Team Design Paper: Andy's Geese

John Qing, Matteo Cericola, Yuan Yoshimi, Sean Yien, Josh Wang (retired)

Dr. Benjamin Lawrence, Ms. Meera Balendran

St. Andrew's College

15800 Yonge Street, Aurora, ON, CA

Abstract

This report serves as the documentation and description for team Andy's Geese in preparation for RoboCup Junior 2024. In this paper, we discuss aspects of our team and the composition of our robots, as well as details of learning. We will begin by describing our team, then identifying the composition of the technology in our robot, and finally discussing some future steps as well as what we had learned.

Introduction

Team Background

Team Photo

L to R: Yuan Yoshimi, Matteo Cericola, Sean Yien, John Qing

Andy's Geese

Team Foundation

Andy's Geese was first formed in February 2024 in preparation for RoboCup 2024. We were planning on competing in our regional tournaments in April of 2024 and possibly in the world tournament in July of 2024. Although our team formed very recently, our founding members were part of the DYMJ-Alpha (picture to the right) who competed in Bordeaux in 2023. Other members also brought a wealth of experience, with one hailing from the Canadian Rescue Line team that participated in the Bordeaux competition, and the other with a wealth of experience in embedded systems programming and a mastery in C++.

L to R: John Qing, David Yang, David Li, Matteo Cericola

DYMJ-Alpha

Team Composition

Our team consists of four people, with one former member who retired from active participation in RoboCup Junior.

Active Members:

John Qing – Engineer, Programmer, Electrical Design, Design Paper Writer, Data Analysis

Matteo Cericola – Programmer, Engineer, Data Analysis

Yuan Yoshimi – Engineer, Programmer, Data Analysis, 3D Designer, Mechanical Deisnger

Sean Yien – Electrical Design, Engineer, Poster Design

Former Members:

Josh Wang – Programmer, Poster Design

Team Accomplishments and Qualification

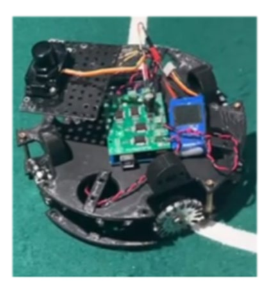
Our team achieved both first place in the regular competition as well as the Superteam competition in the RoboCup Junior Regional Tournament in Montreal, Quebec. We faced many competent teams from Eastern Canada. Less than a week later, our team also achieved first place in the RoboCup Junior National Tournament in Aurora, Ontario (pictured to the left).

After receiving word of our qualification to the RCJ World Championships in Eindhoven, we began work on our final iteration of our robot we intend to bring to the Netherlands.

Robot Development

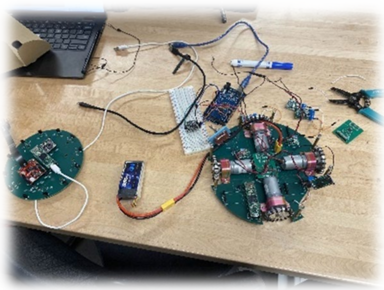
Design Overview

Our robot has undergone many iterations since the first protype design. However, we maintained a similar basic functionality all throughout. We identified the need for robot movement, orientation and ball detection as the required base functionality and therefore the highest priority to complete. As such, the first prototype iteration of the robot to explore these technologies included these three aspects of RCJ Soccer. The image to the right illustrates the prototype iteration of our robot, with the Modern Robotics IR Locator 360[1], the Progressive Automations MultiMoto Arduino Shield with 4 H-Bridges[2],



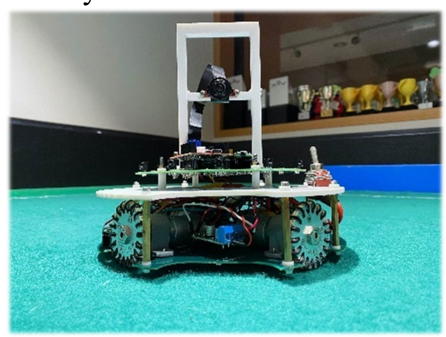
Protype Iteration

a CMPS-12[3] (with BNO055 IC), and an Arduino Mega[4] as the processor (see footnotes for reasons we no longer use these).



The next iteration of the robot, the Regional Competition Iteration, we replaced multiple components and introduced the use of printed circuit boards to facilitate the transfer of data, as well as to provide a solid base and electrical pathways for the power supply. We also introduced the use of a 5V and a 3.3V step-down regulator in order to feed proper voltages to the new Teensy 4.1, BNO085, and ADG732 (referred to internally as the MuxBreakout). The image on the left shows all components mounted on both the top and bottom PCBs.

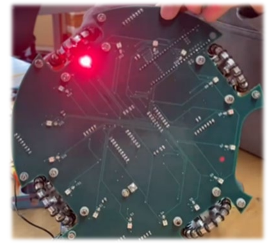
We stopped using the IR Locator 360 due to the unreliability as well as the halted production of the component. Instead, we opted for an IR Ring that employed the use of the TSSP4038, an IR receiver produced by Vishay. We also created a line sensor array with a colour sensor photodiode from Advanced Photonix, the 019-141-411-R SMD photodiode. The first iteration of the line sensor array used the OSRAM LS G6SP.02 SMD LED[5] (see footnotes for reasons we no longer use this component). An OpenMV RT1062 was also mounted on the IR Ring and connected using extension cables to the mounting position on the handle of the robot.



The final iteration of the robot, the World Competition Iteration, involved all of the previous capabilities, with improvements to several systems and components. The bottom PCB had some fixes for mistakes on previous boards, different components, and filleted edges. Some problems with the MuxBreakout were also fixed, like a misplaced trace that was not caught by the DRC in Fusion 360 Electronics. All the printed circuit boards were switched to black, in order to minimize disruption to the IR sensors from reflected IR signatures.

The IR PCB was heavily modified. The sensor array was changed to 24 sensors, with 12 of them using a low-pass filter to convert the reading to an analog voltage based off distance, then read through the ADC (Analog to Digital Converter) on the Teensy 4.1 mounted on the IR Ring and the other 12 being directly read from the digital pins of the Teensy 4.1 by measuring the length of the pulses. This pulse measurement system will be explained further in the electronics section. The camera's communication system was also switched over to SPI, to account for the difficulty in opening more than two serial ports on the Teensy 4.1. This will be discussed further in the technical sections.

Our game strategy was primitive in the Prototype Iteration. The only objective was for the robot to move forwards when the ball was directly in front. There was a BNO055 to do some adjustments, in case the robot ever stopped pointing forwards. The robot could not detect lines and would regularly violate the rules in search of the ball.



The game strategy for the Regional Competition Iteration was slightly more complex, using a better BNO085 to adjust for angle, as well as a camera to aim for the goal. The new motor drivers allowed for more accurate motor speed control. However, some issues with the first LED generation caused the line sensors to malfunction, leading to an inability to control our robot's movements within boundaries.

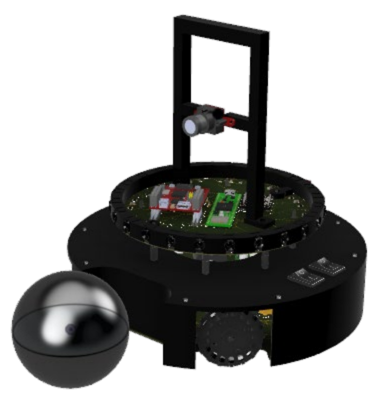
Finally, the game strategy for the Worlds Competition Iteration was much more complex. The angle correction featured a PID controller in order to prevent overturning and subsequently overturning again on the correction turn when the robot was wildly rocked out of control. We also identified five variables that we must always account for during the game. These were: ball angle (relative to the robot's direction), ball distance (relative to the robot), robot direction, robot x (relative to the starting point), robot y (relative to the starting point). With these, we can perform PID again on the values to accurately move behind the ball and direct it to the goal with the help of the mounted camera. The values would work in tandem to the line sensors, which could provide an update to the x and y positions to account for any drift in the accelerometer.

Design Specifics

In this section, the details relating to the choices for the mechanical design, as well as trials and failures are documented. Our GitHub repository[6] (linked in the footnotes) contains all the designs, for public use and for all future teams.

Mechanical Design

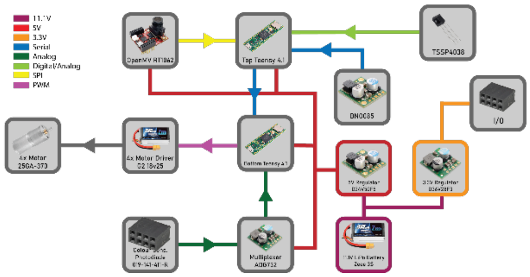
The mechanical aspect of the robot was designed with fusion 360 allowing us to model our robot. The 3D printed pieces consist of a main cover/chassis, handle, motor mount, IR shroud and the motor standoffs.



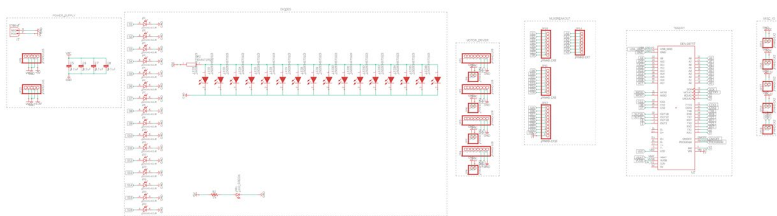
Different mechanical parts of the robot experience varying amounts and types of stress, as well as varying material requirements. We have experimented with different types of filament for the 3D prints such as PLA, PETG and a carbon fiber filament. Initially we planned on using the carbon fiber filament as it has a exceptional strength and performance, but we found out that the prints were poorly done and some were unusable. After testing, all our 3d printed pieces are using the PLA filament because of its reliability and the printer working well with it.

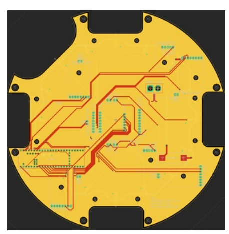
Electrical Design

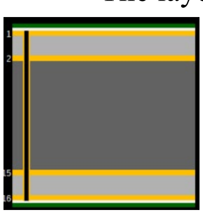
The diagram picture below depicts all the data transferred by all the components and labelled by the communications protocol, as well as the flow of power from the battery. All electrical designs were done on Fusion 360 Electronics, since it provides good integration for the final 3D model.



The schematic for the bottom PCB is shown below. Since we did not go through the trouble of creating custom fusion libraries for the components that we created, like the MuxBreakout board, we ended up just using header pins and accurately measuring the I/O pins of all our own components and the ones we bought commercially.

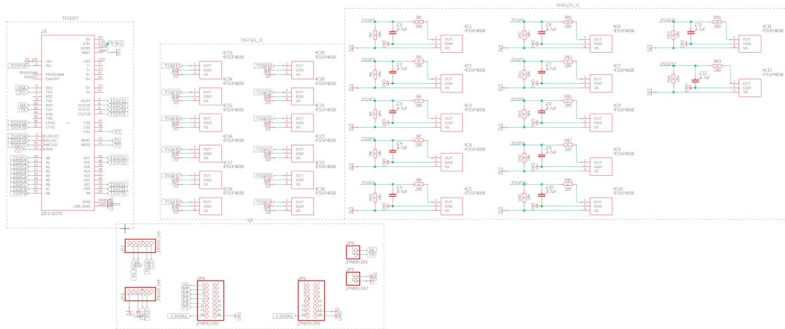




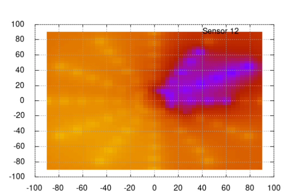


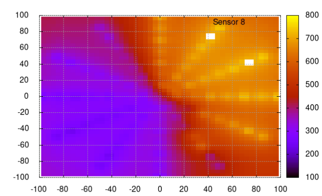
The layer stack for all of our PCBs is shown to the left. We opted for a 4-layer PCB since it had the same production costs as the 2 layer PCB, with the added benefit of a ground plane and a 12V plane, following PCB design conventions. The layer stack we opted for was signal-ground-12v-signal. This allowed for easy access to the ground planes with vias as well as easy access to the 12V source without needing to run them as traces in the signal plane, thereby reducing EM interference between traces.

In the following image, the schematic and the PCB design for the IR Ring is shown. The ANALOG_IC group contains the low-pass filtered IR receivers. The circuit contains a low pass filter (resistor and capacitor) as well as a 10k pull-up resistor.

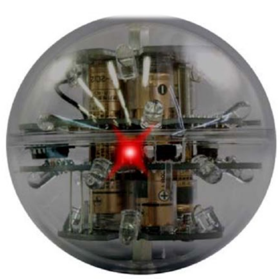


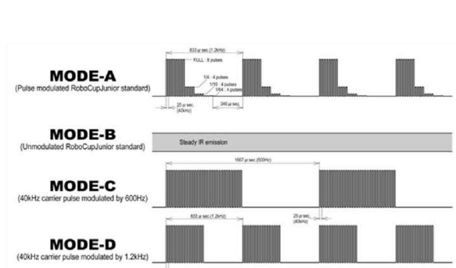
During testing, we discovered that by limiting the field of view with a shroud, we could more effectively isolate the direction of the incoming IR signature. In the following heatmaps, a shroud was applied to one and not in the other. The directional readings were much better after the shroud was applied. Therefore, with this setup, we can accurately determine the direction of the ball down to approximately 7.5-15 degrees and potentially even more accurate as we get closer.



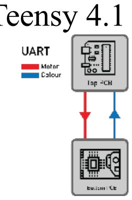


The principle behind the low-pass filters and measuring the length of the pulses lies behind the trapezoid-wave with steps of intensity of the IR LEDs. As seen in the image below (taken from the RCJ forums)[7] Mode A, which contains the trapezoid-wave is used. By capturing where signal strength drops off, we can assign that specific length of the square wave to a predetermined location. This can then be read using pulseIn() or a timer-checker loop and a rough distance can be caalculated. In our experience, however, there does not exist enough steps of the wave for certainty in terms of distance, and a low-pass filter to normalize the signal provides better long-distance detection. We decided to use both in order to achieve the maximum resolution at all distances.



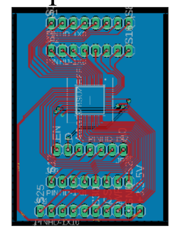


Communication is achieved with both SPI and UART. Due to an issue on the Teensy 4.1 where opening more than two Serial ports would interfere with the operation of the robot, we decided to use SPI for certain aspects of the Robot. UART at the maximum speed of 20,000,000 Baud can operate at the equivalent data transfer rate of 20 mhz.



This is sufficiently fast for us to transfer instructions for our motors from our top Teensy and receive information from the bottom Teensy to compute.

The MuxBreakout, the custom designed component with the ADG732 multiplexer IC includes a mechanism to keep it permanently on. The IC contains WR (Write), EN (Enable) and CS (Chip Select) pins. These are all to be set to logic low according to the truth table from the datasheet provided by the manufacturer while the IC is operating. Therefore, in our component, we connected all of them to the same pin that is to be set to logic low. While these traces cannot be seen in the PCB pictured to the right, they consist of three simple traces designed to keep this IC on at all times, given that we do not need the extra functionality.



We custom mounted most of the SMT components on our PCBs. To do this, we acquired a T962A reflow oven for amateur use. By ordering stencils from the factory, we could use a blank RFID card from the RFID kit provided in Arduino kits and squeegee the solder paste. Since the solder paste was Lead based and was composed of a mixture of 63% Tin and 37% Lead, we worked in our school's advanced chemistry lab, where we had access to fume hoods for proper fume control when using the oven and large quantities of 99.9% IPA (isopropanol alcohol), the manufacturer prescribed method of cleaning solder paste from stencils. We placed all components with tweezers and by hand.

In order to conform to MSL (Moisture Sensitivity Level) requirements, we baked all components which were exposed to room humidity in accordance with the temperature and time described in J-STD-033. Since we were working with MSL 3 components, we baked all of our components at 125°C for 9 hours. Since the thickness was below 1.4mm. We used an oven with precise temperature control and watch glasses to contain the components to be baked.

Software

Behind our robot's program is the consistent development of five values: robot angle, robot x, robot y, ball angle, and ball distance. These values allow the bot to understand the field and determine the best holistic action. In previous years, we had made concrete decisions based on one data collection method. An example would be when our robot detected it was above a line; it would automatically move inward to avoid moving out of bounds. Instead, this year, we have made an algorithm that simply understands it is on the line and restricts movement accordingly. To calculate these values, we are using the Adafruit BNO085 to calculate the robot angle, x, and y, and the IR sensor to find the ball angle and distance from our robot.

These five values are then sent from our top microcontroller to the bottom microcontroller using UART in a six-cycle process. First, 255 is sent to mark the beginning of the cycle, followed by five bytes, each representing each value mapped to a value between 0 and 250. Once received, these values are remapped to their original values to be used to determine the robot's next move, which is determined by a priority list. As to move, our robot needs to understand how it is oriented, our robot adjusts rotation before anything. After this, it takes into consideration the line and restricts movement past it when it is detected. Then the robot moves behind the ball, and only when it is aligned correctly it moves forward to try and score. This priority list generates a desired angle, x, and y position.

From that, three PID controllers, one for calculating y movement, x movement, and rotation, are fed this desired positioning. These controllers find the displacement required to get to the desired position, the derivative of displacement, and the integral of displacement. The output of these functions are associated motor values that add to make up the total motor value.

Each TSSP4038 is associated with a Taylor series function to find the direction and distance of the IR ball. This association is created by measuring the output intensity of each sensor at one-cm increments for two meters. These values are then graphed, and the corresponding Taylor series function is created to fit the curve. Initially, we used an inverse square fit, but as the inverse square law does not fully explain the output, since the silicone also plays a significant role in the shape of the curve, an nth-degree polynomial generated from the Taylor series considers these other factors.

Reflection

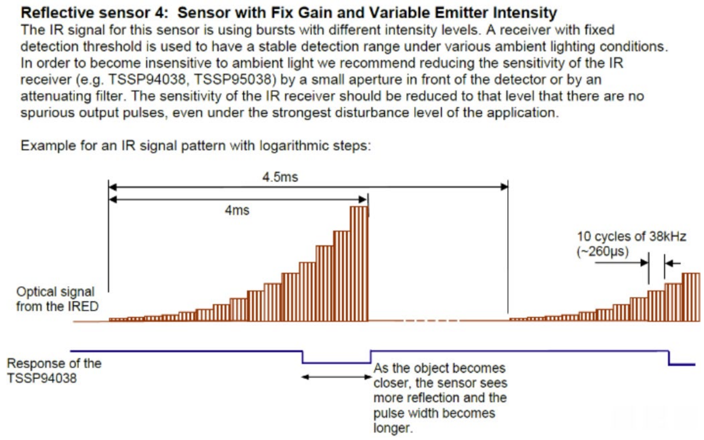
We also explored other aspects of improvement for our robot, namely with the TSSP4P38. This sensor was a derivative of the TSSP4038, but with an AGC that would try and suppress the IR signature. Once it was suppressed, the return of the sensor was the time it takes to suppress the signal as a pwm signal. However, the engineers at Vishay pointed out that the suppression time was approximately 130ms in some cases, which is just too slow for a fast-paced soccer game.

We also noted the importance of being able to detect if we are directly in front of the ball, and noted that in a future design, we can open up a hole in the front kicking section of the robot and allow an IR receiver to check if the ball is in the right position.

Finally, we entertained the possibility of a kicker and a dribbler mechanism. While both in theory would be easy to add on even with less than two weeks until the competition since they can be externally attached modules with wires running to them, we decided against implementing them. This is because of the efficiency of kinetic energy transfer when hitting the ball, and the other two would make little difference unless we are working with special methods of hiding and lobbing the ball. Given our time constraints, we disregarded these and focused on our existing parts and getting them ready for the competition.

One very important lesson we learned was the importance of starting early. Since we started about four to five months before the Eindhoven competition, we were faced with many issues such as procuring motors. Most motors that are commonly used in the robocop competitions like Maxon and Faulhaber motors have a lead time of up to a few months. This would massively surpass our timeframe. Therefore, we opted for fast motors from amazon that are low quality but large in quantity. We could speedily replace all of our motors upon failure. But this does not at all mean that starting early would not be hugely beneficial for our robot quality.

Additionally, after conferring with other teams, we concluded that RoboCup innovation is being stifled by the lack of a better trapezoid waveform from the ball. As seconded by the engineers at Vishay, “you probably do not have enough steps and your timing is too short”.[8] Therefore, more varying intensities could be beneficial, as shown in the diagram below provided in the email from the engineers at Vishay. We could read a pseudo-analog value from our sensors by measuring the width of the pulse as a result, increasing our resolution.



Footnotes and References

[1] https://modernroboticsinc.com/product/ir-locator-360/ – No longer produced
[2] https://www.progressiveautomations.ca/products/lc-82 – No longer used due to low amperage tolerance and tendency to short circuit from minor mis-operation
[3] https://ca.robotshop.com/products/tilt-compensated-magnetic-compass-cmps12 – No longer used in favor of BNO085
[4] https://store.arduino.cc/products/arduino-mega-2560-rev3 – No longer used in favor of Teensy 4.1's Arm Cortex M7 core.

[5] https://www.mouser.ca/ProductDetail/ams-OSRAM/LS-G6SP.02-6D5E-67-G3R3-140-R18-Z?qs=Rp5uXu7WBW8QZPDyhHkSGw%3D%3D – No longer used due to SMD mounting difficulties as well as common short circuits due to the pad positioning

[6] GitHub repository: https://github.com/YuanYoshimi/RCJ-Soccer-Robot

[7] https://junior.forum.robocup.org/t/ir-ball-modulation-change-proposal/3672

[8] Dr. Sebastian Schaefer, Application Engineer, IR-Receiver & IRDA