest difficulty level | * **Ball Path Prediction** will be used to prediction foosball’s future position based on current and past information and the robot foosmen will move to corresponding positions to block the shot |
| The system shall halt within 200ms after unknown objects are present in the foosball playfield | * **Unknown Object Detection** can eliminate the potential hazards of hurting human player * **Emergency Stop Button** will cut off the power supplies to all actuators to stop potential safety hazards |

**Table 4. Function Validation**

## **5.3 Behavior Validation**

Behavior of the system will be easier to validate once the product is completed. However the currently implemented system is implemented in a way such that the individual subsystems can be validated separately. This is achieved by modularizing software subsystems. This also means that as the project progresses, new subsystems will be validated after being integrated into the system. This will ensure the system is be regression tested throughout the development.

# **6 VERIFICATION**

This project is developed under the V-Model development process. Therefore the system and its subsystems will be verified in a bottom-up pattern. The bottom level is **Unit Testing**, in this stage we tested individual units of the system. This includes all individual hardware components of the system, as well as all software functions that will be used in the control program. This helped us to eliminate defective hardware and bugs at code level. The second level is **Module Testing**, where we tested all available subsystems of the system, including available hardware and software modules. The third level is **Integration Testing**. In this stage we integrated all available subsystems and tested their coexistence, particularly focusing on interfacing of the subsystems within the system. The top level is **System Testing**, meant to test the system functionality as a whole to ensure it meets the requirements[4] and project goals[6].

## **6.1 Testing Schedule**

The testing schedule of this project is shown below in [Table 5](#id.4f2gh8gzqstz). (Detailed version can be found in the [reference](#h.dlq60ulbi7h7) section)



**Table 5.Testing Schedule**

## **6.2 Traceability Matrix**

The traceability matrix documents the links between the requirements and the developed product to implement and verify those requirements. The traceability matrix captures all requirements and their traceability in a excel spreadsheet. However, for convenient display purpose, the matrix can be found [here](https://docs.google.com/spreadsheets/d/1ThC3LChBBVlU2AGbgoVK_FY-nrwnhTIHwQQRt85p6UY/edit?usp=sharing), or in the attachment on Avenue referenced as Traceability Matrix[7].

## **6.3 Testing Plan**

### **6.3.1 Hardware Testing**

The table below lists the hardware that will be used in the system and must be tested for its functionality before being implemented in the design. The motors, however, will not be verified and are instead going to be assumed to be working according to their data sheet specifications. The linear rail system as well as other mechanical components are not included, as these are just static components and mechanical interfaces. Hardware datasheets are available in the [Appendix B](#h.axmm9x5bv2ya) .

|  |  |
| --- | --- |
| **Hardware name** | **Description** |
| Nema 17 stepper motors | Motors that are being used for rotational and linear translation |
| Arduino Mega 2560 with RAMPS 1.4 Motor shield | Chosen controller for the motors,sensors and communication with PC |
| Stop and Reset buttons | Buttons that come on the LCD controller and will be used to either reset the system or stop it in case of emergency |
| Limit Switches | Hardware limit switches that will be used to home the motors |
| Stepper Driver A4988 | Motor driver interface |
| LCD Display | LCD to display system status |
| Camera | Input camera for image processing |
| Power Supply | For converting 120V AC to the required 12V DC. |

**Table 6. Used hardware and its description**

#### **6.3.1.1 Arduino Mega with Ramps 1.4 Motor Shield**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.1.1.1 | Verify that controller is working properly | Run a test program to verify the result is the expected output | Algorithm that toggles the required pins | Pins should toggle with a fixed frequency | N/A | N/A |

**Table 7. Arduino Mega with Ramps testing summary**

#### 

#### 

#### **6.3.1.2 Emergency Stop and Reset button**

Emergency stop and Reset buttons are designed for user to stop whole system quickly and easily. The testing of emergency and reset buttons is to ensure those two buttons are working properly.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.1.2.1 | Verify that Reset button works properly | Push button and see the if controller detects an impulse | Impulse from the button | HIGH detected on GPIO pin | N/A | N/A |
| 6.3.1.2.3 | Verify that Reset button works properly | Leave button as is and see the if controller detects an impulse | N/A | LOW detected on GPIO pin | N/A | N/A |
| 6.3.1.2.4 | Verify that the Emergency stop button works properly | Push button and see the if controller detects an impulse | Impulse from the button | HIGH detected on GPIO pin | N/A | N/A |
| 6.3.1.2.4 | Verify that Reset button works properly | Leave button as is and see the if controller detects an impulse | N/A | LOW detected on GPIO pin | N/A | N/A |

**Table 8. Button testing summary**

#### 

#### 

#### **6.3.1.3 Micro Limit Switches**

Micro limit switches are used on linear and rotational motion, the testing is to ensure switches are working properly.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.1.3.1 | Verify that the linear limit switches are working properly | Push limit switch and see the if controller detects an impulse | Impulse from the switch | HIGH detected on GPIO pin | N/A | N/A |
| 6.3.1.3.2 | Verify that the rotational limit switches are working as expected | Push limit switch and see the if controller detects an impulse | Impulse from the switch | HIGH detected on GPIO pin | N/A | N/A |

**Table 9. Limit switches testing summary**

#### 

#### 

#### 

#### **6.3.1.4 LCD Controller and Display**

LCD controller and display are used to display information. The testing is to ensure LCD controller and display are working properly.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.1.4.1 | Verify that the LCD is working properly. | Send in characters and see if it print them out as expected | “This is an LCD test” ot the LCD port using existing firmware | LCD displays “This is an LCD test” | N/A | N/A |

**Table 10. LCD testing summary**

#### 

#### 

#### **6.3.1.5 A4988 Stepper driver and NEMA 17 Stepper motor**

Verify the functionality of the motor driver by sending a series of impulses to see whether the number of steps moved matches the number of impulses sent. Motors steps = 200 steps/revolution for all motor in use.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.1.5.1 | See description above | directly measure rotational angular using protractor | 12v to stepper driver. 200 impulse with pulse rate of 800Hz | 360 degrees rotation | stepper motor rotates 360 degrees | Pass |
| 6.3.1.5.2 | See description above | directly measure rotational angular using protractor | 12v to stepper driver. 200 impulse with pulse rate of 1000Hz | 360 degrees rotation | stepper motor rotates 360 degrees | Pass |
| 6.3.1.5.3 | See description above | directly measure rotational angular using protractor | 12v to stepper driver. 200 impulse with pulse rate of 1250Hz | 360 degrees rotation | stepper motor rotates 360 degrees | Pass |
| 6.3.1.5.4 | See description above | directly measure rotational angular using protractor | 12v to stepper driver. 200 impulse with pulse rate of 1333Hz | 360 degrees rotation | stepper motor rotates 0 degrees | Fail |

**Table 11. NEMA 17 stepper motor and motor driver testing summary**

#### 

#### 

#### 

#### **6.3.1.6 A4988 Stepper driver and NEMA 16 Stepper motor**

Verify the correct functioning of the motor driver by sending impulse and checking if the number of steps moved matches the number of impulse. Motors steps = 200 steps/revolution for all motor in use.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.1.6.1 | Check if the motor is able to withstand a particular frequency | Directly measure rotational angular using protractor | 12v to stepper driver. 200 impulse with pulse rate of 800Hz | 360 degrees rotation | stepper motor rotates 360 degrees | Pass |
| 6.3.1.6.2 | Check if the motor is able to withstand a particular frequency | Directly measure rotational angular using protractor | 12v to stepper driver. 200 impulse with pulse rate of 1000Hz | 360 degrees rotation | stepper motor rotates 360 degrees | Pass |
| 6.3.1.6.3 | Check if the motor is able to withstand a particular frequency | Directly measure rotational angular using protractor | 12v to stepper driver. 200 impulse with pulse rate of 1250Hz | 360 degrees rotation | stepper motor rotates 0 degrees | Fail |
| 6.3.1.6.4 | Check if the motor is able to withstand a particular frequency | Directly measure rotational angular using protractor | 12v to stepper driver. 200 impulse with pulse rate of 1333Hz | 360 degrees rotation | stepper motor rotates 0 degrees | Fail |

**Table 12. NEMA 16 Stepper motor and motor driver testing summary**

#### **6.3.1.7 PlayStation Eye Camera**

Verify System Design[5] 9.1.1. The camera can continuously capture quality image output with frame rate of 75Hz or higher.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.1.7.1 | Testing camera capturing frequency at specific resolution | Testing using Asus G53 computer and counting the number of frame over 60s | Camera setting at 90 fps at 320x240 pixels | more than 4500 high quality frames | 4920 to 5100 high quality frames | Pass |
| 6.3.1.7.2 | Testing camera capturing frequency at specific resolution | Testing using Asus B43 laptop and counting the number of frame over 60s | Camera setting at 90 fps at 320x240 pixels | more than 4500 high quality frames | 4800 to 4960 high quality frames | Pass |

**Table 13. Camera testing summary**

#### 

#### 

#### **6.3.1.8 Power supply**

The testing of power supply is to ensure that it provide correct amount of voltage that the system needs.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.1.8.1 | Verify the correct functioning of the 12 V power supply | Measure output voltage using Tetronix DMM 4020 | 110-120V AC to the SUNSKY S-360-12  Power supply | The output voltage is 12V | 12.0 to 12.4 V | Pass |

**Table 14. Power supply testing summary**

### 

### 

### **6.3.2 Software Testing**

#### **6.3.2.1 Communication**

There is communication between laptop and motor controller. The testing of communication is to ensure that the data is being sent correctly and automatically.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.2.1.1 | Verifying that the integers sent to the motor controller are interpreted correctly | Send data from the Arduino serial monitor and print them out back on the serial monitor | strings of random data that is known | String that is sent is printed out | String are printed as they are ent | Pass |
| 6.3.2.1.2 | Verifying that the software interface used to communicate from PC is sending data properly | Send data from PC interface after having tested that Arduino interprets data correctly. | Characters that are ready to be accepted by motor controller | Serial monitor prints the characters sent to the Arduino using PC interface with serial communication | Correct characters are printed out in the correct order | Pass |

**Table 15. Communication testing summary**

#### 

#### 

#### **6.3.2.2 Image Processing**

These tests verify the image processing component listed in System Design[5] 9.1.1. Image process can get image from camera and cover it to HSV image with frame rate higher than 71 Hz and check if camera connect to computer. The Asus G53 computer was used for these tests

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.2.2.1 | Verify setupCam() function | Function returns false if camera not connected | no camera connected | returns false | returns false | Pass |
| 6.3.2.2.2 | Verify setupCam() function | Function return true if camera connected properly | connect camera to computer | returns true | returns true | Pass |
| 6.3.2.2.3 | Verify getHSV() function by counting output number over 60s | Testing using Asus G53 computer and counting frames over 60s | camera sends at 90 fps at 320x240 pixels | processing more than 4300 HSV images | 4780 to 4900 high quality frames | Pass |
| 6.3.2.2.4 | Verify getHSV() function counting output number over 60s | Testing using Asus B43 laptop and countting frames over 60s | camera sends at 90 fps at 320x240 pixels | processing more than 4300 HSV images | 4300 to 4560 high quality frames | Pass |

**Table 16. Image processing testing summary**

#### **6.3.2.3 Ball Detection**

Ball detection is used to accurately detect ball position on field, processing time and accuracy are important factor for the test. Detail description and requirements are listed in System Design[5] 9.1.1, 9.1.2 and requirements[4] 6.2.6 and 6.2.9. The Asus G53 computer was used for these tests.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.2.3.1 | Verify design 9.1.2- ball is not in table | Remove ball from table and see the output from the software | Image from camera | returns -1,-1,false | returns -1,-1,false | Pass |
| 6.3.2.3.2 | Verify requirement 6.2.6 (The successful rate of ball detection) | Get ball location from camera and compare it with physical measurements of ball position | Image from camera and ball is stable | ~ball location | ball location±1.3mm | Pass |
| 6.3.2.3.3 | Verify requirement 6.2.9 | Keep track of total number of frames and number of frames in which moving detected when ball is physically made to move | Image from camera and ball is moving | successful rate more than 85% | successful rate more than 92% | Pass |
| 6.3.2.3.4 | Verify design 9.1.2 | Obtain a set number of images and compare it to the time it took using time tracking libraries | camera sets at 90 fps at 320x240 pixels | processing rate higher than 71 Hz | processing rate higher than 78 Hz | Pass |
| 6.3.2.3.5 | Verify design 9.1.2 | Obtain a set number of images and compare it to the time it took using time tracking libraries | camera sets at 90 fps at 320x240 pixels | processing rate higher than 71 Hz | processing rate higher than 38 Hz | Fail |

**Table 17. Ball detection testing summary**

#### **6.3.2.4 Unknown Object Detection**

The purpose of unknown object detection is to successfully detect any unknown object that appear above the foosball field. These tests verify the components in System Design[5] 9.1.3. This module should correctly detect unknown image and processing time is less than 5ms.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Test Tool** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.2.4.1 | Verification of design 9.1.3 (processing time is less than 5ms) | Testing using Asus G53 computer | 1000 HSV images with unknown object | Returns TRUE for all images | returns TRUE for 998 out of 1000 images | Pass |
| 6.3.2.4.2 | Verification of design 9.1.3 (Processing time is less than 5ms) | Testing using Asus G53 computer | 1000 HSV images without unknown object | Returns FALSE for all images | Returned FALSE for all images | Pass |
| 6.3.2.4.3 | Verify requirement 6.2.9  (The successful rate of ball detection) | Testing using Asus G53 computer | Image from camera and ball is moving | Rate of success more than 85% | successful rate more than 92% | Pass |
| 6.3.2.4.4 | Verify design 9.1.1, 9.1.2,9.1.3 | Testing using Asus G53 computer | camera sets at 90 fps at 320x240 pixels | processing rate higher than 71 Hz | processing rate higher than 67 Hz | Pass |
| 6.3.2.4.5 | Verify design 9.1.1, 9.1.2,9.1.3 | Testing using Asus B43 laptop | camera sets at 90 fps at 320x240 pixels | processing rate higher than 71 Hz | processing rate higher than 32 Hz | Fail |

**Table 18. Unknown object detection testing summary**

#### 

#### 

#### **6.3.2.5 Foosman Detection**

Foosman detection is used to locate foosmen’s positions (both linear and angular); processing time and accuracy are the most important factor. These tests verify the foosman detection in System Design[5] 9.1.2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Test Tool** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.2.5.1 | Verification of design 9.1.2 (foosmen are not in table) | Testing using Asus G53 computer | Image from camera | Return of  {-1, -1, false} | Return of  {-1, -1, false} | Pass |
| 6.3.2.5.2 | Verification of design 9.1.2 (foosmen are static on table) | Testing using Asus G53 computer | Image from camera and ball is stable | Foosmen linear position | Foosmen linear  position of ±1.4mm | Pass |
| 6.3.2.5.3 | Verification of design 9.1.1, 9.1.2, 9.1.3, 9.1.3 and 9.1.4 | Testing using Asus G53 computer | camera sets at 90 fps at 320x240 pixels | Processing rate of higher than 60Hz | Processing rate  of higher than 42Hz | Fail |
| 6.3.2.5.4 | Verification of design 9.1.1, 9.1.2, 9.1.3 and 9.1.4 | Testing using Asus B43 laptop | camera sets at 90 fps at 320x240 pixels | Processing rate of higher than 60Hz | Processing rate of higher than 12Hz | Fail |

**Table 19. Foosmen detection testing summary**

#### 

#### 

#### 

#### 

#### **6.3.2.6 Goal Detection**

Goal Detection is used to monitor goals at two ends; in term of testing, process time and accuracy are important. These tests verify the design of goal detection listed in System Design[5] 9.1.4.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Test Tool** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.2.6.1 | Verification of design 9.1.4 | Testing using Asus G53 computer | 1000 HSV images with ball in user goal | Returns of {true, false} | Returns  {true, false}  for 778 images,  {false, false} for 222 images. | Fail |
| 6.3.2.6.2 | Verification of design 9.1.4 | Testing using Asus G53 computer | 1000 HSV images without ball in goal | Return of {false, false} | Returns  {false, false} for 1000 images | Pass |
| 6.3.2.6.3 | Verification of design 9.1.4 (the processing time is less than 2ms) | Testing using Asus G53 computer | HSV images | Processing time of less than 2ms | Processing time of less than 0.5ms | Pass |
| 6.3.2.6.4 | Verification of design 9.1.4 (processing time is less than 2ms) | Testing using Asus B43 laptop | HSV images | Processing time of less than 2ms | Processing time of less than 0.9ms | Pass |

**Table 20. Goal detection testing summary**

#### **6.3.2.7 Trajectory Prediction**

Trajectory prediction is used to predict ball position; this will approximate ball speed, ball direction and ball position in the next 2 seconds. These tests verify the design listed in System Design[5] 9.2.1.The test results are affected by environment, such as field ground.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Test Tool** | **Input** | **Expected Behaviour** | **Actual Behaviour** | **Pass/Fail** |
| 6.3.2.7.1 | Verification of design 9.2.1  getBallSpeed() | Testing using Asus G53 computer | continuous foosball positions with timestamp | Returns  foosball actual moving speed | Returns foosball moving speed | Pass |
| 6.3.2.7.2 | Verification of design 9.2.1  getTrajectory() | Testing using Asus G53 computer | continuous foosball positions with timestamp | Returns foosball actual moving direction | Returns  correct foosball moving direction for 700 out of 1000 continuous foosball positions | Fail |
| 6.3.2.7.3 | Verification of design 9.2.1  predictTrajec-  tory() | Testing using Asus G43 computer | continuous Foosball positions with timestamp | Returns  foosball actual position in the next 2 seconds | Returns  correct foosball position in the next 2 seconds for 500 out of 1000 continuous foosball positions | Fail |

**Table 21. Trajectory Prediction testing summary**

#### **6.3.2.8 Motor Controller**

[This Table](#id.o0bqmmnyd4fz) below will encompass all the software functions implemented in the motor controller some of which require hardware to test.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected output** | **Actual output** | **Pass/Fail** |
| 6.3.2.8.1 | Testing of method Distance\_to\_NeededSteps(int linear length,int current motor step number). | Give in inputs of lengths and current motor positions and see if function outputs the expected number of steps. Equation in design doc . | -(9.5,1000)  -(0,0)  -(5,5)  -(3,400)  -(5,2) | - -683  - 0  - 161  - -300  - 164 | - -683  - 0  - 161  - -300  - 164 | Pass |
| 6.3.2.8.2 | Testing of method Angle\_to\_NeededSteps(float angle,int motor\_current). | Give in inputs of angle and current motor position and compare with manually calculated output | - (360,200)  - (0,0)  -(40,100)  -(10,0)  -(30,0) | - 0  - 0  - -77  - 5  - 16 | - 0  - 0  - -77  - 5  - 16 | Pass |
| 6.3.2.8.3 | Testing the method check\_safety\_reset() | Press the buttons and see if the function controller responds immediately | Impulse from stop or reset button | controller prints message of stop or reset | N/A | N/A |
| 6.3.2.8.4 | Testing the method check\_limit\_switch() | Press the switches and see if the controller responds | Impulse from limit switches | Controller ptins message of impulse detected | Controller ptins message of impulse detected | Pass |
| 6.3.2.8.5 | Testing the method Homing\_Config() | Lets the motors run and then press the switch to s | Impulse from limit switches | current motor positions are set to 0 and motors stop moving | N/A | N/A |

**Table 22. Motor Controller testing summary**

### **6.3.3 Integration Testing**

The foosball system has many components which are independently being developed and will have to be tested with each other in a systematically and in-depth manner to ensure that system requirements[4] are met before integration with the entire system. We will be doing integration in the following steps.

|  |  |  |
| --- | --- | --- |
| **Step** | **Integration Test** | **Description** |
| 1 | PC control of motors | This will be done so that we can verify that communication is able to work with insignificant delays along with motor and sensor control |
| 2 | Motors with mechanical linear and rotational system | This will done to test if motors are able to meet the linear and rotational timing requirements. |
| 3 | Image processing with AI | This will be done to check the time delay for these process |
| 4 | AI with full control of linear system | This will be done to see if the system is works with inputs that we will decide |
| 5 | Full system | This will be done to see if all the modules interfaced with each other are working according to requirements |

**Table 23. Integration testing summary**

#### **6.3.3.1 PC control of motors**

Observing if the we can send the distance and angle to the motor controller and get a response from the motors under different conditions. The response of the motors should meet the requirements set by requirements[4] 6.2.1. These scenarios are for when only one instance of length and angle are sent followed or preceded by a possible action.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test ID** | **Input** | **Expected result** | **Actual result** | **Pass/Fail** |
| 6.3.3.1.1 | - length and angle through serial com.  - no switches or user inptuts entered | Movement of motors corresponding to thee lengths and angles. | As expected | Pass |
| 6.3.3.1.2 | - length and angle through serial com.  - limit switches hit | Movement of motors corresponding to those lengths and angles until limit switch is hit after which the linear motor which hit the switch stops. | As expected | Pass |
| 6.3.3.1.3 | - limit switches.hit  - length and angle through serial com. | Movement of motors corresponding to the lengths and angles | As expected | Pass |
| 6.3.3.1.4 | stop button pressed | No movement of motors | N/A | N/A |
| 6.3.3.1.5 | Reset button pressed | Motors go to home position then stop | N/A | N/A |
| 6.3.3.1.6 | - length and angle through serial com  - stop button pressed | Motors show no movement | N/A | N/A |
| 6.3.3.1.7 | - length and angle through serial com  - reset button pressed | Motors go to length and angle until reset is hit after which they go to home the position then stop. | N/A | N/A |
| 6.3.3.1.8 | - stop button pressed  - length and angle through serial com | Motors show no movement | N/A | N/A |
| 6.3.3.1.9 | - stop button pressed  - reset button pressed  - length and angle through serial com | Movement of motors corresponding to those lengths and angles | N/A | N/A |

**Table 24. PC Motor Controls testing summary**

#### **6.3.3.2 Motors with mechanical linear and rotational system**

The testing is ensure that mechanical linear and rotational system is meeting with requirements[4] which defined in 6.2.3.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected result** | **Actual result** | **Pass/Fail** |
| 6.3.3.2.1 | Seeing if the motor is able to meet linear motion requirements defined in 6.2.3 of the requirements doc | Make the linear motor system move from one end to the other a hundred times and record the time. Average speed can be found. High speed cameras will be used. | Step impulses that make the motor go back and forth over the linear rangeAs expected | Linear system meets requirements | As expected | Pass |
| 6.3.3.2.2 | Seeing if the motor is able to meet rotational motion requirements defined in 6.2.5 of the requirements doc | Make the rotational system rotate a 100 times while recording the time. Average angular speed can be found. High speed camera will be used. | Step impulses that make the motor rotate constantly a 100 times. | Rotational system meets requirements | As expected | Pass |

**Table 25. Linear and Rotational motion system testing summary**

#### **6.3.3.3 Image processing with AI**

The testing is to ensure that camera frame rate will not drop by AI and calculate correct expected result.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Method** | **Input** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 6.3.3.3.1 | Checking if the frame rate from the image processing does not drop below required frame rate when interfaced with AI | Run both the modules and keep track of the frame rate. After 2minutes the average frame rate can be calculated. | Moving the ball to different places on the board triggering different strategies | Frame rate does not drop below requirements | N/A | N/A |
| 6.3.3.3.2 | This is about measuring the time it takes for the two modules to process information at each frame and calculate a length and an angle. | Using system clocks to measure the intervals and calculate the average time at the end of at least a 1000 intervals and compare its significance to the system response time defined in 6.2.3 in the requirements doc | Moving the ball to different places on the board triggering different strategies. | Average response time obtained is well below system response time. | N/A | N/A |

**Table 26. Image processing (with active AI) testing summary**

#### **6.3.3.4 AI with full control of linear system**

The testing of AI with control of linear system is to ensure that linear system response AI command correctly.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Input** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 6.3.3.4.1 | Seeing if the system responds correctly from the inputs given to the AI. | Auto generated variables as well as manual edge cases needed by the AI to make a decision | Foosball players move according to predicted movements described in the AI. | N/A | N/A |

**Table 27. Testing summary of AI with full control of the linear system**

### **6.3.4 Full system**

We have tested the PC control of motors, ensuring that the motor controller is working properly. The AI interfacing was tested with the rest of the system, with and without image processing being active. These steps ensured that the synchronisation between the controller and PC is effective. Then the only thing remaining was to see how the system behaves once fully interfaced. The most important component is the timing between obtaining a frame and the foosman reacting to a situation since image processing is the slowest kn. These tests will be aimed at measuring the timing for the entire system.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test ID** | **Input** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 6.3.4.0.1 | Moving of ball to different fixed positions | Foosman player moves to predicted location and reacts in predicted manner | N/A | N/A |
| 6.3.4.0.2 | Rolling the ball away and towards the player | Foosman react in predicted manner as described by the AI | N/A | N/A |
| 6.3.4.0.3 | Press start button to start or resume a game and no unknown objects show in table | The system shall start game if the game is at initial state; or resume current pausing game | N/A | N/A |
| 6.3.4.0.4 | Make the ball enter goal | The system shall goes to waiting mode and wait for user to retrieve the ball and foosball players return default position until user change mode | N/A | N/A |

**Table 28. Full system testing summary**

# **7 APPENDIX A. TESTING TOOLS**

The following tools are used for testing the autonomous foosball table:

* High Speed Camera: iPhone 6
* ASUS G53SX-A1 Intel Core i7 2630QM (2.00 GHz) 12 GB Memory 750 GB HDD NVIDIA GeForce GTX 560M 15.6" Windows 7
* ASUS BU400A-W3117G Intel Core i5-3317U 1.7GHz Windows 10
* Tektronix DMM4020 Digital Multimeter
* Tektronix MSO 2024 Oscilloscopes
* Half Circle Protractor

# **8 APPENDIX B. HARDWARE LIST**

|  |  |  |
| --- | --- | --- |
| **Name** | **Quantity** | **Link of specification** |
| V-Slot Rail-285-LP | 2 | <http://openbuildspartstore.com/v-slot-linear-rail/> |
| V-Slot Gantry Set | 1 | <http://openbuildspartstore.com/v-slot-gantry-set/> |
| Nema 17 Stepper Motor | 2 | <http://www.pbclinear.com/Download/DataSheet/Stepper-Motor-Support-Document.pdf> |
| GT2 Timing Belt-3 Feet | 2 | <http://www.york-ind.com/print_cat/york_2mmGT2.pdf> |
| GT2 Timing Pulley (30 Tooth) | 4 | <http://www.york-ind.com/print_cat/york_2mmGT2.pdf> |
| Micro Limit Switch (with mounting plate) | 2 | <http://openbuildspartstore.com/micro-limit-switch-kit-with-mounting-plate/> |
| Power HD High-Speed Mini Servo HD-1705MG | 2 | <https://www.pololu.com/product/2143> |
| SUNSKY S-360-12  Power Supply | 1 | <http://upload.sunsky-online.com/res/drivers/12V30A.pdf> |
| NEMA 17 Stepper Motor Mount | 2 | <https://www.pololu.com/product/2266> |
| Flexible Helical Coupling | 2 | <http://www.farnell.com/datasheets/43730.pdf> |
| Arduino Mega | 1 | <http://www.microelectronicos.com/datasheets/ArduinoMega2560.pdf> |
| RAMPS 1.4 Stepper Motor Shield | 1 | <http://reprap.org/wiki/RAMPS_1.4> |
| A4988 Stepper Motor Driver  (with heatsink) | 2 | <http://www.robotshop.com/media/files/PDF/datasheet-1182.pdf> |
| 2004 LCD Control Board | 1 | <http://www.farnell.com/datasheets/58820.pdf> |
| Mini Pushbutton Switch | 1 | <http://www.farnell.com/datasheets/1658379.pdf> |
| Resistor(1K) | 1 | <http://www.petervis.com/electronics/Standard_Resistor_Values/1K0.html> |
| PLAYSTATION Eye**™** | 1 | <https://www.playstation.com/manual/pdf/CEJH15001.pdf> |
| LM2596 DC-DC Converter | 1 | <http://www.aliexpress.com/item/DC-DC-Buck-Converter-Step-Down-Module-LM2596-Power-Supply-Output-1-23V-30V/846265709.html> |

**Table 29. Hardware List**