ECE 302 Robotic and Autonomous System Lab

Final Project

Autonomous Tag Partner

Group 303: Elias Fleishman (ef8956@princeton.edu)
Stanley Kong (sk2562@princeton.edu)

Due: 13 December 2024

Contents

Chapter 0

Introduction

The objective of this project was to build an autonomous car that would be able to play a game of "tag" with a user-operated radio-controlled (R/C) car. The autonomous car fled if the user was "it", attempting to evade being tagged by the R/C car, and gave chase if tagged, attempting to tag the R/C car (with the stipulation that the car that is "it" must tag the rear of the other car). To accomplish this, the PID control loop developed for Speed Control was utilized in addition to two new control loops: one dependent on constant streams of video from Pixy2 cameras, and one dependent on signals from proximity sensors. The Pixy2 camera control loop monitored the position of the R/C car relative to the autonomous car, as well as the distance between the front of the "it" car and the rear of the other car. Depending on this position and distance, the steering angle and target speed were updated to track and detect tagging the R/C car or to evade and detect being tagged by the R/C car. In order to keep the game constrained to a defined area, a visible boundary was added that was detectable by the autonomous car. The proximity sensor control loop was then used to detect the walls and other objects. It was only active while the autonomous car was evading the R/C car and monitored a buffer zone extending approximately 3 ft out from the front, front corners, and sides of the car. If an object was detected in this buffer zone, inputs were made to the steering angle in order to avoid a collision with the object. This report details the subsystems and components utilized to implement these control loops and accomplish this objective. We first discuss hardware, then signal generation and delivery, and finally, we discuss the controlling logic that processes the input signals and generates the output signals necessary to play the game of "tag."

Unless specified otherwise, the hardware and controlling logic for speed control remain unchanged from the Speed Control project. This report covers additions and changes made to our car since the Speed Control project in order to accomplish our objective here. For details on our speed control hardware and logic, see our Speed Control report. A demo video is included in our submitted folder.

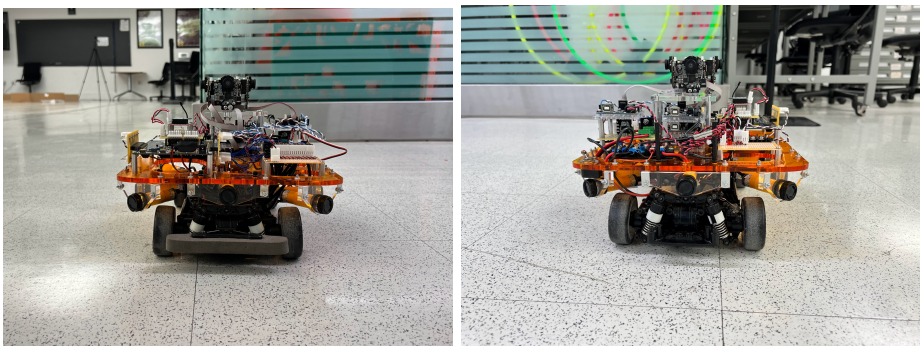

Figure 1: Front and rear view

Figure 2: Left and right side view

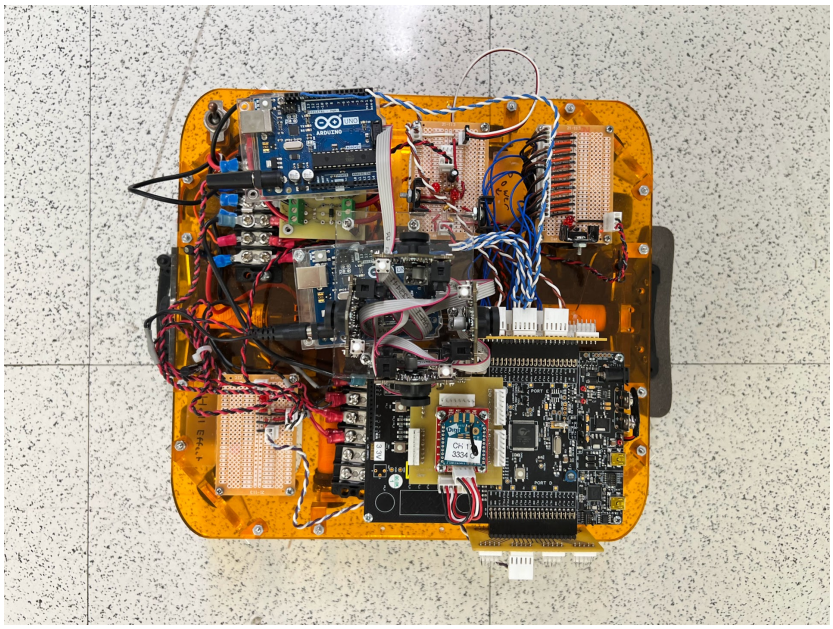

Figure 3: Top view

Chapter 1

Hardware

1.1 Programmable System on a Chip (PSoC) & XBee

The PSoC is a programmable development board that serves as the system's brain, processing input signals and generating output signals. This project utilized input signals from the video and proximity subsystems to generate a Pulse Width Modulation (PWM) output signal to control the vehicle's steering servo. Breakout boards were added to the PSoC's I/O terminals to allow for connections via KK connectors. The XBee is a wireless radio system with both transmitter and receiver modules that allow for wireless, real-time monitoring of the PSoC's execution of its programmed logic. An additional breakout board was added to interface the XBee's transmitter module with the PSoC, while the receiver module was connected to a PC via a USB cable. The PSoC data being received was viewed by utilizing PuTTY to open a serial connection with a baud rate of 115200 to the XBee receiver module.

1.2 Power Regulation

In the circuit shown in Figure 1.1, 9.6V was regulated to both 5V and 6V via 7805 and 7806 voltage regulators, respectively. Decoupling capacitors were placed in parallel with the regulator's input and ground as well as with output and ground to reduce noise and any surges in voltage. A red LED in series with a 330 Ω resistor was placed in parallel with the regulator's output and ground as an indicator that the circuit has power. A single 5V output connector was added to deliver 5V to the video system, while a single 6V output connector was added to deliver power to the steering servo. A 1 μF capacitor was added between the 6V output and ground to further mitigate noise in the power being delivered to the servo. An additional KK connector was added to route the steering servo's controlling signal from the PSOC to the same KK connector that delivers 6V to the servo. The circuit was constructed on a solder-able breadboard with 22 AWG jumper wires and KK connector inputs and outputs.

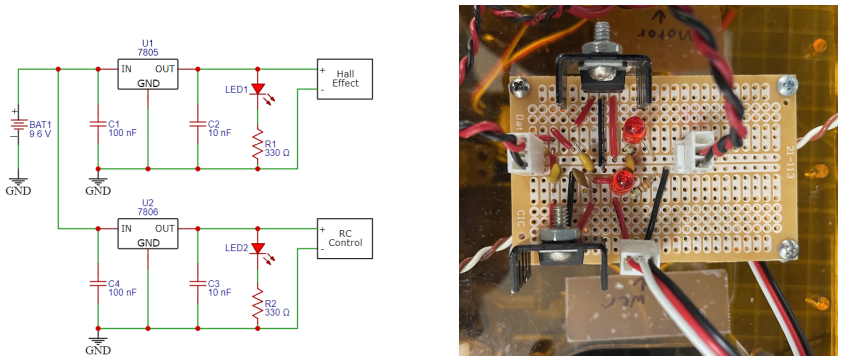

Figure 1.1: Voltage regulator board

1.3 Video

1.3.1 Pixy2 Cameras

The Pixy2 camera was the viewfinder for allowing the autonomous car to Four Pixy2 cameras were placed in the center of the car, with one facing in each direction. The Pixy camera has built-in object recognition based on color. Figure 1.2 shows the Pixy camera view from the PixyMon viewer. The Pixy2 camera was programmed with the red color of the R/C car, a rectangle of which is placed in front of it. The Pixy camera identifies it with high precision, drawing a tight box around it.

Figure 1.2: Pixy camera output

1.3.2 Arduino Boards

Arduino Uno boards were used to process information from the Pixy cameras and send it to the PSoC. Four total Arduino boards were used, one for each camera, and had the same connections. A 5.5 mm outer diameter

barrel jack cable supplied power from the terminal board and a proprietary cable connected the Pixy camera to the Arduino's ICSP pins. A 22 gauge wire connected the Tx pin to the PSoC, twisted with a wire that was connected to ground on the Arduino and opened on the other end. The boards for the front, left, and right cameras were stacked together with standoffs and placed in the middle of the car, while the rear Arduino was positioned above the left terminal board.

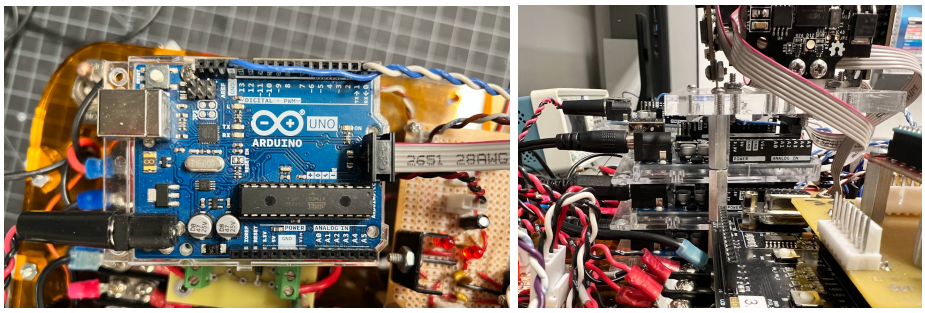

Figure 1.3: Arduino top and side view

1.3.3 Mounting Plate

The mounting plate served to both house three of the four Arduino boards beneath it and fix the four Pixy cameras above it. The board also featured mounting points for an LED panel, which was ultimately not used. A simple design was created in CorelDRAW based on the thickness of the car's acrylic mounting plate and recommendations from course material. The design was cut out of acrylic utilizing a laser cutter. The plate was supported by screwing standoffs onto the power board, voltage regulator board, and the PSoC.

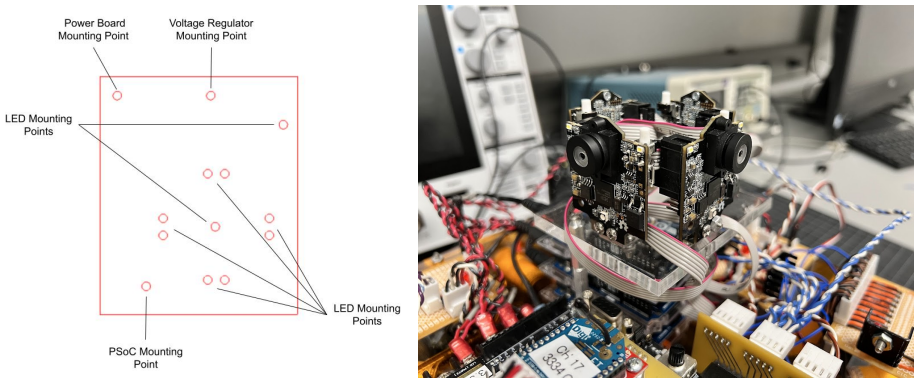

Figure 1.4: Pixy camera mounting plate

1.4 Proximity

Ten infrared proximity sensors were placed around the autonomous car, as can be seen in Figure 1.5, to detect obstacles such as a wall or, in rare

cases, the R/C car. Custom mounts were designed such that the sensors would not interfere with steering. To power the sensors, the same circuit utilized in power regulation to 5V was recreated on a separate solderable breadboard, as can be seen in Figure 1.5. Utilizing the proximity sensor's existing wiring, each of the ten sensors' power and ground was connected to the proximity power board, and the sensors' signal output was connected to the PSOC, all via KK-connectors, as can be seen in Figure 1.5. The sensors work by emitting focused infrared light and measuring the amount of light either reflected or absorbed. The sensors struggled to detect highly reflective objects such as the steel base present in much of the ECE undergraduate lab. Additionally, the focused nature of the sensor's emitted light results in a limited range vertically and horizontally outside of being centered in front of the face of the sensor.

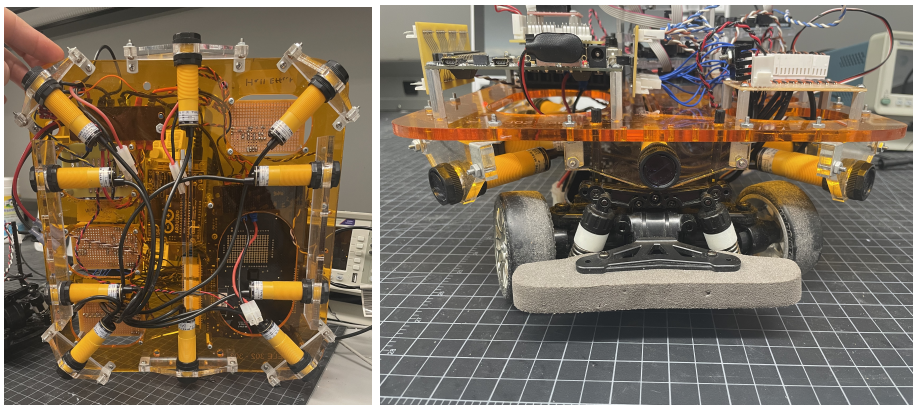

Figure 1.5: Proximity sensor mounting positions

To adjust the sensing distance within which the proximity sensors will activate, a potentiometer within the sensor was manually adjusted via a screw and the activation distance was checked by monitoring the sensor output on an oscilloscope. For the sensors on the front and front corners of the car, the distance was set at approximately 3.5 ft. This setting was determined based on the turning radius of the autonomous car which was approximately 3 ft. The sensors on the side of the car were adjusted to approximately 2.5 ft.

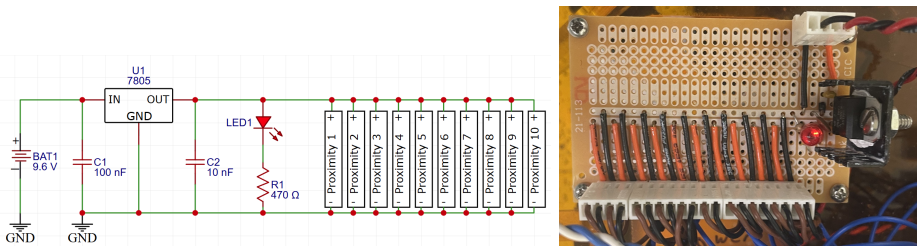

Figure 1.6: Proximity sensor board

1.5 Wiring

As can be seen in Figure 1.7, the autonomous car utilized much of the wiring topology from Speed Control and Navigation, with the addition of a separate terminal block and an additional 9.6V battery to power the four Arduino boards and Pixy2 cameras. To ensure a common ground throughout all subsystems, the ground of the additional terminal block was connected to the ground of the existing terminal block via 14 AWG wire and crimped ring connectors. Connections between subsystems were done so in order to avoid ground loops. The source of power for each subsystem was also the source of ground. However, all signals had their accompanying ground left disconnected in the location where the signal was being received as an input. All connections for the navigation portion of the car were made with 22 AWG twisted pairs and KK connectors.

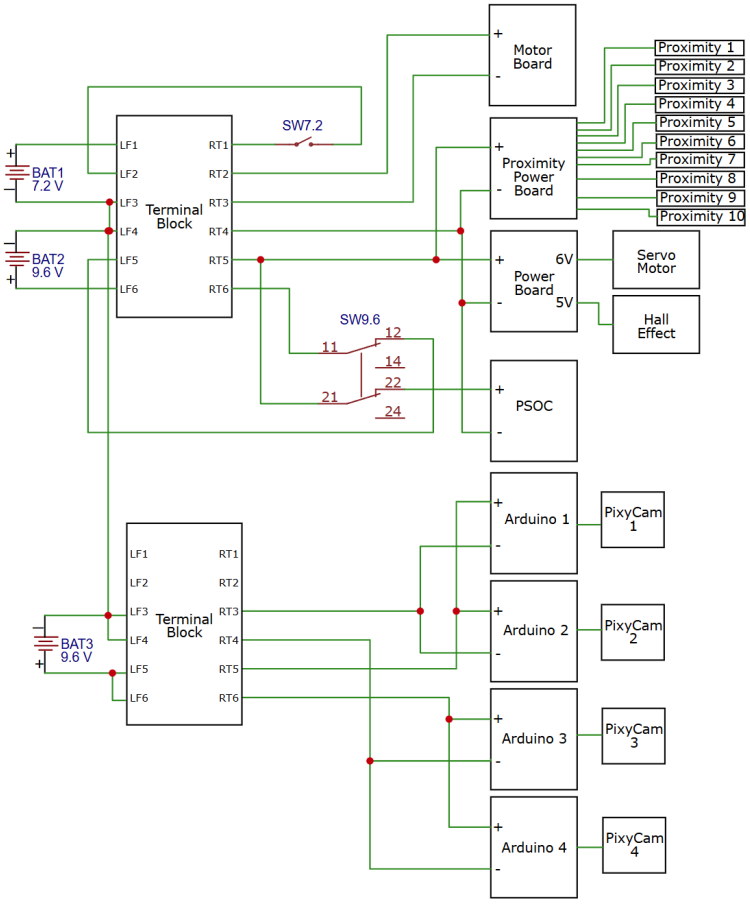

Figure 1.7: Overview of system wiring

1.6 R/C Car

The R/C car utilized for this project was a standard Car Lab car. Red boxes were placed on top of the car's acrylic plate to allow for detection and tracking. Portions of the boxes were blocked off with white paper to ensure that the height of the red box did not exceed the Pixy2 camera frame before the R/C car and autonomous car touched. This enabled distance and tag detection based on the height of the color seen in the video frame. A physical throttle limiter was placed on the controller to limit the R/C car to 4 ft/s, which was the maximum speed utilized in the autonomous car control loops.

Figure 1.8: R/C car

1.7 Barrier

Due to the lack of reliability in the proximity sensor's detection of highly reflective materials, and the limited horizontal detection range, it was necessary to construct a barrier to constrain the autonomous car to a defined area. As can be seen in Figure 1.9, a simple barrier enclosing approximately 210 ft^2 was constructed out of cardboard, and acrylic feet were made with the laser cutter. The cardboard was cut such that its height would be high enough to activate the autonomous car's proximity sensors. The feet were placed on the floor unsecured so that in the event either the autonomous car or R/C car collided with the barrier, it would slide and not result in damage to either the car or barrier.

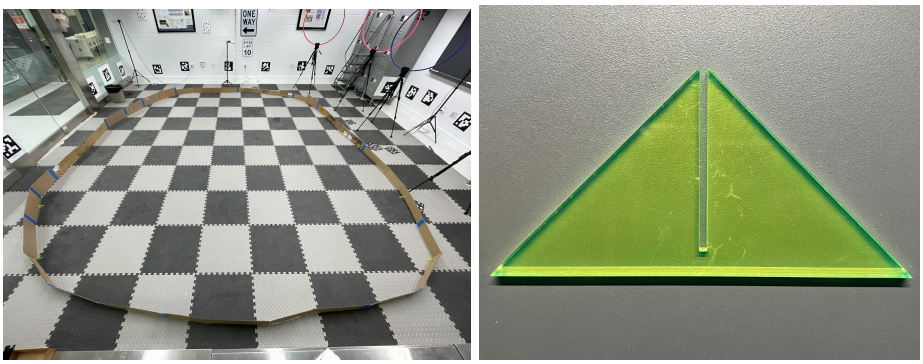

Figure 1.9: Barrier set up ahead of demo; acrylic feet to hold barrier

Chapter 2

Signals

2.1 Video

2.1.1 Pixy2 Video

The Pixy2 has a 60 frames-per-second frame rate and a resolution of 316x208. The Pixy camera itself processes the images and can send specific information to the Arduino. The front camera was instructed to send the height of the object and the middle x-coordinate of the object, or cx. The rear camera was instructed to send only cx, and the side cameras were instructed to send only the existence of objects (1 or 0).

2.1.2 Arduino and PSoC Serial Communication

The Arduino IDE was used to program the Pixy2 cameras to send the signals described above. The Arduino boards then relayed the signals to PSoC through the Universal Asynchronous Receiver-Transmitter (UART) protocol, with a 100 ms delay between each signal. The Arduino and PSoC both used a baud rate of 115200 bits per second. During the debugging phase of this project, UART communication between the PSoC and XBee was also utilized to track whether the car was in flee or chase, what signals were being received from the Arduino, and which cameras were active at which time.

2.2 Proximity

The signals generated by the proximity sensors served as the inputs for the proximity control loop. When nothing is within the set distance of the proximity sensor, the signal remains low at 0V. When something is present within the set distance of the proximity sensor, the signal becomes high at 5V, as can be seen in Figure 2.1. This signal from the various proximity sensors was utilized to determine when the autonomous car needed to take evasive action in order to avoid a head-on collision with either the barrier, or when being chased, the R/C car. The specific proximity sensors that were triggered determined which direction the autonomous car turned, and how long the steering angle was held in that position to make the turn.

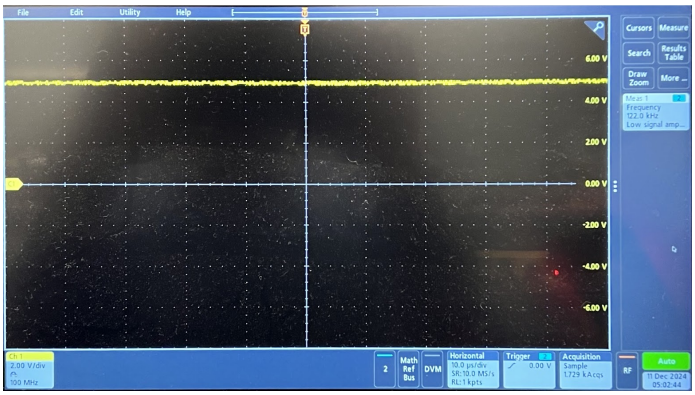

Figure 2.1: Oscilloscope reading from an activated proximity sensor

2.3 Pulse Width Modulation (PWM)

The PWM signal was an output of the PSoC's controlling logic which controlled whether the car's steering servo held the wheels straight, turned them left, or turned them right. The steering servo manipulates the wheels based on the width of the pulse of its input signal. By changing the width of the pulses in the PWM signal, the car's steering could be controlled. A pulse width of 1.5 ms held the wheels straight, a pulse width $<$ 1.5 ms turned the wheels left and a pulse width $>$ than 1.5 ms turned the wheels right. In order to facilitate more straightforward tuning and limit computation, the PWM signal was set to a frequency of 85 kHz which resulted in a period of 3.008 ms.

Chapter 3

Controlling Logic

3.1 PSoC Creator

In addition to the components from speed control, the PSoC's controlling logic consisted of several UARTs, multiplexers, interrupts, digital constants, control registers, and a PWM module. One UART per Arduino and Pixy2 camera allowed for serial communication with the Arduino where the position and distance of the R/C car were extracted from the Pixy2 camera video using functions built into an existing library for the camera. Separate interrupts for each of the front, front left, and front right proximity sensors were set to be triggered as long as the signal from their respective sensor remained high. Since obstacle avoidance was not necessary when the autonomous car was chasing the R/C car, one-bit multiplexers were added between the sensor inputs and their interrupts, with a digital constant of 0 as the multiplexer's other input. A one-bit control register was utilized as the select signal for the multiplexers and corresponded to the current mode of the autonomous car. In chase mode the control register was set to 0, essentially disabling the proximity sensor interrupts, while setting the control register to 1 when evading the R/C car enabled the interrupts. When any one of these three interrupts was triggered, the program was forced to a single interrupt routine where the proximity control loop was implemented. Although interrupts for the rear sensors are present in Figure 3.1, they were not attached to an interrupt routine and were not utilized in any of the control loops. The PWM module was utilized to control the autonomous car's steering servo based on the PWM signal's pulse width. Since a pulse width of 1.5 ms keeps the servo at 0° where the front wheels are straight, the PWM module's clock was set to 85 kHz so that the PWM signal's period was approximately 3 ms. This allowed for minimal calculation of the required duty cycle since a 50% duty cycle would result in a 1.5 ms pulse width. A one-bit multiplexer was added between the speed control's PWM module and output to the car's motor, with a digital constant of 0 at the multiplexer's other input. A one-bit control register was utilized as the multiplexer's select signal which enabled the ability to quickly control power delivery to the car's motor without having to adjust the speed control's PWM signal.

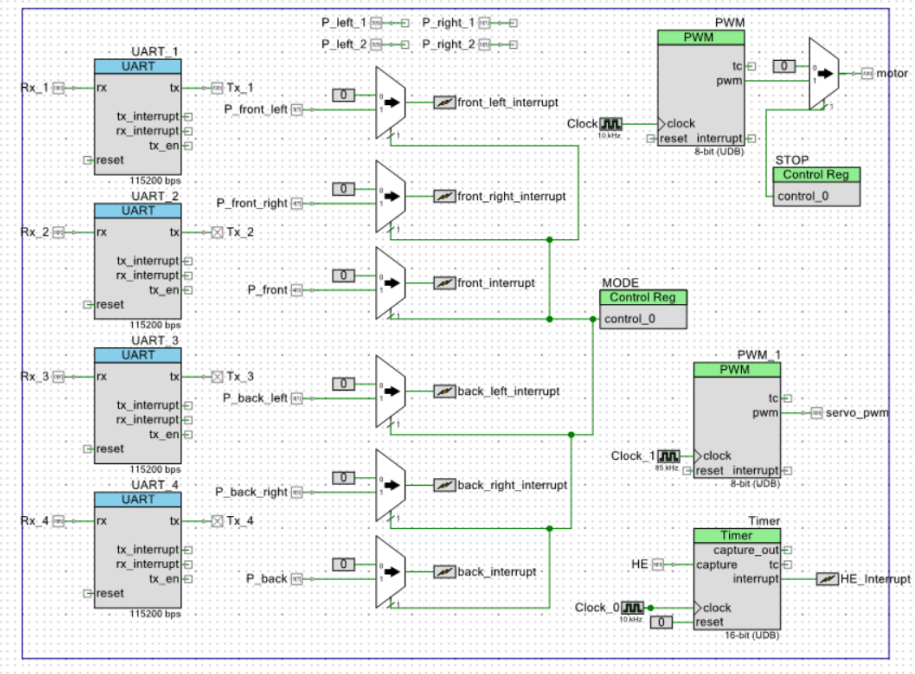

Figure 3.1: PSoC control logic

3.2 Pixy2 Camera Control Loop

The autonomous car uses two control loops for the camera, one for chase and one for flee. The car was programmed to start in chase mode. In chase mode, the front camera is first checked to see if it has detected the R/C car. If it has, the cx value is translated to the servo PWM so that the servo steering angle is pointed in the direction of the R/C car. If the height of the R/C car is less than or equal to 80, it is deemed far away and the autonomous car speed is set to 3 ft/s. The distance between cars at the height of 80 is about 13 inches, as shown in Figure 3.2. If the height is between 80 and 156, the autonomous car's speed is increased to 4 ft/s to help it tag the R/C car. Then if the height is greater or equal to 156, the autonomous car registers a tag and begins fleeing from the R/C car.

Figure 3.2: Distance that activates higher speed and evasive mode

If the R/C car is not seen by the front camera in chase mode, the other three cameras are checked. If one of the side cameras detects the R/C car, the autonomous car will turn in that direction with the highest servo angle until the R/C car returns to the front camera frame. If the rear camera detects the R/C car, the autonomous car will randomly turn left or right until the front camera detects the R/C car. Then the car continues with the front camera control from above.

Once the car enters flee mode, the rear camera is prioritized. Like the chase mode, if the height of the R/C car is less than or equal to 80, it is deemed far away and the autonomous car speed is set to 3 ft/s. If the height is between 80 and 165, the autonomous car's speed is increased to 4 ft/s to help it flee from the R/C car. Additionally, the car activates evasive mode, in which it continuously performs randomized turns for a random time between one to three seconds. Then if the height is greater or equal to 165, the autonomous car registers that it has been tagged and enters chase mode. For the demo, the car was programmed to stop by toggling the multiplexer for the speed motor to select 0. This way, the demo showcases one round of the autonomous car in chase mode and one round of flee mode.

3.3 Proximity Control Loop

The proximity control loop was responsible for controlling the car's steering in the event the possibility of a collision with either an object or the R/C car was detected by the autonomous car's proximity sensors. If any of the front, front left, or front right interrupts were triggered, the status of the proximity sensors on the right and left of the car was evaluated. If only the front left or front right sensor was triggered, the servo PWM signal was adjusted so that the car would turn in the opposite direction of the detected obstacle. If the front sensor and any of the sensors on one side of the car were triggered the servo PWM signal was adjusted so that the car would turn in the opposite direction. If only the front sensor was triggered, the PWM signal was adjusted so that the car would turn left. In all of these cases, the adjustment to the PWM signal was maintained until no obstacles were sensed by either the front, front left, or front right proximity sensors. This control loop was only active when the autonomous car was in chase mode. However, when the control loop was active, it took priority over any other steering input processes. As a result, the portion of the Pixy2 Camera control loop that detected a tag was added to run in the same interrupt routine as the proximity control loop.

Chapter 4

Discussion

Prior to our project demonstration, the R/C car failed to turn left consistently. We are unsure if the servo or the controller was to blame; however, this particular car had its servo motor replaced before being given to us. This, combined with our evasive maneuver algorithm, made it quite difficult to tag the autonomous car. We have included a demonstration video in our submission folder. The autonomous car begins in chase mode, quickly tags the R/C car, and then begins avoiding the barrier as well as being tagged by the R/C car. During our project demonstration to the class, we were able to tag the autonomous car with the R/C car. However, we were unable to recreate it in the demo video included in the submission due to the issues with the R/C car steering. It is worth reiterating as well that a tag only registers if the autonomous car is tagged from the rear.

4.1 Wanted to Accomplish vs Did Accomplish

Compared to our initial proposal for this project, we accomplished the vast majority of what we had initially intended to. Due to a few challenges, however, we were unable to complete everything outlined in our initial proposal and had to make a few adjustments to how we accomplished certain tasks. Firstly, we intended to analyze the proximity sensor signals and the Pixy2 camera data in order to detect a tag. However, the fact that the proximity sensors' threshold would not adjust to a short enough distance (despite a published minimum of 0 cm), combined with the requirement of manual adjustment of the threshold, meant that the sensors would be better suited for obstacle detection rather than tag detection. We instead relied solely on the Pixy2 camera to detect a tag. Secondly, we had initially intended for each car to have an LED panel to indicate which car was "it", with an RF transmitter-receiver pair sending a signal from the autonomous car to the R/C car to indicate what should be displayed on the LED panel. We were unable to get the RF transmitter-receiver pair to work without large amounts of noise, so we attempted only to include the LED panel on the autonomous car. When the LED panel was connected to an Arduino by itself, it functioned as intended. However, the LED panel driver's I2C connection with the Arduino interfered with serial communication between the Arduino and PSoC. Because this im-

pacted the functionality of the autonomous car, we elected to omit the LED panel from our final demo.

4.2 Improvements

Our thought process behind distributing signal analysis among the PSoC and Arduino boards was to extract the relevant data from the Pixy2 camera video using simple code in the Arduino boards and then perform the more complex signal analysis using the PSoC. This would result in minimal hardware connections and limit most of the code debugging to one location, the PSoC. However, this approach required the use of four UARTs (one for each Arduino) and resulted in a less robust design that relied on slower-to-execute software, rather than faster-executing hardware. We were able to get our software-heavy design to work, but in the future, we would try a design that conducted the Pixy2 camera data analysis in the Arduino boards with outputs of the steering servo PWM signal and binary high/low signals to indicate whether or not a tag was detected and when the R/C car was within a specific threshold. Although this design would require more hardware connections, we think it would be more robust and faster to execute.

The proximity sensors we utilized to detect and avoid obstacles were reliable and exhibited quick response times, however, their lack of adjustability is something that could be improved. In the future, would would experiment with ultrasonic distance sensors that would output the distance from an object rather than just whether or not an object is within a set distance. This would enable the detection of obstacles and a tag with the same sensor, reducing reliance on the Pixy2 camera data to detect a tag, which was not always reliable.