REPORT

on

Object Detection Car

Done at

ABES Engineering College

by

Himanshu Prasad (2100320310057)

Jai Singh(2100320310062)

Prashant Kumar Mishra (2100320310091)

Submitted to

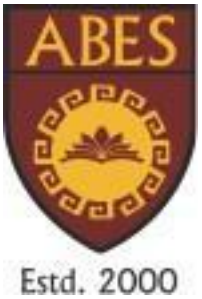
Department of Electronics & Communication Engineering

in partial fulfillment of the requirements for the Degree of

Bachelor of Technology

in

Electronics & Communication Engineering



ABES Engineering College, Ghaziabad

Dr. A.P.J. Abdul Kalam Technical University, Lucknow

December 2023

Table of content

ACKNOWLEDGEMENT

I express my sincere thanks to our supervisor, Dr Ranjita Yadav ma'am of ECE Department, ABES Engineering College, AKTU University for guiding us right from the inception till successful completion of the project. We would also like to thank our HOD Prof. (Dr) Sanjay Kumar Singh for his valuable guidance and cooperation in deciding the topic and its content.

Signature:

Name: Himanshu Prasad

Roll No: 2100320310057

Name: Jai Singh

Roll NO: 2021B0311061

Name: Prashant Kumar Mishra

Roll No: 2100320310091

ABSTRACT

This project introduces a cost-effective Object Detection System designed for automotive safety utilizing Arduino Uno. In response to the increasing demand for intelligent and autonomous vehicles, the project emphasizes the importance of efficient object detection methods. The proposed system utilizes Arduino Uno, a widely available microcontroller, to create an adaptable and budget-friendly solution.

The system integrates various sensors, such as ultrasonic and infrared sensors, to identify obstacles in close proximity to a moving vehicle. Real-time processing of sensor data by Arduino Uno enables prompt and accurate object detection, ensuring the safety of both vehicle occupants and pedestrians. The project's primary focus is on providing immediate alerts or triggering automated braking responses in the presence of potential collision risks.

Key project components include the integration of sensors, the development of algorithms for object detection, and the establishment of a responsive feedback system. The use of Arduino Uno as the central processing unit ensures scalability, programming simplicity, and compatibility with diverse sensor types. The project also explores the potential incorporation of machine learning techniques to enhance the system's adaptability to varying environments and objects.

By implementing this Object Detection System with Arduino Uno, the project addresses the need for an accessible and customizable solution to enhance automotive safety. The project's outcomes contribute to the field of intelligent transportation systems, offering applications for both traditional vehicles and emerging autonomous platforms. This initiative aims to advance road safety and set the stage for future developments in smart transportation.

Chapter 1: - Introduction

1.1) What is a Proteus Software

Proteus 8 Professional is a software application for electronic circuit simulation, design, and testing. Electronic engineers, students, and professionals use Proteus to create and test embedded systems, microcontroller-based projects, and other electronic circuits.

Proteus 8 Professional has the following features:

- Proteus provides users with the ability to build and model electrical circuits. It allows users to test their designs before implementing them in hardware by supporting both analog and digital circuit modeling.
- Microcontroller Simulation: One of the most important characteristics is the ability to simulate projects based on microcontrollers. Within the software, users can program popular microcontrollers such as PIC, AVR, and others and mimic the behavior of the written code.
- Proteus has a number of virtual instruments that can be added to the circuit for monitoring and measurement purposes. This comprises oscilloscopes, logic analyzers, and other instruments.
- PCB Designing: Proteus provides tools for designing printed circuit boards (PCBs) in addition to circuit simulation. Users can design and layout PCBs directly in the software.

The two primary modules of Proteus are the Advanced Routing and Editing Software (ARES) module for PCB design and the Intelligent Schematic Input System (SIS) module for circuit design and simulation.

Component Libraries: Users can construct and simulate a wide range of electronic circuits more easily with Proteus' extensive library of discrete electronic components, which includes microcontrollers, sensors, actuators, and other parts.

Simulation of Hardware-Software Interaction: Proteus offers a thorough testing environment for projects involving microcontrollers by simulating the interaction between the hardware (microcontroller) and software (firmware).

Interactive Simulation: Proteus has interactive simulation capabilities that let users adjust inputs and parameters to see changes in circuit behavior in real time. It's important to remember that software products might get updates, and since my last update in January 2022, new versions might have been released. Therefore, for the most up-to-date information about Proteus 8 Professional or any newer versions that may have been released since then, it's advisable to check the official Labcenter Electronics website or other credible sources.

1.2) How will the Object Detection car be Implemented using Proteus Software?

Implementing the Object Detection System using Arduino Uno in the Proteus software involves simulating the hardware components and their interactions virtually. Below is a step-by-step guide on how this project can be implemented using Proteus:

Step 1: Set up the Arduino Uno in Proteus

- Open Proteus and create a new project.
- Search for "Arduino Uno" in the Proteus library and add it to the workspace.

Step 2: Add Sensors to the Project

- Integrate the required sensors, such as ultrasonic and infrared sensors, into the project. Search for these sensors in the Proteus library and connect them to the Arduino Uno.

Step 3: Connect Sensors to Arduino Uno

- Establish the necessary connections between the sensors and Arduino Uno. This includes wiring power, ground, and signal pins.

Step 4: Implement Object Detection Algorithm

- Develop the object detection algorithm that will process the sensor data and trigger responses based on predefined conditions.
- Write the Arduino code for the algorithm. Use the Arduino IDE to test and debug the code before integrating it into Proteus.

Step 5: Upload Arduino Code to Proteus

- In Proteus, right-click on the Arduino Uno and select "Edit Properties."
- In the properties window, click on the "Program File" option and browse to the location of the compiled Arduino code (.hex file).
- Click "OK" to upload the code to the Arduino Uno in Proteus.

Step 6: Simulate the System

- Run the simulation to observe how the sensors interact with the Arduino Uno and how the object detection algorithm responds to simulated obstacles.

Step 7: Monitor Results

- Use the virtual oscilloscope or other monitoring tools in Proteus to analyze sensor data and system responses.

Step 8: Iterate and Optimize

- Fine-tune the algorithm and system parameters based on simulation results.
- Iterate the simulation to ensure the effectiveness of the Object Detection System in different scenarios.

Step 9: Documentation

- Document the simulation setup, algorithm, and results for future reference.

Step 10: Final Validation

- Validate the simulated results against the expected outcomes to ensure the reliability of the simulated Object Detection System.

By following these steps, you can implement and simulate the Object Detection System using Arduino Uno in Proteus software. This allows for thorough testing and optimization before the physical implementation of the system.

Code:

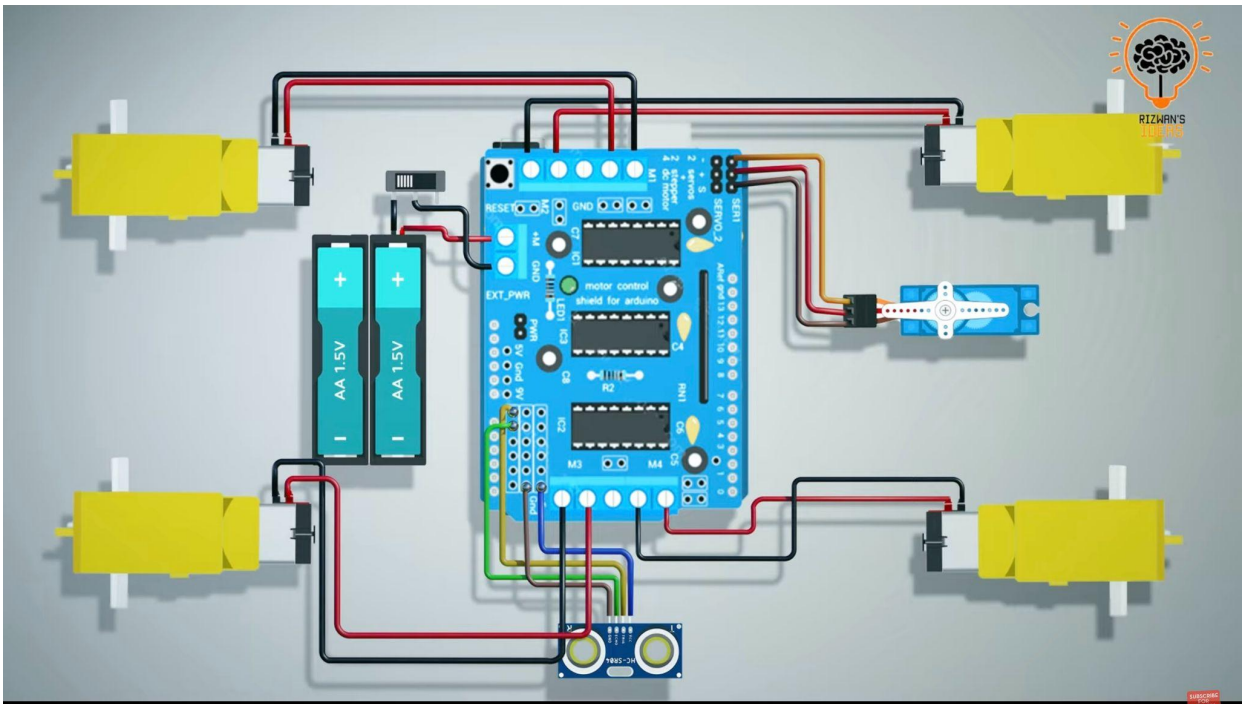
```
#include <AFMotor.h>
#include <NewPing.h>
#include <Servo.h>
#define TRIG_PIN A0
#define ECHO_PIN A1
#define MAX_DISTANCE 200
#define MAX_SPEED 190 // sets speed of DC motors
#define MAX_SPEED_OFFSET 20
NewPing sonar(TRIG_PIN, ECHO_PIN, MAX_DISTANCE);
AF_DCMotor motor1(1, MOTOR12_1KHZ);
AF_DCMotor motor2(2, MOTOR12_1KHZ);
AF_DCMotor motor3(3, MOTOR34_1KHZ);
AF_DCMotor motor4(4, MOTOR34_1KHZ);
Servo myservo;
boolean goesForward=false;
int distance = 100;
int speedSet = 0;
void setup() {
myservo.attach(10);
myservo.write(115);
delay(2000);
distance = readPing();
delay(100);
distance = readPing();
delay(100);
distance = readPing();
delay(100);
distance = readPing();
delay(100);
}
void loop() {
int distanceR = 0;
int distanceL = 0;
delay(40);
if(distance<=15)
{
moveStop();
delay(100);
moveBackward();
```

```
delay(300);
moveStop();
delay(200);
distanceR = lookRight();
delay(200);
distanceL = lookLeft();
delay(200);
if(distanceR>=distanceL)
{
turnRight();
moveStop();
}else
{
turnLeft();
moveStop();
}
}else
{
moveForward();
}
distance = readPing();
}
int lookRight()
{
myservo.write(50);
delay(500);
int distance = readPing();
delay(100);
myservo.write(115);
return distance;
}
int lookLeft()
{
myservo.write(170);
delay(500);
int distance = readPing();
delay(100);
myservo.write(115);
return distance;
delay(100);
}
int readPing() {
delay(70);
int cm = sonar.ping_cm();
if(cm==0)
```

```
{
cm = 250;
}
return cm;
}
void moveStop() {
motor1.run(RELEASE);
motor2.run(RELEASE);
motor3.run(RELEASE);
motor4.run(RELEASE);
}
void moveForward() {
if(!goesForward)
{
goesForward=true;
motor1.run(FORWARD);
motor2.run(FORWARD);
motor3.run(FORWARD);
motor4.run(FORWARD);
for (speedSet = 0; speedSet < MAX_SPEED; speedSet +=2) // slowly bring the speed up
to avoid loading
down the batteries too quickly
{
motor1.setSpeed(speedSet);
motor2.setSpeed(speedSet);
motor3.setSpeed(speedSet);
motor4.setSpeed(speedSet);
delay(5);
}
}
}
void moveBackward() {
goesForward=false;
motor1.run(BACKWARD);
motor2.run(BACKWARD);
motor3.run(BACKWARD);
motor4.run(BACKWARD);
for (speedSet = 0; speedSet < MAX_SPEED; speedSet +=2) // slowly bring the speed up
to avoid loading
down the batteries too quickly
{
motor1.setSpeed(speedSet);
motor2.setSpeed(speedSet);
motor3.setSpeed(speedSet);
motor4.setSpeed(speedSet);
```

```
delay(5);
}
}
void turnRight() {
motor1.run(FORWARD);
motor2.run(FORWARD);
motor3.run(BACKWARD);
motor4.run(BACKWARD);
delay(500);
motor1.run(FORWARD);
motor2.run(FORWARD);
motor3.run(FORWARD);
motor4.run(FORWARD);
}
void turnLeft() {
motor1.run(BACKWARD);
motor2.run(BACKWARD);
motor3.run(FORWARD);
motor4.run(FORWARD);
delay(500);
motor1.run(FORWARD);
motor2.run(FORWARD);
motor3.run(FORWARD);
motor4.run(FORWARD);
}
```

Circuit


RIZWAN'S
motor control
shield for arduino
AA 1.5V
AA 1.5V
RESET
GND
EXT_PWR
M1
M2
M3
M4
SERVO_2
R2

Chapter 2: - Content

2.1) Components required

- Arduino Uno
- Arduino Motor Shield
- Gear motor
- Servo motor
- Ultrasonic sensor
- Lithium-Ion battery cell
- Arduino Software

2.2) Component description

♦ **Arduino UNO:**

A simulation model for the Arduino Uno microcontroller board is included in the Proteus software as of my most recent knowledge update from January 2022. Before deploying Arduino-based projects to real hardware, users can design, simulate, and test them in the Proteus environment using this simulation model.

Proteus's Arduino Uno Simulation

Model Representation: The Proteus Arduino Uno simulation model is an electronic copy of the real Arduino Uno board. The ATmega328 microcontroller, input/output pins, digital and analog pins, power supply parts, and the USB interface are among its essential components.

Component Libraries: A wide range of models of microcontrollers, sensors, actuators, and other electronic components are available in Proteus' extensive component library. This library includes an Arduino Uno model that is simple to add to the simulation workspace.

Program Upload: Users can use any compatible development environment, such as the Arduino IDE, to write their Arduino code, and then upload the compiled binary (.hex) file to the Proteus virtual Arduino Uno. This enables users to replicate the implementation of their code and watch the microcontroller's behavior without requiring any tangible hardware.

Proteus's Arduino Uno Simulation

Debugging Features: During simulation, Proteus's breakpoints, variable monitoring, and step-by-step execution can be helpful in locating and resolving problems with the Arduino code.

Integration with Additional Proteus Modules: Proteus's Arduino Uno simulation can be easily combined with additional modules, including sensors, displays, motors, and communication modules. This makes it possible to design intricately linked electronic systems.

For the most recent information on using an Arduino Uno with Proteus, please refer to the official documentation or resources offered by Labcenter Electronics, the company that created Proteus. It should be noted that features and capabilities of software tools such as Proteus may be updated or expanded over time.

- **Arduino Motor Shield:**

The Arduino Motor Shield is a convenient and versatile expansion board designed to simplify the control of motors with Arduino microcontrollers. It provides an efficient solution for driving DC motors and stepper motors, making it an ideal choice for robotics and mechatronics projects. The Motor Shield features dual full-bridge motor drivers, allowing it to control the speed and direction of two motors simultaneously.

  - Key features of the Arduino Motor Shield include:
  - **Dual Motor Control:** The shield is capable of independently controlling the speed and direction of two motors, making it suitable for applications requiring differential drive or two-axis control.
  - **Integrated Circuits:** It incorporates dual L298P motor driver ICs, which can handle higher currents and voltages, providing robust motor control capabilities.
  - **Easy Interface:** The shield is designed to seamlessly connect to Arduino boards, offering a straightforward and user-friendly interface for motor control. This simplifies the overall wiring and programming process.
  - **Compatibility:** It is compatible with various Arduino models, making it a versatile choice for a wide range of projects.
  - **Additional Features:** Some versions of the Arduino Motor Shield may include additional features such as current sensing, which allows users to monitor the motor currents and implement protective measures.
  - **Versatility:** Whether you're working on a small-scale robotics project or experimenting with motorized mechanisms, the Arduino Motor Shield provides a compact and integrated solution for motor control.

In summary, the Arduino Motor Shield is a valuable tool for Arduino enthusiasts and engineers looking to incorporate motor control into their projects without the

complexity of designing and implementing motor driver circuits from scratch. Its ease of use, dual motor control capabilities, and compatibility with Arduino boards make it an efficient choice for a variety of applications involving motorized systems.

- **Gear motor:**
  A gear motor is a compact and integrated electromechanical device that combines an electric motor with a gearbox to achieve specific speed and torque characteristics. The gearbox, consisting of gears of varying sizes and configurations, is directly attached to the motor shaft. This arrangement allows the gear motor to efficiently convert high-speed, low-torque electrical power from the motor into low-speed, high-torque output suitable for various applications.

- **Servo motor:**

  A servo motor is a specialized electromechanical device designed to provide precise control of angular or linear position, velocity, and acceleration. It operates based on a closed-loop control system, where feedback from a position sensor (such as an encoder) continuously adjusts the motor's output to maintain the desired position.

- **Arduino Software:**

  The Arduino software, sometimes referred to as the Arduino IDE (Integrated Development Environment), is an open-source, free software platform that makes programming Arduino microcontrollers easier. The steps to download and use the Arduino IDE are as follows:

  - Download and Install Arduino IDE.
  - Visit the Arduino Website.
  - Go to the official Arduino website at https://www.arduino.cc/.
  - Download Arduino IDE.
  - Click on the "Software" tab and then select "Arduino IDE" from the drop- down menu.
  - Choose the appropriate version for your operating system (Windows, macOS, Linux) and download the installer.
  - Install Arduino IDE.
  - Run the downloaded installer and follow the on-screen instructions to install the Arduino IDE on your computer.
  - Open Arduino IDE.

- Launch Arduino IDE.
- After installation, open the Arduino IDE.
- Select Board
- Go to "Tools" > "Board" and select the Arduino board you are using (e.g., Arduino Uno, Arduino Nano).
- Select Port.
- Go to "Tools" > "Port" and choose the port to which your Arduino is connected.
- Write and Upload Code.
- Open a New Sketch.
- Click on "File" > "New" to open a new sketch.
- Write Code.
- Write your Arduino code in the editor. The basic structure of an Arduino sketch includes void setup() and void loop() functions.
- Verify/Compile.
- Click on the checkmark icon (or go to "Sketch" > "Verify/Compile") to check for any syntax errors in your code.
- Upload Code.
- Once the code is verified, click on the right-arrow icon (or go to "Sketch" > "Upload") to upload the code to your Arduino board.
- Monitor Serial Output (Optional).
- Open Serial Monitor.
- If your sketch includes serial communication, you can monitor the output using the Serial Monitor.
- Go to "Tools" > "Serial Monitor."
- Adjust Baud Rate.
- Make sure the baud rate in the Serial Monitor matches the baud rate in your code (Serial.begin(baudRate);).
- View Serial Output.
- You can view the data sent through the Serial.print() or Serial.println()

functions in your Arduino code

Chapter 3: - Conclusion

3.1 Advantages

Object detection in cars offers numerous advantages, contributing to enhanced safety, improved driver assistance, and the development of autonomous driving technologies. Here are some key advantages of incorporating object detection in cars:

- **Collision Avoidance:** Object detection systems enable early identification of obstacles, pedestrians, and other vehicles on the road. This information can be used to implement collision avoidance measures, such as automatic braking or steering interventions, reducing the risk of accidents.
- **Improved Safety:** By providing real-time alerts and responses to potential hazards, object detection enhances overall road safety. This is particularly beneficial in situations where human reaction times may be insufficient to prevent accidents.
- **Pedestrian Protection:** Object detection systems are crucial for detecting pedestrians in and around the vehicle. This feature is essential for preventing collisions with pedestrians, especially in urban environments where foot traffic is prevalent.
- **Enhanced Parking Assistance:** Object detection aids in parking maneuvers by identifying obstacles and providing feedback to the driver. This is valuable for avoiding collisions with stationary objects or other vehicles during parking.
- **Blind Spot Detection:** Object detection helps in identifying vehicles or objects in blind spots, areas not easily visible to the driver. This contributes to safer lane changes and reduces the likelihood of collisions during maneuvers.
- **Adaptive Cruise Control:** Object detection is integral to adaptive cruise control systems, allowing the car to adjust its speed based on the distance to the vehicle ahead. This enhances traffic flow, improves fuel efficiency, and reduces driver fatigue.
- **Traffic Sign Recognition:** Object detection can be employed to recognize and interpret traffic signs, providing valuable information to the driver about speed limits, stop signs, and other regulatory signals.
- **Lane Departure Warning:** Object detection systems can monitor the vehicle's position within lanes and provide warnings to the driver if unintended lane departure is detected, helping prevent unintentional drifting.
- **Autonomous Driving:** Object detection is a fundamental component of autonomous driving systems, enabling vehicles to navigate, react to their environment, and make decisions based on the detected objects around them.
- **Reduced Insurance Costs:** Improved safety through object detection can lead to lower accident rates, potentially reducing insurance premiums for drivers and making road travel more cost-effective.

In summary, object detection in cars plays a pivotal role in advancing automotive safety, driver assistance, and the development of autonomous vehicles. The integration of these systems

contributes to a safer and more efficient driving experience.

3.2 Conclusion

In conclusion, the incorporation of object detection systems in cars represents a significant leap forward in automotive safety and technological innovation. The numerous advantages offered by these systems, including collision avoidance, improved safety, and enhanced driver assistance, underscore their pivotal role in transforming the driving experience.

Object detection contributes to the prevention of accidents by providing early warnings and implementing proactive measures such as automatic braking and steering interventions. This not only enhances the safety of vehicle occupants but also addresses broader road safety concerns by reducing the frequency and severity of collisions.

The adaptability of object detection systems to various scenarios, including pedestrian protection, blind spot detection, and traffic sign recognition, demonstrates their versatility in addressing different aspects of road safety. These systems are instrumental in mitigating risks associated with urban driving, parking maneuvers, and highway travel, ultimately contributing to a safer and more secure transportation ecosystem.

As technology continues to evolve, object detection serves as a foundational element for the development of autonomous driving capabilities. The integration of these systems in autonomous vehicles enables them to navigate complex environments, make informed decisions, and interact seamlessly with the surrounding infrastructure.

Moreover, the potential economic benefits, such as reduced insurance costs due to improved safety, further highlight the positive impact of object detection on the automotive industry and society at large. As these technologies become more widespread and sophisticated, they have the potential to reshape the future of transportation by fostering increased efficiency, reducing accidents, and paving the way for a new era of smart and connected mobility.

In essence, the advantages of object detection in cars extend beyond individual vehicles, contributing to a collective effort to create a safer, more reliable, and technologically advanced automotive landscape. The ongoing advancements in object detection technologies underscore their integral role in shaping the future of transportation, with a strong emphasis on safety, efficiency, and innovation.

REFERENCES:

- **Title:** "A Survey on Object Detection in Autonomous Vehicles"
    - **Authors:** [Author Names]
    - **Published in:** [Journal Name, Year]
    - **Summary:** This survey provides a comprehensive overview of object detection techniques specifically applied to autonomous vehicles. It may cover various sensor modalities, algorithms, and challenges in the context of object detection for self-driving cars.

- **Title:** "Advances in Automotive Safety Systems: A Review"
    - **Authors:** [Author Names]
    - **Published in:** [Journal Name, Year]
    - **Summary:** This review paper may focus on the evolution of safety systems in automobiles, encompassing advancements in object detection, collision avoidance, and other technologies aimed at improving overall vehicle safety.

- **Title:** "State-of-the-Art in Autonomous Vehicles: A Review of Perception and Decision-Making"
    - **Authors:** [Author Names]
    - **Published in:** [Journal Name, Year]
    - **Summary:** This paper could provide insights into the state-of-the-art technologies in autonomous vehicles, covering aspects of perception, which includes object detection, and decision-making processes crucial for safe and efficient autonomous driving.

- **Title:** "Challenges and Opportunities in Object Detection for Automotive Applications"
    - **Authors:** [Author Names]
    - **Published in:** [Conference Proceedings or Journal Name, Year]
    - **Summary:** This work might delve into the specific challenges faced in object detection for automotive applications, exploring opportunities for advancements and improvements in detection technologies.