Collision Avoidance System (CAS)

Naman Mishra

07.20.2025

INTRODUCTION

Traveling has become an integral part of our daily lives, whether it is using public transport or driving ourselves. The most important and probably the top priority is safety. No matter if it's a 4-year-old child or a 60-year-old adult, safety is a universal concern for everyone while driving. To satisfy this concern, our vehicles are equipped with a collision avoidance system to give us a warning and give us plenty of time to react and prevent any possibility of accidents.

HYPOTHESIS

We expect that the system will alert drivers whenever it detects an obstacle or another vehicle in the vehicle's path. By identifying potential collisions early, the system is expected to give drivers enough time to respond and avoid accidents. This warning feature should help improve driver reaction times and reduce the number of collisions during testing, whether in simulations or real-world scenarios.

MATERIALS

1. Servo Motor
2. Arduino Board
3. Stepper Motor (with driver to run the stepper motor)
4. Ultrasonic Distance Sensor
5. Passive Buzzer
6. Breadboard
7. External Power source

DESCRIPTION OF MATERIALS

- Servo Motor:

Suitable for applications that require precise control over angular position, a servo motor allows for accurate movement to specific angles. It moves based on a Pulse Width Modulated (PWM) signal, which dictates the rotation. When integrated with a system, the servo motor can adjust the position of components, enabling tasks such as precise

steering or positioning in a wide range of automation applications.

- Arduino Board:

The Arduino Board is a microcontroller board that serves as the system's central nervous system. After processing the data from the ultrasonic sensor, it manages the output parts, which include the stepper motor, servo, and buzzer. It manages decision-making for obstacle detection and alerts and is programmed using the Arduino IDE.

- Stepper Motor:

Suitable for applications such as automated steering or sensor movement, a stepper motor provides precise control over rotation and moves in predetermined steps. When combined with a model, this system can be used to simulate vehicle movement or alter the direction of sensors.

- Ultrasonic Distance Sensor:

By sending out ultrasonic sound waves and timing how long it takes for the echo to return after striking an object, an ultrasonic sensor (such as the HC-SR04) can determine distance. This sensor is essential for identifying roadblocks.

- Passive Buzzer:

This type of buzzer uses electrical signals to produce sound, but it needs an oscillating input signal to produce tones, such as a square wave from the Arduino. The buzzer serves as an alert system in this project, alerting the driver when an impending obstacle is too close.

- Breadboard:

Without soldering, a breadboard enables quick and simple circuit assembly. It is ideal for prototyping because it utilizes metal strips embedded within the board to connect components. For testing and operation, all sensors, the Arduino, and other parts are connected here.

- External Power Source:

When parts like motors require more power than the Arduino can handle, the external power source—such as a battery pack or adapter—provides the voltage and current needed to operate the entire system.

PROCEDURE

For the Stepper Motor:

1. Connect the external power source to the breadboard
2. Connect the stepper motor wires to the driver
3. Ensure the pins for IN-1, IN-2, IN-3, IN-4 are well defined in the Arduino code (IN-1 = pin 8, IN-2 = pin 9, and so on)
4. Power the positive port of the driver to the positive rail of the breadboard where the positive terminal of the external power supply is connected, and connect the negative (ground) to the negative rail of the breadboard in the same rail as the negative terminal of the external power supply.
5. Connect a 9V battery to power the External Power Supply.

For the Ultransonic Distance Sensor:

1. Connect the VCC pin to the 5V pin of the Arduino
2. Connect the GRD (ground) power to the negative rail of the breadboard.
3. Connect the Echo pin to Pin 12
4. Connect the Trigger pin to Pin 13

For the Passive Buzzer:

1. Connect the positive port of the buzzer to the 5V power (make a separate connection from the 5V of the Distance Sensor to the passive buzzer)
2. Connect the negative port to the negative rail as connected before

For the Servo Motor:

1. Connect the power (+ terminal) to the 5V from the Arduino
2. Connect the GRD (- terminal) to the common ground as connected by the other components

NOTE: Make sure all negative ports have a common GRD to ensure proper connection of all devices.

ARDUINO CODE

```
#include <Stepper.h>
#include "SR04.h"
#include "pitches.h"
#include <Servo.h>
#define TRIG_PIN 12
#define ECHO_PIN 13
#define BUZZ_PIN 7
int distance;
int pos = 0;
const int stepsPerRevolution = 64*32;  // For 28BYJ-48 Stepper
Stepper myStepper(stepsPerRevolution, 8, 10, 9, 11);  // N1 - N3 - N2 - N4
SR04 sr04 = SR04(ECHO_PIN, TRIG_PIN);  // Your style
Servo myservo;
void setup() {
  myStepper.setSpeed(15);  // Set stepper speed to 80 RPM
  myservo.attach(6);       // Attach servo to pin 6
  pinMode(BUZZ_PIN, OUTPUT);
  Serial.begin(9600); // initialize the serial port:
  myservo.write(0);  // Start with servo in forward position
}
void loop() {
  distance = sr04.Distance();  // Get distance in cm
```

```
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm")
  if (distance > 5) {
    noTone(BUZZ_PIN);                  // Turn off buzzer
    myservo.write(0);                  // Indicate forward with servo
    Serial.println("Clockwise");
    myStepper.step(stepsPerRevolution);   // Move stepper motor clockwise
  }
  if (distance <= 5) {
    tone(BUZZ_PIN, NOTE_C6);           // Play a tone when object is close
    myservo.write(180);                // Indicate reverse with servo
    Serial.println("Counterclockwise");
    myStepper.step(-stepsPerRevolution);  // Move stepper motor counterclockwise
  }
}
```

CODE EXPLANATION

This section explains how the Arduino code controls the collision avoidance system. The system uses an ultrasonic sensor to detect obstacles and adjusts the motors and buzzer based on the distance readings.

1. Libraries:

   Stepper.h: Used to control the stepper motor (28BYJ-48 in this project).

   SR04.h: Interfaces with the ultrasonic sensor (HC-SR04) to measure distance.

   pitches.h: Provides frequency values for generating sound through the buzzer.

Servo.h: Used to control the servo motor.

2. Pin Definitions:

TRIG_PIN: Connected to the Trigger pin of the ultrasonic sensor.

ECHO_PIN: Connected to the Echo pin of the ultrasonic sensor.

BUZZ_PIN: Connected to the buzzer to alert when an obstacle is detected.

3. Motor and Sensor Setup:

myStepper: Initializes the stepper motor and specifies the number of steps per revolution.

sr04: Initializes the ultrasonic sensor to work with the defined trigger and echo pins.

myservo: Creates an object to control the servo motor.

4. Setup Function:

The setup() function runs once at the beginning of the program to initialize all components:

It sets the stepper motor speed.

Attaches the servo motor to the correct pin.

Configures the buzzer as an output.

Initializes serial communication for debugging (prints the distance on the serial monitor).

Moves the servo to the initial forward position (0°).

5. Loop Function:

The loop() function continuously checks the distance from the ultrasonic sensor and controls the motors and buzzer:

If the distance is greater than 5 cm, the system moves the vehicle forward:

The buzzer is turned off.

The servo motor stays in the forward position.

The stepper motor rotates clockwise, moving the vehicle forward.

If the distance is less than or equal to 5 cm (indicating an obstacle), the system reverses the vehicle:

The buzzer turns on and plays a tone.

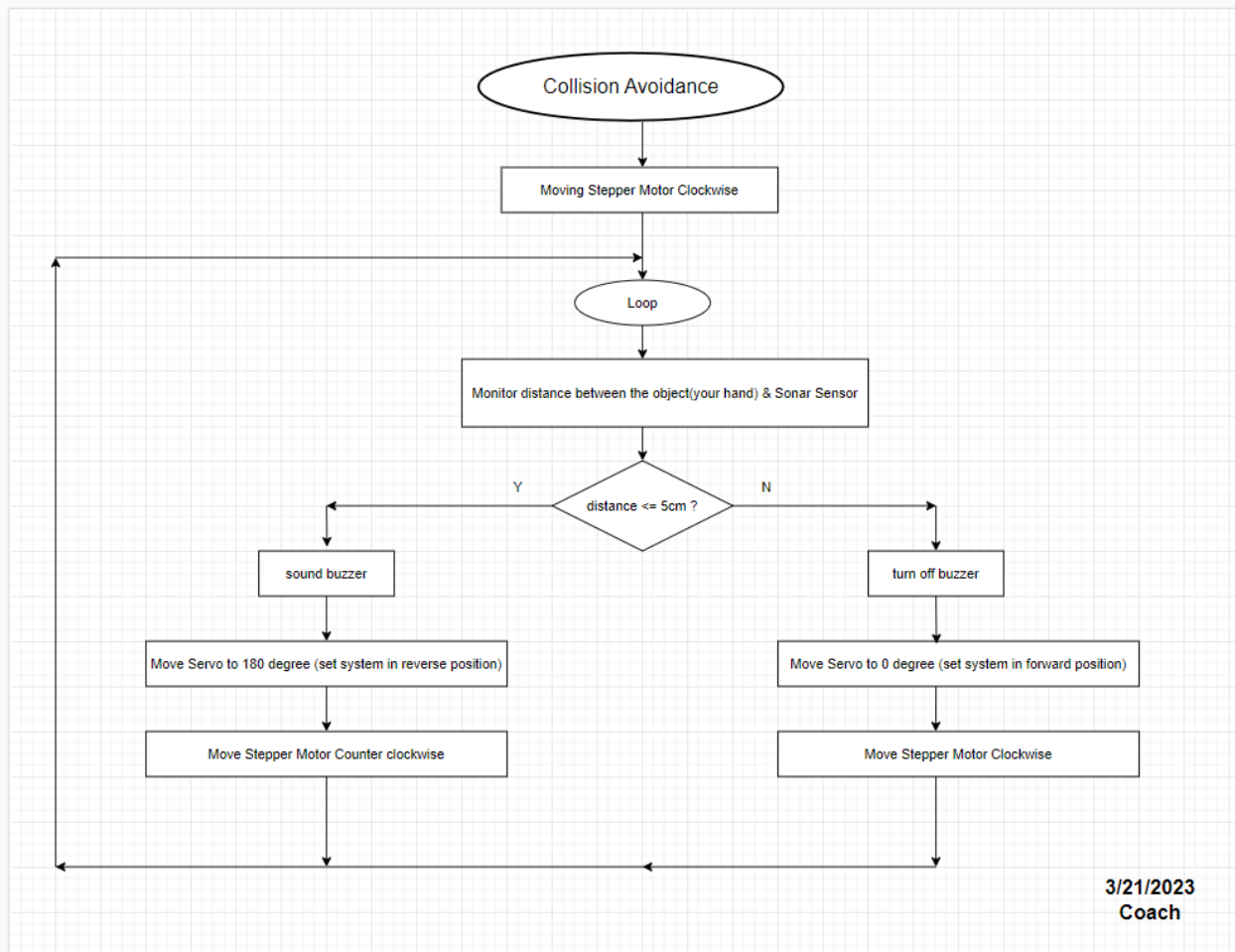
The servo moves to the reverse position (180°).

The stepper motor rotates counterclockwise, reversing the vehicle to avoid the obstacle

LOGIC

The logic behind the collision avoidance system's functioning, from a hardware perspective, is that the stepper motor starts rotating in a clockwise direction when there is no obstacle in sight and the servo motor is stationary in a particular direction. When the ultrasonic distance sensor detects any obstacle, the passive buzzer will beep to warn the driver of an obstacle in the driver's path of travel. This will force the stepper motor to rotate in the counterclockwise direction and the servo motor to reverse its direction to indicate that when there is an obstacle, the safety measures (represented by the stepper and servo motor) kick in and help prevent the vehicle from accidents.

PROJECT FLOWCHART



CONCLUSION

In summary, the project's collision avoidance system demonstrates its ability to recognise obstacles and start the required safety measures, like sounding a buzzer to alert the driver and altering the path of travel. The stepper motor, servo motor, ultrasonic sensor, and passive buzzer work together to create a practical and efficient safety feature that can reduce the likelihood of accidents.

The system performs admirably in real-time tests, accurately detecting obstacles and responding by altering the vehicle's course. Even though the system works as intended, there is room for future advancements like raising the effective detection range and improving sensor accuracy.

REFERENCES

1. Title Image: https://medium.com/@theteamorcad/how-do-collision-avoidance-systems-work-eee02adc745
2. Project Flowchart: Coach, CSC/EEE230 (Summer 2025)

   https://sites.google.com/view/techsys230/arduinosu2025