Bộ Giáo Dục Và Đào Tạo

Trường Đại Học Ngoại Ngữ - Tin Học Thành Phố Hồ Chí Minh

Khoa Công Nghệ Thông Tin

MÔN HỌC : INTERNET VẠN VẬT

ĐỀ TÀI : XÂY DỰNG CẢM BIẾN KHOẢNG CÁCH DI ĐỘNG

Giáo Viên Hướng Dẫn : Thầy Nguyễn Tuấn Anh

Nhóm: 19

Thành Viên :

1. Ngô Gia Bảo – MSSV: 22DH114448
2. Phùng Bảo Khang – MSSV: 22DH111531
3. Nguyễn Kim Ngân – MSSV: 22DH112315

Tp. Hồ chí minh, Ngày .... tháng .... năm ...

LỜI CẢM ƠN

Bài báo cáo môn học Internet vạn vật với đề tài "Xây dựng cảm biến khoảng cách di động" là kết quả của quá trình nỗ lực nghiên cứu, học hỏi và làm việc nhóm của chúng em. Để hoàn thành đề tài này, nhóm em đã nhận được sự hướng dẫn tận tình của thầy Nguyễn Tuấn Anh, giảng viên môn Internet vạn vật. Thầy đã trang bị cho chúng em những kiến thức nền tảng quan trọng và kỹ năng cần thiết để thực hiện đồ án một cách hiệu quả.

Nhóm xin gửi lời cảm ơn chân thành đến thầy, người đã tận tâm hỗ trợ, giải đáp thắc mắc và định hướng giúp chúng em vượt qua những khó khăn trong quá trình thực hiện đề tài. Đồng thời, chúng em cũng xin cảm ơn các bạn cùng lớp đã chia sẻ ý kiến, góp phần giúp nhóm cải thiện và hoàn thiện bài báo cáo.

Mặc dù đã cố gắng hết sức, nhưng do hạn chế về kiến thức và kinh nghiệm, bài báo cáo chắc chắn vẫn còn những thiếu sót. Nhóm rất mong nhận được sự góp ý, đánh giá từ thầy và các bạn để đề tài có thể hoàn thiện hơn.

Chúng em xin chân thành cảm ơn!

MỤC LỤC

DANH MỤC HÌNH

DANH MỤC BẢNG

LỜI NÓI ĐẦU

Hiện nay, nhu cầu đo đạc khoảng cách chính xác giữa các đối tượng đang ngày càng được quan tâm trong bối cảnh các hệ thống tự động hóa và giám sát thông minh phát triển mạnh mẽ. Theo báo cáo của các chuyên gia trong ngành, hơn 70% hệ thống tự động hiện đại đã ứng dụng các giải pháp cảm biến để đảm bảo an toàn và hiệu quả vận hành. Đặc biệt, việc tích hợp cảm biến khoảng cách di động đã giúp tối ưu hóa khả năng phát hiện va chạm và điều chỉnh khoảng cách giữa các đối tượng theo thời gian thực, từ đó giảm thiểu rủi ro và nâng cao độ an toàn trong các ứng dụng như robot tự động, xe tự lái và hệ thống giám sát an ninh.

Cảm biến khoảng cách di động không chỉ cho phép thu thập dữ liệu liên tục mà còn đáp ứng được các yêu cầu về kích thước nhỏ gọn, tiêu thụ năng lượng thấp và khả năng di chuyển linh hoạt. Điều này mở ra nhiều cơ hội ứng dụng trong các lĩnh vực đòi hỏi độ chính xác cao và phản ứng nhanh, giúp tối ưu hóa quy trình hoạt động và giảm thiểu thiệt hại do lỗi hệ thống gây ra. Nghiên cứu này hướng đến việc phát triển một giải pháp tích hợp công nghệ cảm biến hiện đại, nhằm đảm bảo khả năng đo lường chính xác và ổn định trong mọi điều kiện vận hành, góp phần thúc đẩy quá trình chuyển đổi số trong ngành công nghiệp và đời sống hiện đại.

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1.1 Giới thiệu đề tài

Trong thời đại số hiện nay, cảm biến khoảng cách di động đang trở thành một trong những công nghệ then chốt giúp nâng cao hiệu quả của các hệ thống tự động và thông minh. Sự phát triển của công nghệ vi xử lý, kết hợp với các thuật toán xử lý tín hiệu tiên tiến và cảm biến hiện đại, đã cho phép xây dựng những thiết bị có khả năng đo đạc khoảng cách chính xác, thu thập dữ liệu thời gian thực và dễ dàng di chuyển trong các môi trường phức tạp.

Cảm biến khoảng cách di động được ứng dụng rộng rãi trong nhiều lĩnh vực như robot tự động, xe tự lái, giám sát an ninh và các hệ thống IoT. Nhờ khả năng liên tục cập nhật thông tin, các thiết bị này không chỉ giúp tối ưu hóa quá trình điều khiển mà còn đóng vai trò quan trọng trong việc dự báo và ngăn ngừa các rủi ro tiềm ẩn. Sự linh hoạt và khả năng thích nghi với nhiều hoàn cảnh khác nhau của cảm biến di động tạo điều kiện thuận lợi cho việc triển khai các giải pháp công nghệ cao trong đời sống hàng ngày.

Bên cạnh đó, các thách thức như cân bằng giữa độ chính xác và tiêu thụ năng lượng luôn được các nhà nghiên cứu quan tâm. Nỗ lực cải tiến công nghệ vật liệu, thiết kế mạch điện và phát triển các thuật toán hiệu quả đang góp phần làm giảm kích thước thiết bị mà vẫn đảm bảo hiệu suất vượt trội. Qua đó, cảm biến khoảng cách di động không chỉ là công cụ hỗ trợ đắc lực cho các ứng dụng hiện đại mà còn mở ra nhiều triển vọng đổi mới sáng tạo, thúc đẩy sự tiến bộ của xã hội trong kỷ nguyên công nghệ số.

1.2 Mục tiêu của đề tài

Xây dựng hệ thống đo khoảng cách sử dụng cảm biến siêu âm HC-SR04, giúp xác định khoảng cách đến vật cản trong các môi trường khác nhau.

Hiểu về nguyên lý hoạt động của cảm biến siêu âm HC-SR04 và cách thức hoạt động của nó trong việc đo khoảng cách dựa trên thời gian truyền và phản xạ sóng âm.

Nghiên cứu và áp dụng các phương pháp tính toán khoảng cách dựa trên dữ liệu đo từ cảm biến, tối ưu thuật toán để tăng độ chính xác. Xây dựng hệ thống IoT, kết nối

cảm biến với điện thoại di động hoặc máy tính, giúp hiển thị thông tin khoảng cách một cách dễ dàng và trực quan.

Thiết kế ứng dụng di động bằng MIT App Inventor hoặc nền tảng tương tự, giúp người dùng theo dõi thông tin đo khoảng cách từ cảm biến một cách nhanh chóng và tiện lợi.

Lập trình trên Arduino IDE, bao gồm đọc tín hiệu từ cảm biến HC-SR04, xử lý dữ liệu và giao tiếp với ứng dụng di động hoặc máy tính qua Wi-Fi/Bluetooth.

Đánh giá và tối ưu hệ thống bằng cách thử nghiệm trong các môi trường khác nhau (nhiệt độ, độ ẩm, bề mặt vật cản), từ đó cải thiện độ chính xác và độ tin cậy của hệ thống.

→ Mục đích của nghiên cứu là cung cấp một giải pháp đo khoảng cách đơn giản, hiệu quả và dễ sử dụng, có thể ứng dụng trong nhiều lĩnh vực như robot tự hành, an ninh giám sát và IoT.

1.3 Giới hạn đề tài

Phạm vi đo lường:

- Cảm biến siêu âm HC-SR04 có phạm vi đo từ 2 cm đến 400 cm.
- Nếu vật cản quá gần (<2 cm) hoặc quá xa (>400 cm), cảm biến có thể không đo được chính xác hoặc không nhận được tín hiệu phản hồi.

Phạm vi ứng dụng:

- Sai số của HC-SR04 thường trong khoảng ±1 cm, nhưng có thể lớn hơn nếu môi trường có nhiễu.
- Khoảng cách đo bị ảnh hưởng bởi vận tốc âm thanh, thay đổi theo nhiệt độ, độ ẩm và áp suất không khí.
- Độ chính xác giảm nếu bề mặt vật cản không phẳng hoặc có góc nghiêng so với cảm biến.

Ảnh hưởng của môi trường:

- Không thể hoạt động trong chân không: Sóng siêu âm cần môi trường để truyền, nên không hoạt động trong chân không.

- Bị hạn chế trong môi trường nước: Cảm biến HC-SR04 thiết kế để đo trong không khí, không phù hợp với nước hoặc chất lỏng.
- Khả năng bị nhiễu bởi môi trường: Tiếng ồn từ các thiết bị siêu âm khác có thể gây nhiễu, ảnh hưởng đến độ chính xác.

Hạn chế trong ứng dụng:

- Phù hợp cho giáo dục, nghiên cứu và thí nghiệm, nhưng không đảm bảo độ chính xác cao trong công nghiệp hoặc y tế.
- Không thể đo khoảng cách qua vật trong suốt (như kính), vì sóng siêu âm không xuyên qua vật thể rắn.

→ Những giới hạn trên nhằm đảm bảo đề tài được thực hiện trong phạm vi khả thi về thời gian, kinh phí và kỹ thuật, đồng thời vẫn đạt được các mục tiêu nghiên cứu đã đề ra.

1.4 Cấu trúc báo cáo

Báo cáo gồm 5 chương, trình bày toàn bộ quá trình nghiên cứu, xây dựng và thực nghiệm hệ thống đo khoảng cách bằng cảm biến siêu âm HC-SR04 và hiển thị qua màn hình LCD I2C:

- Chương 1 giới thiệu về đề tài, những mục tiêu và giới hạn của đề tài. Ngoài ra, chương này cũng trình bày bố cục của báo cáo để người đọc dễ nắm bắt nội dung.
- Chương 2 trình bày những cơ sở lý thuyết về internet vạn vật, giới thiệu về các thành phần và ứng dụng của internet vạn vật.
- Chương 3 mô tả chi tiết về cách xây dựng hệ thống từ kết nối phần cứng đến xây dựng phần mềm và ứng dụng di động, sau đó tiến hành thực nghiệm để đánh giá và có hướng cải tiến hệ thống.
- Chương 4 kết luận và tổng hợp những việc làm và chưa làm được
- Chương 5 mở rộng thêm về đề tài

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Tổng quan về Internet vạn vật

2.1.1 Giới thiệu về Internet vạn vật

Internet of Things (IoT) hay Internet vạn vật là mạng lưới các thiết bị được kết nối internet để truyền tải dữ liệu với nhau và với đám mây mà không cần đến sự tương tác trực tiếp giữa người với người, hay giữa người với máy tính. Các thiết bị này bao gồm mọi thứ từ đồ gia dụng hàng ngày đến các công cụ công nghiệp phức tạp.

Khi kết nối với internet, các đối tượng này trở nên thông minh hơn, có khả năng gửi, nhận thông tin và thực hiện các hành động tự động dựa trên dữ liệu thu thập được. Các thiết bị IoT có thể là đối tượng được trang bị cảm biến để thu thập thông tin về môi trường xung quanh, máy tính, bộ điều khiển để nhận và xử lý dữ liệu, sau đó tự động tương tác với các thiết bị khác. Ngoài ra, IoT cũng có thể là các đối tượng tích hợp cả hai tính năng trên. (Vettelidc, 2024)

2.1.2 Kiến trúc Internet vạn vật

Hệ thống Internet of Things (IoT) là một tập hợp bao gồm các thiết bị có nhiệm vụ thu thập dữ liệu hiện trường, các bộ kết nối (phần cứng) và dịch vụ truyền thông, lưu trữ dữ liệu, trích xuất và báo cáo (phần mềm) được kết hợp hài hoà giúp xử lý thông tin, tối ưu và nâng cao chất lượng hệ thống, tạo ra sự tiện lợi cho người sử dụng. Các thiết bị được chia vào các tầng riêng biệt (hay còn được gọi là lớp IoT – IoT layers).

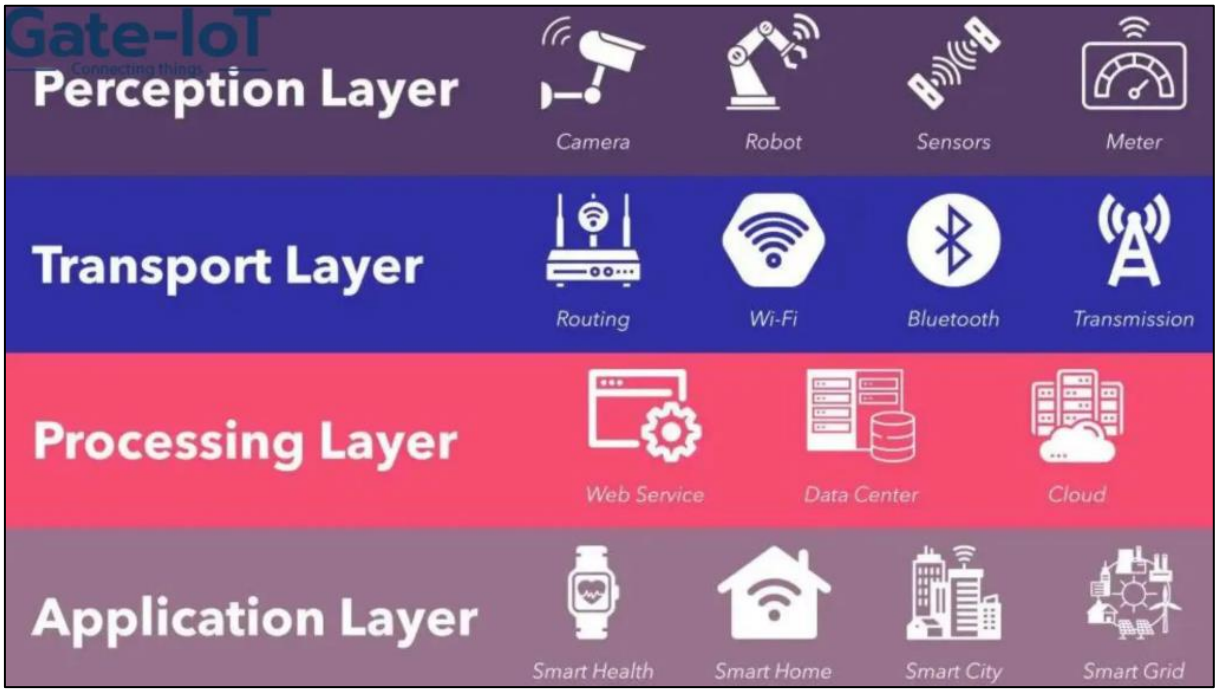
Tuỳ thuộc vào yêu cầu và độ phức tạp của hệ thống, có thể có sự xuất hiện thêm của các tầng thiết bị khác nhằm quản lý dữ liệu và nâng cao bảo mật, bảo trì bảo dưỡng đơn giản. Tuy nhiên về cơ bản, một hệ thống IoT được cấu thành từ bốn tầng bao gồm: Tầng thu thập thiết bị trường (Perception Layer – Edge Devices), tầng tuyền tải thông tin (Transport Layer – Gateways), tầng xử lý dữ liệu (Processing Layer) và cuối cùng là tầng ứng dụng (Application Layer).

Tầng thu thập thiết bị trường (Perception Layer – Edge Devices): Đúng như tên gọi, tầng thiết bị trường chính là những thiết bị vật lý có nhiệm vụ cụ thể như thu thập dữ liệu, đo đạc thông số của môi trường, nhà máy, cơ thể con người, thông số kỹ thuật và tình trạng hoạt động của máy móc…

Tầng truyền tải thông tin (Transport Layer – Gateways): Nếu so sánh những thiết bị vật lý có nhiệm vụ thu thập dữ liệu tại hiện trường như các giác quan của con người như mắt, mũi, tai…thì tầng truyền tải thông tin có nhiệm vụ như những mạch máu và những nơ ron thần kinh, có nhiệm vụ truyền tải thông tin thu thập được đến bộ não (bộ thu thập và xử lý dữ liệu).

Tầng xử lý dữ liệu (Processing Layer): Nhiệm vụ chính của tầng xử lý dữ liệu là thu thập dữ liệu từ thiết bị trường thông qua các giao thức truyền tải, lưu trữ và ứng dụng những thuật toán để dự đoán, đưa ra quyết định cho người sử dụng.

Tầng ứng dụng (Application Layer): Tầng ứng dụng có nhiệm vụ cung cấp cho người dùng những thông tin thu thập được từ hệ thống, tự động hoá quy trình và cải thiện chất lượng, đưa ra quyết định qua những thiết bị khác nhau như điện thoại thông minh, màn hình HMI trong hệ thống công nghiệp… giúp thao tác dễ dàng và đơn giản hoá quá trình vận hành. (Gate-Iot)



Hình 2.1 Cấu trúc tổng quan của Internet Of Things

2.2 Tổng quan về Arduino IDE và MIT App Inventor

2.2.1 Arduino IDE

Arduino IDE là một phần mềm mã nguồn mở chủ yếu được sử dụng để viết và biên dịch mã vào module Arduino. Đây là một phần mềm Arduino chính thức, giúp cho việc biên dịch mã trở nên dễ dàng mà ngay cả một người bình thường không có kiến thức kỹ thuật cũng có thể làm được.

Nó có các phiên bản cho các hệ điều hành như MAC, Windows, Linux và chạy trên nền tảng Java đi kèm với các chức năng và lệnh có sẵn đóng vai trò quan trọng để gỡ lỗi, chỉnh sửa và biên dịch mã trong môi trường.

Có rất nhiều các module Arduino như Arduino Uno, Arduino Mega, Arduino Leonardo, Arduino Micro và nhiều module khác.

Mỗi module chứa một bộ vi điều khiển trên bo mạch được lập trình và chấp nhận thông tin dưới dạng mã.

Mã chính, còn được gọi là sketch, được tạo trên nền tảng IDE sẽ tạo ra một file Hex, sau đó được chuyển và tải lên trong bộ điều khiển trên bo.

Môi trường IDE chủ yếu chứa hai phần cơ bản: Trình chỉnh sửa và Trình biên dịch, phần đầu sử dụng để viết mã được yêu cầu và phần sau được sử dụng để biên dịch và tải mã lên module Arduino.

Môi trường này hỗ trợ cả ngôn ngữ C và C ++. (Fe)

2.2.2 MIT App Inventor

MIT App Inventor là một nền tảng trực quan giúp tạo ứng dụng di động một cách dễ dàng mà không cần lập trình phức tạp. Với MIT App Inventor, người dùng có thể thiết kế giao diện và lập trình ứng dụng bằng cách kéo thả các khối lệnh, phù hợp cho cả người mới bắt đầu và lập trình viên không chuyên.

Nền tảng này hỗ trợ kết nối với nhiều loại vi điều khiển như ESP32, Arduino, ESP8266, HC-06 Bluetooth Module thông qua Wi-Fi hoặc Bluetooth, giúp xây dựng các ứng dụng IoT như giám sát cảm biến, điều khiển thiết bị thông minh và thu thập dữ liệu theo thời gian thực một cách linh hoạt và dễ dàng.

CHƯƠNG 3: PHƯƠNG PHÁP THỰC HIỆN VÀ THỰC NGHIỆM

3.1 Chuẩn bị vật liệu và cách hệ thống hoạt động

3.1.1 Chuẩn bị vật liệu

3.1.1.1 Arduino Uno R3

Arduino Uno R3 là một bảng mạch vi điều khiển nguồn mở dựa trên vi điều khiển Microchip ATmega328. Bảng mạch được trang bị các bộ chân đầu vào/ đầu ra Digital và Analog có thể giao tiếp với các bảng mạch mở rộng khác nhau. Mạch Arduino Uno thích hợp cho những bạn mới tiếp cận và đam mê về điện tử, lập trình… Dựa trên nền tảng mở do Arduino.cc cung cấp các bạn dễ dàng xây dựng cho mình một dự án nhanh nhất (lập trình Robot, xe tự hành, điều khiển bật tắt led…). (Arduinokit, 2020)

Hình 3.1. Arduino Uno R3

– Thông số kĩ thuật Arduino Uno R3

Bảng 3-1 Bảng thông số kĩ thuật Arduino Uno R3

Thông số	**Giá trị**
Vi điều khiển (MCU)	ATmega328P
Điện áp hoạt động	5V

Điện áp đầu vào	7 – 12V (khuyến nghị) 6 – 20V (giới hạn)
Số chân Digital I/O (DIO)	14 (trong đó có 6 chân PWM)
Số chân Analog Input	6
Dòng điện tối đa trên mỗi chân I/O	40 mA
Bộ nhớ Flash	32 KB (trong đó 0.5 KB dùng cho bootloader)
SRAM	2 KB
EEPROM	1 KB
Tốc độ xung nhịp	16 MHz
Giao tiếp	UART, SPI, I2C
Cổng USB	Type-B
Bộ nguồn	Jack DC hoặc cổng USB
Kích thước	68.6 mm × 53.4 mm
Trọng lượng	Khoảng 25g

3.1.1.2 Cảm biến HC-SR04

HC-SR04 là cảm biến siêu âm chủ yếu được sử dụng để xác định khoảng cách của đối tượng mục tiêu. Nó đo khoảng cách chính xác bằng công nghệ không tiếp xúc, tức là không có tiếp xúc vật lý giữa cảm biến và vật thể.

Bộ phát và bộ thu là hai bộ phận chính của cảm biến, bộ phát chuyển đổi tín hiệu điện thành sóng siêu âm, còn bộ thu chuyển đổi tín hiệu siêu âm đó trở lại thành tín hiệu điện. Các sóng siêu âm này là các tín hiệu âm thanh có thể được đo và hiển thị ở đầu nhận.

Nó cung cấp các chi tiết đo lường chính xác và đi kèm với độ phân giải khoảng 3mm, có thể có sự khác biệt nhỏ về khoảng cách tính toán từ đối tượng và khoảng cách thực tế. (dientutuonglai, HC-SR04 là gì)

Hình 3.1 Cảm biến HC-SR04

– Thông số kĩ thuật HC-SR04:

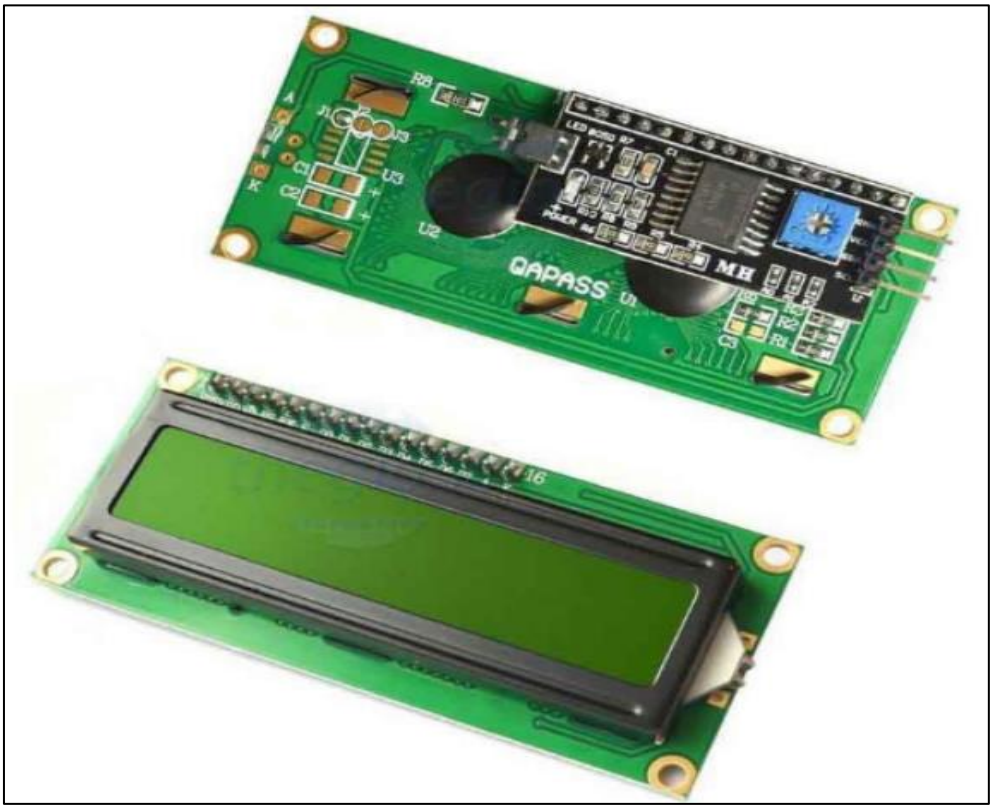
Bảng 3-2 Thông số kĩ thuật của HC-SR04

Thông số	**Giá trị**
Điện áp hoạt động	5V DC
Dòng tiêu thụ	15 mA
Tần số sóng siêu âm	40 kHz
Khoảng cách đo	2 cm – 400 cm
Độ chính xác	±3 mm
Góc phát hiện	~15°
Chu kỳ đo tối thiểu	60ms
Tín hiệu đầu vào	Xung trigger TTL (10µs)
Tín hiệu đầu ra	Xung Echo TTL

Kích thước	45mm x 20mm x 15mm

3.1.1.3 Màn hình LCD I2C

Màn hình LCD I2C này là một thiết bị 16×2 có nghĩa là nó có thể hiển thị 16 cột bằng hai hàng ký tự. Các ký tự là chữ và số, nhưng bạn có thể tạo các ký tự tùy chỉnh cho đồ họa cơ bản, biểu đồ thanh loại đó. Màn hình LCD có loại bộ điều khiển hd44780 thông thường và nó cũng có mạch I2C được kết nối với nó giúp dễ dàng kết nối với bảng Arduino. Màn hình LCD 16X2 không có mạch I2C có mười sáu chân. (Mescu, 2023)



Hình 3.3: Màn hình LCD I2C

– Thông số kĩ thuật màn hình LCD I2C (16x2):

Bảng 3-3 Thông số kĩ thuật của màn hình LCD I2C

Thông số	**Giá trị**
Loại màn hình	LCD 16x2 (16 cột, 2 hàng)

Giao tiếp	I2C
IC điều khiển chính	HD44780
Module I2C sử dụng	PCF8574
Điện áp hoạt động	5V
Địa chỉ I2C mặc định	0x27 hoặc 0x3F (tùy module)
Dòng điện tiêu thụ	Khoảng 20mA
Độ tương phản	Điều chỉnh bằng biến trở
Kích thước màn hình	~80mm x 36mm
Màu nền	Xanh lá
Đèn nền (Backlight)	Có thể bật/tắt dựa vào chỉnh biến trở và ghi code trong Arduino IDE

3.1.2 Cách hệ thống hoạt động

Hệ thống sử dụng cảm biến siêu âm **HC-SR04** để đo khoảng cách bằng nguyên lý phát và nhận sóng siêu âm. Cảm biến phát ra một xung siêu âm, sau đó đo thời gian sóng phản xạ từ vật cản quay lại. Dựa vào thời gian này, hệ thống tính toán khoảng cách bằng công thức:

Dữ liệu đo được có thể hiển thị trên màn hình LCD hoặc gửi đến máy tính để giám sát theo thời gian thực. Hệ thống phù hợp để đo khoảng cách trong không khí, hỗ trợ ứng dụng trong robot tránh vật cản, đo mức chất lỏng, giám sát không gian,...

3.2 Quy trình thực hiện

3.2.1 Thiết lập phần cứng

1: Kết nối Arduino Uno R3 với breadboard:

– Cắm Arduino UNo R3 vào giữa breadboard, đảm bảo mỗi chân được kết nối với một hàng riêng biệt.

2: Kết nối cảm biến HC-SR04 với Arduino:

– VCC đến 5V của Arduino
– GND đến GND của Arduino
– Trigger đến chân số 2 của Arduino
– Echo đến chân số 3 của Arduino

3: Kết nối màn hình LCD_I2C với Arduino:

– GND đến GND của Arduino
– VCC đến 5V của Arduino
– SDA đến chân A4 của Arduino
– SCL đến chân A5 của Arduino

4: Kiểm tra kỹ lưỡng tất cả các kết nối để đảm bảo không có lỗi hay đoản mạch.

3.2.2 Lập trình Arduino

1: Cài đặt Arduino IDE

2: Mở Arduino IDE và tạo một sketch mới.

3: Thêm thư viện LiquidCrystal_I2C bằng cách chọn Sketch > Include Library > LiquidCrystal_I2C.

4: Sao chép và dán đoạn mã sau vào sketch:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

const int trigPin = 2;   // Chân Trigger của cảm biến siêu âm
const int echoPin = 3;   // Chân Echo của cảm biến siêu âm
LiquidCrystal_I2C lcd(0x27, 16, 2);  // LCD địa chỉ 0x27, kích thước 16x2
```

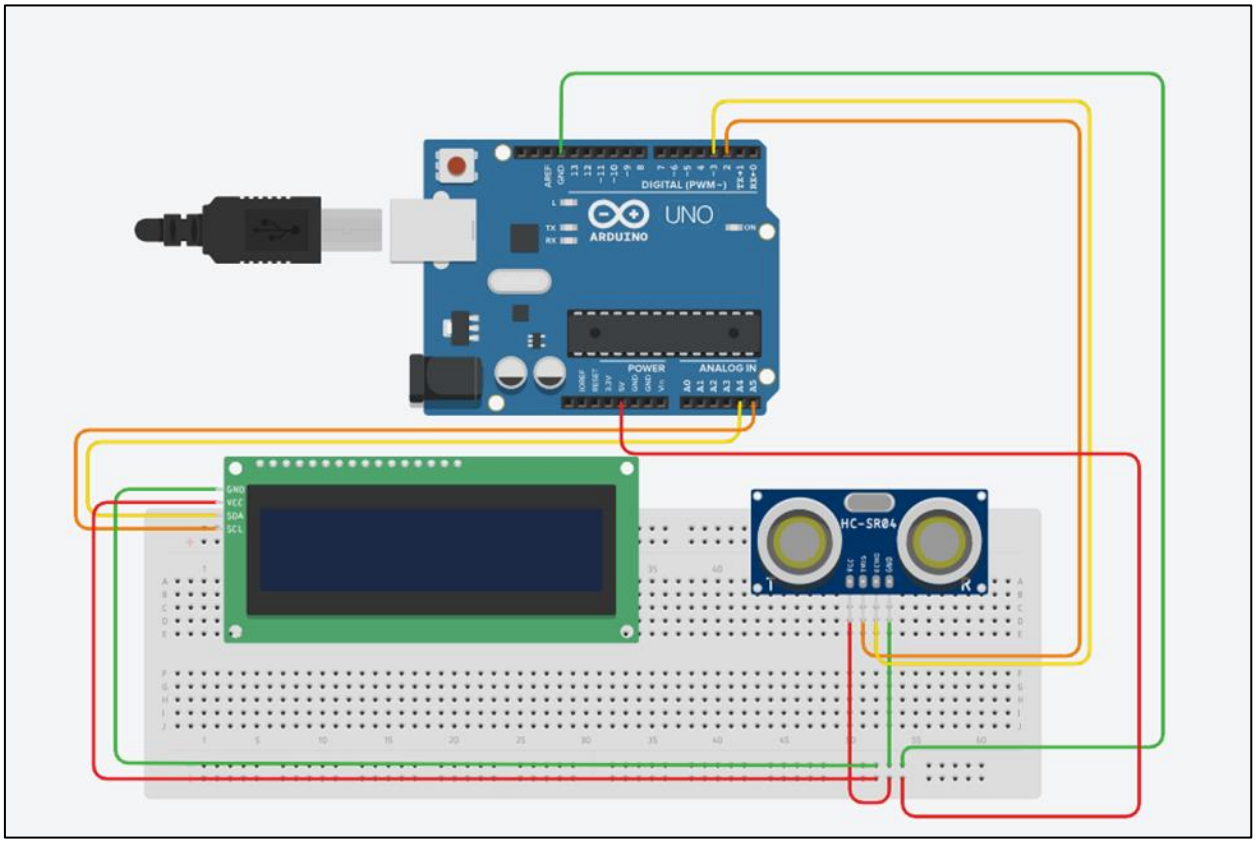
```
void setup()
{
   lcd.init();
   lcd.backlight();  // Bật đèn nền LCD
   lcd.setCursor(0, 0);
   lcd.print("Distance:");
   pinMode(trigPin, OUTPUT);
   pinMode(echoPin, INPUT);
}
void loop()
{
   long duration;
   float distance;

   // Gửi tín hiệu siêu âm
   digitalWrite(trigPin, LOW);
   delayMicroseconds(2);
   digitalWrite(trigPin, HIGH);
   delayMicroseconds(10);
   digitalWrite(trigPin, LOW);
   // Nhận tín hiệu phản hồi
   duration = pulseIn(echoPin, HIGH);
   distance = (duration / 2) / 29.1;  // Chuyển thời gian thành khoảng cách (cm)
```

```
    // Hiển thị lên LCD
    lcd.setCursor(0, 1);
    lcd.print("                "); // Xóa dòng cũ
    lcd.setCursor(0, 1);
    lcd.print(distance);
    lcd.print(" cm");
    delay(500);  // Đợi 0.5 giây trước khi đo lại
}
```

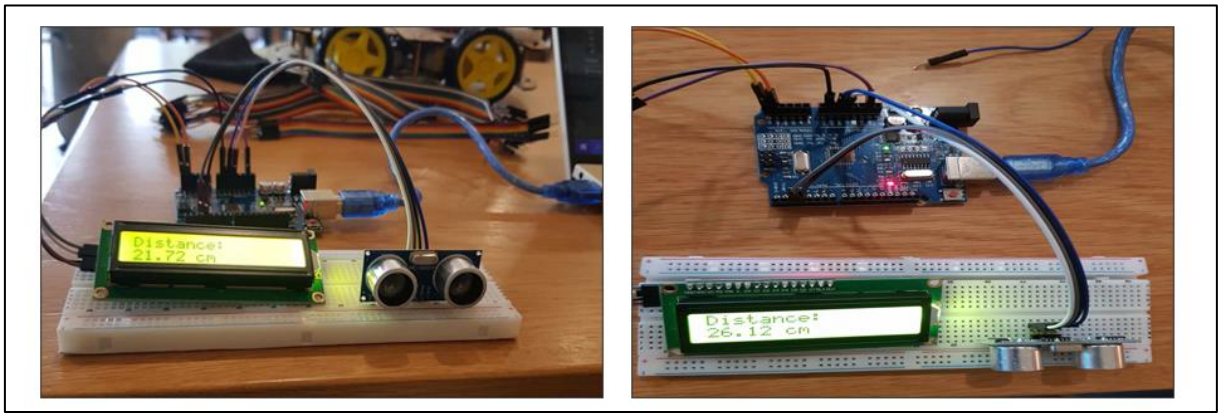
5: Kết nối Arduino với máy tính qua cáp USB.

6: Chọn board Arduino và cổng COM phù hợp trong Arduino IDE.

7: Tải sketch lên Arduino bằng cách nhấn nút "Upload".



Hình 3.2 Sơ đồ mạch sau khi hoàn thành mục 2 trên Tinkercad



Hình 3.3 Hình ảnh thực tế sau khi hoàn thành mục 2

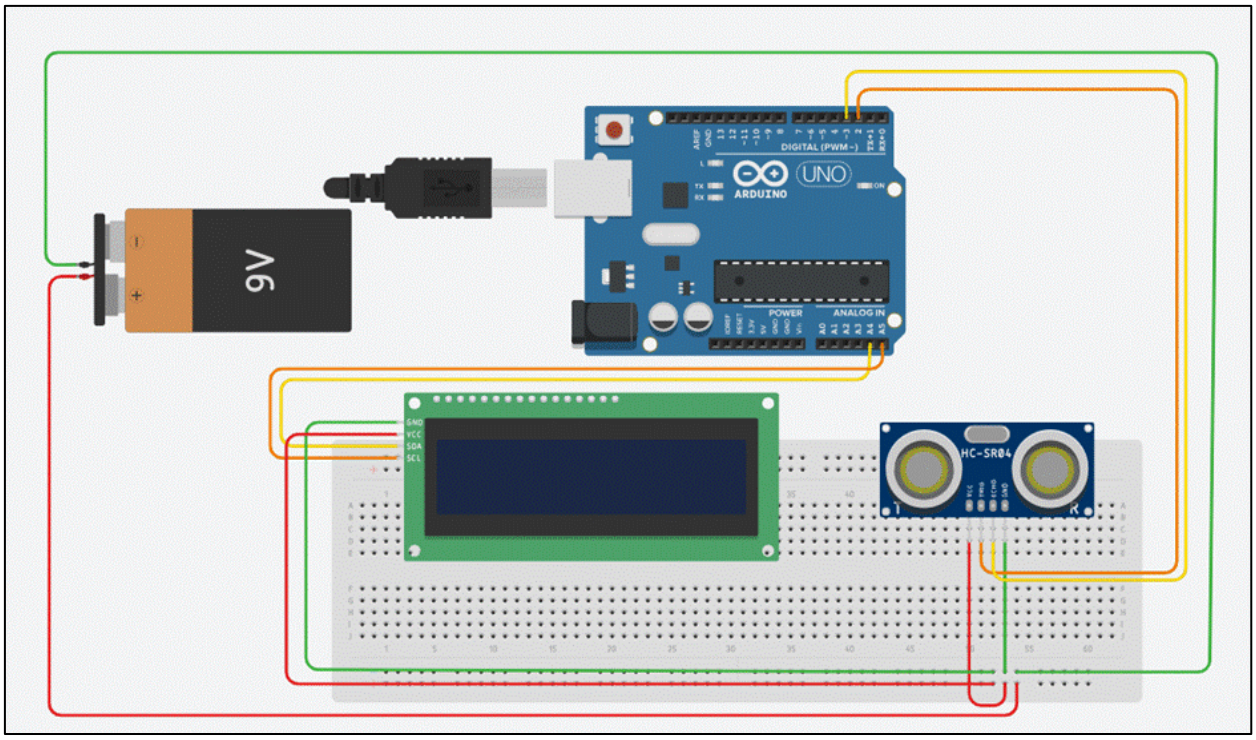
3.2.3 Làm cho thiết bị di động

1: Ngắt kết nối Arduino khỏi máy tính.

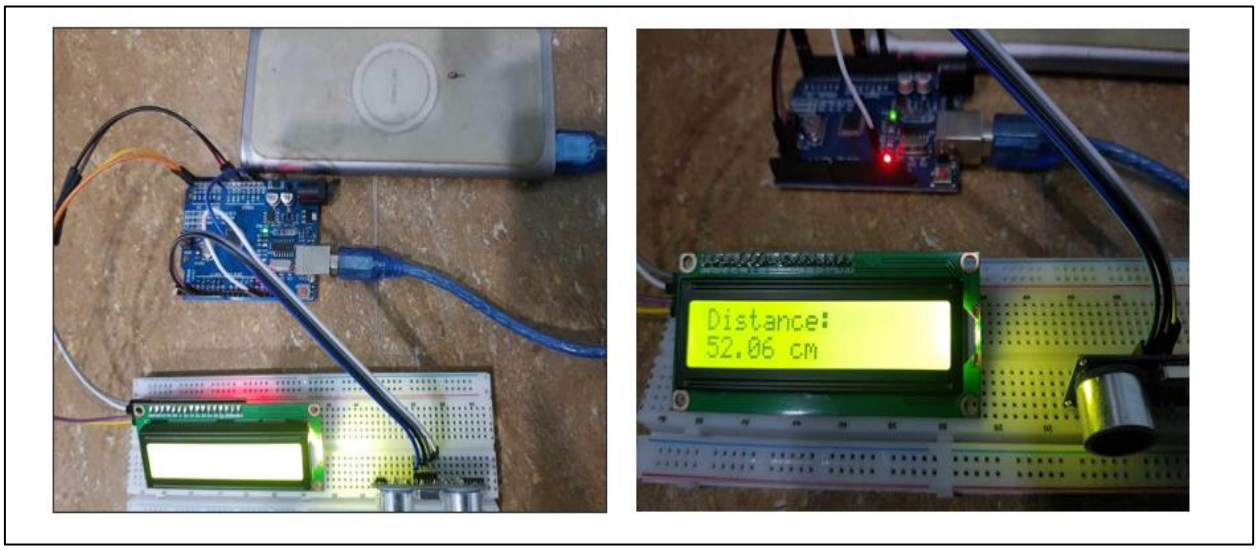
2: Kết nối pin dự phòng USB với Arduino thông qua cổng micro-USB.

3: (Tùy chọn) Đặt toàn bộ mạch vào hộp nhựa hoặc vỏ bảo vệ, chỉ để lộ cảm biến siêu âm và màn hình LCD.

4: Sử dụng băng dính hai mặt hoặc keo nóng để cố định các thành phần vào vỏ bảo vệ.



Hình 3.4 Sơ đồ mạch sau khi hoàn thành mục 3 trên Tinkercad



Hình 3.5 Hình ảnh thực tế sau khi hoàn thành mục 3

3.3 Thực nghiệm

Bảng 3-4 Thực nghiệm đo khoảng cách

Khoảng cách thực tế (cm)		Khoảng cách đo được (cm)	Sai số (%)
10 cm		10.14 cm	1.4
30 cm		30.62 cm	2.06
50 cm		50.55 cm	1.1
100 cm		100.79 cm	0.79

Chú thích:

+ Công thức sai số:

$$\text{Sai số (\%)} = \frac{|Khoảng\ cách\ thực\ tế - Khoảng\ cách\ đo\ được|}{Khoảng\ cách\ thực\ tế} \times 100$$

Các kết luận rút ra được sau thí nghiệm:

- **Độ chính xác của cảm biến:**
  + Cảm biến siêu âm HC-SR04 có độ chính xác khá cao khi đo khoảng cách trong điều kiện lý tưởng.
  + Tuy nhiên, vẫn có sai số nhỏ (~1-2 cm) do yếu tố môi trường hoặc góc đặt vật cản.
- **Sai số trong đo lường:**
  + Sai số có thể xuất hiện do nhiễu từ môi trường hoặc do cách đặt cảm biến.
  + Khi so sánh với thước đo thực tế, độ lệch khoảng **1-2 cm** là chấp nhận được trong nhiều ứng dụng.
- **Ảnh hưởng của môi trường:**

+ **Nhiệt độ và độ ẩm không khí** ảnh hưởng đến vận tốc âm thanh (~340 m/s), dẫn đến kết quả dao động nhẹ.
    - Nếu nhiệt độ tăng, vận tốc âm thanh tăng ⇒ khoảng cách đo được có thể nhỏ hơn thực tế.
    - Nếu độ ẩm cao, không khí loãng hơn ⇒ sóng âm bị hấp thụ nhiều hơn, có thể làm giảm độ chính xác.

– **Độ phẳng và vị trí của vật cản**

+ Để có kết quả đo chính xác, vật cản cần được đặt **vuông góc với cảm biến**.
+ Nếu vật cản có bề mặt nghiêng hoặc không phản xạ tốt (như vải, xốp), tín hiệu phản hồi có thể yếu hoặc không chính xác.

CHƯƠNG 4: MỞ RỘNG

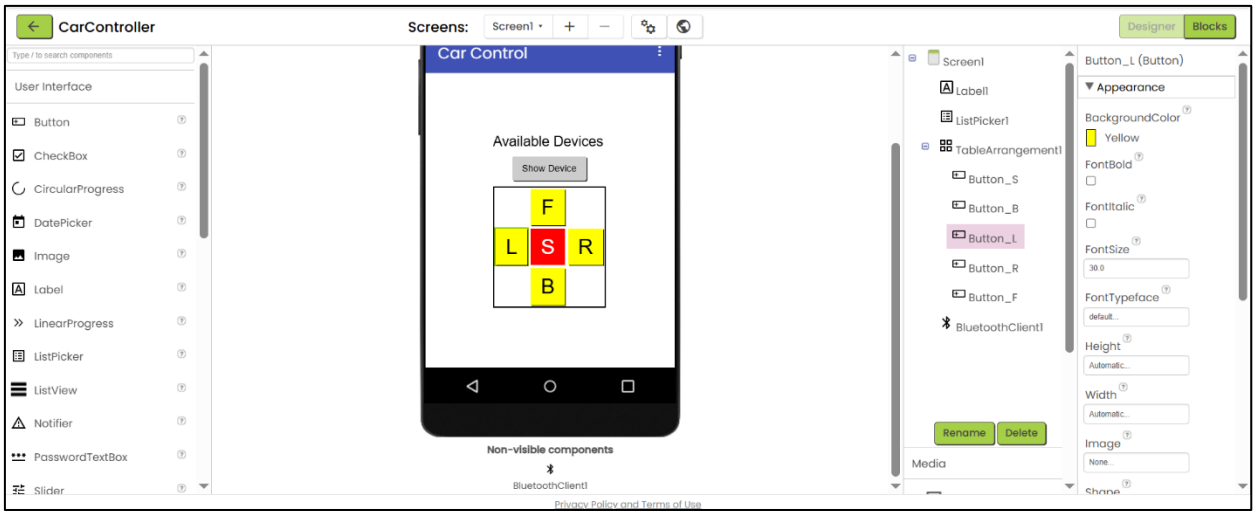
4.1 Sử dụng Bluetooth module để gửi dữ liệu đến điện thoại thông minh

Phát triển thành xe điều khiển chứa cảm biến khoảng cách di động, áp dụng mở rộng 1 đi kèm app điều khiển (làm từ inventor) và thay thế từ dùng pin sạc dự phòng sang pin cơ.

Khó khăn là kết nối pin (do kết nối pin cơ) => Mua thêm động cơ rồi lắp vào với pin.

- Vật liệu: 1 khung xe điều khiển 2 tầng với 4 bánh xe, 1 động cơ L298N, 1 cảm biến hc-06 bluetooth
- **Hướng dẫn gắn:**

+ Động cơ L298N:
  - ENA gắn với chân 6 arduino
  - IN1 gắn với chân 7 arduino
  - IN2 gắn với chân 8 arduino
  - IN3 gắn với chân 9 arduino
  - IN4 gắn với chân 12 arduino
  - ENB gắn với chân 5 arduino
+ Hc-06 bluetooth:
  - VCC gắn với 5V của arduino
  - GND gắn với GND của arduino
  - RX gắn với chân 11 của arduino
  - TX gắn với chân 10 của arduino



Hình 4.1 Giao diện app MIT Inventor

– Code bổ sung:

+ Arduino:

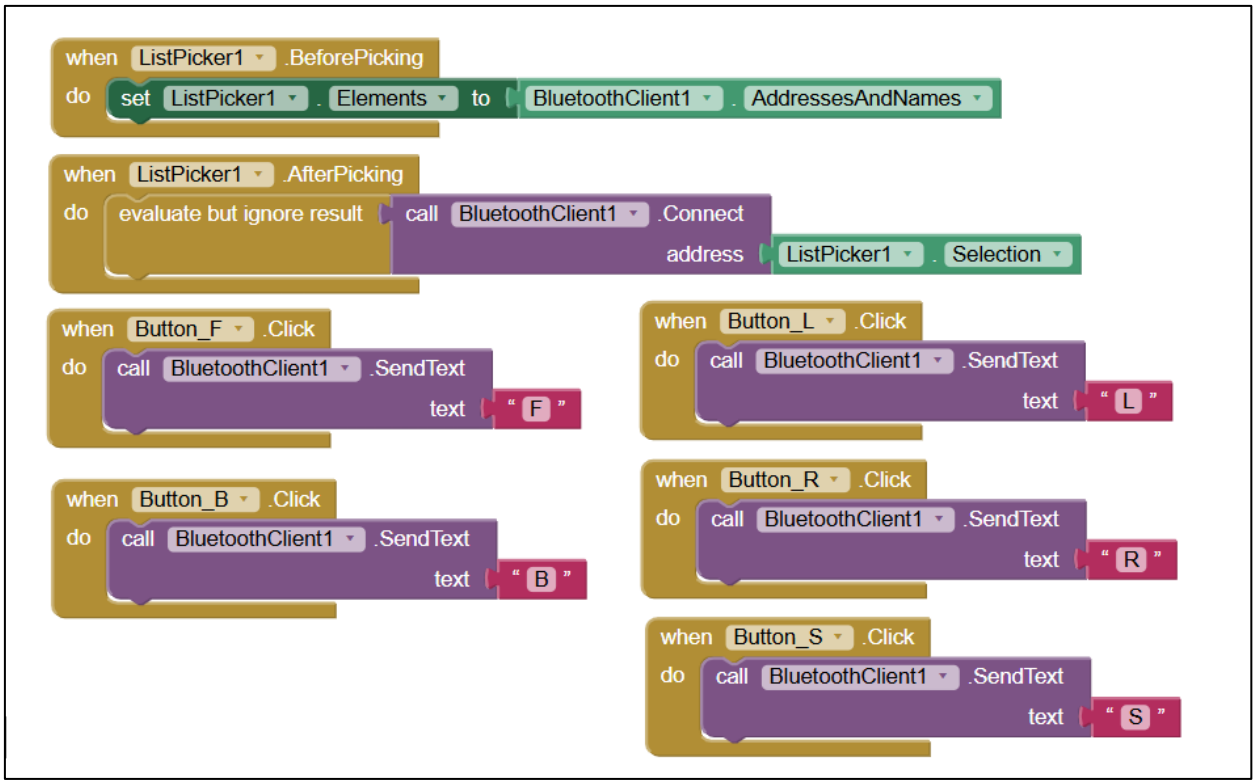
```
#include <SoftwareSerial.h>

const int in1 = 7;

const int in2 = 8;

const int in3 = 9;

const int in4 = 12;

const int ena = 6;

const int enb = 5;

const int bluetoothRx = 11;

const int bluetoothTx = 10;


SoftwareSerial bluetooth(bluetoothRx, bluetoothTx);

void setup()

{

   bluetooth.begin(9600);  // Bật Bluetooth
```

```
    Serial.begin(9600);     // Bật Serial cho debug
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(in1, OUTPUT);
    pinMode(in2, OUTPUT);
    pinMode(in3, OUTPUT);
    pinMode(in4, OUTPUT);
    pinMode(ena, OUTPUT);
    pinMode(enb, OUTPUT);
}
void loop()
{
    // Nhận dữ liệu từ Bluetooth
    if (bluetooth.available())
    {
        char command = bluetooth.read();
        Serial.print("Bluetooth nhận: ");
        Serial.println(command);
        controlCar(command);
    }
}
void controlCar(char command)
{
    switch (command)
```

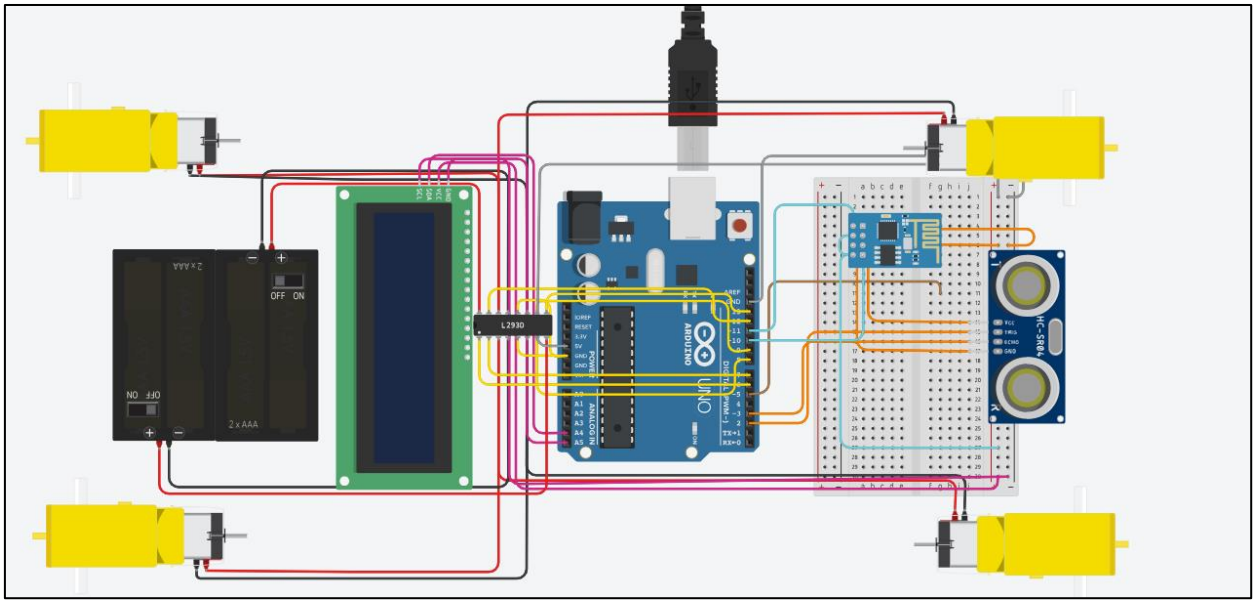
```
{
  case 'F': // Tiến
    analogWrite(ena, 255);
    analogWrite(enb, 255);
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
    break;
  case 'B': // Lùi
    analogWrite(ena, 255);
    analogWrite(enb, 255);
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    break;
  case 'L': // Quẹo trái
    analogWrite(ena, 150);
    analogWrite(enb, 255);
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
```

```
            break;
        case 'R': // Quẹo phải
            analogWrite(ena, 255);
            analogWrite(enb, 150);
            digitalWrite(in1, HIGH);
            digitalWrite(in2, LOW);
            digitalWrite(in3, HIGH);
            digitalWrite(in4, LOW);
            break;
        case 'S': // Dừng
            analogWrite(ena, 0);
            analogWrite(enb, 0);
            digitalWrite(in1, LOW);
            digitalWrite(in2, LOW);
            digitalWrite(in3, LOW);
            digitalWrite(in4, LOW);
            break;
    }
}
```

+ Inventor:



Hình 4.2 Sơ đồ mạch sau khi hoàn thành mở rộng 1 trên Inventor



Hình 4.3 Hình ảnh thực tế sau khi hoàn thành mở rộng 1

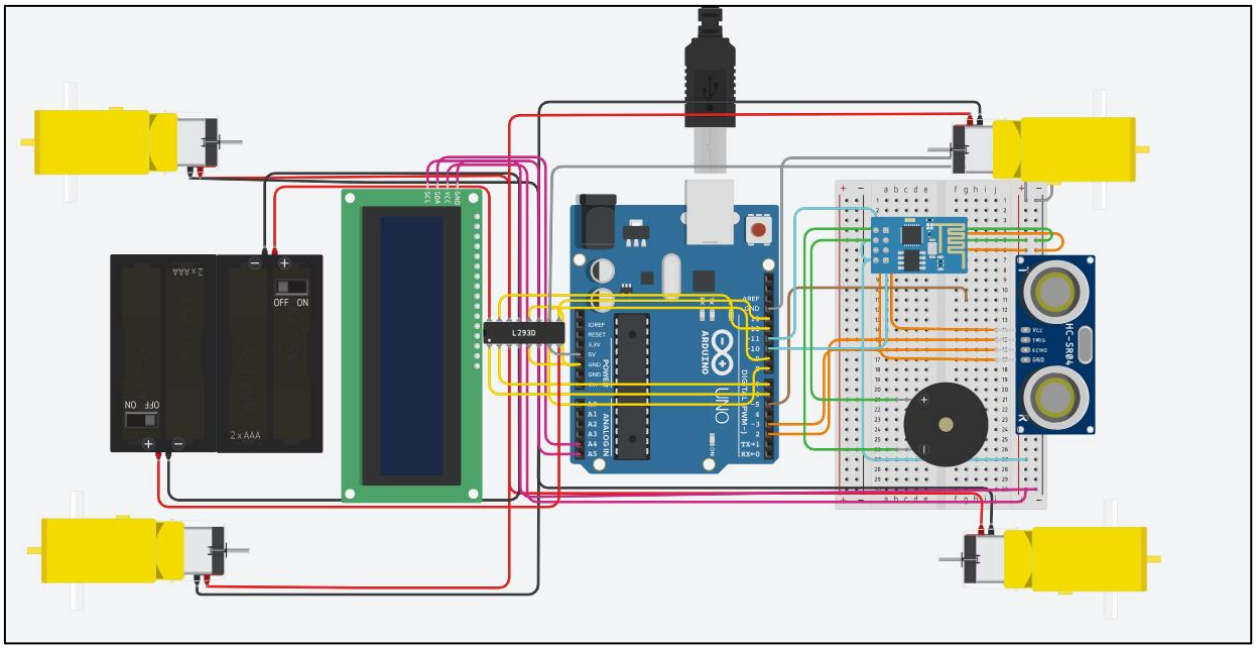
4.2 Thêm một còi báo động khi khoảng cách nhỏ hơn một giá trị cụ thể

- Vật liệu: còi buzzer
- Hướng dẫn gắn:

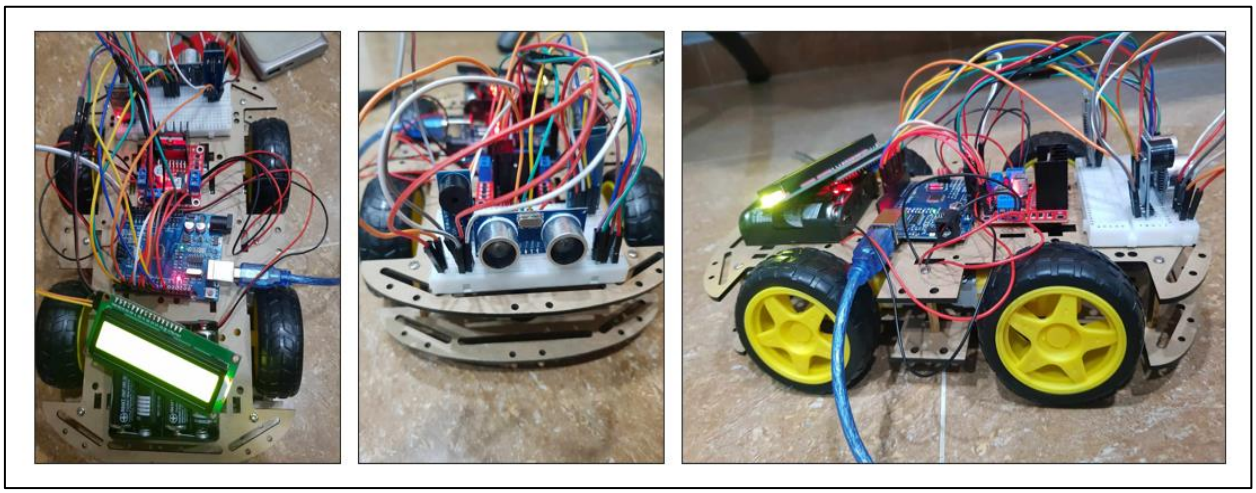
+ GND nối với chân GND của arduino
+ VCC nối với 5V
+ I/O nối với chân 4 của Arduino

– Code bổ sung:

```
#include <SoftwareSerial.h>
const int buzzerPin = 4;
}
void loop()
{
  // Buzzer
  if (distance > 5.0)
  {
    digitalWrite(buzzerPin, LOW);
  }
  else
  {
    digitalWrite(buzzerPin, HIGH);
  }
}
```



Hình 4.4 Sơ đồ mạch sau khi hoàn thành mở rộng 2 trên Tinkercad



Hình 4.5 Hình ảnh thực tế sau khi hoàn thành mở rộng 2

4.3 Thêm một nút để chuyển đổi giữa đơn vị đo cm và inch

– Vật liệu: nút bấm

– Hướng dẫn gắn:

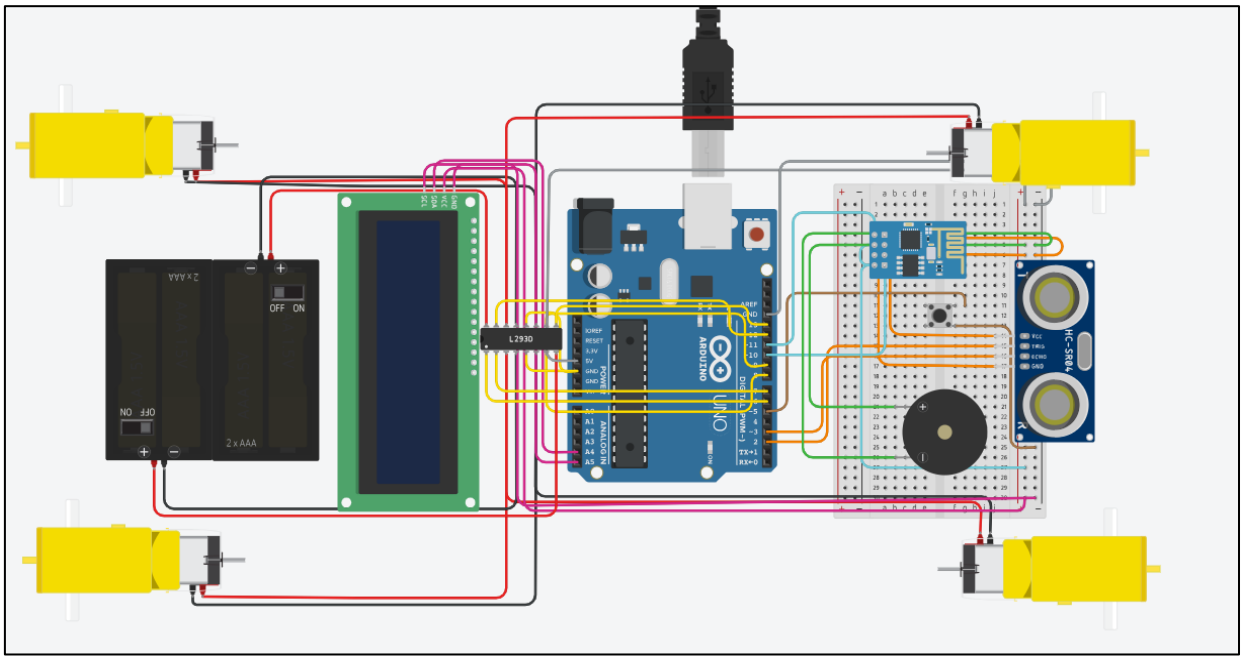
+ Chọn đôi cắm:

- 1 đầu nối với GND của arduino
- 1 đầu nối với chân 13 của Arduino

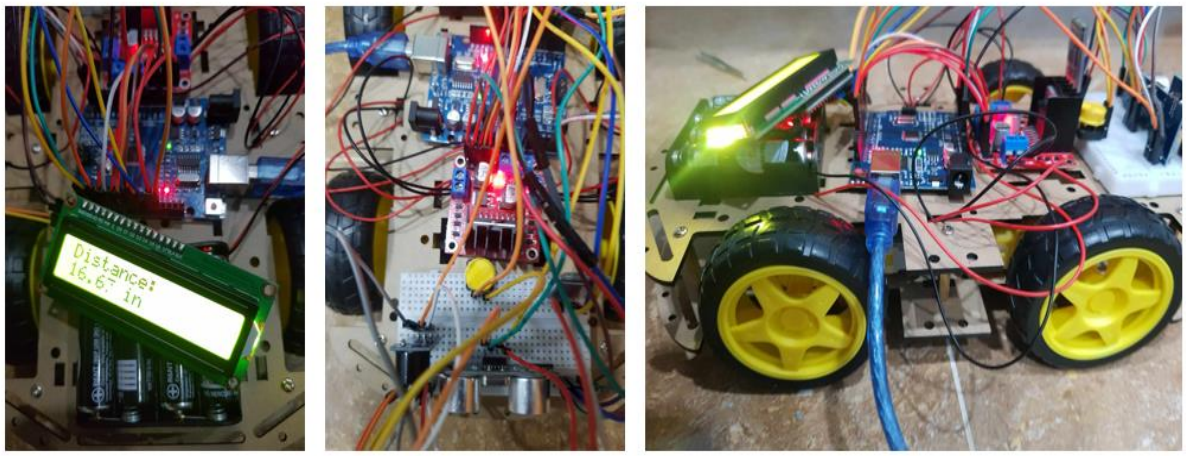
– Code bổ sung:

```
const int buttonPin = 13; // Nút bấm đổi đơn vị
bool isCm = true; // Biến kiểm tra đơn vị hiển thị
void setup()
{
  pinMode(buttonPin, INPUT_PULLUP); // Nút bấm với pull-up nội bộ
}
void loop()
{
  // Kiểm tra nút bấm để đổi đơn vị
  if (digitalRead(buttonPin) == LOW)
  {
    delay(200); // Chống dội phím
    isCm = !isCm;
  }
  lcd.setCursor(0, 1);
  lcd.print("                "); // Xóa dòng cũ
  lcd.setCursor(0, 1);
  if (isCm)
  {
    lcd.print(distance);
    lcd.print(" cm");
  }
  else
  {
```

```
        lcd.print(distance * 0.393701);
        lcd.print(" in");
    }
}
```



Hình 4.6 Sơ đồ mạch sau khi hoàn thành mở rộng 3 trên Tinkercad

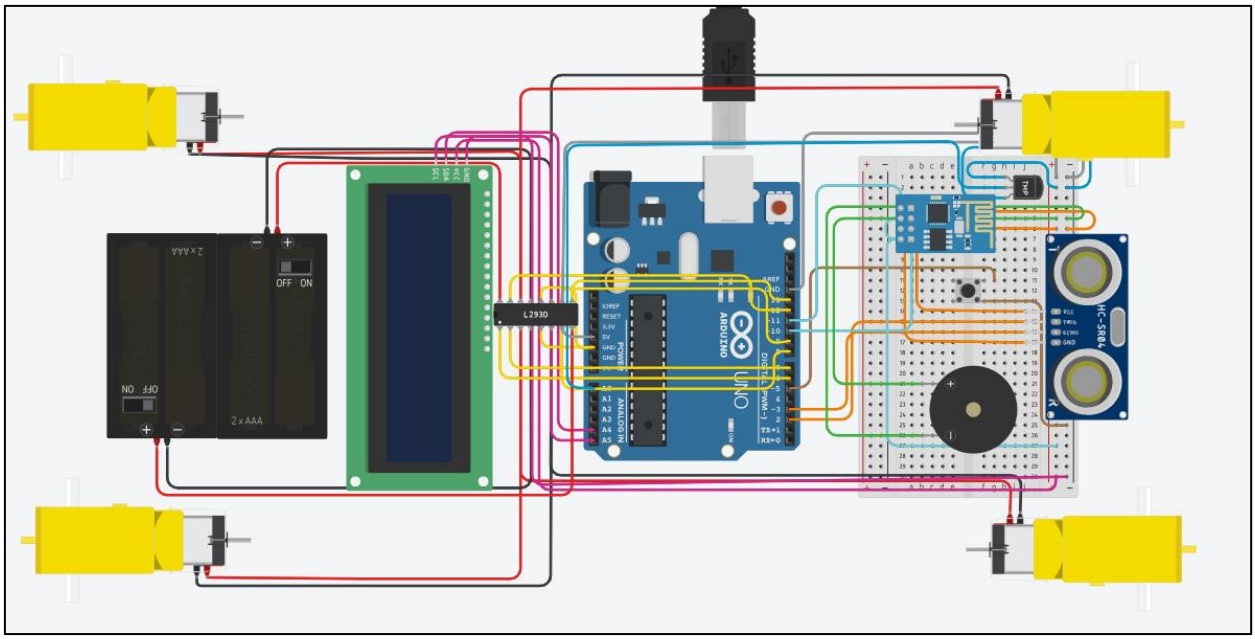


Hình 4.7 Hình ảnh thực tế sau khi hoàn thành mở rộng 3

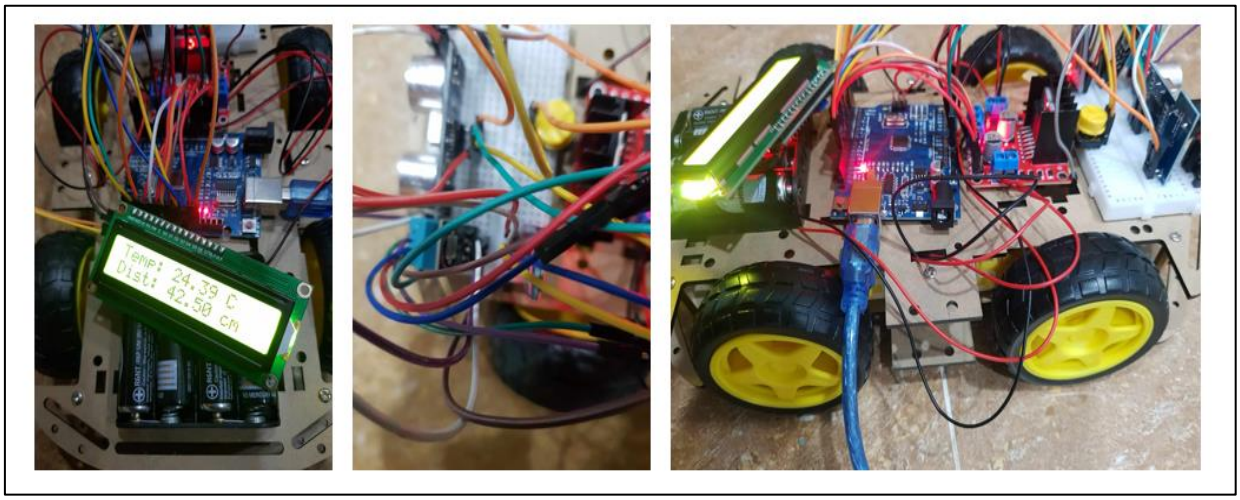
4.4 Tích hợp một cảm biến nhiệt độ để hiệu chỉnh kết quả đo dựa trên nhiệt độ môi trường

- Vật liệu: cảm biến nhiệt độ
- Hướng dẫn gắn:
+ GND gắn với GND của arduino
+ VCC gắn với 5V của arduino
+ chân còn lại cắm chân A0 của Arduino
- Code bổ sung:

```
const int tempSensor = A0; // Cảm biến nhiệt độ LM35
void setup()
{
  pinMode(tempSensor, INPUT); // Cảm biến nhiệt độ
}
void loop()
{
  float tempVoltage = analogRead(tempSensor) * (5.0 / 1023.0);
  // LM35: 10mV = 1°C
  float temperature = (((tempVoltage * 5) / 1023 * 10) * 100.0);
  // Tốc độ âm thanh trong không khí thay đổi theo nhiệt độ
  float speedOfSound = 331.3 + (0.606 * temperature); // m/s
  speedOfSound = speedOfSound * 100 / 1e6; // đổi sang cm/us
  distance = (duration / 2.0) * speedOfSound;  // Chuyển thời gian thành
khoảng cách (cm)
}
```



Hình 4.8 Sơ đồ mạch sau khi hoàn thành mở rộng 4 trên Tinkercad



Hình 4.9 Hình ảnh thực tế sau khi hoàn thành mở rộng 4

CHƯƠNG 5: KẾT LUẬN

5.1 Những kiến thức đã học được:

Trong quá trình thực hiện đề tài, nhóm đã tiếp thu nhiều kiến thức thực tế quan trọng, bao gồm:

- **Nguyên lý hoạt động của cảm biến siêu âm:**
  + Cảm biến siêu âm (như HC-SR04) hoạt động bằng cách phát sóng siêu âm và nhận tín hiệu phản hồi sau khi va chạm vào vật cản.
  + Dựa vào thời gian truyền và phản xạ, ta tính được khoảng cách với công thức:
  + Vận tốc âm thanh thường lấy giá trị trung bình 340 m/s, nhưng có thể thay đổi theo nhiệt độ và độ ẩm không khí.
- **Cách sử dụng cảm biến siêu âm:**
  + Học cách kết nối cảm biến HC-SR04 với vi điều khiển như Arduino, ESP32.
  + Gửi tín hiệu kích hoạt (trigger) và đo thời gian phản hồi (echo).
  + Xử lý tín hiệu để tính toán và hiển thị kết quả chính xác.
- **Lập trình vi điều khiển để đọc dữ liệu cảm biến:**
  + Viết code trên Arduino IDE để điều khiển cảm biến đo khoảng cách.
  + Đọc tín hiệu từ chân Echo, tính toán và xuất kết quả.
  + Gửi dữ liệu đo khoảng cách đến màn hình LCD I2C, OLED hoặc máy tính.
- **Xử lý tín hiệu để tăng độ chính xác:**
  + Lọc nhiễu: Tránh sai số do nhiễu môi trường hoặc đo sai tín hiệu.
  + Hiệu chỉnh dữ liệu: Trung bình nhiều lần đo để tăng độ ổn định.
  + Cải thiện phần mềm: Thêm thuật toán để loại bỏ giá trị bất thường.
- **Ảnh hưởng của môi trường và cách khắc phục:**
  + Nhiệt độ, độ ẩm không khí: Ảnh hưởng đến vận tốc âm thanh, cần hiệu chỉnh nếu yêu cầu độ chính xác cao.
  + Bề mặt vật cản: Vật có bề mặt nhám, góc nghiêng sẽ làm sóng siêu âm phản xạ không đúng hướng, gây sai số.
  + Khoảng cách đo: HC-SR04 có tầm đo từ 2 cm – 400 cm, cần chọn cảm biến phù hợp nếu cần đo xa hơn.

- **Thiết kế mạch và truyền dữ liệu:**
  + Cách thiết kế mạch để đảm bảo tín hiệu ổn định.
  + Kết nối cảm biến với Wi-Fi/Bluetooth để truyền dữ liệu từ xa.
  + Gửi kết quả đo đến web server hoặc ứng dụng IoT.
- **Kiểm thử và tối ưu hệ thống:**
  + So sánh kết quả đo với thước đo thực tế để xác định sai số.
  + Điều chỉnh góc đặt cảm biến để đo chính xác hơn.
  + Kiểm tra hoạt động trong các môi trường khác nhau để tối ưu hiệu suất.
- **Ứng dụng thực tế của cảm biến siêu âm:**
  + Robot tránh vật cản: Giúp robot di chuyển an toàn.
  + Xe tự hành: Hỗ trợ phát hiện vật cản để tránh va chạm.
  + An ninh giám sát: Xác định khoảng cách đối tượng để phát hiện xâm nhập.
  + Đo mực nước: Sử dụng cảm biến để giám sát mực nước trong bể chứa.
  + Hệ thống hỗ trợ đỗ xe: Đo khoảng cách từ xe đến chướng ngại vật.

5.2 Những điều đã làm được:

- **Xây dựng nguyên mẫu hoạt động:** Hệ thống khoảng cách bằng cảm biến HC-SR04 đã hoạt động tốt trong điều kiện thử nghiệm.
- **Kết nối và hiển thị kết quả trên màn hình LCD I2C:** Hệ thống có khả năng đo dựa vào cảm biến HC-SR04 và hiển thị lên màn hình LCD I2C với nguồn là máy tính hoặc pin dự phòng.
- **Độ chính xác chấp nhận được:** Hệ thống có khả năng đo khoảng cách với sai số nhỏ trong môi trường thí nghiệm tiêu chuẩn.
- **Phát triển thành mô hình lớn**: Phát triển thành xe điều khiển sử dụng bluetooth và MIT App Inventor
- **Cảnh báo khoảng cách quá gần:** Có loa cảnh báo nếu khoảng cách quá nhỏ (dưới 5cm)
- **Thuận tiện về xem độ đo theo đơn vị khác nhau**: Có nút nhấn chuyển từ cm sang inch.
- **Điều chỉnh khoảng cách dựa vào nhiệt độ**: sử dụng cảm biến nhiệt độ để hiệu chỉnh khoảng cách của xe.

5.3 Những điều chưa làm được:

- **Độ trễ trong truyền dữ liệu:** Khi gửi dữ liệu từ cảm biến **HC-SR04** đến điện thoại thông qua Arduino, **Bluetooth hoặc Wi-Fi**, vẫn tồn tại một độ trễ nhỏ, có thể gây ảnh hưởng đến phản hồi thời gian thực.
- **Hiển thị dữ liệu trên ứng dụng còn hạn chế:** Giao diện ứng dụng trên điện thoại chưa được tối ưu, còn đơn giản, thiếu các biểu đồ hoặc cách trực quan hóa dữ liệu giúp người dùng dễ theo dõi hơn.

5.4 Những khó khăn đã gặp:

- **Đầu dây lỏng:** Trong quá trình kết nối các linh kiện, một số dây bị lỏng dẫn đến tín hiệu truyền không ổn định.
  → Giải pháp: Thay thế bằng loại dây có lõi cứng hơn để đảm bảo kết nối chắc chắn và ổn định.
- **Arduino bị tắt khi cắm nguồn 5V:** Khi cấp nguồn 5V, Arduino bị mất điện hoặc hoạt động không ổn định.
  → Giải pháp: Điều chỉnh biến trở phù hợp để ổn định nguồn điện cấp cho mạch.
- **LCD_I2C không hiển thị:** Màn hình LCD_I2C không hiển thị nội dung dù đã đấu dây đúng.
  → Giải pháp: Kiểm tra và điều chỉnh biến trở tích hợp trên module LCD_I2C để đảm bảo độ tương phản phù hợp.
- **Sắp xếp mạch hợp lý:** Khi lắp ráp, các dây nối và linh kiện dễ bị chồng chéo, gây khó khăn trong việc kiểm tra và bảo trì.
  → Giải pháp: Vẽ sơ đồ mạch trước khi lắp đặt để có cách bố trí hợp lý, gọn gàng và dễ dàng kiểm soát hệ thống.
- **Bánh xe của động cơ dc không hoạt động:**
  + Do nhiều nguyên nhân khác nhau:
    - Tác động trong khi lắp ráp khung xe.
    - Cắm bánh xe quá chặt.
    - Cắm sai đầu âm/dương

→ Giải pháp: kiểm tra lại khung xe và dây xem đã lắp ráp đúng chưa, xoay bánh xe bằng tay xem có chạy không.

- **Bluetooth không ăn lệnh:** Có thể do điều khiển nó có vấn đề về nút (như là tải hình mũi tên bị đè nút nên không ăn được)
  → Giải pháp: chỉnh sửa lại các nút điều khiển trong MIT App Inventor.

5.5 Hướng phát triển trong tương lai:

- **Nâng cao độ chính xác của cảm biến**
+ Tích hợp thuật toán lọc nhiễu, xử lý tín hiệu tiên tiến để cải thiện độ chính xác khi đo khoảng cách.
+ Thử nghiệm các loại cảm biến khác như LiDAR, cảm biến radar để tăng độ tin cậy trong các điều kiện môi trường phức tạp.
- **Mở rộng kết nối và khả năng truyền dữ liệu**
+ Nâng cấp hệ thống để sử dụng kết nối Wi-Fi hoặc Bluetooth, giúp truyền dữ liệu từ xa qua internet.
+ Tích hợp với nền tảng IoT để quản lý dữ liệu đo khoảng cách trên cloud và điều khiển từ xa qua ứng dụng di động.
- **Cải thiện giao diện và tính năng hiển thị**
+ Phát triển ứng dụng di động hoặc trang web để giám sát dữ liệu theo thời gian thực.
+ Nâng cấp màn hình hiển thị bằng OLED hoặc màn hình cảm ứng để cung cấp thông tin trực quan hơn.
- **Ứng dụng trong thực tế**
+ Triển khai hệ thống đo khoảng cách vào các lĩnh vực như hỗ trợ người khiếm thị, điều hướng robot tự động, hệ thống an ninh cảnh báo vật cản.
+ Phát triển các thiết bị di động đo khoảng cách phục vụ công tác đo đạc trong xây dựng, kiểm tra an toàn công nghiệp.
- **Tối ưu hóa phần cứng và tiêu thụ năng lượng**
+ Sử dụng pin sạc hoặc năng lượng mặt trời để tăng tính di động và kéo dài thời gian hoạt động.

+ Thiết kế mạch nhỏ gọn hơn, giảm tiêu thụ điện năng bằng cách sử dụng các vi điều khiển tiết kiệm điện như ESP32 hoặc STM32.

– **Tích hợp trí tuệ nhân tạo (AI) để phân tích dữ liệu**

+ Áp dụng AI để dự đoán khoảng cách dựa trên dữ liệu cảm biến và phát hiện các yếu tố ảnh hưởng như nhiễu sóng, vật cản không mong muốn.

+ Xây dựng hệ thống tự động hiệu chỉnh và tối ưu hóa dựa trên điều kiện môi trường thực tế.

TÀI LIỆU THAM KHẢO

Arduinokit. (2020, 12 21). *Arduino Uno R3 là gì.* Retrieved 3 31, 2025, from https://arduinokit.vn/mach-arduino-uno-la-gi/

dientutuonglai. (n.d.). *Arduino IDE là gì.* Retrieved 03 31, 2025, from https://dientutuonglai.com/arduino-ide-la-gi.html#:~:text=Arduino%20IDE%20l%C3%A0%20m%E1%BB%99t%20ph%E1%BA%A7n,c%C5%A9ng%20c%C3%B3%20th%E1%BB%83%20l%C3%A0m%20%C4%91%C6%B0%E1%BB%A3c.

dientutuonglai. (n.d.). *HC-SR04 là gì.* Retrieved 3 31, 2025, from https://dientutuonglai.com/tim-hieu-hc-sr04.html

ESP32. (n.d.). (GRobotronics) Retrieved 3 23, 2025, from https://grobotronics.com/esp32-development-board-devkit-v1.html?sl=en&srsltid=AfmBOor1SizJUaKFEEmuSjzlZb-vGAPdMbzPOexz74pVyQQHGADcYQq1

Fe, V. (n.d.). *Phần mềm Arduino IDE là gì.* Retrieved 3 31, 2025, from https://dientutuonglai.com/arduino-ide-la-gi.html#:~:text=Arduino%20IDE%20l%C3%A0%20m%E1%BB%99t%20ph%E1%BA%A7n,c%C5%A9ng%20c%C3%B3%20th%E1%BB%83%20l%C3%A0m%20%C4%91%C6%B0%E1%BB%A3c.

Gate-Iot. (n.d.). *Cấu trúc hệ thống Internet of Things (IoT).* Retrieved 3 31, 2025, from https://gate-iot.com/cau-truc-he-thong-iot/#:~:text=Tuy%20nhi%C3%AAn%20v%E1%BB%81%20c%C6%A1%20b%E1%BA%A3n,%E1%BB%A9ng%20d%E1%BB%A5ng%20(Application%20Layer).

HC-SR04. (n.d.). (Arduino) Retrieved 3 30, 2025, from http://arduino.vn/bai-viet/233-su-dung-cam-bien-khoang-cach-hc-sr04

Introduction to Internet of Things (IoT) – Set 1. (n.d.). (GeeksForGeeks) Retrieved 3 23, 2025, from https://www.geeksforgeeks.org/introduction-to-internet-of-things-iot-set-1/

Mescu. (2023, 4 13). *Màn hình LCD I2C là gì.* Retrieved 3 31, 2025, from https://mecsu.vn/ho-tro-ky-thuat/giao-dien-lcd-i2c-voi-van-ban-chay-tren-man-hinh-arduino-va-cac-ky-tu-tuy-chinh.Ql5#:~:text=M%C3%A0n%20h%C3%ACnh%20LCD%20I2C%20n%C3%A0y,bi%E1%BB%83u%20%C4%91%E1%BB%93%20thanh%20lo%E1%BA%A1i%20%C4%91%C3%B3.

MIT App Inventer Documentation. (n.d.). (MIT App Inventer) Retrieved 3 28, 2025, from https://appinventor.mit.edu/

Vettelidc. (2024, 10 5). *IOT là gì.* Retrieved 3 31, 2025, from https://viettelidc.com.vn/tin-tuc/iot-la-gi-nhung-ung-dung-noi-bat-cua-iot

Viettelidc. (2024, 10 5). *IOT là gì.* Retrieved 3 31, 2025, from https://viettelidc.com.vn/tin-tuc/iot-la-gi-nhung-ung-dung-noi-bat-cua-iot

Viettelidc. (2024, 10 4). *IOT là gì.* Retrieved 3 31, 2025, from https://viettelidc.com.vn/tin-tuc/iot-la-gi-nhung-ung-dung-noi-bat-cua-iot