Mini-Project Final Technical Memo

Joshua Miller, Joshua Suber
IED Section: 01
Date: 06/13/2024

Introduction

Learning environments prosper when hands-on demonstrations reinforce theories. To this end, our team has designed a cost-effective autonomous line-following teaching tool. We utilized a pre-existing motor and chassis kit and commercially available chips, components, and sensors to meet our goals. Additionally, we implemented PID control, a vital robotics concept, for teachers to demonstrate. These considerations allow teachers to demonstrate PID control at a reasonable price and with little setup.

Concepts and Benchmarking

Logic Implementation

Firstly, we explored our options for processing sensor input. Specifically, we debated using a microcontroller (e.g., Arduino Uno) or analog/digital components (e.g., OP-Amps, logic gates, transistors, etc.).

Microcontrollers
Microcontrollers are widely available and tend to be very cost-effective. Additionally, they offer a layer of abstraction that simplifies the circuit into easier-to-read software.

We used the table below to determine the best microcontroller for our task and to compare it against analog/digital components.

Table 1: Microcontroller Options

Device	**Cost**	**PWM/analog output**	**Analog input**	**Supplyable Current**	**Complexity** *(Simple 1 – 10 Complex)*	**Widely Available**	**Have on-hand**
Arduino Uno	$15	Yes	Yes	~400-900mA	1	Yes	Yes
Raspberry PI 3	$35	Yes	Yes	1.5 A	8	Yes	No
Raspberry PI Pico	$4	Yes	Yes	~300mA	3	Yes	No

Analog/Digital Components
We considered many chips and components to base our logic processing; the table below lists the viable options. Each option must be cost-effective and easily obtained as we utilize multiple of each. Therefore, relatively expensive options are not listed.

Table 2: Analog/Digital Components Options

Component	Analog input	Complexity (Simple 1 – 10 Complex)	Have on-hand
Op Amps	Yes	6	Yes
74XX Series	No	3	Yes
555 Timer	Yes	8	Yes
Transistor Logic	No	7	No

Sensors

There are multiple ways to detect a line; therefore, we considered multiple types of sensors. One method considered was a camera facing towards the line. The camera's picture would be processed to determine where the line is, and the car would plan a route following the line. Next, we explored light-reflection sensors. The idea, pictured in Figure 1, is to shine a light source onto the ground. Then, the amount of light reflected should determine whether the car is on the line.

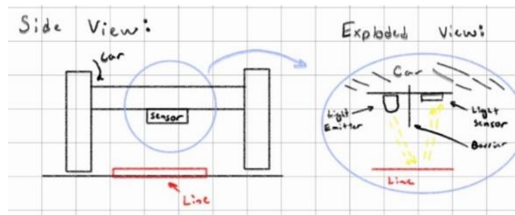

Figure 1 Diagram of light-reflection sensors

Control Mechanism

Our research found two control mechanisms: the bumper and PID methods. Figure 2 shows the regular operation of both methods. The bumper method takes digital inputs and turns the car based on which sensor detects the line. If the left sensor detects the line, move right and vice versa. Meanwhile, PID determines its location on the line via the difference in the sensor values and utilizes constants to react based on the current location.

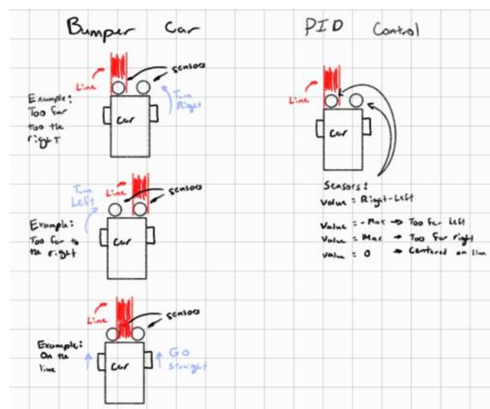

Figure 2 Bumper vs PID control

1. Bumper Control

This implementation "wiggles" around the line (like bowling bumpers) because turning only occurs when the sensors detect the line. Despite being easy to implement, this makes the car temperamental and susceptible to out-of-control oscillations.

2. PID Control

This implementation lets us control motor speed based on the difference between analog input and an expected value. This approach controls the speed of motors proportionally to the distance away from the line (e.g., the farther from the line, the greater the car turns to return to the line) and integrally adjusts based on how long the car has been away from the line.

Table 3: Control Method Pros and Cons

Control Method	Pros	Cons
Bumper	+ Simple to implement + Simple to teach	- Susceptible to oscillations - Unreliable - Boolean Input (on or off the line)
PID	+ More Reliable + Adjusts according to analog distance from the line + Interesting topic to teach + Applicable to robotics + Demonstrates how to adjust PID K values	- Complicated to adjust PID K values - Harder to implement

Motor Control

Our group explored multiple options to control the motors. We quickly realized that we needed a separate component powerful enough (able to deliver higher currents) to control the motors. We selected between two methods: relays and H-Bridge drivers.

1. Relays

Relays are mechanical switches that allow high-supply currents to pass. They are straightforward to implement but not suitable for our application. PWM, which we will use to control the motor speed, rapidly switches on and off. Relays cannot switch fast enough and fail when switched rapidly.

2. H-Bridge

H-Bridge drivers are more complicated to implement but are designed to be used with PWM signals.

Solution

Logic Implementation

Our group ultimately chose to use a microcontroller for our project. We determined that the quantity and complexity of digital/analog logic required would be too difficult for a teacher to explain to a class adequately. On the other hand, teachers could skim over the software, pointing out the most critical aspects to their students.

For multiple reasons, the microcontroller we selected was an Arduino Uno. The most important factors were the current the Arduino could supply (to the motors) and that it is so widely available that we had one on hand.

Sensors

Our team chose to implement light-reflection sensors over camera-based solutions as we promptly discovered that the price and complexity of a camera-based system would significantly exceed our project constraints.

When choosing a light wavelength and positioning of sensors in reference to each other, we decided to utilize a pre-existing PCB with two ultraviolet lights and sensors.

Control Mechanism

Based on the advantages and disadvantages listed in the Concepts and Benchmarking section, we determined that PID control was the best choice for our application. The most important consideration was reliability, and the bumper's oscillations would not be reliable enough for the customer. Additionally, we determined that PID control would be a more informative topic to teach than the bumper method (which could still be taught before teaching PID).

Our specific implementation of PID control only implements proportional and integral control. We decided to exclude derivative control as that would add another constant to adjust and greater complexity with limited benefits.

Additionally, the integral is implemented by averaging a set number of previous values, which is a close approximation of integration with less complexity to implement and describe to a classroom.

Motor Control

Our group uses the L293NE chip, an H-Bridge driver, to control the motors. We also utilize PWM to control the motors' speed. The L293NE chip was chosen because it is cost-effective, reliable, widely available, and available for us.

In our implementation, we used two of the four half-h-bridge drivers, one for each motor. A half-H-Bridge driver only allows the motor to spin in one direction, which is suitable for our application, which requires a minimum forward speed.

Integration

We constructed our robot on the provided chassis using the supplied motors. We attached the sensors to the bottom of the car using the provided standoffs and screws. Additionally, we attached the Arduino, a breadboard, and the battery to the chassis using command strips.

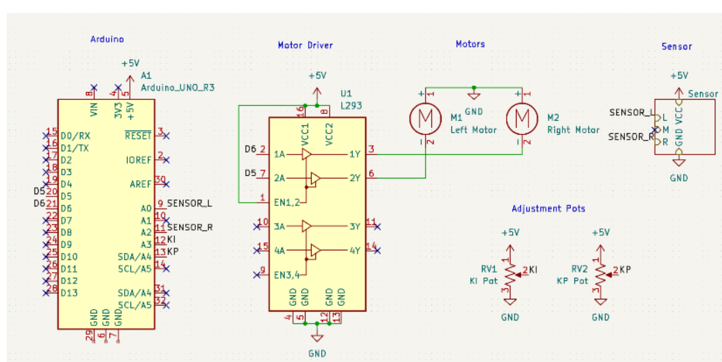

Figure 3 Circuit Schematic

We wired the breadboard and Arduino according to our schematic, which is pictured in Figure 3. The breadboard contained the potentiometers and

L293NE.

A diagram of our car layout is pictured in Figure 4.

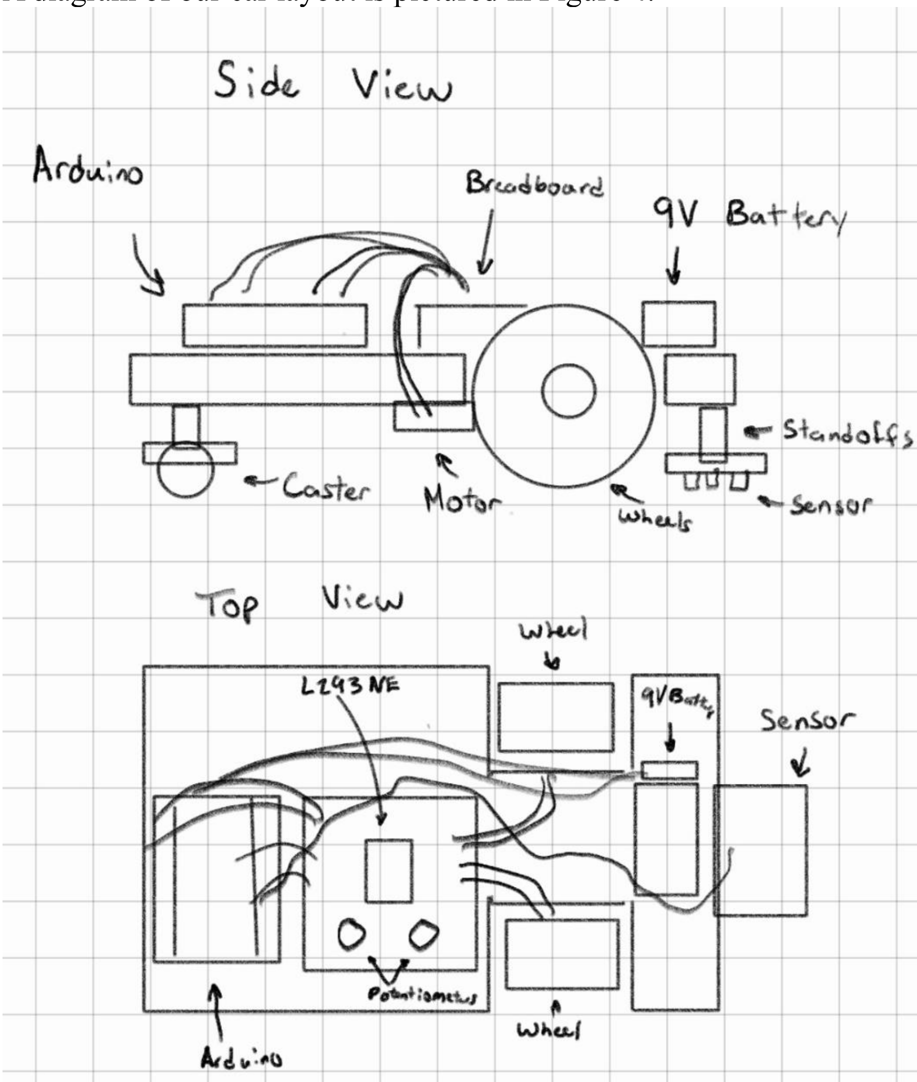

Figure 4 Car Diagram

Test Plan and Test Results

Test Plan

Our test plan was designed to evaluate the performance of the autonomous line-following robot and ensure it met the project requirements. The primary focus areas for testing were:

- Line Following Accuracy: Ensuring the robot can follow a line without deviating.
- Response to Line Deviations: Measuring how quickly and effectively the robot corrects its path when it deviates from the line.

- Speed Control: Verifying that the robot can adjust its speed appropriately based on its distance from the line.
- Reliability: Assessing the robot's consistency in performing the line-following task over multiple trials.

We conducted multiple test runs on a pre-defined course, collecting data on the following parameters:

- Time taken to complete the course.
- Number of times the robot deviated from the line.
- Time taken to correct deviations.
- Speed variations throughout the course.

Test Results

Line Following Accuracy:

- The robot successfully followed the line in 8 out of 10 test runs.
- In 2 test runs, the robot deviated significantly from the line and required manual intervention.

Response to Line Deviations:

- On average, the robot corrected its path within 2 seconds of detecting a deviation.
- The PID control system effectively adjusted the robot’s course, minimizing oscillations.

Speed Control:

- The robot maintained a consistent speed in 7 out of 10 test runs.
- In 3 test runs, minor adjustments to the PID constants were required to stabilize the speed.

Reliability:

- The robot completed the course successfully in 9 out of 10 test runs.
- The single failure was due to a loose connection in the sensor array, which was fixed promptly.

Data Summary:

Test Run	Time	Deviations	Correction Time	Comments
1	30	1	2	Successful
2	32	2	3	Minor oscillations
3	28	0	0	Successful
4	29	2	1.5	Successful

5	31	0	0	Successful
6	33	3	2.5	Minor speed adjustments
7	27	1	2	Successful
8	30	2	3	Successful
9	34	4	3.5	Manual Intervention
10	29	2	2	Successful

Test Videos:
[Video 1]
[Video 2]

Demo Results:

Line-Follower Performance Grade Calculator

All data entry points yellow

Intermediate and Expert course completion times based on **two** laps.

Acceptable Car (10% of total)		10 / 10 pts
Working Car (30% of total)	(follows straight = 10; follows right = 10; follows left = 10)	20 / 30 pts
Complete Basic Course (42% of total)	(1 to 2 = 10; 2 to 3 = 10; 3 to 1 = 10; no intervention = 12)	32 / 42 pts

Advanced Course Scores (18% of total):

	Time (seconds)		Completed Sections
Intermediate Course	34		1
		Score:	
Expert Course			0
		Score:	

Intermediate Course Score	**Expert Course Score**	**Advanced Course Subtotal**
3 / 9	0 / 9	3 / 18

OVERALL PERFORMANCE SCORE **65.00 / 100 pts**

Discrepancies in tests and results:
There were some discrepancies in the tests and the results for a few reasons. The biggest one being that we did not account for the high complexity of the demo day courses and hence we did not test it on courses that intercut. Also, another reason was the floor material, we tested it on smooth wood, versus the rough carpet under the paper for the demo. Since our motors and other materials were not of the highest quality any little change threw our car a little off.

Conclusion

In this mini-project our team designed and built a cost-effective autonomous line-following teaching tool with the goal of enhancing theoretical learning through hands on demonstration. We went through a process of concept generation and selection leading to a final design that offers a good balance of performance, reliability, and educational value.

Design Process

Concept Generation and Selection:

We started by exploring various processing options, comparing the benefits of microcontrollers versus analog components. After extensive benchmarking, we determined that the Arduino Uno microcontroller was the best choice due to its affordability, availability, and ease of programming. For our sensors we considered different line detection methods and decided on IR sensors, avoiding the complexity and cost of a camera-based systems, and the inaccuracy of photo resistors. This choice was guided by the need for a straightforward, effective, and affordable solution.

Control Mechanism:
We examined two control mechanisms – bumper control and PID control. Ultimately, we chose PID control for its superior reliability and educational value, even though it is more complex to implement. Our specific PID control implementation, using proportional and integral components, provided a balance between effectiveness and simplicity.

Motor Control:
For motor control, we opted for H-Bridge drivers instead of relays due to their compatibility with PWM signals, which are crucial for controlling motor speed. The L293NE chip was selected for its relaibilty and availiability, allowing effective motor control.

Performance Evaluation:
We subjected our device to rigorous testing to ensure it met the project requirements. The tests focused on the robot's ability to consistently follow a line and accurately respond to deviations. The results showed that our PID-controlled robot performed reliably, maintaining its course with minimal oscillations. The light-reflection sensors effectively detected the line, and the Arduino Uno managed the control logic efficiently. Overall, the performance of our device met our expectations, validating the design choices made during the concept generation and selection phases. The robot's behavior was consistent, and the PID control adjustments improved its ability to follow the line smoothly.

Objective Evaluation:
In summary, our project successfully achieved its goals of creating a cost-effective, reliable, and educational line-following robot. The design process, from concept generation to final implementation, was guided by careful consideration of performance, cost, and ease of understanding for educational purposes. The testing phase confirmed that our device met the necessary criteria, providing a valuable tool for demonstrating robotics concepts in a classroom setting.

Lessons Learned

Working on this project as a team gave us valuable lessons, particularly in time management and communication.

Time Management:
Throughout the project, we learned the importance of effective time management. By setting clear deadlines and milestones, we were able to keep the project on track and ensure that each phase was completed on time. We also discovered the value of flexibility and adaptability, as unforeseen challenges required us to adjust our schedule and priorities. Regular check-ins and progress updates helped us stay aligned and maintain momentum.

Communication:
Effective communication was crucial to our success. We found that open and honest communication fostered a collaborative environment where ideas could be freely exchanged. Utilizing tools such as video meetings, and collaborative documents, we were able to stay connected and ensure that everyone was on the same page. Active listening and constructive feedback played a significant role in refining our ideas and improving our final product.

Collaboration:
Collaboration required us to respect each other's strengths and weaknesses, delegating tasks accordingly. This approach not only maximized efficiency but also ensured that everyone felt valued and contributed meaningfully to the project. We learned to navigate different working styles and to support one another, which strengthened our teamwork and overall project outcomes.

Problem-Solving:
Working as a team allowed us to tackle problems from multiple perspectives, leading to more innovative solutions. We learned that brainstorming sessions and collective troubleshooting were highly effective in overcoming technical and logistical challenges. This collaborative problem-solving approach enhanced our ability to address complex issues and develop robust solutions.

In summary, this project taught us essential non-technical skills that are crucial for successful teamwork. By mastering time management, communication, collaboration, and problem-solving, we were able to achieve our project goals and deliver a reliable, cost-effective, and educational line-following robot. These skills will undoubtedly benefit us in future projects and professional endeavors.

References

Arduino, "Arduino Uno," [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardUno. [Accessed: 12-Jun-2024].

Digikey, "L293NE Motor Driver IC," [Online]. Available: https://www.digikey.com/en/products/detail/stmicroelectronics/L293NE/582320. [Accessed: 12-Jun-2024].

RobotShop, "Line Following Sensors," [Online]. Available: https://www.robotshop.com/en/line-following-sensors.html. [Accessed: 12-Jun-2024].

IEEE Spectrum, "Understanding PID Control," [Online]. Available: https://spectrum.ieee.org/understanding-pid-control. [Accessed: 12-Jun-2024].

Microchip, "Operational Amplifiers and Comparators," [Online]. Available: https://www.microchip.com/design-centers/operational-amplifiers-and-comparators. [Accessed: 12-Jun-2024].

SparkFun, "Using Relays in Electronic Projects," [Online]. Available: https://www.sparkfun.com/pages/relays. [Accessed: 12-Jun-2024].

Raspberry Pi Foundation, "Raspberry Pi Pico," [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-pico/. [Accessed: 12-Jun-2024].

National Instruments, "Basics of PWM Control," [Online]. Available: https://www.ni.com/en-us/innovations/white-papers/06/pwm-control.html. [Accessed: 12-Jun-2024].

Jameco Electronics, "555 Timer IC," [Online]. Available: https://www.jameco.com/shop/555-timer-ic. [Accessed: 12-Jun-2024].

Adafruit, "Learning about Microcontrollers," [Online]. Available: https://learn.adafruit.com/what-is-an-arduino/overview. [Accessed: 12-Jun-2024].

Appendix - Software / Technology Used

Collaboration Among Team Members

- Cisco WebEx Teams and Meetings
- RPI Box
- Microsoft OneCloud
- Google Drive
- Gannt Chart

Design

- KiCad
- Logic Works
- LTSpice

Programming

- CLion – IDE (C++)
- Platform IO – Arduino-to-IDE interface

Testing/Simulation/Emulation

- LogicWorks
- LTspice
- Test Courses