NTNU
Kunnskap for en bedre verden

Department of Mechanical and Industrial Engineering

TPK4560 - Robotics and Automation, Specialization Project

Improvements of the shuttlecock launcher robot BADDY

Authors:
Hege Rishovd
Ane Sofie Smith Kristiansen

Supervisor:
Amund Skavhaug

December, 2020

Preface

We would like to thank our supervisor Amund Skavhaug for all the help and support for this project, and the loan of equipment. Thanks to Benoit Greslebin, founder of BADDY, for answering our questions. Thanks to Nordic Semiconductor for helping us with their products. In addition, we would like to thank Ingeborg Ardø for providing a helping hand.

This particular project was chosen because we had some experience with badminton and was curious about building robots. We wanted a clearly defined project with few limitations to what could be done. Making the remote controller appealed to us because we wanted to gain knowledge about both hardware and software development, perform 3D-printing, and having a high chance of realizing the project. The work of this report can easily be extended and developed further in a master thesis.

Hege Rishovd and Ane Sofie Smith Kristiansen
December 2020

Executive summary

This report describes the development process of improving an already existing badminton shuttlecock launcher robot, named BADDY. Necessary background research, an interview with the founder of the BADDY project, and a literature study have been performed. Different improvements were considered and discussed with regards to three focus areas; security, performance, and ease of use. One of the suggested improvements were chosen to implement, which was making a remote controller for controlling BADDY. This was done to replace the use of big smartphones when playing badminton. To make the remote controller, three different design processes were performed. These were case designing, hardware development, and software development.

The remote controller was made using 3D-printing, Bluetooth Low Energy (BLE) technology, and Arduino coding. The final result still has a lot of potential for further improvement, and this report ends with a section describing possibilities for further work.

The BADDY robot helps to improve the quality of the game, especially for young players who are in their learning phase. The remote controller will add a new layer of ease of use to BADDY, and the making of a controller will also add new aspects of robot building to the BADDY project.

Table of Contents

1 Introduction

BADDY is the name of an ongoing, open-source robotics project. The project includes both the hardware and software required to build a badminton shuttlecock launcher, named BADDY. There exists a worldwide BADDY community, where people share personal experiences and problems they have encountered with their BADDYs. The community encourage others to mount, improve, and develop their own robot. The documentation is open and the source code is free to use and modify. This report consists of a research and development study. Some possible ways to improve BADDY were examined, and one of the suggested improvements was selected and developed.

1.1 Problem description

This report addresses a problem consisting of two parts in regards to the use of a BADDY robot. The first problem was based on finding weaknesses of the robot and possible ways to improve these weaknesses. The second problem addressed one of the suggested improvements and how to implement it to BADDY. It was discussed in depth how this solution could improve BADDY, as well as how the development and implementation process went.

The second problem was based on the use of BADDY and the way it was controlled. BADDY is, as of today, controlled by an application on a smartphone. A smartphone is big and may compromise the quality of the training session as the player have to hold on to the phone while playing. This report will look into a possible addition to BADDY that will eliminate the need for a smartphone while satisfying the original requirements and goals for the BADDY project; open-source, affordable, and sustainable.

Figure 1: The BADDY logo [8]

1.2 Objectives

In consultation with the supervisor, this report has been limited to study the BADDY project within the focus areas of the specialization course, TPK4560 Robotics and Automation, at the Norwegian University of Science and Technology. This is a 15 credits course that lasted 16 weeks during the fall of 2020. During this time period, the worldwide pandemic, COVID-19, influenced the whole society. The pandemic resulted in several restrictions, of which the use of labs was limited. Both the equipment and workspace needed was affected and less available for use. In addition to this, the delivery time of ordered equipment was also increased. These were all factors that affected the work and progress of this report.

The main objective of this project consists of looking at the already existing BADDY shuttlecock launcher robot and finding possible improvements that can increase the performance of the robot. In order to do so, the candidates had to acquire sufficient knowledge and background theory, in addition to familiarize themselves with previous work that has been done on BADDY. Some possible improvements were considered before they were evaluated based on several factors. One of the suggested improvements was chosen to proceed with and this subsystem has been implemented to the extent time allowed.

When going through the background material, the candidates have become familiar with aspects of badminton, problems that arise during the learning phase, and equipment needed to play. The construction and operation of BADDY have been carefully examined, as well as the history and motivation behind the project have been examined more deeply. Knowledge regarding the use of Bluetooth Low Energy (BLE), have been acquired and the benefits such technology can provide, compared to other wireless technology such as i.e., WiFi or earlier Bluetooth versions. Additionally, knowledge about 3D-printing, 3D-drawing, and design processes have been obtained.

During the research of BADDY, and considering potential improvements that could be implemented, three areas have been emphasized. These were security, performance, and ease of use. The suggested improvements have been compared, and one of the improvements has been chosen to be partly implemented within the restrictions and limitation of available material, tools, and time. A real BADDY robot was purchased, but not delivered in time for implementing the improvement to BADDY. Ordering BADDY through university institutions was time-consuming and additionally, the shipment was stuck at the Norwegian customs office. The absence of BADDY affected and limited the development of this project and the final solution.

The goal of this project was to come up with a new addition to the robot while maintaining the three main goals of the original BADDY project.

- Open-source
- Affordable
- Sustainable

This means that with the suggested improvement explained in this report, you can still make and build BADDY at home at low cost without the need for advanced and expensive equipment. For the rest of the BADDY community to benefit from the final solution given in this report, it has to be added to the open-source project and be easy to sustain.

The suggested improvement that was chosen to proceed with, was to build a remote controller that could replace the need for a smartphone when communicating with BADDY. The original goal of this project was therefore to create a remote controller and make it communicate with BADDY. When pressing a button on the controller, a shuttlecock should be launched by BADDY. Without any BADDY robot at disposal, this goal had to be redefined and simplified. An achievable goal was made, so it would be possible to conduct adequate testing and evaluation of the final result without the need for a BADDY robot.

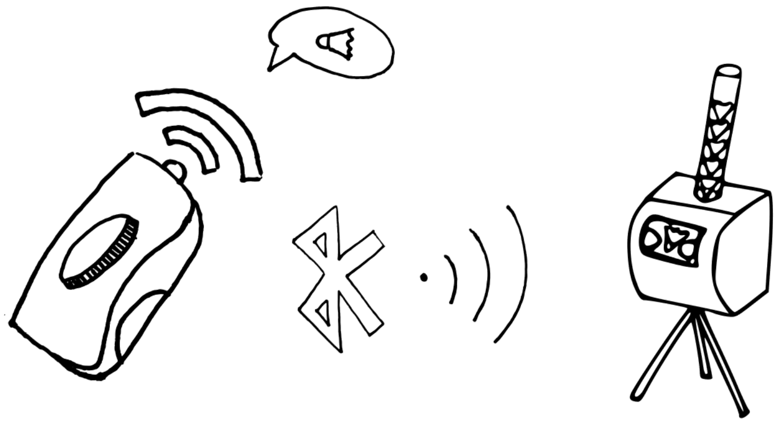


Figure 2: A remote controller communicating with BADDY

The original goal was therefore modified to building a controller, and make it communicate with an Arduino Uno board. To represent the launching of a shuttlecock from BADDY, a LED lamp was used instead. The LED lamp was wired to the Arduino board and programmed to light up when the button was pressed. The implementation of a LED lamp was made due to the lack of a real BADDY, but the solution developed to communicate with the LED light could be implemented into BADDY's Arduino code to launch a shuttlecock. To accomplish this new, main goal, it had to be divided into multiple, smaller sub-goals.

1. Making a component communicate not-wirelessly with a computer.
2. Make the component communicate with a smartphone using Bluetooth.
3. Make the component communicate with another Bluetooth component.
4. Connect two Bluetooth components to their respective, separate Arduino boards and make them communicate with each other through Bluetooth signals.

In this project, only a prototype of a remote controller has been developed. The final product and prototype did differ from the drawings provided in this report. This was done to simplify the model and make it doable within the given time of the project.

1.3 Motivation

In badminton, you want to practice your stroke. The easiest way to do this is to get a shuttlecock thrown perfectly at you a lot of times. A big problem in badminton versus other sports, like tennis, is the ball. In tennis, the ball is spherical with the center of mass in the middle of an even, relatively heavy ball. In badminton, a shuttlecock is used. A shuttlecock is made out of a base of cork with feathers, forming an open, conical shape. This kind of ball is light, with a total weight of 5 grams [64], and hard to throw because it needs a high output speed of approximately 100 km/h. A human will never be able to reach that speed by throwing the shuttlecock, and it will therefore be an advantage to have a robot that can do just that.

Figure 3: Shuttlecocks behave in a different way than regular balls [24]

Another problem appearing in badminton is that two beginners are not fit to practice with each other. They would lack the basic skills needed in order to hit the shuttlecock so the other one could receive it. This usually leads to either the need for one trainer for each player, or inactivity among the other players. To solve both of these problems, monotonous repetition of a specific stroke and teaching new players, a badminton shuttlecock launcher machine could be used.

Many different shuttlecock launcher machines exist on the market today, but most of these are big and expensive. In order to make the badminton machines more accessible for regular people and badminton clubs, an individual company called SPORTVATION, have developed a low-cost, shuttlecock launcher robot, called BADDY, and are selling DIY kits so you can make your own robot [8].

BADDY is continuously being improved and changed, and there are many different implementations and additions that could improve BADDY. This report has looked at different aspects of BADDY, found potential for improvements, and developed a solution for one of the suggested improvements by creating a remote control.

1.4 Report structure

It is assumed that the reader knows the basic concepts of mechatronics and the use of Arduino boards. Apart from this, the necessary background information is presented in section 2. Information about badminton in general, BADDY and technological aspects used later in the report, have been included. Section 3 presents a review of the methods used. In sections 4 and 5, the main work is presented, where possible improvements are described and discussed in section 4, followed by a discussion regarding the different suggested improvements. Section 5 is a detailed description of the process of developing a remote controller. All the choices made in the development phase are discussed continuously throughout this section. At the end of the report, in section 6, an overall discussion and conclusion are presented, with suggestions for future work.

2 Background material

2.1 Badminton

Badminton is a racket sport played by people all over the world. The sport is played by either two or four players on a court which is divided by a net. It is “the fastest racquet sport in the world” [23] and uses a cone-shaped shuttlecock, made out of feathers attached to a cork-tip as a ball. The combination of short, high-intensity rallies and longer rallies with moderate intensity, make badminton a challenging sport. It requires good technique, good reactivity, flexibility, strength, and endurance. A match is played as best of three sets, where each set is played to 21 points. To win a match, a player or team has to win 2 sets of 21 points each [46].

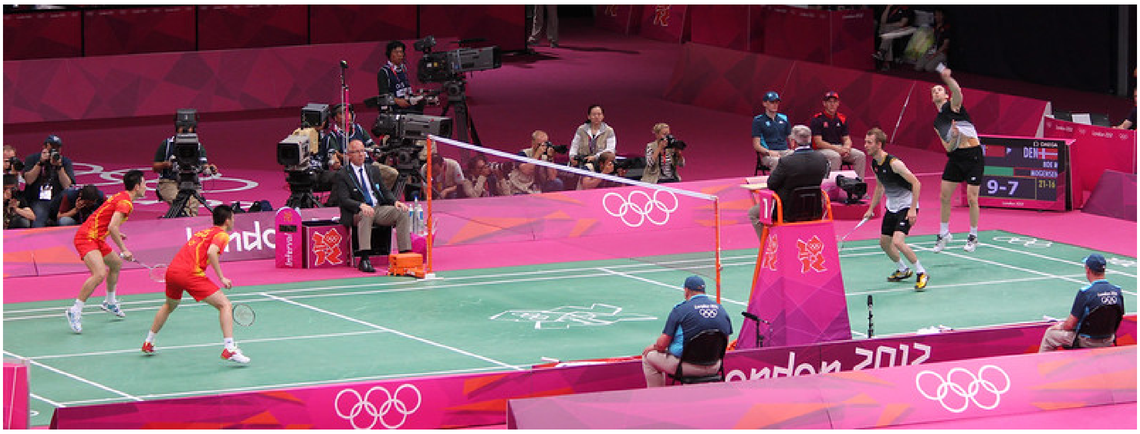


Figure 4: The men’s double final of the 2012 Olympics in London between China and Denmark [61]

A well-known problem for beginners is the lack of technique and the ability to keep a shuttlecock in play for a longer period of time. In order to develop the correct, and necessary technical skills to later be able to hit hard, either with long clear shots or smashes, a player must practice hitting shuttles that approach the player at a height of at least 0.5 meters above his own height. A beginner will find it difficult to send a shuttle in a controlled way to its teammate so that they can practice different strokes precisely because of the lack of technique. This will not be beneficial for either of the players since neither of them will improve their game. This is often solved by the coach hitting the shuttlecocks towards one player at a time, often named ”feeding”. This leads to inactivity among the other players.

Figure 5: Children waiting in line while the teacher is feeding them shuttlecocks [44]

"In teaching badminton one of the lesson objectives would be to keep a shuttle in play at the beginning level, but this is not an easy task for beginners. Because of unpredictable shuttlecock placements, beginners may have difficulty keeping the shuttle in play" [37]. A badminton shuttlecock training machine that launches shuttlecocks with high precision and accuracy would be suitable for this problem. By browsing today's market, it is possible to find a wide range of different shuttlecock launcher machines, but many of these are expensive and difficult to handle in terms of weight and size. It is possible to purchase robots in different price ranges and with different specifications, but the cheapest ones, comes at a price of around 1000 US dollars. Figure 6 illustrates a Siobasi badminton shuttlecock robot, one of the cheaper and more lightweight robots one can purchase today, and comes at a price of 1620 US dollars. The robot has a total weight of 28 kg, where the tripod in itself is 145 cm tall [2]. This is a high performance robot, but due to the dimensions of the robot, it is difficult to transport and set up. A more affordable and sustainable alternative exists in BADDY, which was developed to make such a robot more accessible to the public.

Figure 6: A Siobasi badminton shuttlecock robot [2]

2.2 BADDY

BADDY is an open-source badminton shuttlecock launcher. It was created by a team in France, with the purpose of creating a training partner for everyone to use. The developers had three main goals for the project. First, it should be accessible to all. The software and code behind BADDY is open-source, which means that it is publicly accessible, and can be both modified and used by anyone. Secondly, it should be affordable. As discussed in 2.1, there are several shuttlecock launcher robots on the market today, but these are all expensive with a cost anywhere from 1000 US dollars and up. BADDY was created as an alternative to these expensive robots, therefore the cost had to be significantly decreased. Finally, the robot was meant to be sustainable. If any parts were damaged, it should be possible to fix them at home. Either by purchasing new parts or by 3D-printing or laser cutting them. By being able to fix the robot, it is easy to maintain it. The user only has to change the broken part locally instead of having to submit the entire robot for service. This makes BADDY more sustainable both for the user and the environment. The user won't have to spend money and time submitting it, and the environment is spared the potentially unnecessary pollution that comes with the shipment. The total cost of BADDY is less than 500 USD and it requires annual maintenance that comes at the price of 50 USD. The maintenance can be performed by the owner. BADDY is also quite small with the dimensions of 25 x 20 x 20 cm without the feeder [13], which makes it portable and easy to bring along.

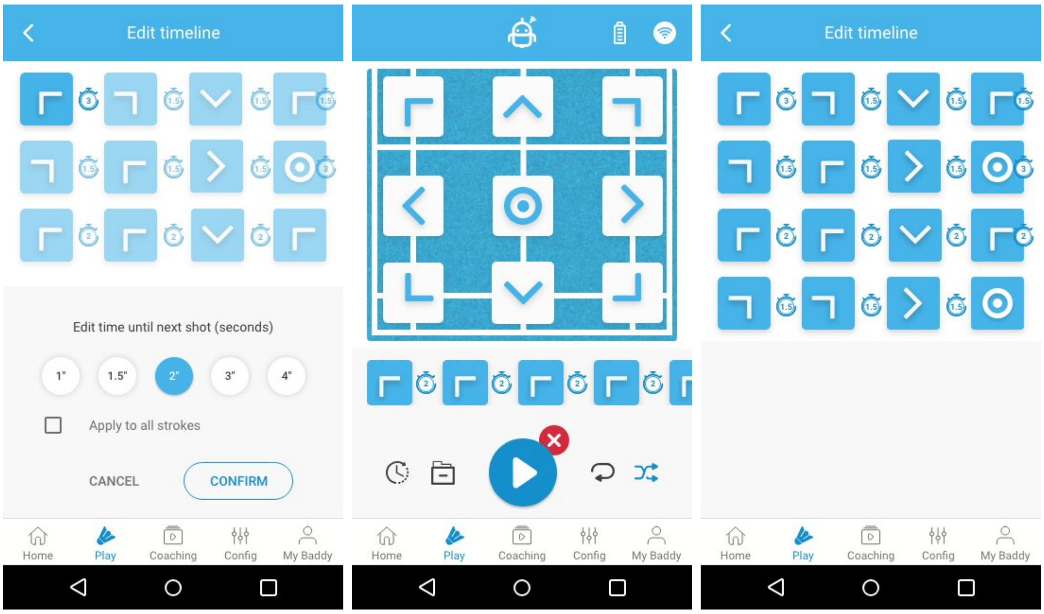
Figure 7: BADDY, the open-source badminton robot and the founder, Benoit Greslebin [8]

In addition to BADDY being open-source, there is also a BADDY community. People from all over the world come together via Facebook or Discord to exchange tips and tricks regarding the use of BADDY, mounting, and general questions.

It is possible to purchase two different DIY kits. One, where all parts are provided and the buyer only has to mount the robot, and one where some parts are not included and have to be 3D-printed or laser cut by the buyer. The later kit is cheaper than the full DIY kit, and comes at a cost of 265 USD [12]. The design files and software are available on BADDY LAB's Github page [11], and the BADDY team has created and made several "follow along" videos available on their YouTube channel [14].

Along with the robot, there exists a mobile application called BADDY that is available on both iOS and Android. The app allows the user to create playing sequences where the fire rate, type of shot, placement on the court and more can be both defined and changed. The newer version of

BADDY, BADDY V2, communicates with the app through a WiFi connection, different from the first version that used Bluetooth.



(a) (b) (c)

Figure 8: The BADDY application that can be downloaded from Google Play Store. Illustration of how the timeline of a playing sequence can be edited considering (a) time between strokes, (b) where on the court the shuttlecock is fired and (c) a complete overview of the sequence [63]

Considering the three main goals and intentions for the BADDY project, accessible, affordable, and sustainable, the use of an application supports an affordable solution. Most people already have a smartphone, and if not, multiple cheap options exist on the market. It is preferable for the user to hold the phone while playing in order to control BADDY from the app. This may be a problem, especially considering the size of smartphones. If the phone is too big, it may compromise the quality of the practice, and there is a risk of dropping the phone while playing.

2.3 Construction of BADDY

BADDY consists of three main components, the "Go Baddy" application presented in section 2.2, the robot itself and the electronics. The robot itself consists of multiple components. These components are mainly the frame, the fire room and the feeder tube, and can be seen in Figure 9. The feeder is placed on top of BADDY and is a long tube that holds the shuttlecocks that feed the launcher. It has a capacity of 30 shuttlecocks. The frame consists of a box and holds BADDY together. The frame can be made of different types of material, where both MDF and plastic have been used. This is a part that can be 3D-printed or laser cut by the user itself.

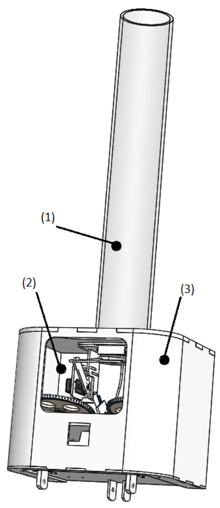


Figure 9: View of BADDY taken from Janton's CAD model [35]. BADDY consists of (1) feeder tube, (2) fire room, and (3) frame

The fire room is located inside the frame, and this room contains a firing system that launches the shuttlecocks in addition to the motors and electronics for the robot. The firing room consists mainly of a neck, switch, retainer and a launcher, as seen in Figure 10. Both the switch and retainer are operated by servomotors, which again controls whether a shuttlecock can be launched or not. These parts can either be in an open or closed position. When they are closed, no shuttlecock will be let through to the launcher. Once they are opened, a shuttlecock will be led from the feeder, towards the retainer and switch, before it reaches the launcher. The launcher is provided with two rotating wheels with separate motors that can spin up to 5000rpm. This allows for the two wheels to rotate at different speeds, which is necessary in order for BADDY to launch different types of shots. The wheels can be seen in Figure 11.

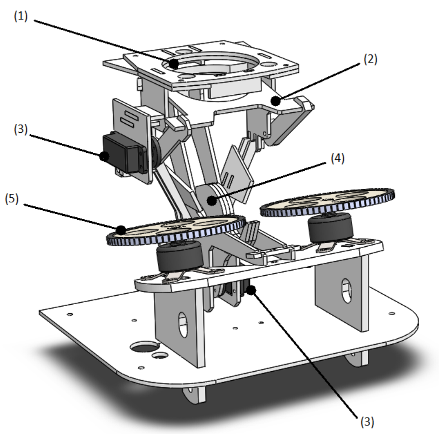


Figure 10: View of the fire room taken from Janton's CAD model of BADDY [35]. The inside of BADDY consists of (1) feeder hole and magnet slots, (2) neck, (3) servomotors, (4) retainer, and (5) wheels [8]

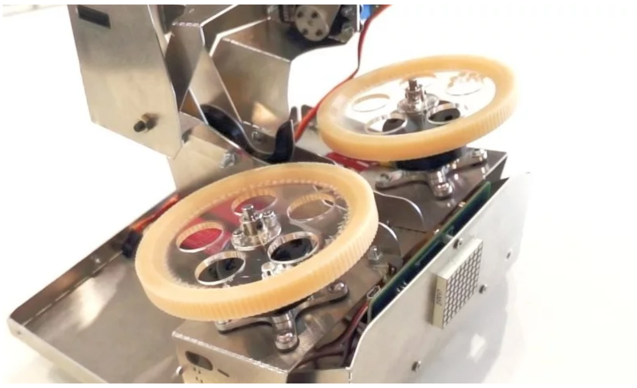


Figure 11: The wheels used inside BADDY [8]

BADDY is provided with 9 different strokes illustrated in Figure 12. These are drops, drives, and clears, where all of these can be fired in 3 different directions, left, center, and right.

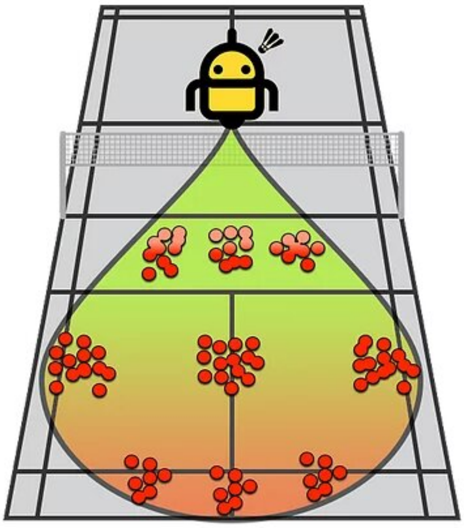


Figure 12: Illustration of the court coverage provided with BADDY's 9 different types of strokes.

BADDY is equipped with a Printed Circuit Board (PCB), as shown in Figure 13. This circuit board holds an ESP8266 WiFi-based Arduino board with different connectors. To provide power to the servomotors and the motors for the wheels, they have to be connected to the circuit board when mounting BADDY. The neck and retainer must be positioned correctly. To be able to assure the correct position of these, the code behind these components have to be included in the Arduino code and initialized to the correct value. BADDY also comes with a 12 V Li-on battery and a charger. The LED panel in front of BADDY, shown in Figure 13, notifies the user when the battery level is under a critical threshold and BADDY needs to be charged.

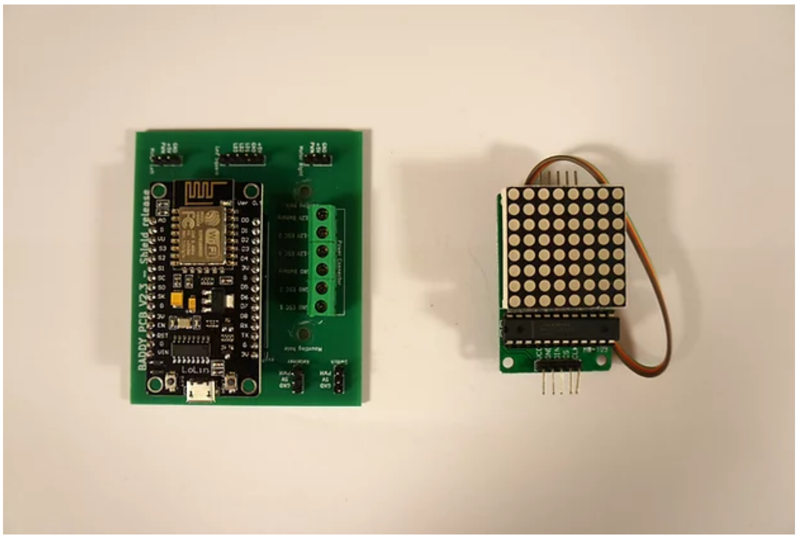


Figure 13: The electronic components used in BADDY. The circuit board on the left is a special designed circuit board with Arduino embedded, and the Arduino LED panel is shown on the right [8]

2.4 Previous work

A report from 2019 written by Hugo Janton [35], addresses, at that time, the latest version of BADDY and possible ways to improve the robot. The main aspect of his report was that the suggested improved design of BADDY should be both affordable, transportable, and usable for everyone. As documented in his report, several elements were considered for improvement and three improved designs were suggested. These were designing a new firing system that would handle variations in the initial angle, increasing the capacity of the feeder to allow for an increment of shuttlecock capacity. And finally the implementation of a safety system in order to create a safe environment around BADDY [35]. An example of his work is shown in Figure 14. As part of the study, Janton made CAD files based on the .dxf files for laser cutting parts that are available on BADDY LAB's Github.

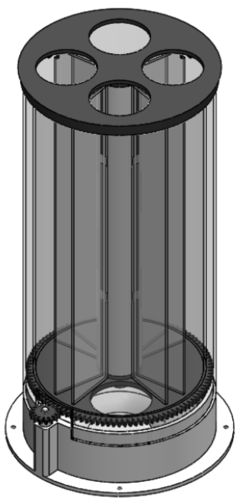


Figure 14: CAD model of a four-barrel rotary feeder Janton designed for his report [35]

2.5 Bluetooth and Bluetooth Low Energy technology

The first version of BADDY used Bluetooth to communicate with the application, but this was replaced with WiFi for the later version. For a wireless remote controller, it would be preferable to reimplement this with new Bluetooth technology that is more optimized for the use of a remote controller.

"Bluetooth is a wireless technology that allows the exchange of data between different devices" [36]. It is a personal area network that works within a short range and allows for communication between different devices [39]. Bluetooth Low Energy (BLE) was released in 2010 as a part of the core specifications of Bluetooth 4.0. BLE is not an upgraded version of the original Bluetooth, but rather a new technology that focuses on applications and devices that are related to and uses the Internet of Things (IoT) [25]. These applications only transfer a small amount of data with low speed, which implies that they require less energy than other devices to transfer data. BLE is a wireless technology that focuses on low power and is mainly targeting devices that communicate within a short range of each other and periodically transfer data [1]. BLE only consumes half of the energy that the original Bluetooth consumes and is therefore optimized for i.e. home automation applications, fitness trackers or smartwatches, which all require low energy and only transmit small amounts of data [25].

3 Method

This chapter addresses how the information provided in this report has been gathered, which research methods have been used, and how they were conducted. This chapter also addresses why the following methods were used.

Several methods were used to provide the necessary knowledge in order to find possible improvements for BADDY. The study was conducted by research and a feasibility study due to little, to none, prior knowledge. In addition to this, an interview with the founder was arranged and a previous project report [35] was also used as part of the prestudy.

3.1 Interview

In order to achieve a deeper understanding of the idea and work behind BADDY, an interview with the founder, Benoit Greslebin, was arranged. Due to distance, given his settlement in France, and COVID-19, the interview was conducted using Skype. Prior to the interview, Mr. Greslebin was provided with some questions in order for him to prepare himself for the interview. The interview was based on the questions Mr. Greslebin was provided with, but other topics were also raised and discussed.

3.2 Literature study

A literature study was used to acquire the necessary knowledge regarding BADDY and relevant topics for the study. This was mainly implemented at the beginning of the project, but also throughout the project as new information emerged. A lot of information is available online, so it was important to find and use credible sources. NTNU's digital library service "Oria" was mainly used to find relevant articles and information. Subjective opinions and input have contributed to the study and the end result by introducing new ideas and points of view.

The CRAAP test was used to assess the trustworthiness of the sources obtained. "The CRAAP acronym [...] stands for the components of the evaluation process: currency, relevance, authority, accuracy and purpose" [43]. The CRAAP test is based on a set of questions and guidelines to help evaluate the sources and information collected. Sources were evaluated based on their reliability and validity. The reliability is closely related to whether the information is gathered in a trustworthy way or not. Here, both the publication date and author have been assessed as well as the information has been verified with other sources. The validity of the source says something about whether the information obtained is relevant to the study or not. Finally, the purpose of the source has been taken into account. The purpose of the article or the intent of the author has also been part of the overall evaluation of the source.

3.3 Previous work

A previous student's project report was made available and used as inspiration and as part of the research for this study. This was the report made by Hugo Janton mentioned in section 2.4. The thesis was used to get an insight into what has been done previously and how an open-source project can be further developed by others than the founders.

4 Possible ways to improve BADDY

BADDY is an open-source mechanics, electronics and software project for a low-cost badminton-shuttlecock throwing robot. The BADDY community wants to develop BADDY further. The project idea is to not only develop a good badminton shuttlecock launcher, but also make people more interested in robotics and be able to build robots themselves. There are many aspects to the relativity new BADDY project which gives BADDY a lot of potential for improvement. As a user of BADDY, you are encouraged to provide feedback and contribute to the project. The project can be divided into two parts. The first one consists of learning everything you need for building a robot. Some of the things you get to know are electronics, coding, physical building, and obtaining the necessary tools. Secondly, the final product will improve your badminton training.

In this section, some possible improvements are explained and discussed. BADDY has virtually unlimited opportunities for improvement, but the main focus of this report was to improve the robot itself, and not the process of learning from building it.

4.1 Improvements

When deciding how to improve BADDY, several things have to be considered. The goal is to keep following the intentions behind BADDY, the improvements should be open-source, easy and cheap to build, and be educational both considering building a robot and playing with BADDY.

Three main categories have been considered for improvements: security, performance and ease of use. Each improvement is categorized under one of them, but some may fit under several.

- Security
    1. Adding sensors to detect humans
    2. Adding a mechanical security shield
- Performance
    1. Adding several tubes to increase the shuttlecock capacity
    2. Expanding movement and range of motion
    3. Material improvement
- Ease of use
    1. Adding a new way to control BADDY
        (a) Adding a 3D-camera control system
        (b) Adding a remote controller

An alternative way to control BADDY is to use a voice-activated control system. This could be an interesting way of controlling, but is not discussed further in this report.

4.1.1 Security

As for now, the only security aspect of BADDY consists of a written disclaimer that is agreed upon with the purchase of a BADDY. In the disclaimer, it is stated that “Kids shall not use BADDY without surveillance, never put your hands in it when switched on, stay at the other side of the court when playing (Keep the net between you and BADDY)" [9].

BADDY can launch shuttlecocks with a maximum speed of 130 km/h [8]. A shuttlecock with this speed can do serious damage to the eyes or other parts of the body. It lies is in human nature to explore and tinker with new, exciting things, especially for children. Considering that BADDY is a robot and may cause harm to those around, it is reasonable to assume that no children are allowed near the robot and that it can only be handled by adults.

Different safety aspects could be added to BADDY in order to increase the security and the need for supervision. Depending on the upgrade, it could make it harder to get injured when using the robot.

Sensors to detect humans

BADDY is composed of moving parts, where the motors can spin at 5000 rpm and the shuttlecocks can reach a maximum speed of 130 km/h when launched [10]. Considering the risk and dangers connected with the usage of BADDY, a security mechanism implemented in the robot would be preferable. By installing ultrasonic distance sensors, as shown in Figure 15, BADDY would be able to detect humans in front of the launcher. If the sensors detect humans too close, the robot should stop all launching until the zone is cleared. This would be a software mechanism and implemented into BADDY's source code. It would also be possible to use infrared sensors instead of ultrasonic ones. Which one to chose was discussed in section 4.3.1.1 in Janton's report [35]. Here he states that ultrasonic sensors are more suited for BADDY because of a wider range of detection.

Sensors can also be placed to monitor particularly vulnerable components like the launching wheels or motors. When the sensors detect any defective components, it should stop all launching.

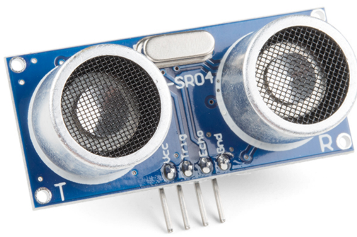


Figure 15: HC-SR04 ultrasonic distance sensor [59]. A sensor that can measure the distance to objects, and can be connected with Arduino [34]

Mechanical security shield

In addition to the sensors, the implementation of a closing mechanism that would close the launching hole has also been taken into consideration. It would work as a shield blocking shuttlecocks from launching. A shield would also stop potential plastic or metal parts from being launched if some of the components within BADDY would break. Additionally, a shield will prevent children from sticking fingers into BADDY's mechanism.

The two safety aspects of using sensors and a closing shield can combine to make BADDY even more secure. They would work as backups for each other. If the sensors detect any humans in front of BADDY, the shield would close. This would add a physical mechanism and not only a software mechanism to avoid launching.

4.1.2 Performance

Adding several tubes to increase the shuttlecock capacity

The tube provided with BADDY, as of today, has a capacity of 30 shuttlecocks [13]. Considering the fact that BADDY has a maximum stroke frequency of 2 strokes per second [8]. It will take 15 seconds before the tube is empty and the player has to take a break to refill the tube. From a player's point of view, this would be time-consuming and result in an ineffective training session.

Due to the low capacity of the tube, an improvement of BADDY could consist of adding several tubes to increase the capacity. Then the player doesn't have to pause the training as often as with the original, single tube.

Figure 16: The shuttlecock holder for Siobasi's machine with a capacity of 180 shuttlecocks [2]

BADDY is a small robot, and the dimensions of the components were thoroughly calculated in the development process, in terms of the weight, robustness, and strength to withstand the power generated by the motors. When considering an improvement of adding multiple tubes, the tubes themselves would cause a weight gain to BADDY. I addition to the tubes, the shuttlecocks will also increase the weight that BADDY has to withstand. Depending on how large the weight gain of this implementation would be, it could result in damaging the robot due to the added weight. When considering the implementation of adding tubes and increasing the shuttlecock capacity, it is important to consider the weight gain and if it would be necessary to build BADDY stronger so it can manage the new weight.

Expanding movement and range of motion

BADDY has 9 different strokes available, as seen in Figure 12. The drop and clear shots have less divergence than the drives or the middle court shots. The Figure illustrates that these shots don't reach the corners at the rear and back of the court. "In order to reach corners at the rear of the court, you have to orient BADDY towards the Left/Right side of the court you want to focus on" [13].

BADDY can be placed directly on the floor as well as on top of a tripod. Placing the robot on the floor optimizes the path for drop shots, but the reach for clear shots are reduced and can be missed by up to one meter [13]. Attaching BADDY to a tripod makes it possible to launch shuttlecock from a more realistic height and the outlet angle for the shuttlecocks allows for better clear shots than without the tripod.

A playing sequence often consists of multiple different shots. This may be a problem both with and without the use of a tripod, considering the limitation of divergence in the clear and drop shots. Regardless of whether BADDY is placed on the floor or on the tripod, some of the shots will only be approximately equal to a real shot, but not exactly the same. BADDY is a lightweight robot with a total weight of less than 5 kg [13]. It is therefore easy to move around, which makes it possible to recreate most shots. The disadvantage of this is that someone physically has to move BADDY and adjust the settings for each time it is moved.

A solution to this may be to implement new functionality to BADDY which allows for more movement both sideways and up and down in order to adjust the launching angle. This would make BADDY more flexible in regards to the different shots and the total coverage of the court can be enhanced.

Material improvement

So far, two versions of BADDY have been developed and multiple materials have been tested and used for the robot. The first version, BADDY V1, was fully made out of plastic, specifically PMMA, ABS and PLA [13]. When the robot is operative and the shuttlecocks are launched, a large force is generated. Plastic is a fragile material, and together with the large force that is generated, this can be dangerous and the plastic may break. Plastic is also the main material that has been used for 3D-printing of the parts.

As an improvement to the performance and sustainability of BADDY, wood materials were implemented and tested. The parts for the frame and box were made of reel wood veneer mouldings, while the internal structure was created by laser cutting MDF [13]. Wood is more durable and elastic than plastic and can bend without breaking. This led to a more sustainable BADDY, but like plastic, wood material also has its weaknesses. “The interaction with water is one of the most critical problems found in MDF since this results in higher swelling of the material” [20]. The use of MDF would not be a good solution if there is a possibility that BADDY might be exposed to water.

Figure 17: BADDY created in wood veneer and MDF [15]

The second version, BADDY V2, was launched with some new functionalities as well as new material. The inside parts for this version were made of metal. Metal is much stronger than both plastic and MDF, and was, therefore, a good alternative to the previously tested materials. There is a much smaller possibility for metal to break and injure someone, compared to plastic. A disadvantage of this new improvement was that the parts have to be made with a laser cutter. The average man doesn't have access to a laser cutter and will not be able to make the parts himself, using for example a 3D-printer. If a part of BADDY breaks, it would be hard to create these parts and they would have to be purchased from the producer. This improvement makes BADDY less of an open-source project since these parts mostly have to be bought.

A possible improvement to BADDY would be to consider the use of new, durable material that is lightweight, easy to access, and can sustain the weight and force needed for BADDY to be a safe and user-friendly robot. For such a change, a possible material would have to be examined and it would be desirable to obtain a material that the average man can access himself, without having to purchase the parts from the producer.

4.1.3 Ease of use

Adding a new way to control BADDY

Currently, a smartphone application is used to control BADDY. This requires the player to have access to a working smartphone. Considering the three main goals for the BADDY project; accessible, affordable and sustainable. The use of an application supports an affordable solution as long as most people already have a smartphone. Most people do own a smartphone today, but it may be a disadvantage that a smartphone is needed in order to use BADDY. A BADDY will be useless if the phone is e.g. left at home or out of power.

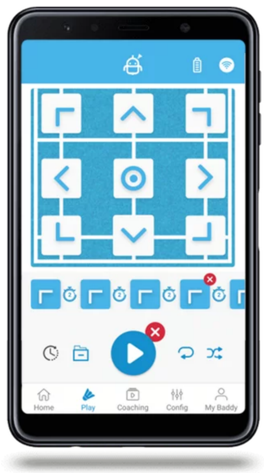


Figure 18: Screen photo of the application GO BADDY [8]

In order to use the application, it is highly preferable for the user to hold the phone while playing to control BADDY. This may be a problem, especially considering the size of smartphones, which are getting bigger and bigger, and it may be hard to press a button without looking at the screen. If the phone is too big, it may compromise the quality of the practice. Usually, when playing badminton the hand without a racket is in a lot of movement and in special positions. A big phone can be annoying to hold on to and may easily be dropped. There is also a bigger risk of getting sweat or water into the phone.

Adding a new way of controlling BADDY, could solve some of these issues. Depending on the solution, it may decrease or increase the number of functions an application is providing. In the application, you have the opportunity to create playing sequences and decide exactly where the shuttlecock should land. Some of this functionality may be lost with an alternative way of controlling BADDY. On the other hand, a new way of controlling might add other elements and/or make BADDY smarter without the need for the same type of control.

3D-camera control system

The development of a three-dimensional sensor technology has contributed to major changes in various industries and the demand for this technology is increasing. "The applicability of 3D-sensing technology has expanded in the robotics industry, owning to the increasing demand for virtualized solutions" [33]. A 3D-camera is capable of capturing movements and distance to a person or an object [28].

Implementing a 3D-camera in BADDY can increase both the user experience and the usability of the robot. A 3D-camera can detect the movements of the player as well as detecting where on the court the player is at this exact moment. With this information, BADDY can decide where and when the next shuttlecock should be launched. The use of 3D-sensing technology can improve the robot in multiple ways and it opens up new ways to use BADDY.

The 3D technology could increase the usability of BADDY by allowing a player to train without having to create a playing sequence in advance. Hence, the player only has to set up BADDY and start playing without spending time on creating the sequence. This could also increase the difficulty of the training since the player won't know where the next shuttlecock is coming from.

It is beneficial for a player to receive feedback on technique and areas to improve from a coach in order to evolve as an athlete. This has traditionally been done through oral feedback from coaches, based on their observations. Depending on the complexity of the exercise and playing sequence, it can be hard for the coaches to see the entire playing sequence and to give feedback on every part of the exercise [42]. Implementing a 3D-camera to BADDY, would not only allow for BADDY to detect players on the court, but also open up to the possibility of filming the players. These films can be used to analyze and improve the player's technique.

Remote controller

A new addition to BADDY could be to create a remote controller that would communicate with BADDY directly with the use of wireless technology. By implementing a remote controller, it will be possible to control BADDY without the need for the application. The application that is currently in use, is connected to the robot through a WiFi connection, something that is a good solution for this type of application. WiFi is a wireless technology that is limited by both short-range and high power consumption [26]. In order for a remote controller to be beneficial over a smartphone, the controller would have to be easy to use and small enough so that it is easy to hold on to. It should also come with long battery life.

The use of WiFi in such a controller could be a solution. This would make the implementation quite easy considering that BADDY is already embedded with a WiFi device. Considering the limitations of WiFi and high power consumption, this would not be a preferable solution. Another technology such as Bluetooth or Bluetooth Low Energy could be a better choice. The first version of BADDY was created with a Bluetooth connection in order to connect with the application. "Bluetooth was originally designed for continuous, streaming data applications" [50]. A disadvantage when it comes to Bluetooth, like WiFi, is that it has a high power consumption. Bluetooth Low Energy would therefore be another option and possibly a better one than both WiFi and Bluetooth. As mentioned in 2.5, BLE is a low power technology and works well with a device that doesn't have to send a large amount of data [50].

A remote controller must be able to signal BADDY to launch a shuttlecock. The number of buttons on the controller may vary. If the controller only has one button, the placement of the launched shuttlecock should be predefined. If it has multiple buttons, they could represent the different strokes, such as the application provides. A remote controller would only need to transmit data when a button is pressed so that the robot knows when to perform an action. This would be a small amount of data that is only transferred periodically, not continuously. If a remote controller should be added to BADDY, it would be preferable to do so, using BLE to connect the two parts. A disadvantage of using BLE would be that the embedded Arduino code in BADDY would have to be modified in order to use Bluetooth, but this would most likely pay off for the end result. Additionally, the hardware in BADDY has to be updated with a Bluetooth device.

4.2 Choice of improvement

The rest of this report will go more in-depth on one of the suggested improvements discussed in section 4.1. The solution selected to implement, a remote controller, was chosen based on the time frame for the project and the desired learning outcome. In order to implement the chosen solution, it was required to acquire knowledge about both hardware and software development, as well as 3D-printing and programming, in order to make two components communicate with each other wirelessly.

The possible solutions presented in section 4.1.1 that focuses on the security aspect of BADDY would, to a greater extent than the chosen solution, depend on the presence of a BADDY. Given the circumstances of the project and the lack of a BADDY robot, it would be difficult to complete and implement these solutions. These improvements would also acquire access to a workshop and miscellaneous equipment, which could lead to problems in regards to accessibility due to the restrictions given COVID-19.

The improvements discussed in section 4.1.2 that suggested to improve the performance of BADDY, would also require the use of a BADDY robot, at the same time as it would be necessary to acquire a lot of new knowledge within multiple areas. Given the time frame of the project, these solutions would not be preferable as the chosen solution to further implement.

By adding a 3D-camera to BADDY, as presented in 4.1.3, the total cost of the robot would increase, which would affect one of the initial goals of BADDY, namely that it should be affordable. If the 3D-camera should break, it would not be possible to fix it without purchasing a new one, something that also would affect the sustainability aspect of the robot.

The solution chosen to implement as an addition to BADDY was the remote controller. This was the suggestion that best suited the different criteria given for this project and the circumstances.

The implementation of a remote controller required the acquisition of knowledge within many areas as stated in section 4.1.3, which would provide a steep learning curve. As mentioned in section 1.2, the lack of a BADDY entailed that the main goal of creating a remote controller to communicate with BADDY, had to be redefined in order to complete the project. The goal for this project then became to make a remote controller by using a 3D-printer and an Arduino board. To illustrate BADDY and to test if the controller worked, a LED light was used. Even without a BADDY, this was a realistic goal for the project and provided a natural development for further work.

5 Implementation of a remote controller

The chosen improvement was to build a remote controller to replace the use of smartphones when communicating with BADDY.

As mentioned in section 1.2, the main goal of this report has been to get a controller to communicate with an Arduino board. This goal was set because a BADDY robot has been unavailable during the time scope of this project. When pressing a button on the controller, a signal has to be transferred to an Arduino board. To check whether the remote controller is sending signals to the Arduino board, a LED lamp attached to the board will be lightened. This process is illustrated in Figure 19.

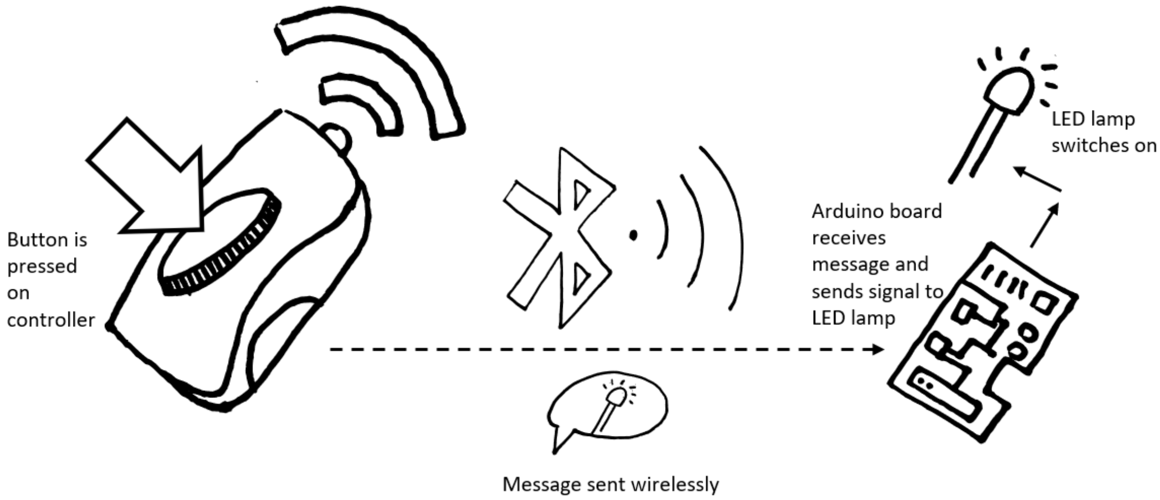


Figure 19: Illustration of signal transmission

There are three main processes to consider when making the controller and they are all carefully described in the next sections.

1. Case design
2. Hardware development
3. Software development

5.1 Technical requirements

A badminton court has a total length of 13.4 meters and a width of 6.1 meters [38]. This gives a diagonal length of 14.3 meters. The controller will therefore mostly be used within a range of 15 meters. If BADDY would be controlled by a trainer on the side, the total range should reach at least 20 meters.

It would be required to use a battery as the power supply to avoid the controller to be constantly wired to an external power source. Wires would lead to restriction of movement for the player. Using a rechargeable battery will lead to the need for more equipment and increase costs. As long as the hardware doesn't drain too much power, a replaceable battery will suffice. The use of a battery easy to get hold of, small in volume on and reasonably cheap, would be benefiting.

When determining which communication type to use, battery consumption would be crucial. Three different communication types were considered, WiFi, Infrared (IR) and Bluetooth. Infrared has a short-range. Even the best IR transmitters don't work well over a range of 5,5 meters [21]. The short-range makes Infrared unsuitable for this controller.

BADDY is now using WiFi to connect with phones and to other BADDY robots. Using WiFi with the controller would then simplify the needed modification of BADDY's circuit board. WiFi communication drains the battery fast, and might therefore not be the best solution.

A previous version of BADDY used Bluetooth to connect with phones. According to Mr. Greslebin, this was undone because Bluetooth versions on phones were regularly updated, which required equally regular updates on BADDY's hardware as well. New versions of Bluetooth are backward compatible with older versions, so this explanation makes little sense. The communication will be between a BADDY and the controller which will act as two fixed devices, instead of many different phones. Bluetooth can therefore be reintroduced as a better alternative to WiFi. Bluetooth provides higher security and requires less power than WiFi. With the Bluetooth Low Energy (BLE) technology, the battery time will increase, and the use of BLE for communication would therefore be the best alternative.

5.2 Case design process

When designing the case for the remote controller, three things were considered. First, it should impact the gameplay as little as possible. The controller is a substitution for a phone, so the controller had to solve the problems by playing with a big phone. Secondly making the case still has to follow the main goals of the BADDY project; open-source, affordable and sustainable. Therefore it should be easy and cheap to make it with open digital drawings. Also considering adding a proposal for an alternative cheap case that's easy to buy online in case the user doesn't have the opportunity to make it themselves. Thirdly the design must be big enough to fit the hardware and be possible to make with appropriate tools.

When considering the impact the controller has on the game, it's necessary to look at the hand the player will use it with. This will be the player's non-racket arm, usually the not dominant hand.

5.2.1 Non-racket arm impact when playing

The non-racket arm has a big impact on the game. It is mainly used for balance, but also to increase the power of the hit, and aiming where to hit [16, 19, 67]. The arm is in a lot of movement, usually with an open hand. It will therefore be disturbing the natural movement and use of this arm by having to hold on to a controller.

5.2.2 Practical design

A case should smoothly fit into a hand. It should be small, light and easy to hold on to. And it must withstand rough treatment. If the controller is attached to your hand, the need for gripping is no longer necessary. Without the need of gripping around a case, the non-racket hand could be used in the same way as in normal play. No need of being afraid of losing the controller anymore. To attach the case to the hand, a rubber band could be used.

The controller also needs a button in order to launch a new shuttlecock. To accomplish the main goal, it's only needed one button on the controller, but the possibility to expand this will be discussed later. The button should be reasonably big so it is easy to hit. If you can use several fingers to hit the button, the player can choose which fingers he/she prefers and therefore make it more flexible and usable. The placing should be in the middle of the palm with the rubber band stretching around the outer side of the hand holding it in place. This will be within reach of both the middle fingers and thumb. The design should make the placement of the case at such a level that the fingers automatically will hit the button when closed. Since the controller should affect the game as little as possible, it must be easy to use the controller and hit the button without much thinking or the need of looking at the controller. There will be a trade-off between making the case small (and unnoticeable) and making the button big and easy to hit. The most useful balance must be found.

The rubber band is a one size fits all. It has a velcro on each end for closing, opening and adjusting size. It is essential that the rubber band is tight so you don't have the feeling of losing the controller, but also not too tight that you can harm the hand. In case the rubber band gets broken, it has to be possible and easy to change it.

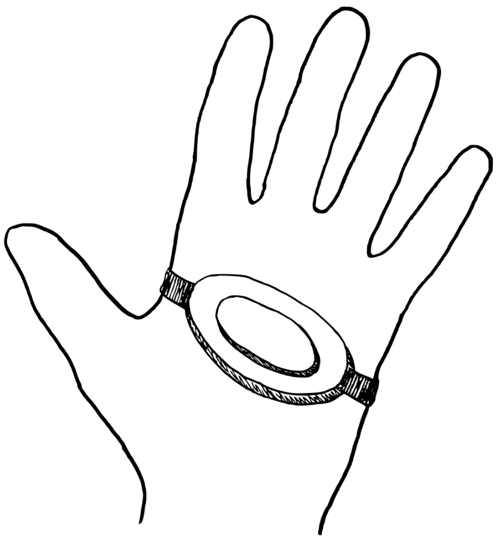


Figure 20: Illustration of imagined design of a small round sphere case with rubber band around backside of the hand

To be able to place hardware inside the case, it needs some sort of opening and closing mechanism. The solution can not be permanent, as it is necessary to change the battery from time to time, and should be easy to access the components inside. This excludes the use of glue or other permanent closing mechanisms. If the case consists of two parts that fit inside each other and are kept closed that way, the use of parts and tools needed will be minimized. Another option is to use screws to keep two parts together. Using screws will be more robust, but also require more tools and make the 3D-model more complicated. A 3D-printing test done later with screw holes showed that this was not an optimal solution.

5.2.3 Making the case

Different options were considered for making the case. It could be possible to use laser cut pieces, make a casting form and cast the case in fiberglass or foam, or make origami dice out of paper.

The casting was excluded because of lack of time and tools, and laser-cut parts and paper origami was too fragile. Instead, 3D-printing was chosen. The main reason was that BADDY already consists of parts that can be 3D-printed, so the controller will not add an extra need for tools. Additionally, 3D-printing is fast, easy to learn and able to do without much research[1].

To test 3D-printing a model made by Daniel Maydana [40] was printed. A button was applied to the 3D-model before printing. The result was poorly printed as shown in Figure 21, but a lot of experience could be taken from it. The support used when printing was hard to remove without breaking important parts and made the closing mechanism for the battery not work. The screw holes were filled with plastic and unusable. The button showed how badly the printing can be even on a small, uncomplicated part when something wrong happens to the printer. The printers used was unstable and unfinished parts occurred many times. A too complex model is not doable on the available 3D-printers and the mechanism for closing the case and changing the battery has to be made simple.

Figure 21: 3D-printed test case. (1) Filled screw holes, (2) support mass not possible to remove, and (3) printing gone wrong

To make the case easy to reproduce, a 3D-model was made in the program *Autodesk Fusion 360*. The experience from the test printing further influenced the design of the case. The first prototype was simplified from the sketched sphere to a cube. This was done for several reasons. Mainly to make it easy to model in the program, to test different types of buttons and to make sure the dimension was right for fitting the hardware. Also, a spherical case can be more problematic to 3D-print, due to the general rule of not having an overhang with an angle larger than 45 degrees [22]. A 3D-printer can not reproduce overhang structures with a larger angle than 45 degrees from the vertical without using support structures. Printing with support structures takes much more time and as experienced, the support structure can be hard to remove. Due to a lack of available 3D-printers, this was not done.

[1] The 3D-printers were not made available as planned due to COVID-19, so the lack of them delayed the project for some weeks

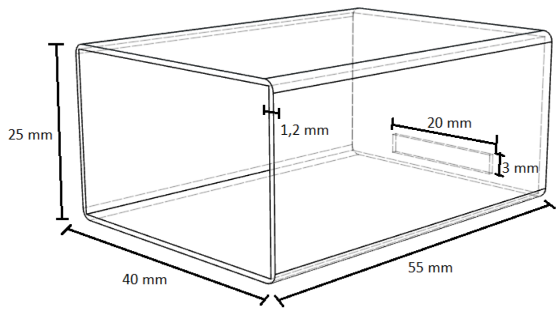


Figure 22: Design of 3D-printed case with dimensions

To avoid designing the closing mechanism on the prototype, one side was removed to make it easy to install the hardware inside. Making the closing mechanism should be done when a better prototype is present. One rectangular hole was made for the rubber band to fit into.

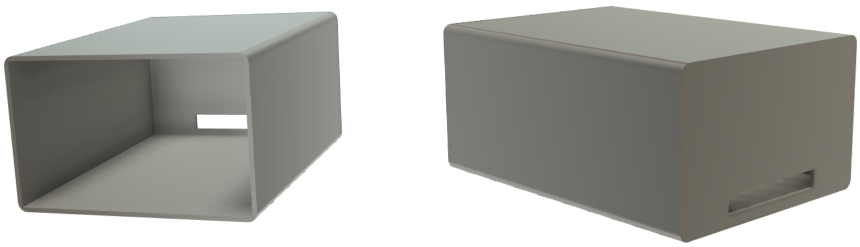


Figure 23: The inside and backside of 3D-printed case

Two types of button functionalities were tested. The first was a separate button with dimensions such that it would fit partly through a hole in the case. Figure 23 shows this. The second was inspired by the button on a computer mouse. The button would not be a separate part, but an extension of the case only attached on one side. This is shown in Figure 25. This second button was proved to be difficult to achieve since the material used was not very elastic. The button broke several times when removing support material. It worked better when making it thinner, but that also made it more fragile.

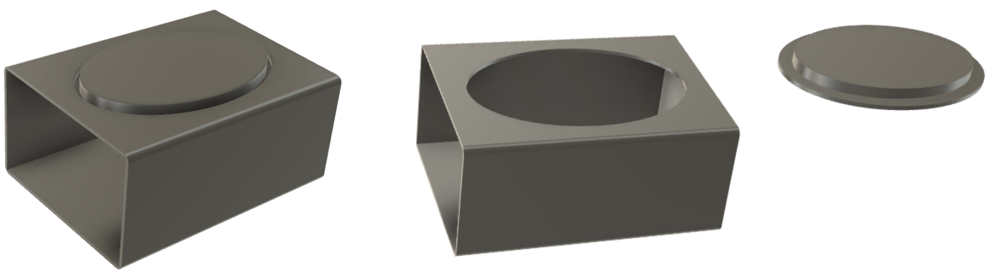


Figure 24: Case with a separate button

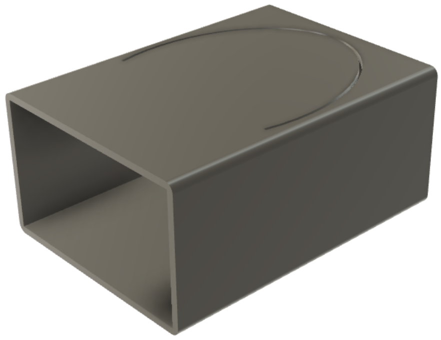


Figure 25: Case with an extension button

When the 3D-model was ready, an STL-file was made and imported into the program *PrusaSlicer*. This program was used because it works with the Prusa 3D-printers that were available. Here the orientation and placement of the model are selected, as seen in Figure 27. The program calculates the time needed for printing and adds supports where the overhang angle is over 45 degrees. The correct printing settings have to be selected, type of printer, type of filament and thickness and quality of the layers. A picture of the settings is shown in Figure 26. When deciding the type of filament you have to select the material. The case was made in Prusament PLA [49] as this was the only available material. This is a very common printing filament. Because of its low melting temperature, 175 degrees Celsius, it is suitable for printing tiny parts, like the case made here. It is also biodegradable material which is a pro to today's societal demands. The case had to be made a lot of times to determine the best design, so by making it in biodegradable material, those not used could be recycled. Prusament PLA is not the easiest to post-process, because of its characteristics as hard, tough and brittle. This was experienced when printing the case and made it difficult to remove support fragments. This led to higher demands for the digital 3D-model.

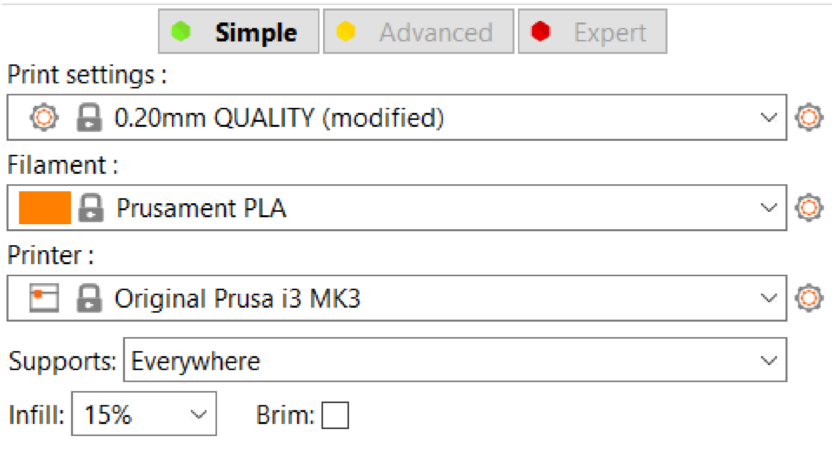


Figure 26: Settings of the print with Prusa Slicer

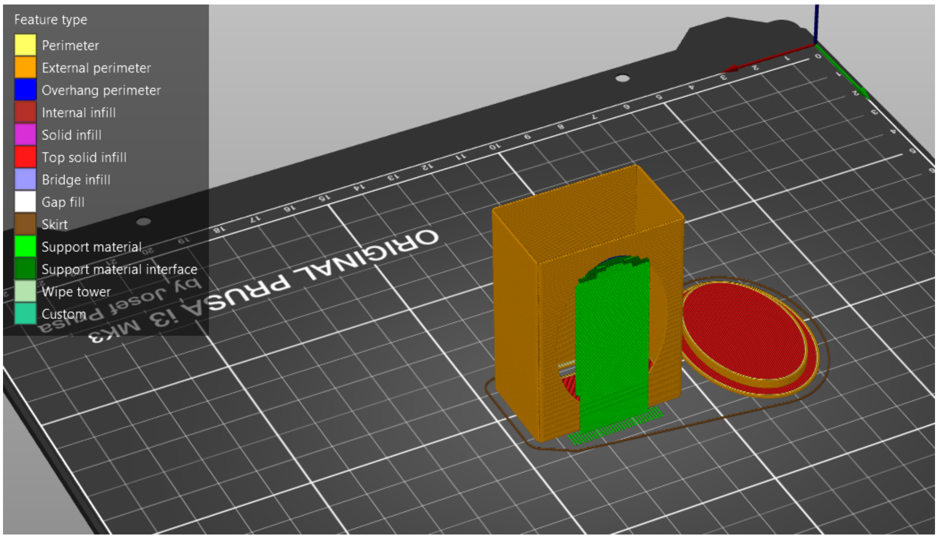


Figure 27: Orientation and support material of the 3D-printing

When the settings were chosen, the model was sliced and exported as a G-node to an SD-card. This was inserted into the Prusa printer used. Three different versions were available, *Prusa printer MK3, MK3S* and *MK2.5*. All of them were tried out, and the best result was produced by *MK3S*. This is the newest of the three printers and has been upgraded with a redesigned filament extruder. The extruder on *MK3S* has a hybrid system with both an optical filament sensor and mechanical lever [3].

The designing process alternated between the two steps (1) designing a 3D-model in *Autodesk Fusion 360* and (2) printing with Prusa. Printing was not straight forward, as the printers had problems with several things. Layer shifting, poor bridging, first layer issues, and extrusion problems were experienced. All of them can be read about in Prusa Knowledge Base [17]. Some of these problems led to the need of changing the 3D-model, like the design of the buttons. Testing and feeling the case in the hand, also made it easier to decide where to do changes. The thickness of the walls making the case had to be changed two times, from too thick (1.5 mm) to too thin (0.75 mm), to the final result of 1.2 mm. The thickness was determined by printing and testing several times and was a trade-off between a solid case and a light, fast printed case. The button design had to be redesigned a few times. Both the thicknesses of the button and spacing between button and case had to be changed due to the outcome of the previously printed cases.

The final design with the printing settings from Figure 27 resulted in a bit less than 2 hours of printing time. The exact printing time was 1 hour and 44 minutes, and in addition to this, the printer uses a few minutes to heat up and cool down before and after printing.

The mechanism to attach the controller to a hand is made of a rubber band with a two cm width. A velcro is sewn at each end of the rubber band so it can close with varying lengths. This makes it flexible and should fit all hand sizes. This mechanism opens up for you to attach the remote control beyond holding it in your hand, but is not necessary to use. You can attach it wherever you want. Maybe you'd rather have it fastened around your wrist or to a fence? A picture of the rubber band with velcro is shown in Figure 28. Instead of using a rubber band, a normal piece of fabric would do the same job, but would be a bit less flexible. Other possibilities could be to use some sort of glue or sticky material to attach the controller to the hand, but it seems not very practical nor comfortable.

Figure 28: 3D-printed case with rubber band mechanism

In Figure 29, the case with the second button option is shown. This solution didn't give the best case printings. The printer had some troubles in the transition between case and button with the support material needed there. Additionally, with the button being too fragile, this solution was rejected. The final result of the prototype with a separate button is shown in Figure 30. How the fingers close over the button is also shown.

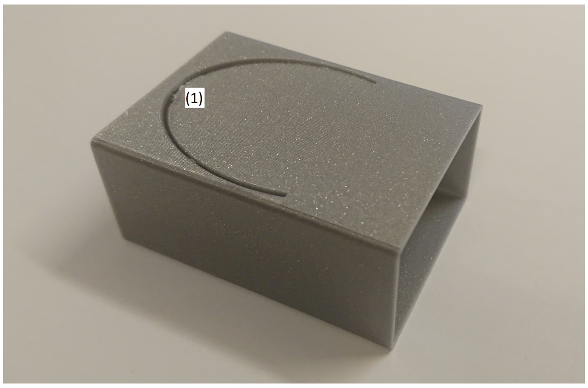


Figure 29: 3D-printed case with the second button option. (1) Transition

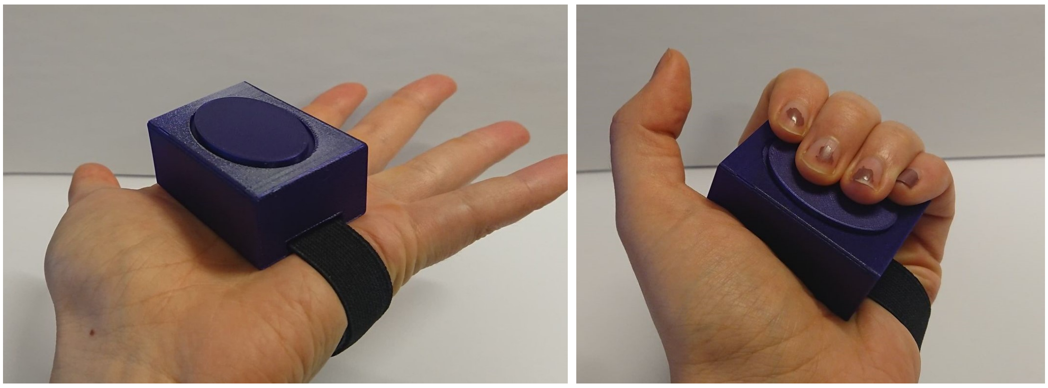


Figure 30: Printed prototype inside a hand

5.2.4 Alternative design

A final case should look more like the drawing in Figure 20. It should be more comfortable to hold on to, smaller and have a closing mechanism. These aspects cause high demands on 3D-design and printing. It might be easier and cheaper to buy an already made low-price case. It exists a lot of them with varying design that could fit some of these demands.

The case should have an on/off button additionally to the launching button. The on/off button would cut the power supply and save the battery lifetime. It would be useful to make it as a switch, where one position indicates the controller to be off and the other indicates it to be on. This should be a small switch located where it does not interfere or accidentally can be switched off when using the controller.

The controller can be used as a supplement to the application. If this was not the case, it would be beneficial to have more buttons on the controller to not lose functionality. The buttons could indicate where on the court the shuttlecock is fired or which sequence BADDY should follow. More buttons can be a good implementation regardless of the availability of the application.

Wall thickness might have to be redetermined. When making the prototype it was important that the printing time was not too long. For a final product, this is not an important factor and the robustness should not come at the expense of time savings. If the hardware is made smaller, the case can be smaller and the wall thickness would not affect the size a remarkable amount.

5.3 Hardware development

When developing hardware for the remote controller, three things were considered. First, the total volume of the components must be small enough to fits inside the 3D-printed case. Secondly, the hardware development had to follow the criteria and main goals for the BADDY project, which was open-source, affordable and sustainable. Therefore the components used to create both the electronics inside the controller and the case had to be cheap and easy to get a hold of. The circuit diagrams and explanations have to be available and open-source, so that building and maintaining this device could be done by anyone. Thirdly, technical requirements must be achieved. These were discussed in section 5.1.

To achieve the main goal of making the controller communicate with an Arduino board, it had to be divided into smaller sub-goals that had to be achieved first. The first sub-goal was to make a component communicate not-wirelessly with a computer. Then, make it communicate with a phone through Bluetooth. The third sub-goal was to make it communicate with another Bluetooth component. And finally, create a Bluetooth connection between a LED light and the controller in order to switch the LED light on and off when pressing a button. An illustration is shown in Figure 31.

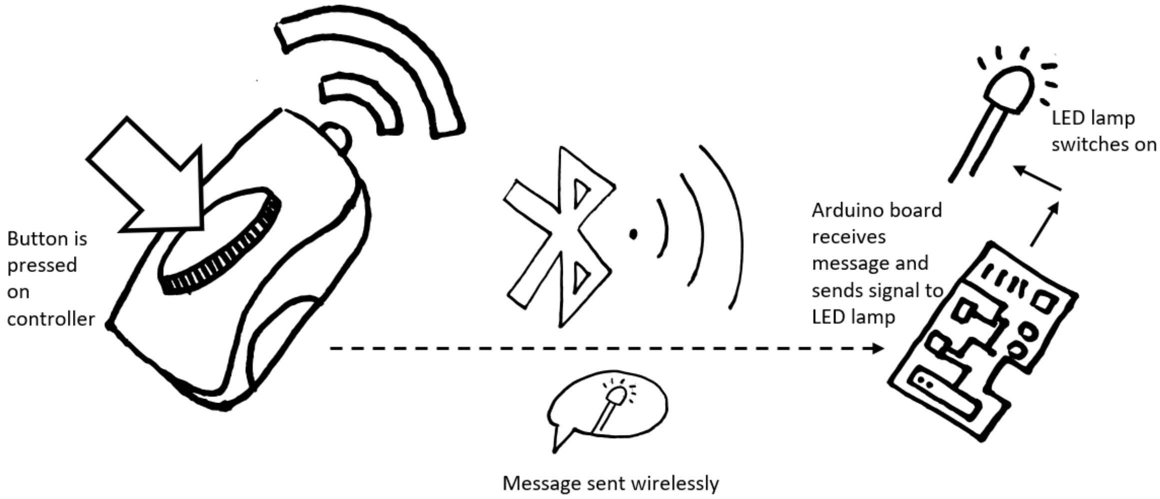


Figure 31: Illustration of signal transmission

5.3.1 Choosing components

As discussed in section 5.1, the battery doesn't have to be rechargeable. The battery selected, was chosen based on the relation between size and power. To fulfill the initial requirements of the BADDY project, the chosen components also had to be low in price and easy to get hold of. The power needed for the battery was decided by the choice of Bluetooth module. Most Bluetooth modules need a power supply of 2-5 V, so the battery should fit this criterion. A lithium button battery CR2032 gives a power supply of 3 V and has a diameter of 20 mm with 210 mAh [29]. This type of battery is easily accessible and the cost is less than 1 dollar when purchasing online. Based on the accessibility, usage and cost, this was chosen as the power supply for the controller. Figure 32 shows the battery chosen for this project and a battery holder in order to connect the battery to the other components [31].

Figure 32: CR2032 battery [45] and holder [31]

Two buttons will be needed for the controller. One power button in order to switch the remote controller on and off, and one launching button. The launching button will transmit a signal to a Bluetooth device when pushed, and trigger an action. The buttons showed in Figure 33, was used for this. These types of buttons are easily accessible and cost less than 1 dollar when purchased online [58, 32]. These particular buttons were chosen due to accessibility and cost, even though there exist many different types on the market. As an alternative to using a power button, an on/off-functionality could have been implemented with a software configuration. The launching button could work as both the launcher button and as the on/off button. By pressing the button shortly, it would launch a shuttle, and by holding it down for a certain amount of time, it would turn the controller on and off.

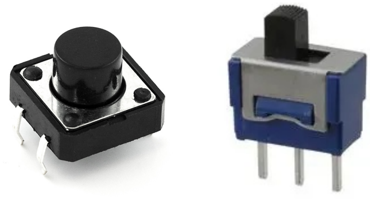


Figure 33: Push button switch to the left [58], slide switch to the right [32]

Wires together with a breadboard were used to attach all the electrical components together, as shown in Figure 34. The dimensions of the breadboard were 8.3 x 4.5 x 45.5 mm [30], which was the smallest breadboard available. It was desirable to make the controller as small as possible, something that required the use of small components. It was chosen to implement this breadboard in the solution since it was the smallest available and it was useful in the beginning phase where different circuits had to be created and tested. Even though this is a quite small breadboard, it was still the biggest component of the inside of the controller. Replacing the breadboard should be considered when improving the controller.

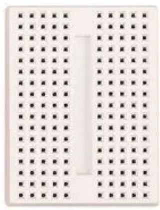


Figure 34: The breadboard [30]

Bluetooth module

An important challenge was to pick a suitable Bluetooth Low Energy (BLE) component for this project. It is possible to purchase many different components from different suppliers, and it was a long process of deciding which to choose. The three Bluetooth modules listed below were considered and tried out and Table 1 provides the relevant data for each of the modules.

1. nRF52840 Dongle from Nordic Semiconductor [53]
2. WRL-12580 - Bluetooth Module Mate Gold from SparkFun Electronics [60]
3. HC-08 Serial Bluetooth Module [48]

	nRF52840 Dongle	WRL-12580 Gold	HC-08 Module
Power supply	1.7 - 5.5 V	3.3 - 6 V	2 - 3.6 V
Bluetooth version	BLE 5.0	Bluetooth 2.0	BLE 4.0
Range	682 m	100 m	80 m
Dimensions	46 x 15 x 4 mm	13.4 x 25.8 x 2 mm	26.9 x 13 x 2.2 mm
Communication	UART/SPI	UART	UART

Table 1: Relevant module data [53, 60, 48]

The first Bluetooth module that was tested, was a development kit from Nordic Semiconductor [55]. The kit was of the type nRF52 - Bluetooth Low Energy [54] and is shown in Figure 35. This kit was chosen due to little previous experience regarding hardware development, and as it was a suggestion from the supervisor, Mr. Skavhaug. Nordic Semiconductor is located in Trondheim and specializes in wireless technology that is aimed towards the IoT [52]. The nRF52 DK purchased, "is a versatile single board development kit for Bluetooth Low Energy [...]" [56] and it is embedded with four buttons and four LED lights. This kit appeared like a good choice for this project, as it is compatible with Arduino, is easy to program, and had buttons that could be used in the remote controller.

Nordic Semiconductor offers multiple "Get started" guides and examples in order to get started with their products. By following the guides provided on their websites, nRF52 DK board was set up and connected to a computer correctly. Then, SEGGER Embedded Studio was downloaded as a development IDE. SEGGER Embedded Studio is an "all-in-one solution for managing, building, testing and deploying embedded applications" [51]. Some essential tools were also needed in order to develop Nordics devices. These were SDK, which is a software development kit, and nRF Command Line Tools, which is a set of command-line tools needed for the programming [57]. After downloading all the essential and necessary software, it was possible to start programming the nRF52 board. Multiple tutorials on how to configure the development kit and output pins on the board exist on the internet. A YouTube tutorial created for handling digital inputs from buttons was used as inspiration for the development of the nRF52 DK [27]. Based on this tutorial, it was possible to make the LED lights on the board light up when a button was pressed. The disadvantage of this development kit was the size of it, and it was difficult to implement a possible solution developed with this kit to source code of BADDY. The board provided in the kit had the dimensions 10.16 x 6.35 cm [56], which would make the remote controller almost the same size as a smartphone, and would therefore not be an improvement.

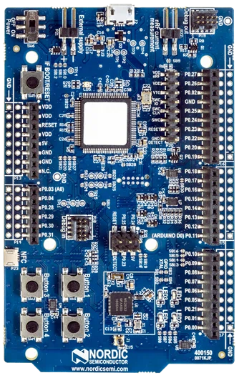


Figure 35: nRF52 Bluetooth Low Energy development kit from Nordic Semiconductor [54]

Based on this discovery, two nRF52840 Dongles [53] that works with the nRF52 kit was purchased. These are USB dongles and can be configured by connecting them to a computer using USB. Nordic Semiconductor has a software program called nRF Connect. This software allows for the initializing and set up of the dongles, as well as connecting them to a smartphone through Bluetooth. The nRF52840 Dongle that was used, can be seen in Figure 36.

Figure 36: nRF52840 Dongle from Nordic Semiconductor [53]

The physical assembly and connection between the dongle and the battery were performed using the breadboard seen in Figure 34. The battery was attached to the board using a battery holder, and wires connected the battery to the dongle as seen in Figure 37. Like the nRF52 DK board, the nRF52840 Dongle is also equipped with a built-in button, a LED light and a lot of GPIO pins. From Table 1, the dimensions of this dongle are given as 46 x 15 x 4 mm, which is significantly smaller than the development kit. This could lead to a potentially smaller remote controller if the components were placed in a clever way in relation to each other, and in a way that allowed the button on the dongle to be pressed when pressing an actual button on the remote controller.

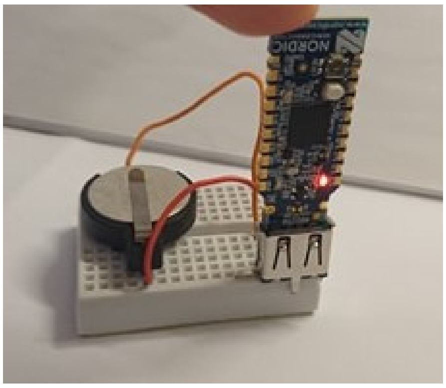


Figure 37: The dongle connected to the battery with the bread board. This was a test to see if the battery could power the dongle

When connecting the dongle to the Arduino board, a problem occurred. The dongle is not equipped with the digital pins, RX and TX, used for receiving and transmitting data, that the Arduino board has. After some research, it was discovered that this could be avoided by using some of the 15 GPIOs (general-purpose input/output) along with the castellated edges of the dongle.

In addition to this problem, it was hard to connect the two dongles together with the nRF Connect software. After some feedback from an employee at Nordic Semiconductor, it was established that the company doesn't have its own Arduino library. In order to use the nRF dongle with another Arduino board, it would be necessary to use third-party implementations. This made the process more complicated, and the choice of using an nRF dongle for this project was reconsidered.

Due to this discovery, other Bluetooth modules were examined and several guides on how to connect Bluetooth devices together were found. One of the tutorials studied, connected two HC-05 Bluetooth Modules, as slave and master device, and made them communicate with each other wirelessly [41]. A newer version of this module, HC-08, is an upgraded version of HC-05 and comes with Bluetooth Low Energy [48], something that seemed promising for this project. These modules were harder to obtain and arrived a week before the project deadline.

In the meantime, a module from SparkFun, WRL-12580 - Bluetooth Module Mate Gold [60], was easily accessible and could be used immediately. As seen in Figure 38, this module is not embedded with a button like the nRF dongle. It is also not compatible with BLE technology, since it uses Bluetooth v2.0. Modules using Bluetooth requires more power than those using BLE, hence, this module was not an appropriate fit for the requirements of the remote controller. Nevertheless, it was available and could be tested while a better alternative was shipped.

Figure 38: SparkFun Bluetooth Mate Gold WRL-12580 ROHS [60]

The module was connected to an Arduino board and it was configured using UART commands. Hen, it was connected to a phone with Bluetooth. No more was done with this module since the HC-08 modules arrived.

The HC-08 modules arrived during the last week of the project. From Figure 39 it can be seen that the HC-08 module doesn't come with an on-board button either. This meant that a separate button had to be connected to one GPIO pin and placed inside the controller. This module fits the requirements, both with regards to using BLE technology, and it has the pins needed in order to configure it correctly. The wiring is shown in section 5.3.2.

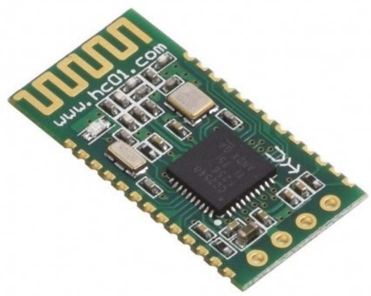


Figure 39: HC-08 Serial Bluetooth Module CC2540 (BLE 4.0) [48]

A final list of components needed to build and use the remote controller is shown in table 2. Additionally, you need a 10k Ohm resistor, some circuit wires and the printed case with the rubber band. The total cost will be around 210 NOK, which is about the same as a tube with shuttlecocks. Making the controller will increase the total cost of BADDY by 4-5%.

Component name	Where to find it	Approximately price	Quantity
HC-08 Serial Bluetooth Module	Smart prototyping [48]	50 NOK	2
Breadboard, White, 8.3 mm	Farnell [30]	16 NOK	1
Battery Holder	Farnell [31]	48 NOK	1
CR2032 button battery	Farnell [29]	7 NOK	1
Push button	SparkFun [58]	5 NOK	1
Slide switch button	Farnell [32]	34 NOK	1

Table 2: Component data

5.3.2 Connecting hardware in controller

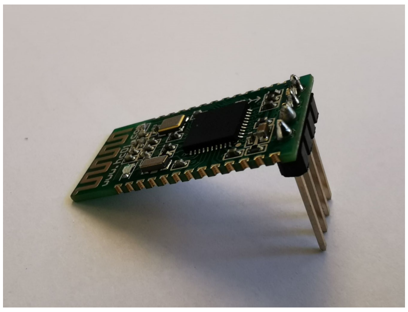


Figure 40: Connector pins soldered to the module

The modules had to be soldered with connector pins as shown in Figure 40, before they could be connected to a breadboard and circuits could be created. Figure 41 illustrates the circuit diagram for the button and slave module, with the respective wired circuit. While the circuit diagram for the LED and master module, as well as the respective wired circuit, can be seen in Figure 42. Since it is a one-way communication, from slave to master, there is no need for RX and TX connection on both circuits.

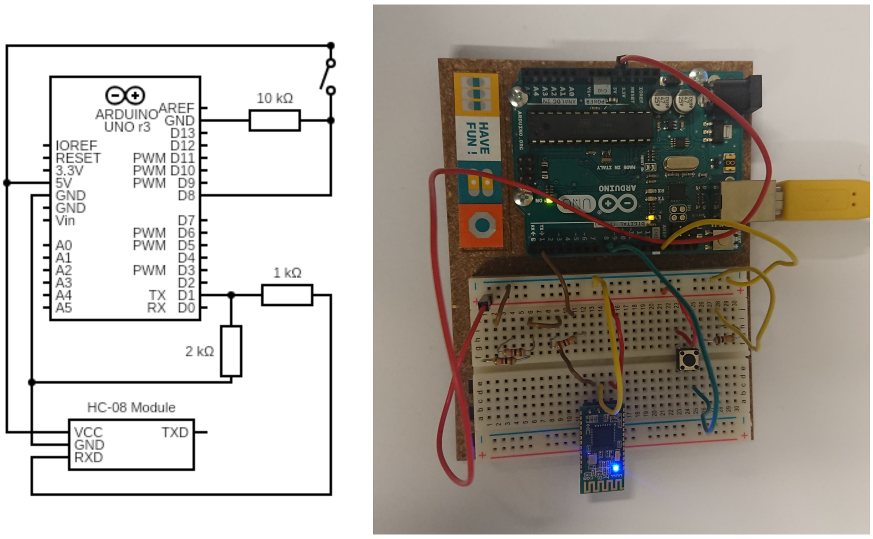


Figure 41: Circuit diagram for the slave module to the left, and wired circuit to the right

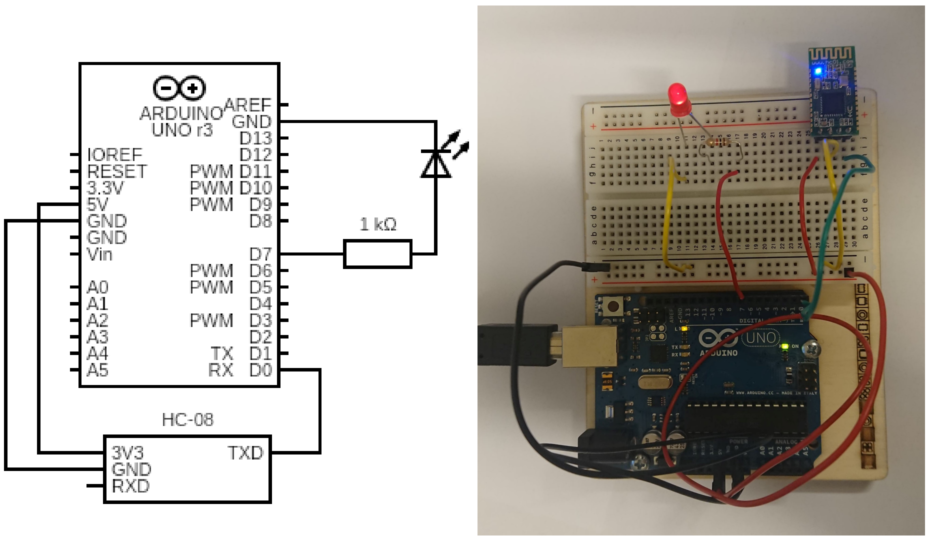


Figure 42: Circuit diagram for the master module to the left, and wired circuit to the right

For the slave module circuit, some resistors have to be placed between the Arduino TX port and the RX port on the module. The Arduino TX pin has a 5 V output, but the logic voltage level of the RX pin for the module is 3.3 V. So, the connecting line between the Arduino and the module needs to be connected through a voltage divider in order to prevent the module from burning.

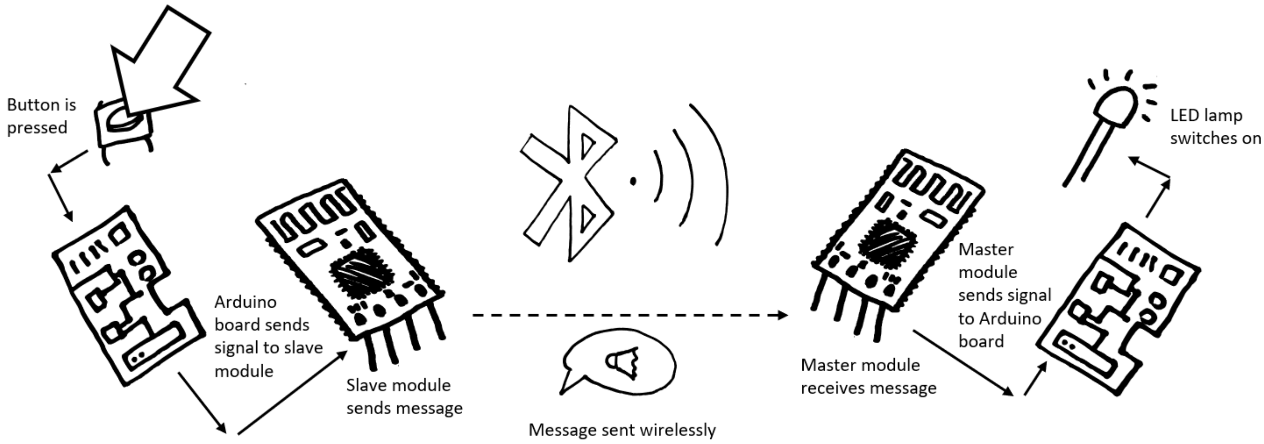


Figure 43: Illustration of signal transmission with two Arduino boards

When the launching button is pressed, a signal goes from the circuit to the first Arduino board and to the slave module. The slave module sends a message to the master module wirelessly with Bluetooth. When the master module receives the message, it signals to the second Arduino board to lighten the LED. This is illustrated in Figure 43. Each time the button is pressed, the state of the LED light is changed to the opposite of what it was. If the light is on (HIGH) and the button is pressed, the LED light turns off (LOW).

5.3.3 Further development of the controller

The four sub-goals of this report has been achieved and is demonstrating the concepts of the remote controller. Still, a few things have to be modified for the controller to work as intended with BADDY. The intended setup is shown in Figure 44.

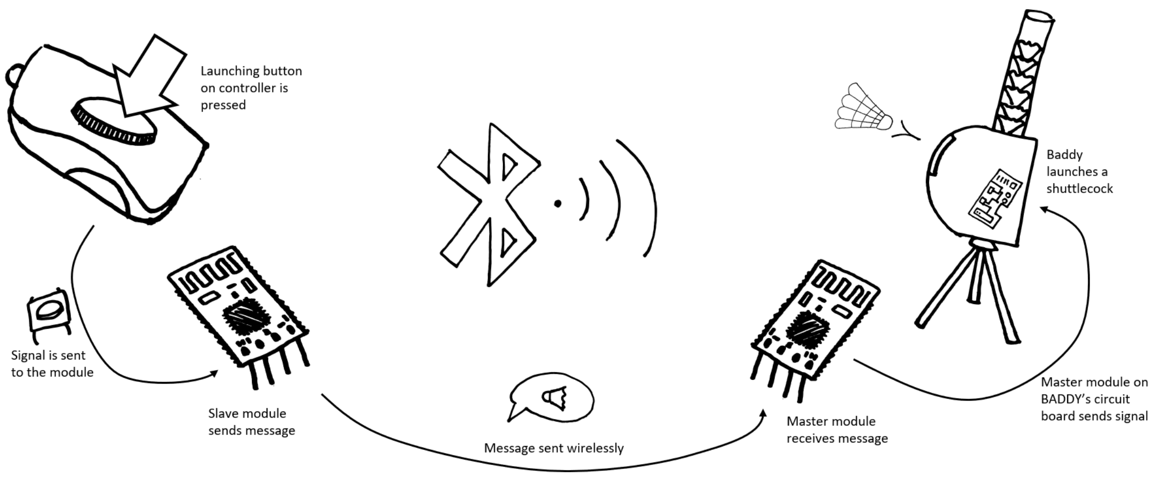


Figure 44: Illustration of the signal transmission. The modules are part of the controller and BADDY

The slave module with the launching button has to be modified to fit inside the controller case. This can be done by getting rid of the Arduino Uno board or switching to one that is smaller. An alternative would be to use an Arduino Nano board, shown in Figure 45. It has dimensions of 18 x 45 mm [62], and could easily fit inside the case. If the Arduino board is removed completely, the module has to be programmed in C code to perform the same logic.

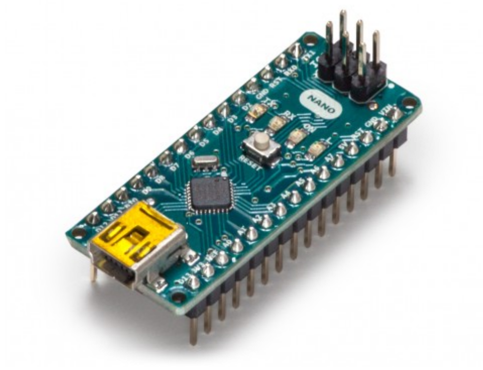


Figure 45: Arduino Nano with dimensions of 18 x 45 mm [62]

One GPIO pin on the module will be used instead of the RX pin (which is mainly used for UART commands) to receive the signal from the launching button. P2.0 is a pin defined as ”input, weak pull up” [47] and is appropriate with a switch button. A pin with ”Weak pull up” has a built-in pull-up resistor. With an open switch, the pin would be read as high, and a closed switch will be read as low.

A power button for switching the controller on and off should also be placed and connected to the circuit of the controller. This was not taken into account when designing the case, so the case doesn't have any functionality for this.

A circuit diagram for the controller is shown in Figure 46, along with the wired circuit without any Arduino board. A slide switch was not available when making the circuit, so it is missing from the wired model.

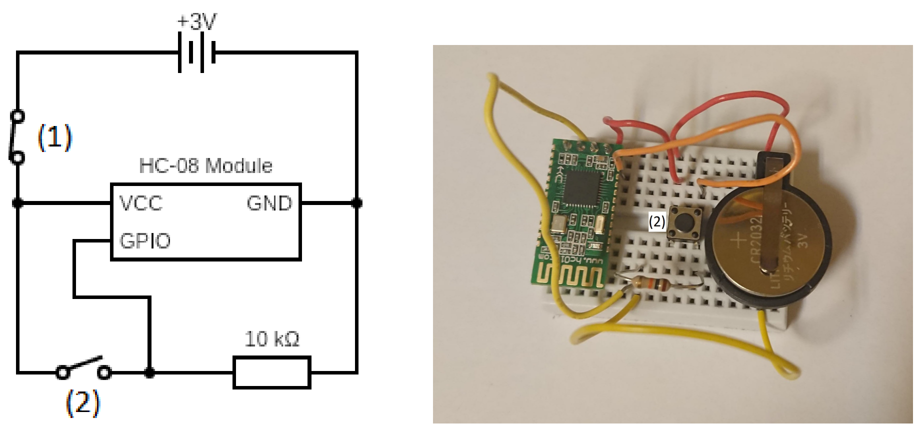


Figure 46: Circuit diagram of the controller to the left, where (1) is the power switch button and (2) is the launcher button. Wired circuit to the right

Figure 47 shows the wired circuit inside the case. The button functionality doesn't work with the used button component. This can be solved either by redesigning the case with respect to the button, or some other button component has to be used for the wired circuit. The wires could also be exchanged for shorter ones to make the whole circuit more compact and clean (long wires were used for clarity and to keep to the same color coding as in Figure 41 and 42).

Figure 47: The hardware of the controller placed inside the case. So far, no closing mechanism has been created

5.4 Software development

Since the modules are used with Arduino boards, Arduino code was used to program the functionality. The program *Arduino IDE* was downloaded [4] and worked as the programming platform. Arduino language is based on C and C++ language and has libraries and tools for easy setup of microcontrollers [65]. Arduino code files end with .ino and have to be saved in a folder with the same name as the code file. The folders and files used are shown in Figure 48.

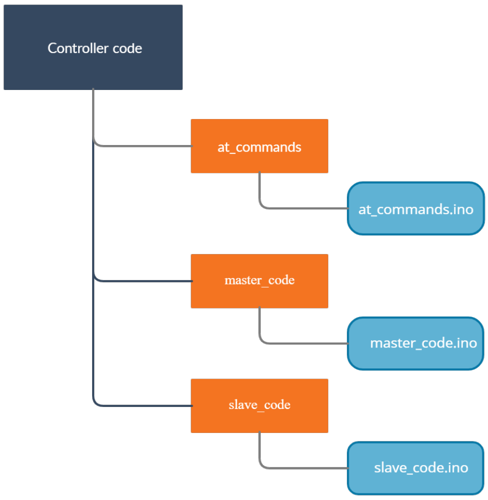


Figure 48: Folder structure for the Arduino files

To use the HC-08 modules with an Arduino board, the library SoftwareSerial had to be used [5]. This library is embedded with built-in functions for using the pins at the Arduino board, in order to communicate with a UART (Universal asynchronous receiver-transmitter) device on the modules [66].

The HC-08 Modules had to be configured with UART commands before they could be used. These modules come with a default slave mode. To be able to get an automatic connection between a pair of modules, one has to be changed to a master device. A module with slave mode uses less power than one without. Since it would be an advantage to create the controller in a way that it uses as little power as possible, the slave module was implemented here. The extra power drain by the master module would be irrelevant since it is connected to BADDY that has a wired power source. Some short UART commands, as shown in Figure 49, had to be implemented on the slave module, mainly to obtain a unique Bluetooth address. This address was then used to make an automatic and secure connection line between the master and the slave modules. The code for this can be seen in Listing 1 in Appendix A. It was based on a sketch made by Jim Lindblom [18] and has been modified to fit the needs of this project. By using the "AT+BIND=<address>"-command, the master module will automatically try to connect to the device with this address when turned on. The moment the slave module turns on, it will be connected to the master.

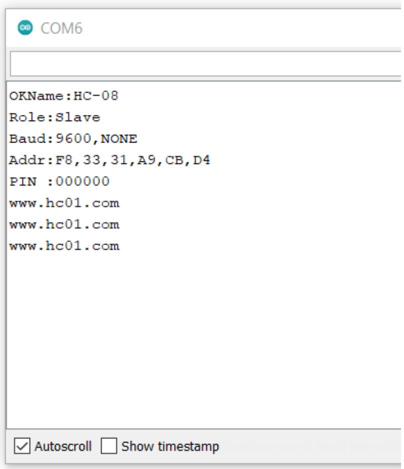


Figure 49: Serial monitor with information received from the slave HC-08 module

The remote controller must be equipped with a button. When this button is pressed, a new shuttlecock should be launched from BADDY. In order to simulate this, a LED light was used as the shuttlecock. To tell if the button and LED worked correctly, the LED was first connected to the button on the same Arduino board. When the button was pressed, the LED light would turn on. The code is given in Listing 2, Appendix B. Arduino has a lot of example code on the web site and in the Arduino IDE. The code for this test was a composition of the blink-example [7] and the button-example [6].

The LED and button could not be on the same Arduino board when realizing the project. It was therefore necessary to connect one of the Bluetooth modules to each of them, where the signals had to go through the modules. The slave module was connected to the button and the master module was connected to the LED. The code used for both the slave and master module is shown respectively in Listing 3 and 4 in Appendix C. This code is a combination of the previous code in Listing 2 and a code sketch made by Dejan Nedelkovski [41].

When adding the controller to BADDY, the master code has to be implemented into BADDY's programming code. Instead of making the LED lamp light up, BADDY should launch a new shuttlecock. It can also be necessary to modify the slave code to be able to work without an Arduino board. If this improvement of BADDY is accepted by the founders, the final code base will be uploaded to BADDY LAB's Github page [11] and be accessible for everyone.

6 Discussion, future work and conclusion

Most of the discussion of improvements and choices made can be read earlier in the report. This is an overall discussion about how the remote controller fits within the three main goals of the BADDY project.

- Open-source
- Affordable
- Sustainable

All the work performed on the remote controller has been presented in this report, including 3D-drawings of the case and software code. Anyone who wants to build this controller can do so. It is up to the founders of the BADDY project if they want to include the remote controller into their project and make the controller an addition to BADDY.

Some of the tools used, like 3D-printers and sewing machines, can be quite expensive to purchase, but it might not be necessary to buy them. The BADDY community already encourages BADDY builders to contact local Hackerspaces or similar places for loan of equipment and to receive help. It is also possible to sew the rubber band by hand, instead of using a sewing machine.

Using a breadboard in the hardware of the controller is not the most compact way of connecting components, but it makes it easy to replace components if they are broken. The two hour 3D-printing time makes it fast to reproduce the case if it gets broken. The current design of the case and hardware makes it easy to change the battery, but it could be improved by adding a closing mechanism designed to easily access the battery. The final case presented in this report, with one side open, makes the components inside vulnerable to dust as well as there is a risk for the components to fall out. Therefore, the hole must be closed when playing. It can be done easily with tape.

3D-printing plastic might not be the best building material for the case. It can be fragile and easily torn apart if the user lacks experience in 3D-printing and doesn't have adequate tools. While playing badminton the player might be careless and eager, and by accident squeeze the controller so hard that it breaks. Laser cutting parts are already implemented in building BADDY and this method could also be used to create a controller, but might not be a better alternative than 3D-printing. It would be hard to create a robust, rounded case that is comfortable to hold with straight parts. A better alternative might be to create a casting form and cast the case in a stronger material, like fiberglass. This will create a very robust case, but it still needs some sort of opening and closing mechanism to get the hardware inside. The opening mechanism can be created in several ways and the use of screws is one option that has to be considered. It can be hard to get a robust opening mechanism without screws.

Different alternatives to the HC-08 Bluetooth module have been tested and documented in this report. All of them could probably work in the end, and which one to use is hard to tell. More testing has to be performed in order to determine this. The final choice should be based on multiple factors, of which the price and battery consumption should be one of the deciding factors.

It is not necessary to have the remote controller in order to use BADDY. The controller will be an extra feature that can be chosen to purchase or create if needed or for fun. It will therefore not be a part of the original low-cost version of BADDY. Since the controller works as an extra feature to BADDY it will add some extra cost. This extra cost will be small compared to the more expensive robots on the market, but the total cost of BADDY will increase.

If you have built the remote controller yourself, it will be easy to repair and maintain it. If something breaks, you have built it before and can do it again, which is sustainable.

The actual implementation of the improvement discussed in this report has been left for future work due to lack of time and necessary equipment. In the future, it is planned to improve the software behind the HC-08 Bluetooth modules and make it possible for BADDY to communicate with the controller. In order to do so, BADDY first has to be mounted and tested. The provided code will have to be modified in order to make the slave module work without an Arduino board or with an Arduino Nano board. Another part of the future work will be to develop the case for the controller and make it fit all the requirements this study has provided. It should be more comfortable to hold, smaller and more rounded, and get an opening mechanism. In order to get the case smaller, other options than using a breadboard for connecting components are possible. Using soldering or making a Printed Circuit Board (PCB) would decrease the total volume of the components, which would lead to a reduction in the size of the case. These are more permanent solutions than using a breadboard and increases the requirements for the building process and tools needed. A mistake would not be as easy to undo.

A step by step guide or YouTube video on both how to build the controller and how to use it should be created. This should be done when the controller has been tested with a BADDY and the design process is closer to a finished product.

Conclusion

This study has addressed how BADDY, a badminton shuttlecock launcher robot, can be improved by implementing new additions to an existing robot. Different improvements were considered and discussed within three focus aspects; security, performance and ease of use. The chosen improvement was to make a remote controller, and the design process was separated into three parts; case design, hardware development and software development. The remote controller was made using a 3D-printer, Bluetooth Low Energy (BLE) technology, and Arduino coding. The final result still has a lot of potential for improvement and it still has to be implemented with a BADDY. This will include adding the developed Arduino code into BADDY's codebase and attach the Bluetooth module to BADDY's hardware. The total volume of the hardware components has to be decreased and the controller has to be modified to work without an Arduino board or with an Arduino Nano. The case button has to be adapted to the hardware button and an on/off button should be added. The case should be smaller, more rounded and a closing mechanism should be implemented.

When a final remote controller, ready for mass production, is presented, the controller will add a new layer of ease of use to the project. It is no revolutionary, life important improvement, but an extension that will give people the opportunity to use BADDY in an easier way. The controller will give players the opportunity to practice their technique better than before, and hopefully, save some phones from falling to the ground. This implementation will also add new aspects of robot building to the BADDY project.

The candidates have all intentions to continue the work with this project in their master's degree in the spring of 2021. They will improve the remote controller further and follow the advice stated in this section. If a remote controller with working functionality is successfully provided, it will be submitted to the BADDY project[2].

[2] The work will be published for future development regardless of the final product state

Bibliography

[1] Mathieu Abet. WHAT IS BLUETOOTH LOW ENERGY? URL https://elainnovation.com/what-is-ble.html. Accessed: 07.12.2020.

[2] Alibaba.com. Remote control portable badminton robot shuttlecock robot. URL https://www.alibaba.com/product-detail/Remote-control-portable-badminton-robot-shuttlecock_60791626592.html?spm=a2700.7724857.normalList.72.22952371FzR801. Accessed: 16.11.2020.

[3] Aniwaa. ORIGINAL PRUSA I3 MK3S. URL https://www.aniwaa.com/product/3d-printers/prusa-research-original-prusa-i3-mk3s/. Accessed: 28.11.2020.

[4] Arduino. Arduino IDE download, . URL https://www.arduino.cc/en/software. Accessed: 12.10.2020.

[5] Arduino. SoftwareSerial Library, . URL https://learn.sparkfun.com/tutorials/using-the-bluesmirf/example-code-using-command-mode. Accessed: 04.12.2020.

[6] Arduino. Button, . URL https://www.arduino.cc/en/Tutorial/BuiltInExamples/Button. Accessed: 14.10.2020.

[7] Arduino. Blink, . URL https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink. Accessed: 14.10.2020.

[8] BADDY. Baddy website, . URL https://www.baddy.space. Accessed: 15.11.2020.

[9] BADDY. Disclaimer, . URL https://www.baddy.space/make-robots-play-sports. Accessed: 28.10.2020.

[10] BADDY. Frequently Asked Questions, . URL https://www.baddy.space/baddy-badminton-feeding-machine-diy. Accessed: 28.10.2020.

[11] BADDY. BADDY github page, . URL https://github.com/baddy-lab/baddy-makers-edition. Accessed: 07.12.2020.

[12] BADDY. BADDY V2 - Maker kit, . URL https://www.baddy.space/product-page/baddy-v2-maker-kit. Accessed: 17.11.2020.

[13] BADDY. Specifications, . URL https://www.baddy.space/baddy-badminton-diy-robot-specifica. Accessed: 28.10.2020.

[14] Baddy. Home [YouTube Channel], . URL https://www.youtube.com/channel/UCYEWljJYV7ifch94jTNyJbA/featured. Accessed: 17.11.2020.

[15] Baddy. BADDY KICKSTARTER campaign incoming [Video]. YouTube, . URL https://www.youtube.com/watch?v=YVO95fcGQrA. Accessed: 20.11.2020.

[16] Master Badminton. Badminton Body Balance. URL https://www.masterbadminton.com/badminton-body-balance.html. Accessed: 29.09.2020.

[17] Prusa Knowledge Base. Print Quality Troubleshooting. URL https://help.prusa3d.com/en/tag/mk3s/print-quality-troubleshooting_225. Accessed: 28.11.2020.

[18] Jim Blom. Using the BlueSMiRF. URL https://learn.sparkfun.com/tutorials/using-the-bluesmirf/example-code-using-command-mode. Accessed: 04.12.2020.

[19] A. M. Cabrero. BADMINTON TECHNIQUES. *KSU Kent State University*, pages 1–5, 2017.

[20] Pedro Henrique Gonzalez de Cademartori, Wido Herwig Schreiner, and Washington Luiz Esteves Magalhães. Facile one-step fabrication of highly hydrophobic medium density fiberboard (mdf) surfaces via spray coating. *Progress in organic coatings*, 125:153–159, 2018. ISSN 0300-9440. doi: 10.1016/j.porgcoat.2018.08.032.

[21] Jeffrey B. Carruthers and Joseph M. Kahn. Angle Diversity for Nondirected Wireless Infrared Communication. *IEEE TRANSACTIONS ON COMMUNICATIONS*, 48(6):965, 2000.

[22] Dibya Chakravorty. 3D Printing Support Structures – The Ultimate Guide. URL `https://all3dp.com/1/3d-printing-support-structures/`. Accessed: 17.11.2020.

[23] Olympic Channel. Badminton: How to play, rules, and all you need to know. URL `https://www.olympicchannel.com/en/stories/features/detail/badminton-guide-how-to-play-rules-olympic-history/`. Accessed: 15.11.2020.

[24] Chris Eason. Shuttlecock. URL `https://www.flickr.com/photos/mister-e/3294913646/in/photolist-62ai2Y-EZB7S-EZC3B-EZCVe-DwfP8b-EZDon-EZAB5-EZAA4-EZCUq-9YVWkr-pXd7RR-5FuYzV-uK7tG-pWZhbQ-jhYkUJ-2j5DPY1-4rabgQ-9PokFv-6aNPCP-6a4uES-7RjhBZ-5hxwMU-adCdSw-2i9sA3c-7moYP9-3b6B3R-CEWFjj-7v9kzX-4jQwu-4LLBEV-ewKB3E-nknj4M-9PG6eu-8hziEe-TmXnq8-wrysRr-5AxAzj-4hrTxP-8EGrtr-5xGCCf-RrH6ZM-arSBRz-DAwccM-rWeoY-jGFVK-8kwsis-22aFqmq-7i1eWS-9i3Y3-7AQSfq`. Accessed: 15.11.2020.

[25] Ellisys. Ellisys Bluetooth Video 1: Intro to Bluetooth Low Energy [Video]. YouTube. URL `https://www.youtube.com/watch?v=eZGixQzBo7Y`. Accessed: 07.12.2020.

[26] ElProCus. What is WiFi Technology How Does It Work? URL `https://www.elprocus.com/how-does-wifi-technology-work/`. Accessed: 07.12.2020.

[27] Sumair's Embedded Engineering. nRF5 SDK - Tutorial for Beginners Pt 6 - Reading Digital Input from Buttons [Video]. YouTube. URL `https://www.youtube.com/watch?v=zFANmKZFMrc`. Accessed: 05.10.2020.

[28] K. Sasaki et al. 3D Sensing Technology for Rea-Time Quantification of Athletes' Movements. *FUJITSU Sci. Tech. J.*, 54(4), 2018. Accessed: 10.11.2020.

[29] Farnell. Multicomp CR2032, . URL `https://no.farnell.com/multicomp/cr2032/cell-lithium-cr2032-210mah-3v/dp/2065171?ost=2065171`. Accessed: 30.11.2020.

[30] Farnell. Breadboard, White, 8.3 mm, 34.5 mm, 45.5 mm, . URL `https://no.farnell.com/multicomp/mc001799/breadboard-34-5mm-x-45-5mm-white/dp/2770327?scope=partnumberlookahead&ost=2770327&searchref=searchlookahead&exaMfpn=true`. Accessed: 30.11.2020.

[31] Farnell. Battery Holder, Coin Cell - 20mm x 1, . URL `https://no.farnell.com/keystone/103/battery-holder-20mm-coin-cell/dp/676469?scope=partnumberlookahead&ost=676469&searchref=searchlookahead&exaMfpn=true`. Accessed: 30.11.2020.

[32] Farnell. Slide Switch, . URL `https://no.farnell.com/nidec-copal/8ss1022-z/slide-switch-spdt-6a-125vac-th/dp/2854816?st=slide%20switch`. Accessed: 12.10.2020.

[33] FutureBridge. 3D Sensing - New Ways of Sensing the Environment. `https://www.futurebridge.com/blog/3d-sensing-new-ways-of-sensing-the-environment/`. Accessed: 10.11.2020.

[34] Arbi Abdul Jabbaar. ultrasonic sensor hc-sr04 with arduino tutorial.

[35] Hugo Janton. Improvement of baddy the shuttlecock launcher. 2019.

[36] Steven John. 'What is Bluetooth?': A beginner's guide to the wireless technology. URL https://www.businessinsider.com/what-is-bluetooth?r=US&IR=T. Accessed: 06.12.2020.

[37] Insook Kim. Teaching Badminton through Play Practice in Physical Education. *Journal of physical education, recreation dance*, 88(8):7–14, 2017. ISSN 0730-3084. doi: 10.1080/07303084.2017.1356768.

[38] Bryan Maddock. Badminton Court. URL https://www.dimensions.com/element/badminton-court. Accessed: 19.11.2020.

[39] MAKERDEMY. What is BLE? (2020) — Bluetooth Low Energy — Learn Technology in 5 Minutes [Video]. YouTube. URL https://www.youtube.com/watch?v=NSkIHdV6NoY. Accessed: 07.12.2020.

[40] Daniel Maydana. Handheld enclosure for miscellaneous projects. URL https://grabcad.com/library/handheld-enclosure-for-miscellaneous-projects-1. Accessed: 08.12.2020.

[41] How To Mechatronics. How To Configure and Pair Two HC-05 Bluetooth Modules as Master and Slave. URL https://howtomechatronics.com/tutorials/arduino/how-to-configure-pair-two-hc-05-bluetooth-module-master-slave-commands/. Accessed: 01.12.2020.

[42] Hughes Mike and Franks Ian. *Essentials of Performance Analysis in Sport: second edition.* London: Taylor and Francis, London, 2015. ISBN 9781138022997 1138022993 1138022985 9781138022980. doi: 10.4324/9781315776743. Accessed: 10.11.2020.

[43] Lauree Padgett. Keeping promises and all that CRAAP. *Information today*, 34(4):19, 2017. ISSN 8755-6286. Accessed: 14.11.2020.

[44] Cork Sports Partnership. URL https://www.corksports.ie/index.cfm/page/schools-badminton?fbclid=IwAR0movOQBKAFZEk6TtNBY1tSkztF4_OWud02-7dw1VffXewgk-ZaxDBngSw. Accessed: 15.11.2020.

[45] PCoard. CR2032 3V Lithium Button Cell Battery. URL https://www.pcboard.ca/cr2032-lithium-battery.html. Accessed: 30.11.2020.

[46] Michael Phomsoupha and Guillaume Laffaye. The Science of Badminton: Game Characteristics, Anthropometry, Physiology, Visual Fitness and Biomechanics. *Sports Medicine*, 45(4):473–495, 2015. ISSN 1179-2035. doi: 10.1007/s40279-014-0287-2. URL https://doi.org/10.1007/s40279-014-0287-2. Accessed: 15.11.2020.

[47] professorcad. HC-08 BLUETOOTH UART COMMUNICATION MODULE V3.1 USER MANUAL . URL https://www.professorcad.co.uk/Bluetooth/HC-08A%20version%20english%20datasheet.pdf. Accessed: 12.10.2020.

[48] Smart Prototyping. HC-08 SERIAL BLUETOOTH MODULE. URL https://www.smart-prototyping.com/HC-08-4-0-BLE-Serial-Bluetooth-Module-CC2540. Accessed: 30.11.2020.

[49] Prusamenty. Prusament PLA. URL https://prusament.com/materials/pla/. Accessed: 28.11.2020.

[50] Brian Ray. Bluetooth Vs. Bluetooth Low Energy: What's The Difference? URL https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy. Accessed: 07.12.2020.

[51] SEGGER. Embedded Studio - The All-In-One Embedded Development Solution. URL https://www.segger.com/products/development-tools/embedded-studio/. Accessed: 08.12.2020.

[52] Nordic Semiconductor, . URL https://www.nordicsemi.com/About-us. Accessed: 08.12.2020.

[53] Nordic Semiconductor. nRF52840 Dongle, . URL https://www.nordicsemi.com/Software-and-tools/Development-Kits/nRF52840-Dongle. Accessed: 28.11.2020.

[54] Nordic Semiconductor. nRF52-DK Development Kit, . URL https://www.nordicsemi.com/Software-and-Tools/Development-Kits/nRF52-DK. Accessed: 28.11.2020.

[55] Nordic Semiconductor, . URL https://www.nordicsemi.com/. Accessed: 28.11.2020.

[56] Nordic Semiconductor. nRF52 DK, . URL https://www.nordicsemi.com/Software-and-tools/Development-Kits/nRF52-DK. Accessed: 08.12.2020.

[57] Nordic Semiconductor. Nordic tools and downloads, . URL https://infocenter.nordicsemi.com/index.jsp?topic=%2Fug_getting_started%2FUG%2Fgs%2Fdevelop_sw.html&cp=1_0_2. Accessed: 08.12.2020.

[58] SparkFun. Momentary Pushbutton Switch - 12mm Square. URL https://www.sparkfun.com/products/9190. Accessed: 06.12.2020.

[59] Sparkfun. Ultrasonic Distance Sensor - HC-SR04, . URL https://www.sparkfun.com/products/15569. Accessed: 19.11.2020.

[60] Sparkfun. SparkFun Bluetooth Mate Gold, . URL https://www.sparkfun.com/products/12580. Accessed: 30.11.2020.

[61] Antony Stanley. Cai Yun and Fu Haifeng vs. Mathias Boe and Carsten Mogensen In The Men's Doubles Badminton Final. URL https://www.flickr.com/photos/antonystanley/8172623247/in/photolist-dsbQan-cPCg3f-cPChvC-cPC9RQ-dsbQXH-cPC6T5-cJYrML-cNKKz1-cPCcfd-cJYkUj-cPCb5s-cPCdkm-dsbWZL-cK1DPN-cPC4CL-cJYq4b-cPCesN-cK1oVE-dc23H7-cK1wJJ-cJYpbw-cK1B2E-cJYrjo-cJYsh7-cJYkqm-cK1Ae5-cJYxQu-cK1mTb-cJZpM3-cK1tpJ-cJZ7Z1-cJYpKE-cK1BoJ-cK1ufq-cJYUB9-cJZyXw-d9YQpu-cJYYCN-cJYScG-cJYDe1-cK1d8S-cJYLT1-cJZdTq-cJYvWW-cJZ4Ym-cJZ3fE-cJZtz1-cJYzQs-cJZ2Gb-cJZzDs. Accessed: 15.11.2020.

[62] Arduino store. ARDUINO NANO. URL https://store.arduino.cc/arduino-nano. Accessed: 12.10.2020.

[63] BADDY team. Baddy. URL https://play.google.com/store/apps/details?id=com.baddy&hl=no. Accessed: 19.11.2020.

[64] Thebadmintonguide. Badminton Shuttle – All You Need to Know About the Badminton Projectile. URL https://help.prusa3d.com/en/tag/mk3s/print-quality-troubleshooting_225. Accessed: 28.11.2020.

[65] Wikipedia. Arduino IDE, . URL https://en.wikipedia.org/wiki/Arduino_IDE. Accessed: 12.10.2020.

[66] Wikipedia. Universal asynchronous receiver-transmitter, . URL `https://en.wikipedia.org/wiki/Universal_asynchronous_receiver-transmitter`. Accessed: 06.12.2020.

[67] B. Wan P. Visentin Q. Jiang M. Dyck H. Li G. Shan Z. Zhang, S. Li. The Influence of X-Factor (Trunk Rotation) and Experience on the Quality of the Badminton Forehand Smash. *Journal of Human Kinetics*, 53:10, 2016.

Appendix

A Arduino code for UART commands

Listing 1: Code using UART commands to set up HC08 modules. The code is based on a sketch made by *Jim Lindblom* [18]

```
#include <SoftwareSerial.h>  //include library for UART communication

int bluetoothTx = 2;  // TX pin of HC-08 Bluetooth Module, Arduino D2
int bluetoothRx = 3;  // RX pin of HC-08 Bluetooth Module, Arduino D3

//create an instance of a SoftwareSerial object
SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);

void setup()
{
  Serial.begin(9600);  // Begin the serial monitor at 9600bps
  // Begin The Bluetooth Module, default is 9600bps
  bluetooth.begin(9600);
  bluetooth.print("AT");  // Test command
  delay(100);  // Short delay, wait for the Module to send back OK
  bluetooth.print("AT+RX"); // Check the basic parameters
  delay(100);
  bluetooth.print("AT+NAME=HC-MrMaster"); // Change name on the module
  delay(100);
  bluetooth.print("AT+ROLE=M"); // Change role from slave to master
  delay(100);
  // Set connect ability so it can be connected, default is 0
  bluetooth.print("AT+CONT=0");
  delay(100);
  // Binding to slave for secure and automatic connection
  bluetooth.print("AT+BIND=F8,33,31,A9,CB,D4");
  delay(100);
  bluetooth.print("AT+RX"); // Check new basic parameters
}

void loop()
{
  if(bluetooth.available())  // If the bluetooth sent any characters
  {
    // Send any characters the bluetooth prints to the serial monitor
    Serial.print((char)bluetooth.read());
  }
  // If anything is typed into the Serial monitor
  if(Serial.available())
  {
```

```
    // Send any characters the Serial monitor prints to the bluetooth
    bluetooth.print((char)Serial.read());
  }
}
```

B Arduino code for button and LED

Listing 2: Arduino code making a LED lamp turn on when a button is pressed

```
const int buttonPin = 8; // the number of the pushbutton pin
const int ledPin =  7;  // the number of the LED pin
int buttonState = 0; // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

C Arduino code for slave and master module

Listing 3: Arduino code for slave module. The code is based on a sketch made by *Dejan Nedelkovski* [41]

```
#include <SoftwareSerial.h>
int bluetoothTx = 0;  // TX pin of HC-08 Bluetooth Module, Arduino D1
int bluetoothRx = 1;  // RX pin of HC-08 Bluetooth Module, Arduino D0,
// Rx is not needed since communication is only going one way

//create an instance of a SoftwareSerial object
SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);
```

```
#define button 8  // the number of the pushbutton pin
int buttonState = 0; // variable for reading the pushbutton status
void setup() {
  // initialize the LED pin as an output:
  pinMode(button, INPUT);
  //Default communication rate of the Bluetooth module
  Serial.begin(9600);
  // Begin The Bluetooth Module, default is 9600bps
  bluetooth.begin(9600);
}
void loop() {
 // read the state of the pushbutton value:
 buttonState = digitalRead(button);

 // Check if the pushbutton is pressed.
 // If it is, the buttonState is HIGH:
 if (buttonState == HIGH) {
 // Sends message with '1' to the master module
   Serial.print(bluetooth.write('1'));
   // Delay needed such that the signal is not sent more than once
   // on the time you hit the button
   delay(1000);
 }
}
```

Listing 4: Arduino code for master module. The code is based on a sketch made by *Dejan Nedelkovski* [41]

```
#include <SoftwareSerial.h>
int bluetoothTx = 0;  // TX pin of HC-08 Bluetooth Module, Arduino D1
int bluetoothRx = 1;  // RX pin of HC-08 Bluetooth Module, Arduino D0,
// Tx is not needed since communication is only going one way

//create an instance of a SoftwareSerial object
SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);
#define ledPin 7 // the number of the LED pin
int data = 0; // variable for recived data, starting value of 0

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // start with LED lamp off
  digitalWrite(ledPin, LOW);
  // Default communication rate of the Bluetooth module
  Serial.begin(9600);
  // Begin The Bluetooth Module, defaults 9600bps
  bluetooth.begin(9600);
}
void loop() {
```

```
  // Enables the bluetooth module to listen
  bluetooth.listen();
  // Checks whether data is comming from the serial port
 if(bluetooth.available() >0){
    data = bluetooth.read(); // Reads the data from the serial port
 }

  // Controlling the LED
  // If the recived data equals the message for button chlicked
 if (data == '1') { // The LED lamp will switch state
  digitalWrite(ledPin, !digitalRead(ledPin));
  data = 0; // variable for recived data changes to 0
 }
}
```