Contents lists available online at TALENTA Publisher

DATA SCIENCE: JOURNAL OF COMPUTING AND APPLIED INFORMATICS (JoCAI)

Journal homepage: https://jocai.usu.ac.id

Simulation of Vehicle Distance Detection for Traffic Order

Milagros Racacho Baldemor

Don Mariano Marcos Memorial State University, Philippines

email : milagros_baldemor@yahoo.com.ph

ARTICLE INFO

Article history:
Received 29 November 2023
Revised 10 January 2023
Accepted 26 January2024
Published online 31 January 2024

Keywords:
Distance
HC-SR04
ESP32

ABSTRACT

The use of ultrasonic sensors, especially HC-SR04, in microcontroller projects is expanding thanks to its ability to accurately detect distance. In this final project, the HC- SR04 is connected to an ESP32 to measure the distance of an object and provide feedback in the form of sound when the object approaches the sensor within a certain distance. The HC-SR04 sensor works by emitting ultrasonic waves and measuring the time it takes for the waves to reflect back to the sensor. The ESP32, as the microcontroller connected to the sensor, processes this time data to calculate the object's distance from the sensor. When the distance of the object is below a predetermined threshold, the ESP32 will activate the buzzer as a sound signal. This implementation can be applied in various systems, for this test we used it on the zebra crossing system automatically. The test results show that this system is able to detect distance with sufficient accuracy and provide a fast and consistent sound response according to changes in object distance.

Corresponding Author:
mailto:milagros_baldemor@yahoo.com.ph

IEEE style in citing this article:
M. R. Baldemor "Simulation of Vehicle Distance Detection for Traffic Order," DATA SCIENCE: JOURNAL OF COMPUTING AND APPLIED INFORMATICS (JoCAI), vol. 8, no. 1, pp. 37-44. 2024.

1. Introduction

In this era of modern technology, devices that can connect to the internet play a very important role in various applications, whether in industry, home automation, or personal devices. Industrial and home automation devices that can connect to the internet are referred to as the Internet of Things (IoT). IoT devices typically use sensors, software, and technologies that allow IoT devices to be connected, controlled, and monitored directly by/with other devices.

In this project, we designed and implemented a system that uses an HC-SR04 sensor connected to an ESP32 microcontroller to measure the distance of objects. When the object approaches the sensor and is within a certain distance, the system will give feedback in the form of sound through a buzzer. This buzzer will be activated by the ESP32 when the distance of the detected object is below a predeterminedthreshold.

e- ISSN: 2580-829X | p- ISSN: 2580-6769

This system can be applied in various practical situations. For example, in an automated zebra crossing system that provides a warning when a vehicle gets too close to the white line on a zebra crossing to give an audible signal when they are approaching an object.

This project will discuss in detail how the HC-SR04 sensor works, how it is integrated with the ESP32, as well as the algorithm used to process the distance data and activate the buzzer. We will also present the test results of this system to evaluate its performance and accuracy under various conditions. With this research, we hope to provide an inexpensive, easy-to-implement, and effective solution for various practical applications that require distance measurement. So the purpose of this research is to create a device that can detect vehicles from a certain distance.

2. Theoretical Foundation

Internet of things or commonly called IoT is a series of connected devices, IoT devices can also share data between one device and another and can be connected (send / receive data) via an internet connection (via the cloud). Sensors are hardware devices that can detect changes in the surrounding environment and collect data about those changes. IoT sensors can detect many things such as temperature, motion, and sound. The internet of things (IoT) is a concept that aims to extend the benefits of internet connectivity that is connected continuously [1].

One sensor that is often used is an ultrasonic sensor, such as the HC- SR04. This sensor is famous for its ability to accurately measure distance using the principle of ultrasonic waves. It works by sending out ultrasonic waves and measuring the time it takes for the waves to return after bouncing off an object. With this information, the sensor can calculate the distance between the sensor and the object. HC-SR04 ultrasonic sensor is an ultrasonic sensor that uses a frequency of 40Hz. The HC-SR04 ultrasonic sensor is a sensor that can be used to measure the distance between objects and the HC-SR04 sensor. The HC-SR04 ultrasonic sensor consists of 4 pins, namely Vcc, Trigger, Echo and Ground [2]. The way this sensor works is based on the principle of the reflection of a sound wave so that it can be used to interpret the existence (distance) of an object with a certain frequency. It is called an ultrasonic sensor because this sensor uses ultrasonic waves (ultrasonic sound). Ultrasonic waves are sound waves that have a very high frequency, namely 20,000 Hz [3].

The ESP32 is a very popular microcontroller due to its high capabilities, including Wi-Fi and Bluetooth connectivity, and a wide range of input/output features that allow integration with various sensors and actuators. The ESP32 microcontroller is an integrated SoC (System on Chip) microcontroller with 802.11 b/g/n WiFi, Bluetooth version 4.2, and various peripherals. ESP32 is a fairly complete chip, there is a processor, storage and access to GPIO (General Purpose Input Output). ESP32 can be used for a replacement circuit on Arduino, ESP32 has the ability to support connecting to WI-FI directly.This board has a USB to UART interface that is easily programmed with application development programs such as the Arduino IDE. The board's power source can be provided through the micro USB connector [4]. Connecting the HC-SR04 with the ESP32 gives us a powerful platform for developing interactive and real-time distance measurement systems.

3. Result and Discussion

3.1 System Design Concept

To form the IoT circuit in this simulation, we used :

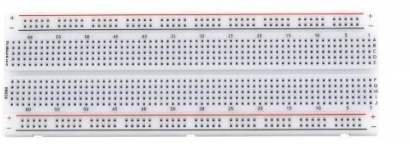

Figure1. BreadBoard

Figure 2. Jumper Cable

Figure3. ESP32

Figure 4. Buzzer

Figure 5. Ultrasonic sensor (HC-SR04)

Figure 6. LED

Figure 7. Ruler

Figure 8. Laptop

Figure 9. Toy Car

Ultrasonic sensors are useful as inputs, while LEDs and *buzzers* are outputs in this IoTdevice simulation.

3.2 Program

This code is a program for the ESP32 microcontroller that uses an ultrasonic sensor (HC-SR04) to measure distance and then sends the data to the Blynk platform for distance monitoring. This program also controls buzzer and LED based on the detected distance. Here is the program and its explanation:

```
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID "TMPL6vG_R5K2m"
#define BLYNK_TEMPLATE_NAME "Sensor Jarak"
#define BLYNK_AUTH_TOKEN "T3c2smqYh_X3xuF4Omku_kFLMX9aTFvU"

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

#define BLYNK_PRINT Serial: Mengatur output Blynk untuk dicetak di Serial
Monitor.
#define BLYNK_TEMPLATE_ID, BLYNK_TEMPLATE_NAME, BLYNK_AUTH_TOKEN:
Mendefinisikan ID template Blynk, nama template, dan token otentikasi untuk
menghubungkan ke aplikasi Blynk.
#include <WiFi.h>, #include <WiFiClient.h>, #include <BlynkSimpleEsp32.h>:
Menginklusi library yang diperlukan untuk koneksi Wi-Fi dan Blynk.


// Blynk Auth Token
char auth[] = BLYNK_AUTH_TOKEN;

// WiFi credentials
char ssid[] = ".";
char pass[] = "17171717";

#define TRIG_PIN 5
#define ECHO_PIN 18
#define BUZZER_PIN 4
#define LED_PIN 2

BlynkTimer timer;

auth, ssid, pass: Menyimpan token otentikasi Blynk dan kredensial WiFi.
TRIG_PIN, ECHO_PIN, BUZZER_PIN, LED_PIN: Mendefinisikan pin untuk sensor
ultrasonik, buzzer, dan LED.
BlynkTimer timer: Membuat objek timer Blynk.


void setup() {
  // Debug console
  Serial.begin(115200);

  // Blynk
```

```
  Blynk.begin(auth, ssid, pass);

  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(LED_PIN, OUTPUT);

  // Set initial state
  digitalWrite(LED_PIN, LOW);
  digitalWrite(BUZZER_PIN, LOW);

  // Setup a function to be called every second
  timer.setInterval(1000L, checkDistance);
}
```

Serial.begin(115200): Memulai komunikasi serial dengan baud rate 115200.
Blynk.begin(auth, ssid, pass): Menghubungkan ESP32 ke Blynk dengan kredensial yang diberikan.
pinMode(): Mengatur mode pin untuk sensor ultrasonik, buzzer, dan LED.
digitalWrite(): Mengatur keadaan awal LED dan buzzer menjadi mati (LOW).
timer.setInterval(1000L, checkDistance): Menjadwalkan fungsi checkDistance untuk dipanggil setiap detik.

```
void loop() {
  Blynk.run();
  timer.run(); // Initiates BlynkTimer
}
```

Blynk.run(): Menjalankan fungsi Blynk.
timer.run(): Menjalankan fungsi timer.

```
void checkDistance() {
  long duration,
  distance;

  // Memicu HC-SR04
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
```

```
    // Membaca sinyal Echo
    duration = pulseIn(ECHO_PIN, HIGH);
    distance = (duration / 2) / 29.1;

  if (distance < 10) { // Jarak deteksi dalam cm
    digitalWrite(BUZZER_PIN, HIGH); // Nyalakan buzzer
    digitalWrite(LED_PIN, HIGH);    // Nyalakan LED
    Blynk.virtualWrite(V1, HIGH);   // Virtual LED widget di Blynk
  } else {
    digitalWrite(BUZZER_PIN, LOW);  // Matikan buzzer
    digitalWrite(LED_PIN, LOW);     // Matikan LED
    Blynk.virtualWrite(V1, LOW);    // Matikan Virtual LED di Blynk
  }

  // Tampilkan jarak di Serial Monitor
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
}
```

Sets the TRIG pin to trigger the ultrasonic sensor. Measures
the duration of the echo signal with pulseIn(). Calculates the distance based on the echo signal duration. If the distance is less than 10 cm, the buzzer and LED will light up, and the status will be sent to Blynk. If the distance is greater than or equal to 10 cm, the buzzer and LED will turn off, and the status will be sent to Blynk.
Display the distance on the Serial Monitor.
The program as a whole functions to measure the distance with the HC- SR04 sensor, control the buzzer and LED based on the distance, and send the distance data and device status to the Blynk platform for real-time distance monitoring.

3.3 Implementation

We placed two toy cars in a position before the zebra crossing, but still close to the zebra crossing. In front, there is an IoT circuit that functions to detect the distance on the car so that it does not cross the zebra crossing. When the car passes the zebra crossing, the sensor will be triggered, then the LED lights will light up and the buzzer will make a sound.

When doing live simulations, we also use a ruler to determine the distance. We set a distance of 20 cm and when the car is under 20 cm, the sensor activates, and the LED and buzzer light up. When the car is moved back, the sensor turns off and when it is moved forward again, the sensor activates again, and the LED lights up and the buzzer sounds. Likewise, when the truck is at a distance of less than 20 cm, the sensor will activate and make a sound.

The device can also be monitored by a computer in real-time. Here, the distance sensor uses the blynk cloud. It contains an indicator LED, buzzer, and distance. It can be seen on the monitor screen in the form of a distance written below 20 cm, the indicator LED is active, and the buzzer is also active. For example, if the car is backed up, the indicator LED and buzzer will automatically turn off and the distance will be at a distance of 22 cm because we created the distance sensor backwards to that point.

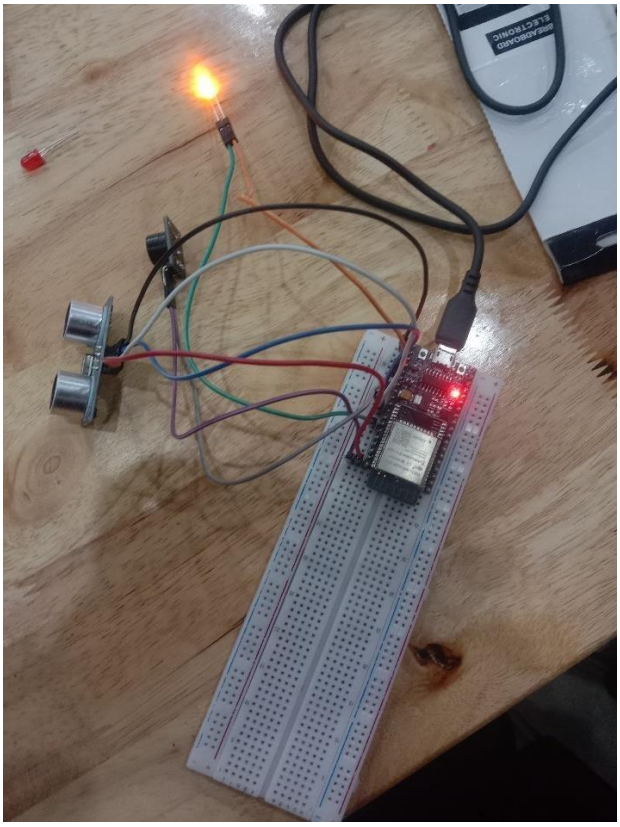

Figure 10. Overview of the circuit created

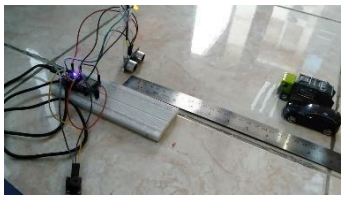

Figure 11. Device State When System is On

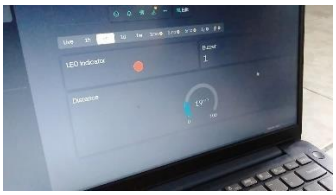

Figure 12. BLYNK Condition When the System is On

Figure 13.Device State When the System is Off

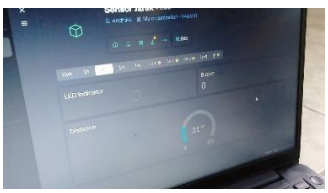

Figure 14. BLYNK Condition When the System is Off

4. Conclusion

The conclusion of this simulation is to successfully implement a distance monitoring system based on HC-SR04 ultrasonic sensor controlled by ESP32 microcontroller. The program created also allows users to monitor the distance in real time through the *Blynk* platform and provide physical feedback through *buzzers* and LEDs. In detail, here is the conclusion we got:

a. The program successfully connects the ESP32 to the Blynk platform using the WiFi credentials and authentication token provided.
b. Distance Measurement: The HC-SR04 ultrasonic sensor is used to measure distance by emitting an ultrasonic signal and calculating the return reflection time to determine the distance to the object.
c. If the detected distance is less than 20 cm, the buzzer and LED will turn on as a warning. If the distance is greater than or equal to 20 cm, the buzzer and LED willturn off.
d. Distance data and device status (on/off for buzzer and LED) are sent to the Blynk platform so that users can monitor device status from the Blynk app in real-time.
e. The program also prints the measured distance to the Serial Monitor, which isuseful for debugging and monitoring directly through a computer connected to theESP32. Overall, this program demonstrates how IoT (Internet of Things) can be used to wirelessly measure and monitor distances and provide physicalfeedback and data to *cloud* applications for remote monitoring.

References

[1] Panduardi, F., & Haq, E. S. (2016). Wireless Smart Home System Menggunakan Raspberry Pi. Jurnal Teknologi Informasi Dan Terapan, 3(1), 320–325.

[2] Fauzan, Mohamad Nurkamal. Dkk. 2019. Tutorial Membuat Prototipe Prediksi Ketinggian Air (PKA) Untuk Pendeteksi Banjir Peringatan Dini Berbasis IOT. Bandung : Kreatif Industri Nusantara

[3] Santoso, Hari. (2015). Panduan Praktis Arduino Untuk Pemula.v1. Indonesia: ELANGSAKTI.com

[4] Wagyana, A., & Rahmat. 2019. Prototype Modul Praktik Untuk Pengembangan Aplikasi Internet Of Things (Iot). Jurnal Ilmiah Setrum, 240-241.