****

| **Department of**  **Computer Science**  **and Engineering**  **Independent University Bangladesh** |
| --- |

| **TECH FEST PROJECT REPORT Of Team : “Cyber\_Symphony”**  **CSE 204 LAB**  **Summer-2023** |
| --- |

**1. Project Title:**

| Line Following Robot |
| --- |

**2. Faculty Information:**

| Name of Faculty (C0URSE) |  | | |
| --- | --- | --- | --- |
| Designation |  | Department |  |
| Name of Faculty (LAB) |  | | |
| Designation |  | Department |  |

**3. Student Information:**

| Name & ID |  | Department |  |
| --- | --- | --- | --- |
| Course section |  | Lab section |  |
| E-mail |  | Contact No |  |

| Name & ID |  | Department |  |
| --- | --- | --- | --- |
| Course section |  | Lab section |  |
| E-mail |  | Contact No |  |

| Name & ID |  | Department |  |
| --- | --- | --- | --- |
| Course section |  | Lab section |  |
| E-mail |  | Contact No |  |

| Name & ID |  | Department |  |
| --- | --- | --- | --- |
| Course section |  | Lab section |  |
| E-mail |  | Contact No |  |

| Name & ID |  | Department |  |
| --- | --- | --- | --- |
| Course section |  | Lab section |  |
| E-mail |  | Contact No |  |

| Name & ID |  | Department |  |
| --- | --- | --- | --- |
| Course section |  | Lab section |  |
| E-mail |  | Contact No |  |

**4. Abstract:**

| A "line-following robot" is a smart robot that can follow a black line on a white surface. It uses an Arduino UNO to control its movements, infrared (IR) sensors, two wheels, and motors. There are five IR sensors that detect the black line and send information to the Arduino. When the robot sees the line, it moves forward, left, or right. The IR sensors work by bouncing signals off surfaces. If it bounces back from a white surface, the robot knows it's on the line. |
| --- |

**5. Introduction:**

| Imagine a robot that can follow a line you draw for it. It's like a little explorer that sticks to the path you set, whether it's a white line on a black surface or the other way around. The robot has special eyes called IR sensors on its belly that can see the line. These sensors tell an Arduino, which is like the robot's brain, where the line is. The Arduino then tells the robot's wheels to move in the right direction, so it follows the line. It's like magic: when the sensors see white, they tell the robot to go that way! |
| --- |

**6. Problem Statement:**

| Line-following robots are cool, but they have some limitations:   * They can only follow lines that are 1 or 2 inches wide on a white surface. * They're not very smart; they can't adapt to tricky situations without extra sensors. * They need a specific path to follow, either all white or all black, because of how their sensors work. * They're not very fast, and they struggle with lines that are different thicknesses or at strange angles. |
| --- |
| *But, we can make them better! We could give them more wheels to be more stable and add sensors to avoid obstacles. We could also make them go faster and use better motors. With these changes, our robot could handle turns better and deal with situations where the line disappears or breaks.* |

**7. Topology with Circuit diagram:**

***Component’s Description:***

| **1. Motor Driver:**   * Description: A motor driver is a module that controls the direction and speed of motors. It's like the robot's power manager. * Purpose: To regulate and control the movement of the robot's wheels. |  |
| --- | --- |
| **2. Arduino Nano R3:**   * Description: The Arduino Nano R3 is a small programmable microcontroller board. * Purpose: It acts as the brain of the robot, receiving input from sensors and controlling the motors accordingly. |  |
| **3. Chassis Board (Handmade):**   * Description: A custom-made board, usually from cardboard or similar material, that serves as the robot's body. * Purpose: To provide a structure for mounting the motors, wheels, sensors, and other components |  |
| **4. N20 Motor (800 RPM):**   * Description: A small, high-speed motor. * Purpose: To drive the robot's wheels, making it move along the line. |  |
| **5. 42mm x 19mm Wheel:**   * Description: A wheel that attaches to the motors. * Purpose: To provide traction and allow the robot to move. |  |
| **6. Freewheel/Ball Caster:**   * Description: A small, rolling component that provides balance and support to the robot. * Purpose: To help the robot balance and move smoothly. |  |
| **7. 5 IR Sensor Array Module with Crush Detector:**   * Description: A module with five Infrared (IR) sensors and a crush detector. * Purpose: To detect the black line on the surface and obstacles in the robot's path. |  |
| **8. HC-SR04 Ultrasonic Sensor:**   * Description: An ultrasonic sensor that measures distance. * Purpose: To help the robot avoid obstacles by detecting how far they are. |  |
| **9. XL6009 Boost Module with Display:**   * Description: A module that boosts voltage and may have a display. * Purpose: To regulate and boost power for the motors and other components, and possibly display information. |  |
| **10. Li-Ion Battery (x2):**   * Description: Rechargeable Lithium-Ion batteries. * Purpose: To provide power to the robot, usually used in pairs for longer operating times. |  |
| **11. Jumper Wires:**   * Description: Wires with connectors at the ends. * Purpose: To connect different components on the breadboard or elsewhere in the circuit. |  |
| **12. Breadboard (170 Tie Point):**   * Description: A solderless breadboard for prototyping circuits. * Purpose: To create temporary connections between components during testing and development. |  |

***ASSEMBLY AND PIN CONNECTION:***

| **1. Sonar Sensor (Ultrasonic Distance Sensor):**   * Description: Measures distance using ultrasonic waves. * Pin Connections:   + TRIGGER\_Pin (Trigger): Connected to pin 2 on Arduino Nano.   + ECHO\_Pin (Echo): Connected to pin 3 on Arduino Nano. |
| --- |
| **2. Motor Driver:**   * Description: Controls the direction and speed of motors. * Pin Connections:   + ena (Enable A): Connected to pin 10 on Arduino Nano.   + inA (Input A): Connected to pin 9 on Arduino Nano.   + inB (Input B): Connected to pin 8 on Arduino Nano.   + inC (Input C): Connected to pin 7 on Arduino Nano.   + inD (Input D): Connected to pin 6 on Arduino Nano.   + enb (Enable B): Connected to pin 5 on Arduino Nano. |
| **3. Infrared (IR) Sensor Array:**   * Description: Array of IR sensors used for line detection. * Pin Connections:   + IR sensors connected to analog pins A0, A1, A2, A3, A4 on Arduino Nano. |
| **4. Other Components:**   * No specific pin connections mentioned in the code, but these components are used in the project:   + Two Li-Ion Batteries (power source).   + Jumper wires for connecting various components.   + Breadboard (for prototyping circuits).   + Wheels connected to motors.   + Freewheel/Ball Caster for stability. |
| *The code initializes the components and sets up functions for motor control, line following, and obstacle detection using the sonar sensor. The robot follows a black line and stops when an obstacle is detected within a certain distance.* |

***Circuit Explanation***

***Driver Section:***

| **Motor Driver:**  This section includes the motor driver module. Connect the motor driver to the Arduino Nano R3 and the motors. The motor driver controls the direction and speed of the motors.   * + Connect the "ena" and "enb" pins of the motor driver to pins 10 and 5 on the Arduino Nano, respectively.   + Connect the "inA" and "inB" pins of the motor driver to pins 9 and 8 on the Arduino Nano for one motor.   + Connect the "inC" and "inD" pins of the motor driver to pins 7 and 6 on the Arduino Nano for the other motor. |  |
| --- | --- |
| **Arduino Nano R3:**  This section includes the Arduino Nano R3, which serves as the brain of the robot. It controls the motor driver and processes sensor data.   * Connect the Arduino Nano to a power source and ensure it's properly grounded. |  |
| **Infrared (IR) Sensor Array**:  This section involves the IR sensor array used for line detection. These sensors are connected to analog pins on the Arduino Nano.   * Connect each IR sensor's signal pin (analog output) to one of the analog pins A0, A1, A2, A3, A4 on the Arduino Nano. * Connect the power and ground pins of the IR sensors to the appropriate voltage and ground pins on the Arduino Nano. |  |
| **Ultrasonic Distance Sensor (Sonar):**  The ultrasonic distance sensor is used for obstacle detection.   * + Connect the "TRIGGER\_Pin" of the sensor to pin 2 on the Arduino Nano.   + Connect the "ECHO\_Pin" of the sensor to pin 3 on the Arduino Nano. |  |
| **Li-Ion Batteries**:  These power the circuit and motors. Ensure proper connections for power and ground to the motor driver and Arduino Nano. |  |

| **Other Components**: Jumper wires, breadboard, wheels, freewheel/ball caster are used for various connections and mechanical stability. |
| --- |

***Work Flow Algorithm:***

| **Initialization:**   * + The Arduino Nano and all the sensors are initialized.   + The motor driver pins are set up to control the motors. |
| --- |
| **Line Following:**   * + The robot starts moving forward.   + The IR sensor array continuously detects the surface beneath it.   + If the sensor detects a black line, it means the robot is on the line, so it continues moving forward.   + If a sensor detects white, it means the robot is off the line, so it needs to adjust its path.   + Based on which sensors detect black or white, the robot makes decisions to follow the line.   + For example, if the leftmost sensor detects black and the others detect white, it turns slightly left to get back on track. |
| **Obstacle Detection:**   * + The ultrasonic distance sensor (Sonar) measures distances in front of the robot.   + If it detects an obstacle too close (within a defined range), it triggers an obstacle avoidance routine.   + During obstacle avoidance, the robot stops, reverses, or takes a different path to avoid the obstacle. |
| **Feedback Control:**   * + The robot uses a feedback control mechanism.   + It continuously reads the sensor data and calculates an error value based on the difference between the expected position (on the line) and the actual position (where it detects black or white).   + The error value determines how much the robot should adjust its motors to correct its path. |
| **Motor Control:**   * + The motor driver receives commands from the Arduino Nano to control the speed and direction of the motors.   + Depending on the error value calculated, the robot adjusts the motor speeds to stay on the line.   + For example, if it detects a deviation to the left, it increases the speed of the right motor or decreases the speed of the left motor to correct its path. |
| **Looping:**   * + The entire process of reading sensor data, calculating errors, and adjusting motors is repeated in a loop.   + This loop ensures that the robot continuously follows the line and avoids obstacles. |
| **Halt on Obstacle:**   * + If the ultrasonic distance sensor detects an obstacle too close, the robot halts its movement until the obstacle is cleared. |
| **Completion:**   * + The robot continues following the line until it reaches its destination or the task is completed. |
| *This iterative process of reading sensors, making decisions based on sensor data, and controlling the motors allows the Line Follower Robot to navigate along a predefined path, following the line while avoiding obstacles. The feedback control mechanism ensures that the robot can make real-time adjustments to stay on course.* |

| **Block diagram:** |
| --- |
|  |
| **UML Diagram:** |
|  |
| **Circuit Diagram:** |
|  |

| **Flowchart:** | **Flowchart Description:** |
| --- | --- |
|  | Initialize Arduino and Sensors:   * The process begins with the initialization of the Arduino microcontroller and the sensors, including the infrared sensors for line tracking and the ultrasonic sensor for obstacle detection.   Set up Motor Driver:   * The motor driver is configured to control the robot's wheels, allowing it to move forward, backward, or turn.   Start Moving Forward:   * The robot starts moving forward along a predefined path.   While (Line Detected?):   * + This loop checks if the robot still detects a line on the ground.   + If a line is detected (Yes), the robot continues moving forward.   While (Obstacle Detected?):   * + Within the first loop, there is a nested loop that continuously checks if an obstacle is detected by the ultrasonic sensor.   + If an obstacle is detected (Yes), the robot takes the following actions:     - Stops to avoid a collision.     - Executes an "Avoid Obstacle" routine to navigate around the obstacle.     - Resumes following the line once the obstacle is avoided.   .  While (Line Followed?):   * + After avoiding an obstacle, or if no obstacle is detected, the robot checks if it is still following the line accurately.   + If it is not following the line (No), the robot:     - Adjusts motor speeds to correct its path.     - Continues to "Follow Line" using the infrared sensors for guidance.   Task Completed:   * The flowchart ends when the robot successfully follows the line and completes its task. It signifies the successful navigation of the robot. |

***CODE:***

| **#include<NewPing.h>**  **#define TRIGGER\_Pin 2**  **#define ECHO\_Pin 3**  **#define ena 10**  **#define inA 9**  **#define inB 8**  **#define inC 7**  **#define inD 6**  **#define enb 5**  **#define MAX\_DISTANCE 20**  **NewPing sonar(TRIGGER\_Pin, ECHO\_Pin, MAX\_DISTANCE);**  **const int irPins[5] = {A0,A1,A2,A3,A4};**  **void wheel(int lm, int rm);**  **int s[5] = {0,0,0,0,0};**  **int i, lastSensor,lastError;**  **int uturn = -100;**  **int base\_L= -120;**  **int base\_R=-120;**  **float kp=3.1416;**  **float kd=1.2255;**  **void setup()**  **{**  **//Serial.begin(9600);**  **mot\_init();**  **other\_init();**  **ustad\_samne\_plastic();**  **}**  **void loop()**  **{**  **//void stop();**  **line\_follow();**  **//wheel(200,200);**  **unsigned int distance = sonar.ping\_cm();**  **Serial.print("Distance: ");**  **Serial.print(distance);**  **Serial.println(" cm");**  **if (distance < MAX\_DISTANCE) {**  **ustad\_samne\_plastic();**  **Serial.println("Obstacle detected! Motors stopped.");**  **} else {**    **line\_follow();**  **}**  **};**  **void other\_init()**  **{**  **lastSensor=0;**  **lastError=0;**  **//Serial.begin(9600);**  **}**  **void mot\_init()**  **{**  **pinMode(inA,OUTPUT);**  **pinMode(inB,OUTPUT);**  **pinMode(inC,OUTPUT);**  **pinMode(inD,OUTPUT);**  **pinMode(ena,OUTPUT);**  **pinMode(enb,OUTPUT);**  **}**  **void wheel(int lm, int rm)**  **{**  **if(lm==0)**  **{**  **digitalWrite(inC,HIGH);**  **digitalWrite(inD,HIGH);**    **}**  **if(lm>0)**  **{**  **digitalWrite(inC,HIGH);**  **digitalWrite(inD,LOW);**  **}**  **else if(lm<0)**  **{**  **digitalWrite(inC,LOW);**  **digitalWrite(inD,HIGH);**  **}**  **if(rm==0)**  **{**  **digitalWrite(inA,HIGH);**  **digitalWrite(inB,HIGH);**  **}** | **if(rm>0)**  **{**  **digitalWrite(inA,HIGH);**  **digitalWrite(inB,LOW);**  **}**  **else if(rm<0)**  **{**  **digitalWrite(inA,LOW);**  **digitalWrite(inB,HIGH);**  **}**  **if(lm>254) lm=254;**  **if(lm<-254) lm=-254;**  **if(rm>254) rm=254;**  **if(rm<-254) rm=-254;**  **analogWrite(ena,abs(rm));**  **analogWrite(enb,abs(lm));**  **}**  **int readSensor()**  **{**    **s[0]=digitalRead(irPins[0]);**  **s[1]=digitalRead(irPins[1]);**  **s[2]=digitalRead(irPins[2]);**  **s[3]=digitalRead(irPins[3]);**  **s[4]=digitalRead(irPins[4]);**  **s[0]=1-s[0];**  **s[1]=1-s[1];**  **s[2]=1-s[2];**  **s[3]=1-s[3];**  **s[4]=1-s[4];**  **int error,sum;**  **sum=s[0]+s[1]+s[2]+s[3]+s[4];**  **if(sum!=0)**  **{**  **error=(s[0]\*10+s[1]\*20+s[2]\*30+s[3]\*40+s[4]\*50)/sum-30;**  **}**  **else**  **{**  **error=420;**  **}**  **if(s[0]==1) lastSensor=1;**    **else if(s[4]==1) lastSensor=2;**  **//Serial.print(error);**  **//Serial.print(" ");**  **return error;**  **}**  **void ustad\_samne\_plastic()**  **{**    **digitalWrite(inA, LOW);**  **digitalWrite(inB, LOW);**  **digitalWrite(inC, LOW);**  **digitalWrite(inD, LOW);**    **analogWrite(ena, -255);**  **analogWrite(ena, 255);**  **}**  **void line\_follow()**  **{**  **int error,corr;**  **float p,d;**  **error=readSensor();**  **if(error==420)**  **{**  **if(lastSensor==1) wheel(-uturn,uturn);**  **else if(lastSensor==2) wheel(uturn,-uturn);**  **}**  **else**  **{**  **p=kp\*error;**  **d=kd\*(error-lastError);**  **corr=p+d;**  **// Serial.println(corr);**  **wheel(base\_L+corr,base\_R-corr);**  **if((error-lastError)!=0) delay(5);**  **lastError=error;**  **}**  **}** |
| --- | --- |

**8. The possible outcome of the Project (Progress with Expectation):**

| **Successful Line Following**:  The robot should be able to accurately follow a line on the ground using infrared sensors, with the option to fine-tune the performance for precision. |
| --- |
| **Effective Obstacle Avoidance:**  The robot should stop when obstacles are detected within the specified range using the ultrasonic sensor, ensuring it avoids collisions during operation. |
| **Customization and Expansion:**  Depending on your goals, you can further customize and expand the project by integrating additional features, enhancing user interfaces, or participating in robotics competitions. |

**9. Discussion:**

| **Project Overview:**  The project revolves around the development of a line-following robot with built-in obstacle avoidance capabilities.  This robot combines sensor technologies, precise motor control, and control algorithms to navigate a predefined path while avoiding collisions with obstacles in real-time. The key components utilized in this project are infrared sensors for line tracking and an ultrasonic sensor (HC-SR04) for detecting obstacles. The robot's primary functions include tracking the line and reacting to obstacles to ensure safe and efficient navigation. |
| --- |
| **Sensor Integration:**  Infrared sensors are employed to detect and follow a line on the ground. These sensors generate digital readings, which are then analyzed to maintain the robot's position on the path. The robot's ability to accurately follow the line depends on the performance of these sensors.  The ultrasonic sensor, on the other hand, plays a pivotal role in obstacle detection.  It measures distances to identify objects in front of the robot. When the sensor detects an obstacle within the predetermined range, the robot takes evasive action to prevent a collision. This feature ensures the robot's safety during operation. |
| **Control System:**  The robot's control system involves precise motor control and a proportional-derivative (PD) controller. Motor control adjusts the wheel speeds to keep the robot aligned with the line or to make corrections as needed. The PD controller, governed by the proportional (kp) and derivative (kd) gains, enables the robot to respond to deviations from the desired path more smoothly and accurately. |
| **Challenges and Considerations:**  Developing a line-following robot with obstacle avoidance capabilities presents several challenges and considerations:   * PID Parameter Tuning: Achieving optimal line following demands fine-tuning of the PID parameters (kp and kd). These values significantly impact the robot's ability to track the line accurately, and their calibration can be challenging. * Real-World Testing: To ensure the robot's reliability in various environments, extensive real-world testing is necessary. This testing helps identify and address issues related to line quality, lighting conditions, and the ability to detect and react to obstacles effectively. * Safety Precautions: Safety is a paramount concern when designing robots that move autonomously. Implementing robust safety mechanisms, such as emergency stops, is crucial to prevent accidents during operation. * Power Management: Efficient power management is essential, particularly for battery-powered robots. Ensuring that the robot can operate for an adequate duration before requiring recharging or replacement of batteries is vital. |
| **Customization and Future Development:**  One of the strengths of this project is its potential for customization and expansion. The PID parameters and motor control values can be adjusted to optimize the robot's performance. Furthermore, the project can be extended in various ways, including:   * Integration of additional sensors for advanced navigation and environmental awareness. * Implementation of path planning algorithms for complex routes or mapping. * Incorporation of remote control capabilities using wireless communication modules like Bluetooth or RF. * Participation in robotics competitions or exhibitions to showcase the robot's capabilities to a broader audience. |
| *In conclusion, the line-following robot with obstacle avoidance capabilities is a versatile and adaptable project that combines sensor technology, control algorithms, and precision engineering. Its success depends on the careful calibration of parameters and rigorous testing to ensure reliable performance in real-world scenarios. The project provides a solid foundation for learning and experimenting with robotics and offers opportunities for future development and customization.* |

**10. Future Plan:**

| In the future, there are several exciting avenues to explore and enhance this line-following robot with obstacle avoidance. First and foremost, further refinement of the robot's algorithms and control parameters can improve its accuracy and responsiveness. Additionally, integrating advanced sensors such as cameras or LIDAR for environment mapping and more sophisticated navigation opens up possibilities for autonomous exploration and mapping. Remote control options and wireless communication capabilities can extend its versatility. Moreover, integrating machine learning techniques for path planning and decision-making can enhance its adaptability in complex environments. Participating in robotics competitions, collaborating on research projects, or adapting the robot for specific applications like warehouse automation or surveillance are also promising directions. Finally, continuous documentation and sharing of the project's progress and code can contribute to the wider community's understanding and development of robotics. |
| --- |

**11. Budget Information:**

| ***SL*** | ***Component*** | ***Quantity*** | ***Unit Cost (BDT)*** | ***Total (BDT)*** |
| --- | --- | --- | --- | --- |
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |
| **Total Cost (BDT)** | | | |  |

| Signature of Faculty: |  | Date |  |
| --- | --- | --- | --- |