Autonomous Driving for the Texas Instruments Cup

Mohammed Fareed
Kate Gleason College of Engineering
Department of Computer Engineering
Rochester, NY
mff9108@rit.edu

Trent Wesley
Kate Gleason College of Engineering
Department of Computer Engineering
Rochester, NY
taw8452@rit.edu

***Abstract*—Autonomous driving is progressing and becoming more prevalent in society as time goes by. A robust autonomous driving system offers the potential for a future where driving safety and efficiency are dramatically increased. The Rochester Institute of Technology Texas Instruments Car Cup required the application of various autonomous driving techniques and algorithms to race. The objective of this project was to program a miniature battery powered car to autonomously race around a track. The car was controlled by an MSP432 microcontroller board used the input from a linescan camera to control its servo and motors for steering and speed control. This paper documents the theory, code development, and the challenges to race in the fall 2023 Texas Instruments Cup.**

I. Introduction

Every semester, Rochester Institute of Technology (RIT) hosts a Texas Instruments (TI) autonomous car race. This challenges computer engineering students to use their skills and knowledge regarding microcontrollers, motor control, control systems, and problem solving. Given that there is an average of 6 million car accidents in the US every year, education and experience with autonomous driving can help shape a world where driving is much safer [1].

The cars utilized had various components with different functions. The cars included two motors (one for each back wheel), a servo for steering, and MSP432 microcontroller board for control, a line-scan camera, and OLED display, and a 7.2V battery pack. A picture of the car used is shown in Figure 1.

Fig. 1. Car.

The racetrack developed for the TI Cup tested various aspects of autonomous racing. It combined performance on straight segments, turns, intersections, hills, and wobbles. To perform well on the track, a car would need to be able to handle all of these conditions. A picture of the track is shown in Figure 2.

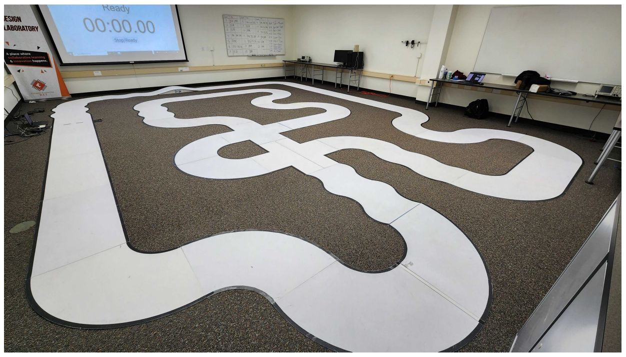
Fig. 2. Racetrack for TI Cup.

The rest of this paper is organized as follows. Section 2 discusses background information regarding the TI Cup, some general theory applied, some of the MSP432 modules used, and some of the hardware used. Section 3 covers the proposed method and specific implementations for the project. Section 4 presents the results of the race and performance of the car. Section 5 gives a conclusion of the project. Section 6 gives the authors' acknowledgements. Section 7 lists the references used for this paper.

II. Background

A. RIT TI Car Cup

The RIT TI Car Cup is hosted every fall and spring semester. Each team has three attempts to complete one lap on the track as fast as possible. The first time a lap is completed is the final time. If a team doesn't finish in three attempts, they are disqualified. For an lap to be considered successful, the car must finish in under 60 seconds and have at least two wheels on the track at all times. Race results were recorded by a laser timer for accuracy.

B. Materials

The car has many components and generates a price from them. Reference [2] was observed to create the bill of materials shown in Table I.

TABLE I
BILL OF MATERIALS

Part	Qty	Cost (USD)
Parallax TSL-1401 Line Scan Camera	1	$80.00
Servo Steering Arms	1	$17.99
Motor Driver - RB-WAV-77	1	$28.9
Car Chassis Kit - ROB0170	1	$98.75
Brushed DC Motor Kit - KIT0167	1	$25.00
UCTRONICS Module 12864 SSD1306 OLED	1	$6.99
Bluetooth Module HM-10	1	$10.99
Tenergy 7.2V High Capacity 6-Cell Battery Pack	1	$39.99
Sourcingpower Universal RC Battery Charger	1	$19.99
Fielect 5Pcs F-F 6Pin Jumper Wire Ribbon Cable	1	$6.69
5pcs Tamiya Male Power Connector Cable	1	$8.68
Zip Ties	1	$18.99
Total		**$363.05**

The bill of materials gives information about parts, quantity, and cost in USD. As shown in Table I, a total of approximately $363.05 would need to be spent to construct a similar car.

C. Camera

The autonomous vehicle utilized in the TI Cup features the Parallax TSL-1401 line-scan camera, a major component of the car's navigation system. This camera is designed to capture a one-dimensional array of light intensity across its field of view, using 128 photo-diodes arranged linearly. Each photo-diode corresponds to a pixel in the captured image, enabling the camera to detect variations in light intensity along a line.

The TSL-1401's capability to differentiate between light and dark areas is central to the car's ability to follow the track. In the context of the racetrack, which features distinct contrasts between the track (light) and the surroundings (dark), the camera provides real-time data for the microcontroller to process and determine the car's position relative to the track boundaries. This camera operates by scanning the track surface and generating an output signal that varies depending on the reflected light intensity. Brighter surfaces, like the white parts of the track, result in higher signal values, whereas darker areas, such as off-track surfaces or track borders, produce lower signal values. This difference in signal strength is what enables the autonomous vehicle to detect and stay within the track boundaries. Figure 3 shows a wiring diagram for how the camera was connected to the MSP432.

The diagram shows that camera being powered by 3.3V and ground. The camera's SI, CLK, and AO pins are connected to P5.5, P4.7, and P5.4, respectively. The camera's AO pin is connected to the MSP432's ADC pin. The analog output from the camera is converted to a digital signal using the car's Analog to Digital Converter (ADC), allowing for precise digital processing and decision-making.

For optimal performance, the camera's placement angle and focus were carefully calibrated. The angle ensures that the camera has a clear and unobstructed view of the track ahead, while the focus is adjusted to maximize clarity and contrast in the captured image. This calibration was crucial in ensuring that the camera could reliably detect the contrast between the

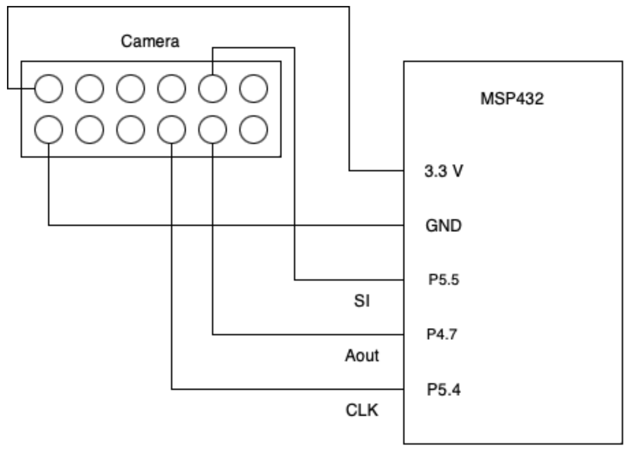

Fig. 3. Camera Wiring Diagram.

track and its surroundings under various lighting conditions encountered during the race.

D. Motors

The autonomous vehicle in the TI Cup was equipped with two key types of motors: Brushed DC Motor Kits (KIT0167) for propulsion and a servo motor for steering.

Brushed DC Motors for Propulsion: The rear wheels of the car were powered by two independently controlled Brushed DC Motors. Speed control was implemented through Pulse Width Modulation (PWM) signals, allowing for precise adjustments in motor speed in response to the track's demands.

Servo Motor for Steering: The car's steering mechanism was controlled by a servo motor. The servo's role was to precisely adjust the angle of the front wheels, enabling the car to follow the intended path on the racetrack. Implementing effective steering control was one of the project's significant challenges, requiring meticulous tuning to achieve optimal responsiveness and accuracy.

Figure 4 shows a wiring diagram for how the motors were connected to the MSP432.

The diagram shows that the board had the motors interfaced using the RB-WAV-77 motor driver and the servo directly connected to the MSP432. motors were powered by 7.2V and ground, and their PWM pins were connected to P2.4-2.7, where each motor was assigned to a pair of pins. The direction of the DC motor was controlled by alternating the PWM signal between the two pins of the motor. The servo's PWM pin was connected to P5.6 and was powered by the on-board 5V and ground. Its angle was controlled by varying the PWM signal between 0.05 and 0.1 duty cycle.

E. PID theory

Proportional-Integral-Derivative (PID) control is a widely used feedback loop mechanism in control systems, including autonomous vehicles. This control strategy is crucial in maintaining a desired system behavior, such as steering and speed control in the context of autonomous racing. PID control is implemented by calculating an error value, which is the

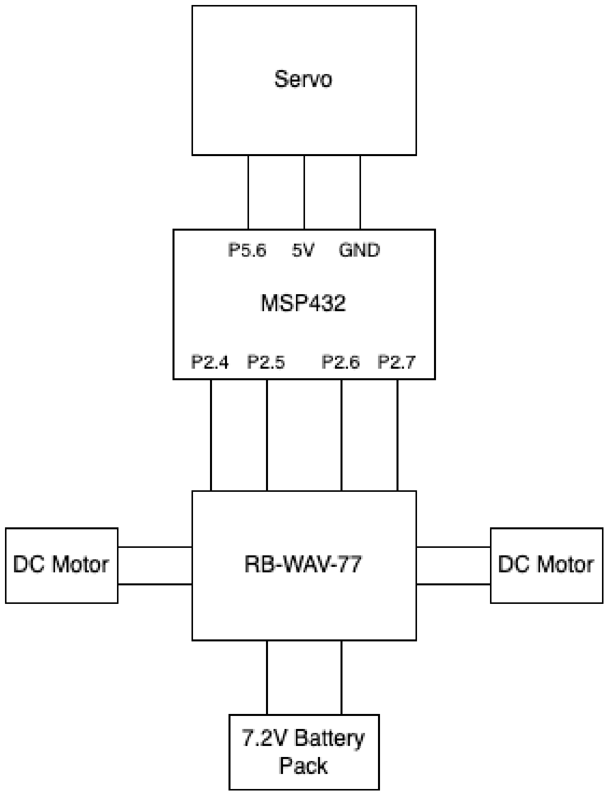

Fig. 4. Motor Wiring Diagram.

difference between the desired and actual system states. This error value is then used to adjust the system's behavior to minimize the error and achieve the desired state. The three components of PID control are as follows:

- A PID controller continuously calculates an error value as the difference between a desired set point and a measured process variable. It then applies a correction based on proportional, integral, and derivative terms, denoted as P, I, and D, respectively.
- The **Proportional** term produces an output value proportional to the current error. In the autonomous vehicle, this helps to steer the car more aggressively when it deviates further from the track centerline.
- The **Integral** term focuses on the accumulation of past errors. It seeks to eliminate residual steady-state errors by integrating the error over time.
- The **Derivative** term predicts system behavior and thus can prevent the system from overshooting the setpoint. By reacting to the rate of change of the error, it provides a damping effect.

Figure 5 shows a block diagram of a PID controller, illustrating how the feedback loop is implemented in the context of the car's steering and speed control systems.

The figure depicts the structured framework of the PID controller as implemented in our autonomous vehicle. The block diagram showcases three distinct paths representing the Proportional, Integral, and Derivative terms, each path

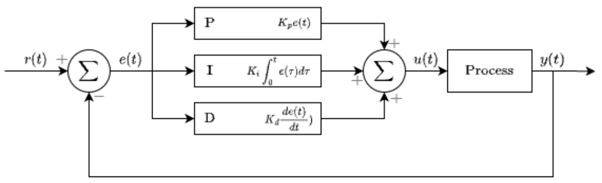

Fig. 5. PID Controller Block Diagram.

applying a mathematical operation on the error signal $e(t)$. The Proportional path amplifies the error by a constant K_p, providing immediate correction proportional to the error. The Integral path integrates the error over time, scaled by K_p, to eliminate residual steady-state errors. The Derivative path, scaled by k_d, takes the derivative of the error, offering a predictive correction and dampening the system response to prevent overshoot.

These three signals converge at a summing junction, producing a combined output that dictates the control action to the process, which is the car's steering and throttle mechanism. This integration of responses ensures that the vehicle responds accurately to deviations from the desired trajectory, maintaining precise control throughout the race.

Tuning of the PID parameters (K_p, K_i, K_d) was a critical process in the solution design, involving iterative testing to achieve a balance between responsiveness and stability. The PID parameters were tuned to ensure that the car could follow the track's centerline while maintaining a reasonable speed. The PID controller was implemented in the car's code to control the steering servo, enabling the car to follow the track's centerline.

F. Timers and Interrupts

Effective timing and event management are critical in the control systems of autonomous vehicles. In our project, we utilized a suite of timers and interrupts on the MSP432 microcontroller to coordinate the activities of the motors, servo, and camera.

Timer A0 for Motor PWM: Timer A0 was dedicated to controlling the Pulse Width Modulation (PWM) for the Brushed DC Motors. By adjusting the timer period, the duty cycle of the PWM signal can be fine tuned, thus controlling the speed of the motors. This allowed for dynamic speed adjustments based on the car's immediate requirements for speed and precision on the track.

Timer A2 for Servo PWM: Similarly, Timer A2 was configured to manage the PWM for the servo motor responsible for steering. The precise timing of this timer was crucial for achieving smooth and responsive steering behavior, enabling the car to navigate the twists and turns of the racetrack with agility.

Timer 32 and SysTick Timer for Camera Control: The operation of the line-scan camera was managed by two timers. Timer 32 controlled the Start Integration (SI) signal, initiating the camera's capture sequence. In parallel, the SysTick Timer generated the clock (CLK) signal, ensuring that the camera's photo-diodes were sampled at consistent intervals for accurate light intensity readings.

Interrupts: Interrupts were used to manage the camera's output signal. The camera's analog output was connected to the MSP432's ADC pin, which was configured to trigger an interrupt when a new value was available. This interrupt sets a global boolean variable that is monitored by the main loop. The main loop would then read the camera's output and process the data to determine the car's position relative to the track boundaries.

G. Analog to Digital Converter

The Analog to Digital Converter (ADC) is a crucial component of the microcontroller that bridges the analog world with the digital system. In the autonomous vehicle, the ADC's primary role was to convert the analog signals from the Parallax TSL-1401 line-scan camera into digital values for processing and decision-making.

Functionality of the ADC:

- The TSL-1401 camera captures the track's image as an array of analog light intensities. The ADC on the MSP432 microcontroller translates these intensities into a series of digital values, which can range from 0 to 2^14-1 (16383), representing a 14-bit resolution.
- This high-resolution conversion is needed for distinguishing between the varying shades of gray that correspond to the track and its boundaries, allowing for finer control over the vehicle's steering and throttle based on the perceived path. Even slight variations in light intensity readings can lead to significant changes in the vehicle's trajectory. Thus, the accuracy of the ADC's conversions is vital.

Integration with Control System:

- Upon the ADC receiving analog values, the microcontroller's firmware is configured to initiate the ADC conversion process, which involves sampling the signal and converting it into a digital format.
- The converted digital data is then fed into the PID control algorithms, informing the subsequent adjustments to the vehicle's motors and steering servo.
- To ensure data accuracy and consistency, the ADC conversion process is synchronized with the camera's SI and CLK signals.

III. Proposed Method

A. Camera Vision and Filtering

The linescan camera's output is an array of 128 integers representing the brightness it sees. The linescan camera used had a maximum value of 2^{14}-1 (16383), and would tend to saturate at that value pretty easily. This phenomenon was utilized to filter out the carpet from what was the track. If an array element was less than 16381, it wouldn't be considered as part of the track and would be treated as 0. Otherwise, the element would be treated as a 1. This creates a binary representation of track versus carpet. Figure 6 shows a plot characterizing the typical behavior of the linescan camera.

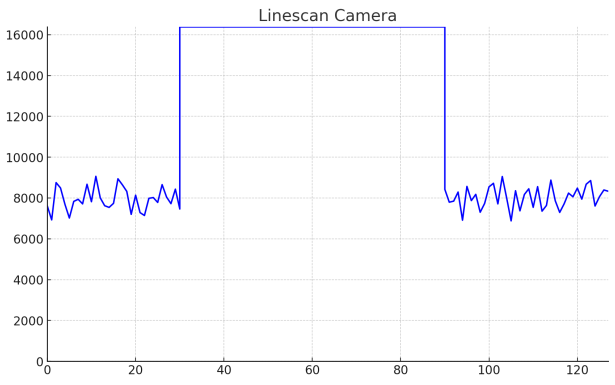

Fig. 6. Unfiltered Linescan Output.

In Figure 6, the saturated component is where the linescan camera sees the bright white track along the 128 elements of the array. The rest is the where the camera sees the darker carpet beside the track. As stated, this output was filtered into a binary representation of where there is and isn't track. This output is modeled by Figure 7.

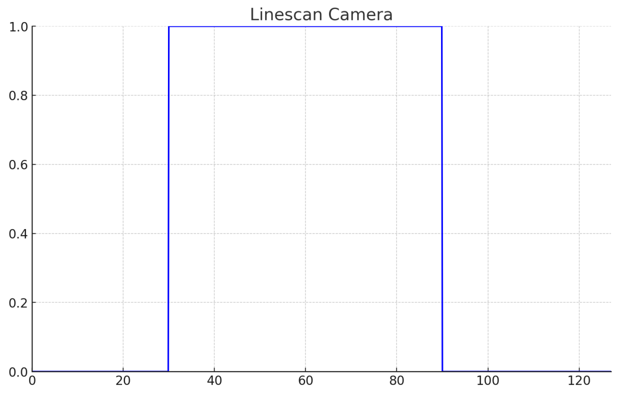

Fig. 7. Filtered Linescan Output.

This filtered output is simpler and unbiased by the values unassociated with the track. Using the binary output, a weighted average was calculated to find where the center of the track is. The calculation is shown in Equation 1.

$$\text{midpoint} = \frac{\sum_{i=0}^{127} i * x_i}{\sum_{i=0}^{127} x_i} \tag{1}$$

In Equation 1, i represents the index in the array and x_i represents the binary value at the index in the array. The resulting midpoint can the be used to determine the position of the middle of the track and steer towards it.

B. Carpet Detection

Carpet detection is important for preventing damage from hitting a wall when the car would drive off the track. The program logic created to detect whether the car has left the track is very simple. Every element of the raw output array from the linescan camera was added up. This sum was compared to a brightness threshold calibrated through trial and error. If the sum was less that the threshold, the car would stop the DC motors and therefore stop the car from moving.

C. PID Implementation

PID was utilized for steering with a servo to be able to handle turns and reduce oscillations on straight paths. Error is an essential variable for PID to function. For this project, error was the calculated as shown in Equation 2.

$$\text{error} = \text{midpoint} - 64.5 \tag{2}$$

On a scale from 0 to 127 (indices of linescan camera output), the midpoint would normally be 63.5. In this case, it was 64.5 to adjust for the offset of the camera. This calculated error was then used for PID control. The overall equation for to control the servo with PID is shown in Equation 3.

$$ServoPWM = K_p{*}\text{error1}+K_i{*}\frac{\text{error} + \text{error2} + \text{error3}}{3}+K_d{*}(\text{error1}-2{*}\text{error2}+\text{error3}) \tag{3}$$

In this Equation 3, error1 refers to the most recent error, error2 refers to the error before error1, and error3 refers to the error before error2. This error history allows the PID algorithm to respond in a more optimal way than if it only had access to the the most recent error.

This calculation alone is not sufficient since a servo PWM outside of the range 0.05 to 0.1 could break the car. For this reason, code was added which would set the servo's PWM to 0.05 if the calculated value was less than 0.05. Additionally, the code would set the servo's PWM to 0.1 if the calculated value was greater than 0.1.

D. Variable Speed

Variable speed was implemented so that the car would slow down when turning and speed up when going straight. Specifically, this was implemented with an equation relating the absolute value of servo's PWM input (0.05 to 0.1 duty cycle) minus its middle position (0.075 duty cycle). The more the servo position varied from the center, the slower the car was programmed to go. This is shown in Equation 4.

$$\text{MotorPWM} = -18 * |\text{ServoPWM} - 0.075| + 0.434 \tag{4}$$

If the motor PWM calculated is less than 0.36, it was set to 0.36 by the code to ensure it was maintaining a decently fast speed.

IV. Results

A. Race Results

Acknowledgments

References

[1] "Car Accident Statistics in the U.S. — Driver Knowledge," DriverKnowledge, 2019. https://www.driverknowledge.com/car-accident-statistics/

[2] K. Jacowleff and M. Teichman, "Texas Instruments Autonomous Car Cup Project Fall 2022," 2022.