Auto-Docking Cat Toy

ECE 445 Final Report, Spring 2019

By Robert Malito, Yuhao Liu, Justin Li

TA: Soumithri Bala

Sponsor: Petronics, Inc.

Project No.60

Abstract

The goal of the project is to design, build, validate and analyze a method of robot self-docking using only simple infrared signal guidance in a minimal indoor environment setting. We placed an infrared LED array on the docking station and mounted one primary infrared receiver in a pin-holed housing on the robot platform sponsored by a local startup company Petronics. The receiver is connected to a NRF52832 BLE SoC with integrated Cortex M4. By establishing controls over the robot's built in Bluetooth function, we were able to command the robot to reach the docking platform of size $100mm \times 100mm$ with a success rate of 80%. The test setting we used is a $3m \times 3m$ empty lab room with standard lab floor, where the robot was placed in an arbitrary location with arbitrary initial facing.

Table of Contents

1. Introduction

1.1. Problem Statement and Solution

1.1.1. Background

An awesomely engaging cat toy called the Mousr has been developed and put into production by a startup company here on campus called Petronics[1]. Its small profile and nimble movement abilities make Mousr a very enticing toy for all felines. And the company developing the Mousr found out a problem that they wanted to be solved.

1.1.2. Problem Statement

While buyers are mostly satisfied with the Mousr product, the battery life has been addressed as the most limiting aspect of the device. Currently, the Mousr's battery can only support around 2 hours of continuous operation[2]. Buyers of this product would like to be able to leave the Mousr to entertain their cats for extended periods of time when they leave for work or vacation.

1.1.3. Solution Proposed

In order to solve this issue, Petronics came up with an idea to let the robot automatically go back to a charging dock when its battery level is low. We purpose a feasible way to identify the dock station and navigate the Mousr robot back to recharge itself.

Our goal is to enable the robot to detect and navigate back to the docking station in a relatively simple environment. Mousr should be able to do this quickly when battery percentage is less than 10%. We would also like to introduce as little extra cost as possible to the already built system.

1.2. Functionality Overview

The system we came up with had the following functionalities: IR signal modulation and Encoding, BLE communication, IR signal De-modulation on the receiver end, robot command interface and finally the path algorithm.

1.2.1. IR signal Modulation & Encoding

The IR modulation function is able to generate modulated 38 kHz infrared signal. And then the encoding module will encode modulated IR signal based on NEC standard which is commercially used by other electronics. Eventually the modulated and encoded signal will be output through PWM supported I/O pin and feed into the IR LED array. This allows us to send encoded message via infrared signal which can later be used as localization without interfered by natural infrared sources for example room lights and the sunlight.

1.2.2. BLE Communication

BLE communication is established between the nRF52832 development platform and the robot's built-in Bluetooth function. Both ends should be able to send and receive message via a simulated BLE_UART port in at 400 Byte/Second rate.

This allows us to establish and maintain a secured channel to send commands to the robot as well as to receive status updates from the robot. This allows us to have a precise control and feedback loop which is used in our navigation algorithm.

1.2.3. IR signal De-Modulation and Decoding

The IR de-modulation function is able to generate de-modulated and decode the message sent by the emitter on the dock when the receiver sees the signal. The decoded message will be then fed into one of the threads that handles the I/O interrupt and used in our navigation algorithm.

1.2.4. Robot Command Interface

This module is to pack commands and unpack status updates data packages. The multi-bytes-long commands are packed in little endian and in accordance with the API format provided by our sponsor. The packed message will be then sent to the robot via BLE_UART channel mentioned above. On the other hand, the received data will be handled by another interrupt handler that handles incoming message from BLE. After that we unpacked data and updated status in our algorithm.

1.2.5. Navigation Algorithm

One of the most important part of this project is the navigation algorithm. It uses the signal decoded from the IR message hander and calculate the percentage of successfully decoded package. We then use the percentage to determine the relative signal strength and signal boundary. According to the boundary of signal we will command the robot to turn towards the center of two mapped boundaries and then processed until signal is lost or it has reached the destination. This part is most essential to our design because it determines the success rate of the design.

1.3. Subsystem Overview

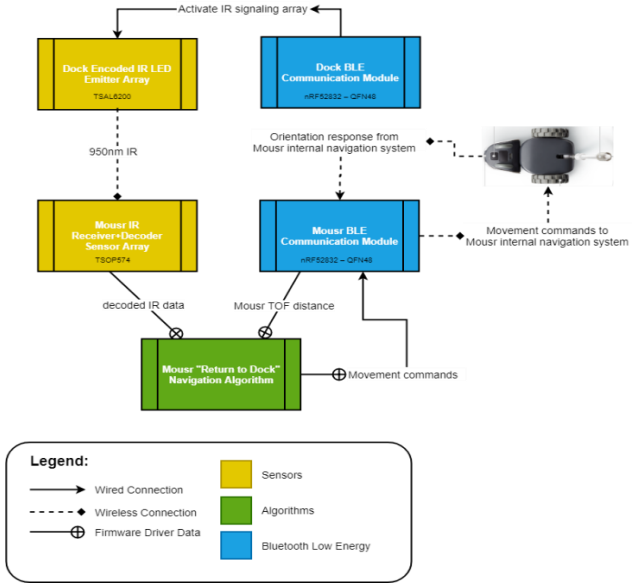

Figure 1. High level block diagram

1.3.1. Docking Station Subsystem

The docking station holds the infrared LEDs and modulate and encode the IR data. The encoded data then transmitted via infrared emission to the IR receiver.

1.3.2. Receiver and nRF52 Platform subsystem

The receiver and nRF52 platform is used to decode IR data, to send movement commands to and receive updates from the robot via BLE UART. Status update package include the robot's current yaw, pitch, and roll and TOF value.

1.3.3. Robot Subsystem

The robot subsystem is a platform provided by our sponsor. This subsystem executes the movements command and reply to the nRF52 platform with status packages via BLE. And the robot also holds the extra sensor assembly with the pin-holed housing.

2. Design

2.1. Equations and Simulations

2.1.1. Emitter Dock

The first stages of the dock's physical design process required a functioning emitter driver. Once the relevant emitter firmware had been written it was time to simulate the infrared LED's by visualizing the output on an oscilloscope. The output signal needed to be modulated as close to 38 kHz as possible in order to correspond to the center of the bandpass filter of our infrared receiver.

$$frequency = \frac{1}{period}$$

$$period = \frac{1}{38000} = 26.315789\ \mu s$$

Taking the period in microseconds from the equation above at a 50% duty cycle yields a 13 µs high and a 13 µs low. This period and duty cycle achieve a close approximation of the desired frequency spectrum. With modulation mathematically sound, the binary encoding can be done. The chosen IR encoding protocol was the NEC encoding standard. This features a high of 560 µs, low of 560 µs for a logical 0, and a high of 560 µs, low of 1690 µs for a logical 1. The success of correct microsecond precision of both 38 kHz modulation and implementation of NEC encoding standard was visualized using an oscilloscope. This simulation was the first step towards functioning IR communication. Figure 2 shows in lab simulations for IR communication:

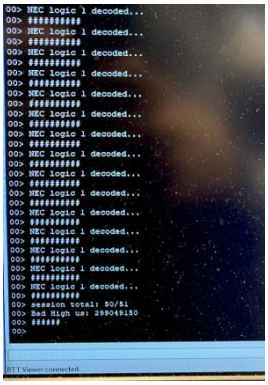

Figure 2. RTT output from IR communication testing

2.1.2. Receiver Mounted on Mousr

Similarly, to the emitters, a receiver driver was the first step in running lab simulations with the IR receiver. Through various trials of driver firmware modification and recompiling, a successful infrared receiver driver and hardware was confirmed. This confirmation was achieved by applying a function generator to the GPIO pin the receiver functions on. The function generator settings chosen were calculated as follows:

$$NEC\ logical\ 1\ \ period = 560\ \mu s\ (\text{high}) + 1690\ \mu s\ (\text{low}) = 2250\ \mu s$$

$$frequency = \frac{1}{2250\ \mu s} = 444.4444\ Hz$$

$$duty\ cycle = \frac{560\ \mu s}{2250\ \mu s} = 24.88\%$$

Figure 3 shown below gives the radial response of the IR receiver in simulation:

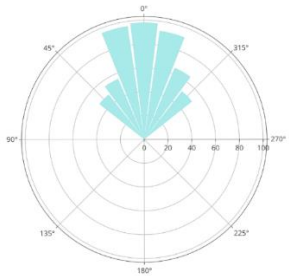

Figure 3. Radial receiver response visualization

2.1.3. BLE Bi-Directional Communication

Troubleshooting different Bluetooth simulations was the main method of achieving the ultimate goal of a development board acting as a BLE central and the Mousr acting as peripheral. The first test involved connecting two development boards together and controlling one board's LEDs wirelessly. As this was accomplished, the next test was sending proprietary commands from an Android Phone running nRFConnect to the Mousr. Once confirmation of command sending was established, it was possible to connect to Mousr's BLE Nordic UART Service from a development board and begin bi-directional communication.

2.1.4. Navigation Algorithm

The navigation algorithm implemented to guide Mousr back to its dock increases its precision as the session progresses. The number of degrees turned per rotation is fixed between 22.5° = $(\frac{360}{16})$ and 5.625° = $(\frac{360}{64})$. This range was chosen due to two factors. The angle of half intensity for the IR emitters is 17.5°, and the most precise yaw adjustment the Mousr is capable of making is 5°. The precision thresholds were chosen based on these empirical cutoffs, and countless early simulations. The navigation also performs an angle bisection to estimate the signal center relative to its current location. The right and left signal falloff edges found by the algorithm are positive and negative angles respectively. The bisector is calculated as follows:

$$yaw_{goal} = \frac{right_{edge} + left_{edge}}{2}$$

2.2. Design Alternatives

2.2.1. Emitter Dock

During the early stages of our IR communication tests we noticed that the wide-angle emitters were performing far worse than the narrow angle emitters. The main area of difference seemed to be in the distance the receiver was able to detect a signal. The narrow angle emitters offered a range greater than 3 meters; the wide angle was struggling to achieve 1-meter range. Due to the shift in hardware there was a corresponding shift in design. The initial design consisted of using a narrow angle emitter in the center and wide-angle emitters on the sides. This was

replaced by a narrow emitter “bouquet” to achieve maximum distance and dispersion from a near point-source. This change increased the range and performance of our design.

The PCB design crafted for the emitters was simple but lacked forethought. As seen in Figure 4 below there are few components. The nRF52 development board was unable to directly supply the power needed to drive all the IR LEDs. In order to achieve the correct operational voltage and current (1.5V, 0.1A), an NPN bipolar junction transistor (BJT) was needed. The microcontroller was successfully able to activate the correct power supply through the BJT[3].

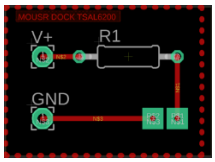

Figure 4. Simple LED PCB

The choice was made to strobe each LED in the “bouquet” with a 100 milliseconds timeout between each pulse. This design choice aims to improve the life expectancy of the LEDs while avoiding dangers of interference and phase offset.

Lastly, in simulation, the NEC logical 1 had a better decode rate by the receiver than the NEC logical 0. This may be related to their respective duty cycles (24.88% vs. 50%) and the decoding strategy implemented.

2.2.2. Receiver Mounted on Mousr

Initial testing of the IR receiver proved extremely noisy [8]. The bare surface mount device soldered to the printed circuit board was receiving signals from the emitters at angles as wide as 80°. Having a 160° frontal cone was not useful for the precise direction strategy that was envisioned (see Figure 5 below). The decision made to prevent this noise included building a housing for the receiver, painting the inside black to absorb radiation, and pin-holing the housing in alignment with the SMD chip; see Figure 5 below.

The SoftDevice, signal processing propagation, and activity on other Bluetooth threads seemed to be skewing the microsecond precision wave measurements used to classify the received IR. In order to counter this, the accepted range of decoding NEC bits was expanded and relaxed. The rigid decoding worked during function generator testing because there were no other concurrent Bluetooth processes or real wave propagation.

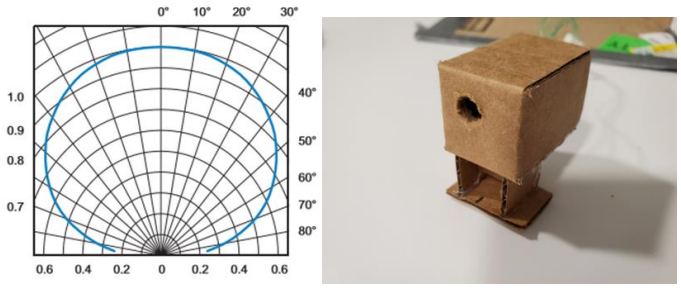

Figure 5. Receiver FOV datasheet and pin-holed housing

2.2.3. BLE Bi-Directional Communication

When working with BLE services there were some issues that were not immediately prepared for. One commonplace mishap encountered was endianness of packed bytes sent and received over Nordic UART Service. This was solved relatively quickly through careful hex byte analysis. Two other more persistent issues were encountered when working on the nRF52 framework. The first was blocking vs non-blocking delays. In order to ensure synchronization of commands sent to Mousr *nrf_delay* was used. This delay is blocking and greedily holds up processor cycles. The development boards would often encounter a fatal error during these delays. To solve this, a low-frequency application timer was configured. This had its own handler that was called when the delay expired and allowed for other threads to progress while another thread was in a non-blocking delay.

The second issue involved successfully sharing resources between threads and was an issue for several days. A *memcpy* was the source of the issue. In C, *memcpy* is built for speed, not safety. Mutex spinlocks were attempted to prevent issues, but they did not working due to the SoftDevice configuration. In the end, global Booleans dynamically allocated on the heap were used to overcome the resource management troubles.

2.2.4. Navigation Algorithm

In early testing, as Mousr approached the dock it would often turn too widely for the pinhole to detect an IR signal. If Mousr started far from the dock it could also miss the incoming IR with too large of rotations. The adjustment made to solve this was varied turn precision. The variation starts with wide 22.5° turns. If no signal is found the precision is increased (decreasing the angle). When Mousr is moving towards a signal the precision is also increased if the signal is lost and the search state begins again.

When making 5.625° turns (nearly the lowest offered by the hardware of Mousr) the motor will sometimes fail to overcome the friction between the wheels and the floor. This failure seems to be specifically noticeable when the motor is switching polarity. If the previous turn made by Mousr was clockwise and the next turn is counter-clockwise, the Mousr seems to have increased trouble making this pivot. Petronics admitted to giving our group an ancient Mousr to work with. This could be an additional source of motor failure and inaccuracy.

Debugging the firmware for the navigation algorithm was tricky. The built-in logging feature *NRF_LOG* lacks formatting abilities and can be printed at incorrect times due to SoftDevice. A third-party logging function called *SEGGER_RTT_printf* was the most useful for formatting and printing floating point and dynamic (heap allocated) variables. A balance between both log functions was necessary for complete debugging.

2.3. Design Description and Justification

2.3.1. Emitter Dock

The emitter dock consists of an nRF52 development board powering nine individual 940nm IR LEDs. No two LEDs in the array are on at the same time. The LED's follow a repeating strobe pattern that takes roughly 150 – 200 milliseconds for each LED to pulse and cool down. During the pulse, an LED sends 50 NEC logical 1's. The pulses are modulated manually very near 38 kHz. These modulations are physically filtered by the IR receiver's hardware. The pulses (highs) and lows are generated using a system timer on the development board and are output over

analog out to the “base” pin of a BJT transistor. The goal with the LED arrangement on the dock is to create approximately 180° of IR light coming from as close to a point source as possible with the PCB size limitations. This will allow the pin-holed receiver to make accurate decisions from anywhere in front of the charging dock.

2.3.2. Receiver Mounted on Mousr

The IR receiver mounted onto Mousr is a wide angle SMD built for applications such as TV remote sensing; see Figure 6 below. The addition of radiation absorbing housing as well as a pinhole reduce the interference incident to the tiny SMD chip and increase the precision Mousr has when determining where the IR signal is coming from. The data output of the receiver hardware is connected to the development board’s General Purpose I/O (GPIO) pin. When the receiver detects IR in its bandpass range (near 38 kHz) it outputs the signal to the GPIO pin and triggers a GPIO event handler every polarity toggle. This handler takes advantage of a high frequency system timer to measure the length of the pulse and the low in microseconds. With both data points recorded the times can be decoded into the corresponding NEC bit value (example: 560 µs pulse followed by 1690 µs low decodes to a logical 1).

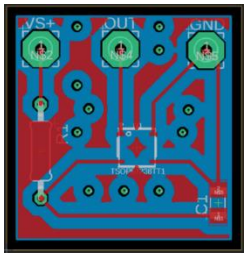

Figure 6. PCB for IR receiver

2.3.3. BLE Bi-Directional Communication

The communication channel established to communicate with Mousr was built for emulated UART over BLE. The development board serves as the central device, the Mousr serves as the peripheral. Once the central has connected to the peripherals broadcasted Nordic UART Service (NUS), raw bytes can be sent and received over the emulated serial port. Movement commands are sent to Mousr as 15-byte arguments, and often contain a yaw goal for Mousr to rotate to. Orientation response packets occur at a rate of around 20 Hz and are 20-bytes in length. The exact composition of these byte commands is proprietary to Mousr’s creator, and are not relevant to this document.

2.3.4. Navigation Algorithm

The main ideas of the navigation algorithm are aligning Mousr to the center of the incident IR light before it moves forward and increasing the turn precision (reducing angle) as the distance to the dock decreases. The navigation scheme assumes Mousr is within range of the IR radiation, and that Mousr will start in an arbitrary direction not necessarily facing IR emitter source.

The first state of the algorithm involves finding the relative yaw at which the Mousr can receive the signal. This is achieved with a counter-clockwise 90° “blind spot” check followed by a clockwise rotation of 360° + 90°. These turns are made progressively at the rate of the current angle of precision. If no signal is found after the full clockwise turn, the angle of precision is divided by 2 resulting in twice as many turns in a rotation as the previous precision. Once the

signal direction is found from the point turn the next phase of edge detection begins. The edges are where the number of IR bits decoded drops below a threshold defined through testing. Edge detection accuracy is achieved through leveraging the narrow incident light angle of the pin holed receiver housing. Both the right and left edge angle values are captured using high precision (11.25° – 5.125°) rotations. Using these edge angle values, the bisector of these angles is a yaw goal that Mousr should most likely follow to get closer to the center of the dock. With the right and left edges found and the bisector calculated the Mousr can rotate and move forward to the desired yaw. One the IR signal drops below the detection threshold, a stop command is sent to Mousr and the entire process restarts from finding the direction of the signal; see Figure 7 below. In general, the angle precision is always kept between 22.5° and 5.125° and is attempted to be reduced each step of the Mousr's journey back to dock.

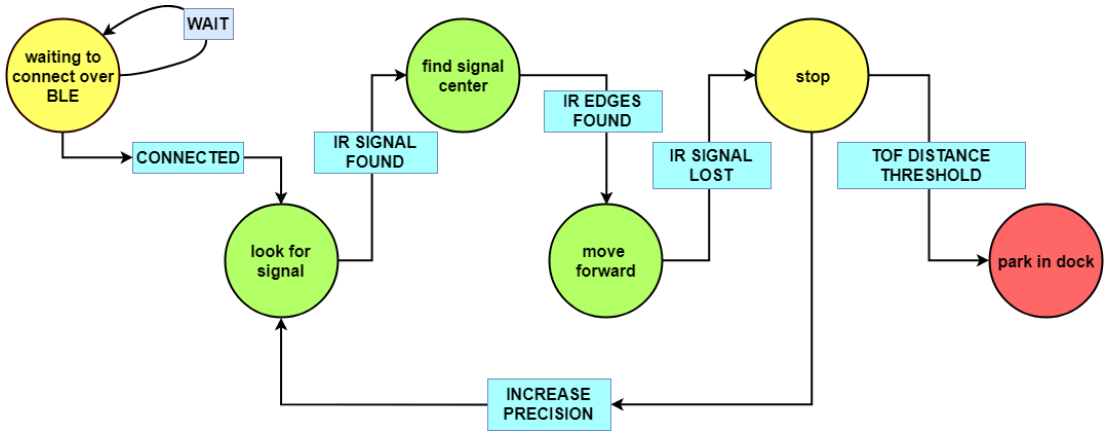

Figure 7. High level navigation state machine

3. Cost and Schedule

3.1. Summary of Cost

3.1.1. Cost of Material

Different from the planned cost of material, we added some extra cost for purchasing extra modules for both initial testing and serving as contingency plan if the sensor we made wouldn't work. So extra cost for tools and jump wires are also included. And the total estimated cost is $208.2 (See Table 1 in Appendix B).

3.1.2. Cost of Labor

Based on the fact shown on ECE Illinois website[4], we calculated an estimate hourly salary of $38/hour. One average we worked 8 hours per week for each member during the semester. And we consider 90% of the work during the 13-week-long project, except our regular meetings with Petronics.

$$3 \times \$38/hr \times 8hrs/week \times 13weeks/0.9 = \$13173.33$$

3.1.3. Cost of Production

Furthermore, for each piece that will potentially be produced in the future, we have a $10.786 BOM cost introduced into the already existed BOM cost (See Table 2 in Appendix B).

3.1.4. Summary

By summarizing the cost above, we have a total cost of $13381.53 for developing the project in a 13-week-long period. And the extra cost will be introduced to the actual production of the product will be $10.786.

3.2. Schedule

3.2.1. Progress Timeline

(See Appendix B)

3.2.2. Reflection on Schedule

We mostly stuck with our planned project timeline with some delay in March when there were many midterms and spring break. And there was another event of the PCB order being delayed for more than one week. We also spent time meeting regularly with the sponsor during the end of the semester and did some intensive testing and debugging with their technical team which is not listed in the schedule.

4. Requirements and Verification

See Appendix A for Requirements and Verification tables

4.1. Sensor Functionality

The conditions we set for our project was that the dock would be able to transmit an encoded 950 nm wavelength infrared signal that the Mousr's receiver can detect and decode with a range of around 3 meters. This signal must also be decoded in less than a second and must not be mistaken for household infrared noise, and the receiver must not mistake any household infrared noise as the emitter's signal.

To verify these conditions, we fed the receiver with the desired encoded infrared signal at around three meters away and see if the receiver had gotten the correct decoded data and that the timespan of each signal was correct (the desired duration of each 'high' signal is 560 microseconds); see Figure 8 below. Once it was determined that the receiver had properly received the desired infrared signal, we then tested it again with additional infrared noise in the room to confirm that it can tell the difference between our desired signal and infrared noise.

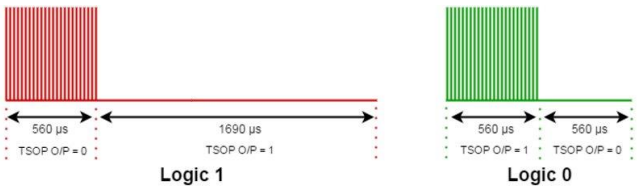

Figure 8. NEC infrared encoding standard

In addition, we also tested for the sensor's ability to receive these signal packets within a small period of time at different angles. We determined that without any sort of casing to restrict the field of view of the sensor the sensor will pick up signal packets from all angles, including directly behind. This was most likely due to the emitted signals bouncing off the walls of the room we were testing in rather than background noise being interpreted as signals. We afterwards resorted to putting the sensor within a cardboard casing with a pinhole in it to focus the sensor's vision cone

We had plans to also make use of the Mousr's belly sensor to detect the texture of the floor in order to adjust the Mousr's decision making with the use of a convolutional neural network. The requirement was for the Mousr to detect whether the floor was rough, smooth, or "too noisy" within five seconds with around 65% accuracy. This would have been verified by testing the sensor on a variety of smooth and rough surfaces to check for how it will classify the surfaces, then testing it under simulated "noisy" conditions to see how it would take measurements in unpredictable instances such as being rocked around on bumpy surfaces. We didn't get around to testing for this as it was a secondary concern since our testing environment will always be on a smooth surface since it is inside the ECE Building, though it could be a possible feature in the future.

Another requirement that we wanted to test was for the underbelly sensor to accurately measure the distance from it to the dock within 10ms and within a millimeter of error. This

would be done using the Mousr's Time of Flight function. To perform verification, the sensor would be fixed to a stable surface, then an emitter at a specified measured distance will send the signal. We then compared the distance that we measured physically to the distance that the sensor measured, and how many measurements the sensor made in a second. When we got to testing, determining distance based on Time of Flight was rather erratic and unreliable, with errors on average from 10 to 20 millimeters. We also noticed that when the sensor is right up at the emitter, the distance measured is about 3 centimeters which could be reasoned as being the distance from the outer casing of the Mousr to the sensor itself within.

4.2. Emitter Functionality

We had to also test that the emitters (TSAL6200 LEDs) will properly function when supplied with a 1.5 voltage source and a supplied current of 100 milliamps. This was done immediately after soldering the emitter boards by hooking up the emitter board to the DC power generator and increasing the voltage steadily from 0 volts to 1.5 volts. As infrared was not observable to the naked eye, we used our phone's unpolarized cameras to view the infrared lights. The first few times our LEDs didn't work properly as a result of the resistors used being too large in value, of which we then replaced with smaller resistors and tested again. Through this verification we determined that the optimal resistor to put in series to a TSAL6200 LED with an operating point of 1.5 volts and 100 milliamps has a resistance of 33 Ohms.

We also needed to verify that the LEDs were strobing in a regular pattern instead of constantly been lit up. This was to make sure that the LED infrared signals did not directly interfere with each other and to improve the operational life expectancy for each of the LEDs as strobing will allow the LEDs to cool down between activation states. To do so, we hooked up the LED boards to our control board and again used our phone cameras to check that the LEDs were strobing correctly (in sequence and not simultaneously).

4.3. BLE Communication

The receiver must be able to successfully communicate with the Mousr's internal hardware given signals from the charging dock indoors at a distance up to 8 meters. The testing was done by first putting the Mousr at a location within the room. This would be repeated ten times, each time with the Mousr at a different position. The charging dock in each of these tests will remain in the same place. We then confirmed whether the receiver would react to an infrared signal being sent to it.

We also tested for the distance that the receiver and dock successfully communicate. This was done by placing the sensor in front of the dock and with each test moving it back 0.5 meters, up to 8 meters. It was observed that at the maximum tested distance of 8 meters the dock and receiver still was able to communicate each other without significant issue.

4.4. Navigation

For the device to work, the receiver must be able to successfully communicate with the Mousr via Bluetooth and feed it the correct commands to navigate it back to the charging dock with a high degree of reliability (about 85% chance to successfully navigate back to the dock). The Mousr must also know to stop once it has successfully arrived on the dock.

For the completed project, we tested the refitted Mousr's ability to navigate back to the test charging dock given several different positions and orientations. The conditions we tested for to prove its success was a combination of when the Mousr was to the left, right, or direct front of the dock; when it was facing directly towards the dock, at a 90 degree angle to the right or left away from the dock, or facing directly opposite to the dock; and finally testing for when it was close to the dock at increments of 0.5 meters, up to 5 meters.

Over the course of 30 trials, 25 of them had the Mousr successfully return to the charging dock, which roughly meant we met our reliability goal. Each of these trials were organized into sets that tested the Mousr's response to variation in one of the conditions mentioned above, starting with seeing if the Mousr will correctly determine the direction to head towards the charging dock, then testing if the Mousr will correctly dock given different angles of approach, and finally with the last few trials tested for the maximum range of the Mousr by putting it further and further away from the charge dock. At the maximum tested range, the Mousr was able to successfully navigate back to the dock, and the Mousr was able to guide itself back to the dock even when it started off facing away from the emitter. The instances where the Mousr did not successfully dock were instances where the Mousr was entering the dock at an angle where it would lose sight of the emitter after passing the dock. We revised the coding of the receiver so that the detection cone of the receiver was narrower by telling it to only accept signals above a certain intensity (in this instance interpreted from the number of packets received) while scanning. This effectively reduced the vision cone from 180 degrees of vision to 90 degrees, reducing the number of instances where the Mousr missed its target.

5. Conclusion

5.1. Accomplishment

The goal of the project is to design, build, validate and analyze a method of robot self-docking using only simple infrared signal guidance in a minimal indoor environment setting. First, we successfully establish connection with the robot via emulated UART channel over BLE, which can handle more than 400 Bytes of information per second. Next, we were able to modulate and encode the data using NEC standard and transmit through infrared signal, as well as decode the signal on the receiver end. Finally, we successfully conducted over 30 tests, in which we were able to command the robot to reach the docking platform of size $100mm \times 100mm$ with a success rate of 80%. The test environment we had is a $3m \times 3m$ empty lab room with standard lab floor, where the robot was placed in an arbitrary location with arbitrary initial facing.

5.2. Uncertainties and Unsatisfactory Result

5.2.1. Algorithm Efficiency

The efficiency of the algorithm has not met without expectation right now. The average for the robot to reach the docking station is 1 minute. In the worst-case scenario, it may even take up to 2 minutes to reach back to its dock.

One cause of the problem is we are not familiar with how the timer works with the NRF52832 platform. If we handle the interruption event raised by sensors and packages coming back from the robot too frequently, the system can crash easily. So, for this reason we must tune down the frequency of processing interruption to receiving 50 packets per 200ms with an 100ms wait period and thus causing a rather slow performance.

5.2.2. Edge Cases that cause failure

In the $3m \times 3m$ test environment we setup now, there are blind spots as illustrated in the figure 9 below. The orange box is the docking station, and the grey area on each side of the dock that each forms an 30° of angle from the wall.

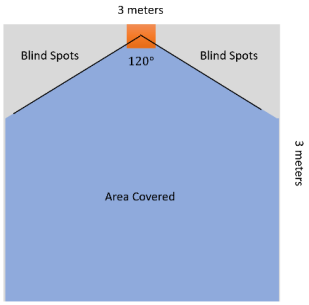

Figure 9. Test environment scale model

When placed in the blind spot area, the robot will not able to find the signal source from the docking station, and it will do nothing but keep sweeping for signal.

The potential cause of comes from the following design reasons. First, the LED emitter array we set up has only a 120° angle of coverage. So that the robot will not receive any signal in the blind spot area. And then the dock is not properly design to let the robot enter from the side. If the robot approaches the dock from an angle larger than 30° from the center line of the room, it will not able to enter the dock.

5.3. Future Works

5.3.1. Mousr Integration

The design now uses an extra NRF52832 chip on the outside of the robot and controls the robot via Bluetooth. This is a specific requirement from our sponsor. However, during the actual production of the updated version of the robot, there will only be one NRF52832 SoC integrated in the robot. And the next step is to migrate the software into the robot's system and test its performance.

5.3.2. Wireless Charing

Wireless charging is another way to improve the success rate for the robot to charge. Currently, the charging mechanism relies on accurate alignment of the electrodes. If instead, we use wireless charging, which has less requirements in alignment than traditional electrodes charging, we will be able to further increase the success rate of charging.

5.3.3. Machine Learning

A hot topic right now is machine learning. In our case, we can explore further the possibility of integrating signal pattern recognition to our design. With certain level of distinguishable signal pattern created by dock, the robot can be able to determine its own relative position.

5.4. Ethical Considerations

There are several safety and ethics issues that are relevant to our project. Pertaining to point #1 of the IEEE Code of Ethics [5], we must ensure that the materials we use to build the Mousr is non-toxic to pets, as cats tend to hold things in their mouths and such the cat may accidentally ingest the material. A thing that must also be taken into consideration is the product's impact on the environment, whether the materials it is made of can be potential pollutants such as the outer shell and battery.

According to the user guide for Mousr [6], the device uses a lithium-ion polymer battery which contains hazardous materials. To avoid harm, the battery must not be overcharged and must not be exposed to extreme temperatures. The battery must also not be left to charge overnight because of issues that can be caused from overcharging. We should figure out a way to program the device so that it becomes active and leaves the charging station when it has detected that it is at full battery.

The sensor of the Mousr could also pose a potential privacy issue to the end-user. We must ensure that the data that the Mousr collects to navigate will not be used for malicious means,

such as making sure the data cannot be transmitted from the device and that it cannot be used by third parties.

6. Acknowledgement

We would like to express our deepest appreciation to all those who provided us the possibility to complete this report. A special gratitude we give to our senior design project instructor, Professor Jiang, whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project.

Furthermore, we would also like to acknowledge with much appreciation the crucial role of the staff at Petronics, Mr. Dave Cohen and Mr. Russell Jones who not only provided resources to make this project possible, but as well generously offer technical support when we had problem.

Last but not least, many thanks go to the two of the teaching assistants, Mr. Nicholas Ratajczyk and Mr. Soumithri Bala, who have invested their full effort in guiding the team to achieve the goal.

7. References

[1][2] Petronics. "Mousr." Petronics. [Online]. Available: https://petronics.io/products/mousr

[3] Hewes, J. (2019). Electronics Club - Transistor Circuits - functional model, base, collector, emitter, use as switch, inverter, Darlington pair. [online] Electronicsclub.info. Available at: https://electronicsclub.info/transistorcircuits.htm.

[4] ECE Illinois, https://ece.illinois.edu/about/rankings-and-statistics.asp

[5] "IEEE Code of Ethics." IEEE - Advancing Technology for Humanity. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[6] Mousr User Guide. PDF. Champaign: Petronics, October 24, 2018. [Online]. Available: http://static.petronics.io/manuals/mousr-user-guide.pdf

[7] "UNITED STATES DEPARTMENT OF LABOR." Occupational Safety and Health Administration. [Online]. Available: https://www.osha.gov/enforcement/directives/std-01-12-002

[8] R. Malito, “nRF52 DK improper GPIOTE --> PPI configuration (no interrupts being raised on input toggle),” Nordic DevZone. [Online]. Available: https://devzone.nordicsemi.com/f/nordic-q-a/46152/nrf52-dk-improper-gpiote----ppi-configuration-no-interrupts-being-raised-on-input-toggle. [Accessed: 02-May-2019].

[9] Vishay. "TSAL6200 Product Information." Vishay. [Online]. Available: https://www.vishay.com/ir-emitting-diodes/list/product-81010/tab/specifications/ s

[10] Vishay. "TSOP572.., TSOP574.. Product Information." Vishay. [Online]. Available: https://www.vishay.com/ir-receiver-modules/list/product-82434/tab/documents/

[11] Vishay. "CQY36N Product Information." Vishay. [Online]. Available: https://www.vishay.com/ir-emitting-diodes/list/product-81001/tab/documents/

Appendix A: Requirements and Verification Table

Table 1. Requirements and Verification Table

Requirements	Verification
Must be able to transmit encoded IR @ 950nm that receiver can detect and decode in realistic indoor environments with a range > 3 meters.	A. Output IR receiver's received intensity in a lab test taking measurements to determine the upper and lower bounds of intensity range. B. Once the range of the emitter has been established, lab tests within the newly defined distance bounds can confirm that the emitter is able to transmit the encoded signal correctly
Must be able to decode IR @ 950nm signal in less than 1s and not misinterpret household IR noise	A. Feed receiver array with targeted IR signal frequencies B. Pull out decoded data and time elapsed C. Compare the results with the correct value and time spent D. Retest transmission with IR noise in room
Must be able to take a burst (7-15) of mm-accuracy distance measurements within 10ms	A. Fix sensor to stable surface B. Compare physically measured distance to sensor measured distance (in mm) C. See how many measurements can be taken in a second and divide to get number per ms
Must return whether surface is "Rough", "Smooth" or "Too Noisy" in under 5000ms with moderate accuracy (65%+)	A. Test sensor and algorithm on a multitude of flat and rough surfaces to ensure correct classification B. Test a variety of noisy measurements to simulate Mousr's unpredictable state during usage C. Use data from tests to compile accuracy and speed requirement validation
Must be able to output the correct commands to navigate Mousr to dock when in receiving range with a high degree of reliability (85%+ success rate in testing).	A. Run a series of lab tests and confirm that algorithm is outputting the correct movement command based upon forced IR inputs B. Run tests simulating actual returns to charging dock C. Compile data from test to establish success rate in ideal conditions

Must be able to successfully establish a line of wireless communication between the dock and the Mousr indoors at up to 8m	A. Pair the Mousr robot with the dock station and send data packages to the dock B. Move the robot 0.5m further from the dock each time to see when will the dock stop receiving the data package C. Determine the maximum range of transmission from the robot to the dock
Must be able to successfully communicate with charging dock and Mousr internals indoors at up to 8m	A. Put the Mousr robot at 5-10 distinctive locations within the room B. Fix the charging station's position C. Try sending command that switch the IR sensor ON and OFF to the station see if the station reacts.

Appendix B: Tables and Figures

Table 2. Cost of Material

Parts	Part No.	Unit Cost	Number	Subtotal
Bluetooth Dev Kit	PCA10040	$39	2	$78
Bluetooth SoC	nRF52832	$2.83	4	$21.32
IR Transmitter A	CQY36N	$0.468	20	$9.72
IR Transmitter B	TSAL6200	$0.37	20	$7.4
IR Receiver	TSOP57436	$1.47	10	$14.7
Misc. Circuit Components	/	$0.1	50	$5
IR LED and Receiver Pack	/	$6.98	1	$6.98
Receiver and LED module	/	$7.99	2	$15.98
Transistor Pack	/	$14.85	1	$14.85
Hot Glue Gun	/	$9.25	1	$9.25
Jump Wires	/	$0.25	20	$5
Shipping				$20
Total Cost				**$208.2**

Table 3. Cost of Production

Parts	Part No.	Unit Cost	Number	Subtotal
IR Transmitter B	TSAL6200	$0.152	9	$1.368
Bluetooth SoC	nRF52832	$2.71	1	$2.71
IR Receiver	TSOP57436	$0.798	1	$0.798
Misc. Circuit Components	/	$0.1	50	$1
Dock Housing	/	$5	1	$5
Total Cost				**$10.786**

Table 4. Progress Timeline

No.	Week	Robert	Yuhao	Justin
1	2/4/2019	Project proposal	Project proposal	Project proposal
2	2/11/2019	Contact with Petronics	Gather List of material	
3	2/18/2019	Design Documentation	Design Documentation	Design Documentation
4	2/25/2019	Schematics refine and get familiar with the SDK	Get familiar with the SDK and PCB layout	
5	3/4/2019	PCB final check, Software environment setup	Software environment setup, research on how IR behaves	
6	3/11/2019	Model the dock	Test version 1 software	
7	3/18/2019		Modify software	

8	3/25/2019	Test software version 2	Test software version 2	
9	4/1/2019	Integration of hardware and software platform	Integration of hardware and software platform	
10	4/8/2019	Test software version 3	Test software version 3	
11	4/15/2019	Final integration	Final integration	Soldering PCB
12	4/22/2019	Prepare presentation and final software test	Prepare presentation and final software test	Prepare presentation and final software test
13	4/29/2019	Final presentation and report	Final presentation and report	Final presentation and report