University of Tartu

Faculty of Science and Technology

Institute of Technology

Märten Josh Peedimaa

Rhea: An open-source table tennis ball launcher robot for multiball training

Bachelor's thesis (12 ECTs)
Computer Engineering

Supervisors: Eva Mõtshärg, M.Sc.
Meelis Pihlap, M.Sc.

Tartu 2024

Abstract/Resümee

Rhea: An open-source table tennis ball launcher robot for multiball training
Multiball training is a commonly used technique by table tennis athletes to enhance their skill and performance. Existing commercial robot solutions for this purpose are closed source, making repairs and modifications challenging. Whilst non-commercial solutions do exist, they are limited in capability, and none are specifically made with multiball training and replicability in mind. This thesis documents the development of an open-source table tennis ball launcher robot named Rhea for the use of multiball training. The work analyses the robot's accuracy compared to a commercial robot. The work shows, that Rhea is suitable for multiball training and is comparable to a commercial solution.

CERCS: T125 Automation, robotics, control engineering; T455 Motors and propulsion systems

Keywords: table tennis, sports training robots, 3D printing, embedded systems, open-source, multiball training

Rhea: Avatud lähtekoodiga lauatennise palliviske robot mitmikpall treenimiseks
Mitmikpall treenimine on lauatennise sportlaste seas levinud treenimisviis. Olemasolevad robot kommertslahendused mitmikpall treeninguks ei võimalda Mittekommertslikud lahendused on vähem levinud ning ükski pole tehtud konkreetselt mitmikpall treeningut ja taasloomist silmas pidades. Käesolev lõputöö dokumenteerib Rhea nimelise lauatennise palliviske roboti arenguskäiku. Lõputöö analüüsib roboti täpsust ning võrdleb seda kommertslahendusega. Lõputöö näitab, et Rhea sobib mitmikpall treeningu kasutamiseks ning on võrreldav kommertslahendusega.

CERCS: T125 Automatiseerimine, robootika, juhtimistehnika; T455 Mootorid ja ajamid

Märksõnad: lauatennis, 3D printimine, manussüsteemid, avalikseadmed, mitmikpall treenimine

Contents

Abstract/Resümee **2**

List of figures **5**

List of tables **10**

Acronyms **11**

1 Introduction **12**

2 Prior work **13**

2.1 Commercial solutions 13

2.1.1 PowerPong and Butterfly robots 13

2.1.2 Newgy Robo-Pong series 16

2.1.3 JOOLA iPONG series 17

2.2 Non-commercial solutions 18

2.2.1 AIMY 18

2.2.2 Robot by Ponnusamy et al. 19

2.2.3 Spin-Ninja Table Tennis Training Robot 20

3 Requirements **22**

4 Development of Rhea's mechanical components **23**

4.1 Ball Pusher 24

4.1.1 First revision 24

4.1.2 Second revision 26

4.1.3 Pipe 28

4.2 Launch Unit 29

4.2.1 First revision 29

4.2.2 Second revision 31

4.2.3 Third revision 34

4.3 Pivot Mechanism 35

4.4 Fixture 37

5 Overview of Rhea's electronic system **40**

6 Firmware and user software **44**

6.1 Firmware 44

6.2 User software 47

7 Validation **48**
7.1 Initial accuracy tests 48
7.2 Automated testing setup 49
7.3 Comparison with a commercial robot 52

8 Discussion **56**

9 Conclusion **57**

10 Acknowledgements **58**

References **59**

Appendix A Publicised materials **63**

Appendix B Silicone experiments and moulds **63**

Appendix C Automated testing setups **67**

Non-exclusive license **69**

List of figures

1 PowerPong's *OMEGA* robot mounted on a table tennis table with a ball collection net in the background. The robot can be controlled using the tablet shown in the foreground. [7] 13

2 The *AMICUS PRIME* robot sold by Butterfly, mounted on a table tennis table. The robot can be controlled using the tablet shown in the foreground. A net behind the robot is used for reusing balls returned by the player. [9] 14

3 Control panels of the two lower-tier *AMICUS* series robots offered by Butterfly. [9] 14

4 Butterfly's *AMICUS PRIME* robot [9]. Balls accelerated by the three traction wheels are deflected by the annotated ball deflector. The robot can adjust the deflector's altitude angle and rotation (yaw) to affect a ball's trajectory and direction. 15

5 The *Robo-Pong 3050XL* robot sold by Newgy. The robot is mounted on a table tennis table and features a net (shown in the background) for capturing and reusing balls returned by the player. A smartphone application can be used to control the robot. [13] 16

6 The *iPONG V300* robot sold by JOOLA. The robot is placed on top of a table tennis table and can adjust its launch direction by rotating its upper section. An infrared remote is used to configure the robot. [16] 17

7 AIMY, an open-source table tennis ball launcher for versatile and high-fidelity trajectory generation, positioned behind a table tennis table. A sample launch trajectory of a ball is illustrated. [11] 18

8 Cropped image of AIMY's Launch Unit with annotated locations for the linear motor and servomotor used to adjust launch direction. [17] 18

9 3D model and prototype (a) and user interface (b) of the robot by Ponnusamy et al. [24] 20

10 Spin-Ninja, placed on a table tennis table. Side view (top left), top-down view (top right), and rear view (bottom). [25] 21

11 3D model of Rhea's final version with annotated components. 23

12 3D model of the Launch Unit with Cartesian axes. 23

13 Annotated section analysis of the Ball Pusher's first revision. The crank, connecting rod and piston make up a slider-crank linkage. Balls fall from the input rail to the housing and are pushed upwards into the Pipe. 24

14 Section analyses of the Ball Pusher's first revision illustrating a ball being pushed upwards into the Pipe. The movement of the piston (green) and ball stopper (blue) is illustrated in four consecutive steps. A rubber band forces the stopper to the position illustrated by the first and last steps by pulling upwards on a small tab on the left side of the stopper, effectively preventing balls from falling back down. 25

15 Prototype of Rhea using the first revision of the Ball Pusher. Table tennis balls ready to be pushed upwards are placed on an angled rail. 25

16 Section analysis of the Ball Pusher's second revision with annotated components. 26

17 The second revision of the Ball Pusher. A clear rubber band is used to force the ball stopper into an upright position. A funnel (not depicted) can be pressure fit on top of the large opening to direct balls into the housing. 27

18 Exploded view of the Ball Pusher's slider-crank linkage. Two retaining pins prevent the connecting rod from coming loose. 27

19 3D model (a) and section analysis (b) of the Pipe. Blue arrows depict the trajectory of a ball from the Ball Pusher, through the Pipe and into the Launch Unit. 28

20 Section analysis of the Pipe illustrating its slightly larger inner diameter of 42 mm allowing a 40 mm table tennis ball to freely move upwards. 28

21 3D model (a) and prototype (b) of the Launch Unit's first revision. Red circles in both subfigures indicate slots used for mounting a motor to a single blade. 29

22 First revision of the Launch Unit fixed to a table using a vice and a metal rod. O-rings stretched around the motors are used for gripping the ball. . . 30

23 Annotated 3D model of the Launch Unit's second revision highlighting the additions and changes from the first revision. 31

24 Section analyses of the Launch Unit's second revision. Three steps illustrate how a hook (yellow), driven by a servomotor, buffers a ball received from the Pipe. A downwards-facing infrared sensor is used to detect the presence of a ball. When a ball is detected, the servomotor rotates the hook counterclockwise, pushing the ball towards the three launching motors. . . 32

25 Final version of the moulds and respective positives made using Shore A 5 polysiloxane used as traction wheels for launching the ball. 33

26 Two iterations of the motor guard. 33

27 Section analyses of the Launch Unit's second (a) and third (b) revision. The third revision's chamber features a steeper angle for receiving a ball from the Pipe. This prevents the ball from rolling back (to the right) when the pitch of Rhea is adjusted and the Launch Unit rotates clockwise along the x-axis. Two holes (one depicted) at the bottom allow for securing the third revision to the Pivot Mechanism using M3 bolts. 34

28 Two side views of thee Launch Unit's third, final revision. Red squares highlight how two M3 bolts are used to secure the Launch Unit to the Pivot Mechanism. 34

29 Assembled (a) and exploded (b) 3D models of the Pivot Mechanism. A bearing is pressure fit into the housing, allowing for its inner ring to freely rotate. The driven gear is secured to the connecting shaft which is pressure fit into the bearing's inner ring. A servomotor, secured to the housing, rotates the driver gear which causes the bigger gear (red) to rotate. The connecting shaft is fitted inside the Launch Unit. 35

30 Three configurations illustrating the Pivot Mechanism rotating the Launch Unit to one side in order to alter a launch's yaw angle. The left column depicts a top-down view whilst the right column depicts a view from the side. The Pivot Mechanism is capable of rotating the Launch Unit 30° in either direction. 36

31 Top-down view of Rhea mounted on a table tennis table. The highlighted purple area shows the theoretical targeting area enabled by the Pivot Mechanism. The highlighted green area shows the effective targeting area of 75% when taking into account the height of the net. 37

32 Prototype of the Fixture with its components annotated. Prototypes of other components are shown to illustrate how they would connect to the Fixture. 38

33 Annotated 3D model of the Fixture. 3D printed dowels are used for securing various components. The dowels are shown in an extracted state with arrows depicting their mounting direction. The pivot and pitch adjustment dowels are rounded. 38

34 Side view of the Fixture demonstrating how Rhea's components are connected together. The Pipe and the Ball Pusher are secured to their respective Fixture connectors using M3 bolts. 39

35 Section analyses of Rhea illustrating how the pitch can be manually regulated by moving the pitch adjustment dowel into one of three slots in the Fixture. The pitch can be adjusted to 0, 10 or 20 degrees. 39

36 Schematic of Rhea's electronic components and their connections. Blue connections represent PWM (*Pulse-Width Modulation*) signals, green connections analog signals, and purple connections USB (*Universal Serial Bus*) signals. 40
37 ESC (*Electronic Speed Controller*) used for regulating the speed of a launching motor using PWM. Hot glue applied to the ESC's connectors to prevent damage when the wires are twisted. The ESC is later wrapped in electrical tape to cover its exposed components. 41
38 Protoboard used for distributing 5 and 3.3 V power to the servomotors and IR reflective sensors, as well as for connecting the resistors used by the IR reflective sensors. 42
39 Annotated 3D model of the electronics enclosure with its lid removed. Six M3 bolts are used to secure the protoboard and ESP32 module to the enclosure. The DC-DC converter and wiring are not depicted. 42
40 Rhea mounted on a table next to the electronics enclosure. The wires are tied together using cable ties and electrical tape. An AC/DC power supply and a USB wire are connected to the enclosure. 43
41 State machine of the firmware running on the ESP32 module. 45
42 Structure of an individual 8-byte packet, consisting of a header (1 byte), an ID (1 byte), content (5 bytes), and a footer (1 byte). 45
43 Contents of the AddShot (a), RemoveShot (b), and StartSession (c) packets. Type, name, description, and value range is provided for each field. 46
44 GUI (*Graphical User Interface*) of the user software running on a computer used for managing a training session with Rhea. Two example shots with different parameters are configured. 47
45 First accuracy measurement setup using the Launch Unit's third revision. The orange arrows depict the launch direction of the ball. 48
46 Landing location data for Rhea at a 10° angle for Shore A 22 and Shore A 5 polysiloxane traction wheels. The mean, average standard deviation, and confidence ellipse with 3 times the standard deviation are illustrated for both polysiloxane hardnesses. 49
47 Annotated model depicting the automatic setup used for gathering ball landing location data. A camera mounted above the table records slow motion video. The transparent red pyramid represents the camera's field of view. A piezoelectric element placed on the table detects ball bounces. . . 50
48 3D printed smartphone mount located above the table tennis table, fixed to a wooden ceiling using screws. The smartphone's camera provides a top-down view of one side of the table and it can be slid in and out of the mount. 50

49 Gathering calibration data for resolving the camera's intrinsic and extrinsic properties for the automatic test setup. A 6x9 flat calibration chessboard is moved to various locations on the table, and recorded by a camera. Three sample locations are pictured. 51
50 Detecting a ball using OpenCV's SimpleBlobDetector (a) and constructing a ray from the camera towards the detected ball (b). The ray's intersection point with the table's plane represents the ball's landing location. 52
51 Six target landing locations used for automated testing. 53
52 Landing location data for Rhea at a 10° angle gathered using the automated test setup and Shore A 22 traction wheels. Two datasets shown: one where the target location is not changed between launches (a), and one where it is changed (b). The mean, average standard deviation, and confidence ellipse with 3 times the standard deviation are illustrated for each target location. 54
53 Landing location data for the Butterfly *AMICUS PRIME* gathered using the automated test setup. Two datasets shown: one where the target location is not changed between launches (a), and one where it is changed (b). The mean, average standard deviation, and confidence ellipse with 3 times the standard deviation are illustrated for each target location. 55
54 Removing dried caulk from the mould's first iteration. 63
55 Unevenly dried caulk inside the second iteration of the mould. 64
56 Caulk drying inside third iteration moulds, inner surfaces lubricated with oil. 64
57 Removing dried caluk from a third iteration mould. 65
58 Fourth iteration of the moulds, poured with Shore A 22 polysiloxane. . . . 65
59 Fifth iteration the moulds, poured with Shore A 22 polysiloxane. 66
60 Rhea mounted on a table tennis table at a 10° altitude angle. The launched balls are caught using a ball catch net (*JOOLA Carbon Graphite Ball Catch Net* [54]) and returned under the table using a gutter for convenience. A roll of yellow masking tape holds down the piezoelectric element on the opposite side of the table for detecting ball bounces. A smartphone camera (not pictured) mounted on the ceiling records the landing locations of the balls. 67
61 Butterfly AMICUS PRIME mounted on a table tennis table with the included ball catch net behind it. The launched balls were caught by hand and returned to the robot's side of the table. A roll of yellow masking tape holds down the piezoelectric element on the near side of the table for detecting ball bounces. A 3D printed smartphone camera (green) can be seen mounted to the wooden ceiling. 68

List of tables

1 Specifications for the *AMICUS PRIME* and *Power Pong OMEGA* robots. Both robots have the same capabilities. [7, 9] 15
2 Specifications for the *Robo-Pong 3050XL* robot sold by Newgy. [13] 16
3 Specifications for JOOLA's *iPONG V300* robot. [16] 17
4 Specifications for AIMY by Dittrich et al. [11] 19
5 Specifications for the robot by Ponnusamy et al. [24] 20
6 Specifications for Spin-Ninja. [25] 21
7 Packets used for communication between the user software and firmware. . 46

Acronyms

ADC (*Analog-to-Digital Converter*). Electronic system used to convert an analog signal into a digital signal. 38, 41

API (*Application Programming Interface*). Software library of routines for application in a specific domain or problem. 11, 15, 41

AWG (*American Wire Gauge*). Standardised system for measuring wire gauge (diameter). 38

BLDC (*Brushless Direct Current*). Techinque used in direct current electric motors to eliminate friction caused by brushes. 11, 27

BOM (*Bill of Materials*). List of materials required to manufacture an end product. 7, 18, 56

ESC (*Electronic Speed Controller*). Electronic circuit used for regulating the speed of an electric motor. 27, 38, 41, 43

GUI (*Graphical User Interface*). Visual interface enabling user interaction with software. 15, 17, 44

ITTF (*International Table Tennis Federation*). Governing body for all national table tennis associations. 7

PLA (*Polylactic acid*). Thermoplastic polyester commonly used in 3D printing. 18, 19, 29, 53

PWM (*Pulse-Width Modulation*). Technique used in electronic circuits to represent an analog signal's amplitude by varying the duty cycle of a rectangular wave. 27, 37, 38, 41

UART (*Universal Asynchronous Receiver-Transmitter*). A peripheral device which enables communication through an asynchronous serial link. 41, 43, 44

USB (*Universal Serial Bus*). Common interface used for connecting different types of electronics. 37, 41, 47

1 Introduction

Table tennis (also known as ping-pong) is a racket-based sport played by two people (singles) or four people (doubles). Table tennis has its roots in 20th-century England and has since achieved worldwide popularity thanks to its affordability, simple ruleset, and the possibility of playing year-round [1]. The ITTF (*International Table Tennis Federation*) estimated in 2023 the sport has over 30 million active competitive players [2]. In 2018, the major ITTF events drew a viewership of approximately one billion [3].

One of the most common methods used by table tennis athletes to improve their gameplay is multiball training [4]. Multiball training consists of a coach feeding balls to a trainee in quick succession. The direction and speed of each ball can be altered, requiring the trainee to adapt to various situations they might face during a real match. Multiball training utilising a robot ball launcher in place of a coach has been proven to be a highly effective training method [5].

This thesis focuses on the development and manufacture of an open-source table tennis ball launcher robot named Rhea that can be used for multiball training. The keyword "open-source" implies that interested parties will be able to build the robot and improve its design as they see fit. This is enabled by publishing the BOM (*Bill of Materials*), 3D models, assembly instructions, and source code of the robot.

The structure of this thesis is as follows. Section 2 analyses existing commercial and non-commercial solutions for multiball training. Section 3 outlines the requirements that were established prior to the development phase. Sections 4, 5, and 6 cover the development of the mechanical, electronic, and software components of Rhea respectively. Section 7 analyses the accuracy of Rhea compared to a commercial robot using an automated testing setup. In Section 8 the author gives a subjective overview of the thesis and discusses future work. Finally, Section 9 summarises the thesis.

2 Prior work

In the context of robotics, table tennis has seen the development of different types of robots that either aid the player in training, play the game fully autonomously or provide a way to generate data for machine learning applications. The following subsections cover commercial and non-commercial solutions that can be used for multiball training. Commercially available robots are offered by multiple manufacturers in a wide (∼150 to ∼2000 EUR) price range. Non-commercial solutions are uncommon and some are not under explicit permissive licenses, or do not provide enough insight to allow for replication by third parties. All prices given are from the time of writing (2024-05-20).

2.1 Commercial solutions

2.1.1 PowerPong and Butterfly robots

PowerPong and Butterfly are companies that sell table tennis robots manufactured by a third company, LCS Sportszer, in Hungary. The robots are based on the same hardware but both companies provide their own software and user interfaces for interacting with the robot. [6] The robots use 3 motors, mounted 120° apart, with hard foam wheels attached to transfer the motors' angular momentum to the ball. Both robots feature a net and a ball recycling system to capture and reuse balls returned by the player, effectively removing the need for periodically restocking the robot with balls.

Figure 1: PowerPong's *OMEGA* robot mounted on a table tennis table with a ball collection net in the background. The robot can be controlled using the tablet shown in the foreground. [7]

PowerPong's current series' flagship is the *Power Pong OMEGA* (shown in Figure 1)

with a price of 2195 USD. Included with the *OMEGA* robot are a remote control and an application that may be installed on a smartphone or tablet that can be used to program various training drills. [7]

Butterfly offers the *AMICUS* robot which they provide in three different tiers: *START*, *EXPERT*, and *PRIME* that differ in their user interfaces [8]. *PRIME* (shown in Figure 2) is the most expensive of the three, costing 1799 EUR, and comes with a tablet and application which can be used to program training drills akin to PowerPong's application [8]. The *START* and *EXPERT* tiers are not controllable by an application and instead use a control panel as can be seen in Figure 3.

Figure 2: The *AMICUS PRIME* robot sold by Butterfly, mounted on a table tennis table. The robot can be controlled using the tablet shown in the foreground. A net behind the robot is used for reusing balls returned by the player. [9]

(a) *AMICUS START*'s control panel.

(b) *AMICUS EXPERT*'s control panel.

Figure 3: Control panels of the two lower-tier *AMICUS* series robots offered by Butterfly. [9]

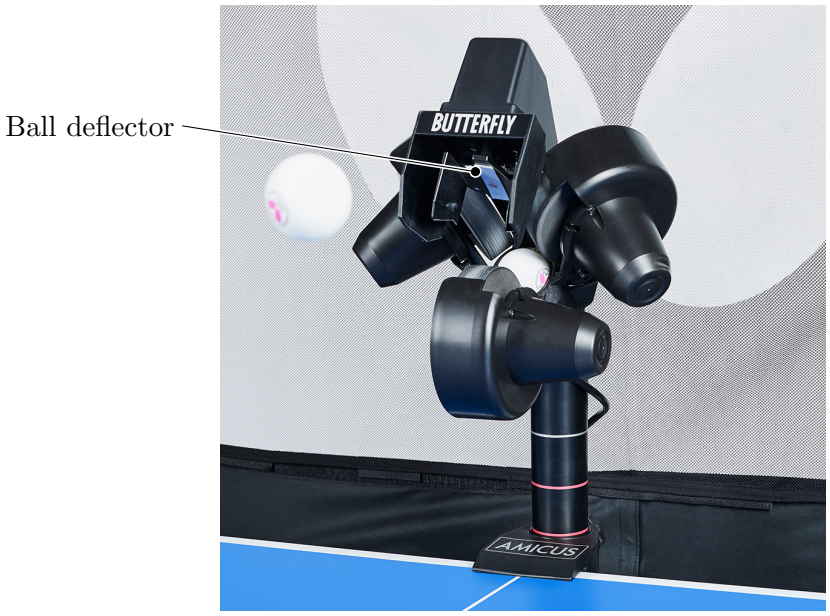


Figure 4: Butterfly's *AMICUS PRIME* robot [9]. Balls accelerated by the three traction wheels are deflected by the annotated ball deflector. The robot can adjust the deflector's altitude angle and rotation (yaw) to affect a ball's trajectory and direction.

Figure 4 illustrates the ball deflector that both robots use to affect the ball's trajectory and direction.

Parameter	Value
Price	1799 EUR (*AMICUS PRIME*) 2195 USD (*Power Pong OMEGA*)
Launching motors	3
Launch frequency	5-120 balls per minute
User interface	Smartphone/tablet application
Weight	9 kg

Table 1: Specifications for the *AMICUS PRIME* and *Power Pong OMEGA* robots. Both robots have the same capabilities. [7, 9]

Both companies provide their own custom application with the PowerPong's application being more actively updated [10]. Both applications are closed-source and no API (*Application Programming Interface*) is provided that would allow for programming fully customisable training exercises [11]. Table 1 outlines the key parameters for both robots.

2.1.2 Newgy Robo-Pong series

Newgy is a company producing and selling table tennis products with a focus on robots. Their *Robo-Pong* series has six robots of which the flagship *Robo-Pong 3050XL* costs 2199.99 USD. [12, 13]

The *Robo-Pong 3050XL* (shown in Figure 5) uses two BLDC (*Brushless Direct Current*) motors to launch the balls and a third motor to rotate the head of the robot. This design differs from the usual three-motor launching mechanism in that it allows the robot to generate a purely sidespinning ball.

Figure 5: The *Robo-Pong 3050XL* robot sold by Newgy. The robot is mounted on a table tennis table and features a net (shown in the background) for capturing and reusing balls returned by the player. A smartphone application can be used to control the robot. [13]

Parameter	Value
Price	2199.99 USD
Launching motors	2
Launch frequency	1-120 balls per minute
User interface	Smartphone/tablet application Microsoft Windows application
Weight	8 kg

Table 2: Specifications for the *Robo-Pong 3050XL* robot sold by Newgy. [13]

Additionally, the robot is capable of producing serves by tilting its head downwards and launching a ball that first bounces on its own side of the table [14]. A net is used

to capture and reuse balls, akin to the PowerPong and Butterfly robots. The robot's specifications are outlined in Table 2.

2.1.3 JOOLA iPONG series

JOOLA, a manufacturer and retailer of table tennis accessories, offers the *iPONG V200* and *iPONG V300* for 169 EUR and 219 EUR respectively [15, 16]. The *iPONG V300* (shown in Figure 6) is not mounted on the edge of the table tennis table as the previous robots, it is simply placed on the table. The robot is capable of varying its direction by rotating its upper section and uses two motors for accelerating the ball. The upper half of the robot acts as a ball reservoir which must be restocked manually. An infrared remote is used to control the robot wirelessly.

Figure 6: The *iPONG V300* robot sold by JOOLA. The robot is placed on top of a table tennis table and can adjust its launch direction by rotating its upper section. An infrared remote is used to configure the robot. [16]

Parameter	Value
Price	219 EUR
Launching motors	2
Launch frequency	5-70 balls per minute
User interface	Infrared remote
Weight	1.1 kg

Table 3: Specifications for JOOLA's *iPONG V300* robot. [16]

The *iPONG V300* is an entry-level robot and cannot produce as complex shots as the previous robots but makes up for the fact with its price tag and lower weight. Table 3 summarises the *iPONG V300*'s parameters.

2.2 Non-commercial solutions

2.2.1 AIMY

Dittrich et al. [11] have developed AIMY, an open-source table tennis ball launcher for versatile and high-fidelity trajectory generation, shown in Figure 7. AIMY is a triangular prism-shaped, standalone (not table-mounted), ball-launching robot with five independent degrees of freedom. It uses three motors for launching the balls and the same hard foam wheels as the Butterfly *AMICUS* and PowerPong *OMEGA*. The launch direction can be adjusted by a servomotor for horizontal (side-to-side) orientation and a linear motor for vertical (up and down) altitude modification (shown in Figure 8).

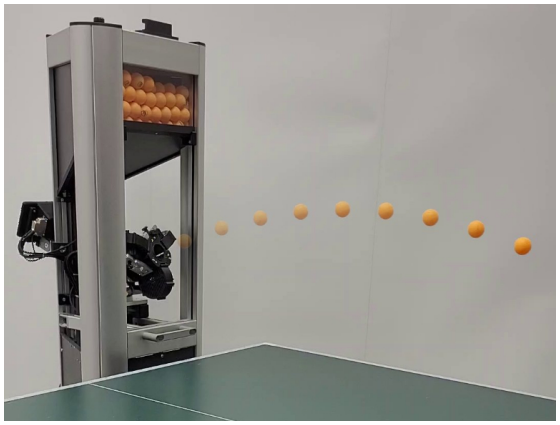


Figure 7: AIMY, an open-source table tennis ball launcher for versatile and high-fidelity trajectory generation, positioned behind a table tennis table. A sample launch trajectory of a ball is illustrated. [11]

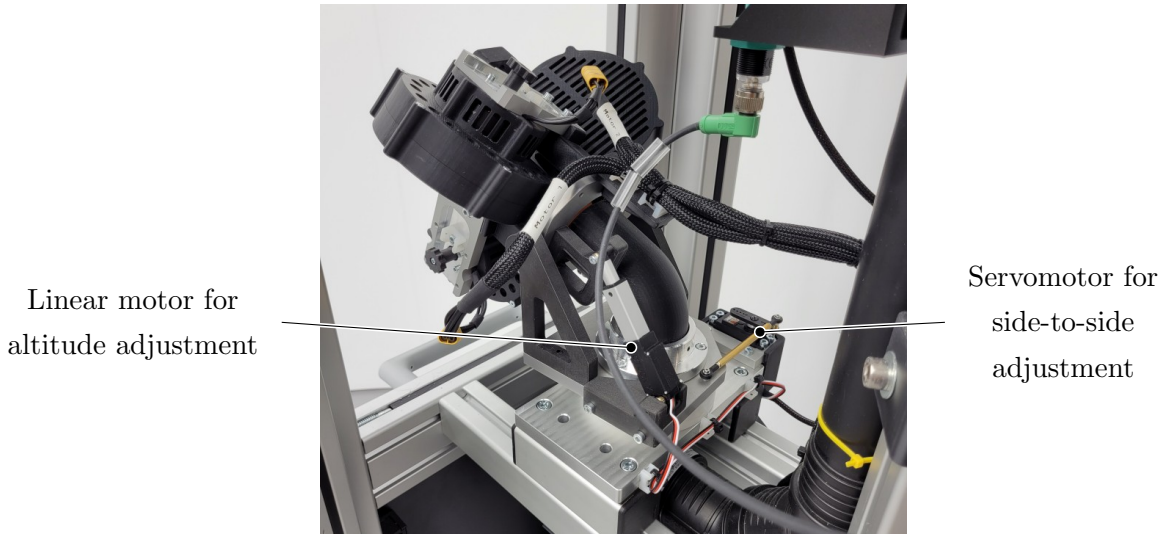


Figure 8: Cropped image of AIMY's Launch Unit with annotated locations for the linear motor and servomotor used to adjust launch direction. [17]

AIMY was developed to produce large-scale, real-world data of consistent arbitrary

ball trajectories with considerable spin. This data can be used for reinforcement learning by robots that play table tennis, such as the robots used by Abeyruwan et al. [18] and Tebbe et al. [19]. Although not designed with multiball training in mind, AIMY serves as a rare example of a well-documented open-source table tennis ball launcher that can be utilised for this purpose.

Source code for controlling AIMY [20] and data processing [21] are provided by the Intelligent Soft Robots GitHub page. Additionally, 3D models [22] of AIMY and datasets [23] for evaluating the accuracy of the robot are provided online.

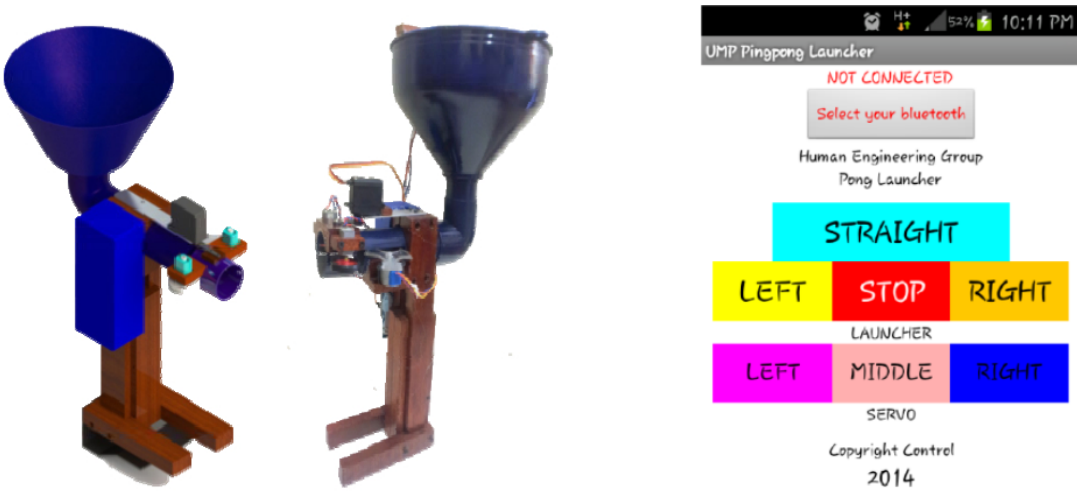
Parameter	Value
Price	1500 EUR (estimated)
Launching motors	3
Launch frequency	120 balls per minute
User interface	Tkinter GUI Python and C++ APIs over Wi-Fi or Ethernet
Weight	-

Table 4: Specifications for AIMY by Dittrich et al. [11]

Table 4 highlights AIMY's key parameters. The authors provide a list of third-party hardware which mostly consists of the electrical components used. However, it does not include the additional aluminium frame, metal fastenings, custom machined plates nor 3D printed parts. Analysing technical drawings published by the authors [22] yielded a rough estimated cost of 1500 EUR for the whole robot.

2.2.2 Robot by Ponnusamy et al.

Ponnusamy et al. [24] have published an article of a low-cost automated table tennis launcher. As can be seen in Figure 9 a, the robot uses a hopper to hold balls and two brushed DC motors to accelerate them. A servomotor pushes balls from the funnel toward the accelerating motors. Another servomotor is used to adjust the left/right direction of the launched balls. All the motors are controlled using an Arduino UNO microcontroller. A Bluetooth module is attached to the Arduino, allowing the user to control the launcher using an Android application shown in Figure 9 b.



(a) 3D model (left) and prototype (right).

(b) User interface.

Figure 9: 3D model and prototype (a) and user interface (b) of the robot by Ponnusamy et al. [24]

Parameter	Value
Price	50 USD
Launching motors	2
Launch frequency	7-8 balls per minute
User interface	Android application
Weight	-

Table 5: Specifications for the robot by Ponnusamy et al. [24]

The robot is remarkable for its low cost of 50 USD but no assembly information other than an annotated figure of a 3D model is provided by the authors. Moreover, no source code for the microcontroller's firmware or the Android application is published, rendering reproduction by third parties difficult.

2.2.3 Spin-Ninja Table Tennis Training Robot

Esmael et al. have published a paper introducing Spin-Ninja [25], a table tennis training robot (shown in Figure 10). The robot sits on top of the table in a manner similar to the JOOLA *iPONG V300* robot. Spin-Ninja uses a three-motor configuration to accelerate the ball, akin to the PowerPong and Butterfly robots. Adjustment of the launch direction is done manually. A bucket, placed on top of the wooden electronics box, houses balls ready to be pushed towards the launching wheels. The balls are pushed by a cross that is driven by a motor.

An Arduino microcontroller controls the launch and ball pusher motors and communi-

cates with a Raspberry Pi. The Raspberry Pi is connected to an LCD screen, which can be seen in the rear view in Figure 10, acting as a user interface for controlling the robot.

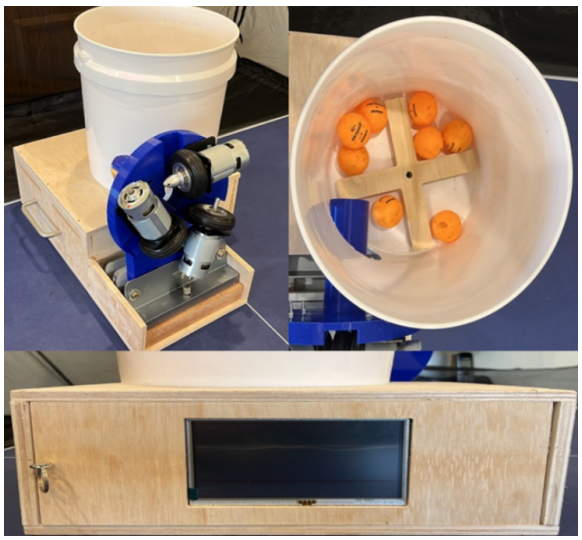


Figure 10: Spin-Ninja, placed on a table tennis table. Side view (top left), top-down view (top right), and rear view (bottom). [25]

Parameter	Value
Price	300 USD
Launching motors	3
Launch frequency	-
User interface	Tkinter GUI
Weight	-

Table 6: Specifications for Spin-Ninja. [25]

The authors estimate a material cost of 300 USD to manufacture the robot but no instructions on how to manufacture or assemble the components are published. Similarly to the robot developed by Ponnusamy et al., no source code for programs running on the Raspberry Pi or the Arduino is available.

3 Requirements

The aim of this thesis was to develop a table tennis ball launcher robot suitable for the use of multiball training. The following list of requirements was compiled prior to Rhea's development to constrain design decisions.

1. **Ease of Manufacture**: Manufacture and assembly of Rhea must be achievable with common hobbyist tools. This includes a 3D printer capable of printing PLA (*Polylactic acid*) with a minimum 20x20x20 cm build volume, soldering equipment (soldering station, solder, flux), hot glue gun, and screwdrivers.

2. **Material and Component Accessibility**: The materials and components required must be as generic as possible, allowing for substitution if necessary. For example, electronic components should be simultaneously available from different vendors (e.g. Farnell, DigiKey, Mouser).

3. **Affordability**: The total material and component costs must not exceed 250 EUR. This includes but is not limited to 3D printer filament, electronic components, and fastenings (bearings, screws, bolts, nuts).

4. **Targeting and Accuracy**: The launcher must be able to target 80% of the opposite side of the board and reliably hit a target area with a standard deviation of $\sigma = 5$ cm.

5. **Replicability**: The BOM, assembly instructions, 3D models, firmware, and software must be open sourced to permit the replication and customisation of the robot.

4 Development of Rhea's mechanical components

Rhea consists of several mechanical components, which can be seen in Figure 11. Most components went through multiple revisions. To comply with the first requirement the printing material was limited to PLA and the mechanical components were split into several smaller details. PLA is easy to 3D print, inexpensive, and does not require an enclosure [26]. The largest model required a printing volume of 20x20x14 cm.

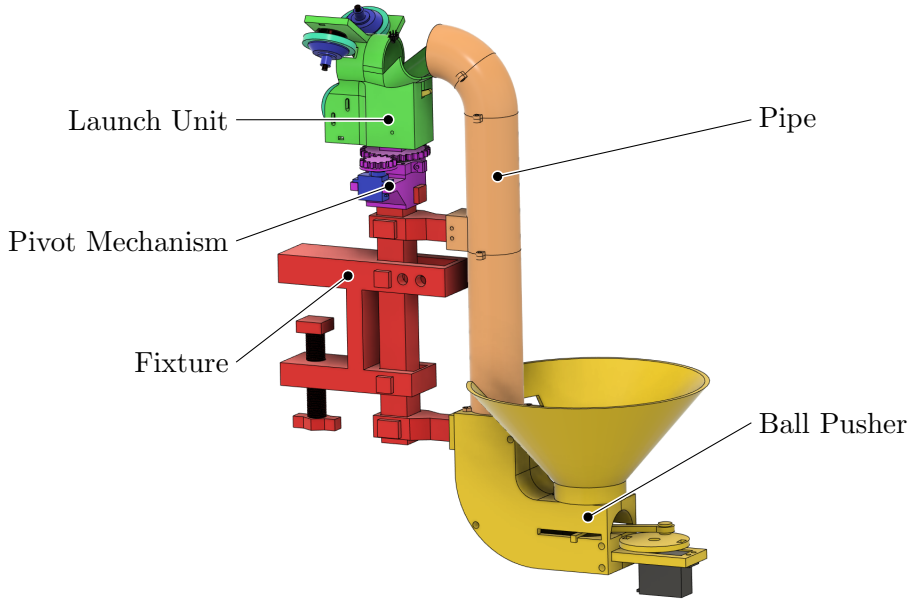


Figure 11: 3D model of Rhea's final version with annotated components.

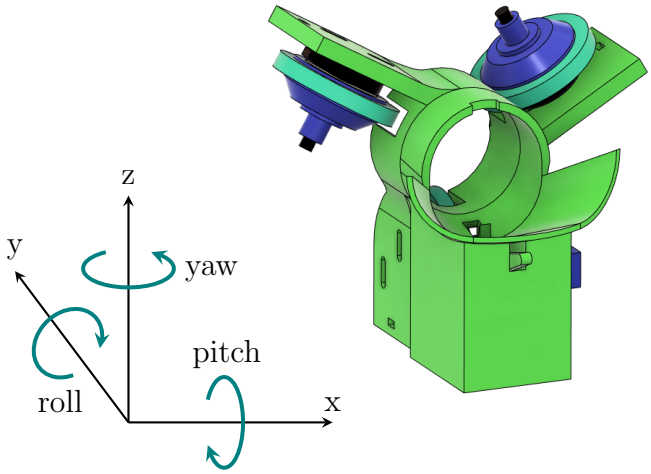


Figure 12: 3D model of the Launch Unit with Cartesian axes.

It is important to clarify the coordinate system and axes of rotation for the upcoming subsections. With respect to the Launch Unit, Figure 12 establishes the three principal directions, along with their respective axes of rotation. A table tennis ball will exit the Launch Unit in the positive y direction.

The following subsections cover the functionality and development of each component. The subsections are ordered in a way that follows the trajectory of the ball through the whole system, starting from the Ball Pusher.

4.1 Ball Pusher

The Ball Pusher is the component responsible for delivering the balls to the Launch Unit. It is mounted lower than the table to minimise the robot's footprint above the table for aesthetic reasons. Moreover, this allows for balls successfully returned by the player to be collected and reused. A slider-crank linkage, driven by a continuous servomotor (*Parallax Continuous Rotation Servo #900-00008* [27]), was developed to push the balls upwards through the Pipe.

4.1.1 First revision

Figure 13 depicts a section analysis of the Ball Pusher's first revision. An angled input rail holds balls ready to be pushed upwards. The servomotor's output shaft is connected to a crank. In turn, the crank is connected to a connecting rod that is further connected to a piston. The servomotor is secured to a holder using M3 bolts and nuts. Finally, the holder is friction-pressed into a slot in the housing. When the piston is at the bottom, a ball can move from the input rail into the housing. As the crank rotates, the piston reciprocates, pushing the ball upwards.

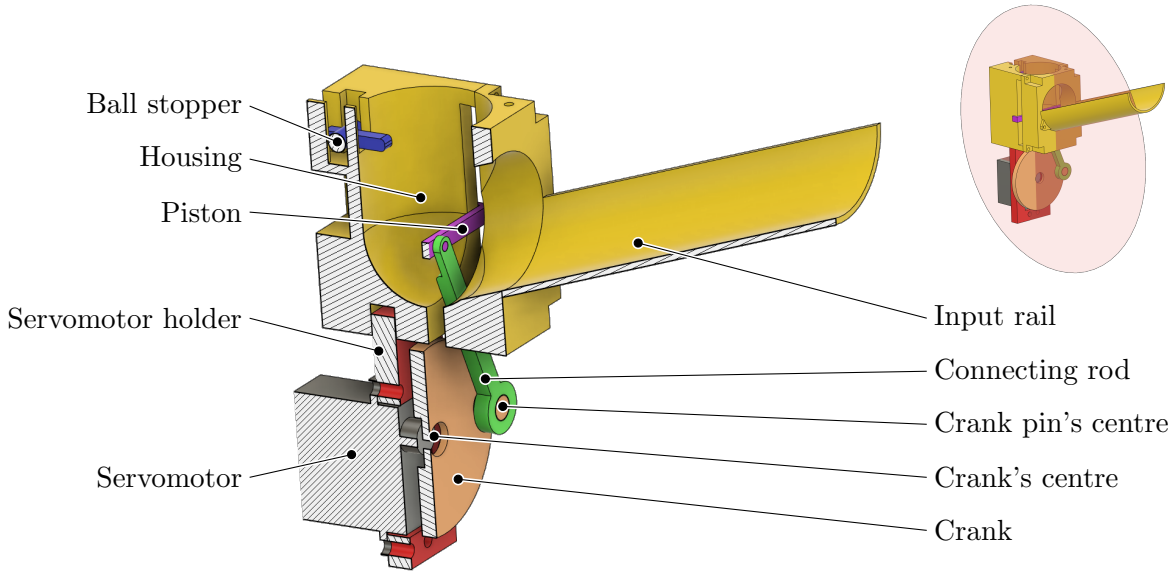


Figure 13: Annotated section analysis of the Ball Pusher's first revision. The crank, connecting rod and piston make up a slider-crank linkage. Balls fall from the input rail to the housing and are pushed upwards into the Pipe.

To stop balls from falling back down, a small rubber band loaded ball stopper acts as

a non-return valve. The throw, or the distance from the crank's centre to the crank pin's centre, dictates the stroke length of the piston. A stroke length of 50 mm was chosen, 40 mm for the ball and 10 mm extra in order to not constrain the lever's positioning. Figure 14 illustrates how the slider-crank linkage and ball stopper work. The servomotor has a maximum speed of 50 revolutions per minute [27], enabling a maximum ball launching rate of 50 balls per minute.

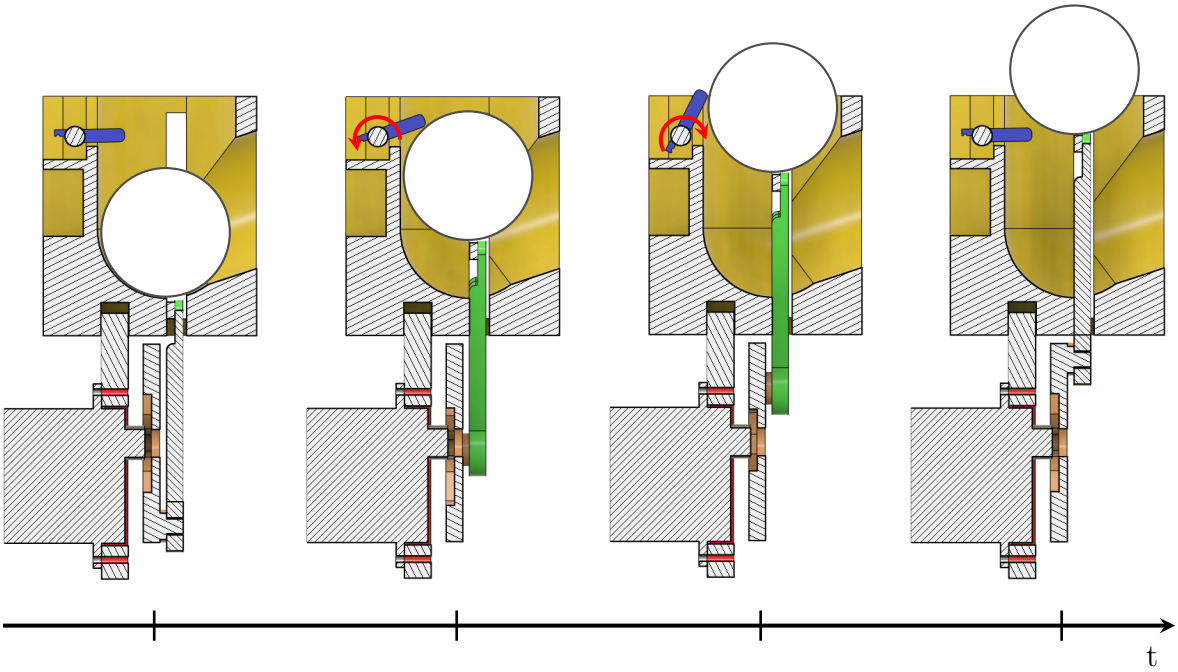


Figure 14: Section analyses of the Ball Pusher's first revision illustrating a ball being pushed upwards into the Pipe. The movement of the piston (green) and ball stopper (blue) is illustrated in four consecutive steps. A rubber band forces the stopper to the position illustrated by the first and last steps by pulling upwards on a small tab on the left side of the stopper, effectively preventing balls from falling back down.

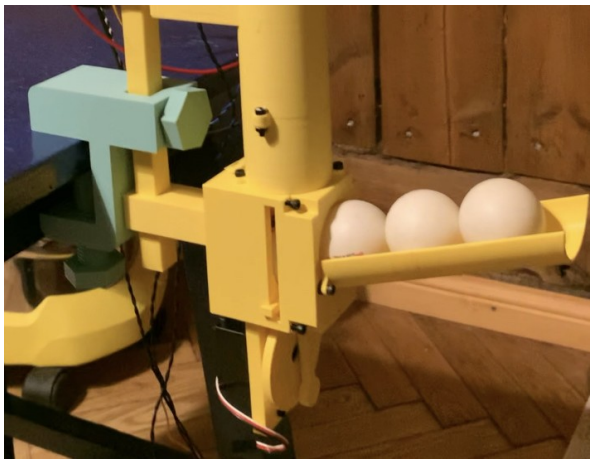


Figure 15: Prototype of Rhea using the first revision of the Ball Pusher. Table tennis balls ready to be pushed upwards are placed on an angled rail.

Figure 15 depicts the first revision of the Ball Pusher being tested. During testing of the first revision's printed prototype, it became apparent that a vertically moving piston was not ideal. A ball being pushed upwards would get stuck between the piston and the housing. This issue became more prevalent when the Ball Pusher was slightly tilted and the ball would roll backwards, out of the housing. Additionally, the lack of an integrated encoder in the servomotor meant that the angular displacement of the crank was unknown.

4.1.2 Second revision

The issues mentioned in the last subsection were addressed in the Ball Pusher's second (final) revision, depicted in Figure 16. The slider-crank linkage was rotated 90° to a horizontal configuration and a 90° bend was added to the housing to redirect the balls upwards. A funnel was added for directing balls into the housing.

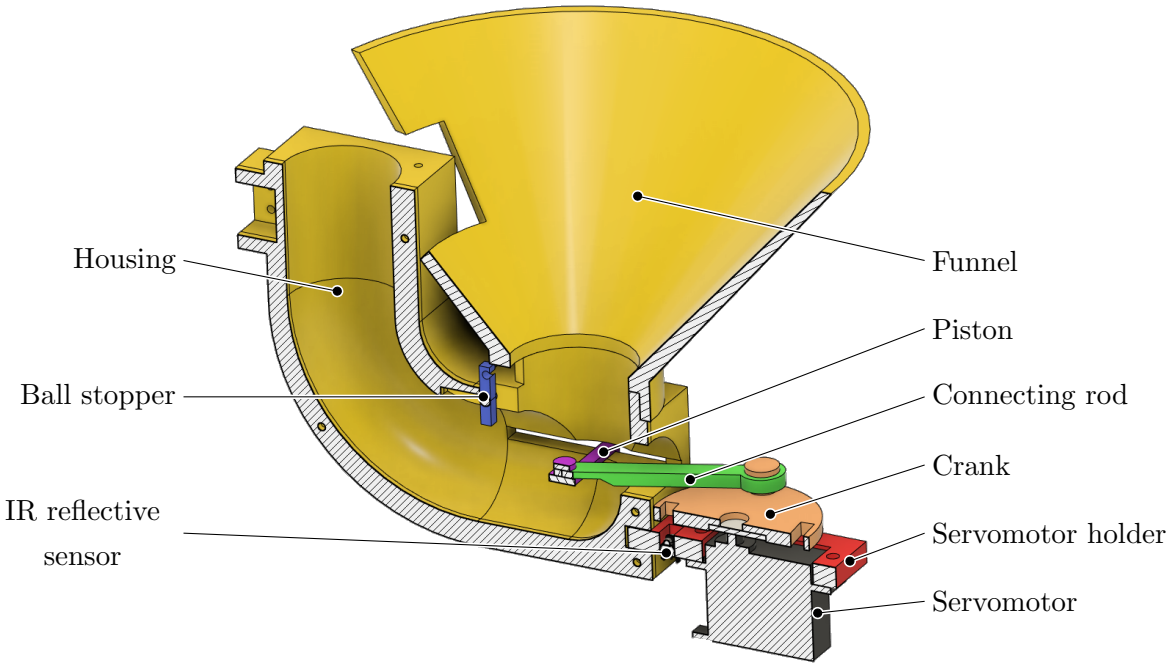


Figure 16: Section analysis of the Ball Pusher's second revision with annotated components.

The servomotor holder was modified to make room for an infrared reflective sensor (*Vishay TCRT5000* [28]) that would serve as an encoder for the crank. Combined with two newly added rectangular slots in the crank, the infrared reflective sensor could be used to determine if the servomotor has completed a full revolution. This removes the possibility of pushing a new ball into the Launch Unit before the previous one has been launched.

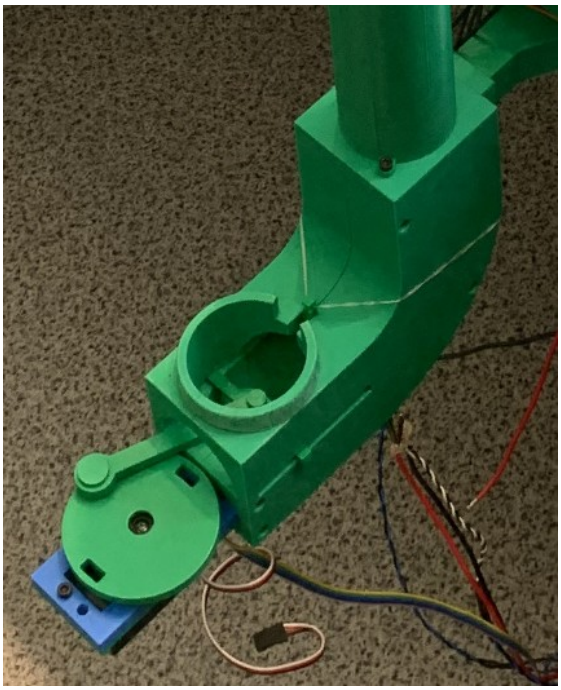


Figure 17: The second revision of the Ball Pusher. A clear rubber band is used to force the ball stopper into an upright position. A funnel (not depicted) can be pressure fit on top of the large opening to direct balls into the housing.

Figure 17 shows the placement of the rubber band that forces the ball stopper upright. Additionally, the large mounting hole for the funnel can be seen top of the housing.

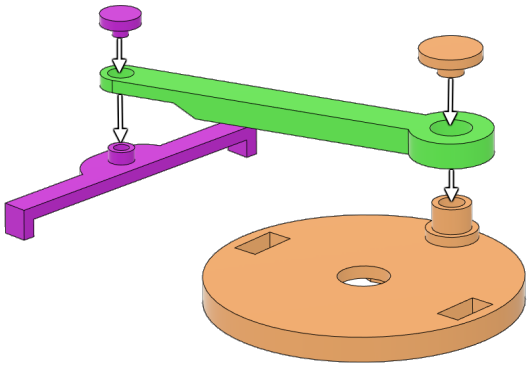
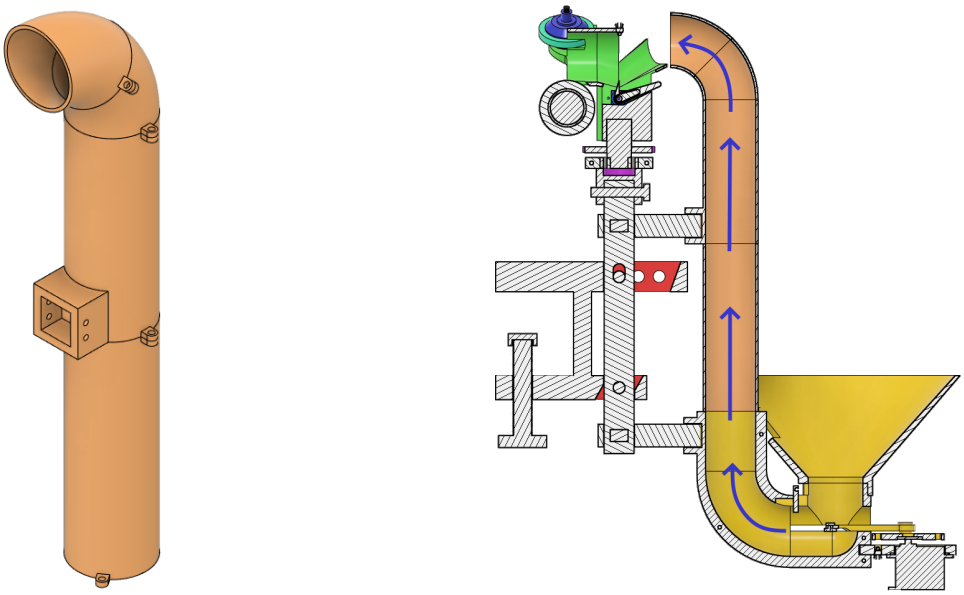


Figure 18: Exploded view of the Ball Pusher's slider-crank linkage. Two retaining pins prevent the connecting rod from coming loose.

Finally, two retaining pins to the slider-crank linkage were added: one for the piston and one for the crank (Figure 18). These pins help prevent the connecting rod from coming loose when the crank is rotating.

4.1.3 Pipe

Balls exiting the Ball Pusher move upwards through the Pipe (Figure 19) and are delivered to the Launch Unit. The Pipe consists of four 3D printed parts, secured together using M3 bolts and nuts. The lower section connects to the Ball Pusher and the middle section has a slot for securing the Pipe to the Fixture. The two top 45° sections form a 90° junction that redirects the balls so that they move horizontally.



(a) 3D model. (b) Section analysis.

Figure 19: 3D model (a) and section analysis (b) of the Pipe. Blue arrows depict the trajectory of a ball from the Ball Pusher, through the Pipe and into the Launch Unit.

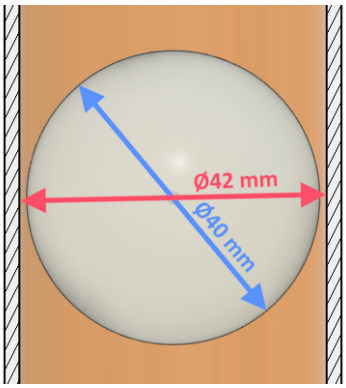


Figure 20: Section analysis of the Pipe illustrating its slightly larger inner diameter of 42 mm allowing a 40 mm table tennis ball to freely move upwards.

Figure 20 shows a table tennis ball inside the Pipe, highlighting its slightly larger 42 mm inner diameter. This diameter is also used in the Launch Unit, elaborated on in the next section.

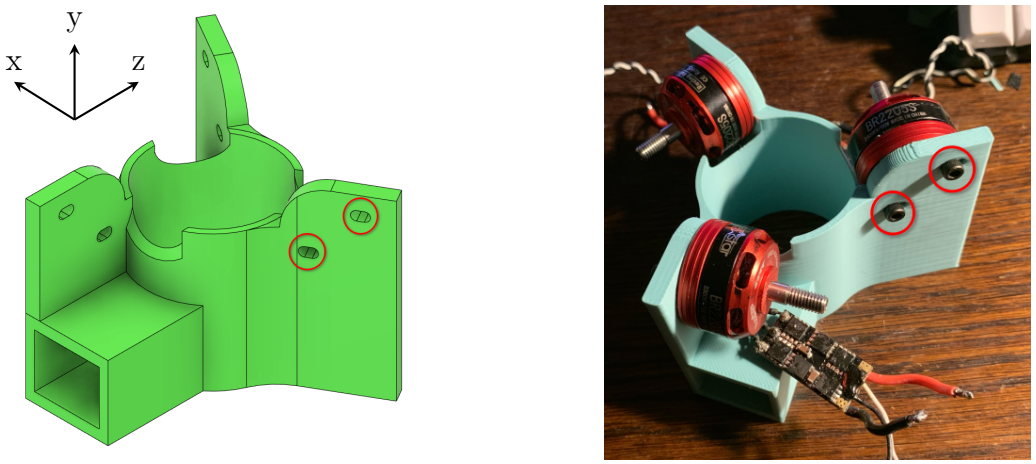
4.2 Launch Unit

In order to launch a ball, it must be accelerated by an external force. In the context of table tennis robots, this is achieved by briefly making contact with a rotating surface fixed to a motor. The choice of material for the surface must provide sufficient grip whilst not damaging the balls. This method is commonly used instead of a linear launching mechanism due to the relative fragility of the balls. It also avoids having to deal with the sudden forces that a linear accelerator would produce when starting or stopping.

The Launch Unit of Rhea is responsible for receiving balls from the Pipe and accelerating them. In competitive table tennis, applying different types of spin to the ball plays an important role. Most notably, applying rotation along the x-axis (topspin and backspin) and rotation along the z-axis (sidespin). Rotation along the y-axis (cork spin) is uncommon because it requires the player to brush the bottom of the ball along the x-axis (refer to Figure 12) with their racket while simultaneously applying momentum along the y-axis at the same time. Spin applied to the ball causes it to undergo a lift force produced by the Magnus effect, providing another way for a player to influence the trajectory of a ball. To facilitate the generation of sufficient top, back, and sidespin, a three-motor setup was implemented in Rhea's Launch Unit.

4.2.1 First revision

The first revision (shown in Figure 21) of the Launch Unit featured a hollow cylinder with an internal diameter of 42 mm and three "blades" spaced 120° apart for mounting the motors. The 42 mm bore diameter was chosen to fit a standard 40 mm table tennis ball with a loose fit, akin to the Pipe's internal diameter.



(a) 3D model. (b) Prototype with motors attached.

Figure 21: 3D model (a) and prototype (b) of the Launch Unit's first revision. Red circles in both subfigures indicate slots used for mounting a motor to a single blade.

Three-phase BLDC drone motors (*Racerstar BR2205S 2300KV* [29]) were used for accelerating the ball. Such drone motors are readily available in online hobbyist radio-control stores and they feature standardised threaded holes for mounting. The motors used for Rhea have two M3 threads spaced 19 mm apart.

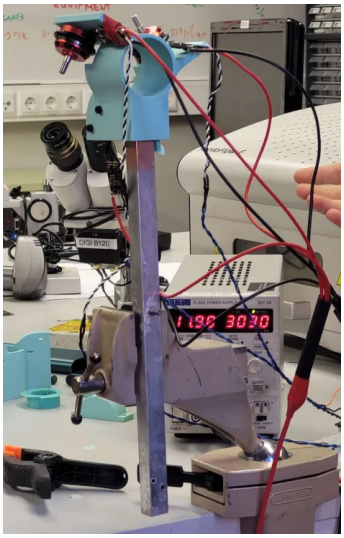


Figure 22: First revision of the Launch Unit fixed to a table using a vice and a metal rod. O-rings stretched around the motors are used for gripping the ball.

Cuts were made to the cylinder for all three motor positions to accommodate the aforementioned elastic material that would grip the ball. For mounting the motors, slots were created in each blade (refer to Figure 21). These slots allow for the adjustment of the contact point between the elastic material and the ball. In addition, a rectangular hole was created to allow a rod to be connected to the Launch Unit, which could then be fixed to a table using a vice as can be seen in Figure 22.

Powering the motors was done by connecting each motor's ESC (*Racerstar MS Series 35A* [30]) to a 12 V DC power supply. Akin to drone motors, ESCs can be obtained from radio-control stores and they do not need to be rated for high (10+ A) current. Testing showed that the motors drew 1 A of current at 15% of maximum throttle, which was enough to launch a ball across the length of the table tennis table. An Espressif ESP32 development board (*DOIT ESP32 DevKit V1* [1]) was connected to the ESCs' signal wires via a breadboard to regulate the speed of the motors using PWM.

O-rings stretched around the motors' outer walls were used as the first elastic material that would make contact with the ball. These proved to be unsuitable due to their hardness causing them to grip the ball insufficiently and leave markings on its surface. An alternative solution to o-rings was implemented in the next revision of the Launch Unit.

[1]Official documentation/product page for the module could not be found. Several resources for the module are available online such as the one by Zerynth [31].

4.2.2 Second revision

The second revision of the Launch Unit (depicted in Figure 23) saw the addition of a chamber, responsible for receiving balls from the Pipe and buffering them before launch. The chamber featured a curved surface allowing the Launch Unit to rotate whilst being able to receive incoming balls from the stationary Pipe. A notch at the top of the chamber was made for mounting an infrared reflective sensor used for detecting the presence of a ball. The o-rings were replaced with polysiloxane traction wheels, mounted on motor guards which were fixed to the motors. The bottom motor mount's slots were rotated 90° clockwise along the x-axis so that its three wire connections would face downwards. Finally, a small slot was added at the end of each motor mount to secure the motors' wires using a cable tie. The upcoming paragraphs will cover some of the additions in depth.

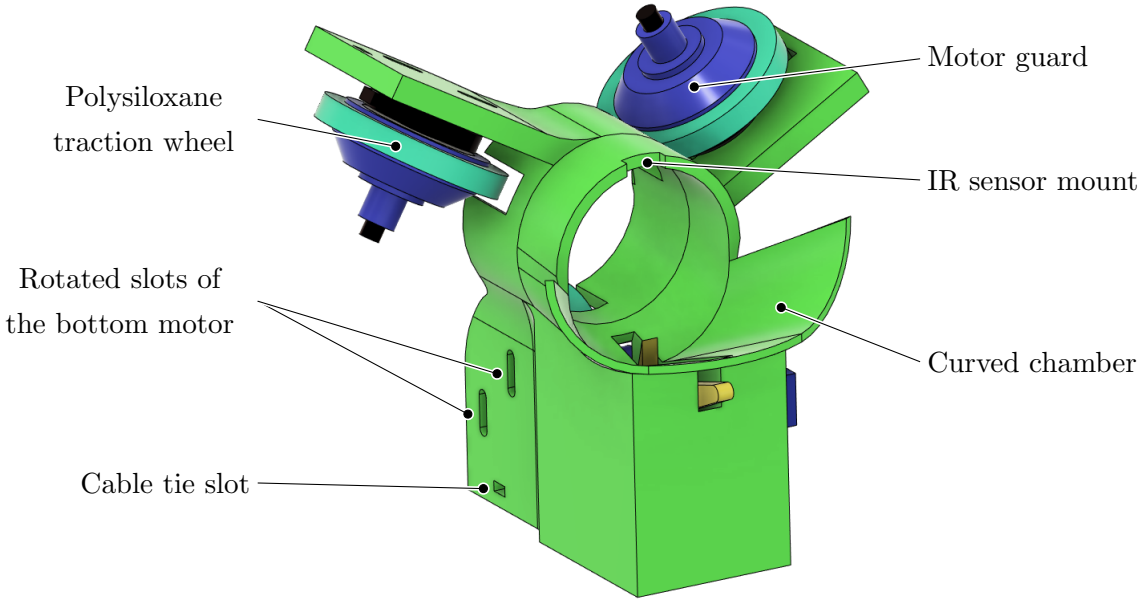


Figure 23: Annotated 3D model of the Launch Unit's second revision highlighting the additions and changes from the first revision.

A ball, exiting the Pipe and entering the Launch Unit, inherently gains both random linear and angular velocities before it reaches the motors. It is this inherent unpredictability in the ball's motion that necessitates the use of an intermittent stop. Without it, the repeatability of consecutive launches could be compromised. Removing the unpredictability was achieved by incorporating a servomotor (*TowerPro SG90* [32]) rotating an L-shaped hook, acting as both a stopper for an incoming ball and a pusher for a received one.

The functionality of the stopper can be seen in Figure 24. Three section analyses of the Launch Unit are shown, portraying the subsequent steps of a ball moving through the chamber. Initially, the hook is in position to stop an incoming ball. A downwards-facing infrared reflective sensor (*Vishay TCRT5000* [28]) enables the Launch Unit to detect the presence of a ball. After a ball is detected, the hook rotates counterclockwise, as indicated

by the red arrow, causing the longer arm of the hook to push the ball further towards the motors. The ball is pushed at a constant rate towards the motors until it makes contact with the rotating polysiloxane traction wheels, which causes the ball to rapidly accelerate and exit the Launch Unit.

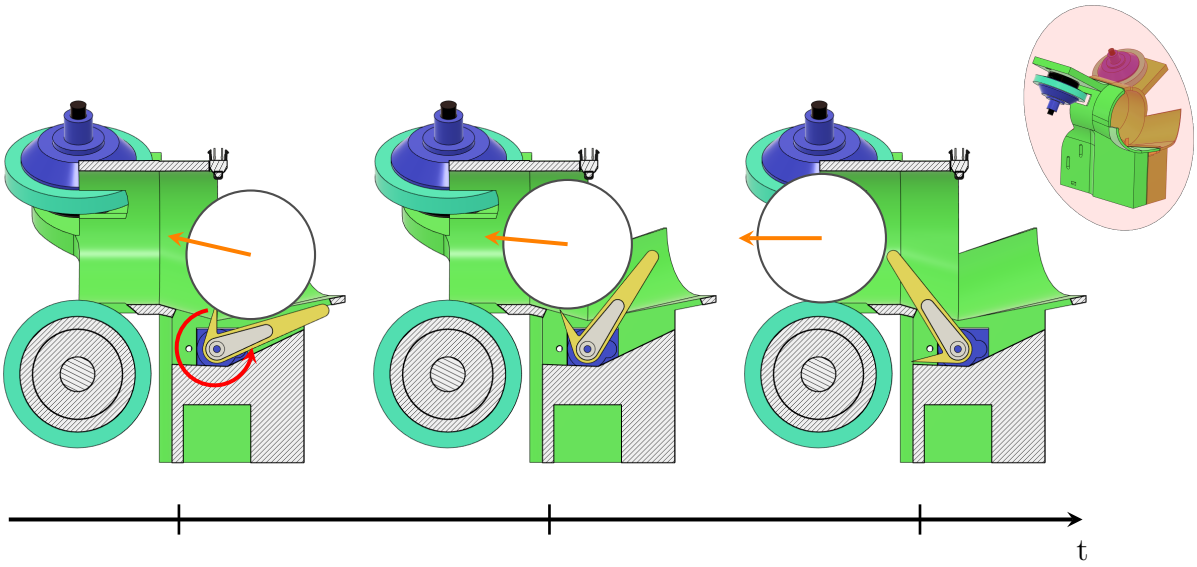


Figure 24: Section analyses of the Launch Unit's second revision. Three steps illustrate how a hook (yellow), driven by a servomotor, buffers a ball received from the Pipe. A downwards-facing infrared sensor is used to detect the presence of a ball. When a ball is detected, the servomotor rotates the hook counterclockwise, pushing the ball towards the three launching motors.

The unsuitable o-rings led to experimentation with different silicone-based materials, the first of which was caulk, commonly used for sealing joints and gaps. Negative moulds (hereinafter moulds) in the shape of a cylinder were modelled after the motors and 3D printed to be filled with silicone caulk. After three iterations of moulds, caulk proved inapt due to it adhering to the PLA and rendering its removal from the mould difficult.

The next trials, done with pourable two-component polysiloxanes, resulted in easy removal from the moulds and were available with different hardness ratings. The hardness of such polysiloxanes is usually provided in the Shore A scale with 0 being the softest (e.g. very soft gel pads) and 100 being the hardest (e.g. very hard polyurethane bushings). Two polysiloxanes, with rated Shore A hardness of 5 (*Polistep PLS-505* [33]) and 22 (*Zhermack Elite Double 22* [34]), were obtained. The mould underwent three further iterations to ultimately yield a sixth, final mould (shown in Figure 25) that was easy to print and use. All five initial iterations of the mould and experiments with silicones can be found in Appendix B.

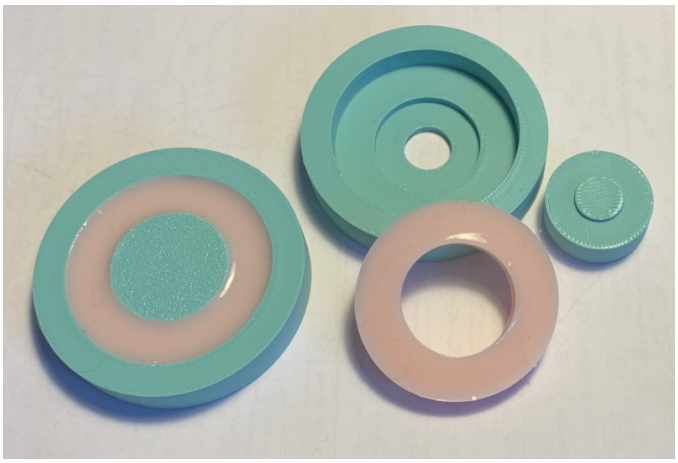


Figure 25: Final version of the moulds and respective positives made using Shore A 5 polysiloxane used as traction wheels for launching the ball.

At first, the polysiloxane traction wheels were simply stretched around the motors' outer housing. However, during testing of the Launch Unit's second revision, issues arose regarding the centrifugal force the wheels experience. At 8% of throttle, the wheels would expand and detach from the motors' housings, potentially dangerously flying away. Furthermore, mounting a wheel to be perfectly aligned with a motor's axis of rotation proved difficult.

(a) First iteration.

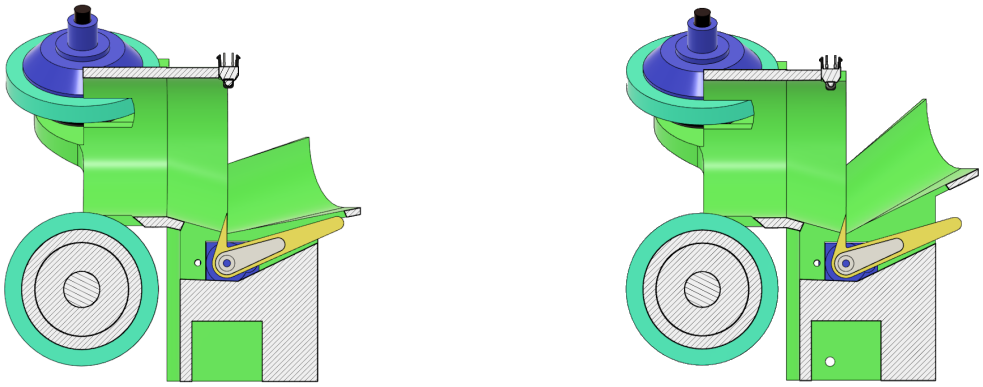
(b) Second iteration.

Figure 26: Two iterations of the motor guard.

To remedy these issues a motor guard was designed as depicted in Figure 26. The guards were secured to the motors by flange nuts that came with the motors. In addition, the guard increased the effective radius of the motor housing, meaning the polysiloxane would be stretched out further than before, increasing the expansion/detachment threshold speed to 20% of throttle. Testing showed that balls launched with 15% throttle would overshoot the table. As a result, the 20% threshold is never realistically reached, eliminating the detachment problem.

4.2.3 Third revision

Minor flaws in the Launch Unit's second revision were observed during testing. These flaws were related to how a ball passes through the chamber and how the Launch Unit connects to the Pivot Mechanism. The third and final revision of the Launch Unit remedied these issues.



(a) Second revision. (b) Third revision.

Figure 27: Section analyses of the Launch Unit's second (a) and third (b) revision. The third revision's chamber features a steeper angle for receiving a ball from the Pipe. This prevents the ball from rolling back (to the right) when the pitch of Rhea is adjusted and the Launch Unit rotates clockwise along the x-axis. Two holes (one depicted) at the bottom allow for securing the third revision to the Pivot Mechanism using M3 bolts.

Figure 27 depicts the second and third revision of the Launch Unit for visual comparison. The curved surface of the chamber was made steeper to prevent balls from rolling back towards the Pipe when Rhea's pitch is adjusted. The infrared reflective sensor mount was moved towards the launching motors for clearance.

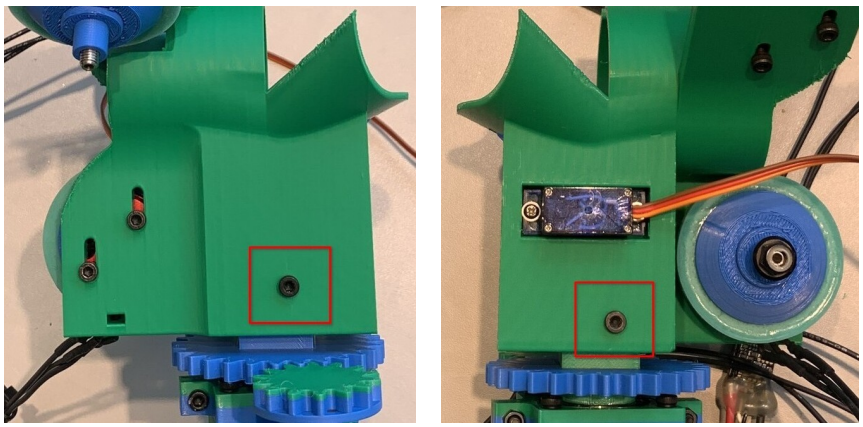
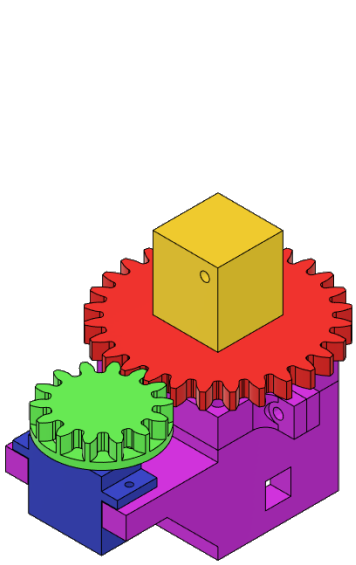


Figure 28: Two side views of thee Launch Unit's third, final revision. Red squares highlight how two M3 bolts are used to secure the Launch Unit to the Pivot Mechanism.

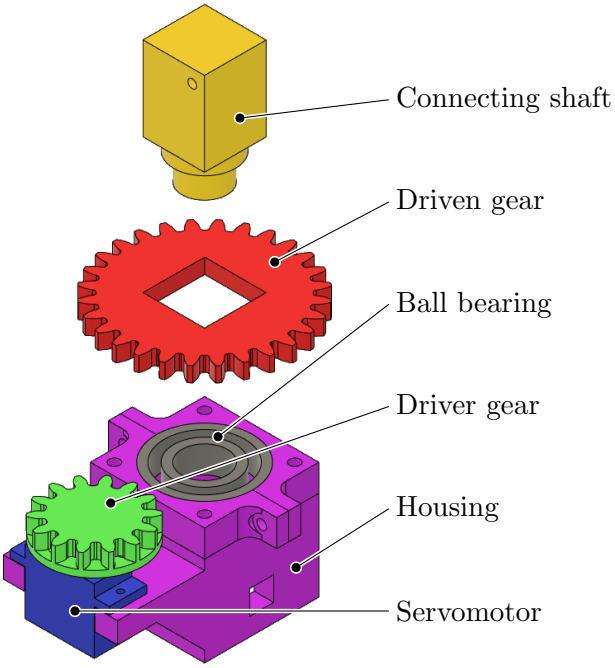
The bottom rectangular hole, previously used to mount the Launch Unit to a rod, was now used for connecting the Launch Unit to the Pivot Mechanism (Figure 28).

4.3 Pivot Mechanism

The Pivot Mechanism allows for the adjustment of the yaw angle (refer to Figure 12) of the Launch Unit. As can be seen in Figure 29, this was implemented with the use of a 15x32x9 metal ball bearing connecting the stationary Pivot Mechanism to the Launch Unit, enabling it to rotate independently from the rest of Rhea. A servomotor (*TowerPro SG90* [32]) rotates a small gear driving a bigger gear which is connected to a shaft. Figure 30 illustrates how the Pivot Mechanism enables Rhea to adjust its launch direction.



(a) 3D model.



(b) Exploded 3D model with annotations.

Figure 29: Assembled (a) and exploded (b) 3D models of the Pivot Mechanism. A bearing is pressure fit into the housing, allowing for its inner ring to freely rotate. The driven gear is secured to the connecting shaft which is pressure fit into the bearing's inner ring. A servomotor, secured to the housing, rotates the driver gear which causes the bigger gear (red) to rotate. The connecting shaft is fitted inside the Launch Unit.

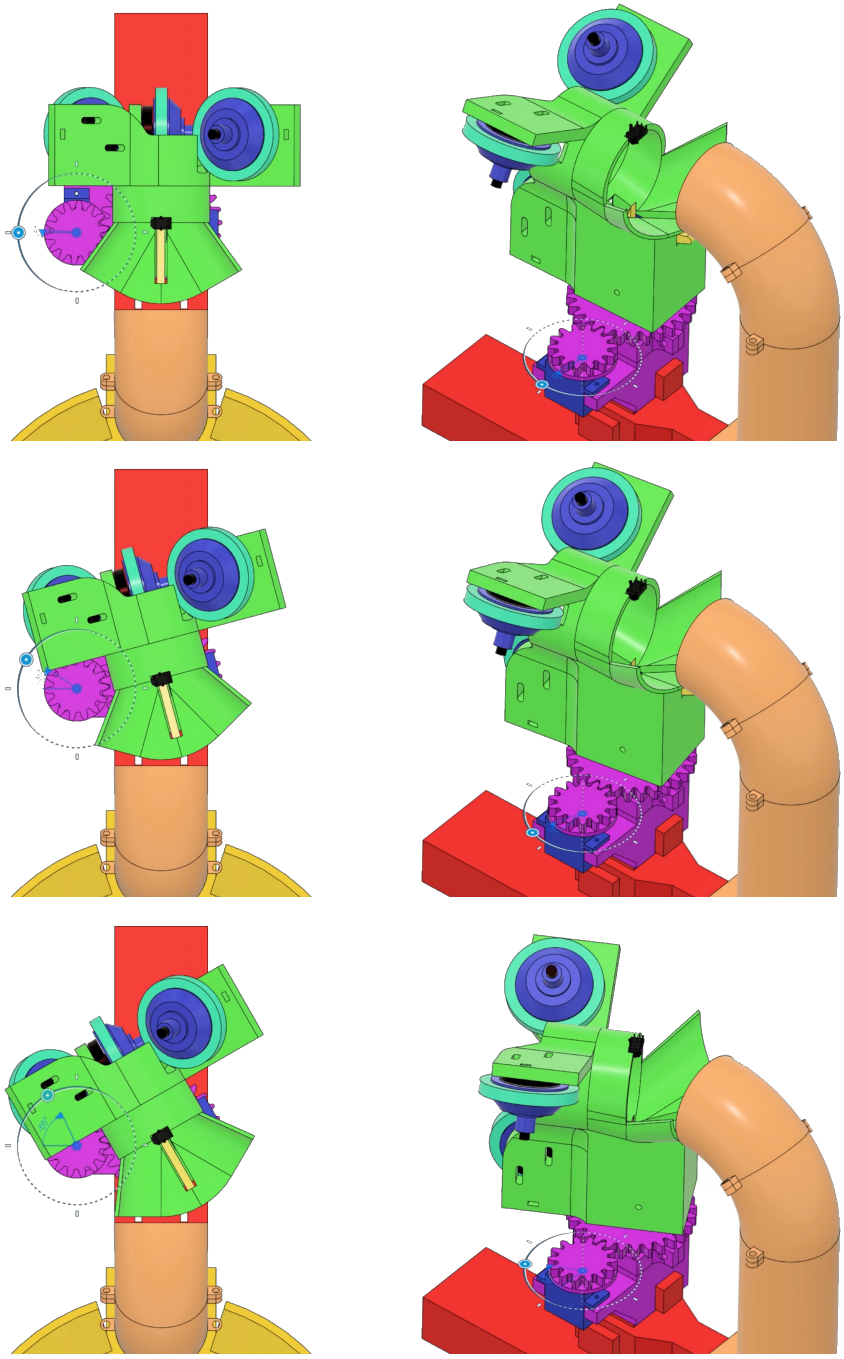


Figure 30: Three configurations illustrating the Pivot Mechanism rotating the Launch Unit to one side in order to alter a launch's yaw angle. The left column depicts a top-down view whilst the right column depicts a view from the side. The Pivot Mechanism is capable of rotating the Launch Unit 30° in either direction.

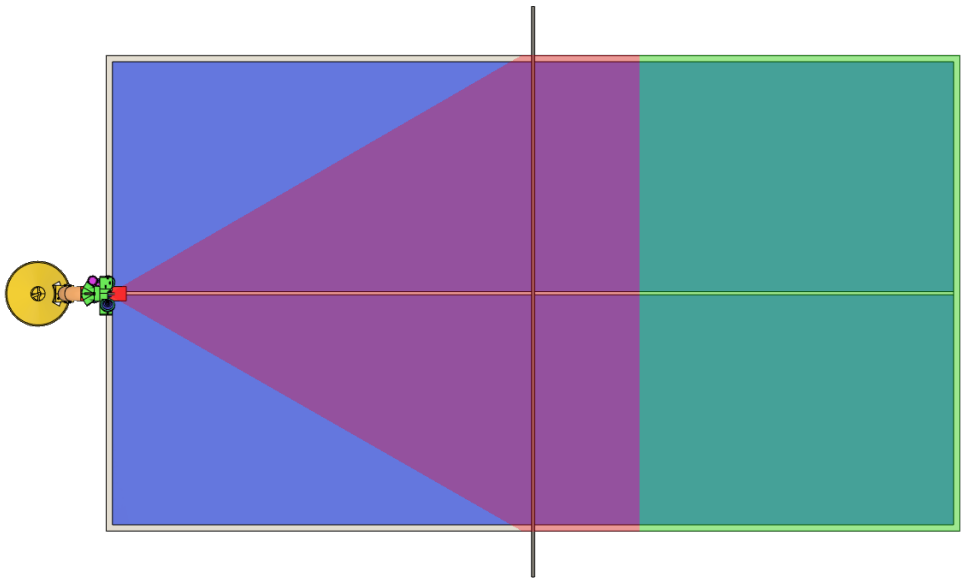


Figure 31: Top-down view of Rhea mounted on a table tennis table. The highlighted purple area shows the theoretical targeting area enabled by the Pivot Mechanism. The highlighted green area shows the effective targeting area of 75% when taking into account the height of the net.

A yaw angle adjustment range of 60° was chosen to enable targeting the whole side of the table opposite the robot. Since the SG90 servomotor can only rotate 90° in both directions [35] a 1:2 gear reduction ratio was chosen that would cover the 60° range whilst providing sufficient torque. Figure 31 illustrates how Rhea can effectively target 75% of the opposite side of the table as a result of the rotating Launch Unit.

4.4 Fixture

All mechanical components of Rhea connect to the Fixture which can be mounted on a table tennis table using a clamping mechanism. Development of the Fixture occurred in parallel with the other components as it required constant changes. Early prototypes (such as the one depicted in Figure 32) featured a clamping mechanism and a vertical rod which could be secured using a 3D printed bolt. Rectangular slabs pressure fit around the rod would be used to secure components like the Ball Pusher and the Pipe.

The final version of the Fixture used a set of 3D printed dowels (Figure 33) for securing the various components. The dowels featured a square end to enable their easier removal.

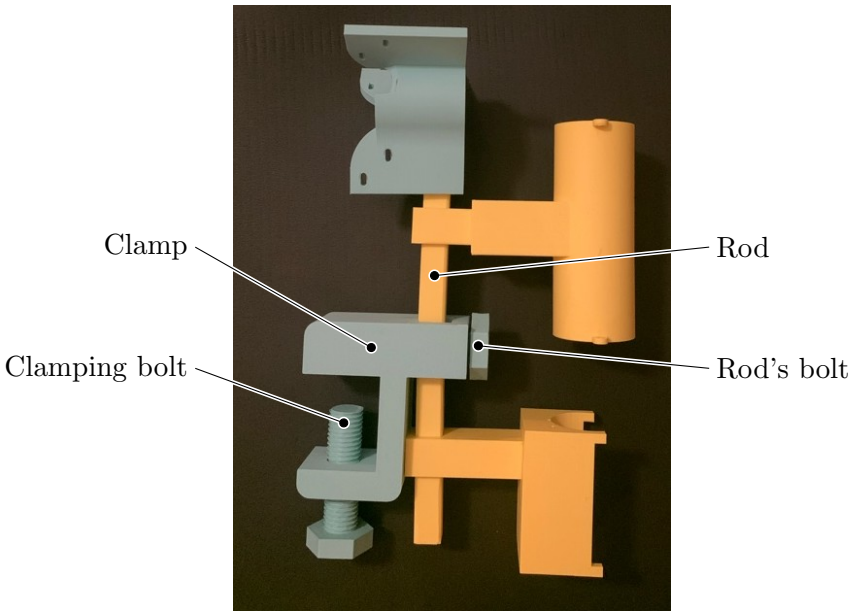


Figure 32: Prototype of the Fixture with its components annotated. Prototypes of other components are shown to illustrate how they would connect to the Fixture.

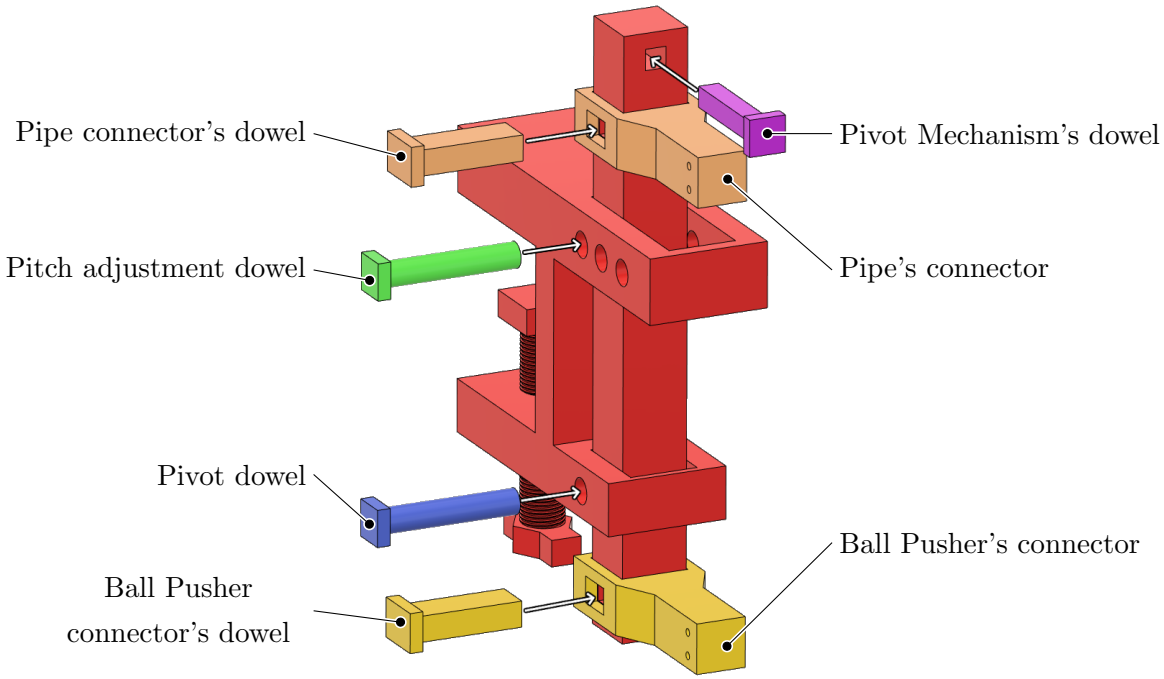


Figure 33: Annotated 3D model of the Fixture. 3D printed dowels are used for securing various components. The dowels are shown in an extracted state with arrows depicting their mounting direction. The pivot and pitch adjustment dowels are rounded.

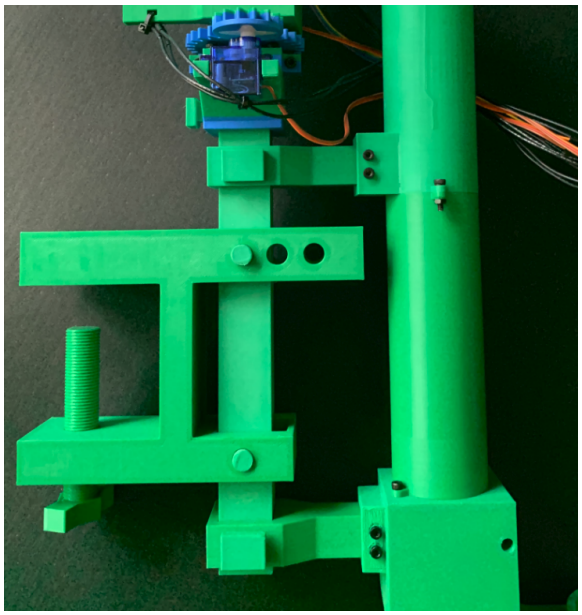
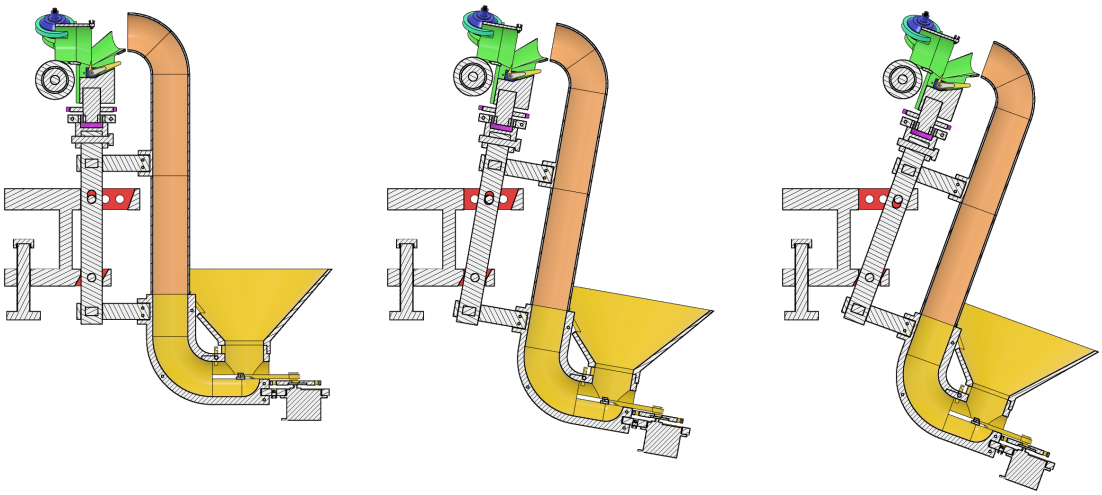


Figure 34: Side view of the Fixture demonstrating how Rhea's components are connected together. The Pipe and the Ball Pusher are secured to their respective Fixture connectors using M3 bolts.

Two connectors for securing the Ball Pusher and the Pipe were designed. These connectors would connect to the rod using the aforementioned dowels and be secured to their respective components using M3 bolts as can be seen in Figure 34.



(a) 0° altitude. (b) 10° altitude. (c) 20° altitude.

Figure 35: Section analyses of Rhea illustrating how the pitch can be manually regulated by moving the pitch adjustment dowel into one of three slots in the Fixture. The pitch can be adjusted to 0, 10 or 20 degrees.

Combined with three circular holes added to the clamp, two rounded dowels enabled Rhea's launch angle to be manually adjusted (Figure 35). A greater launch angle allows for a more curved ball trajectory. This permits the ball to be launched at a slower speed, which is more optimal for less fast-paced training sessions.

5 Overview of Rhea's electronic system

Rhea consists of several software-controlled active components. An ESP32 module (*DOIT ESP32 DevKit V1*) was chosen to control components and communicate with a computer. The ESP32 module is relatively inexpensive (~5 EUR) and offers all the peripherals required to control the various components. The module exposes the ESP32's pins via standard 2.54 mm pin-headers and a micro USB port, which enables wiring the components with relative ease. Figure 36 depicts a schematic of all the components and their connections used in Rhea.

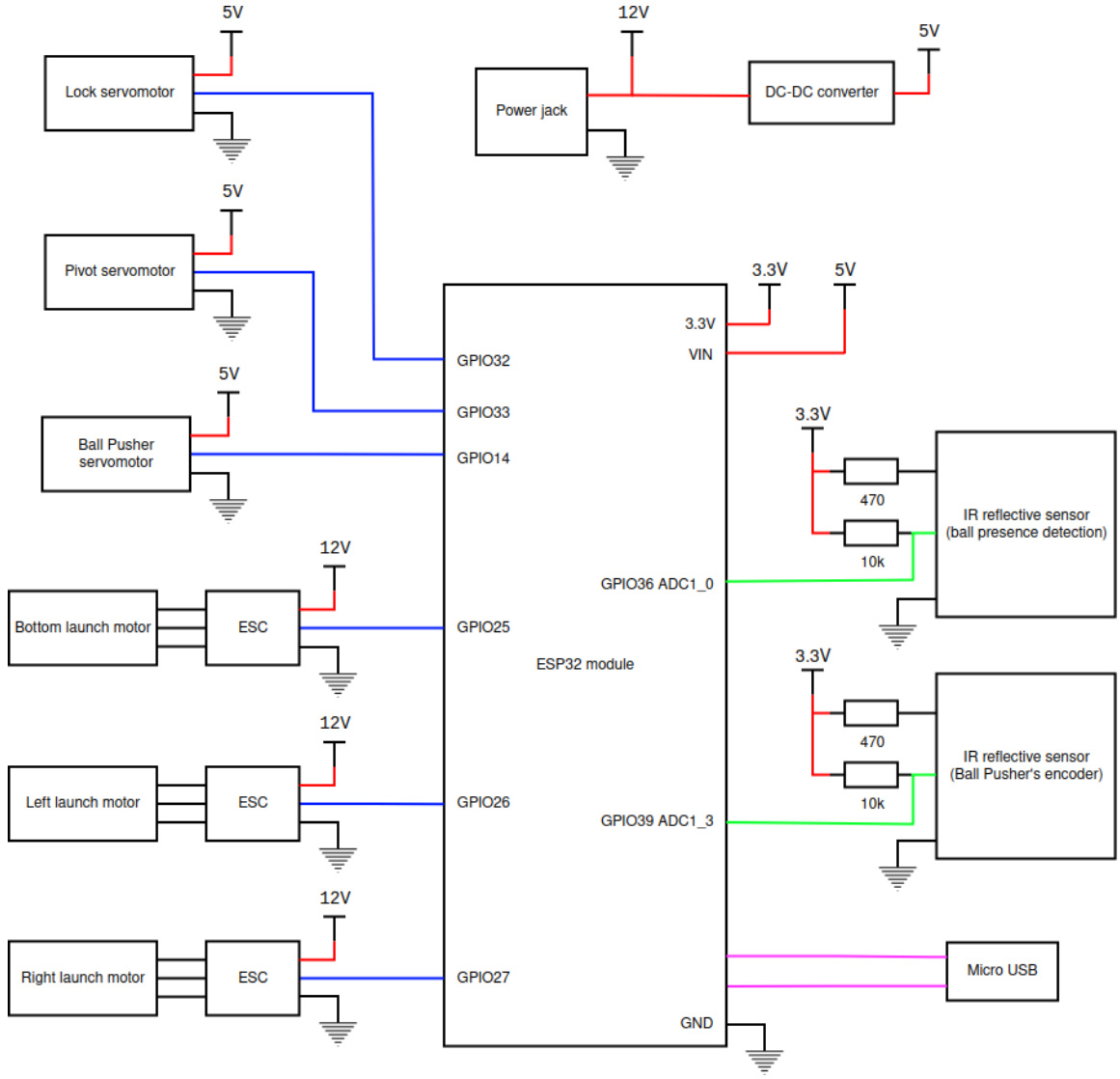


Figure 36: Schematic of Rhea's electronic components and their connections. Blue connections represent PWM signals, green connections analog signals, and purple connections USB signals.

The system is powered using a 12 V AC/DC power supply (*TIGER Power Supplies TP1024* [36]) that is plugged into a DC power connector (*Wurth Elektronik 694106402002* [37]). The ESP32 module and all three servomotors are powered using 5 V power from a DC-DC

converter (*OKYSTAR OKY3501-1* [38]) that steps down the 12 V to 5 V. Both of the infrared reflective sensors output an analog signal that is measured by the ESP32's ADC (*Analog-to-Digital Converter*). Since the ESP32's ADC input voltage range is 0 to 3.3 V the infrared reflective sensors are powered using 3.3 V from the ESP32 module's onboard voltage regulator.

Testing revealed that the drone motors drew a combined peak current of 3 A when accelerating and 1 A when spinning at 15% throttle. The whole system's current draw was measured to be approximately 1.2 A at 12 V DC when the motors stopped accelerating and reached 15% throttle. The AC/DC power supply is rated for 5 A making it suitable for powering the whole system.

The servomotors and the infrared reflective sensors were wired using 22 AWG (*American Wire Gauge*) ribbon cable. The wires included with the drone motors were extended using 18 AWG wire and connected to the ESCs. The ESCs were connected to the DC power connector using 14 AWG wire and lever wire connectors (*WAGO 221* [39] & *LeverGard 5-Wire Splicing Connector* [40]). Hot glue was applied to the ESC connections (Figure 37) to prevent damage when the wires are twisted.

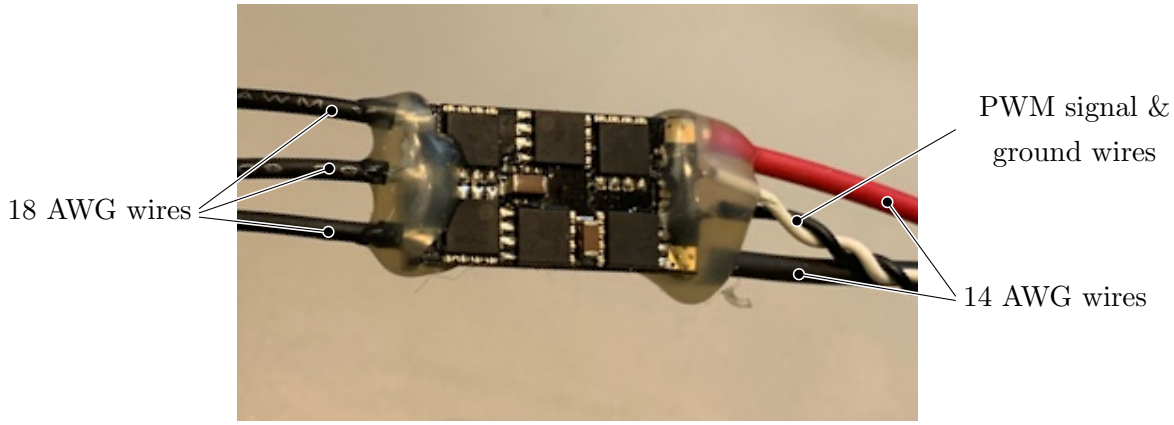


Figure 37: ESC used for regulating the speed of a launching motor using PWM. Hot glue applied to the ESC's connectors to prevent damage when the wires are twisted. The ESC is later wrapped in electrical tape to cover its exposed components.

A protoboard (*DIKAVS Perma-Proto 1/4* [2]), shown in Figure 38, was used to distribute the 5 V and 3.3 V power to the servomotors and the infrared reflective sensors respectively. The protoboard also houses the four resistors required by the infrared sensors. A 3D printed enclosure (depicted in Figure 39) was made to house the ESP32 module, protoboard, DC power connector, DC-DC converter, and lever wire connectors.

[2]DIKAVS does not have an official website but offers their products on AliExpress [41].

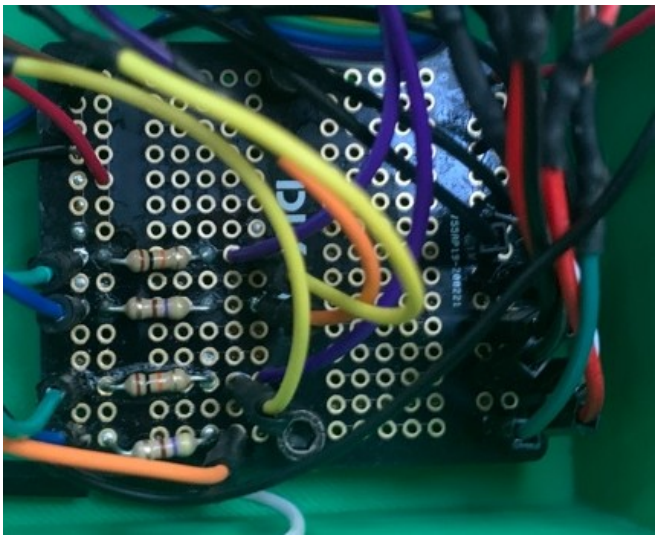


Figure 38: Protoboard used for distributing 5 and 3.3 V power to the servomotors and IR reflective sensors, as well as for connecting the resistors used by the IR reflective sensors.

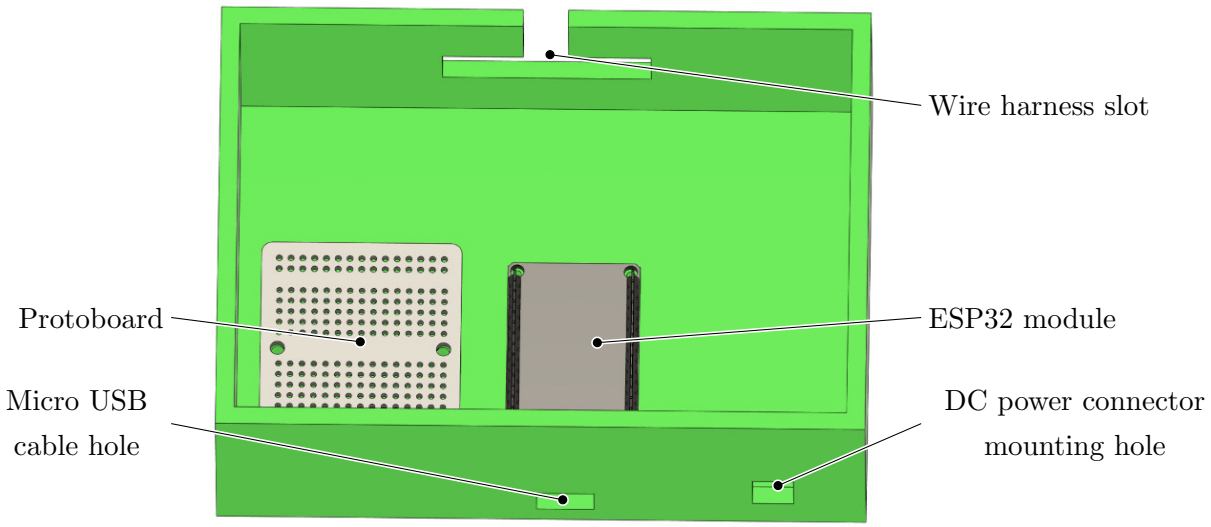


Figure 39: Annotated 3D model of the electronics enclosure with its lid removed. Six M3 bolts are used to secure the protoboard and ESP32 module to the enclosure. The DC-DC converter and wiring are not depicted.

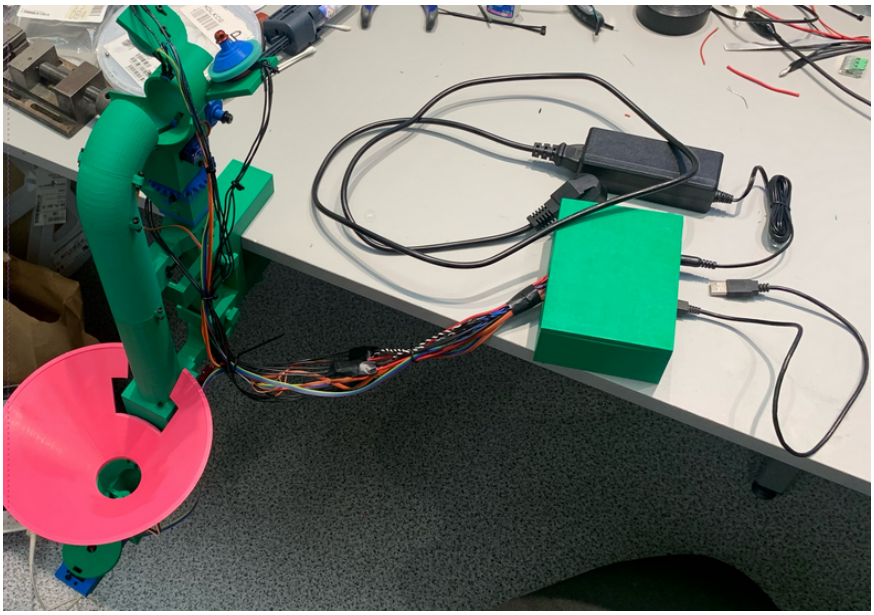


Figure 40: Rhea mounted on a table next to the electronics enclosure. The wires are tied together using cable ties and electrical tape. An AC/DC power supply and a USB wire are connected to the enclosure.

Figure 40 shows how the electronic components used in Rhea are wired to a standalone electronics enclosure. The wires are tied together using cable ties and electrical tape to form a wiring harness that fans out to each individual electronic component.

6 Firmware and user software

The software used for controlling Rhea consists of two parts: low-level firmware running on the ESP32 and high-level user software running on a computer. Communication between the two is achieved via UART (*Universal Asynchronous Receiver-Transmitter*) over USB. Source code for the ESP32 firmware and user software can be found in Appendix A.

6.1 Firmware

The low-level firmware was written in C++ using the Espressif IoT Development Framework (ESP-IDF) [42]. The ESP-IDF provides an API for accessing the ESP32's various peripherals. Controlling the servomotors was implemented using the LED Control peripheral [43] which can generate PWM signals. Reading the analog signals from the infrared reflective sensors was done using the ADC Oneshot Mode Driver peripheral [44]. Finally, communication with the computer was implemented using the UART driver peripheral [45].

All the servomotors and ESCs used in Rhea are controlled using 50 Hz PWM signals. The two *TowerPro SG90* servomotors used by the Launch unit and the Pivot Mechanism are controlled by varying their signal's duty cycle between 1–2 ms. The *Parallax Continuous Rotation Servo #900-00008* used by the Ball Pusher operates on a 1.3–1.7 ms duty cycle, with 1.3 ms driving the motor clockwise and 1.7 ms driving the motor counterclockwise at maximum speed. To push a ball, the firmware toggles the servomotor, on driving at its maximum speed. The ADC peripheral continously measures the crank's angular position using the infrared reflective sensor as an encoder and toggles the servomotor off once one full revolution is detected. Each ESC operates on a 1–2 ms duty cycle but as testing revealed only a maximum duty cycle of 1.15 ms corresponding to 15% maximum throttle is required.

The firmware has an internal state machine (Figure 41), which works based on shots configured by the user software. Each shot consists of a launch direction (yaw angle) and three motor speeds. When a session is started the firmware starts cycling through all the configured shots until it is stopped by the user or a timeout occurs. The computer sends a keepalive packet to the ESP32 every two seconds to prevent the launching motors from rotating indefinitely if connection between the two is lost.

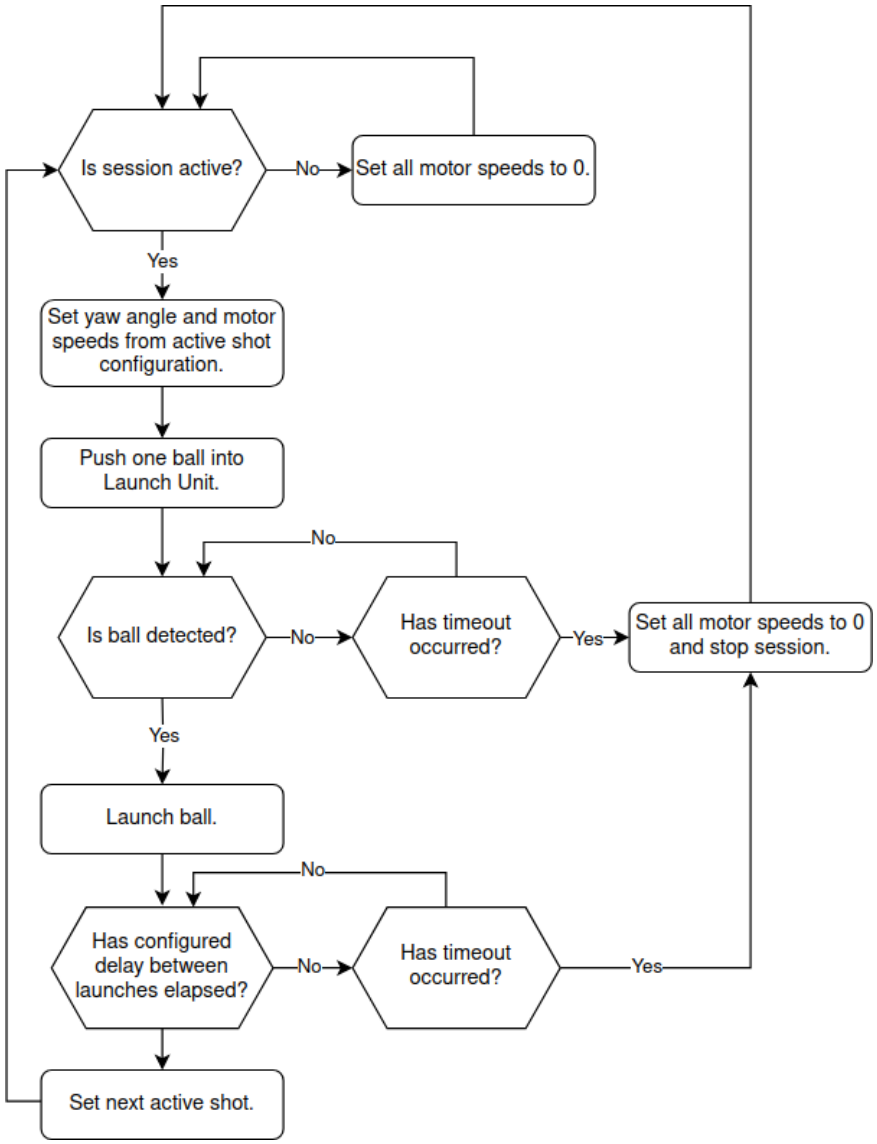


Figure 41: State machine of the firmware running on the ESP32 module.

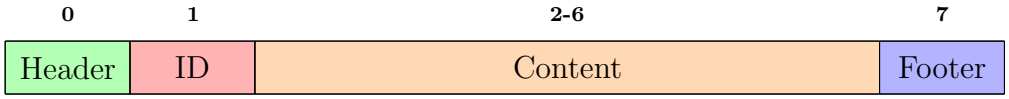


Figure 42: Structure of an individual 8-byte packet, consisting of a header (1 byte), an ID (1 byte), content (5 bytes), and a footer (1 byte).

A packet-based protocol for communication between the computer and ESP32 was established. As depicted in Figure 42 each packet consists of 8 bytes and contains a header, an ID, the packet's contents, and a footer. The values `0x47` and `0xED` were chosen for the header and footer respectively. Each packet is distinguished by its ID which conveys information about what contents the packet has. Table 7 gives an overview of the five different packets used and their functionality.

ID	Name	Content length (bytes)	Description
0	AddShot	4	Add a shot to the session.
1	RemoveShot	1	Remove a shot from the session.
2	StartSession	1	Start cycling configured shots.
3	StopSession	0	Stop ongoing session.
4	Keepalive	0	Notify ESP32 that connection is still valid.

Table 7: Packets used for communication between the user software and firmware.

Packets sent from the computer are received by the ESP32's UART peripheral and are processed one byte at time. Malformed packets not containing a header, a valid ID, or a footer are discarded. Figure 43 describes all the packets that contain additional fields. The StopSession and Keepalive packets do not contain any additional fields.

Type	Name	Description	Range
int8_t	YawAngle	Desired yaw angle.	-30 to 30
uint8_t	BottomSpeed	Desired bottom motor speed.	0 to 100
uint8_t	LeftSpeed	Desired left motor speed.	0 to 100
uint8_t	RightSpeed	Desired right motor speed.	0 to 100

(a) AddShot.

Type	Name	Description	Range
uint8_t	ShotIndex	Index of the shot to be removed.	0 to 255

(b) RemoveShot.

Type	Name	Description	Range
uint8_t	Frequency	Balls per minute.	1 to 50

(c) StartSession.

Figure 43: Contents of the AddShot (a), RemoveShot (b), and StartSession (c) packets. Type, name, description, and value range is provided for each field.

The contents of the AddShot packet are clamped and mapped to their corresponding duty cycle lengths. Namely, the YawAngle (corresponding to the Pivot Mechanism's servomotor), is mapped to 1.1–1.9 ms and the three motor speeds for the ESCs are mapped to 1–1.15 ms. The frequency field of the StartSession packet is also clamped.

6.2 User software

A user can program Rhea's training exercises using software running on a computer. The software was written in Python and the GUI was implemented using pyimgui [46]. The `struct` [47] and `pyserial` [48] Python modules were used to form the communication packets and send them over UART respectively.

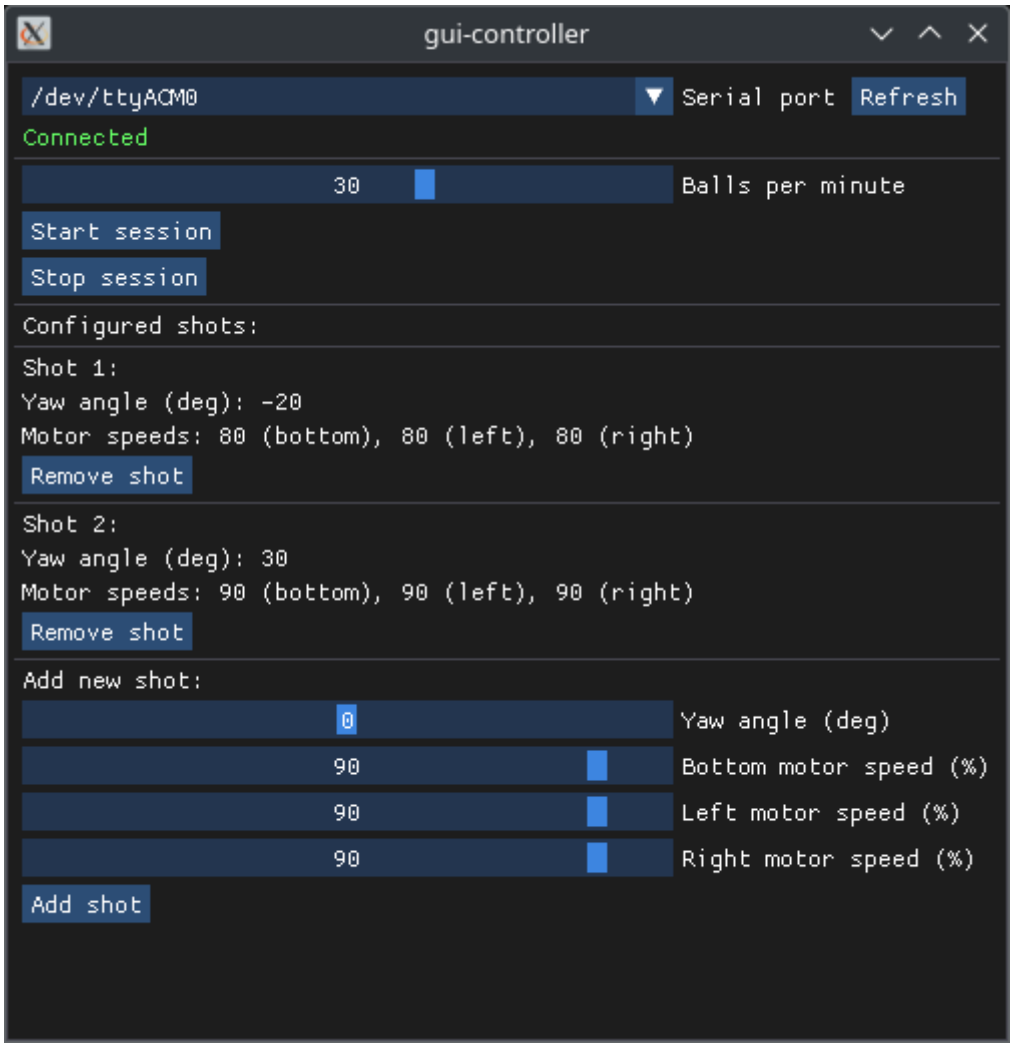


Figure 44: GUI of the user software running on a computer used for managing a training session with Rhea. Two example shots with different parameters are configured.

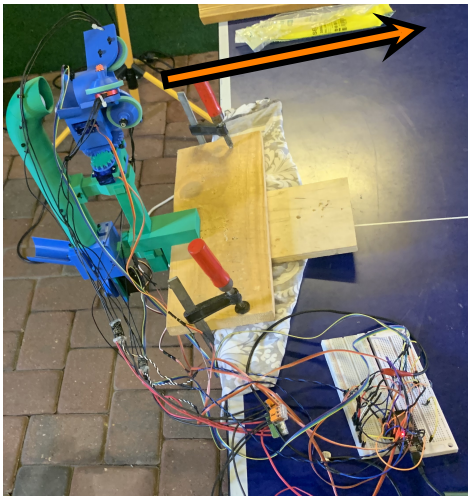
Figure 44 depicts the GUI used to control Rhea. The serial port can be chosen using a drop-down menu. A session can be constructed from multiple shots, with each shot's parameters being individually controllable. After configuring the desired shots, the user can set the ball launching frequency and start the session using a button, which causes the firmware to start cycling through the configured shots. The session can be terminated at any time by the user.

7 Validation

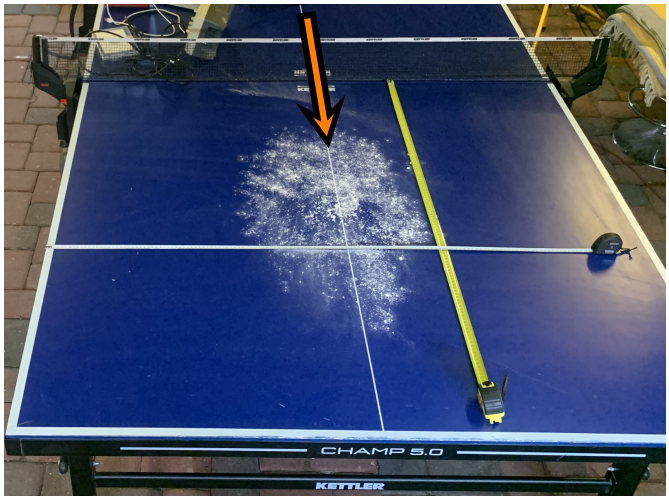
Multiple tests were carried out to assess the accuracy of Rhea. This was achieved by mounting Rhea on a table tennis table and observing the landing area of balls launched at varying speeds and directions. Furthermore, landing location data for the Butterfly *AMICUS PROFESSIONAL* robot was gathered using the same testing setup in order to compare Rhea's accuracy to that of a commercially available robot.

7.1 Initial accuracy tests

The first measurements were done with the Launch Unit's third revision. Since the Fixture did not yet support adjusting the pitch, a 10° elevation angle was achieved with the use of metal clamps and wooden slabs as shown in Figure 45 a. As can be seen in Figure 45 b, two tape measures were laid on the opposite side of the table, which was sprinkled with flour to assess the landing location of the ball. After each launch, the ball left a small imprint in the flour, indicating its landing location. This mark's position relative to the table's edges was measured by eye using the two tape measures.



(a) Rhea mounted on a table tennis table at a 10° elevation angle using wooden slabs.



(b) Rhea's opposite side of the table, sprinkled with flour. Two tape measures laid perpendicularly to visually assess the ball's landing location.

Figure 45: First accuracy measurement setup using the Launch Unit's third revision. The orange arrows depict the launch direction of the ball.

Landing location data for 30 launches was gathered for two polysiloxane hardnesses (Shore A 22 and Shore A 5) as can be seen in Figure 46. Each ball was launched with the same direction (parallel to the table's longer edge) and the same motor speeds (10%). The average standard deviation for each dataset was calculated by taking the mean of the standard deviations of the x-coordinates and y-coordinates. For both datasets, the

three sigma confidence ellipse indicates that based on the data, 98.89% of future launches should land inside the ellipse [49].

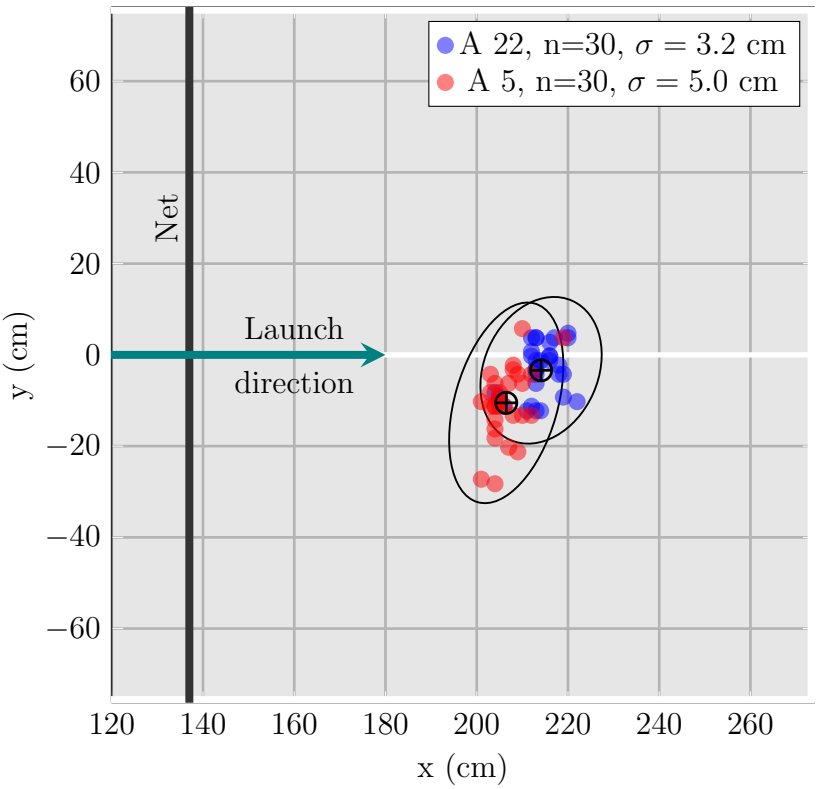


Figure 46: Landing location data for Rhea at a 10° angle for Shore A 22 and Shore A 5 polysiloxane traction wheels. The mean, average standard deviation, and confidence ellipse with 3 times the standard deviation are illustrated for both polysiloxane hardnesses.

The larger landing location spread of the Shore A 5 traction wheels may have arisen from greater energy losses caused by material deformation experienced by the A 5 polysiloxane. Pinching diameter, defined as the diameter of the largest ball that can fit through the traction wheels without any interference, has been shown to affect the landing accuracy [11]. Since replacing the traction wheels required removing the motors, the pinching diameter of the two datasets was not identical. Nevertheless, the data indicated that both polysiloxane hardnesses were suitable for use and satisfied the fourth requirement (Targeting and Accuracy).

7.2 Automated testing setup

Although sufficient for the initial accuracy tests, visually assessing the landing location using flour was inaccurate as the center of the imprint left in the flour was not always clearly visible. Moreover, this process required a human to manually inspect each landing location between launches and was therefore unfavourable for gathering larger datasets.

A more automatic solution was developed using a smartphone camera and a piezoelectric element as shown in Figure 47. A 3D printed smartphone holder was fixed to the ceiling using screws as can be seen in Figure 48. The smartphone's camera records a top-down view of the table at 240 frames per second and is used to detect where a ball lands on the table. An ESP32 microcontroller detects a ball bounce using the piezoelectric element, timestamps it, and relays it to a computer over a USB connection. This timestamp was later used to correlate a ball bounce event with an appropriate frame from the camera's video.

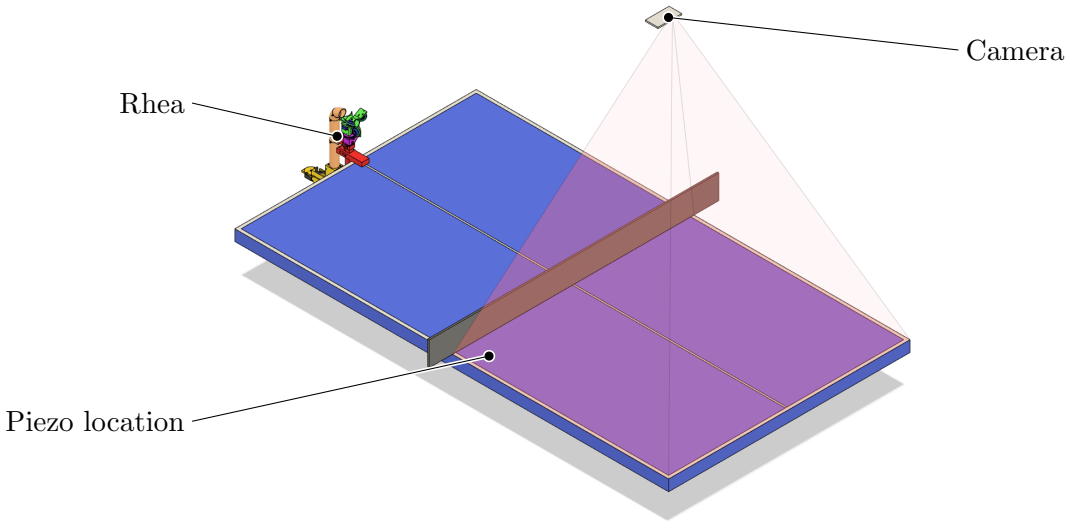


Figure 47: Annotated model depicting the automatic setup used for gathering ball landing location data. A camera mounted above the table records slow motion video. The transparent red pyramid represents the camera's field of view. A piezoelectric element placed on the table detects ball bounces.

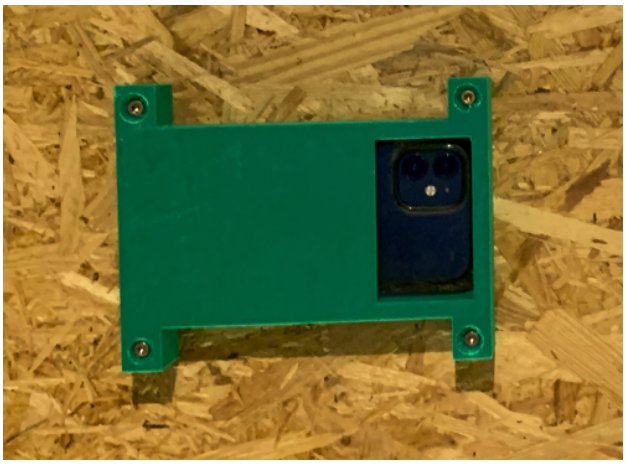


Figure 48: 3D printed smartphone mount located above the table tennis table, fixed to a wooden ceiling using screws. The smartphone's camera provides a top-down view of one side of the table and it can be slid in and out of the mount.

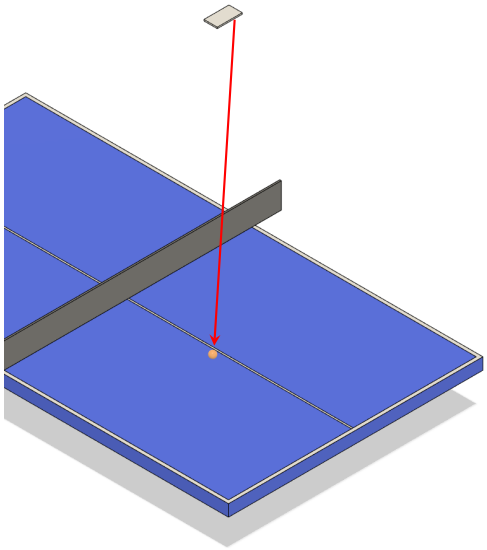
Before mounting the camera, a ball bounce event was recorded with the camera's view angle parallel to the table, providing a synchronized piezo timestamp and camera frame. Another such synchronization event is recorded at the end of a data-gathering session. These two events are used to map all the ball bounce events gathered during the session to a correct camera frame, mitigating any drift that might have occurred in the microcontroller's internal clock.

For mapping a 2D screen point on a camera frame to a location on the table the camera's intrinsic and extrinsic properties needed to be resolved. This was achieved by mounting the camera above the table and moving a flat 6x9 calibration chessboard on the table in a gridlike pattern, covering the whole table (Figure 49). OpenCV's [50] `findChessBoardCorners` method was used to find the chessboard locations for every frame, which were then used by the `calibrateCamera` and `getOptimalNewCameraMatrix` functions to calculate the camera's intrinsic and extrinsic properties. These properties were used to undistort frames captured by the camera.

Figure 49: Gathering calibration data for resolving the camera's intrinsic and extrinsic properties for the automatic test setup. A 6x9 flat calibration chessboard is moved to various locations on the table, and recorded by a camera. Three sample locations are pictured.

The `calibrateCamera` function also provided the estimated rotation and translation vectors for each of the chessboards which were used to fit a 3D plane representing the table using singular value decomposition [51] with NumPy's `numpy.linalg.svd` [52].

(a) Detected ball highlighted by a green circle.



(b) Model of the respective situation illustrating the constructed ray (red) from the camera.

Figure 50: Detecting a ball using OpenCV's SimpleBlobDetector (a) and constructing a ray from the camera towards the detected ball (b). The ray's intersection point with the table's plane represents the ball's landing location.

Figure 50 illustrates how the camera detects a ball and how a ray is constructed towards the detected ball. Finding the landing location of a ball is broken down into the following steps:

1. Calculate the corresponding camera frame from piezo timestamp.
2. Undistort the camera frame and detect the ball's 2D screen location using SimpleBlobDetector.
3. Construct a ray from the camera towards the ball.
4. Find the intersection point of the ray and the table's plane.

7.3 Comparison with a commercial robot

Dittrich et al. have published accuracy data for various commercial robots, as well as for their own developed robot, AIMY [11]. This data was recorded with a specialized vision system, developed by Gomez-Gonzalez et al. [53]. Recreating this setup was beyond the scope of this thesis hence the data published by Dittrich et al. would be unsuitable to compare to the data gathered using the automated test setup described in the previous subsection. In order to compare Rhea's accuracy to an existing robot, a Butterfly *AMICUS PRIME* robot was obtained. Six target locations, three in the length-wise middle of the

opposite side of the table and three near the edge (Figure 51), were chosen to assess the accuracy and repeatability of both robots.

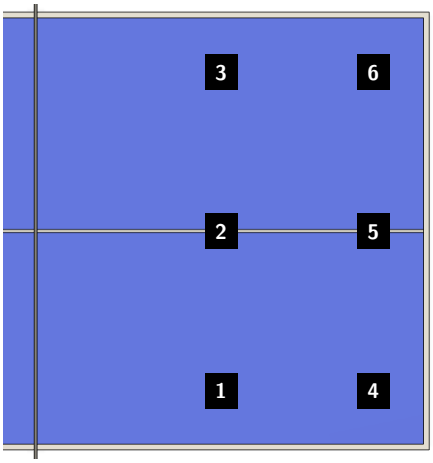
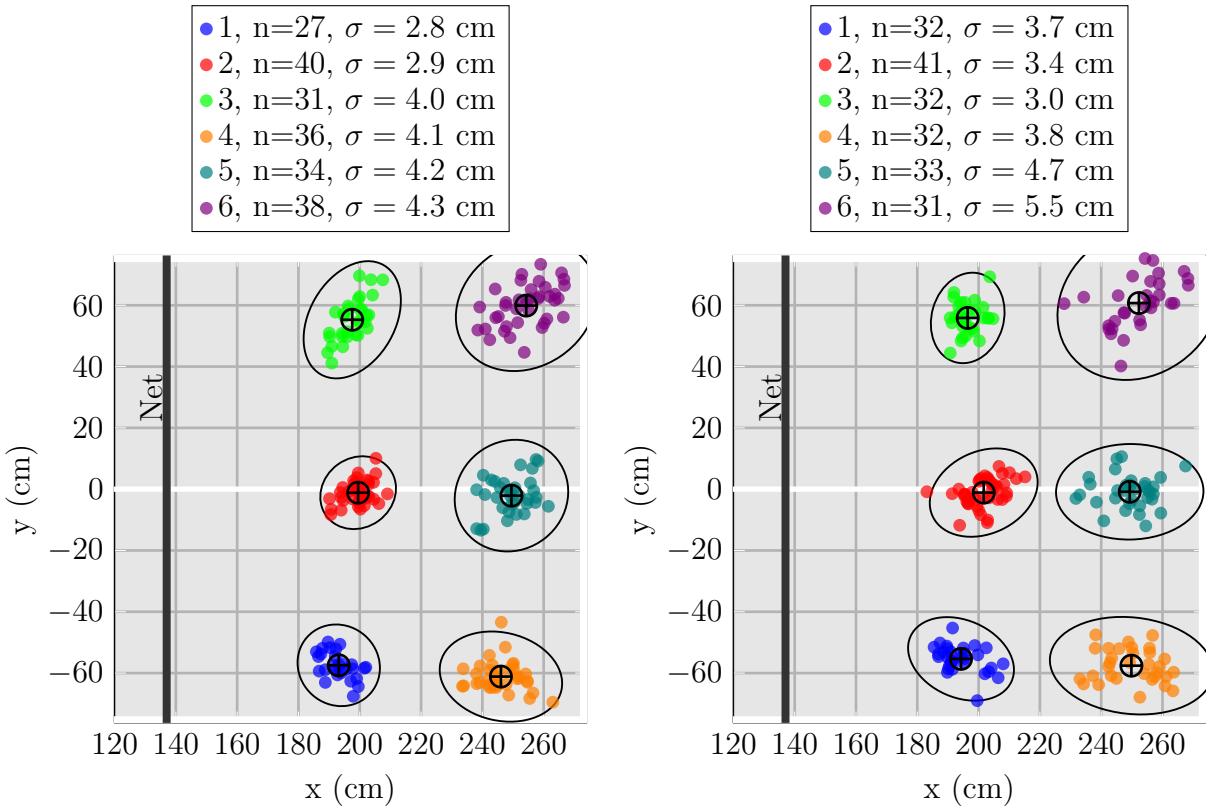


Figure 51: Six target landing locations used for automated testing.

For both robots two datasets were gathered: one without changing the target location (a 111..., 222..., 333..., 444..., 555..., 666... "single" pattern) and one where the target location is changed between every launch (a 123456123456... "cluster" pattern). These patterns were chosen to assess if changing the direction or speed between launches influenced the accuracy of the robot. For Rhea, the altitude angle was set to 10° using the Fixture, the yaw angle was controlled by the Pivot Mechanism and the launch distance was altered by changing motor speeds. The *AMICUS PRIME* was configured to target roughly the same landing locations by adjusting its trajectory (altitude angle), placement (yaw direction) and launch speed. Both robots were configured to launch a ball every four seconds without applying any intentional spin. However, some inherent spin was still introduced due to subtle variations in motor speeds.

Figure 52 shows data gathered using Rhea for the "single" and "cluster" patterns. The data indicates that the further target locations (4, 5, 6) have a bigger error than the nearer ones. This may be explained by variability in launch characteristics, meaning slight differences in how the traction wheels make contact with the ball between launches add up the further away the target is from the launcher. Not only can these slight differences affect the direction of the launch but the unintentional spin as well, meaning that the lift experienced by the ball from the Magnus effect differs from launch to launch. The "cluster" data shows a greater error overall which can be explained by the launch direction differing slightly between launches which is caused by the Pivot Mechanism's servo not achieving the same exact yaw angle every time.

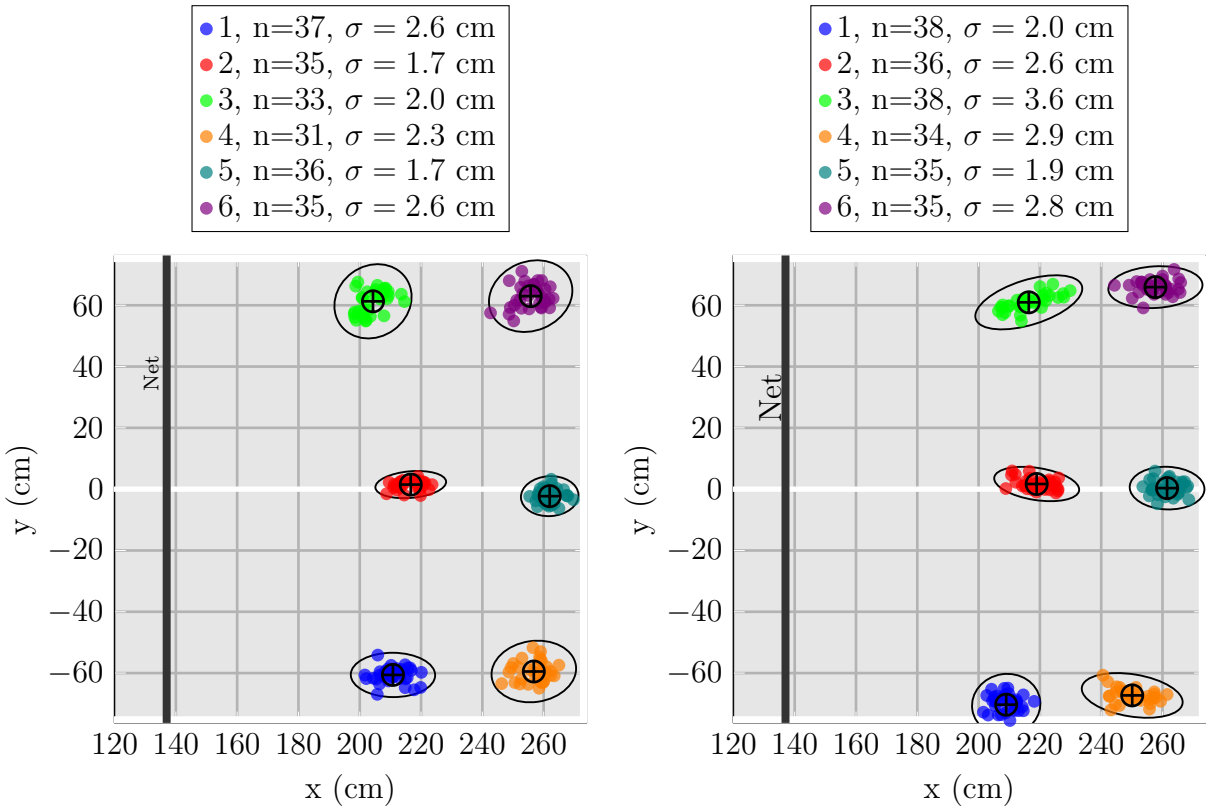


(a) Single pattern.

(b) Cluster pattern.

Figure 52: Landing location data for Rhea at a 10° angle gathered using the automated test setup and Shore A 22 traction wheels. Two datasets shown: one where the target location is not changed between launches (a), and one where it is changed (b). The mean, average standard deviation, and confidence ellipse with 3 times the standard deviation are illustrated for each target location.

Datasets using the Butterfly *AMICUS PRIME* for both patterns can be seen in Figure 53. The *AMICUS PRIME*'s performance was more consistent across the two measurement patterns. Additionally, the further middle location's (5) error was comparable to the nearer middle's one (2), indicating that the *AMICUS PRIME*'s accuracy is less affected by distance compared to Rhea's.



(a) Single pattern.

(b) Cluster pattern.

Figure 53: Landing location data for the Butterfly *AMICUS PRIME* gathered using the automated test setup. Two datasets shown: one where the target location is not changed between launches (a), and one where it is changed (b). The mean, average standard deviation, and confidence ellipse with 3 times the standard deviation are illustrated for each target location.

Although Rhea is less accurate than the Butterfly robot, it manages to perform well in targeting different locations of the board, with a mean standard deviation of $\sigma = 3.8$ cm. All of the landing locations, except for one of the cluster pattern's further targets (Figure 52 b location 6), showed targeting accuracy within the range specified by the fourth requirement (Targeting and Accuracy). Figures of both robots being used with the automated testing setup can be found in Appendix C.

8 Discussion

This section reflects the author's subjective opinion based on testing and using Rhea and also discusses potential future work.

Manufacture of Rhea requires components totaling 180.80 EUR (BOM in Appendix A) and is possible with common hobbyist tools, most notably a 3D printer. By design, the requirements for the 3D printer are minimal, with all mechanical components printed using PLA and requiring a minimal build volume of only 20x20x14 cm. However, some of the components require printing in a non-ideal orientation leading to excessive use of support filament. These issues could be remedied by splitting the models into multiple parts, allowing for alternative printing orientations. Furthermore, the M3 mounting holes used in the Pipe are orientated in a way that may cause them to break when excessive force is applied. The Pipe's components could be substituted with existing materials, such as those used in plumbing, cutting down on printing time and making the construction less prone to breaking.

Setting up Rhea for a training session involves mounting it on a table using the Fixture's clamp, manually adjusting the pitch, powering it using an AC/DC adapter, and connecting it to a computer. Extensive on-site assembly is not required and the setup takes around three minutes (not accounting for the setup time of a ball catch net, if used).

Training using Rhea is feasible but does have some caveats. The primary issue comes from the funnel used in the Ball Pusher, which suffers from clogging. An additional servomotor could be added to act as a stirrer for the balls in the funnel preventing them from getting stuck. The requirement of a computer for controlling Rhea affects its portability and potentially the user experience. The ESP32 microcontroller has built-in WiFi and Bluetooth support meaning a smartphone or tablet application could be used instead of a computer, similar to the Butterfly *AMICUS PRIME* and PowerPong *OMEGA* robots. This would also eliminate the need for a wired connection between the user interface and Rhea's electronics enclosure. However, developing such an application is beyond the scope of this thesis.

Rhea is able to target 75% of it opposite side of the table, 5% lower than the 80% established in the Targeting and Accuracy requirement. This limitation stems from the Fixture's maximum 20° elevation angle. More specifically, a greater elevation angle would yield a more curved ball trajectory, which could land nearer to the net that separates the two courts.

This thesis did not involve assessing the spin generation capabilities of Rhea. Measuring spin would necessitate a separate testing setup which is beyond the scope of this thesis. Spin information would provide insight into the effectiveness of the three-motor launching setup compared to human players.

9 Conclusion

The objective of this bachelor's thesis was to develop and manufacture an open-source table tennis ball launcher robot which could be used for multiball training. This thesis extensively documents the development of Rhea's mechanical, electronic, and software components and assesses the repeatability of its launches.

All established requirements, with the exception of were fully met. The manufacture and assembly of Rhea is possible with common hobbyist tools and requires only readily available materials and components, the total cost of which is 180.80 EUR. Thanks to the adjustable Fixture and Pivot Mechanism, Rhea is able to target 75% of its opposite side of the table. Validation revealed that Rhea is able to repeatedly target different locations of the table with a mean standard deviation of σ=3.8 cm, making it suitable for the use of multiball training. Resources required to replicate Rhea are open-sourced and publicly available online.

10 Acknowledgements

I would like to express my deepest gratitude to my supervisors, Eva Mõtshärg and Meelis Pihlap, for providing me with guidance, expertise, and thorough feedback throughout the entire thesis. Their support has been instrumental in shaping this work. Additionally, I would like to thank Tartu Spordiselts Kalev for lending me their training robot for data gathering purposes. Finally, I would like to thank my family and friends for their support and encouragement during this process.

References

[1] Victor Barna. *table tennis*. Encyclopedia Britannica. 2023-12-04. URL: https://www.britannica.com/sports/table-tennis (accessed on 2023-12-02).

[2] The International Table Tennis Federation. *ITTF Statues 2023 (clean version)*. 2023. URL: https://documents.ittf.sport/document/311.

[3] The International Table Tennis Federation. *Table Tennis in 2018: a year to remember in the media!* 2018-12-31. URL: https://www.ittf.com/2018/12/31/table-tennis-2018-year-remember-media/ (accessed on 2023-12-02).

[4] JOOLA USA. *Multiball Training is Fundamental to Early Success in Table Tennis*. 2023. URL: https://joolausa.com/multiball-training-is-fundamental-to-early-success-in-table-tennis/ (accessed on 2023-11-05).

[5] Dinesh Jayabalakrishnan and Rajath Kamal Achanta. "A study on quantizing high level table tennis for robot training in India". In: *International Journal of Table Tennis Sciences* No. 8 (2013). URL: https://www.ittf.com/ijtts_nr-8/.

[6] LCS Sportszer Kft. *PowerPong*. URL: https://www.lcs-sport.hu/en/powerpong-2/ (accessed on 2023-12-03).

[7] PowerPong. *Paddle Palace Power Pong OMEGA Robot*. URL: https://www.powerpong.org/collections/professional-robots/products/power-pong-omega-table-tennis-robot (accessed on 2023-11-05).

[8] Butterfly. *AMICUS ROBOTS*. URL: https://en.butterfly.tt/amicus (accessed on 2023-11-05).

[9] Butterfly. *AMICUS PRIME*. URL: https://en.butterfly.tt/amicus-prime.html (accessed on 2023-11-05).

[10] Megaspin. *Power Pong 5000 Robot vs Butterfly Amicus Prime*. 2022-10-13. URL: https://www.megaspin.net/articles/573/power-pong-5000-robot-vs-butterfly-amicus-prime (accessed on 2024-05-15).

[11] Alexander Dittrich et al. *AIMY: An Open-source Table Tennis Ball Launcher for Versatile and High-fidelity Trajectory Generation*. 2023. arXiv: 2210.06048 [cs.RO].

[12] Newgy. *Newgy's Ping-Pong/Table Tennis Robots*. URL: https://www.newgy.com/collections/robots (accessed on 2023-11-06).

[13] Newgy. *Robo-Pong 3050XL Table Tennis Robot*. URL: https://www.newgy.com/collections/robots/products/robo-pong-3050xl (accessed on 2023-11-06).

[14] S. Dubina. *Serve Return - Newgy 3050XL (spinny serves)*. 2018. URL: https://www.youtube.com/watch?v=0vVYdE9LU2s (accessed on 2023-11-06).

[15] JOOLA. *JOOLA ROBOT iPONG V200*. URL: https://www.joola.shop/en/robots/122/joola-robot-ipong-v200 (accessed on 2024-05-10).

[16] JOOLA. *JOOLA ROBOT iPONG V300*. URL: https://www.joola.shop/en/accessories/robots/573/joola-robot-ipong-v300 (accessed on 2024-05-10).

[17] Alexander Dittrich et al. *AIMY project web page*. URL: https://webdav.tuebingen.mpg.de/aimy/ (accessed on 2024-05-18).

[18] Saminda Abeyruwan et al. *i-Sim2Real: Reinforcement Learning of Robotic Policies in Tight Human-Robot Interaction Loops*. 2022. arXiv: 2207.06572 [cs.RO].

[19] Jonas Tebbe et al. *Sample-efficient Reinforcement Learning in Robotic Table Tennis*. 2024. arXiv: 2011.03275 [cs.RO].

[20] Intelligent Soft Robots. *Low-level Ball Launcher Control*. URL: https://github.com/intelligent-soft-robots/ball_launcher_beepy.

[21] Intelligent Soft Robots. *AIMY Evaluation and Target Shooting*. URL: https://github.com/intelligent-soft-robots/aimy_target_shooting.

[22] Intelligent Soft Robots. *Ball Launcher Hardware*. URL: https://github.com/intelligent-soft-robots/ball_launcher_hardware.

[23] Alexander Dittrich. *Table Tennis Ball Trajectories with Spin*. Version V1. 2022. DOI: 10.17617/3.BIVUVL. URL: https://doi.org/10.17617/3.BIVUVL.

[24] Barath Ponnusamy, Wong Fei Yong, and Zulkifli Ahmad. "A low cost automated table tennis launcher". In: *Journal of Engineering and Applied Sciences* 10 (2015-01), p. 6.

[25] Mohammad Esmael et al. "Spin-Ninja Table Tennis Training Robot". In: *2023 5th International Conference on Bio-engineering for Smart Technologies (BioSMART)*. 2023, pp. 1–6. DOI: 10.1109/BioSMART58455.2023.10162116.

[26] Simplify3D. *PLA, Ultimate Materials Guide*. URL: https://www.simplify3d.com/resources/materials-guide/pla/ (accessed on 2024-05-18).

[27] Parallax. *Parallax Continuous Rotation Servo*. URL: https://www.parallax.com/product/parallax-continuous-rotation-servo/ (accessed on 2024-05-14).

[28] Vishay. *TCRT5000, TCRT5000L, Reflective Optical Sensor with Transistor Output*. URL: https://www.vishay.com/docs/83760/tcrt5000.pdf (accessed on 2024-05-14).

[29] Racerstar. *Racerstar Racing Edition 2205S BR2205S 2300KV 2-4S Brushless Motor For X210 220 QAV250 FPV Frame*. URL: https://www.racerstar.com/Racerstar-Racing-Edition-2205S-BR2205S-2300KV-2-4S-Brushless-Motor-For-X210-220-QAV250-FPV-Frame-p-68.html (accessed on 2024-05-14).

[30] Ariesrc. *Racerstar Racing Edition 2205S BR2205S 2300KV 2-4S Brushless Motor For X210 220 QAV250 FPV Frame*. URL: Racerstar%20MS%20Series%2035A%20ESC%20BLHeLi_S%20OPTO%202-4S%20Supports%20Dshot600%20For%20RC%20Drone%20FPV%20Racing%20Multi%20Rotor (accessed on 2024-05-14).

[31] Zerynth. *SERVO MOTOR SG90 DATA SHEET*. URL: http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf (accessed on 2024-05-15).

[32] TowerPro. *SG90 Digital - TowerPro*. URL: https://www.towerpro.com.tw/product/sg90-7/ (accessed on 2024-05-14).

[33] Polistep. *Two-component moulding Silicone on Tin Catalyst PLS-505 (hardness 5-ShA)*. URL: https://polistep.lv/en/two-component-silicone/tproduct/282059541-912148850761-two-component-moulding-silicone-on-tin-c (accessed on 2024-05-18).

[34] Zhermack. *Elite Double 22*. URL: https://www.zhermack.com/en/product/elite-double-22/ (accessed on 2024-05-18).

[35] Imperial College London. *DOIT Esp32 DevKit v1*. URL: https://olddocs.zerynth.com/r2.5.2/official/board.zerynth.doit_esp32/docs/index.html (accessed on 2024-05-15).

[36] TIGER Power Supplies. *TIGER PART: TP1024 datasheet*. URL: https://static.rapidonline.com/pdf/85-7637_v1.pdf (accessed on 2024-05-15).

[37] Wurth Elektronik. *WR-DC DC Power Jack Vertical THT Ø 6.4*. URL: https://www.we-online.com/en/components/products/DC_THT_VERTICAL_6_4_69410X402002 (accessed on 2024-05-15).

[38] OKYSTAR. *OKY3501-1*. URL: https://www.okystar.com/product-item/dc-dc-buck-module-6-24v12v24v-to-5v3a-car-97-5-oky3501-1/ (accessed on 2024-05-15).

[39] WAGO. *COMPACT splicing connector; 3-conductor; with operating levers*. URL: https://www.wago.com/ca-en/installation-terminal-blocks-and-connectors/compact-splicing-connector/p/221-413 (accessed on 2024-05-15).

[40] Gardner Bender. *LeverGard 5-Wire Splicing Connector*. URL: https://www.gardnerbender.com/en/p/19-PCL5/LeverGard-5-Wire-Splicing-Connect (accessed on 2024-05-15).

[41] AliExpress DIKAVS. *DIKAVS Store*. URL: https://www.aliexpress.com/store/1552478 (accessed on 2024-05-19).

[42] Espressif Systems. *ESP-IDF: Espressif IoT Development Framework*. URL: https://github.com/espressif/esp-idf.

[43] Espressif. *LED Control (LEDC), ESP-IDF API reference*. URL: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/ledc.html (accessed on 2024-05-15).

[44] Espressif. *Analog to Digital Converter (ADC) Oneshot Mode Driver, ESP-IDF API reference*. URL: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/adc_oneshot.html (accessed on 2024-05-15).

[45] Espressif. *Universal Asynchronous Receiver/Transmitter (UART), ESP-IDF API reference*. URL: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/uart.html (accessed on 2024-05-15).

[46] pyimgui developers. *pyimgui*. URL: https://github.com/pyimgui/pyimgui.

[47] Python Software Foundation. *struct – Interpret bytes as packed binary data, Python 3.12.2 documentation*. URL: https://docs.python.org/3/library/struct.html (accessed on 2024-05-19).

[48] pySerial developers. *pySerial*. URL: https://github.com/pyserial/pyserial.

[49] Bin Wang, Wenzhong Shi, and Zelang Miao. "Confidence Analysis of Standard Deviational Ellipse and Its Extension into Higher Dimensional Euclidean Space". In: *PloS one* 10 (2015-03), e0118537. DOI: 10.1371/journal.pone.0118537.

[50] OpenCV. *Camera Calibration and 3D Reconstruction*. URL: https://docs.opencv.org/4.9.0/d9/d0c/group__calib3d.html (accessed on 2024-05-09).

[51] John D'Errico. *Best fitting a plane (ax + by + cz = 0) where c could be 0*. URL: https://www.mathworks.com/matlabcentral/answers/297854-best-fitting-a-plane-ax-by-cz-0-where-c-could-be-0#answer_230360 (accessed on 2024-05-09).

[52] NumPy. *numpy.linalg.svd*. URL: https://numpy.org/doc/stable/reference/generated/numpy.linalg.svd.html (accessed on 2024-05-09).

[53] Sebastian Gomez-Gonzalez et al. "Reliable Real-Time Ball Tracking for Robot Table Tennis". In: *Robotics* 8.4 (2019). ISSN: 2218-6581. DOI: 10.3390/robotics8040090. URL: https://www.mdpi.com/2218-6581/8/4/90.

[54] JOOLA. *iPONG Carbon Fiber Table Tennis Ball Catch Net*. URL: https://joolausa.com/carbon-fiber-table-tennis-ball-catch-net/ (accessed on 2024-05-15).

A Publicised materials

The following list contains resources for the manufacture and assembly of Rhea. All resources are provided free of charge and are licensed under the GNU General Public License Version 3.

- 3D models. `https://github.com/joshpmmst/Rhea/tree/main/models`
- Printing and assembly instructions. `https://github.com/joshpmmst/Rhea/tree/main/printing-and-assembly`
- Firmware code. `https://github.com/joshpmmst/Rhea/tree/main/firmware`
- User interface code. `https://github.com/joshpmmst/Rhea/tree/main/software`
- BOM. `https://github.com/joshpmmst/Rhea/tree/main/bill-of-materials`

B Silicone experiments and moulds

Figures 54 and 55 show the difficulties faced when removing caulk from the moulds. Figure 56 contains the final tests using caulk with the inner mould walls lubricated with oil to deter adhesion. As can be seen in Figure 57, oil did not help.

Figures 58 and 59 show the final two iterations of the moulds, secured with M3 bolts and nuts, poured with Shore A 22 polysiloxane.

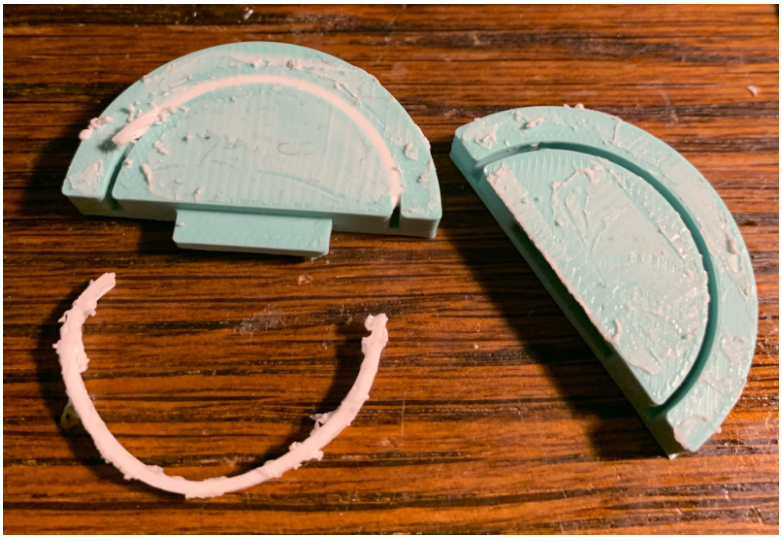


Figure 54: Removing dried caulk from the mould's first iteration.

Figure 55: Unevenly dried caulk inside the second iteration of the mould.

Figure 56: Caulk drying inside third iteration moulds, inner surfaces lubricated with oil.

Figure 57: Removing dried caluk from a third iteration mould.

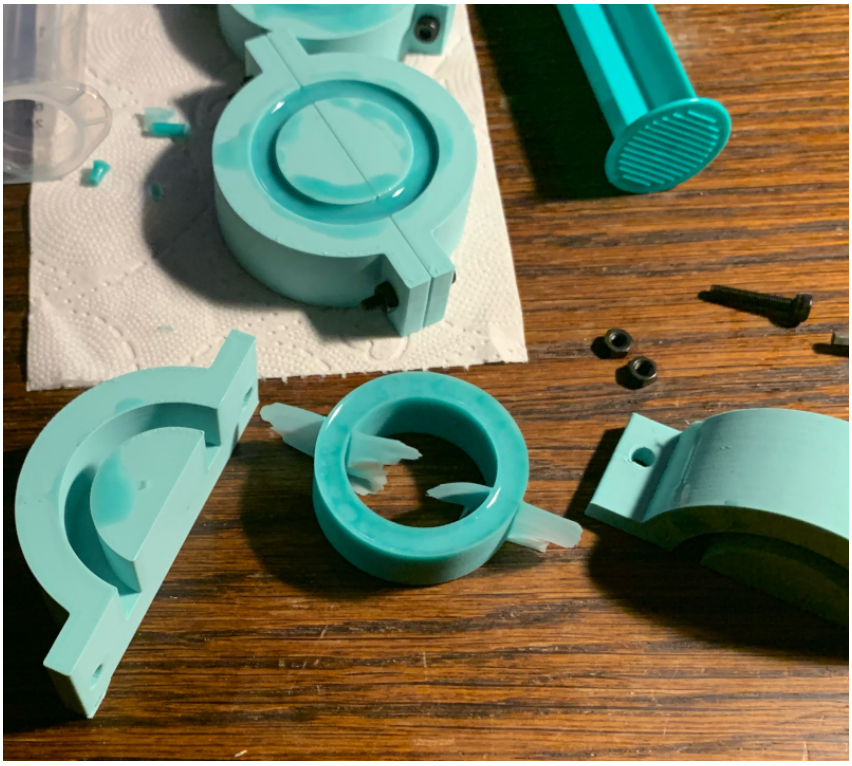


Figure 58: Fourth iteration of the moulds, poured with Shore A 22 polysiloxane.

Figure 59: Fifth iteration the moulds, poured with Shore A 22 polysiloxane.

C Automated testing setups

Figure 60: Rhea mounted on a table tennis table at a 10° altitude angle. The launched balls are caught using a ball catch net (*JOOLA Carbon Graphite Ball Catch Net* [54]) and returned under the table using a gutter for convenience. A roll of yellow masking tape holds down the piezoelectric element on the opposite side of the table for detecting ball bounces. A smartphone camera (not pictured) mounted on the ceiling records the landing locations of the balls.

Figure 61: Butterfly AMICUS PRIME mounted on a table tennis table with the included ball catch net behind it. The launched balls were caught by hand and returned to the robot's side of the table. A roll of yellow masking tape holds down the piezoelectric element on the near side of the table for detecting ball bounces. A 3D printed smartphone camera (green) can be seen mounted to the wooden ceiling.

Non-exclusive license to reproduce the thesis and make the thesis public

I, Märten Josh Peedimaa,

1. grant the University of Tartu a free permit (non-exclusive license) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright, my thesis

   ”Rhea: An open-source table tennis ball launcher robot for multiball training“,

   supervised by Eva Mõtshärg and Meelis Pihlap.

2. I grant the University of Tartu a permit to make the thesis specified in point 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in points 1 and 2.

4. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Märten Josh Peedimaa
20/05/2024