KHOA CƠ KHÍ GIAO THÔNG

ĐỒ ÁN TỐT NGHIỆP

Tên đề tài:

NGHIÊN CỨU, THIẾT KẾ MÔ HÌNH XE THÔNG MINH SỬ DỤNG CÔNG NGHỆ GIAO TIẾP V2V

SVTH: Đặng Minh Hiếu_ 20C4A

GVHD: TS. Hoàng Thắng

Đà Nẵng, 2025

TÓM TẮT

Tên đề tài: Nghiên cứu, thiết kế mô hình xe thông minh sử dụng công nghệ giao tiếp V2V

Sinh viên thực hiện: Đặng Minh Hiếu

Số thẻ SV: 103200012 Lớp: 20C4A

Đồ án tập trung nghiên cứu và xây dựng mô hình xe điện thông minh tích hợp ba hệ thống cốt lõi: hệ thống dò line, hệ thống tránh vật cản, và đặc biệt là công nghệ giao tiếp giữa các phương tiện (Vehicle-to-Vehicle – V2V) nhằm nâng cao tính an toàn và tự động hóa trong điều khiển phương tiện.

Trong mô hình, hai xe điện được thiết kế: một xe dẫn đầu và một xe đi sau. Xe dẫn đầu sử dụng cảm biến line để dò đường, kết hợp cảm biến siêu âm để phát hiện vật cản phía trước. Khi gặp chướng ngại vật, xe lập tức dừng lại và gửi tín hiệu "STOP" đến xe sau thông qua giao tiếp không dây sử dụng module LoRa. Khi vật cản được giải phóng, xe truyền lệnh "GO", cho phép xe phía sau tiếp tục hành trình. Cơ chế này mô phỏng chính xác nguyên lý hoạt động của hệ thống V2V trên các xe thông minh thực tế, nơi các phương tiện có thể trao đổi thông tin về trạng thái vận hành, cảnh báo nguy hiểm và tự động điều chỉnh hành vi mà không cần sự can thiệp từ con người.

Hệ thống dò line được điều khiển bằng thuật toán PID giúp xe di chuyển ổn định theo vạch định sẵn, trong khi hệ thống tránh vật cản sử dụng cảm biến siêu âm để tạm dừng chuyển động khi gặp chướng ngại và tự động tiếp tục khi an toàn. Công nghệ V2V đóng vai trò trung tâm trong việc đồng bộ hóa trạng thái giữa hai xe, đảm bảo cả hai xe hoạt động phối hợp an toàn và hiệu quả.

Đồ án áp dụng vi điều khiển Arduino Uno và các module truyền thông LoRa để hiện thực hóa mô hình, đồng thời xây dựng cơ sở lý thuyết vững chắc dựa trên điều khiển PID, giao tiếp không dây và nguyên lý vận hành của hệ thống xe điện. Kết quả đạt được không chỉ minh chứng tính khả thi của công nghệ V2V trong thực tế, mà còn đặt nền móng cho việc phát triển các hệ thống điều khiển đoàn xe thông minh, hỗ trợ định hướng tương lai cho xe tự hành và hệ thống giao thông thông minh.

LỜI NÓI ĐẦU

Sau quá trình học tập và nghiên cứu tại **Trường Đại học Bách Khoa – Đại học Đà Nẵng**, được trang bị những kiến thức chuyên ngành về **Cơ khí Động lực**, em đã có cơ hội vận dụng và phát triển những kiến thức đó thông qua việc thực hiện **Đồ án tốt nghiệp (Capstone Project)**. Đây là một cột mốc quan trọng giúp em tổng hợp kiến thức đã học, rèn luyện kỹ năng tư duy, làm việc độc lập cũng như tiếp cận thực tế nghề nghiệp.

Trong suốt quá trình thực hiện đồ án, em đã nhận được sự hỗ trợ tận tình và quý báu từ các thầy cô, đơn vị hướng dẫn. Em xin bày tỏ lòng biết ơn sâu sắc đến **TS. Hoàng Thắng** – giảng viên hướng dẫn đã luôn tận tình chỉ bảo, định hướng, góp ý và đồng hành cùng em trong suốt thời gian thực hiện đồ án. Những nhận xét và hướng dẫn của thầy là kim chỉ nam giúp em hoàn thiện đề tài một cách tốt hơn.

Em cũng xin chân thành cảm ơn **TS. Lưu Đức Lịch** – giảng viên phản biện đã dành thời gian xem xét, đánh giá và đưa ra những nhận xét, góp ý xác đáng để em có thể nhìn nhận lại đề tài một cách khách quan hơn và rút ra nhiều bài học quý báu.

Đồng thời, em xin gửi lời cảm ơn đến các thầy cô trong **Khoa Cơ khí Giao thông** đã luôn tạo điều kiện học tập thuận lợi, truyền đạt kiến thức nền tảng cũng như hỗ trợ em trong suốt quá trình học tập và thực hiện đồ án.

Em cũng xin chân thành cảm ơn **Xí nghiệp Đầu máy Sài Gòn – cơ sở Đà Nẵng**, nơi đã tạo điều kiện cho em thực tập thực tế, tiếp cận quy trình vận hành, sửa chữa đầu máy và tích lũy thêm nhiều kinh nghiệm thực tiễn quý báu.

Mặc dù đã nỗ lực hoàn thành đồ án với tinh thần nghiêm túc và trách nhiệm, nhưng do thời gian có hạn và năng lực bản thân còn hạn chế, bài báo cáo không thể tránh khỏi những thiếu sót. Em rất mong nhận được sự thông cảm cùng những góp ý quý báu từ quý thầy cô để có thể rút kinh nghiệm và hoàn thiện hơn trong các công việc tương lai

Em xin chân thành cảm ơn!

MỤC LỤC

DANH SÁCH CÁC BẢNG, HÌNH VẼ

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

Viết tắt	Ý nghĩa
V2V	Vehicle-to-Vehicle (Giao tiếp giữa các phương tiện)
CCS	Cruise Control System (Hệ thống điều khiển hành trình)
ECU	Electronic Control Unit (Bộ điều khiển điện tử)
ACC	Adaptive Cruise Control (Điều khiển hành trình thích ứng)
PID	Proportional-Integral-Derivative (Điều khiển tỉ lệ – tích phân – vi phân)
DSRC	Dedicated Short Range Communications (Truyền thông tầm ngắn chuyên dụng)
EV	Electric Vehicle (Xe điện)
RES/COAST	Thiết lập/Giảm tốc độ
RES/ACC	Resume/Accelerate (Khôi phục/Tăng tốc)
LoRa	Long Range (Công nghệ truyền thông tầm xa)
GPS	Global Positioning System (Hệ thống định vị toàn cầu)
HEV	Hybrid Electric Vehicle (Xe lai điện)
IFV	Intelligent Following Vehicle (Xe thông minh bám theo)
FLD	Front-Line Detection (Dò line phía trước)
ODA	Obstacle Detection Algorithm (Thuật toán phát hiện vật cản)

MỞ ĐẦU

Mục đích thực hiện đề tài

Theo thống kê, tỷ lệ lớn các vụ tai nạn giao thông xảy ra do sự thiếu chú ý hoặc sai sót của người lái xe. Để giải quyết vấn đề này, việc nghiên cứu và phát triển hệ thống Cruise control kết hợp với công nghệ Vehicle-to-Vehicle (V2V) đang trở nên hết sức cần thiết. Hệ thống cruise control truyền thống giúp duy trì tốc độ của xe một cách tự động, nhưng khi kết hợp với V2V, hệ thống này có thể trở thành một công cụ mạnh mẽ để tránh va chạm. Thông qua việc giao tiếp với các phương tiện xung quanh, xe có thể nhận diện được tình trạng giao thông, khoảng cách với các xe khác và nhận diện các tình huống nguy hiểm. Khi phát hiện một nguy cơ va chạm, hệ thống có thể ngay lập tức đưa ra cảnh báo cho người lái hoặc tự động điều chỉnh tốc độ và khoảng cách để giảm thiểu rủi ro.

Mục tiêu nghiên cứu

Xây dựng mô hình xe điện và tích hợp công nghệ thông minh.

Nâng cao an toàn giao thông, giảm thiểu nguy cơ va chạm.

Giảm tiêu thụ nhiên liệu và khí thải nhờ tối ưu hóa tốc độ và khoảng cách.

Phạm vi và đối tượng nghiên cứu

Tìm hiểu khái quát về cấu tạo, nguyên lý hoạt động của xe điện.

Tìm hiểu chức năng, nguyên lý hoạt động của hệ thống Cruise Control System và Công nghệ Vehicle-to-Vehicle.

Thiết kế mô hình xe và lắp đặt các bo mạch của hệ thống CCS kết nối với công nghệ giao tiếp thông qua module LoRa.

Phương pháp nghiên cứu

Ứng dụng phần mềm Aduino IDE đưa thuật toán vào bo mạch Aduino Uno

Kiểm nghiệm độ chính xác ứng với điều kiện tốc độ và khoảng cách của 2 xe.

Xây dựng mô hình xe thông minh sử dụng hệ thống Cruise Control tích hợp các yếu tố giao tiếp V2V để kiểm tra hệ thống phản ứng thông tin từ các xe khác.

Đánh giá độ trễ trong giao tiếp và tỷ lệ lỗi để xác định hiệu suất của hệ thống.

Nội dung nghiên cứu

Dựa vào việc trao đổi thông tin giữa các phương tiện ở gần để cảnh báo người lái về các tình huống nguy hiểm có thể dẫn đến va chạm.

Hệ thống Cruise Control thông minh duy trì vận tốc ổn định giúp người lái thư giãn hơn trong các chuyến đi dài và khi tích hợp với công nghệ V2V tạo một mạng lưới và truyền thông tầm ngắn chuyên dùng (DSRC) để nâng cao tính an toàn cần thiết.

Chương 1: TỔNG QUAN

1.1. Sự phát triển của công nghệ trong ngành công nghiệp ô tô.

1.1.1. Lịch sử phát triển ô tô

Thật khó để xác định việc phát minh ra ô tô điện cho một nhà phát minh hay quốc gia nào. Lịch sử xe điện trải qua thời kỳ của một loạt các bước đột phá, từ pin tới động cơ điện. Vào đầu thế kỷ 19 các nhà đổi mới ở Ba Lan và Hoa Kỳ bắt đầu đưa ra khái niệm xe chạy bằng pin và tạo ra một số ô tô điện quy mô nhỏ đầu tiên. Thời kỳ này xe chạy bằng nguồn năng lượng điện đã có vị thế cạnh tranh tương đương với xe chạy bằng động cơ hơi nước. Theo dòng lịch sử, vào khoảng những năm 1832 và 1839, Robert Anderson người Scotland đã phát minh ra loại xe điện chuyên chở đầu tiên. Năm 1842, hai nhà phát minh người Mỹ là Thomas Davenport và Scotsmen Robert Davidson trở thành những người đầu tiên đưa pin vào sử dụng cho ô tô điện [1].

Hình 1.1 Loại xe điện chuyên chở đầu tiên

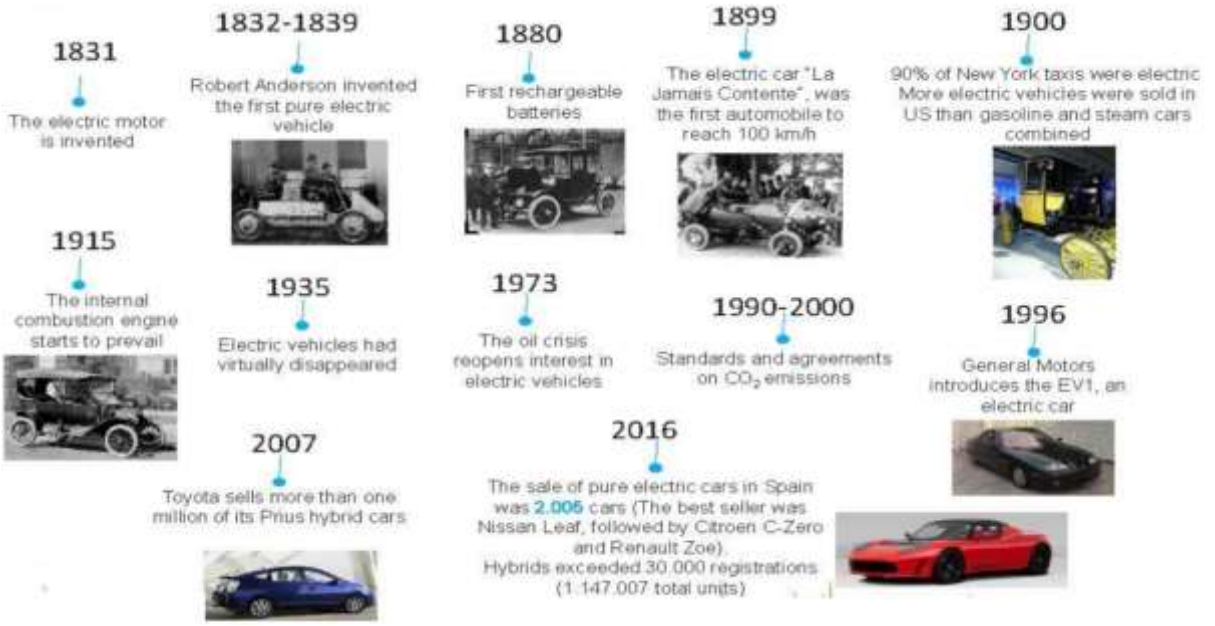
Đến những năm 1865, Camile Faure đã thành công trong việc nâng cao khả năng lưu trữ điện trong pin giúp cho xe điện có thể di chuyển một quãng đường dài hơn.

Tại Hoa Kỳ, chiếc ô tô điện chở sáu hành khách có khả năng đạt tốc tối đa 14 dặm một giờ được thành công đầu tiên ra mắt vào khoảng năm 1890 nhờ William Morrion. Sau vài năm, xe điện trở nên phổ biến và nhà sản xuất rải rộng xuất hiện ở khắp Hoa Kỳ. Đến những năm 1900s, ô tô điện đang ở thời hoàng kim, chiếm 1/3 tổng số phương tiện lưu thông trên đường.

Trong khoảng năm 1930s, xe điện bước vào thời kỳ trở ngại với rất ít tiến bộ trong công nghệ. Sự cải tiến động cơ đốt trong và giá thành nhiên liệu rẻ, nguồn tài nhiên dầu mỏ dồi dào đã cản trở nhu cầu về các phương tiên sử dụng nhiên liệu thay thế và xe điện dường như không được nhắc tới.

Đến cuối những năm 1960 và đầu những năm 1970. Giá dầu tăng vọt và tình trạng thiếu xăng dầu lên đến đỉnh điểm và Lệnh cấm dầu mỏ của khối Ả Rập năm 1973 đã tạo

ra mối quan tâm ngày càng tăng và khi đó nhiều nhà sản xuất ô tô bắt đầu khám phá các lựa chọn cho phương tiện sử dụng nhiên liệu thay thế, bao gồm cả ô tô điện.

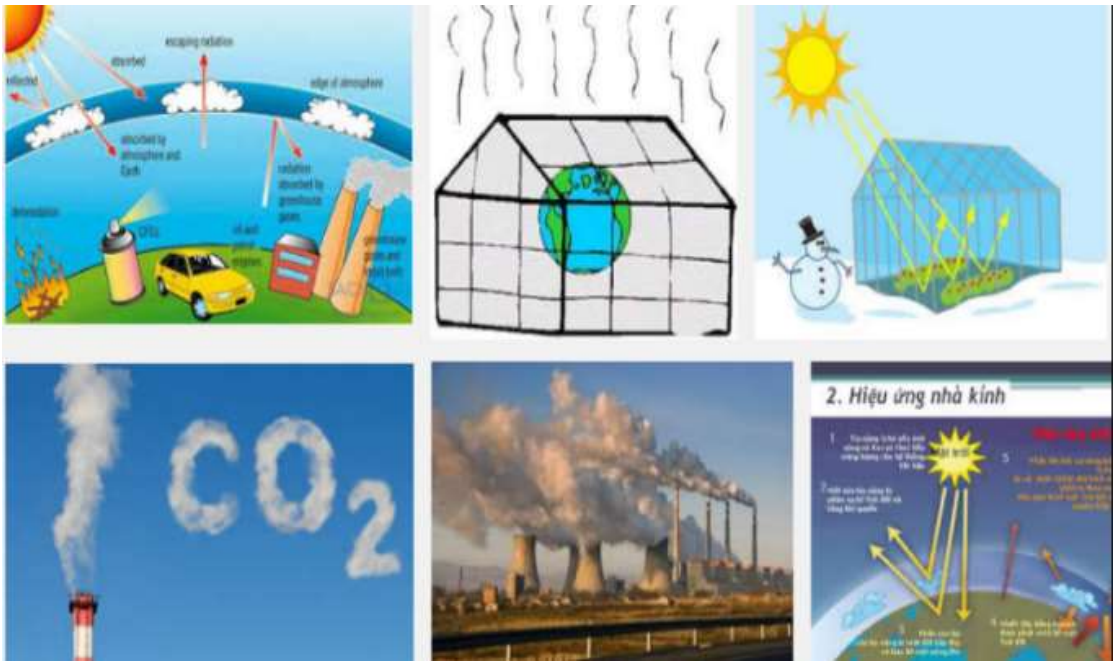


Hình 1.2 Lịch sử hình thành ô tô điện

Tuy nhiên, những phương tiện được phát triển và sản xuất vào những năm 1970 vẫn có những nhược điểm so với ô tô chạy bằng xăng. Xe điện trong thời gian này có hiệu suất hạn chế, tốc độ thấp và xe điện có xu thế giảm dần. Đến những năm 1990s, việc quan tâm tới biến đổi khí hậu trở lại. Trong thời gian này, các nhà sản xuất ô tô bắt đầu sửa đổi một số mẫu xe phổ biến của họ thành xe điện. Điều này có nghĩa là các loại xe điện giờ đây đã đạt được tốc độ và hiệu suất điện gần hơn nhiều so với các loại xe chạy bằng xăng [2]. Nhưng do chi phí sản xuất cao, ô tô điện thời kỳ này gần như không khả thi về mặt thương mại. Mặc dù công chúng không chú ý nhiều đến xe điện, nhưng các nhà khoa học vẫn đang làm việc để cải tiến công nghệ xe điện, mật độ lưu trữ và khả năng an toàn cũng như tăng tuổi thọ trên pin xe điện.

1.1.2. Những mối quan tâm về ngành công nghiệp ô tô

Ô tô là một trong những phương tiện di chuyển hàng ngày và vô cùng quen thuộc với mỗi người. Sự xuất hiện của động cơ đốt trong vào cuối thế kỷ 19 đã tạo nên bước đột phá cho ngành công nghiệp ô tô . Số lượng ô tô được sử dụng tại Việt Nam và trên thế giới ngày một nhiều và phổ biến ở mọi tầng lớp dân cư. Rõ ràng, không thể phủ nhận những lợi ích mà ô tô mang lại trong giao thương, buôn bán và cuộc sống hàng ngày. Tuy nhiên, ô tô trang bị động cơ đốt trong chính là đối tượng gây ô nhiễm môi trường hàng đầu.



Hình 1.3 Các nguyên nhân gây ra từ việc sử dụng nhiên liệu hoá thạch [1]

Hiện nay, mặc dù công nghệ xử lý khí thải trên ô tô đã rất phát triển, nhưng không thể nào loại bỏ hoàn toàn ô nhiễm do nó gây ra. Một lượng lớn các chất độc hại vẫn được thải trực tiếp vào khí quyển. Phần lớn các xe hiện nay đều sử dụng động cơ xăng, khí thải của động cơ này chủ yếu chứa CO2 (gây hiệu ứng nhà kính), CO (ảnh hưởng tới đường hô hấp) hay HC (tác động nguy hại tới mắt, phổi) [2, 3]. Đối với các xe trang bị động cơ diesel, lượng khí thải còn nguy hại hơn khi trong thành phần của nó chứa rất nhiều NOx (nguyên nhân của các cơn mưa axit) và bụi PM (gây viêm hô hấp, hen và ung thư). Theo những nghiên cứu gần đây, sự phát thải bụi PM của xe trang bị động cơ diesel có thể cao gấp 10-40 lần so với xe trang bị động cơ xăng [4]. Do đó, dù sử dụng động cơ xăng hay động cơ diesel đều có nguy cơ ảnh hưởng đến môi trường và sức khỏe con người. Để hạn chế những tác hại này, các quốc gia trên thế giới đang có xu hướng chuyển dần sang sử dụng ô tô điện và ô tô sử dụng năng lượng sạch.

Hình 1.4 Mật độ lưu lượng giao thông

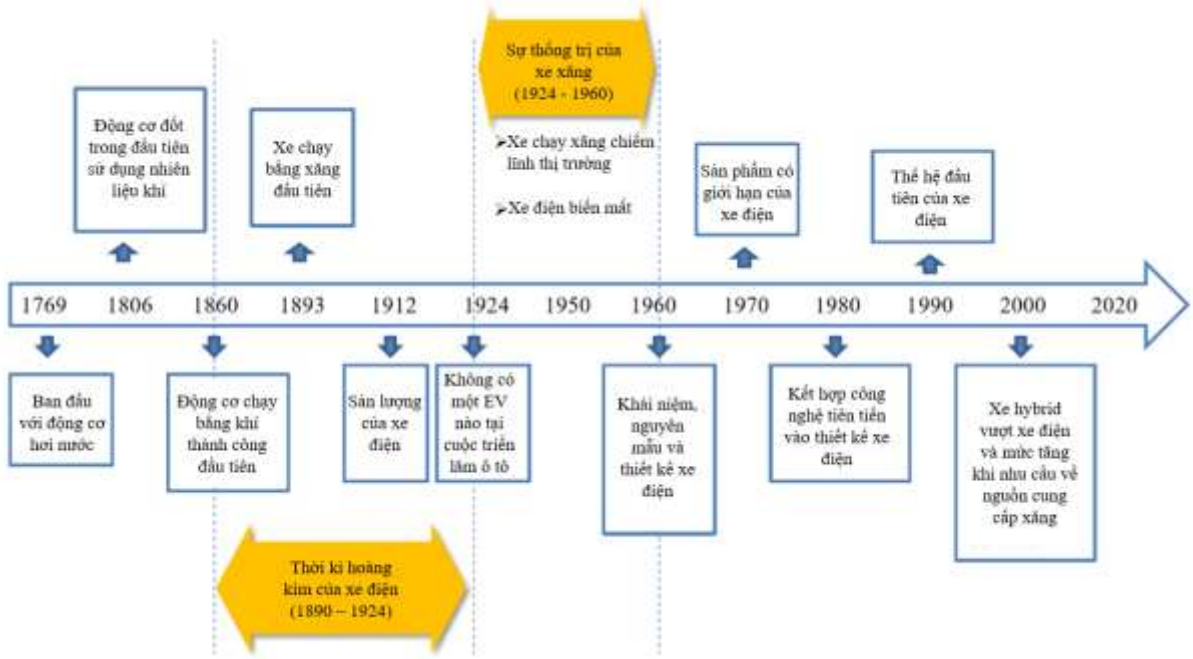
Mật độ giao thông ngày càng gia tăng đang trở thành một mối lo ngại lớn tại Việt Nam, đặc biệt ở các đô thị lớn như Hà Nội và TP. Hồ Chí Minh. Sự phát triển nhanh chóng của dân số và số lượng phương tiện cá nhân, nhất là ô tô, đã dẫn đến tình trạng ùn tắc kéo dài, ô nhiễm không khí và tai nạn giao thông gia tăng. Hạ tầng giao thông chưa theo kịp tốc độ đô thị hóa, trong khi ý thức tham gia giao thông của một bộ phận người dân còn hạn chế, làm trầm trọng thêm vấn đề. Nếu không có những giải pháp đồng bộ và hiệu quả, như mở rộng hạ tầng, phát triển giao thông công cộng và nâng cao ý thức cộng đồng, thì mật độ giao thông cao sẽ tiếp tục là rào cản lớn đối với sự phát triển bền vững của đất nước.

Hình 1.5 Sự cố xảy ra tai nạn nghiêm trọng do bất cẩn của người lái

1.1.3. Xu hướng sử dụng nhiên liệu sạch của ô tô

Ô tô điện không còn quá xa lạ với mọi người. Với ưu điểm hạn chế/ không phát thải trực tiếp khí độc, ô tô điện đang là xu hướng chính trong ngành công nghiệp ô tô. Nguồn động lực trên xe lúc này không chỉ còn duy nhất là động cơ đốt trong thông thường, thay vào đó sẽ là sự kết hợp/ thay thế bởi động cơ điện công suất cao. Những năm gần đây, số lượng ô tô điện được sử dụng ngày một gia tăng, đặc biệt là ở các quốc gia phát triển.

Ô tô chạy điện về nguyên tắc là ô tô sạch tuyệt đối (zero emission) đối với môi trường không khí trong thành phố. Nguồn điện dùng để chạy ô tô được nạp vào acquy do đó quãng đường hoạt động độc lập của ô tô phụ thuộc vào khả năng tích điện của acquy. Nếu nguồn điện được sản xuất từ các nguồn năng lượng tái sinh (thuỷ điện, pin mặt trời...) thì ô tô dùng điện là loại phương tiện lí tưởng nhất về mặt ô nhiễm môi trường. Tuy nhiên, nếu nguồn điện được sản xuất từ nhiên liệu hoá thạch thì ưu điểm này bị hạn chế nếu xét về mức độ phát ô nhiễm tổng thể. Ngày nay, ô tô chạy bằng acquy đã đạt được những tính năng vận hành cần thiết giống như ô tô sử dụng nhiên liệu lỏng truyền thống.



Hình 1.6 Xu hướng cải tiến ô tô sử dụng nhiên liệu sạch

Sự phát triển của xe điện phụ thuộc nhiều vào công nghệ acquy. Nhiều nước đã đầu tư rất lớn cho việc nghiên cứu phát triển acquy để nâng cao hiệu quả nhằm đáp ứng yêu cầu của xe điện, nhưng cho tới nay, kết quả đạt được vẫn còn hạn chế. Khả năng lưu trữ năng lượng thấp của acquy làm giới hạn tầm hoạt động của xe. Vì vậy xe điện chỉ được dùng trong phạm vi ngắn (50 km – 200 km) tùy vào khối lượng của xe và mật độ lưu trữ năng lượng. Thực tế cho thấy rằng, tính năng của xe điện sẽ không bao giờ vượt qua được xe sử dụng nhiên liệu lỏng, ngay cả với những giá trị lạc quan về khả năng lưu trữ năng lượng của acquy.

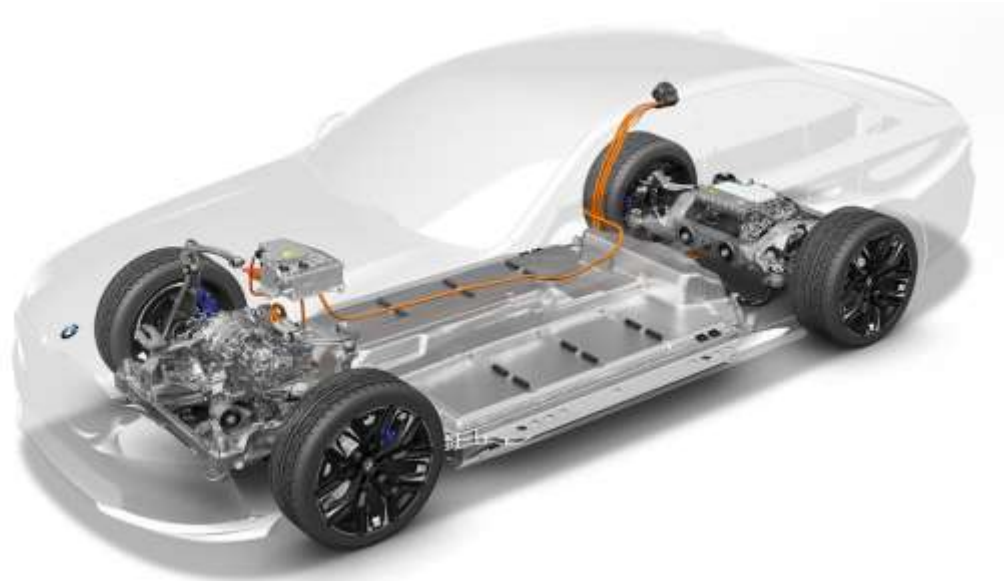
Ô tô điện có nhiều ưu điểm hơn ô tô truyền thống sử dụng động cơ đốt trong như không phát thải khí ô nhiễm, hiệu quả cao, không phụ thuộc vào dầu mỏ và hoạt động êm dịu, không gây ồn. Các nguyên tắc hoạt động cơ bản của ô tô điện cũng tương tự như ô tô sử dụng động cơ đốt trong, chỉ khác nhau về động cơ và năng lượng sử dụng.

Tuy nhiên, ô tô điện thực sự vẫn chưa phải là giải pháp tối ưu để bảo vệ môi trường. Nguyên nhân chính bởi vì chúng vẫn phải sử dụng nguồn điện nhân tạo (nhiệt điện, thủy điện, điện hạt nhân). Mặc dù chúng không phát thải trực tiếp khí độc ra ngoài môi trường, nhưng vẫn phần nào tác động tiêu cực đến môi trường xung quanh. Một giải pháp mới được đưa ra chính là sử dụng nguồn năng lượng sạch. Năng lượng sạch ở đây chính là các nguồn năng lượng từ thiên nhiên và không gây phát thải ô nhiễm. Một số dạng năng lượng có thể được xem xét đến, như gió, thủy triều, ánh sáng mặt trời, địa nhiệt, khí hydro hay nước. Trong số năng lượng kể trên, có thể sử dụng nước, khí hydro hoặc ánh sáng mặt trời để chuyển đổi thành năng lượng của ô tô. Những nguồn năng lượng này gần như không tốn chi phí (hoặc có thể là chi phí rất nhỏ) để mua, chúng không gây phát thải ô nhiễm và rất thân thiện với môi trường. Xu hướng sử dụng các mẫu xe tiêu thụ năng lượng sạch đang rất được quan tâm.

1.2. Điện hoá trên ô tô

1.2.1. Giới thiệu chung

Xe ô tô điện là loại phương tiện được cung cấp năng lượng bởi động cơ điện. Thay vì sử dụng các động cơ đốt trong với các nhiên liệu như xăng hoặc dầu diesel, ô tô điện sử dụng năng lượng được cung cấp từ một bộ pin sạc. Ô tô chạy hoàn toàn bằng năng lượng điện gọi là xe điện thuần túy (EV). Ngoài ra, loại xe ô tô vừa có thể chạy bằng điện, vừa có thể chạy bằng các nhiên liệu khác được gọi là xe điện lai - xe hybrid (HEV)[2].



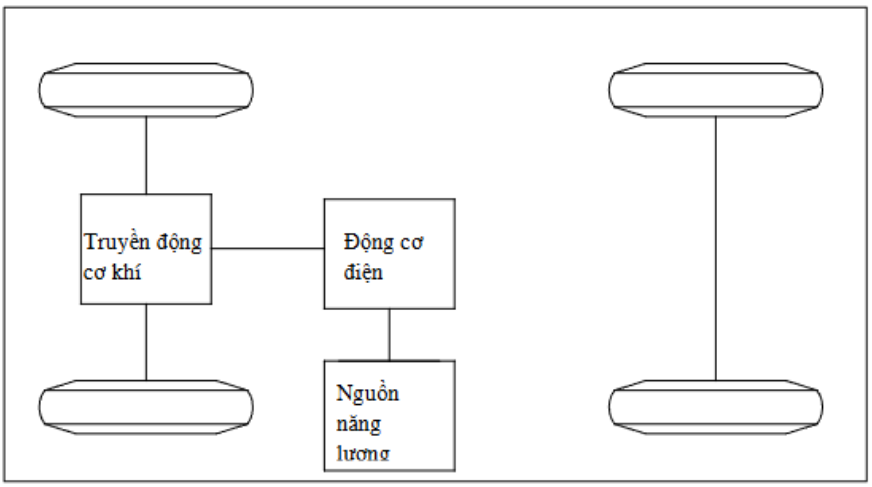
Hình 1.7 Tổng quan xe điện thuần tuý

Ô tô điện sử dụng một động cơ điện cho lực kéo; acquy, pin nhiên liệu cung cấp nguồn năng lượng tương ứng cho động cơ điện.

Ô tô điện có nhiều ưu điểm hơn các loại phương tiện sử dụng động cơ đốt trong, chẳng hạn như không phát thải khí ô nhiễm, hiệu suất cao, độc lập với nguồn năng lượng từ dầu mỏ, yên tĩnh và hoạt động trơn tru. Các nguyên tắt hoạt động cơ bản giữa ô tô điện và phương tiện sử dụng động cơ đốt trong tương tự nhau.Tuy nhiên, một số khác biệt giữa phương tiện sử dụng động cơ đốt trong và ô tô điện, chẳng hạn như sử dụng một bồn chứa xăng so với nguồn pin, động cơ đốt trong so với động cơ điện, và khác nhau về yêu cầu truyền dẫn.

1.2.2. Cấu hình ô tô điện

Trước đây, các xe điện chủ yếu được chuyển đổi từ các ô tô thông thường bằng cách thay thế động cơ đốt trong và thùng nhiên liệu với một động cơ điện và pin trong khi giữ lại tất cả các thành phần khác, như trong hình 1.6 . Nhược điểm như: khối lượng lớn, tính linh hoạt và hiệu suất thất là những nguyên nhân làm cho xe điện khó áp dụng rộng rãi [3]. Hiện nay, ô tô hiện đại được tạo ra có chủ ý dựa vào nguyên bản của thân và khung sườn được thiết kế riêng. Điều này đáp ứng các yêu cầu về cấu trúc duy nhất cho ô tô và làm cho các nguồn động lực đẩy bằng điện được sử dụng linh hoạt hơn.

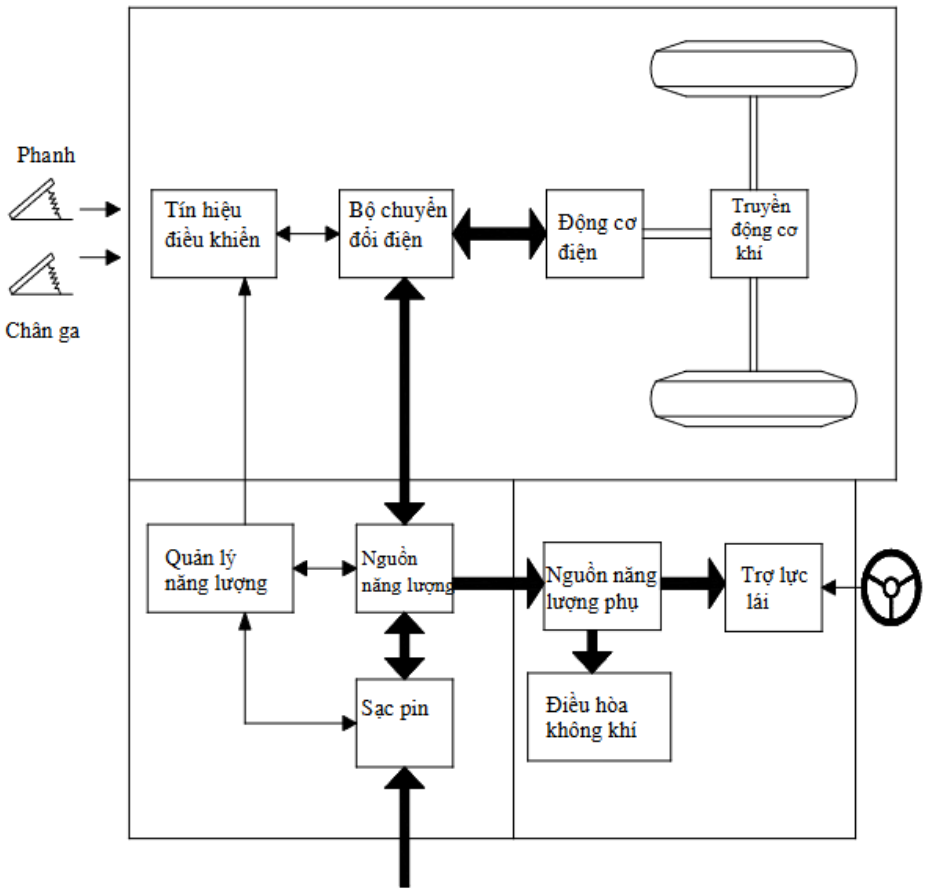


Hình 1.8 Ô tô điện cổ điển

Một ô tô điện cơ bản được minh họa trong hình 1.6. Nó bao gồm ba hệ thống chủ yếu: hệ động lực điện, hệ thống năng lượng, và hệ thống phụ trợ.

- Hệ động lực điện bao gồm hệ thống điều khiển xe, bộ chuyển đổi điện, các động cơ điện truyền động cơ khí, và bánh chủ động.
- Hệ thống năng lượng bao gồm nguồn năng lượng bộ phận quản lý năng lượng, và bộ phận tiếp năng lượng điện.
- Hệ thống phụ trợ bao gồm trợ lực lái, điều hòa, nguồn cung cấp năng lượng phụ trợ.

Dựa trên các yếu tố đầu vào điều khiển từ chân ga và bàn đạp phanh, hệ thống điều khiển xe cung cấp tín hiệu điện thích hợp cho bộ chuyển đổi năng lượng điện có chức năng điều chỉnh dòng điện giữa điện động cơ và nguồn năng lượng. Những nguồn năng lượng được tái sinh trong quá trình phanh có thể được nạp vào nguồn năng lượng chính. Hầu hết pin EV dễ dàng có khả năng tiếp nhận nguồn năng lượng tái sinh này.



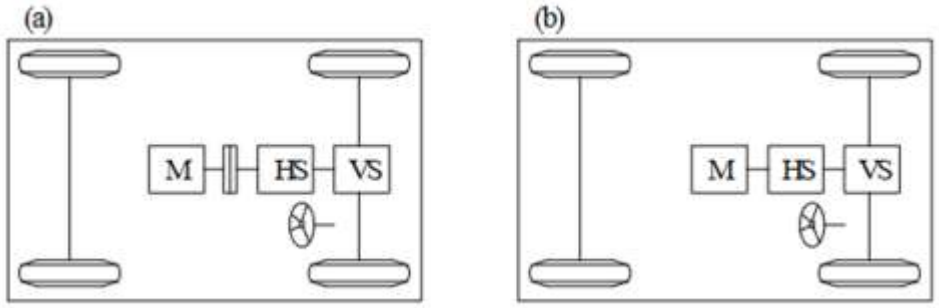
Hình 1.9 Ô tô điện hiện đại

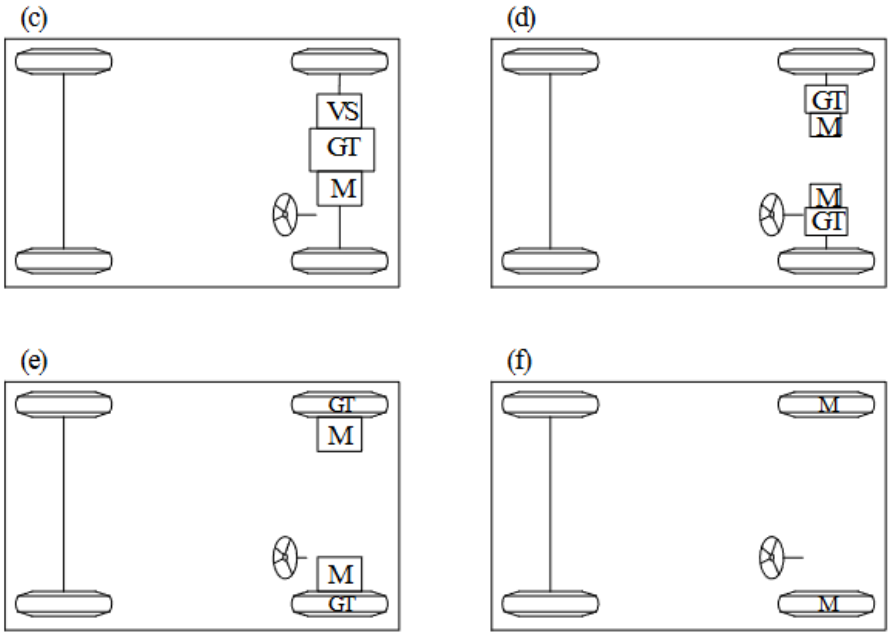
Bộ phận quản lý năng lượng cùng với bộ phận điều khiển kiểm soát hoạt động phanh tái sinh và phục hồi năng lượng của nó. Nó cũng kết hợp với các bộ phận tiếp năng lượng để kiểm soát quá trình này và giám sát việc sử dụng các nguồn năng lượng.

Nguồn cung cấp năng lượng phụ có chức năng cung cấp năng lượng cần thiết với các điện áp khác nhau cho tất cả các thành phần phụ của xe như: điều hòa không khí, trợ lực lái, hệ thống đèn chiếu sáng...

1.1.3. Các dạng mô hình xe điện

Có nhiều loại EV có thể cấu tạo khác nhau do các biến thể dựa trên đặc điểm của động lực điện và các nguồn năng lượng, như trong hình 1.8.





Hình 1.10 Các loại cấu hình ô tô điện

M: động cơ điện; HS: hộp số; VS: truyền lực chính và vi sai; GT: hộp giảm tốc

a. Hình 1.8a minh họa dạng cơ bản nhất của xe điện, trong đó động cơ điện thay thế cho động cơ đốt trong của xe truyền thống. Hệ thống truyền động bao gồm: động cơ điện, ly hợp, hộp số và bộ vi sai. Trong một số trường hợp, ly hợp và hộp số có thể được tích hợp hoặc thay thế bằng một hộp số tự động.

b. Nhờ vào đặc tính làm việc của động cơ điện, với khả năng duy trì công suất liên tục trong dải tốc độ rộng, tỉ số truyền cố định có thể được sử dụng thay cho hộp số nhiều cấp. Điều này loại bỏ nhu cầu sử dụng ly hợp, giúp giảm trọng lượng và kích thước hệ thống truyền động cơ khí. Đồng thời, cấu hình này cũng đơn giản hóa việc điều khiển xe, vì không cần phải thay đổi tỉ số truyền khi vận hành.

c. Tương tự cấu hình (b), động cơ điện, cặp bánh răng giảm tốc cố định và bộ vi sai có thể được tích hợp thành một cụm đặt giữa hai bán trục dẫn động. Cấu hình này giúp hệ thống truyền động gọn gàng, dễ bố trí và tăng độ tin cậy trong vận hành.

d. Trong cấu hình hình 1.8d, bộ vi sai truyền thống được loại bỏ, thay vào đó là hai động cơ điện độc lập, mỗi động cơ dẫn động một bánh xe. Khi xe rẽ hoặc quay vòng, hai động cơ hoạt động với tốc độ khác nhau, đảm bảo sự linh hoạt và ổn định khi vào cua.

e. Để tiếp tục đơn giản hóa thiết kế và điều khiển, động cơ điện có thể được đặt bên trong bánh xe (in-wheel motor). Một cặp bánh răng giảm tốc nhỏ được tích hợp bên trong bánh để giảm tốc độ quay và tăng mô-men xoắn đầu ra của động cơ.

f. Ở mức độ tích hợp cao hơn, hệ truyền động bánh răng giữa động cơ và bánh xe được loại bỏ hoàn toàn. Roto của động cơ điện tốc độ thấp được nối trực tiếp với bánh xe chủ động. Trong cấu hình này, việc điều chỉnh tốc độ quay của động cơ cũng đồng thời điều khiển tốc độ xe. Tuy nhiên, để đảm bảo khả năng khởi động và tăng tốc, động cơ điện cần có mô-men xoắn lớn hơn so với các cấu hình khác.

1.3. Giới thiệu các hệ thống và công nghệ được trang bị trên xe điện

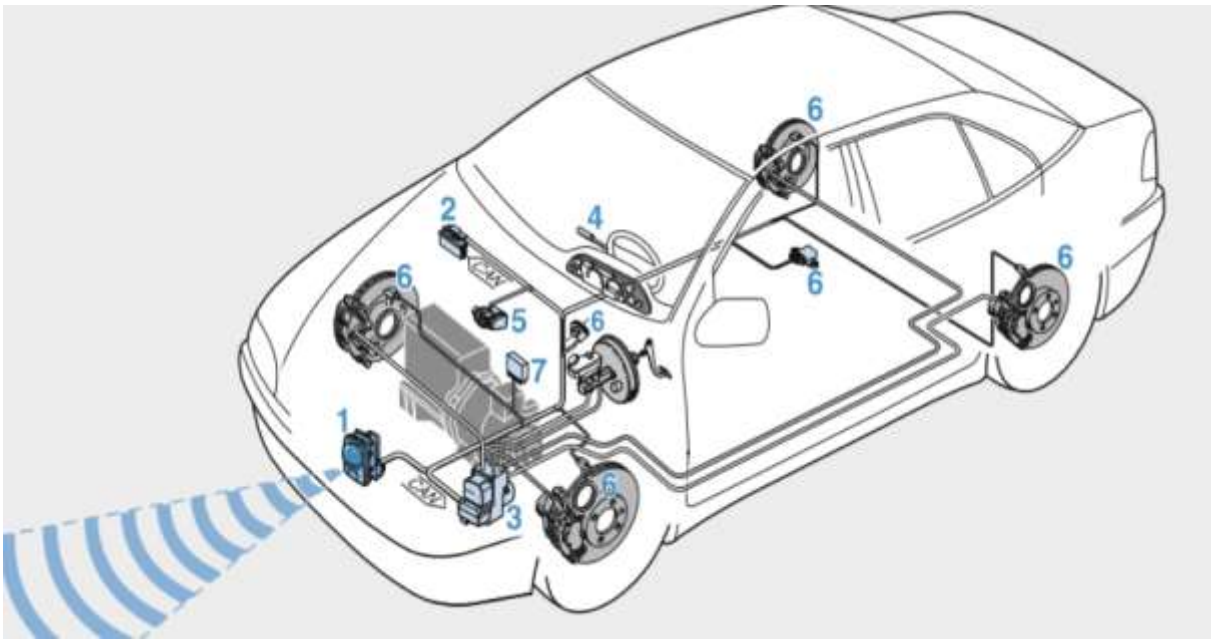
1.3.1. Giới thiệu hệ thống Cruise Control

a, Khái niệm của hệ thống Cruise Control

Hệ thống ga tự động Cruise Control (CCS) duy trì xe chạy tại một tốc độ do lái xe đặt trước bằng cách điều chỉnh tự động góc mở bướm ga, do đó người lái không cần phải giữ chân ga. Hệ thống ga tự động Cruise Control (CCS) đặc biệt có ích khi lái xe liên tục không nghỉ trong nhiều giờ trên đường cao tốc hay đường xuyên quốc gia vắng người, do người lái có thể thả chân ga đạp ga và xe sẽ chạy ở một tốc độ không đổi cho dù là lên hay xuống dốc.

Nhờ có hệ thống ga tự động Cruise Control (CCS) những chuyến hành trình dài sẽ ít gây mệt mỏi hơn. Hệ thống ga tự động Cruise Control (CCS) được áp dụng nhiều trên những ôtô Mỹ hơn những ôtô Châu Âu, bởi vì những con đường ở Mỹ rộng lớn hơn và nói chung thẳng hơn.

Với sự phát triển không ngừng của giao thông, hệ thống ga tự động Cruise Control (CCS) đang trở thành hữu ích hơn, những ôtô đời mới tương lai sẽ được trang bị hệ thống ga tự động Cruise Control (CCS), nó sẽ cho phép ôtô của bạn đi theo ôtô phía trước nó trong một đoàn xe nhờ liên tục điều chỉnh tăng tốc hoặc giảm tốc để bảo đảm một khoảng cách an toàn.



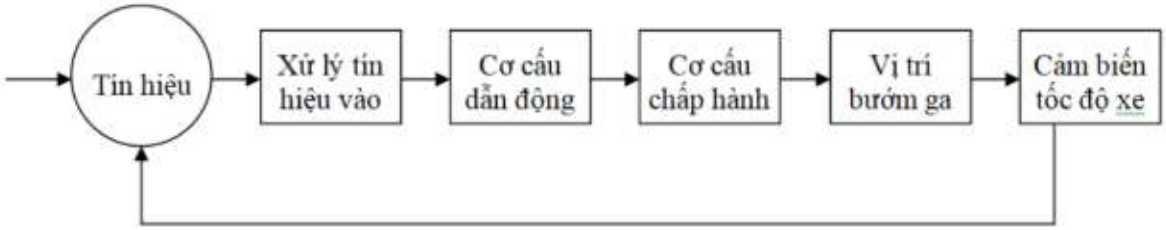
Hình 1.11 Sơ đồ bố trí của hệ thống cruise control

1. Cảm biến khoảng cách, rada và bộ điều khiển; 2. Hộp ECU; 3. Bộ điều khiển can thiệp phanh thông qua ESP; 4. Công tắc điều khiển trên vô-lăng và màn hình hiển thị; 5. Can thiệp điều khiển động cơ bằng van tiết lưu điều chỉnh bằng điện; 6. Cảm biến tốc độ bánh xe

Ngày này đa phần các dòng xe ô tô từ hạng B trở lên như Toyota Vios, Hyundai Accent, Honda City, Mazda 2… đều được trang bị hệ thống Cruise Control.

b, Nguyên lý hoạt động của hệ thống Cruise Control

Hệ thống CCS hoạt động theo nguyên lý điều khiển hồi tiếp (Close-loop control), sơ đồ nguyên lý thể hiện như sau:



Hình 1.12 Sơ đồ điều khiển CCS

Tín hiệu đầu vào chủ yếu là tốc độ theo ý muốn của người lái và tốc độ thực của xe. Các tín hiệu quan trọng khác là sự điều chỉnh Faster-accel/Slower-coast của người lái, Resume, On/Off, công tắc phanh, và tín hiệu điều khiển động cơ. Tín hiệu đầu ra chủ yếu là trị số của bộ trợ lực điều khiển bướm ga, đèn báo ON của CCS.

Khi người lái bật Cruise Control, cảm biến tốc độ xe sẽ truyền tín hiệu về bộ điều khiển. Sau đó bộ điều khiển sẽ truyền lệch đến van chân không, van này kết nối trực tiếp với bướm ga. Van chân không sẽ điều khiển độ mở bướm ga phù hợp. Nhờ đó mà xe sẽ tự động duy trì tốc độ được cài đặt, người lái có thể nhìn tín hiệu đèn báo Cruise Control hiển thị trên bảng đồng hồ để biết chúng hoạt động hay chưa. Khi hệ thống Cruise Control đã hoạt động người lái có thể buông chân ga và điều khiển tốc độ hoàn toàn bằng nút tăng giảm được trang bị trên vô lăng.

Hệ thống điều khiển hành trình phù hợp sử dụng trong điều kiện có thể duy trì một tốc độ ổn định trên quãng đường dài như khi chạy xe trên đường trường hay đường cao tốc. Cruise Control được đánh giá mang đến nhiều lợi ích.

c, Ưu nhược điểm của hệ thống Cruise Control

Ưu điểm:

- Hệ thống Cruise Control giúp người lái giảm sự mệt mỏi khi điều khiển xe trên đường dài

- Tiết kiệm nhiên liệu hiệu quả vì ECU sẽ tự động tính lượng xăng phù hợp nhất (đối với xe sử dụng động cơ đốt trong) và tối ưu được hiệu suất pin (đối với xe điện)

- Giúp người lái kiểm soát được tốc độ xe, hạn chế vi phạm quy định giới hạn tốc độ.

Nhược điểm:

- Xe dễ bị trượt trên đường trơn vì không có sự điều khiển chủ động của người lái.

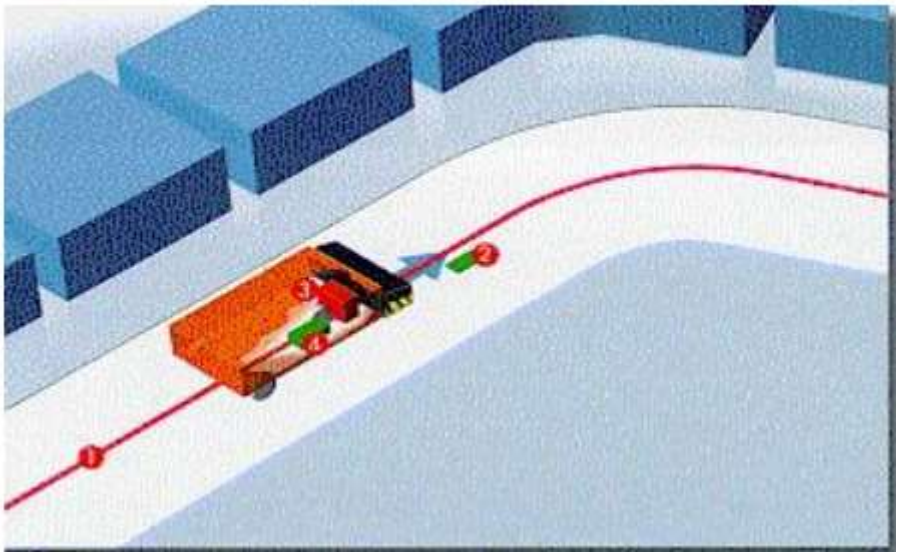
- Chỉ phù hợp trên đường cao tốc, ít xe cộ.

- Không phù hợp cho những người lái chưa có nhiều kinh nghiệm sử lý các tình huống bất chợt vì dễ gây ra tai nạn

1.3.2. Giới thiệu hệ thống dò line

a, Khái niệm

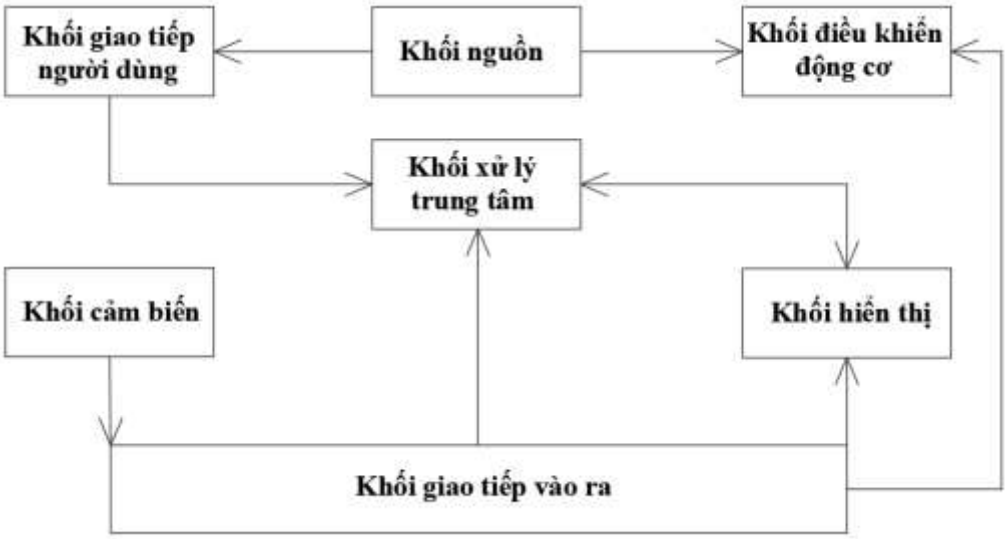
Mô hình dò đường là loại mô hình có thể di chuyển theo một quỹ đạo định sẵn được gọi là mô hình dò đường (mô hình tự động bám đường). Mô hình dò đường có thể di chuyển theo một đường, đường đi có thể được nhìn nhận như một dòng màu đen trên một bề mặt (hoặc ngược lại) hoặc nó có thể là đường vô hình như một từ trường.



Hình 1.13 Mô hình xe dò đường

b, Cấu tạo chung của mô hình dò đường

Với những chỉ tiêu về công nghệ và thống nhất các chức năng chính của mô hình ta tiến hành thiết kế và đưa ra sơ đồ khối chức năng và cấu tạo, hoạt động chi tiết của từng khối như mô ta dưới đây:

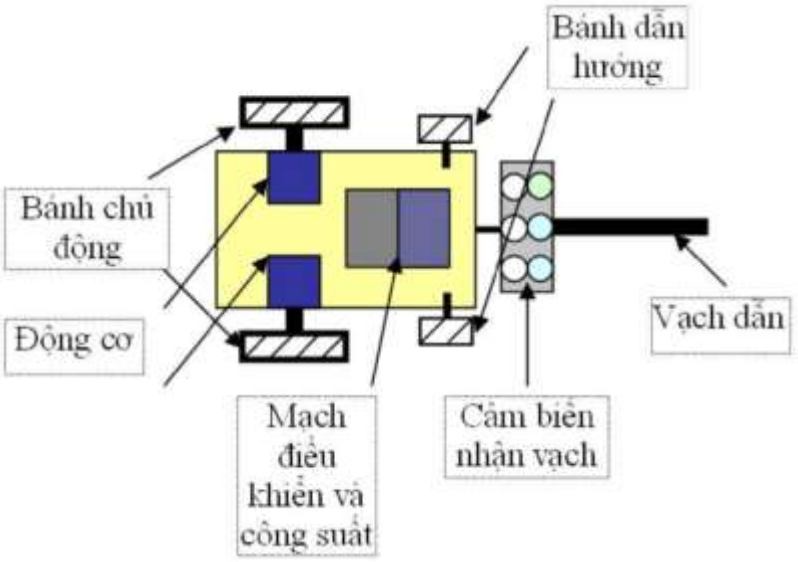


Hình 1.14 Cấu trúc cơ bản của mô hình dò đường

c, Nguyên tắc hoạt động của mô hình dò đường

Mô hình chuyển động theo một quỹ đạo được định trước nhờ vạch dẫn, hệ thống hai bánh được dẫn động bởi 2 động cơ điện một chiều thông qua mạch điều khiển và mạch công suất. Thường thì các vạch dẫn sẽ có màu khác với màu nền của quỹ đạo chuyển động. Để mô hình chuyển động đúng theo quỹ đạo cần có bộ phận cảm biến, bộ

phận này có nhiệm vụ phân biệt vạch dẫn và màu nền, đưa tín hiệu điện tương ứng về mạch điện điều khiển. Mạch điều khiển có nhiệm vụ thu nhận thông tin phản hồi từ bộ phận cảm biến từ đó điều khiển tốc độ và chiều quay của hai động cơ điện một chiều sao cho xe luôn bám và chuyển động theo vạch dẫn. Nhìn chung, về mặt cấu tạo thì mô hình dò đường được thể hiện theo sơ đồ sau đây:



Hình 1.15 Cấu tạo mô hình xe dò đường

1.3.3. Giới thiệu công nghệ V2V

a, Khái niệm

Giao tiếp giữa xe với xe (V2V) là một công nghệ phòng tránh va chạm, dựa vào việc trao đổi thông tin giữa các phương tiện ở gần để cảnh báo người lái về các tình huống nguy hiểm có thể dẫn đến va chạm.

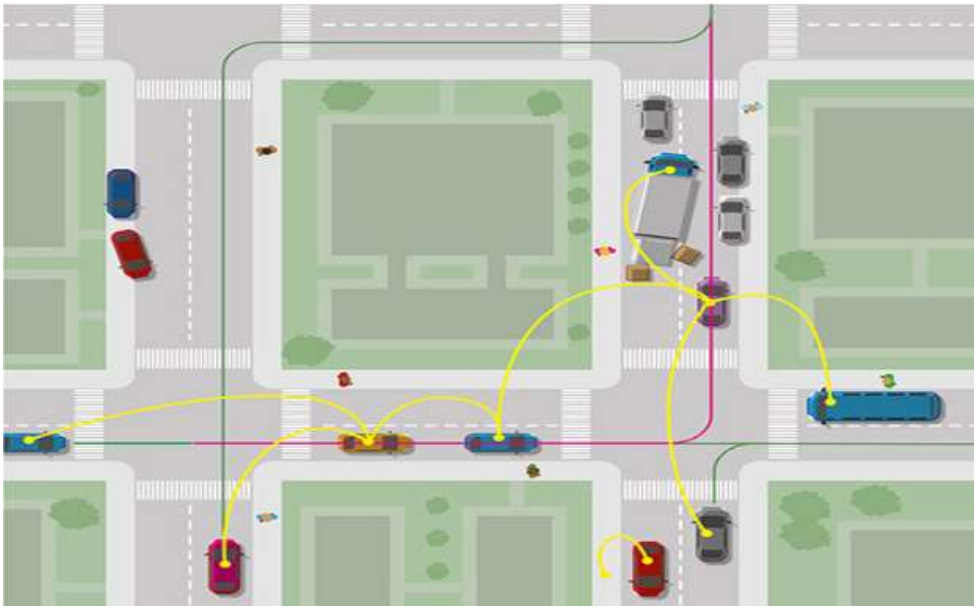
Hình 1.16 Trao đổi thông tin giữa 2 phương tiện

V2V cũng cho phép các xe trao đổi với nhau thông tin về tốc độ, vị trí và hướng đi của chúng. Công nghệ này cho phép các phương tiện phát và nhận thông tin đa hướng (tối đa 10 lần mỗi giây), tạo ra "nhận thức" 360 độ về các phương tiện khác ở gần. Các phương tiện có thể sử dụng thông báo từ những xe xung quanh để xác định mối đe dọa va chạm tiềm ẩn khi người lái chưa kịp phát hiện.

Sau đó, công nghệ này có thể sử dụng các cảnh báo bằng hình ảnh, xúc giác và âm thanh để cảnh báo lái xe. Những cảnh báo này cho phép lái xe nhận thức được sự nguy hiểm và thực hiện hành động để phòng tránh va chạm.

b, Nguyên lý hoạt động

V2V sẽ dựa trên mạng lưới ngang hàng, trong đó mỗi thành phần của mạng (một phương tiện) có thể tạo, nhận và chuyển tiếp tin nhắn. Với cách tiếp cận này, có thể tạo ra một mạng lưới mở rộng ở những khu vực đông dân cư mà không cần cơ sở hạ tầng đắt tiền. Thông thường, mỗi phương tiện sẽ có thể truyền thông tin về tốc độ, hướng đi, vị trí, phanh và ý định rẽ của chúng - mặc dù danh sách này có thể sẽ mở rộng theo thời gian.

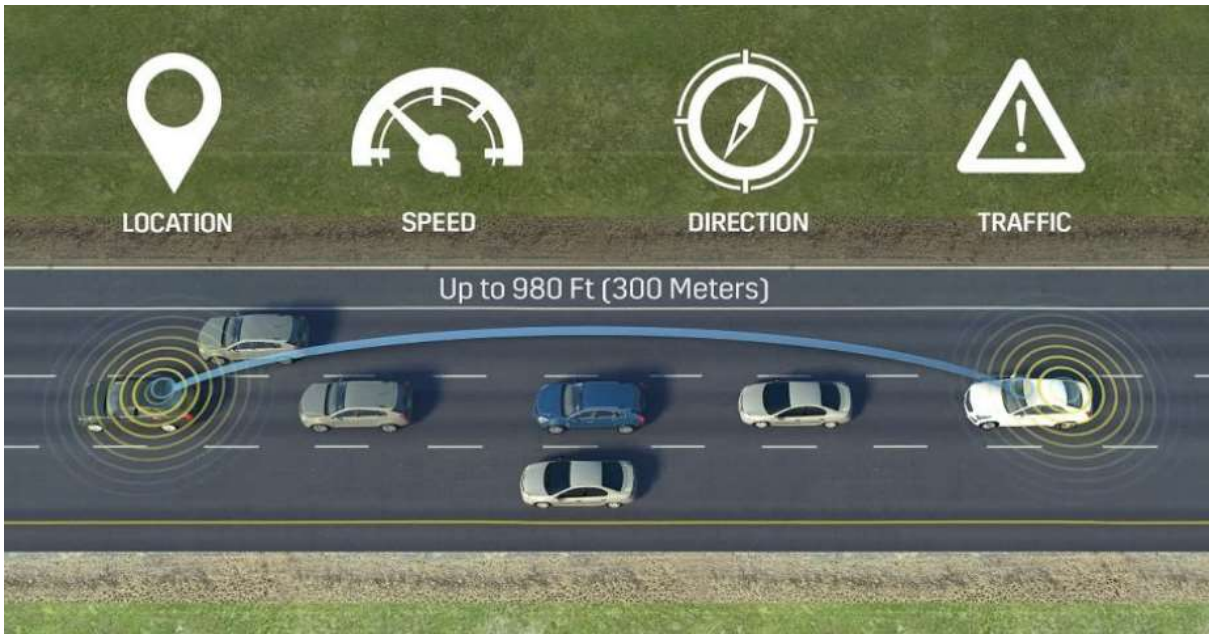


Hình 1.17 Hệ thống V2V sẽ tạo thành mạng lưới ở những khu vực đông đúc

Hệ thống liên lạc V2V bao gồm các thiết bị, được lắp đặt trong xe, sử dụng liên lạc vô tuyến tầm ngắn chuyên dụng (DSRC) để trao đổi tin nhắn/thông điệp chứa thông tin về xe (ví dụ: tốc độ, hướng đi, trạng thái phanh của xe). Các thiết bị V2V sử dụng thông tin này từ các phương tiện khác và xác định xem có cần cảnh báo cho người lái phương tiện đó hay không, điều này có thể ngăn ngừa va chạm phương tiện.

c, Những lợi thế đối với công nghệ V2V

Tin nhắn V2V có phạm vi khoảng 300 mét, vượt quá khả năng của các hệ thống có cảm biến siêu âm, camera và radar. Trong một số trường hợp, khoảng cách gần gấp đôi, cho phép có nhiều thời gian hơn để cảnh báo người lái xe. Ngoài ra, các tin nhắn vô tuyến này có thể "nhìn thấy" xung quanh các góc cua hoặc "xuyên qua" các phương tiện khác đang xử lý, chẳng hạn như tình huống mà một phương tiện đang tới xuất hiện từ phía sau một chiếc xe tải hoặc có thể từ một con hẻm cụt. Trong những tình huống đó, giao tiếp V2V có thể phát hiện mối đe dọa sớm hơn nhiều so với cảm biến radar hoặc camera.



Hình 1.18 Phạm vi giao tiếp giữa các xe

Ngoài ra, công nghệ V2V cũng có thể được kết hợp với radar và camera hiện có để mang lại những lợi ích thậm chí còn lớn hơn so với phương pháp đơn lẻ. Cách tiếp cận kết hợp này cũng có thể nâng cao độ chính xác của hệ thống, trở thành nền tảng để hiện thực hóa các phương tiện tự động trên các con đường.

1.4. Tình hình cải thiện tính an toàn hiệu quả

An toàn giao thông và hiệu suất di chuyển đang trở thành mối quan tâm hàng đầu khi tỷ lệ tai nạn ngày càng tăng, phần lớn do lỗi của người lái và tình trạng tắc nghẽn giao thông. Để giải quyết vấn đề này, công nghệ giao tiếp giữa các phương tiện (Vehicle-to-Vehicle – V2V) đã ra đời, giúp các xe có thể "nói chuyện" với nhau, từ đó giảm nguy cơ va chạm và cải thiện lưu thông trên đường. Trong thế giới thực, nơi các xe được trang bị một ăng-ten đơn giản, một chip máy tính và Công nghệ GPS (Hệ thống định vị toàn cầu) [5], xe của bạn sẽ biết các xe khác đang ở đâu, ngoài ra các xe khác cũng sẽ biết bạn đang ở đâu cho dù xe đang ở trong điểm mù, dừng phía trước trên đường cao tốc nhưng khuất tầm nhìn, quanh góc khuất hay bị các xe khác chặn. Các xe có thể dự đoán và phản ứng với các tình huống lái xe thay đổi và sau đó ngay lập tức cảnh báo người lái xe bằng các thông báo cảnh báo khẩn cấp. Nếu người lái xe không phản hồi thông báo cảnh báo, xe có thể tự dừng lại an toàn, tránh va chạm. Công nghệ này có thể phát hiện chướng ngại vật hoặc bất kỳ phương tiện nào khác thông qua sự kết hợp của các loại cảnh báo khác nhau, chẳng hạn như âm thanh, video hoặc xúc giác. Theo cách này, nó cảnh báo người lái xe về bất kỳ mối đe dọa tiềm ẩn nào hoặc thậm chí thực hiện các hành động xa hơn để ngăn chặn nó. Thông thường, các phương tiện liên lạc này có phạm vi 300 mét và có thể phát hiện bất kỳ thứ gì trên đường đi của chúng hoặc bất kỳ thứ gì nhận được từ một phương tiện khác khi nó nằm trong phạm vi này [7]. Công nghệ LoRa đóng vai trò quan trọng trong V2V nhờ phạm vi truyền tín hiệu rộng và khả năng chống nhiễu tốt. Khi kết hợp với hệ thống Cruise Control, V2V không chỉ giúp duy trì tốc độ ổn định mà còn tự động điều chỉnh khoảng cách với xe phía trước, đảm bảo hành trình mượt mà và an toàn hơn. Điều này đặc biệt hữu ích trong điều kiện giao thông đông đúc,

giúp giảm thiểu va chạm do phản ứng chậm của tài xế. Công nghệ V2V không chỉ giảm thiểu tai nạn mà còn cải thiện hiệu suất vận hành và bảo mật thông tin giữa các phương tiện. Bằng cách mã hóa dữ liệu truyền tải, V2V giúp đảm bảo an toàn thông tin và ngăn chặn nguy cơ xâm nhập từ bên ngoài. Với tiềm năng to lớn trong việc giảm thiểu tai nạn giao thông và nâng cao hiệu suất di chuyển, công nghệ V2V đang được HTSA và các nhà sản xuất ô tô hàng đầu tích cực nghiên cứu và triển khai. Trong tương lai gần, V2V sẽ trở thành tiêu chuẩn mới trong ngành công nghiệp ô tô, mang lại sự an toàn và tiện lợi tối đa cho người tham gia giao thông.

Hình 1.19 Sự kết hợp giữa hệ thống Cruise Control và công nghệ V2V

1.5. Kết luận chương

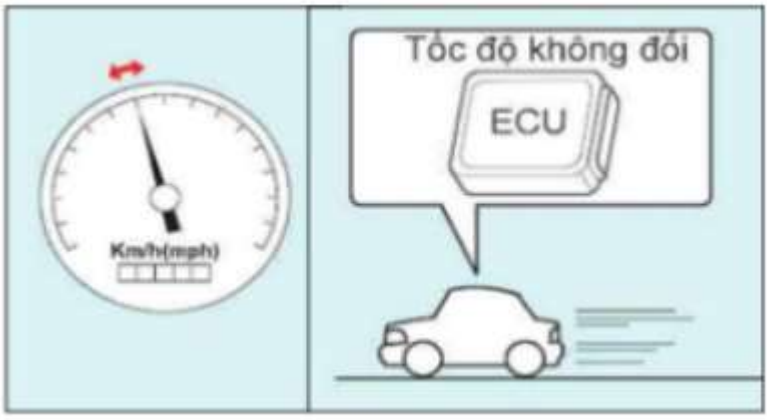
Xe điện đang dần trở thành xu hướng chủ đạo trong ngành công nghiệp ô tô nhờ những lợi ích về môi trường và hiệu suất vận hành. Sự phát triển mạnh mẽ của công nghệ đã thúc đẩy ngành ô tô điện với những cải tiến về pin, hệ thống truyền động và khả năng sạc nhanh, giúp mở rộng phạm vi hoạt động và nâng cao trải nghiệm người dùng. Bên cạnh đó, các hệ thống hiện đại như Cruise Control giúp duy trì tốc độ ổn định, hệ thống dò đường hỗ trợ điều hướng thông minh, và công nghệ giao tiếp V2V (Vehicle-to-Vehicle) giúp các phương tiện trao đổi dữ liệu để tăng cường an toàn và hiệu quả giao thông. Những tiến bộ này không chỉ định hình tương lai của xe điện mà còn mở ra kỷ nguyên mới của phương tiện thông minh và kết nối.

Chương 2: CƠ SỞ LÝ THUYẾT

2.1. Chức năng của hệ thống điều khiển ga tự động thích ứng trên xe điện.

2.1.1. Chức năng điều khiển tốc độ không đổi

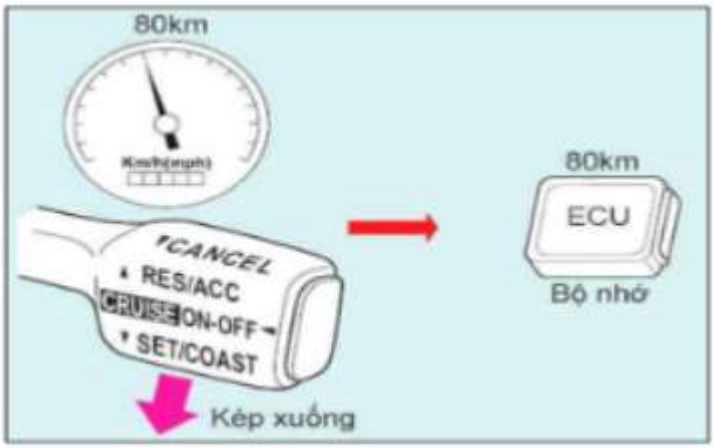
Hệ thống điều khiển ga tự động trong ECU điều khiển xe so sánh tốc độ thực tế với tốc độ đặt trước. Nếu tốc độ xe cao hơn tốc độ đặt trước, sẽ gửi tín hiệu điều khiển mô men xoắn động cơ điện thấp hơn đến bộ biến tần động cơ điện để giảm tốc độ xe. Nếu tốc độ xe cao hơn tốc độ đặt trước sẽ gửi tín hiệu điều khiển mô men xoắn động cơ điện cao hơn đến bộ biến tần động cơ điện để tăng tốc độ xe. Tín hiệu điều khiển mô men xoắn mục tiêu động cơ điện được cập nhật liên tục để đảm bảo tốc độ xe không đổi.



Hình 2.1 Chức năng điều kiển tốc độ không đổi

2.1.2. Chức năng cài đặt tốc độ (SET)

Khi công tắc SET/COAST được bật và nhả ra khi xe đang chạy trong dải điều khiển tốc độ của chế độ ga tự động (khoảng 40 đến 200 km/h với công tắc CRUISE bật), ECU điều khiển xe lưu giá trị này vào bộ nhớ và duy trì tại tốc độ đó khi nhả bàn đạp tăng tốc.



Hình 2.2 Chức năng cài đặt tốc độ

Khi công tắc CRUISE bật, kí hiệu Cruise control màu trắng sẽ hiển thị trên đồng hồ hiển thị điện tử. Khi cài đặt thành công trên đồng hồ hiển thị điện tử, kí hiệu Cruise

Control được hiển thị sẽ chuyển từ màu trắng sang màu xanh. Khi nhả bàn đạp tăng tốc, xe sẽ chạy ở tốc độ cài đặt.

Hình 2.3 Kí hiệu Cruise Control

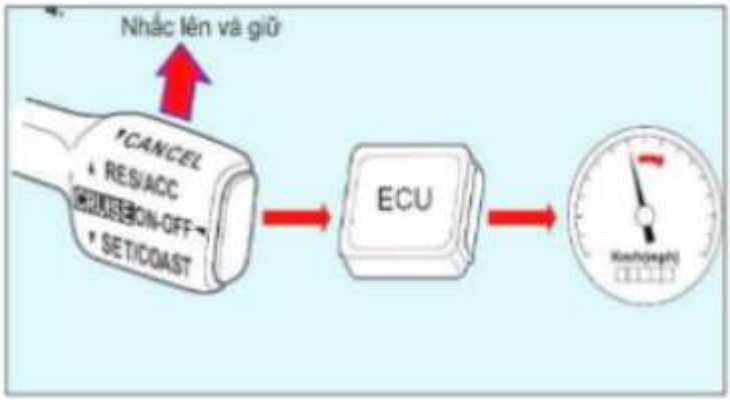
Một số trường hợp xe không thể cài đặt được:

- Công tắc CRUISE tắt.
- Tốc độ của xe dưới dải tốc độ của chế độ ga tự động (dưới 40 km/h).
- Cần số điều khiển điện tử không ở chế độ D.
- Công tắc phanh tay điện tử đang bật.
- Phanh đang được sử dụng bởi người lái.

2.1.3. Chức năng tăng tốc (ACCELERATE)

Khi muốn tăng tốc độ cài đặt trước, ta có 2 cách để thực hiện

Cách thứ nhật là sử dụng công tắc RES/ACC trong lúc chế độ ga tự động đã cài đặt. Nếu công tắc RES/ACC được bật và giữ, xe sẽ tăng tốc liên tục, nhả ra lập tức để tăng tốc độ cài đặt trước theo mức tăng cài đặt tuỳ theo nhà sản xuất, thông thường 10 km/h.



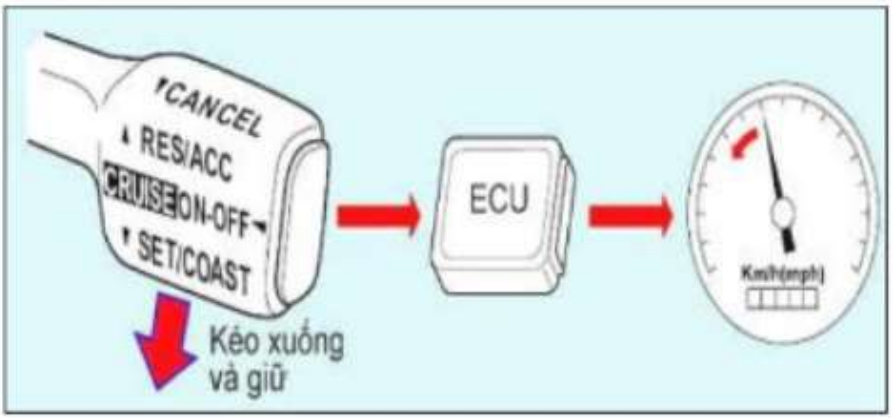
Hình 2.4 Sử dụng công tắc RES/ACC để tăng tốc độ

Cách thứ hai là sử dụng bàn đạp tăng tốc. Nhấn và giữ bàn đạp tăng tốc đến tốc độ mong muốn, rồi gạt công tắc SET/COAST và nhả ra ngay lập tức, ECU điều khiển xe lưu giá trị và bộ nhớ và duy trì tại tốc độ đó khi nhả bàn đạp tăng tốc.

2.1.4. Chức năng giảm tốc (COAST)

Khi muốn giảm tốc độ cài đặt trước, có 2 cách để thực hiện:

Cách thứ nhất là sử dụng công tắc SET/COAST trong lúc chế độ ga tự động đã cài đặt. Nếu công tắc RES/ACC được bật và giữ, xe sẽ giảm tốc liên tục, nhả công tắc SET/COAST khi xe đạt tốc độ mong muốn. Nếu công tắc SET/COAST được bật và nhả ra lập tức để giảm tốc độ cài đặt trước theo mức giảm cài đặt tuỳ theo nhà sản xuất, thông thường 10 km/h.

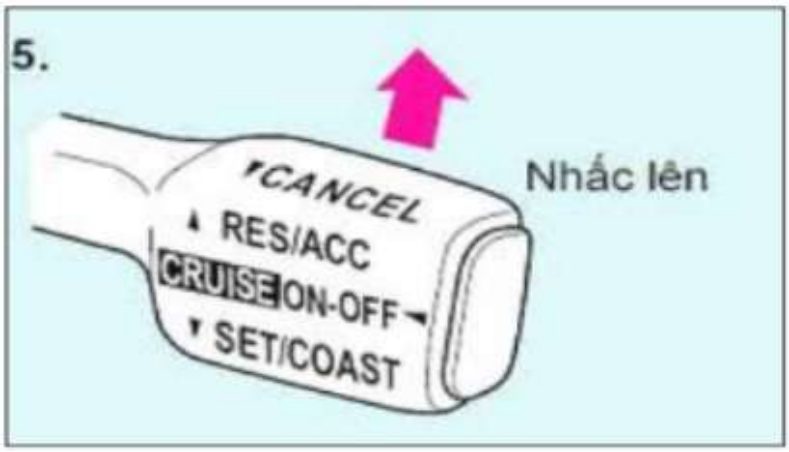


Hình 2.5 Sử dụng công tắc SET/COAST để giảm tốc độ

Cách thứ hai là sử dụng bàn đạp tăng tốc. Nhấn và giữ bàn đạp tăng tốc đến tốc độ mong muốn thấp hơn, rồi gạt công tắc SET/COAST và nhả ra lập tức, ECU điều khiển xe lưu giá trị này vào bộ nhớ và duy trì tại tốc độ đó khi nhả bàn đạp tăng tốc.

2.1.5. Chức năng phục hồi (RESUME)

Sau khi chế độ điều khiển chạy tự động bị huỷ, ta vẫn có thể phục hồi tốc độ đã cài đặt lần cuối trước khi chức năng CCS bị tạm huỷ bằng cách bật công tắc RES/ACC và nhả ra lập tức và xe phải có vận tốc trong dải tốc độ của chế độ ga tự động.



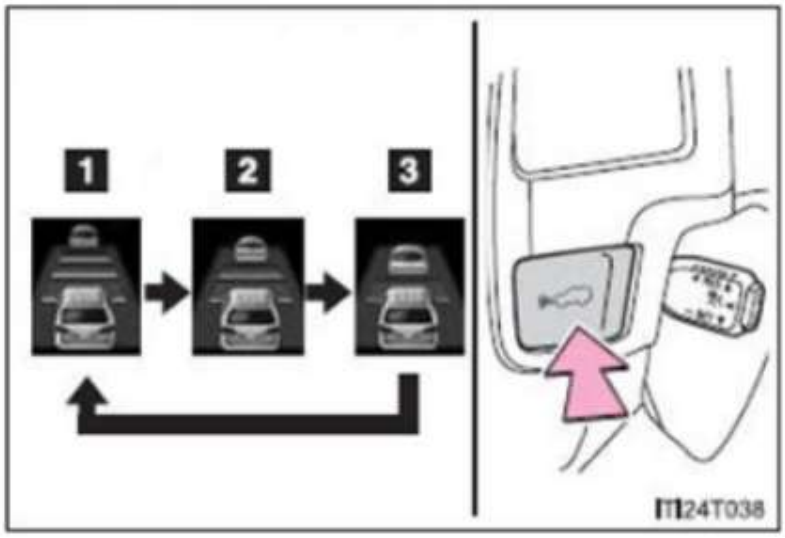
Hình 2.6 Sử dụng công tắc RES/ACC để phục hồi tốc độ

2.1.6. Chức năng giữ khoảng cách với phương tiện phía trước

Trong chế độ kiểm soát khoảng cách từ xe đến xe, CCS tự động duy trì khoảng cách với phương tiện đi phía trước dựa trên tốc độ xe phía trước đó (tối đa là tốc độ cài đặt) và xe di chuyển ở tốc độ cài đặt khi phía trước trống, hoặc phát hiện xe phía trước chuyển làn. Khi phát hiện một phương tiện chạy chậm hơn ở phía trước, CCS sẽ giảm

tốc độ xe xướng để đảm bảo khoảng cách gửi tín hiệu điều khiển mô men xoắn động cơ điện và gửi tín hiệu đến ABS ECU để phanh (khoảng 40% tổng lực phanh) nếu cần thiết.

Bật công tắc DISTANCE để thay đổi khoảng cách với xe phía trước với 3 mức xa (60m), trung bình (45m) và gần (30m).



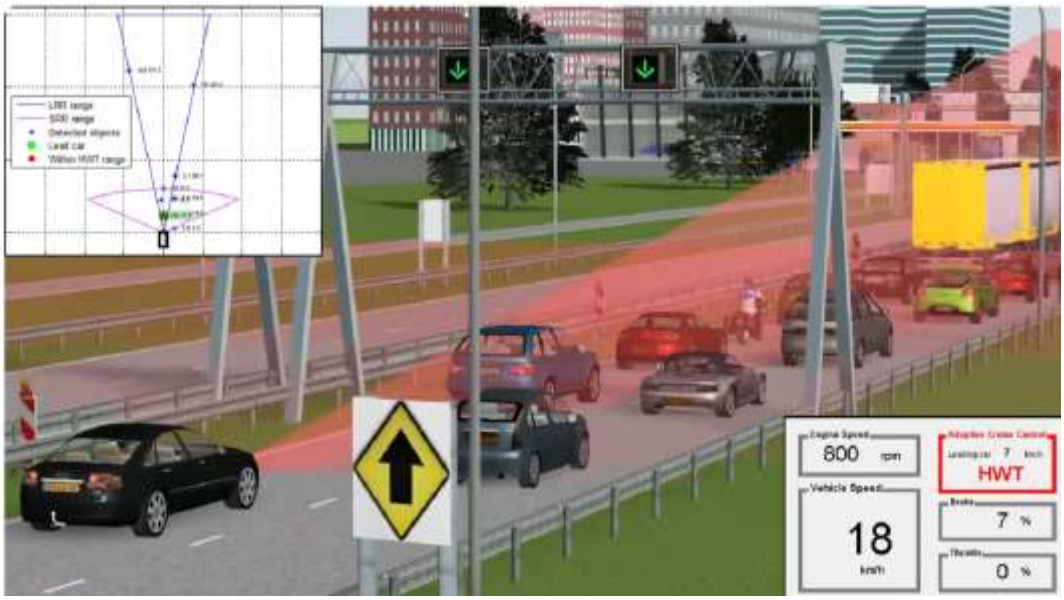
Hình 2.7 Sử dụng công tắc DISTANCE để thay đổi khoảng cách với xe phía trước

1. Mức xa; 2. Mức trung bình; 3. Mức gần

2.1.7. Chức năng Stop and Go

CCS Stop and Go có thể hoạt động ở tốc độ dưới 30 km/h. Chức năng này có thể duy trì khoảng cách thiết lập đến chiếc xe trước ngay cả ở tốc độ rất thấp và có thể giảm tốc độ dừng hoàn toàn.

Nếu chiếc xe có hộp số tự động và kẹt xe trong khoảng thời gian ngắn, CCS Stop and Go có thể thiết lập cho xe chuyển động lại. Khi xe dừng lại lâu hơn, người lái xe chỉ cần kích hoạt lại hệ thống, ví dụ như bằng cách nhấn nhẹ vào bàn đạp tăng tốc để trở lại kích hoạt CCS stop and go trở lại. Bằng cách này, CCS Stop and Go hỗ trợ điều khiển ngay cả khi giao thông đông đúc và ùn tắc giao thông.



Hình 2.8 Hình mô phỏng hoạt động hệ thống Cruise Control Stop and Go

2.1.8. Chức năng điều khiển giới hạn tốc độ thấp và tốc độ cao

Giới hạn tốc độ thấp là tốc độ thấp mà hệ thống chạy tự động có thể đặt được, xấp xỉ 40 km/h. Điều khiển chạy tự động không thể đạt dưới tốc độ này. Nếu tốc độ xe giảm xuống thấp hơn dải tốc độ điều khiển ga tự động do xe phải leo dốc hay mặt đường xấu, chế độ điều khiển ga tự động sẽ tự tạm huỷ. Khi tốc độ tăng lên cao hơn tốc độ giới hạn thấp thì ta có thể khôi phục lại hệ thống điều khiển ga tự động.

Giới hạn tốc độ cao là tốc độ cao nhất mà hệ thống điều khiển ga tự động có thể thiết lập tốc độ này ở khoảng 200 km/h. Hệ thống điều khiển ga tự động không thể thiết lập cao hơn tốc độ này nhờ công tắc ACCELERATE.

2.1.9. Chức năng huỷ thường

Khi ECU điều khiển chạy tự động nhận được bất kì tín hiệu nào sau đây trong khi xe đang chạy chế độ tự động, điều khiển ga tự động sẽ bị huỷ. ECU điều khiển xe sẽ ngắt tín hiệu điều khiển mô men xoắn mục tiêu động cơ điện đến bộ biến tần động cơ điện.

Chức năng huỷ thường được kích hoạt khi:

- Tín hiệu công tắc đèn phanh bật (đạp phanh)
- Tín hiệu công tắc bàn đạp phanh bật (đạp phanh)
- Tín hiệu từ cần số điều khiển (khi từ chế độ D trả về chế độ N)
- Tín hiệu công tắc CANCEL bật (gạt hoặc nhấn công tắc điều khiển)
- Tín hiệu công tắc CRUISE tắt. (gạt hoặc nhấn công tắc điều khiển)

2.1.10. Chức năng huỷ tự động

Khi có bất kỳ một trong các điều kiện sau xảy ra và xe đang chạy ở chế độ điều khiển chạy tự động, tốc độ đặt trước trong bộ nhớ bị xoá và điều kiện chạy tự động bị huỷ.

Bảng 2.1 Tín hiệu huỷ tự động

STT	Tín hiệu huỷ tự động
1	Tốc độ xe giảm xuống dưới hạn tốc độ thấp (xấp xỉ 40 km/h)
2	Tốc độ xe giảm thấp hơn tốc độ đặt trước khoảng 16 km/h (khi đang leo dốc)
3	Nguồn đến hệ thống điều khiển chạy tự động tạm thời bị ngắt quá 5 giây
4	Hở mạch trong dây điện công tắc đèn phanh hay bóng đèn phanh bị cháy
5	Tín hiệu cảm biến tốc độ xe không bình thường
6	Tín hiệu tốc độ xe không vào ECU trong một khoảng thời gian xác định (khoảng 140 giây)
7	Công tắc RESUME đã bật khi công tắc chính bật
8	Khi có ngắn mạch trong công tắc điều khiển hay công tắc không bình thường
9	Tín hiệu ra bộ biến tần động cơ điện của bộ vi xử lý (bên trong ECU) không bình thường
10	Các tín hiệu vào từ công tắc điều khiển không bình thường

2.1.11. Chức năng chuẩn đoán

Có các chức năng chẩn đoán sau:

- Đèn báo: thông báo cho lái xe nếu có hư hỏng xảy ra trong hệ thống ga tự động
- Báo mã chẩn đoán: thông báo cho kỹ thuật viên bản chất của hư hỏng
- Chức năng kiểm tra tín hiệu vào: cho phép kỹ thuật viên kiểm tra tình trạng của các tín hiệu nhập vào ECU điều khiển xe từ cảm biến và công tắc.
- Chức năng kiểm tra tín hiệu huỷ: thông báo cho kỹ thuật viên tín hiệu đầu vào nào đã huỷ điều khiển chạy tự động lần cuối.

2.2. Chức năng của công nghệ giao tiếp V2V trên xe điện

V2V hoạt động nhờ sử dụng các tín hiệu không dây để trao đổi thông tin về hướng đi, tốc độ, vị trí… giữa các xe. Nhờ vậy mà các phương tiện có thể giữ một khoảng cách an toàn, giảm nguy cơ xảy ra va chạm.

Công nghệ Vehicle-to-Vehicle (V2V) cho phép các xe điện trao đổi dữ liệu với nhau theo thời gian thực nhằm nâng cao an toàn và hiệu suất giao thông. Dưới đây là một số chức năng chính:

- Cảnh báo va chạm: Xe có thể nhận thông tin từ các phương tiện xung quanh về tốc độ, hướng di chuyển và vị trí, giúp cảnh báo người lái về nguy cơ va chạm sớm hơn.

- Điều phối giao thông thông minh: Hệ thống V2V giúp các xe tự động điều chỉnh tốc độ và khoảng cách để giảm ùn tắc và tối ưu luồng giao thông.

- Hỗ trợ vượt xe an toàn: Xe có thể nhận tín hiệu từ phương tiện phía trước hoặc đối diện để xác định thời điểm vượt an toàn, giảm nguy cơ tai nạn.

- Cải thiện hiệu suất hệ thống Cruise Control: Khi kết hợp với công nghệ điều khiển hành trình thích ứng (Adaptive Cruise Control - ACC), V2V giúp xe điện tự động điều chỉnh tốc độ theo xe phía trước một cách mượt mà hơn.

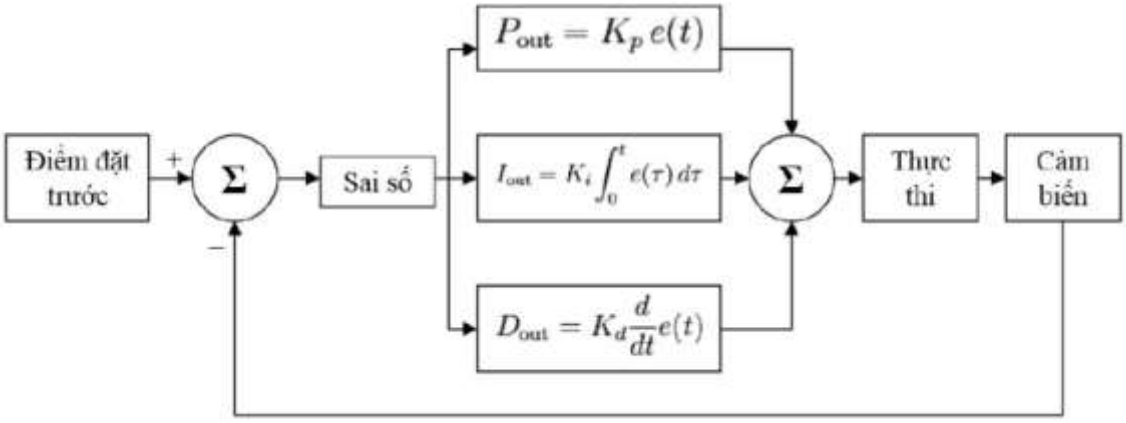
- Cảnh báo phanh gấp: Khi một xe phía trước đột ngột phanh gấp, tín hiệu V2V có thể truyền ngay lập tức đến các xe phía sau để phản ứng kịp thời, tránh tai nạn liên hoàn.

- Hỗ trợ tại giao lộ: V2V giúp các phương tiện dự đoán và tránh va chạm tại các ngã tư bằng cách chia sẻ thông tin về quyền ưu tiên và trạng thái di chuyển của xe khác.

2.3. Lý thuyết thuật toán điều khiển PID

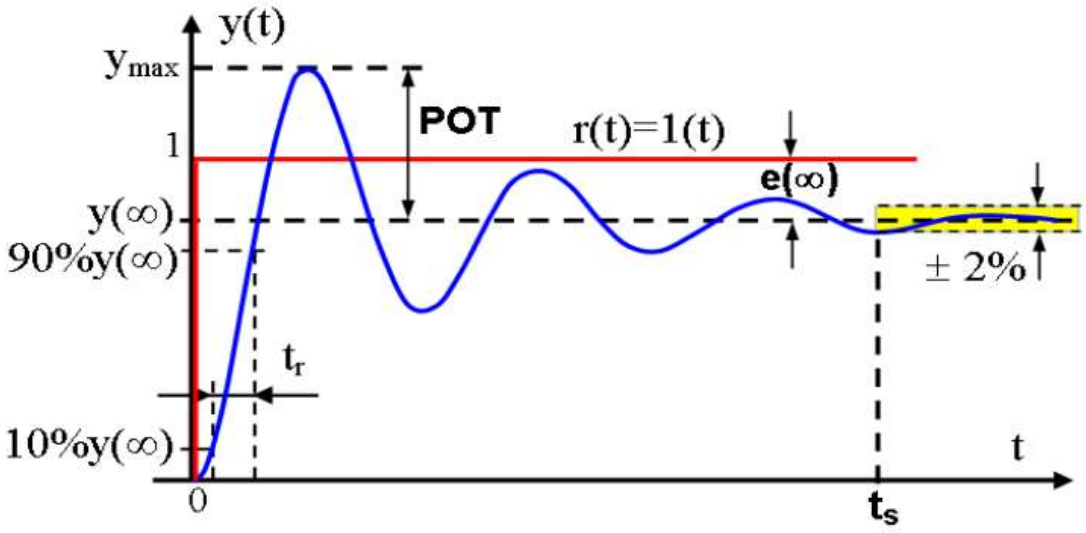
2.3.1. Lý thuyết cơ sở PID

Một bộ điều khiển PID (Proportional Integral Derivative) là một cơ chế phản hồi vòng điều khiển (bộ điều khiển) tổng quát được sử dụng rộng rãi trong các hệ thống điều khiên công nghiệp. Bộ điều khiên PID là bộ điều khiển được sử dụng nhiều nhất trong các bộ điều khiển phản hồi. Bộ điều khiển PID sẽ tính toán giá trị "sai số" là giá trị đo thông số biến đổi và giá trị đặt mong muốn. Bộ điều khiển sẽ thực hiện giảm tối đa sai số bằng cách điều chỉnh giá trị điều khiển đầu vào. Trong trường hợp không có kiến thức cơ bản (mô hình toán học) về hệ thống điều khiển thì bộ điều khiển PID là bộ điều khiển tốt nhất. Tuy nhiên, để đạt được kết quả tốt nhất, các thông số PID sử dụng trong tính toán phải điều chỉnh theo tính chất của hệ thống-trong khi kiểu điều khiển là giống nhau, các thông số phải phụ thuộc vào đặc thù của hệ thống.



Hình 2.10 Sơ đồ khối của bộ điều khiển PID

Giá trị đặt là giá trị mong muốn đạt được tại ngõ ra của đối tượng điều khiển Process variable (Biến quá trình) là tín hiệu hồi tiếp mà bộ điều khiển nhận được từ đối tượng điều khiển. Control variable (biến điều khiển) là giá trị ngõ ra của bộ điều khiển. Error là giá trị sai lệch giữa giá trị đặt và giá trị hiện tại ở ngõ ra của đối tượng điều khiển và bộ điều khiển PID luôn đưa ra tín hiệu điều khiển CV dựa trên giá trị của error. Output là giá trị ngõ ra của đối tượng điều khiển.



Hình 2.11 Đáp ứng hệ thống điều khiển sử dụng bộ điều khiển PID

Triệt tiêu sai số xác lập (Steady-State Error): Hệ thống cần đảm bảo rằng sai số giữa tín hiệu đầu ra và tín hiệu đặt (setpoint) tiến dần về 0 khi đạt trạng thái ổn định. Điều này đồng nghĩa với việc hệ thống có khả năng theo sát giá trị mong muốn trong thời gian dài mà không bị lệch, đảm bảo độ chính xác trong điều khiển.

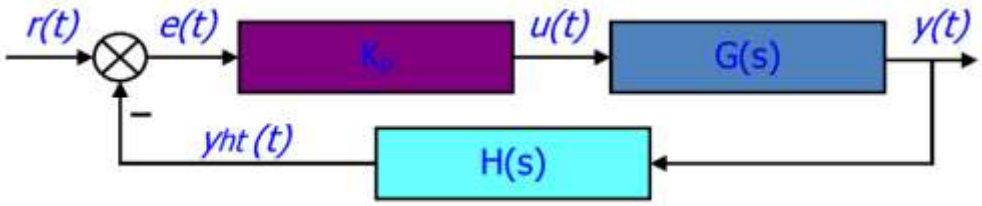
Giảm thời gian đáp ứng (Response Time): Thời gian xác lập (settling time – ký hiệu là t□) là thời gian cần thiết để đầu ra ổn định trong một khoảng sai số cho phép (ví dụ: ±2%). Hệ thống cần được điều chỉnh để rút ngắn thời gian này, từ đó cải thiện tốc độ phản ứng trước sự thay đổi tín hiệu vào, tăng khả năng thích nghi với các điều kiện vận hành thực tế.

Giảm độ vượt lố (Overshoot): Độ vượt lố (*POT*) là phần đầu ra vượt quá giá trị mong muốn trong quá trình chuyển tiếp. Việc giảm độ vượt lố giúp tránh tình trạng quá điều khiển, làm cho hệ thống trở nên an toàn hơn, nhất là đối với các ứng dụng có tính chất chính xác hoặc liên quan đến cơ cấu cơ khí.

Hạn chế dao động (Oscillation): Một hệ thống có dao động nhiều sẽ dễ mất ổn định và không đảm bảo được đầu ra mong muốn. Do đó, việc giảm biên độ dao động giúp hệ thống vận hành êm ái hơn, tránh gây ra mài mòn cơ khí hoặc dao động không mong muốn trong các bộ phận liên quan.

Cải thiện tốc độ tăng (Rise Time – t_r): Tốc độ tăng là thời gian để đầu ra từ 10% đến 90% giá trị cuối cùng. Mục tiêu là đảm bảo thời gian tăng nhanh để hệ thống đạt đáp ứng nhanh mà vẫn giữ được sự ổn định.

*** Hoạt động của khâu P (khâu tỉ lệ):**



Hình 2.12 Hoạt động của khâu tỷ lệ

Khâu này luôn đưa ra tín hiệu điều khiển luôn tỉ lệ với giá trị sai lệch.

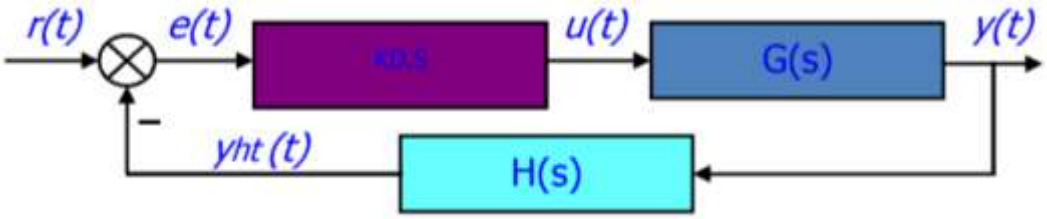
Hàm truyền của bộ điều khiển P: $K(s) = K_P$ (2.1)

Hàm truyền đặc tính thời gian: $Y(s) = K_P.G(s).E(s)$ (2.2)

Hàm truyền sai số hệ thống: $E(s) = \dfrac{R(s)}{1 + K_P.G(s).H(s)}$ (2.3)

K_P càng lớn thì tốc độ đáp ứng càng nhanh. K_P càng lớn thì sai số xác lập càng nhỏ. K_P tăng quá lớn thì vọt lố càng cao, nếu tăng nữa thì hệ thống mất ổn định và dao động không tắt dần.

*** Hoạt động của khâu D (khâu vi phân):**



Hình 2.13 Hoạt động của khâu D

Đây là khâu luôn đưa ra tín hiệu điều khiển tỉ lệ với tốc độ thay đổi của giá trị sai lệch.

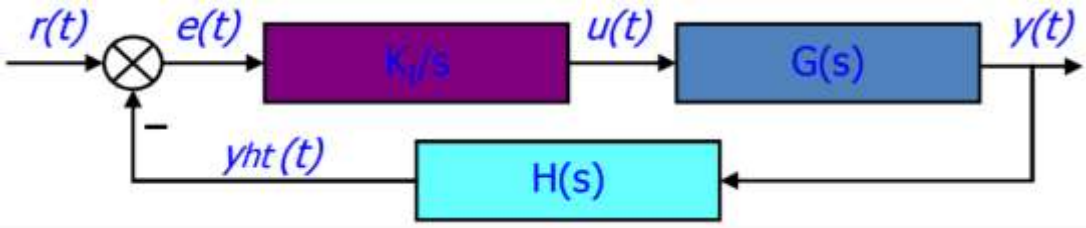
Hàm truyền của bộ điều khiển D: $K(s) = K_d.s$ (2.4)

Hàm truyền đặc tính thời gian: $Y(s) = K_d.G(s).E(s).s$ (2.5)

Hàm truyền sai số hệ thống: $E(s) = \dfrac{R(s)}{1 + s.K_d.G(s).H(s)}$ (2.6)

Khâu vi phân không thể sử dụng một mình mà phải sử dụng kết hợp với khâu P hoặc khâu I. K_d càng lớn thì độ vọt lố càng nhỏ. K_d càng lớn thì đáp ứng quá độ càng nhanh. Khâu D rất nhạy với nhiễu tần số cao do hệ số khuyết đại tại tần số cao.

*** Hoạt động của khâu I (khâu tích phân):**



Hình 2.14 Hoạt động của khâu I

Là khâu đưa ra tín hiệu điều khiển dựa trên giá trị của sai lệch và thời gian xảy ra sai lệch.

Hàm truyền của bộ điều khiển D: $K(s) = \dfrac{K_i}{s}$ (2.7)

Hàm truyền đặc tính thời gian: $Y(s) = \dfrac{K_i.G(s).E(s)}{s}$ (2.8)

Hàm truyền sai số hệ thống: $E(s) = \dfrac{s.R(s)}{1 + K_i.G(s).H(s)}$ (2.9)

K_i càng lớn thì đáp ứng quá độ càng chậm. K_i càng lớn thì sai số xác lập càng nhỏ. Đặc biệt hệ số khuyết đại của khâu tích phân bằng vô cùng khi tần số bằng 0, triệt tiêu sai số xác lập với hàm nấc. K_i càng lớn độ vọt lố càng cao.

Bộ điều khiển PID là sự kết hợp của các khâu P,I,D do đó khắc phục được nhược điểm của mỗi khâu riêng lẻ.

Như vậy, ngõ ra của bộ điều khiển PID thể hiện qua hàm truyền:

$$u_{PID}(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} \quad (2.10)$$

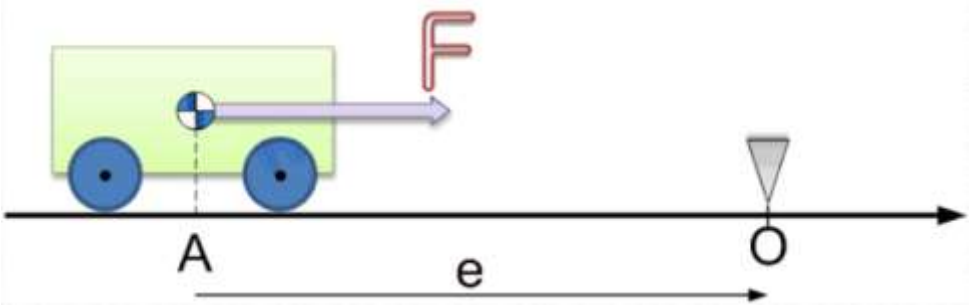
Với: $u_{PID}(t)$ là tín hiệu điều khiển.

$e(t)$ = Giá trị đặt – Giá trị thực tế.

K_p, K_i, K_d là các hệ số điều chỉnh.

2.3.2. Phân tích giải thuật điều khiển PID

PID là cách viết tắt của các từ Propotional (tỉ lệ), Integral (tích phân) vàDerivative (đạo hàm). Tuy xuất hiện rất lâu nhưng đến nay PID vẫn là giải thuật điềukhiển được dùng nhiều nhất trong các ứng dụng điều khiển tự động. Để giúp hiểu rõ hơn bản chất của giải thuật PID, hãy theo dõi ví dụ điều khiển vị trí của một xe trên đường thẳng. Giả sử bạn có một xe (đồ chơi...) có gắn một động cơ DC. Động cơ sinh ra một lực để đẩy xe chạy tới hoặc lui trên một đường thẳng như trong hình 2.15.



Hình 2.15 Mô hình xe đẩy – vị trí mục tiêu và sai số điều khiển

Gọi F là lực do động cơ tạo ra điều khiển xe. Ban đầu xe ở vị trí A, nhiệm vụđặt ra là điều khiển lực F (một cách tự động) để đẩy xe đến đúng vị trí o với các yêucầu: chính xác (accurate), nhanh (fast response), ổn định (small overshot)

Một điều rất tự nhiên, nếu vị trí hiện tại của xe rất xa vị trí mong muốn (điểmO), hay nói cách khác sai số (error) lớn, chúng ta cần tác động lực F lớn để nhanh chóng đưa xe về O. Một cách đơn giản để công thức hóa ý tưởng này là dùng quan hệtuyến tính:

$$F = K_p * e \quad (2.11)$$

Trong đó K_p là một hằng số dương nào đó mà chúng ta gọi là hệ số p(Propotional gain), e là sai số cần điều khiển tức khoảng cách từ điểm O đến vị trí hiện tại của xe. Mục tiêu điều khiển là đưa e tiến về 0 càng nhanh càng tốt. Rõ ràng nếu Kp lớn thì F cũng sẽ lớn và xe rất nhanh chóng tiến về vị trí O. Tuy nhiên, lực F quá lớn sẽ gia tốc cho xe rất nhanh (định luật II của Newton: F = ma). Khi xe đã đến vị trí o (tức = 0), thì tuy lực F = 0 (vì $F = K_p * e$) nhưng do quán tính xe vẫn tiếp tục tiến vềbên phải và lệch điểm O về bên phải, sai số e lại trở nên khác 0, giá trị sai số lúc nàyđược gọi là overshot (vượt quá). Lúc này, sai số e là số âm, lực F lại xuất hiện nhưngvới chiều ngược lại để kéo xe về lại điếm O. Nhưng một lần nữa, do K_p lớn nên giá trị lực F cũng lớn và có thể kéo xe lệch về bên trái điểm O. Quá trình cứ tiếp diễn, xe cứ mãi dao động quanh điểm O. Có trường hop xe dao động càng ngày xàng xa điểm O. Bộ điêu khiên lúc này được nói là không ôn định. Một đề xuất nhằm giảm overshot của xe là sử dụng một thành phần "thắng" trong bộ điều khiển. Sẽ rất lý tưởng nếu khi xe đang ở xa điểm O, bộ điều khiển sinh ra lực F lớn nhưng khi xe đã tiến gần đến điểm O thì thành phần "thắng" sẽ giảm tốc độ xe lại. Chúng ta đều biết khi một vật dao động quanh 1 điểm thì vật đó có vận tốc cao nhất ở tâm dao động (điểm O). Nói một cách khác, ở gần điểm O sai số e của xe thay đổi nhanh nhất (cần phân biệt: e thay đổinhanh nhất không phải e lớn nhất). Mặt khác, tốc độ thay đổi của e có thể tính bằngđạo hàm của biến này theo thời gian. Như vậy, khi xe từ A tiến về gần O, đạo hàm của sai số e tăng giá trị nhưng ngược chiều cùa lực F (vì e đang giảm nhanh dần). Nếu sử dụng đạo hàm làm thành phần "thắng" thì có thể giảm được overshot của xe. Thành phần "thắng" này chính là thành phần D (Derivative) trong bộ điều khiển PID mà chúng ta đang khảo sát. Thêm thành phần D này vào bộ điều khiển p hiện tại, chúng ta thu được bộ điều khiển PD như sau:

$$F = K_p * e + K_d * \frac{de}{dt} \tag{2.12}$$

Trong đó (de/dt) là vận tốc thay đổi của sai số e và Kd là một hằng số không âm thắng được lực ma sát tĩnh. Bạn hãy tưởng tượng tình huống bạn dùng sức của mình để đẩy một xe tải nặng vài chục tấn, tuy lực đẩy tồn tại nhưng xe không thể di chuyển. Như thế, xe sẽ đứng yên mãi dù sai số e vẫn chưa bằng 0. Sai số e trongtình huống này gọi là steady state error xảy ra, 2 thành phần P và D mất tác dụng, thành phần điều khiển mới sẽ "cộng dồn" sai số theo thời gian và làm tăng lực F theothời gian. Đến một lúc nào đó, lực F đủ lớn để thắng ma sát tĩnh và đẩy xe tiến tiếp vềđiểm O. Thành phần "cộng dồn" này chính là thành phần I (Integral - tích phân) trong bộ điều khiển PID. Vì chúng ta điều biết, tích phân một đại lượng theo thời gian chínhlà tổng của đại lượng đó theo thời gian. Bộ điều khiển đến thời điểm này đã đầy đủ làPID:

$$F = K_p * e + K_d * \frac{de}{dt} + K_i * \int e dt \tag{2.13}$$

Như vậy, chức năng của từng thành phần trong bộ điều khiển PID giờ đã rõ.Tùy vào mục đích và đối tượng điều khiển mà bộ điều khiển PID có thể được lượt bớt để trở thành bộ điều khiển P, PI hoặc PD. Công việc chính của người thiết kế bộ điều khiển PID là chọn các hệ số Kp, Kd và Ki sao cho bộ điều khiển hoạt động tốt và ổn định (quá

trình này gọi là PID gain tuning). Đây không phải là việc dễ dàng vì nó phụ thuộc vào nhiều yếu tố.

2.3.3. Kinh nghiệm chọn Kp, Kd, Ki được thực hành

- Chọn Kp trước: thử bộ điều khiển p với đối tượng thật (hoặc mô phỏng), điều chỉnh Kp sao cho thời gian đáp ứng đủ nhanh, chấp nhậnovershot nhỏ.

- Thêm thành phần D để loại overshot, tăng Kd từ từ, thử nghiệm và chọn giá trị thích hợp. Steady state error có thể sẽ xuất hiện.

- Thêm thành phần I để giảm steady state error. Nên tăng Ki từ bé đến lớn để giảm steady state error đồng thời không để cho overshot xuất hiện trở lại.

2.4. Kết luận chương

Chương này đã trình bày tổng quan về hệ thống điều khiển ga tự động thích ứng trên xe điện, trong đó xe có thể duy trì tốc độ và khoảng cách an toàn bằng cách sử dụng công nghệ cảm biến và thuật toán điều khiển. Bên cạnh đó, công nghệ giao tiếp V2V (Vehicle-to-Vehicle) đóng vai trò quan trọng trong việc truyền tải thông tin giữa các phương tiện, hỗ trợ hệ thống kiểm soát hành trình thích ứng (CCS) để tối ưu hóa hiệu suất vận hành.

Trong phạm vi nghiên cứu, thuật toán điều khiển PID được áp dụng nhằm điều chỉnh tốc độ động cơ dựa trên dữ liệu từ cảm biến, đảm bảo khả năng bám đuổi và phản ứng nhanh với các thay đổi của xe dẫn đầu. Công nghệ truyền thông LoRa được sử dụng như một phương án giao tiếp không dây, giúp trao đổi dữ liệu giữa các phương tiện với độ trễ thấp và phạm vi truyền xa.

Cuối cùng, sơ đồ khối hệ thống đã được xây dựng nhằm mô tả chi tiết cách thức hoạt động và tương tác giữa các thành phần, bao gồm vi điều khiển, cảm biến, động cơ và mô-đun truyền thông. Các phương án điều khiển PID cũng đã được phân tích nhằm tối ưu hóa độ chính xác và độ ổn định của hệ thống. Những nội dung này tạo tiền đề quan trọng cho việc triển khai và phát triển các hệ thống điều khiển xe tự hành thông minh trong tương lai.

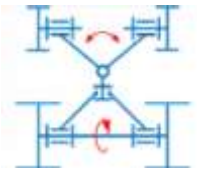
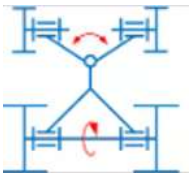
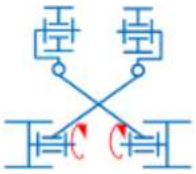
Chương 3: THIẾT KẾ MÔ HÌNH XE ĐIỆN VÀ MÔ HÌNH ĐƯỜNG

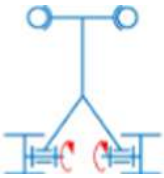
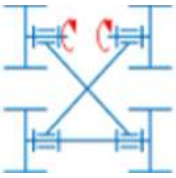
3.1. Ưu nhược điểm của mô hình xe điện

Bài toán mô hình xe được giải quyết bằng nhiều loại cấu hình khác nhau. Trong đó phương án dẫn động 3 và 4 bánh với bánh trước hoặc sau tự lựa được dùng phổ biến nhất. Dù cơ cấu nào đi nữa, nó cũng phải giải quyết được khả năng cân bằng của robot và sự tiếp xúc của bánh xe với mặt đường đặt biệt khi bo các vòng cua có bán kính nhỏ. Ưu nhược điểm của một số loại xe 3, 4 bánh được thể hiện trong (bảng 3.1).

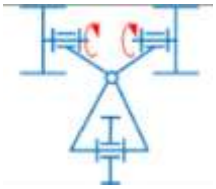
a, Cấu tạo

Bảng 3.1 Đánh giá ưu, nhược điểm của mô hình xe điện

Xe 4 bánh			
Mô hình			
Bám đường	Tốt: Có kết cấu khớp bản lề	Không tốt	Không tốt
Vào cua	Khó: Do hiện tượng trượt	Khó	Dễ: Kết cấu bánh tự lựa
Điều khiển	Không phức tạp: Lái và chuyển động tách biệt	Không phức tạp	Phức tạp: Lái và chuyển động kết hợp
Kết cấu	Phức tạp	Đơn giản	Đơn giản

Mô hình			
Bám đường	Không tốt	Không tốt	Tốt: Có kết cấu khớp nối
Vào của	Dễ	Trung bình	Khó
Điều khiển	Phức tạp	Phức tạp	Không phức tạp
Điều khiển	Đơn giản	Đơn giản	Phức tạp

Xe 3 bánh

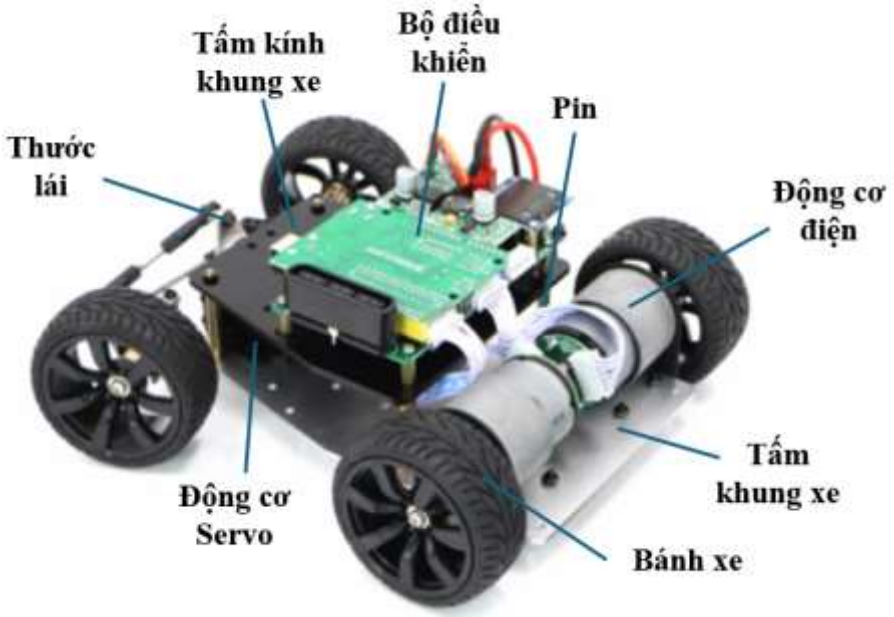
Mô hình		
Bám đường	Tốt	Tốt
Vào cua	Khó	Dễ
Điều khiển	Không phức tạp	Phức tạp
Kết cấu	Đơn giản	Đơn giản

b, Về số lượng bánh xe

Số bánh	**4 bánh**	**3 bánh**
Đồng phẳng	Khó đảm bảo đồng phẳng	Luôn đồng phẳng
Độ ổn định khi có vật cản	Vẫn giữ được độ ổn định	Khó giữ được độ ổn định
Lật khi vào cua	Khó	Dễ
Ma sát	Nhiều	Ít

3.2. Cấu tạo và nguyên lý làm việc của mô hình xe điện

3.2.1. Cấu tạo mô hình xe điện



Hình 3.1 Cấu tạo mô hình xe điện

Động cơ DC: Đóng vai trò truyền động chính để xe di chuyển về phía trước hoặc lùi lại tùy theo điều khiển.

Động cơ servo điều chỉnh lái: Được sử dụng để điều khiển góc quay của bánh xe trước nhằm cho xe thay đổi hướng di chuyển.

Thước lái: Là một thanh liên kết bằng nhựa hoặc kim loại nối giữa càng bánh xe trái và phải để truyền chuyển động từ servo đến bánh xe.

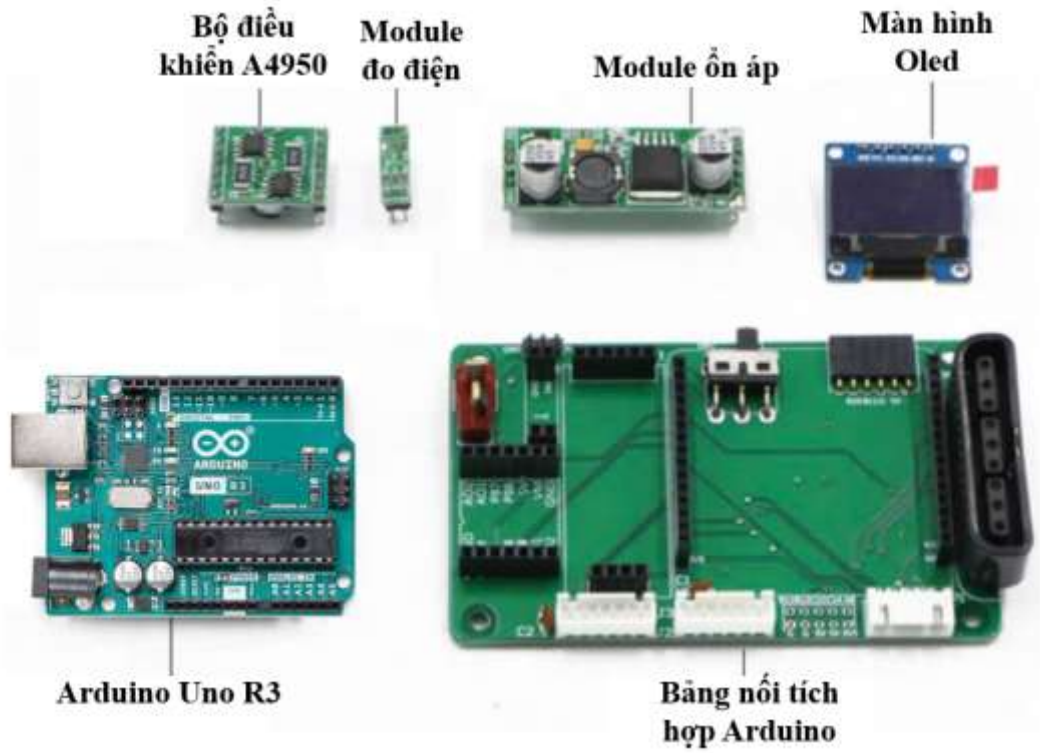
Tấm khung sườn xe: Nhiều tầng (2 tầng chassis) giúp tách biệt nguồn – mạch điều khiển – cảm biến; tăng diện tích gắn linh kiện.

Bánh xe: Được thiết kế đơn giản nhưng vẫn đảm bảo chức năng cơ bản: truyền động, chịu lực, và đảm bảo tiếp xúc mặt đường.

Pin và bộ sạc pin: Pin trong ô tô điện là bộ phận dùng để lưu trữ năng lượng cho xe chạy. Để xe điện có thể vận hành được thì pin phải được sạc đầy. Loại pin được sử dụng để lắp đặt trong ô tô điện là lithium. Sử dụng pin loại này vì nó có tỷ lệ xả thải thấp, ít gây ô nhiễm môi trường. Bộ sạc pin với chức năng kiểm soát mức điện áp của pin. Việc này được thực hiện thông qua điều chỉnh tốc độ sạc trên ô tô. Hơn thế nữa, bộ sạc pin có có khả năng theo dõi nhiệt độ của pin giúp duy trì tuổi thọ của pin.

Bộ điều khiển tích hợp trên mô hình xe điện là trung tâm điều hành toàn bộ hoạt động của xe. Nó bao gồm các phần cứng và phần mềm để điều khiển động cơ, nhận dữ liệu cảm biến, xử lý tín hiệu, và thực hiện các thuật toán (như line-following, tránh vật, giao tiếp,...).

3.2.2. Cấu tạo bộ điều khiển tích hợp



Hình 3.2 Bộ điều khiển tích hợp

Chip điều khiển động cơ A4950: Là mạch cầu H chuyên dụng, cho phép điều chỉnh tốc độ và hướng quay của động cơ DC thông qua điều chế xung PWM từ vi điều khiển.

Vi điều khiển (Arduino UNO): Đóng vai trò trung tâm xử lý, nhận tín hiệu từ cảm biến và điều khiển các bộ phận khác như động cơ, servo và chip điều khiển.

Bảng nối tích hợp Arduino (Shield mở rộng): Cung cấp các cổng kết nối dễ dàng cho cảm biến, động cơ, màn hình thông qua các chân JST hoặc header. Ngoài ra, có công tắc và jumpers để chọn nguồn hoặc chuyển đổi chức năng các chân.

Module ổn áp: Là mạch giảm áp (buck converter) có thể điều chỉnh đầu ra. Nhận điện áp đầu vào cao (như 7.4V hoặc 12V) và hạ xuống 5V hoặc 3.3V cho phù hợp với các linh kiện. Đảm bảo cung cấp điện ổn định cho Arduino và các module khác.

Màn hình OLED: Màn hình nhỏ (0.96 inch) độ phân giải 128x64 pixel. Giao tiếp với Arduino qua I2C (chân SDA và SCL) và dùng để hiển thị thông tin như điện áp, dòng điện, trạng thái xe hoặc cảm biến.

Module đo điện: Dùng để đo điện áp và dòng điện tiêu thụ trong mạch và gửi dữ liệu đo về Arduino qua giao tiếp analog hoặc I2C (tùy loại module) giúp giám sát tình trạng năng lượng của hệ thống trong thời gian thực.

3.2.3. Nguyên lý hoạt động

Nguyên lý hoạt động của mô hình xe điện được xây dựng dựa trên sự phối hợp giữa bộ vi điều khiển Arduino Uno R3 và các mô-đun chức năng khác nhằm điều khiển chuyển động và giám sát hệ thống. Nguồn điện từ pin được cấp qua module ổn áp để đảm bảo mức điện áp ổn định cho toàn bộ mạch. Bộ vi điều khiển Arduino nhận lệnh điều khiển từ chương trình và điều phối hoạt động của động cơ điện thông qua mạch driver A4950 để tạo ra lực kéo cho hai bánh sau, đồng thời điều khiển động cơ servo quay bánh trước thông qua hệ thống thước lái giúp xe chuyển hướng. Các module đo điện sẽ giám sát dòng và áp tiêu thụ, truyền dữ liệu về Arduino, và thông tin này được hiển thị trực quan trên màn hình OLED. Toàn bộ các linh kiện được gắn cố định trên khung sườn và tấm kính, tạo nên một hệ thống xe điện hoàn chỉnh và linh hoạt.

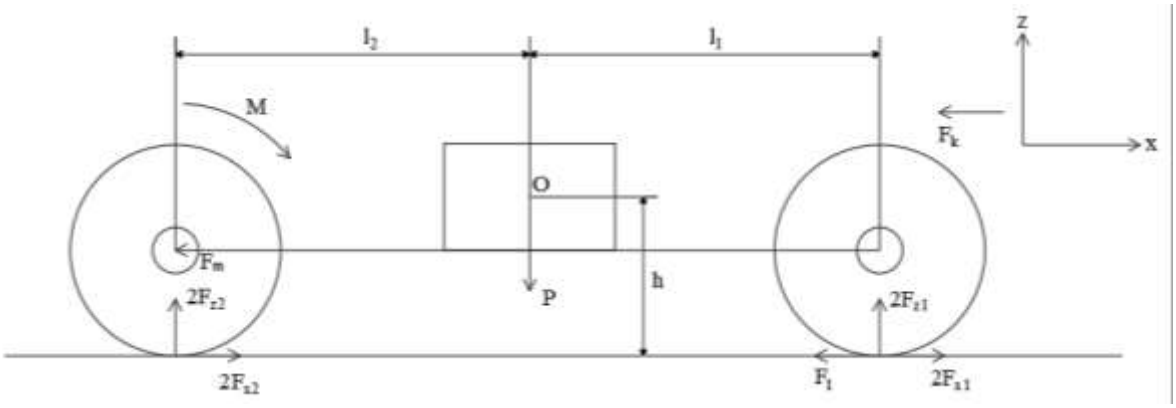
3.3. Tính toán thiết kế mô hình xe điện

Việc xây dựng kết cấu cơ khí đóng vai trò vô cùng quan trọng. Nó ảnh hưởng đến việc phân phối kết cấu, cấu trúc hệ thống điện và cả phần lập trình điều khiển của mô hình. Thiết kế cơ khí xác định điều khiện cần thiết của kết cấu xe để đáp ứng những yêu câu đã đặt ra. Các phép tính toán được dùng để xác định:

- Kích thước xe: chiều dài, chiều rộng, chiều cao xe, vị trí đặt tải trọng…
- Công suất và moment của động cơ
- Dung sai lắp ghép xe
- Mô phỏng động lực học của mô hình xe

3.3.1. Tính toán kích thước xe

Dựa vào phân tích tổng quan, phương án lựa chọn xe 4 bánh. Hai bánh sau dẫn động, hai bánh trước được điều hướng thông qua hệ thống điều chỉnh góc lái.



Hình 3.3 Các lực tác dụng lên xe

Các thông số từ (hình 3.3)

- O là vị trí đặt trọng tâm xe.
- $2F_{x1}$ là lực ma sát ở 2 bánh trước.
- $2F_{x2}$ là lực ma sát ở 2 bánh sau.
- $2F_{z1}$ là phản lực tại 2 bánh trước.
- $2F_{z2}$ là phản lực tại 2 bánh sau.
- $L = l_1 + l_2$ là khoảng cách tâm bánh trước đến bánh sau
- F_c là lực cản tổng hợp. $F_c = F_l + F_k + F_m$

Trong đó:

- Lực cản lăn F_l là ma sát giữa lốp và mặt đường do biến dạng
- Lực cản không khí F_k là do không khí cản trở xe di chuyển
- Lực cản nội tại hệ thống F_m là ma sát trong bộ truyền động, trục, ổ bị,…

Do xe dẫn động bánh sau nên $F_{x1} = 0$

Hệ phương trình động lực học của xe:

$$\begin{cases} \sum F_x = 0 \\ \sum F_z = 0 \\ \sum M_{o/y} = 0 \end{cases} \Leftrightarrow \begin{cases} 2F_{x1} + 2F_{x2} = \sum m.a \\ 2F_{z2} + 2F_{z1} = \sum m.g \\ -2F_{z1}.l_1 + 2F_{z2}.l_2 - 2F_{x2}.h = 0 \end{cases} \quad (3.1)$$

Bỏ qua tổn thất không liên quan đến tiếp xúc mặt đường (ví dụ như lực gió, ma sát ổ trục, v.v). Lúc đó có thể bỏ qua F_c để đơn giản hoá hệ phương trình:

$$\Leftrightarrow \begin{cases} F_{x2} = \frac{m.a}{2} \\ F_{z1} = \frac{m.g}{2} - F_{z2} \\ F_{z2} = \frac{1}{2}.m.\frac{a.h+g.l_1}{l_1+l_2} \end{cases} \quad (3.2)$$

Giả sử gia tốc lớn nhất mà xe đạt được là a_{max}. Mà $\rightarrow \begin{cases} F_{x1} = 0 \\ F_{x2} = \mu . F_{z2} \end{cases}$ (3.3)

Từ (3.1) và (3.2):

$$m.a_{max} = \mu.m.\frac{a.h + g.l_1}{l_1 + l_2} \quad \Leftrightarrow \quad a_{max} = \mu.\frac{g.l_1 + a.h}{l_1 + l_2} \tag{3.4}$$

* Nhận xét: Dựa vào (3.3), gia tốc lớn nhất mà xe đạt được sẽ càng lớn nếu trọng tâm càng dịch về phía sau (l_1 càng lớn) và càng thấp (h càng thấp). Tuy nhiên, dựa vào phương trình thứ hai của hệ phương trình (3.1) nhận thấy, nếu gia tốc a, trong trường hợp này là a_{max}, càng lớn sẽ dẫn đến $F_{z1} \rightarrow 0$. Lúc đó xảy ra hiện tương bánh trước không còn bám được trên mặt đường. Vì vậy:

$$F_{z1} \geq 0 \Leftrightarrow \frac{mg}{2} - F_{z2} = \frac{mg}{2} - \frac{1}{2} m.\frac{ah + gl_1}{l_1 + l_2} = \frac{g.l_2 - a.h}{l_1 + l_2} \geq 0 \Leftrightarrow a \leq \frac{g.l_2}{h} \tag{3.5}$$

3.3.2. Tính toán công suất cho động cơ

Công suất của động cơ cần đủ lớn để giúp xe tăng tốc và tắng được lực cản khi chạy ở vận tốc cài đặt.

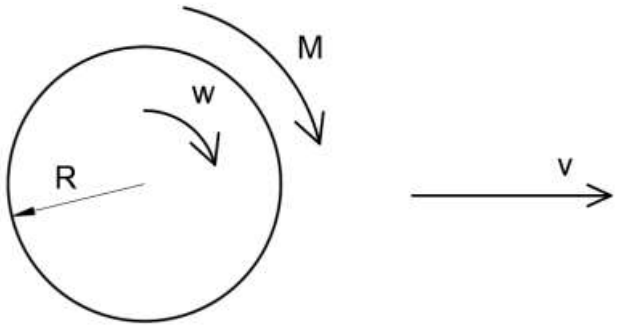
$$P = K.(P_1 + P_2) \tag{3.6}$$

Trong đó:

- P_1 : công suất cần để xe tăng tốc
- P_2 : công suất cần khi xe chạy ở tốc độ v
- K: hệ số tổn thật năng lượng khi truyền động (chọn K=2)

Các thông số ước tính:

- Vận tốc mong muốn: v = 1 (m/s)
- Thời gian tăng tốc mong muốn: t = 1 (s) $\Leftrightarrow$ a = 1 (m/s^2)
- Ước lượng bánh kính xe: R = 32,5 (mm)
- Khối lượng bánh xe: M = 0,025 kg
- Khối lượng xe và tải: m = 3 kg



Hình 3.4 Mô hình phân tích lực bánh chủ động

Tốc độ góc của bánh xe:

$$\omega = \frac{v}{\pi.d}.60 = 293,8(v/p) \quad (3.7)$$

Công suất cần cung cấp cho xe di chuyển ổn định với tốc độ v

$$P_2 = F_{can}.v \Leftrightarrow P_2 = \mu_{can}.mgv \quad (3.8)$$

(xe chạy ở tốc độ thấp trong môi trường không có gió nên bỏ qua lực cản từ không khí)

Công suất cần cung cấp cho xe tăng tốc từ 0 đến v trong thời gian t

Động năng của xe

$$K_d = \frac{1}{2}.m.v^2 \quad (3.9)$$

Công suất cần cung cấp để đạt được gia tốc a = v/t

$$P_1 = \frac{K_d}{t} = \frac{mv^2}{2t} \quad (3.10)$$

Công suất cần thiết của động cơ:

$$P = 2.(P_1 + P_2) \Leftrightarrow P = 2.(\frac{m.v^2}{2t} + \frac{1}{2}.m.v^2) \Leftrightarrow \text{P} = 6 \text{ (W)} \quad (3.11)$$

3.3.3. Tính toán moment xoắn động cơ

Tổng lực kéo cần thiết động cơ tạo ra:

$$\text{F}_\text{k} = \text{F}_\text{cản} + \text{F}_\text{tăng tốc} = \mu_{can}.mgv + ma \quad (3.12)$$

Moment cực đại của bánh xe:

$$\tau_{\max} = F_{x2}.R = \frac{1}{2}.\mu mg.R \quad (3.13)$$

Vậy để đảm bảo điều kiện bánh xe lăn không trượt:

$$\tau \le \tau_{\max} \Leftrightarrow \tau \le 0,35(Nm) \quad (3.14)$$

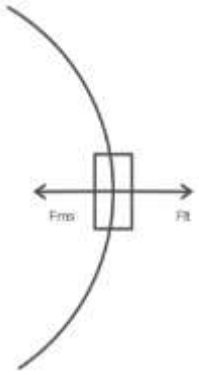
Từ các thông số tính toán trên động cơ được chọn là MG531 gắn ở hai bánh dẫn động với các thông số động cơ:

- Công suất định mức: 7W
- Điện áp làm việc: 7 – 13V
- Loại động cơ: Nam châm vĩnh cửu có chổi than
- Dòng điện khi động cơ bị giữ: 5,4A
- Momen xoắn giữ: 15 kgf.cm
- Momen xoắn định mức: 1,5 kgf.cm

- Dòng không tải: 540mA
- Tốc độ trước khi giảm tốc (không tải): 15000 v/p
- Tốc độ sau khi giảm tốc (có tải): 315 v/p

Tính toán lại vận tốc xe: $v = \frac{\omega.\pi.d}{60} = 1,08(m/s)$ (3.15)

3.3.4. Điều kiện để xe vào cua mà không bị trượt



Hình 3.5 Các lực tác dụng khi xe vào cua

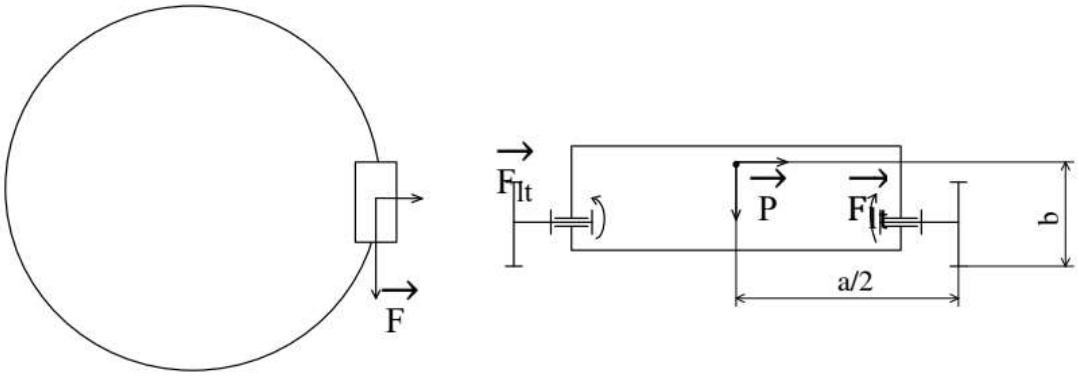
$$F_{ms} \geq \frac{mv^2}{r} \Leftrightarrow \mu mg \geq \frac{mv^2}{r} \Leftrightarrow v \leq \sqrt{\mu rg} \Leftrightarrow v \leq 1,5(\text{m/s}) \quad (3.16)$$

(chọn $\mu = 0,7$ là hệ số ma sát giữa cao su và gỗ sơn bóng)

Từ tính toán v =1,08 (m/s) ban đầu thoả mãn điều kiện để xe không bị trượt khi vào cua.

3.3.5. Điều kiện để xe vào cua không bị lật

Khi vào cua, do ảnh hưởng của lực li tâm nên xe có thể bị lật. Để phòng ngừa trường hợp này, ta phải thiết kế khoảng cách giữa 2 bánh xe và chiều cao xe cho phù hợp để xe không bị lật. ta có mô hình toán như sau:



Hình 3.6 Mô hình tính toán và phân tích lực khi xe ôm cua

Gọi khoảng cách giữa 2 bánh xe là a, chiều cao trọng tâm xe là b. Ta có:

Moment gây ra lật xe: $M_1 = F_n.b = \frac{m.v^2}{R}.b$ (3.17)

Moment chống lật xe: $M_2 = P.\frac{a}{2} = mg.\frac{a}{2}$ (3.18)

Để xe không bị lật thì: $M_2 \geq M_1$ (3.19)

$$\Rightarrow \frac{b}{a} \leq \frac{Rg}{2v^2} \quad (3.20)$$

3.3.6. Tính toán góc lái

a, Xây dựng đường cong lý thuyết

Để đảm bảo động học quay vòng của bánh xe dẫn hướng cần thoả mãn

$$Cotg\alpha - Cotg\beta = \frac{B}{L} \quad (3.21)$$

Trong đó:

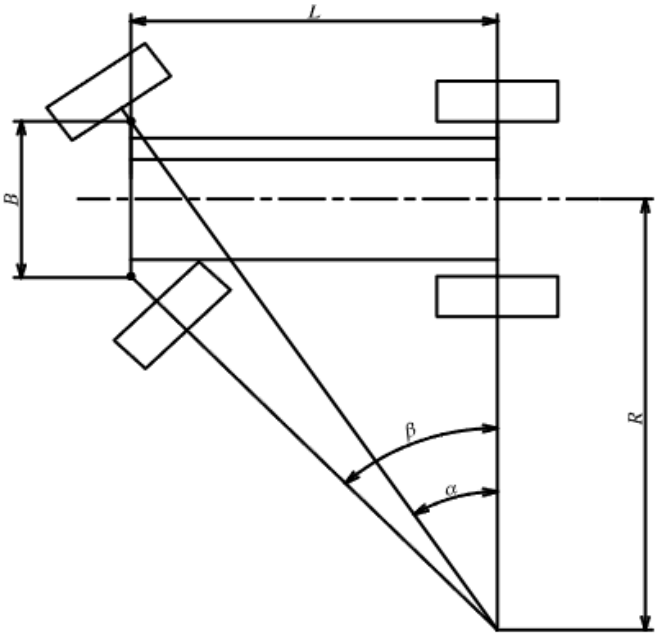
α : Góc quay vòng của bánh xe dẫn hướng bên ngoài.

β : Góc quay vòng của bánh xe dẫn hướng bên trong.

B : Chiều rộng cơ sở.

L: Chiều dài cơ sở.

Để thỏa mãn một cách chính xác biểu thức trên thì dẫn động lái phải có 18 khâu và có cấu tạo phức tạp. Vì vậy, trong thực tế người ta thường sử dụng các cơ cấu dẫn động đơn giản hơn mà vẫn đảm bảo được gần đúng công thức trên, trong đó cơ cấu được sử dụng phổ biến hơn cả là hình thang lái Đan tô. Kinh nghiệm cho thấy, nếu lựa chọn các thông số của hình thang lái một cách hợp lý thì có thể thỏa mãn được công thức 2.1



Hình 3.7 Sơ đồ động học quay vòng

Xác định góc quay lớn nhất của bánh xe dẫn hướng phía trong β_{max} :

$$\beta_{max} = arctg \frac{L}{R_{min} - \frac{B}{2}} \quad (3.22)$$

Với R_{min} = 0,2 m suy ra

$$\beta_{max} = arctg \frac{143}{200 - \frac{122}{2}} = 45° \quad (3.23)$$

Từ 2.1 qua các phép biến đổi ta có:

$$\alpha = arctg \frac{L}{B + L.Cotg\beta} \quad (3.24)$$

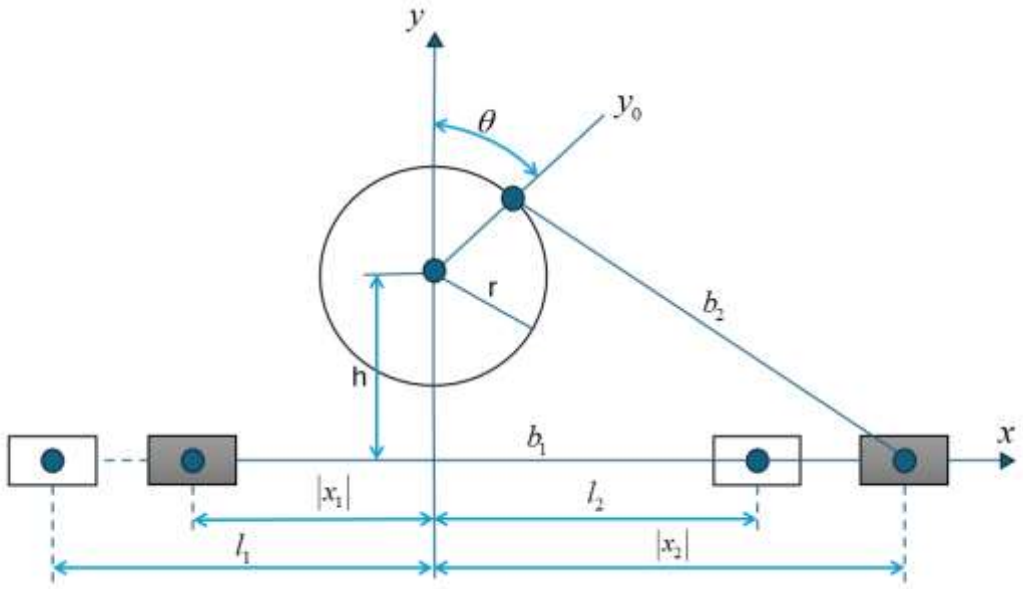
Thay vào phương trình 2.3 ta có phương trình:

$$\alpha = arctg \frac{143}{122 + 143.Cotg45°} = 30° \quad (3.25)$$

*** Nguyên lý hoạt động của hệ thống lái Ackermann**

Khi xe rẽ, bánh xe bên trong cần phải quay với góc lớn hơn bánh xe bên ngoài để đảm bảo chúng luôn đi theo quỹ đạo tròn mà không bị trượt ngang. Nguyên lý Ackermann đảm bảo rằng tất cả các bánh xe luôn có cùng tâm quay khi xe vào cua.

Chúng ta có thể lái xe bằng cách kiểm soát hướng tốc độ của một điểm duy nhất. Một định luật hình học chi phối hướng của bánh xe phía trước, vì hướng lý thuyết của một bánh xe không độc lập với bánh xe kia. Định luật này được gọi là Ackermann, có thể giải thích cơ chế lái. Vì cơ chế của xe chúng ta không tuân theo hình học Ackermann, nên một hoặc nhiều bánh xe phải trượt để tuân theo các định luật động học.



Hình 3.8 Sơ đồ góc lái

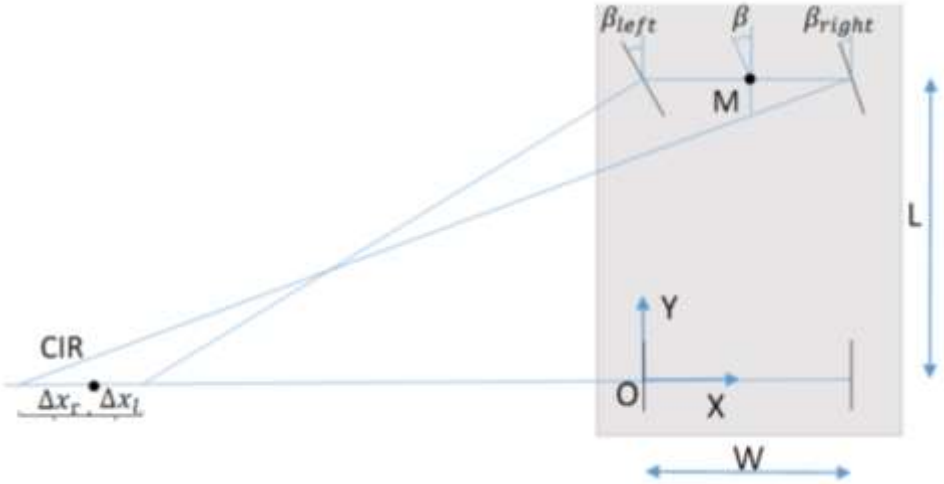
Có thể điều chỉnh được h, r để tìm được cấu hình tốt nhất.

$$|x_2| = \sqrt{b_2^2 - (h + r.\mathrm{Cos}\,\theta)^2}$$
$$|x_1| = b_1 - |x_2| \tag{3.26}$$

Nếu $\theta = 0$ khi đó $|x_1| = l_1, |x_2| = l_2$, l_1, l_2 đã biết và cố định. Ta có thể tìm được góc của mỗi bánh xe trong mặt phẳng x-z

$$\beta_{trái} = \frac{l_2 - |x_2|}{R}, \beta_{phai} = \frac{|x_2| - l_1}{R} \tag{3.27}$$

Hình 3.9 Góc lái thực tế



Hình 3.10 Động học lái

* Xác định phương trình đường đi của hai bánh trước trong hình 3.10

Khi xe quay quanh tâm, hai bánh trước tạo thành hai đường thẳng có góc nghiêng khác nhau do hệ thống Ackerman.

- Bánh trước bên trái có góc quay β_{trai} cộng thêm một hiệu chỉnh $\Delta\beta$.
- Bánh trước biên phải có góc quay β_{phai} trừ đi hiệu chỉnh $\Delta\beta$.

$$y = \tan(\beta_{trai} + \Delta\beta)x + L \tag{3.28}$$

$$y = \tan(\beta_{phai} - \Delta\beta)x + L - \tan(\beta_{trai} - \Delta\beta) \quad (3.29)$$

Hệ số góc của phương trình tuyến tính xuất phát từ định nghĩa góc Ackermann. Khi xe quay quanh CIR, hai bánh trước tạo thành các đường thẳng có góc là $\tan(\beta_{trai} - \Delta\beta)$ và $\tan(\beta_{phai} - \Delta\beta)$

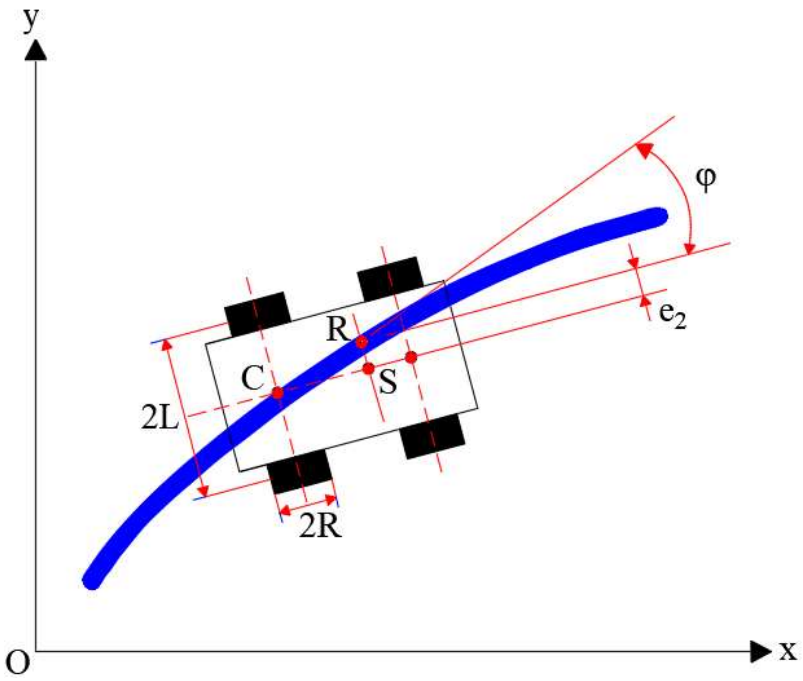
Khoảng cách từ trục trước đến trục sau L, được sử dụng dể dịch chuyển đường thẳng theo phương y.

Hiệu chỉnh $\Delta\beta$: Do bánh trước bên trái và bánh trước bên phải không có cùng góc quay nên ta cần một giá trị hiệu chỉnh để phản ánh sự chênh lệch.

* Cách tiếp cận này đơn giản như không chính xác vì một số lý do:

- Trượt luôn xảy ra khi có lực tác động lên lốp, do đó giả thuyết về điều kiện lái động học chỉ đúng với tốc độ rất thấp.
- Bán kính cong phụ thuộc rất nhiều vào tốc độ, nhưng mô hình này không thể chứng minh được sự phụ thuộc này.
- Chia đều các lỗi giữa hai bánh xe là một mẹo để xác định góc lái, nhưng nó không có giá trị vật lý. Trên thực tế, trượt liên quan đến sự phân bổ khối lượng trên xe cũng như các điều kiện động.

3.3.7. Mô phỏng động học của xe



Hình 3.11 Hệ tọa độ của xe

Hệ tọa độ tuyệt đối (hệ tọa độ gốc) là hệ tọa độ cố định được đặt trong môi trường và được biểu diễn bằng (X,Y). Hệ tọa độ tương đối (hệ tạo độ xe) là hệ tọa độ gắn liền với xe và được biểu thị bằng (X_r, Y_r). Gốc của hệ tọa độ xe là P.

Vị trí xe so với hệ tọa độ xe được xác định bằng ma trận vị trí

$$q = \begin{bmatrix} x & y & \theta \end{bmatrix}^T \tag{3.30}$$

Để chuyển đổi vị trí của xe từ hệ tạo độ tương đối (P X_r Y_r) sang hệ tọa độ tuyệt đối (OXY) ta sử dụng ma trận chuyển đổi R được xác định như sau:

$$\xi = R(\theta)\xi_R \tag{3.31}$$

Trong đó R(θ) là ma trận quay của xe quanh trục thẳng đứng, với:

$$R(\theta) = \begin{bmatrix} \mathrm{Cos}\,\theta & -\mathrm{Sin}\,\theta & 0 \\ \mathrm{Sin}\,\theta & \mathrm{Cos}\,\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.32}$$

Vận tốc tuyến tính của xe trong hệ tọa độ bằng trung bình vận tốc tuyến tính của hai bánh xe theo hệ tọa độ xe:

$$\frac{v_R + v_L}{2} = R\frac{\varphi_R + \varphi_L}{2} \tag{3.33}$$

Vận tốc góc của xe là:

$$\omega = \frac{\dot{v}_R + \dot{v}_L}{2} = R\frac{\dot{\varphi}_R + \dot{\varphi}_L}{2} \tag{3.34}$$

Các vận tốc của xe trong hệ toạ độ có thể biểu diễn dưới dạng các vận tốc của điểm trung tâm P trong khung xe như sau:

$$\begin{cases} \dot{x}_p^r = R\frac{\dot{\varphi}_R + \dot{\varphi}_L}{2} \\ \dot{y}_p^r = 0 \\ \dot{\theta} = \omega = R.\frac{\dot{\omega}_R + \dot{\omega}_L}{2} \end{cases} \tag{3.35}$$

Suy ra:

$$\dot{q}^r = \begin{bmatrix} \frac{R}{2} & \frac{R}{2} \\ 0 & 0 \\ \frac{R}{2L} & -\frac{R}{2L} \end{bmatrix} \times \begin{bmatrix} \dot{\varphi}_R \\ \dot{\varphi}_L \end{bmatrix} \tag{3.36}$$

Với:

- R = 0,0325 (m) : là bán kính bánh xe;
- L = 0,096 (m) : là khoảng cách giữa 2 bánh xe;
- $\dot{\varphi}_R, \dot{\varphi}_L = 0,4$: là vận tốc góc của bánh phải, trái của xe

$\dot{x}_p^r = 1,08 m/s$: xe đi thẳng với tốc độ 1,08 m/s

$\dot{y}_p^r = 0$: không có chuyển động ngang

$\dot{\theta} = 0$: không có quay khi xe chạy thẳng

Ma trận vận tốc theo hệ tuyệt đối được thể hiện như sau:

$$\dot{q}^l = \begin{bmatrix} \dot{x}_P^l \\ \dot{y}_P^l \\ 0 \end{bmatrix} = R(\theta) \begin{bmatrix} \dot{x}_P^r \\ \dot{y}_P^r \\ \dot{\theta}^r \end{bmatrix} = \begin{bmatrix} \frac{R}{2}\text{Cos}\,\theta & \frac{R}{2}\text{Cos}\,\theta \\ \frac{R}{2}\text{Sin}\,\theta & \frac{R}{2}\text{Sin}\,\theta \\ \frac{R}{2L} & -\frac{R}{2L} \end{bmatrix} \begin{bmatrix} \dot{\varphi}_R \\ \dot{\varphi}_L \end{bmatrix} \tag{3.38}$$

Với R = 0,0325 (m) là bán kính bánh xe

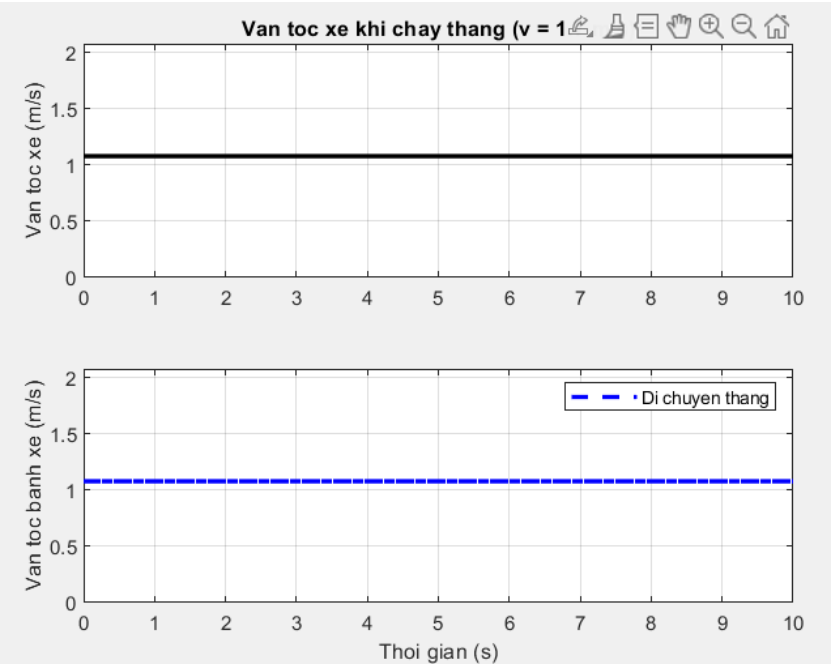
L = 0,096 (m) là khoảng cách giữa 2 bánh xe

$\dot{\varphi}_R, \dot{\varphi}_L = 0,5$: là vận tốc góc của bánh phải, trái của xe

$\theta = 0$(độ): là góc quay của bánh xe

Ma trận $\dot{q}^l$ còn được thể hiện theo vận tốc dài v và vận tốc ω theo công thức sau:

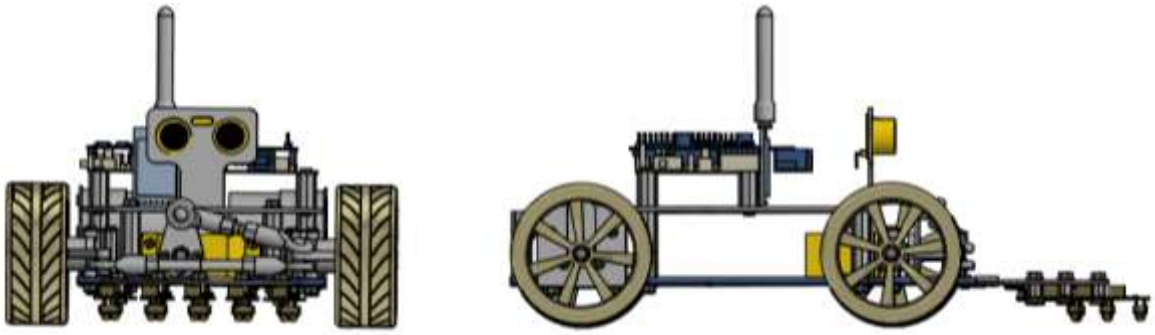
$$\dot{q}^l = \begin{bmatrix} \text{Cos}\,\theta & 0 \\ \text{Sin}\,\theta & 0 \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{3.39}$$



Hình 3.12 Vận tốc xe và bánh trái, phải theo thời gian khi xe chạy thẳng

3.3.8. Các thông số cơ bản của xe

Từ việc tính toán các thông số kích thước cơ bản của xe, cùng với bản thiết kế xe trên phần mềm Inventor như (hình ..), bảng thông số của xe được trình bày trong (bảng ..)



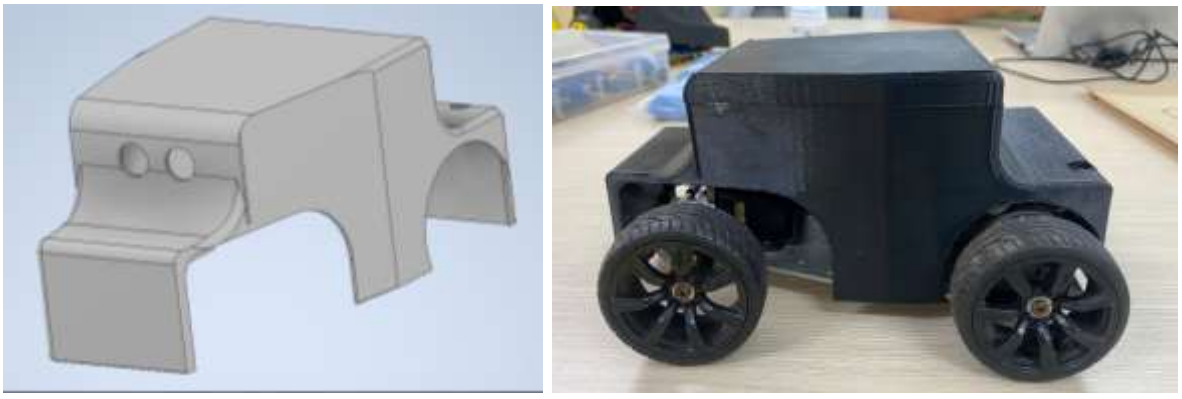
Hình 3.13 Mô hình thiết kế

Bảng 3.2 Thông số thiết kế mô hình xe

Thông số	Giá trị
Khoảng cách từ tâm bánh điều hướng đến tâm bánh dẫn động	a = 140 mm
Chiều rộng xe	b = 176 mm
Chiều dài xe	c = 211 mm
Chiều cao tính từ sàn đến trọng tâm	h = 35 mm
Bán kính bánh xe	r = 32,5 mm

3.3.9. Thiết kế in 3D vỏ xe

Công nghệ in 3D, đặc biệt là phương pháp FDM (Fused Deposition Modeling), được ứng dụng rộng rãi trong chế tạo mẫu mô hình nhờ khả năng tạo hình nhanh, chính xác và tiết kiệm chi phí. Với nguyên lý hoạt động dựa trên việc đùn sợi nhựa nhiệt dẻo (như PLA, ABS) qua đầu phun nóng và đắp từng lớp vật liệu theo mô hình 3D định sẵn, FDM cho phép tạo ra những chi tiết có hình dạng phức tạp mà các phương pháp gia công truyền thống khó thực hiện. Đây là lựa chọn lý tưởng cho việc thiết kế và sản xuất vỏ mô hình xe, đặc biệt trong môi trường nghiên cứu và học tập.



Hình 3.14 Thiết kế vỏ xe và cản bọc cụm cảm biến

Quá trình thực hiện bao gồm các bước chính: đầu tiên là thiết kế mô hình vỏ xe bằng phần mềm CAD (như Autodesk Inventor), sau đó xuất file sang định dạng .STL để đưa vào phần mềm slicing (ví dụ: Ultimaker Cura) nhằm thiết lập thông số in như chiều cao lớp, tốc độ in, nhiệt độ... Tiếp theo, file G-code được nạp vào máy in 3D để tiến hành in thực tế. Sau khi in xong, sản phẩm sẽ được xử lý hoàn thiện bề mặt (nếu cần) và lắp ráp lên khung xe mô hình để kiểm tra độ phù hợp và đánh giá hiệu quả thiết kế.

3.4. Thiết kế bố trí các bộ phận, hệ thống trang bị trên mô hình xe điện

3.4.1. Phương án chọn và bố trí động cơ

Phương án	**Hình ảnh**	**Ưu điểm**	**Nhược điểm**
Động cơ bước		- Có thể điều khiển chính xác vị trí theo bước.	- Hiện tượng trượt bước, muốn hồi tiếp cần gắn them encoder, lúc dừng tại 1 vị trí động cơ ngậm điện để giữ rotor.
Động cơ servo MG995		- Tránh được hiện tượng trượt bước như Step DC - Mô-men xoắn mạnh, chịu tải tốt. - Không yêu cầu độ chính xác cao tuyệt đối trong góc giới hạn, khiển soát vị trí.	- Nhiệt độ cao khi hoạt động lâu, ảnh hưởng đến hiệu suất. - Có thể tiêu thụ dòng điện lớn khi chịu tải (lên đến ~1.5A), yêu cầu nguồn cấp đủ mạnh. - Thiết lập bộ điều khiển phức tạp.
Động cơ DC gắn encoder		- Tốc độ tỉ lệ với điện áp hiệu dụng, đặc tuyến torque/dòng tuyến tính, khả năng bị quá tải thấp. Giá thành thấp	- Tản nhiệt kém

Động cơ giảm tốc vàng 3-9V		- Với giá thành rẻ, dễ sử dụng nên chiếc động cơ này được sử dụng rất nhiều với mô hình nghiên cứu các nghành kĩ thuật.	- Chỉ dùng với các mô hình nhỏ, tai trọng và momen thấp.

* Dựa vào tính toán công suất ta chọn loại động cơ điện phù hợp:

Động cơ servo sử dụng điều khiển thước lái dẫn hướng:

Hình 3.15 Động cơ servo MG995

Thông số kỹ thuật

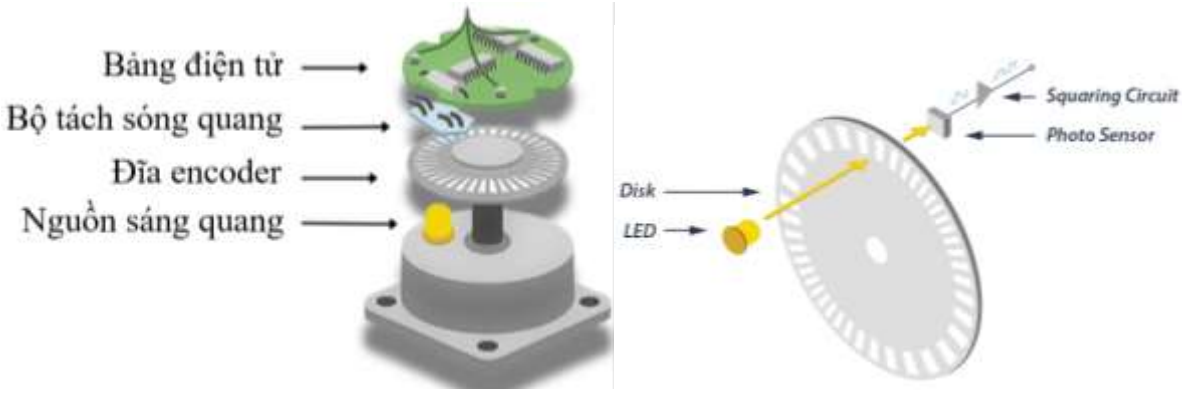
- Điện áp hoạt động 3.5 ~ 8.4V
- Dòng điện cung cấp <1000mA
- Xung yêu cầu: xung vuông điện áp đỉnh từ 3~5
- Nhiệt độ hoạt động: từ 0 đến 60°C
- Kiểu bánh răng: bánh đồng
- Lực kéo: + 13 kg/cm: tại 4.8V
+ 15 kg/cm: tại 6V
- Tốc độ quay: + 0.17 sec/600: tại 4.8V chế độ không tải
+ 0.13 sec/600: tại 6V chế độ không tải
- Kích thước: 40 x 20 x 47mm
- Khối lượng: 48g

Động cơ DC gắn encoder sử dụng để di chuyển xe:

Hình 3.16 Động cơ DC gắn encoder

Thông số kỹ thuật

- Mã động cơ: MG531
- Công suất định mức: 7W
- Điện áp làm việc: 7 – 13V
- Loại động cơ: Nam châm vĩnh cửu có chổi than
- Dòng điện khi động cơ bị giữ: 5,4A
- Momen xoắn giữ: 15 kgf.cm
- Momen xoắn định mức: 1,5 kgf.cm
- Dòng không tải: 540mA
- Tốc độ trước khi giảm tốc (không tải): 15000 v/p
- Tốc độ sau khi giảm tốc (có tải): 500 v/p



Hình 3.17 Nguyên lý bộ encoder

Bộ mã hóa (encoder) sử dụng nhiều công nghệ khác nhau để tạo ra tín hiệu, bao gồm cơ học, từ tính, điện trở và quang học – trong đó công nghệ quang học được sử dụng phổ biến nhất. Trong cảm biến quang học, bộ mã hóa cung cấp tín hiệu phản hồi dựa trên sự gián đoạn của chùm tia sáng, như minh họa trong Hình 3.18.

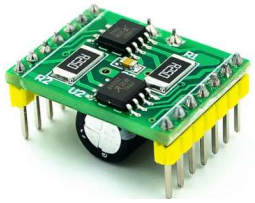
Một chùm ánh sáng phát ra từ đèn LED sẽ đi qua một đĩa mã (Code Disk) trong suốt được tạo hoa văn bằng các vạch mờ, tương tự như nan hoa của bánh xe đạp. Khi trục bộ mã hóa quay, các vạch mờ này sẽ ngắt quãng chùm sáng. Tín hiệu ánh sáng đã được điều biến này sau đó được thu nhận bởi Bộ thu quang điện (Photodetector

Assembly), thường là một mảng điốt quang hay cảm biến quang. Cảm biến quang phản ứng bằng cách tạo ra dạng sóng hình sin, sau đó được chuyển đổi thành sóng vuông hoặc chuỗi xung. Các tín hiệu xung này sẽ được truyền tới bộ đếm hoặc bộ điều khiển để xử lý và thực hiện các chức năng điều khiển mong muốn.

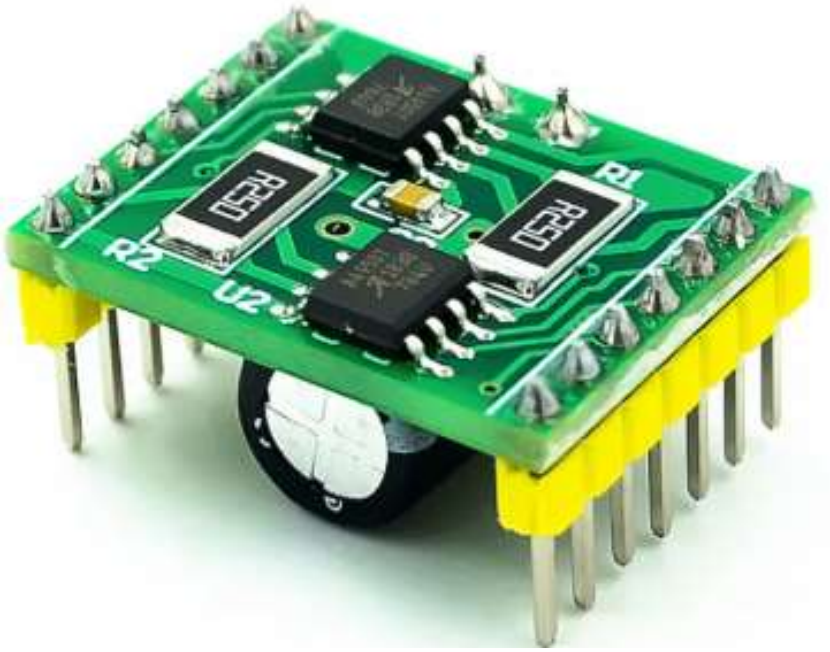
Bên cạnh tín hiệu xung thông thường, bộ mã hóa gia tăng còn tạo ra một xung chỉ số (index pulse) mỗi vòng quay, còn được gọi là xung đánh dấu hoặc xung tham chiếu. Xung này xuất hiện tại một điểm cơ học xác định trong mỗi vòng quay của trục bộ mã hóa và được đưa ra trên một kênh riêng biệt (kênh Z) so với các kênh tín hiệu thông thường. Xung chỉ số đóng vai trò quan trọng trong các ứng dụng điều khiển chuyển động, giúp xác định chính xác vị trí tham chiếu cơ học của hệ thống.

3.4.2. Phương án chọn và bố trí mạch điều khiển

Phương án	Hình ảnh	Ưu điểm	Nhược điểm
Bộ điều khiển L298		- Có thể điều khiển hai động cơ DC độc lập hoặc một động cơ bước. - Mạch sử dụng cầu H (H-Bridge) cho phép thay đổi chiều quay của động cơ dễ dàng. - Cung cấp dòng tối đa khoảng **2A mỗi kênh**, phù hợp với nhiều động cơ có công suất vừa và nhỏ. - Hỗ trợ dải điện áp đầu vào từ 5V đến 35V nên thích hợp với nhiều loại động cơ khác nhau. - Điều khiển tốc độ động cơ bằng cách điều chỉnh độ rộng xung PWM. - Bảo vệ mạch khỏi xung ngược điện áp từ động cơ, giảm nguy cơ hỏng hóc.	- Do sử dụng MOSFET cũ hơn, hiệu suất chỉ khoảng **60-70%**, gây hao phí năng lượng và sinh nhiệt lớn. - Dòng điện tối đa **2A/kênh** không đủ để điều khiển các động cơ mạnh hơn. - Cần ít nhất **4 chân GPIO** để điều khiển hai động cơ, có thể chiếm nhiều tài nguyên trên vi điều khiển. - Điện áp sụt do sử dụng transistor Darlington, có thể làm giảm khoảng **1.5V - 2V** điện áp đầu ra

Bộ điều khiển A4950		- Sử dụng MOSFET với RDS(on) thấp nên giảm điện áp sụt và tăng hiệu suất. - Ít toả nhiệt hơn L298. - Cung cấp dòng liên tục 3.5A và xung tối đa 5A, hợp với động cơ công suất vừa. - Điều khiển tốc độ bằng PWM với tần số cao cho động cơ chạy êm hơn và không bị ồn. - Tích hợp MOSFET cầu H bên trong, không cần diode ngoài như L298.	- Nếu cần điều khiển nhiều động cơ, cần nhiều IC A4950 hoặc chọn driver khác. - Dòng điện tối đa 5A xung / 3.5A liên tục, thấp hơn so với các driver như BTS7960 (~40A). - Điều khiển động cơ DC, không thể điều khiển động cơ bước như L298. - Khi tải nặng, cần tản nhiệt hoặc PCB thiết kế tốt để tránh quá nhiệt.

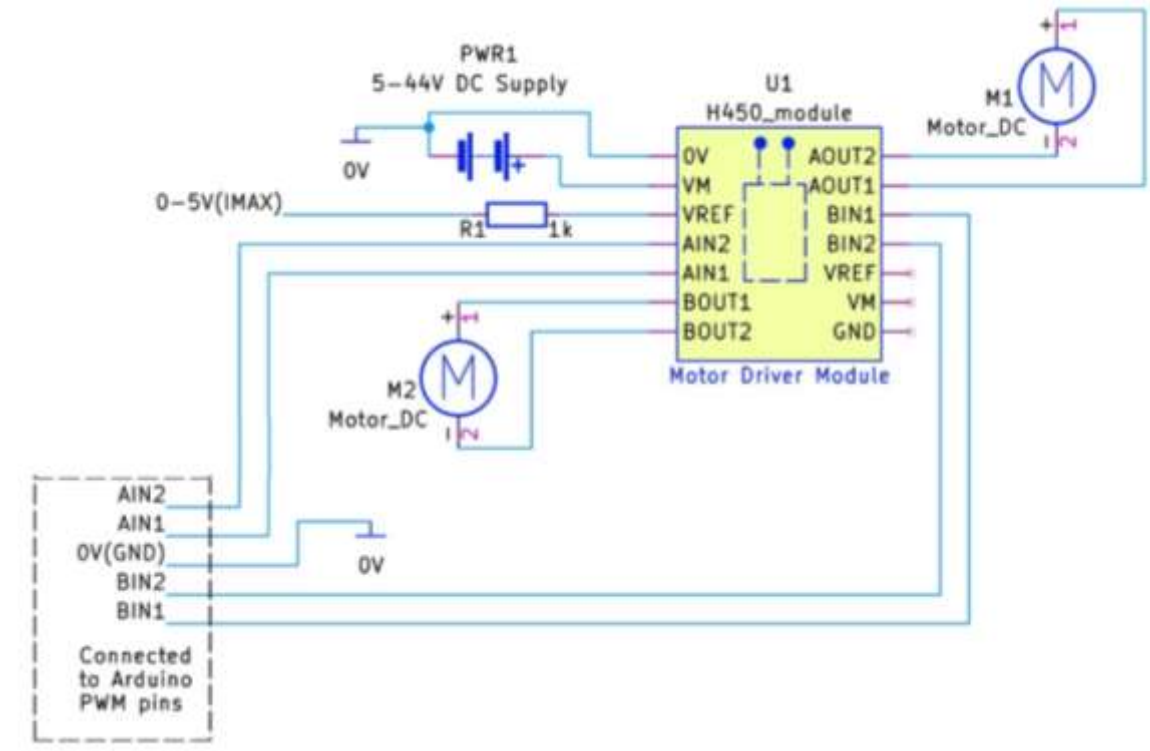
* Dựa vào tính năng nổi trội ta chọn loại mạch điều khiển phù hợp phù hợp:



Hình 3.18 Mạch điều khiển A4950

Thông số kỹ thuật :

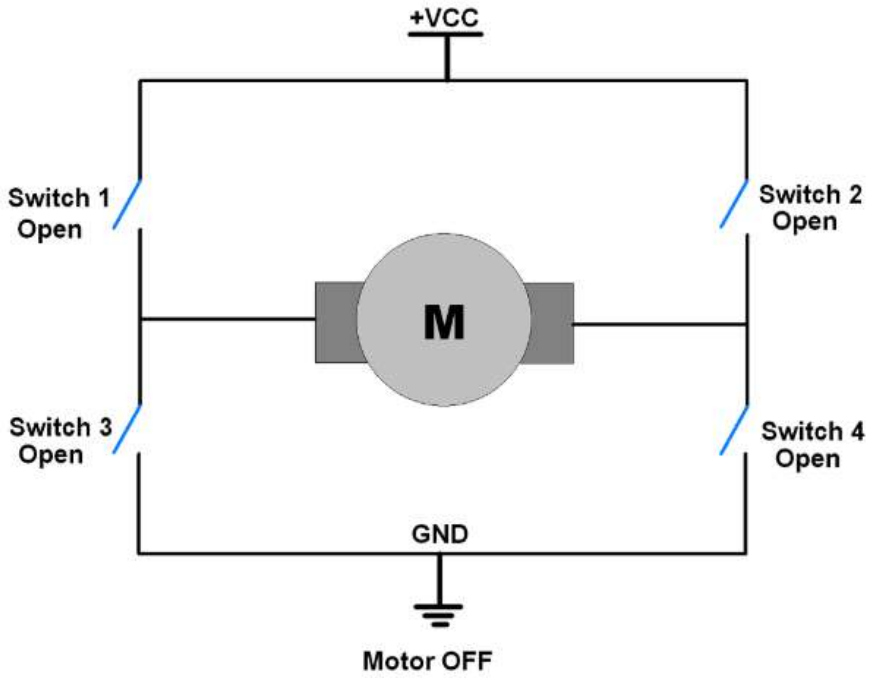
- Điện áp điều khiển : 7.6~40VDC
- Có bảo vệ giới hạn dòng điện, 2A (quá dòng sẽ không cắt nguồn mà chỉ giới hạn dòng điện đầu ra)



Hình 3.19 Sơ đồ mạch điều khiển A4950

*** Tìm hiểu về mạch cầu H**

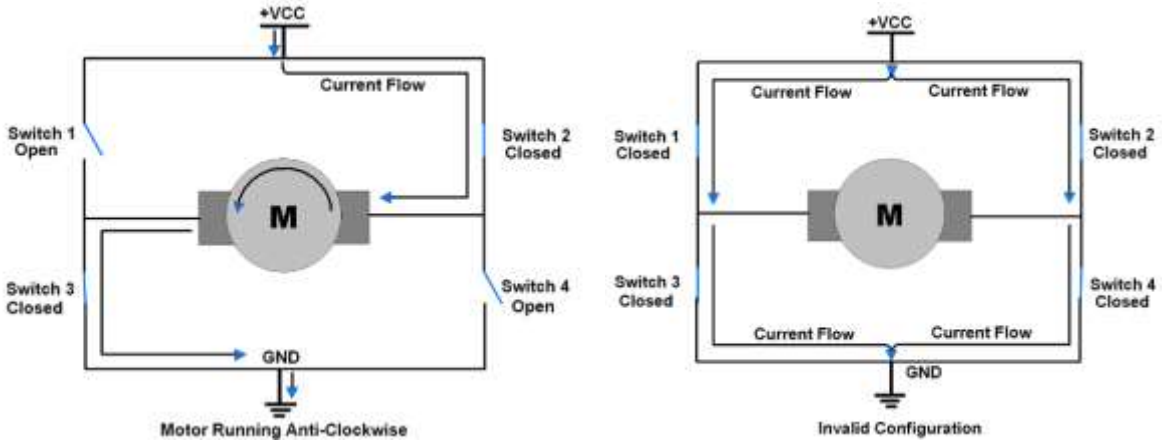
Mạch cầu H là một mạch đơn giản dùng để điều khiển động cơ DC quay thuận hoặc quay nghịch. Trong thực tế, có nhiều kiểu mạch cầu H khác nhau tùy vào cách chúng ta lựa chọn linh kiện có dòng điện, áp điều khiển lớn hay nhỏ, tần số xung PWM… Và chúng sẽ quyết định đến khả năng điều khiển của cầu H.



Hình 3.20 Mô hình mạch cầu H

Một động cơ DC có thể quay thuận hoặc quay nghịch tùy thuộc vào cách bạn mắc cực âm và dương cho motor đó. Ví dụ, động cơ DC có hai đầu A và B. Nếu bạn nối A vào cực dương (+) và B vào cực âm (-) của nguồn thì động cơ quay theo chiều thuận (giả sử cùng chiều kim đồng hồ). Bây giờ bạn nối ngược lại, A vào (-) và B vào (+), động cơ sẽ quay nghịch (giả sử ngược chiều kim đồng hồ).

Tương tự, khi ta đóng S1 và S4, ta đã cho A nối với cực dương (+) và B nối với cực âm (-) của nguồn, một dòng điện chạy từ nguồn qua S1 qua động cơ qua S4 về mass làm động cơ quay theo chiều thuận (hình 2.8a).



Hình 3.21 a, Động cơ quay chiều thuận ; b, Động cơ quay chiều nghịch

Ngược lại (hình 3.22b), khi ta đóng S2 và S3, động cơ quay nghịch

3.4.3. Khối nguồn

Pin 18650 được ứng dụng trong rất nhiều các lĩnh vực thiết bị số như pin laptop, pin sạc dự phòng, pin máy khoan pin và trong các đồ án xe tự hành cơ vừa và nhỏ. Pin 18650 là cell pin Lithium-ion có thể sạc lại trong ngành công nghiệp pin, thường được sử dụng cho pin laptop hoặc sạc dự phòng. Là pin có thể sạc lại được và sử dụng công nghệ Lion (Lithium Ion). Pin 18650 có điện áp dao động quanh khoảng 3.7V – 4.2V.

Pin 18650 có đường kính 18mm và chiều dài khoảng 650mm, cái tên 18650 cũng bắt nguồn từ đây để phân biệt với các dòng pin khác. số 0 chính là để phân biệt đây là pin dạng hình trụ. Ngoài ra nó cũng có người anh em là pin 26650 với kích thước bé hơn và dung lượng lớn hơn nhưng ít phổ biến hơn 18650. Trong đồ án nhóm quyết định sử dụng pin cell thày vì các loại pin thông thường bởi một số lí do sau đây:

- Khối lượng, kích thước nhỏ gọn.

- Có thể sạc đầy và sử dụng tiếp.

- Đáp ứng đủ loại công suất, chỉ phụ thuộc vào công suất sử dụng và thời gian sử dụng (tức là công suất sử dụng càng lớn thì thời gian sử dụng càng ngắn và ngược lại).

Em lựa chọn loại pin cell sau đây:

Thông số kỹ thuật của Pin cell 18650 Samsung 2600mAh

- Chất liệu pin: pin lithium ion;

- Thương hiệu pin: Samsung;

- Mẫu pin: 18650;

- Thông số kỹ thuật của pin: đường kính 18mm * dài 65mm;

- Dung lượng pin: 2600 mAh;

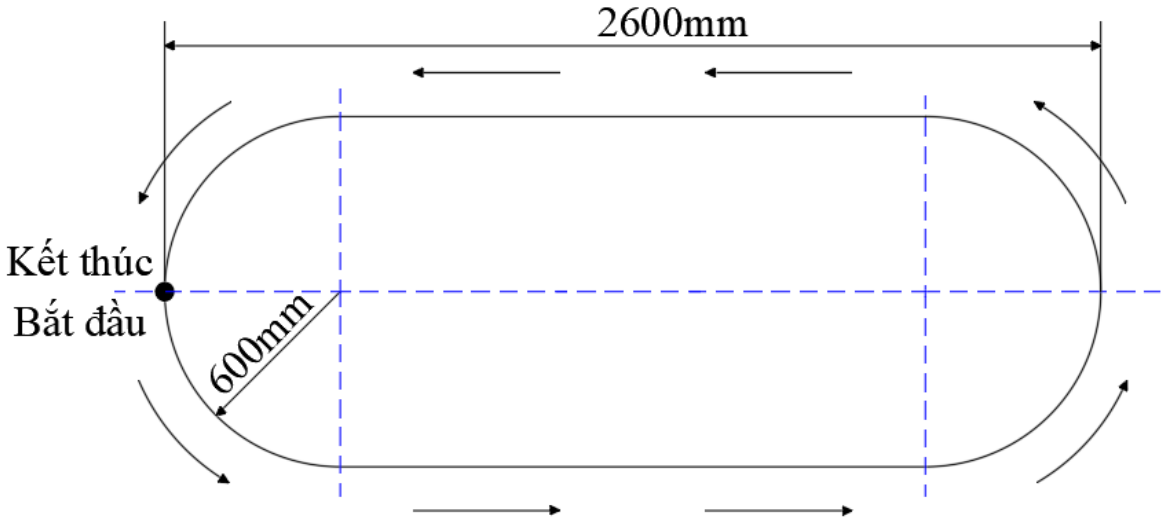
- Điện trở trong của pin: 45,6 mΩ (milliohms);

- Số lần xả: 1000 chu kỳ trở lên;

- Tiêu chuẩn điện áp: điện áp tiêu chuẩn 3,6 ~ 3,7V, điện áp 4.2V sau khi đầy điện, 2,75V sau khi xả.

Hình 3.22 Pin cell 18650 Samsung 2600mAh

3.5. Phương án chọn và thiết kế mô hình đường sá



Hình 3.23 Biên dạng đường có vòng cua

Ưu điểm của biên dạng đường này:

1. Dễ điều hướng: Các đoạn cong ở hai đầu giúp xe tự hành quay đầu một cách mượt mà, giảm thiểu nguy cơ mất cân bằng hoặc trượt bánh.

2. Giảm mài mòn: Do chuyển hướng trơn tru, bánh xe ít bị mài mòn hơn so với khi quay gấp.

3. Tăng tốc độ di chuyển: Xe có thể duy trì tốc độ cao hơn khi vào cua vì không phải giảm tốc độ đột ngột.

4. Thích hợp cho cảm biến dò line: Đường cong liên tục giúp cảm biến dễ theo dõi và ít bị nhiễu hơn so với góc vuông.

Nhược điểm:

1. Chiếm diện tích lớn hơn: Các góc bo tròn chiếm nhiều không gian hơn so với góc vuông.

2. Phức tạp khi lập trình: Cần thuật toán phức tạp hơn để điều khiển tốc độ và hướng khi vào đoạn cong.

3. Khó tinh chỉnh PID: Việc tinh chỉnh PID khó hơn ở đoạn cong do thay đổi góc liên tục.

4. Yêu cầu cảm biến chính xác: Cảm biến phải có độ nhạy và độ chính xác cao để theo dõi biên dạng cong.

Biên dạng đường được chọn để thiết kế trong quá trình thực nghiệm mô hình di chuyển:

Hình 3.24 Thiết kế và chế tạo đường sa bàn

Chương 4: THIẾT KẾ PHẦN ĐIỆN ĐIỀU KHIỂN

4.1. Giới thiệu về phần mềm Arduino IDE và các bo mạch Arduino UNO

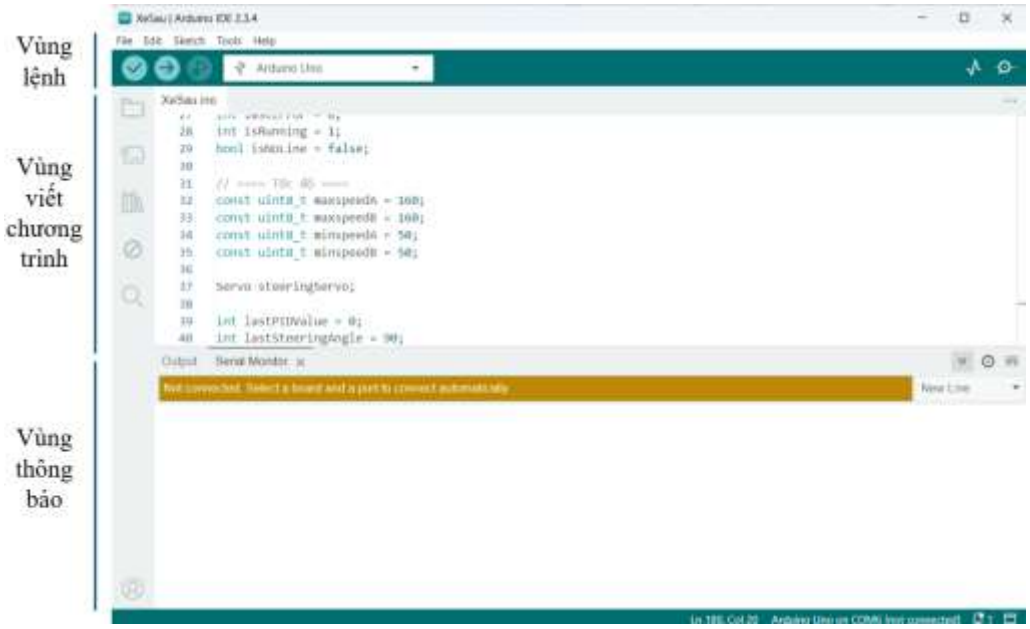
4.1.1. Giới thiệu phần mềm Arduino IDE

a, Giao diện

Arduino IDE là một phần mềm với một mã nguồn mở, được sử dụng chủ yếu để viết và biên dịch mã vào module Arduino. Nó bao gồm phần cứng và phần mềm. Phần cứng chứa đến 300,000 board mạch được thiết kế sẵn với các cảm biến, linh kiện. Phần mềm giúp bạn có thể sử dụng các cảm biến, linh kiện ấy của Arduino một cách linh hoạt phù hợp với mục đích sử dụng.

Arduino IDE có một giao diện đơn giản, dễ sử dụng giúp người dùng thuận tiện hơn trong thao tác. Dưới đây là một số tính năng chúng ta thường sử dụng:

- Nút kiểm tra chương trình (Verify): giúp dò lỗi phần code định truyền xuống bo mạch Arduino.
- Nút tải đoạn code vào bo mạch Arduino (Upload): giúp nhập đoạn code vào bo mạch Arduino.
- Vùng lập trình: người dùng sẽ viết chương trình tại khu vực này.
- Thanh Menu: gồm những thẻ chức năng nằm trên cùng như File, Edit, Sketch, Tools, Help.



Hình 4.1 Giao diện của phần mềm Arduino

b, Vùng lệnh

Bao gồm các nút lệnh menu (File, Edit, Sketch, Tools, Help). Phía dưới là các icon cho phép sử dụng nhanh các chức năng thường dùng của IDE được miêu tả như sau:

Icon	Chức năng

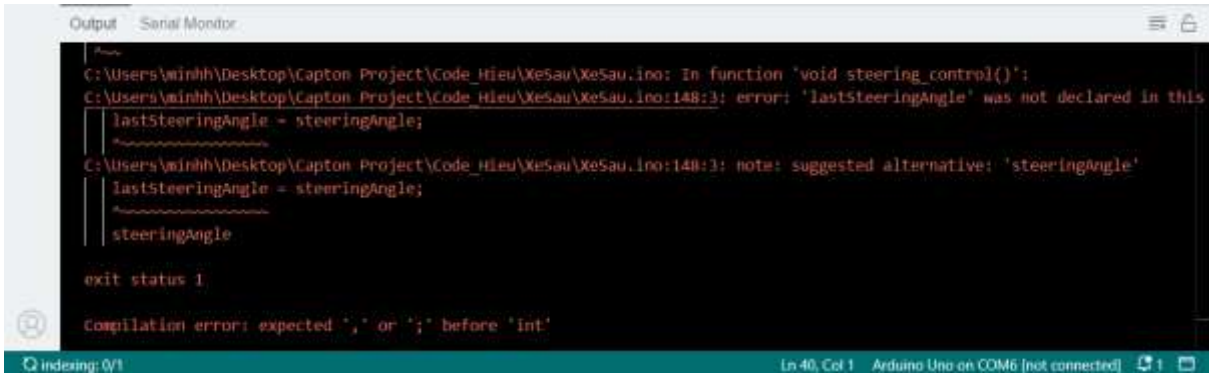
	Biên dịch chương trình đang soạn thảo để kiểm tra các lỗi lập trình.
	Biên dịch và upload chương trình đang soạn thảo.
	Mở một trang soạn thảo mới.
	Mở các chương trình đã lưu.
	Lưu chương trình đã soạn.
	Mở cửa Serial Monitor để gửi và nhận dữ liệu giữa máy tính và board Arduino.

Hình 4.2 Mô tả vùng lệnh

c, Vùng viết chương trình

Viết các đoạn mã của mình tại đây. Tên chương trình được hiển thị ngay dưới dãy các Icon, ở đây nó tên là "Blink". Để ý rằng phía sau tên chương trình có một dấu "§". Điều đó có nghĩa là đoạn chương trình chưa được lưu lại.

d, Vùng thông báo (debug)

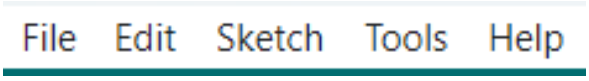


Hình 4.3 Vùng thông báo

Những thông báo từ IDE sẽ được hiển thị tại đây. Để ý rằng góc dưới cùng bên phải hiển thị loại board Arduino và cổng COM được sử dụng. Luôn chú ý tới mục này bởi nếu chọn sai loại board hoặc cổng COM sẽ không thể upload được code của mình.

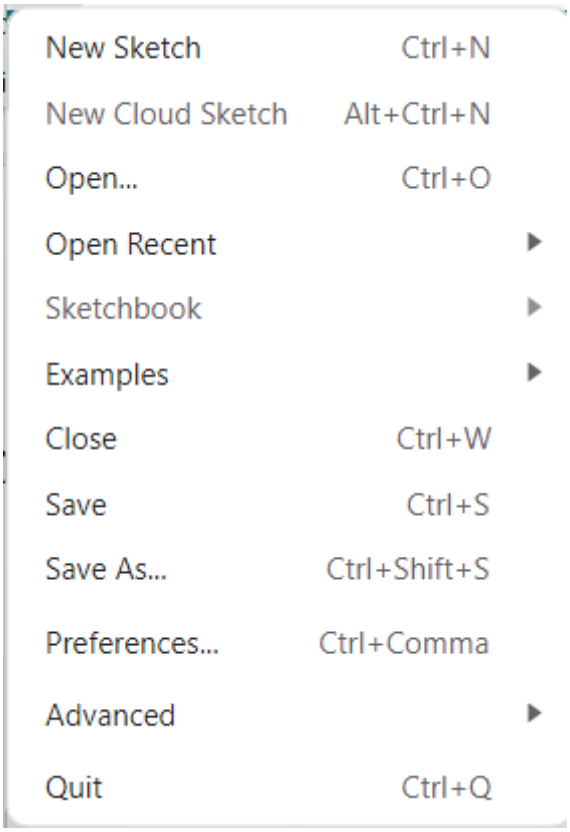
Bạn có thể tìm thấy một vài hướng dẫn khắc phục các lỗi thường gặp khi lập trình Arduino tại Lỗi của Arduino? Và các lỗi thường gặp khi lập trình Arduino

e) Arduino IDE Menu:



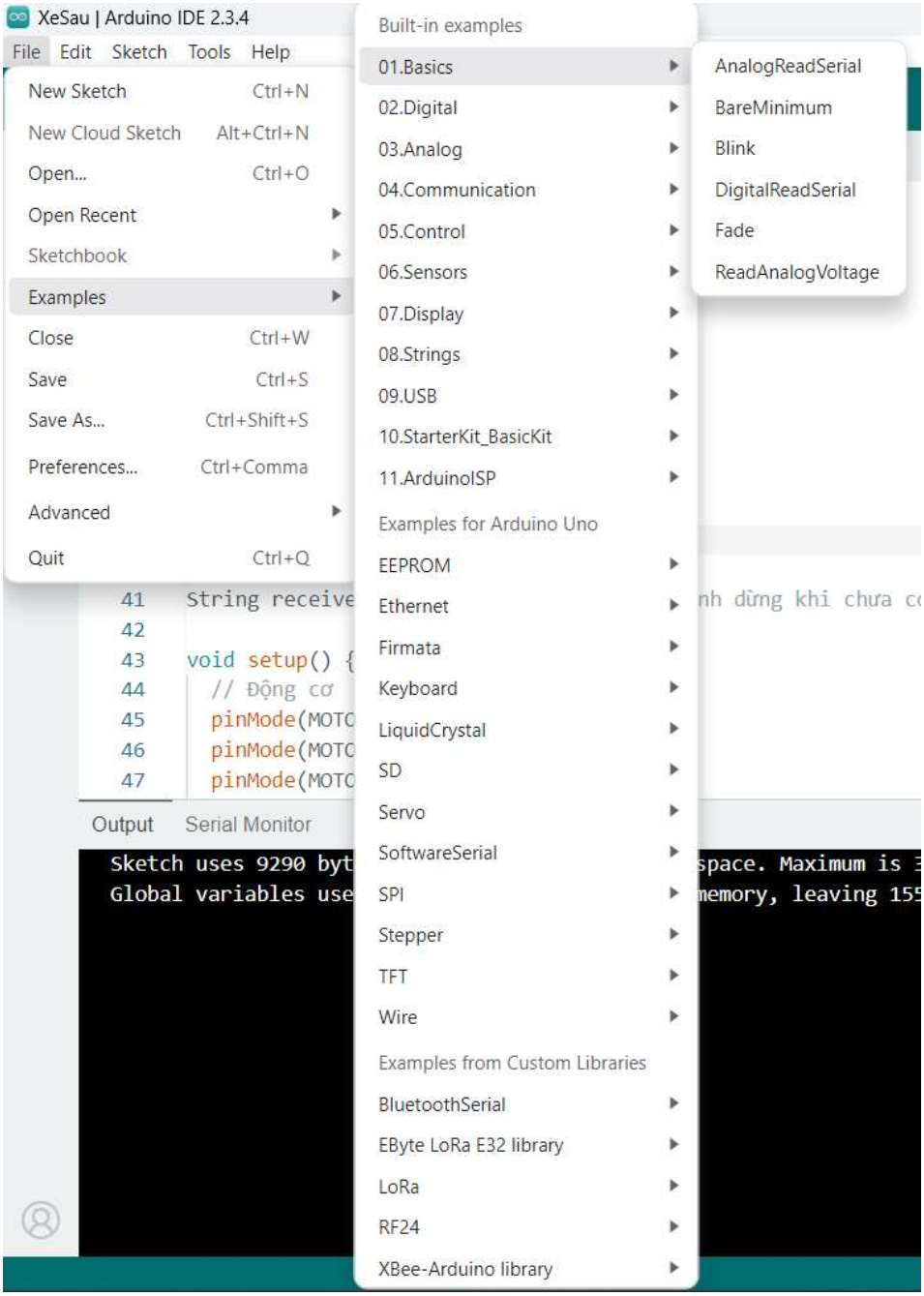
Hình 4.4 IDE Menu

File menu:



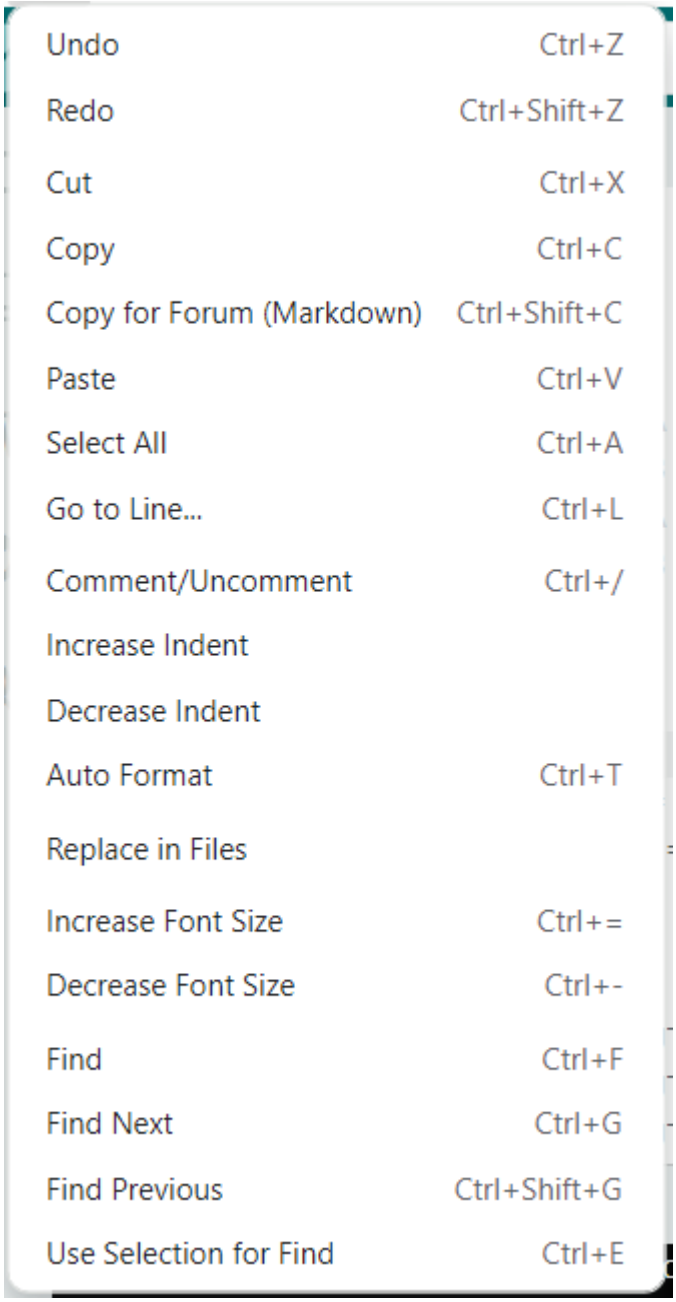
Hình 4.5 File menu

Trong file menu chúng ta quan tâm tới mục Examples đây là nơi chứa code mẫu ví dụ như: cách sử dụng các chân digital, analog, sensor …



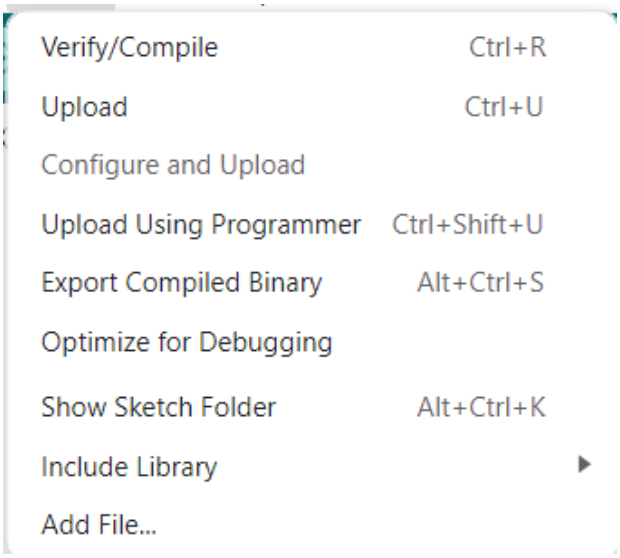
Hình 4.6 Click Examples

Edit menu:



Hình 4.7 Edit menu

Sketch menu

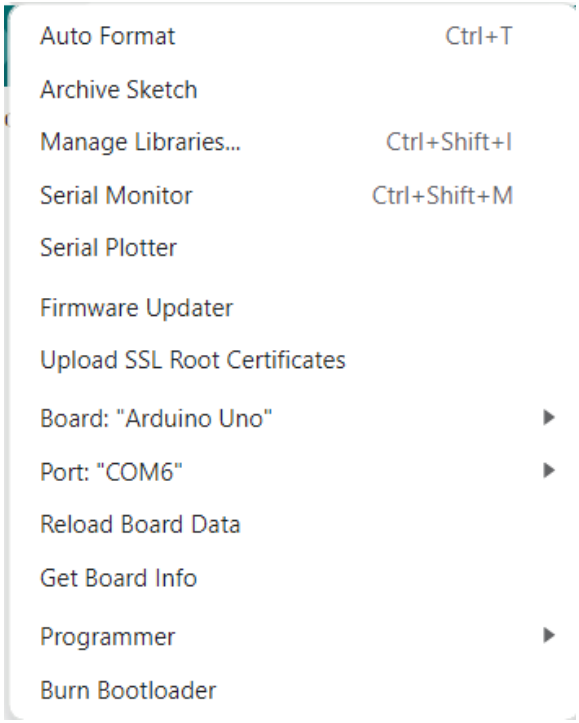


Hình 4.8 Sketch menu

Trong Sketch menu :

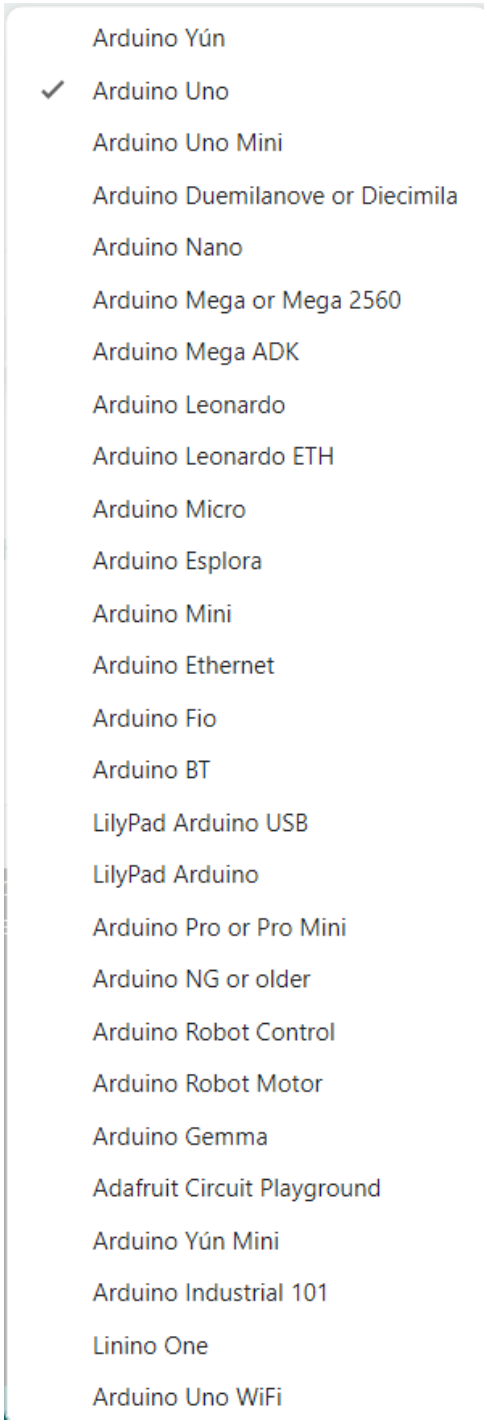
- Verify/ Compile : chức năng kiểm tra lỗi code.
- Show Sketch Folder : hiển thị nơi code được lưu.
- Add File : thêm vào một Tap code mới.
- Import Library : thêm thư viện cho IDE

Tool memu:



Hình 4.9 Tool menu

Trong Tool menu ta quan tâm các mục Board và Serial Port Mục Board : các bạn cần phải lựa chọn bo mạch cho phù hợp với loại bo mà bạn sử dụng nếu là Arduino Uno thì phải chọn như hình:



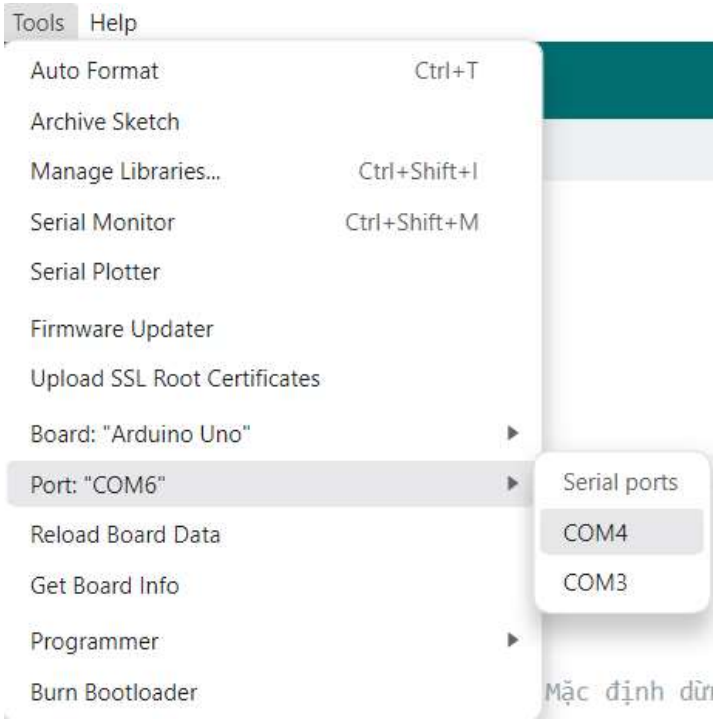
Hình 4.10 Chọn Board

Nếu các bạn sử dụng loại bo khác thì phải chọn đúng loại bo mà mình đang có nếu sai thì code Upload vào chip sẽ bị lỗi.

Serial Port: đây là nơi lựa chọn cổng Com của Arduino. Khi chúng ta cài đặt driver thì máy tính sẽ hiện thông báo tên cổng Com của Arduino là bao nhiêu, ta chỉ việc vào Serial Port chọn đúng cổng Com để nạp code, nếu chọn sai thì không thể nạp code cho Arduino được.

f) Một số lưu ý

Khi lập trình, cần chọn port (cổng kết nối khi gắn board vào) và board (tên board mà người dùng sử dụng). Giả sử, người dùng đang dùng mạch Arduino Uno, và khi gắn board này vào máy tính bằng cáp USB nó được nhận là COM4 thì người dùng chỉnh như thế này là có thể lập trình được.

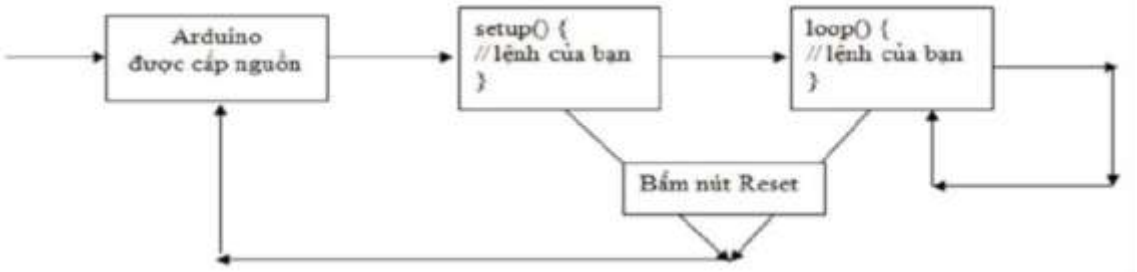


Hình 4.11 Cách tìm cổng COM

Cấu trúc ban đầu của chương trình trong lập trình Arduino IDE khá đơn giản, chỉ bao gồm hai hàm setup() và loop().

- Khi chương trình của chúng ta bắt đầu chạy, những lệnh trong setup() sẽ được chạy trước tiên. Do đó, ta thường dùng hàm này để khởi tạo trạng thái và giá trị của các biến. Mọi người cũng hay dùng hàm này để khởi tạo các thông số trong phần mềm ứng dụng.
- Sau khi setup() chạy xong, các lệnh trong loop() sẽ được chạy. Đây là một vòng lặp vô tận, do đó các dòng code trong hàm này sẽ được lặp lại mãi cho đến khi nào bạn ngắt nguồn của board Arduino mới thôi. Hoặc bạn có thể tắt nó bằng nút Reset trên bảng mạch. Lúc này, chương trình của bạn sẽ trở về lại trạng thái như khi mới bật, tức là bắt đầu chạy lại từ hàm setup().

Có thể xem quá trình này bằng hình dưới:



Hình 4.12 Quá trình lập trình Arduino IDE

*** Cấu trúc của một chương trình lập trình Arduino**

Phần 1 : Khai báo biến

- Đây là phần khai báo các yếu tố như: kiểu biến, tên biến, định nghĩa các chân cắm trên board. Một số kiểu khai báo biến thông dụng hay dùng:

```
* #define
```

- Define được dịch là định nghĩa. Hàm này có tác dụng định nghĩa, hay còn được hiểu là là gán: gán một chân cắm, một ngõ ra nào đó với 1 cái tên mình thích.

Ví d6 c6 thể: #define led 13

- Lưu ý : sau #define thì không có dấu phẩy
- Bạn có thể khai báo các kiểu biến khác như: int (kiểu số nguyên), float,…

Phần 2 : Thiết kế

- Phần này dùng để xây dựng cho chương trình. Bạn cần nhớ rõ cấu trúc của nó: void setup().
- Cấu trúc của nó sẽ có dấu ngoặc ở đầu và ở cuối. Nếu bị thiếu phần này khi chạy chương trình thì chương trình sẽ báo lỗi.
- Phần này dùng để quản lý các tốc độ truyền dữ liệu, kiểu chân ra hay chân vào.

Trong đó:

Serial.begin(9600); Dùng để truyền dữ liệu từ board Arduino lên laptop.

PinMode(biến, kiểu vào hoặc ra); Ví dụ: pin Mode(ChanDO, INPUT); được dùng để xác định kiểu chân cắm là vào hay ra.

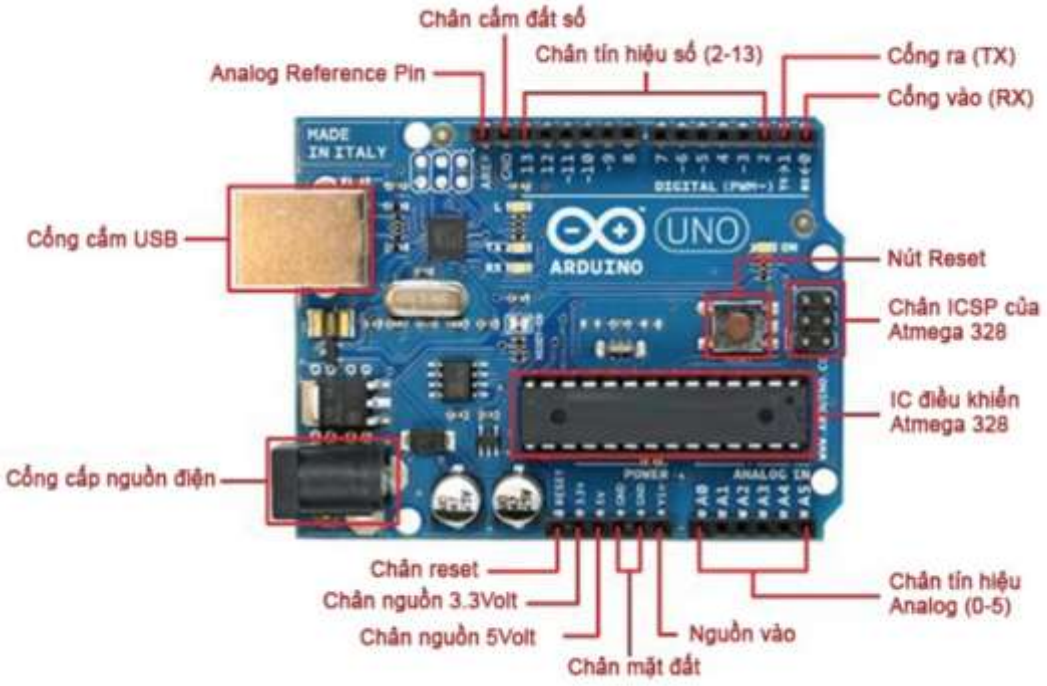
* Vòng lặp :

Dùng để viết các lệnh trong chương trình để board Arduino thực hiện các tính năng mà chúng ta mong muốn. Vòng lặp thường được bắt đầu bằng đoạn mã: void loop()

File mã nguồn (source code) của Arduino IDE có đuôi mở rộng là .ino. Để nhúng lệnh điều khiển vào Arduino thì cần biên dịch file .ino thành file .hex. File .hex chứa lệnh thực thi được biểu diễn dưới dạng hệ cơ số 16. Board mạch Arduino chỉ hiểu và chạy được những lệnh này

4.1.2. Giới thiệu bo mạch Arduino Uno R3

Arduino Uno là bo mạch vi xử lý hoạt động dựa trên ATmega328. Bo mạch này có 14 chân input/output digital (trong đó có 6 chân được dùng cho điều chế xung đầu ra PWM), 6 đầu vào analog, tần số giao động thạch anh là 16MHz, kết nối USB, jack cắm nguồn, chân tiêu đề ICSP, một nút reset. Bo mạch này chứa tất cả các tính năng cần thiết để hỗ trợ kết nối với các vi điều khiển khác. Nguồn sử dụng cho bo mạch có thể qua USB, sử dụng pin hoặc nguồn thông qua bộ chuyển đổi AC–DC.



Hình 4.13 Bo mạch Arduino Uno R3

Arduino Uno khác với tất cả các bo mạch khác ở chỗ nó không sử dụng chip điều khiển nối tiếp FTDI USB. Thay vào đó các tính năng của ATmega16U2 được lập trình để chuyển đổi USB nối tiếp.

Ở phiên bản thứ hai: bo mạch Uno có điện trở nối đường 8U2 HWB với đất, do đó ta dễ dàng hơn trong việc thiết lập chế độ DFU.

Ở phiên bản sửa đổi thứ 3 của bo mạch có các tính năng mới dưới đây:

- Sơ đồ chân 1.0: Thêm các chân SDA và SCL gần với chân AREF và 2 chân mới được đặt gần chân RESET, IOREF cho phép Shield nhận nguồn cấp từ bo mạch. Trong tương lai, Shield sẽ tương thích với các bo mạch có sử dụng AVR, có điện áp hoạt động là 5V và Arduino Due hoạt động ở 3.3V. Một số chân trong bo mạch không được kết nối gì để dành cho mục đích trong tương lai.

- Mạch Reset mạnh mẽ hơn.

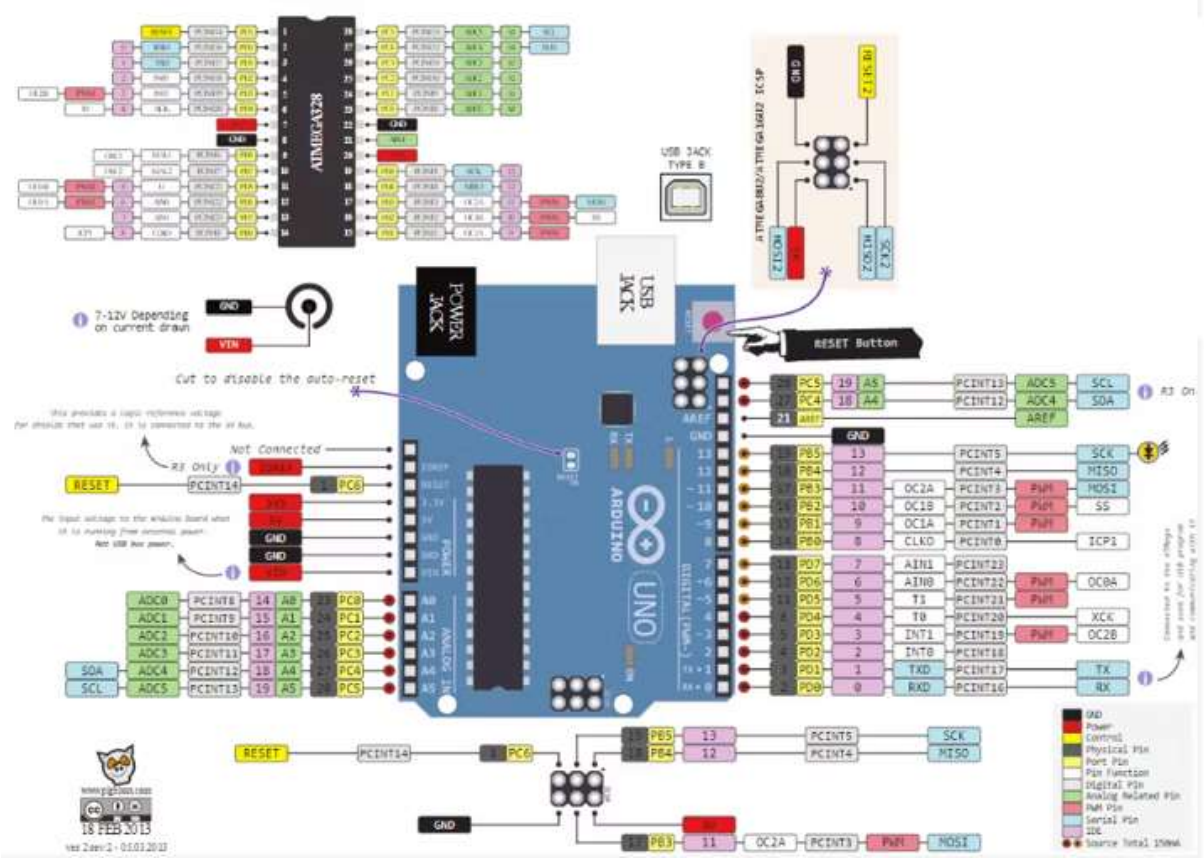
- ATmega 16U2 thay thế cho 8U2.

"Uno" là từ trong tiếng Ý và được đặt tên để đánh dấu việc phát hành phiên bản Arduino 1.0. Uno và phiên bản 1.0 sẽ là phiên bản tham khảo của Arduino, và luôn có sự cải tiến. Uno là một trong những bo mạch mới nhất trong một loạt các bo mạch USB Arduino, và là mô hình tham chiếu nền tảng của Arduino, để so sánh với phiên bản trước đó, xem các thông số của bo mạch Arduino.

Bảng 4.1 Tóm tắt các thông số chính của bo mạch Arduino Uno

Chip xử lý	ATmega328
Điện áp hoạt động	5V
Điện áp vào (khuyến nghị)	7-12V
Điện áp vào (giới hạn)	6-20V
Số chân I/O Digital	14 (6 chân đầu ra PWM)
Số chân đầu vào Analog	6
Dòng DC trên chân I/O	40 mA
Dòng DC trên chân 3.3V	50 mA
Flash Memory	32 KB (0.5 KB sử dụng cho bootloader)
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Tốc độ xung	16 MHz

4.1.3. Cấu tạo của Arduino Uno R3



Hình 4.14 Sơ đồ nối chân kit Arduino Uno

Trên bo mạch có 14 chân digital có thể được sử dụng cho mạch đích vào hoặc ra, sử dụng các hàm pinMode(), digitalWrite(), và digitalRead(). Chúng hoạt động ở mức điện áp 5V. mỗi chân có thể cung cấp hoặc nhận dòng cực đại là 40mA và được nối với điện trở mặc định từ 20-50 KΩ. Ngoài ra còn có một số chân có chức năng đặc biệt:

- Chân nối tiếp: 0 (RX) và 1 (TX). Sử dụng để nhận và truyền dữ liệu TTL. Chân này được nối với chân tương ứng của ATmega82U.

- Chân ngắt ngoài: 2 và 3. Chân này có thể được thiết lập để thực hiện ngắt khi điện áp quá thấp hoặc thay đổi đột biến giá trị điện áp. Thường sử dụng hàm ngắt attachInterrupt().

- PWM: 3, 5, 6, 9 và 10. Xung PWM có độ rộng là 8 bits. Khi điều chế xung đầu ra sử dụng hàm analogWrite().

- SPI: 10(SS), 11 (MOSI), 12 (MISO), 13 (SCK). Các chân này hỗ trợ truyền dẫn SPI sử dụng thư viện SPI.

- LED: 13. Có một đèn LED được nối với chân 13. Khi điện áp ở mức cao LED sáng và ở mức thấp thì LED tắt.

Uno có 6 đầu ra analog, có nhãn là A0 đến A5, mỗi chân được cung cấp 10 bits dữ liệu (tương ứng với 1024 giá trị khác nhau). Trên bo mạch Ethernet có 6 chân đầu vào tương tự, có nhãn từ A0 đến A5, mỗi chân sử dụng 10 bits (tức là có 1024 giá trị khác nhau). Nguồn cấp cho các chân này là từ 0-5V, mặc dù nó có thể thay đổi phạm vi hoạt động của chúng bằng cách sử dụng chân AREF và hàm analogReference(). Ngoài ra, một sô chân có chức năng chuyên biệt:

- TWI: A4 (SDA) và A5 (SCL). Hỗ trợ truyền thông TWI sử dụng thư viện Wire. Còn có hai chân khác trên bo mạch là:

- AREF: Điện áp tham chiếu cho đầu vào tương tự. Sử dụng với hàm analogReference().

- Reset. Được dùng để thiết lập lại vi điều khiển. Thường sử dụng nút reset để hỗ trợ các khối trên bo mạch.

Các thiết bị dựa trên nền tảng Arduino được lập trình bằng ngôn riêng. Ngôn ngữ này dựa trên ngôn ngữ Wiring được viết cho phần cứng nói chung. Và Wiring lại là một biến thể của C/C++. Một số người gọi nó là Wiring, một số khác thì gọi là C hay C/C++. Riêng mình thì gọi nó là “ngôn ngữ Arduino”, và đội ngũ phát triển Arduino cũng gọi như vậy. Ngôn ngữ Arduino bắt nguồn từ C/C++ phổ biến hiện nay do đó rất dễ học, dễ hiểu.

4.1.4. Tìm hiểu nguyên lý của các chân Arduino Uno

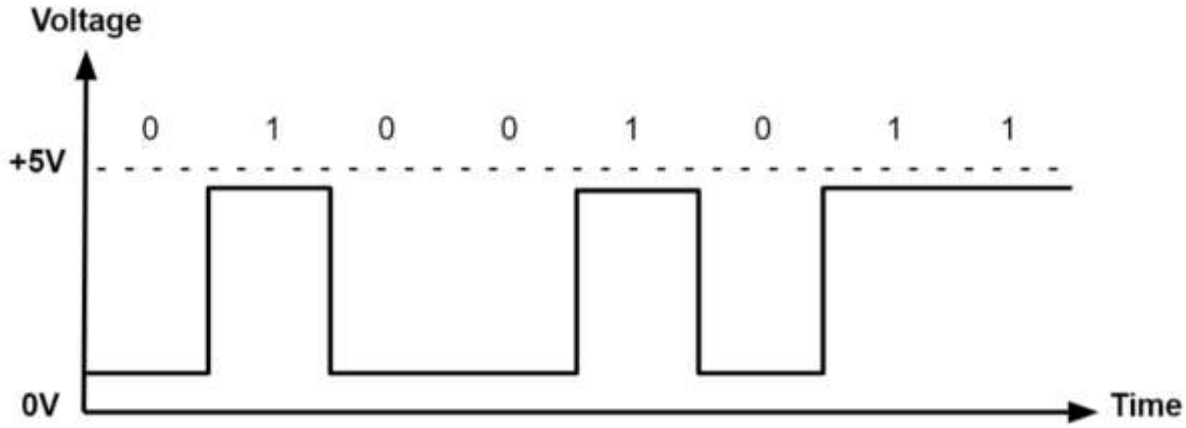
a, Tín hiệu Digital

Tín hiệu số là tín hiệu biểu diễn dữ liệu dưới dạng chuỗi các giá trị rời rạc. Tín hiệu số chỉ có thể nhận một giá trị từ một tập hợp hữu hạn các giá trị có thể tại một thời điểm nhất định. Với tín hiệu số, đại lượng vật lý biểu diễn thông tin có thể là nhiều thứ:

- Dòng điện hoặc điện áp thay đổi
- Pha hoặc phân cực của trường điện từ

- Áp suất âm thanh
- Độ từ hóa của phương tiện lưu trữ từ tính

Tín hiệu số được sử dụng trong tất cả các thiết bị điện tử kỹ thuật số, bao gồm thiết bị máy tính và thiết bị truyền dữ liệu. Khi được biểu diễn trên đồ thị điện áp so với thời gian, tín hiệu số là một trong hai giá trị và thường nằm trong khoảng từ 0V đến VCC (thường là 1,8V, 3,3V hoặc 5V) (xem Hình 2).



Hình 4.15 Tín hiệu Digital

*** Các chân Digital**

Gồm 14 chân (D0 đến D13). Có thể hoạt động ở chế độ Input (đầu vào) hoặc Output (đầu ra).

Một số chân có chức năng đặc biệt:

- D0 (RX) & D1 (TX): Dùng để giao tiếp UART (Serial).
- D2 - D13: Chân digital thông thường, có thể dùng làm GPIO (General Purpose Input/Output).
- D3, D5, D6, D9, D10, D11: Hỗ trợ PWM (Pulse Width Modulation).

*** Hoạt động như tín hiệu số (ON/OFF, 0V hoặc 5V)**.

Kỹ thuật số là cách biểu diễn điện áp theo 1 bit: 0 hoặc 1. Các chân kỹ thuật số trên Arduino là các chân được thiết kế để được cấu hình làm đầu vào hoặc đầu ra theo nhu cầu của người dùng. Các chân kỹ thuật số có thể bật hoặc tắt. Khi BẬT, chúng ở trạng thái điện áp CAO là 5V và khi TẮT, chúng ở trạng thái điện áp THẤP là 0V.

Trên Arduino, khi các chân kỹ thuật số được cấu hình là đầu ra , chúng sẽ được đặt ở mức 0 hoặc 5 vôn.

Khi các chân kỹ thuật số được cấu hình là đầu vào, điện áp được cung cấp từ một thiết bị bên ngoài. Điện áp này có thể thay đổi trong khoảng 0-5 vôn được chuyển đổi thành biểu diễn kỹ thuật số (0 hoặc 1). Để xác định điều này, có 2 ngưỡng:

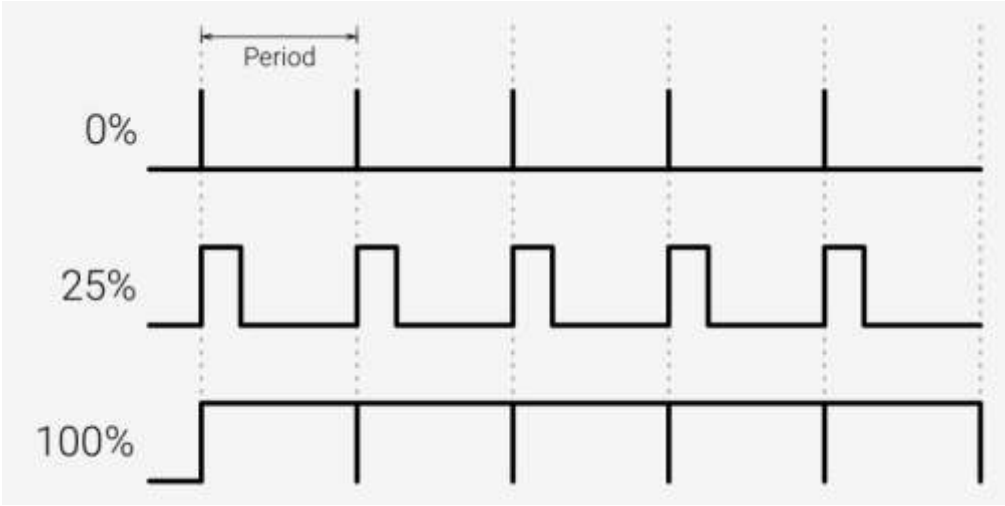
- Dưới 0,8v - được coi là 0.
- Trên 2v - được coi là 1.

Khi kết nối một thành phần với chân kỹ thuật số, hãy đảm bảo rằng các mức logic khớp nhau. Nếu điện áp nằm giữa các ngưỡng, giá trị trả về sẽ không xác định.

*** Hoạt động của xung PWM**

Điều chế độ rộng xung (PWM) là một kỹ thuật điều chế được sử dụng để mã hóa một thông điệp thành tín hiệu xung . PWM bao gồm hai thành phần chính: tần số và chu kỳ nhiệm vụ . Tần số PWM quyết định thời gian cần thiết để hoàn thành một chu kỳ (chu kỳ) duy nhất và tốc độ tín hiệu dao động từ cao xuống thấp. Chu kỳ nhiệm vụ xác định thời gian tín hiệu duy trì ở mức cao trong tổng chu kỳ. Chu kỳ nhiệm vụ được biểu thị theo phần trăm.

Trong Arduino, các chân hỗ trợ PWM tạo ra tần số không đổi ~ 500Hz, trong khi chu kỳ nhiệm vụ thay đổi theo các thông số do người dùng thiết lập. Xem hình minh họa sau:



Hình 4.16 Hoạt động của xung PWM

Tín hiệu PWM được sử dụng để điều khiển tốc độ động cơ DC, làm mờ đèn LED và nhiều mục đích khác.

PWM (Pulse Width Modulation - Điều chế độ rộng xung) mô phỏng tín hiệu analog bằng cách tạo sóng vuông có chu kỳ bật/tắt thay đổi.

Dùng analogWrite(pin, value); với value từ 0 đến 255.

- 0 → Giống như LOW (0V).
- 255 → Giống như HIGH (5V).
- 127 → Xung có 50% thời gian ở mức HIGH, 50% ở mức LOW (~2.5V trung bình).

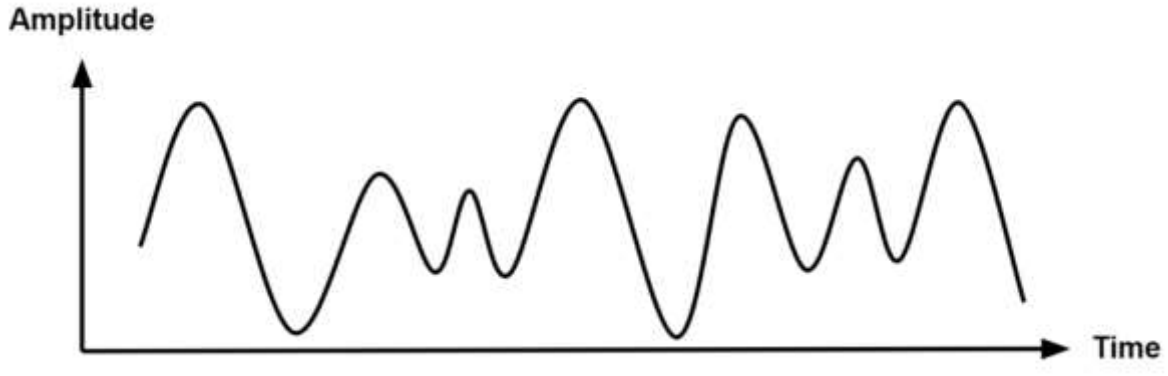
Tần số PWM mặc định trên Arduino Uno R3:

- D3, D9, D10, D11: ~490 Hz.
- D5, D6: ~980 Hz.

b, Tín hiệu Analog

Tín hiệu tương tự thay đổi theo thời gian và thường bị giới hạn trong một phạm vi (ví dụ: +12V đến -12V), nhưng có vô số giá trị trong phạm vi liên tục đó. Tín hiệu tương tự sử dụng một đặc tính nhất định của môi trường để truyền tải thông tin của tín hiệu, chẳng hạn như điện di chuyển qua dây dẫn. Trong tín hiệu điện, điện áp, dòng điện hoặc

tần số của tín hiệu có thể thay đổi để biểu diễn thông tin. Tín hiệu tương tự thường là phản ứng được tính toán đối với những thay đổi về ánh sáng, âm thanh, nhiệt độ, vị trí, áp suất hoặc các hiện tượng vật lý khác. Khi được biểu diễn trên đồ thị điện áp so với thời gian, tín hiệu tương tự phải tạo ra đường cong trơn tru và liên tục. Không được có bất kỳ thay đổi giá trị rời rạc nào (xem Hình 1).



Hình 4.17 Tín hiệu Analog

*** Các chân Analog**

Gồm 6 chân (A0 đến A5). Hoạt động như ADC (Analog-to-Digital Converter) để đọc tín hiệu analog (0V - 5V). Có thể dùng làm chân digital bình thường nếu cần.

Dùng analogRead(pin); để đọc điện áp từ **0V - 5V**. Trả về giá trị từ **0 - 1023** (độ phân giải 10-bit).

- 0 tương ứng với **0V**.
- 1023 tương ứng với **5V**.
- 512 tương ứng với **2.5V**.

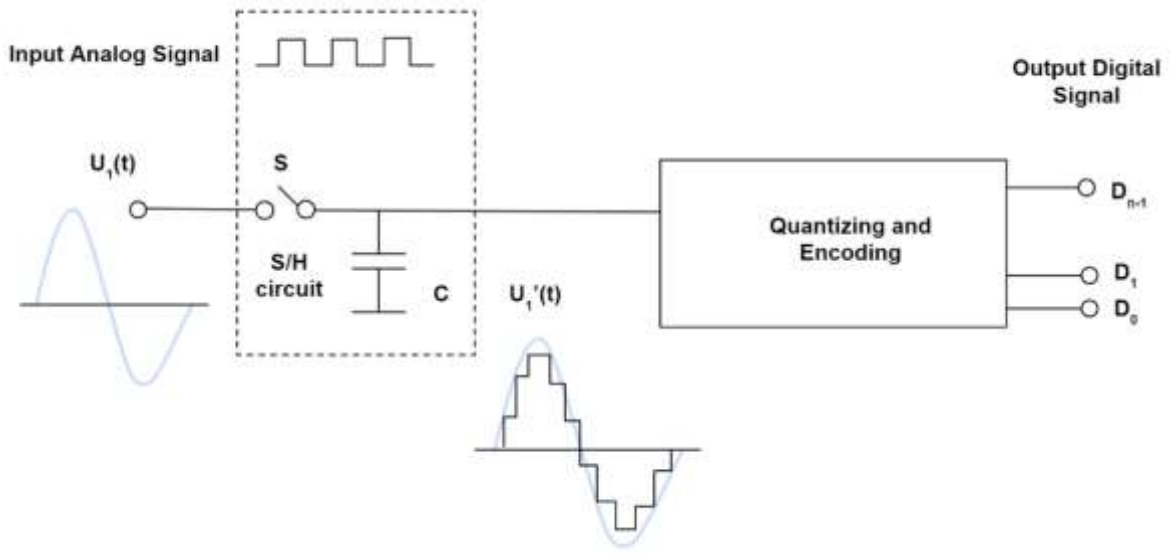
*** Hoạt động của tín hiệu Analog**

Arduino Uno không có DAC (Digital-to-Analog Converter), nên không thể phát tín hiệu analog thực sự.

Nếu cần phát tín hiệu analog, phải dùng PWM (các chân D3, D5, D6, D9, D10, D11).

c, Chuyển đổi tính hiệu Analog qua Digital (ADC)

Hình 4.18 cho thấy hoạt động của ADC. Đầu vào là tín hiệu tương tự, được xử lý thông qua mạch giữ mẫu (S/H) để tạo ra biểu diễn kỹ thuật số gần đúng của tín hiệu. Biên độ không còn có giá trị vô hạn nữa và đã được "lượng tử hóa" thành các giá trị rời rạc, tùy thuộc vào độ phân giải của ADC. ADC có độ phân giải cao hơn sẽ có kích thước bước nhỏ hơn và sẽ biểu diễn chính xác hơn tín hiệu tương tự đầu vào. Giai đoạn cuối cùng của ADC mã hóa tín hiệu đã số hóa thành luồng bit nhị phân biểu diễn biên độ của tín hiệu tương tự. Đầu ra kỹ thuật số hiện có thể được xử lý trong miền kỹ thuật số.

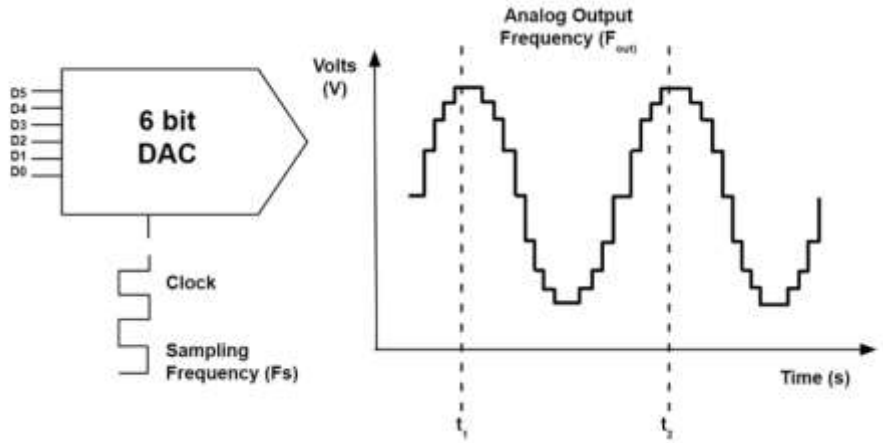


Hình 4.18 Kiến trúc ADC cho chuyển đổi tín hiệu tương tự sang tín hiệu số

d, Chuyển đổi tínhiệu Digital qua Analog (DAC)

DAC cung cấp hoạt động ngược lại. Đầu vào DAC là luồng dữ liệu nhị phân từ hệ thống con kỹ thuật số và đầu ra là giá trị rời rạc, được xấp xỉ như tín hiệu tương tự. Khi độ phân giải của DAC tăng lên, tín hiệu đầu ra sẽ xấp xỉ hơn với tín hiệu tương tự thực sự mượt mà và liên tục (xem Hình 7). Thường có một bộ lọc sau trong chuỗi tín hiệu tương tự để làm mịn hơn nữa dạng sóng. Như đã đề cập trước đó, nhiều hệ thống được sử dụng ngày nay là "tín hiệu hỗn hợp", nghĩa là chúng dựa vào cả hệ thống con tương tự và kỹ thuật số. Các giải pháp này yêu cầu ADC và DAC để chuyển đổi thông tin giữa hai miền.

Time	D5	D4	D3	D2	D1	D0
1/Fs	0	1	1	1	1	1
2/Fs	1	0	1	0	1	0
3/Fs	1	1	1	0	0	0
.	.	.	.	.	.	.
.	.	.	.	.	.	.
N-2/Fs	0	1	0	1	0	1
N-1/Fs	0	0	0	1	1	1
N/Fs	1	1	0	1	0	1



Hình 4.19 DAC 6-Bit để chuyển đổi tín hiệu số sang tín hiệu tương tự

4.2 Thiết kế hệ thống dò đường

4.2.1. Đề xuất phương án hệ thống dò đường

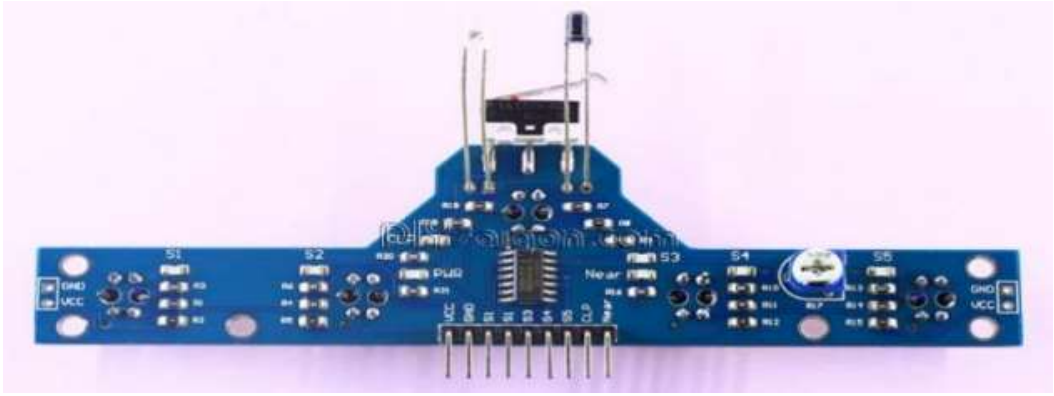
Phương án	Hình ảnh	Ưu điểm	Nhược điểm
Line Scan Camera		- Đạt độ phân giải cực cao vì nó quét từng dòng một và ghép lại thành ảnh hoàn chỉnh, có thể nhận biết các line phức tạp. - Hoạt động tốt trong nhiều điều kiện ánh sáng khác nhau. - Có thể theo dõi nhiều vạch line, hỗ trợ điều hướng linh hoạt hơn.	- Thời gian đáp ứng, giải thuật điều khiển phức tạp, tuỳ thuộc nhiều vào phần cứng của MCU. - Chi phí cao. - Chuỗi dữ liệu liên tục, đòi hỏi vi xử lý mạnh. - Nếu camera bị bám bụi hoặc rung lắc, có thể ảnh hưởng đến khả năng nhận diện đường line.
Cảm biến hồng ngoại		- Thời gian đạt giá trị xác lập ngắn 15 ns, dễ dàng thiết kế mạch cảm biến. - Tốc độ nhận diện line gần như tức thì, phù hợp với robot dò line tốc độ cao. - Tiêu thụ ít điện năng. - Giá thành rẻ. - Thời gian đáp ứng, giải thuật điều khiển đơn giản.	- Dễ bị ảnh hưởng nhiễu ánh sáng bởi môi trường. - Không linh hoạt với nhiều vạch line cùng lúc. - Phụ thuộc vào độ phản xạ của bề mặt. - Không nhận diện đường cong phức tạp tốt.

Để phù hợp độ tương xứng giữa nhiều hệ thống trên xe như giá thành rẻ, giải thuật điều khiển đơn giản, tốc độ nhận diện đường line nhanh.

⇨ Lựa chọn phương án 2: **Cảm biến hồng ngoại**

4.2.2. Đề xuất lựa chọn loại cảm biến hồng ngoại

a, Phương án 1: Đọc tín hiệu Digital bằng thanh cảm biến BFD-1000:



Hình 4.20 Thanh cảm biến dò line hồng ngoại BFD-1000

Ban đầu, tác giả quyết định sử dụng thanh cảm biến dò đường BFD-1000 như (hình 4.1) và thực hiện đọc tín hiệu digital. Trên thanh cảm biến có 5 cảm biến hồng ngoại hướng xuống đất nhằm phát hiện line, một cảm biến hồng ngoại đặt phía trước và đi cùng với nó là một công tắc hành trình báo hiệu đã đụng vật. Tín hiệu ngõ ra dạng số. Thanh cảm biến dò line hồng ngoại BFD-1000 có các thông số kỹ thuật như (bảng 4.2).

Bảng 4.2 Thông số kỹ thuật của thanh cảm biến dò line hồng ngoại BFD-1000

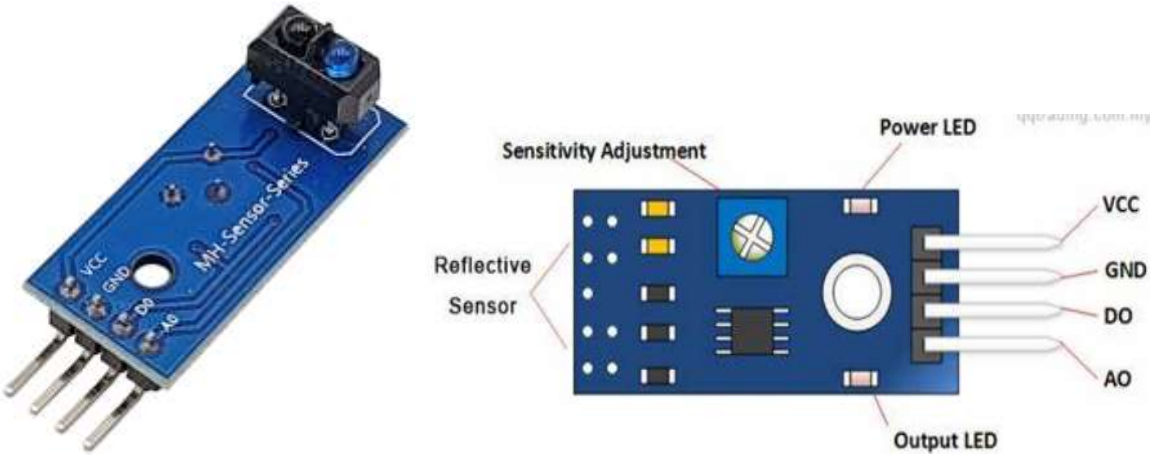
Thông số	Giá trị
Điện áp hoạt động	3,3 ~ 5V
Khoảng cách phát hiện	0,5 ~ 40mm
Ngõ ra	Tín hiệu số, 7 chân tín hiệu , 2 chân cấp nguồn
Số LED	5
Kích thước	128 × 45 × 12mm

Nhưng trong khi thực nghiệm, tác giả gặp các vấn đề sau:

- Khoảng cách giữa 2 cảm biến không phù hợp.
- Phải đặt thanh cảm biến sát đường line để đọc tín hiệu hiệu quả.
- Tín hiệu truyền về không ổn định.

Vì vậy, em quyết định thực hiện chuyển sang **phương án 2.**

b, Phương án 2: Đọc tín hiệu Analog bằng cảm biến TCRC 5000.



Hình 4.21 Cảm biến hồng ngoại TCRT 5000

Bảng 4.3 Thông số kỹ thuật của cảm biến TCR 5000

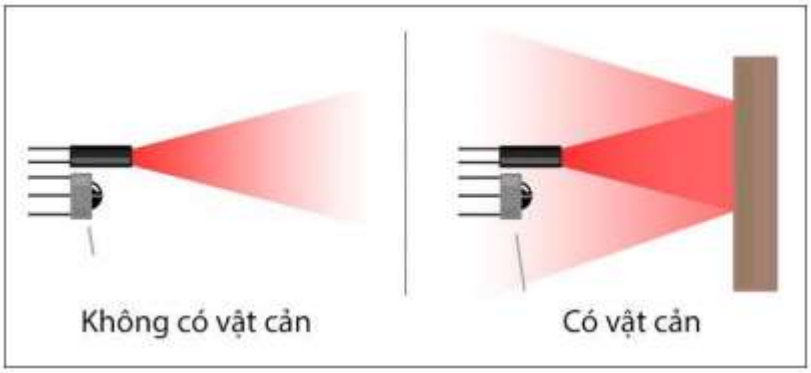
Thông số	Giá trị
Kích thước LxWxH (mm)	10,2×5,8×7 (mm)
Dòng I_C Dòng I_F	100 mA 60 mA
Khoảng cách hoạt động tốt nhất	2,5 mm
Phạm vi hoạt động	0,2 – 15 mm
Góc phát	16°
Góc thu	30°

4.2.3. Tổng quan về cảm biến hồng ngoại

Cảm biến hồng ngoại (hay còn gọi là IR Sensor), chúng là một thiết bị điện tử có khả năng phát hiện bức xạ hồng ngoại trong môi trường xung quanh. Thực tế, tên tiếng anh của cảm biến hồng ngoại là Passive Infrared, viết tắt là PIR dịch sát nghĩa là "hồng ngoại thụ động". Cảm biến hồng ngoại phát ra các tia vô hình với mắt người, vì bước sóng của nó dài hơn bước sóng ánh sang khả kiến (tức là ánh sang có thể nhìn thấy được). Bất cứ thứ gì phát ra nhiệt (mọi thứ nhiệt độ nằm trên 5 độ Kelvin) đều phát ra bức xạ hồng ngoại.

Hồng ngoại hay còn gọi là tia hồng ngoại là một bức xạ điện từ. Hồng ngoại tức là ngoài bước sóng đỏ. Màu đỏ là màu có bước sóng dài nhất trong ánh sang thường. Thông thường những vật thể có nhiệt độ trên 35 độ C sẽ phát ra bước song hồng ngoại. Bức xạ hồng ngoại được vô tình phát hiện ra bởi một nhà thiên văn học tên là William Herschel vào năm 1800. Trong khi đó nhiệt độ của từng màu ánh sang, ông nhận thấy rằng nhiệt độ vượt qua ngoài ánh sang đỏ là cao nhất. Có hai loại cảm biến hồng ngoại là cảm biến dạng chủ động và thụ động. Cảm biến hồng ngoại hoạt động bằng cách phát ra và phát hiện bức xạ hồng ngoại. Cảm biến hồng ngoại chủ động thường cấu tạo có 2 phần: diode phát sáng (LED) và máy thu. Khi một vật thể đến gần cảm biến, ánh sáng hồng ngoại từ

đèn LED sẽ phản xạ khỏi vật thể và được thiết bị thu phát hiện. Cảm biến hồng ngoại hoạt động đóng vai trò là cảm biến tiệm cận và thường được sử dụng trong các hệ thống phát hiện vật cản, ở đây là phát hiện phôi trong mô đun đựng phôi. Hồng ngoại thụ động có nghĩa là chỉ nhận các tia hồng ngoại phát ra từ các vật thể khác như người, động vật hoặc một nguồn nhiệt bất kì, chứ tự nó không phát ra tia hồng ngoại nào cả. Sauk hi nhận biết được nguồn nhiệt, bộ phận cảm biến sẽ phân tích để xác định điều kiện báo động.



Hình 4.22 Nguyên lí hoạt động của cảm biến hồng ngoại

4.2.4. Cơ chế hoạt động của cảm biến

Hình 4.23 Các kiểu đường line

Ta có thể sử dụng đường line trắng trên nền đen hoặc ngược lại sao cho cảm biến ở xe có thể nhận diện và chạy theo.

Hình 4.24 Cơ chế hoạt động của cảm biến

Cảm biến dò line hay được biết đến chính xác là cảm biến hồng ngoại hướng xuống bề mặt di chuyển. Chúng giúp phát hiện bề mặt phản xạ hoặc hấp thụ ánh sáng ở khoảng cách gần. Chúng thường có mắt hồng ngoại chuyên thu ánh sáng và mắt chuyên phát ánh sáng. Trường hợp mắt phát phát tín hiệu:

- Nếu bề mặt phản xạ lại ánh sáng, tín hiệu đó sẽ được mắt thu thu nhận → Từ đó ta xác định được tín hiệu và đưa ra vùng sáng xác định (Những bề mặt, vùng phản xạ gần như phản xạ hết những ánh sáng đi qua nó).

- Nếu bề mặt không phản xạ lại ánh sáng, không có tín hiệu về mắt thu → Từ đó ta không nhận được tín hiệu và xác định được vùng tối (Những bề mặt, vùng tối hấp thụ gần như hết ánh sáng đi qua nó).
- Khoảng làm việc của cảm biến: < 0.5m

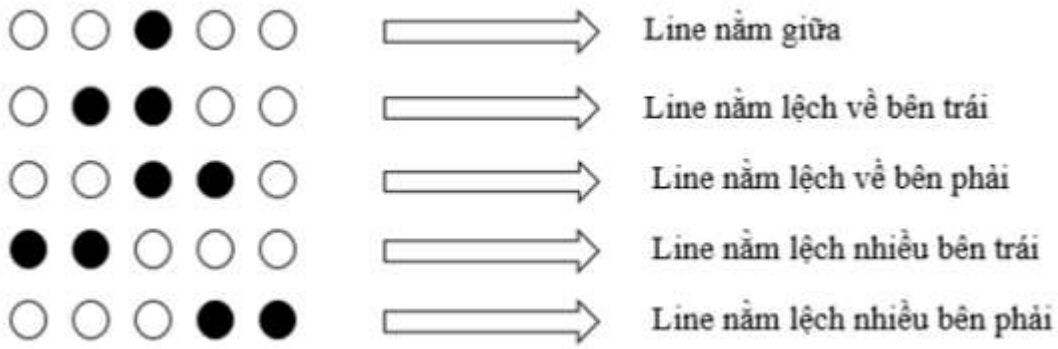
Như đã nói ở trên cảm biến dò line sẽ hoạt động theo nguyên lý thu – phát ánh sáng. Các mắt phát sẽ phát ra tia hồng ngoại có bước sóng hồng ngoại. Mắt thu ở bề mặt chứa vùng sáng sẽ hấp thụ ánh sáng hồng ngoại đó. Ở trạng thái bình thường mắt thu có nội trở rất lớn (hàng trăm Kilo-Ohm). Khi thu nhận tia hồng ngoại vào nội trở của mắt thu giảm (vài chục Kilo-Ohm).

Vì thế người ta gọi là Robot dò line do trong môi trường làm việc luôn có những đường Line có tính chất phản xạ ánh sáng cao. Đây sẽ là con đường di chuyển chủ yếu của Robot.

4.2.5. Phương pháp đọc tín hiệu cảm biến

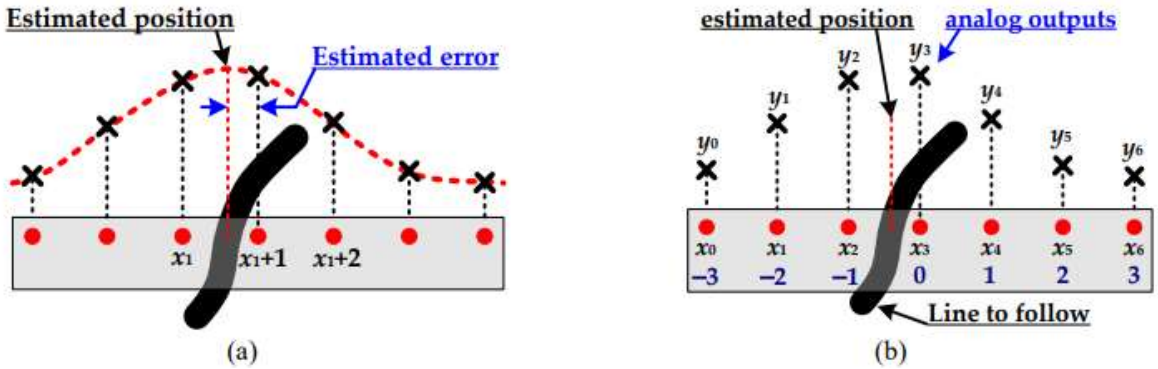
Đọc tín hiệu Digital: Tín hiệu đầu ra của cảm biến vẫn là analog nhưng sau đó thông qua mạch lấy ngưỡng hoặc lấy ngưỡng bằng lập trình để cho ra 2 giá trị logic 0 và 1 ứng với vị trí cảm biến trên đường line hoặc ngoài đường line.

Để điều khiển robot theo quỹ đạo, người thiết kế lập trình xác định độ lệch tương đối giữa quỹ đạo của robot và quỹ đạo mong muốn, sau đó so sánh độ lệch đó thành các mức và điều khiển lái robot quay về quỹ đạo. Sai số dò line phụ thuộc vào khả năng phân biệt các trạng thái của hệ thống, hay khoảng cách giữa các sensor, do đó tốc độ xử lý rất nhanh.



Hình 4.25 Tín hiệu đọc về digital

Đọc tín hiệu analog: Đọc theo dạng analog tín hiệu analog đọc được từ cảm biến qua phép xấp xỉ để tìm ra vị của xe so với tâm đường line. Các giải thuật xấp xỉ theo bậc 2, theo trọng số cho sai số dò line khác nhau. Thuật toán được mô tả như (hình 1.23). Thời gian xử lý phụ thuộc vào thời gian đọc ADC tất cả các sensor của vi điều khiển, do đó sẽ lâu hơn phương pháp thứ nhất, tuy nhiên độ chính xác cao hơn nhiều.



Hình 4.26 Thuật toán phát hiện đường thẳng thông qua a) nội suy bậc hai và b) trung bình có trọng số

Các giá trị đọc về qua phép xấp xỉ để tìm ra vị trí của đường line. Phương pháp này cho ra sai số phát hiện line nhỏ, có thể tới 2.6mm

Bảng 4.4 Bảng so sánh tín hiệu đọc analog và digital

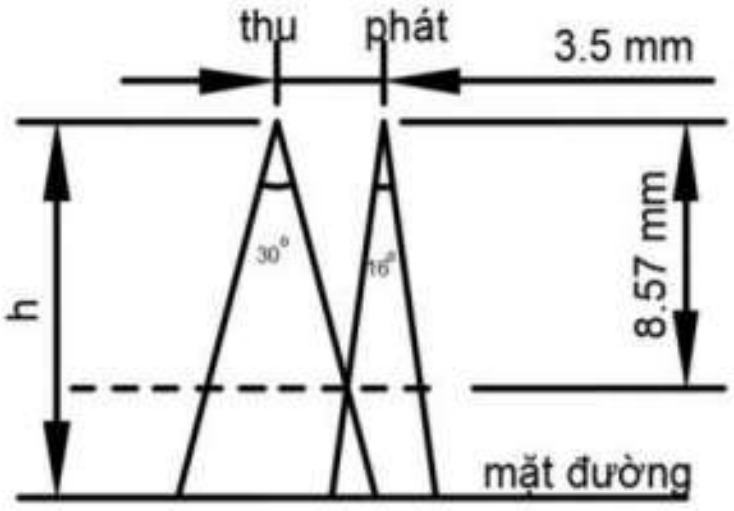
	Đọc tín hiệu analog	Đọc tín hiệu digital
Độ chính xác	Cao	Thấp
Thời gian đọc tín hiệu	Tốn nhiều thời gian	Tốn ít thời gian
Giải thuật xử lý	Phức tạp	Đơn giản

4.2.6. Thiết kế bố trí cảm biến hồng ngoại

Cảm biến hồng ngoại sử dụng cảm biến ánh sáng nên có sơ đồ rất giống cảm biến ánh sáng. Sự khác biệt duy nhất là việc bổ sung LED hồng ngoại và đầu dò hồng ngoại yêu cầu kết nối 5V và nối đất.

Khoảng cách giữa cảm biến và đường line:

- Góc chiếu của Emitter: α = 16°.
- Góc chiếu của Detector: β = 30°.
- Khoảng cách giữa Emitter và Detector : d = 3,8 $m\,m$.



Hình 4.27 Vùng hoạt động của 1 cặp cảm biến hồng ngoại

Xác định độ dài L :

$$L \times \tan(\alpha) + L \times \tan(\beta) = d \quad (4.1)$$

Nên:

$$L = \frac{d}{\tan\alpha + \tan\beta} = 9,3mm \quad (4.2)$$

Do đó khoảng cách giữa cảm biến với mặt đường để xuất hiện vùng giao thoa là: h + 0,7 mm > L = 9,3 mm → h > 8,6 mm

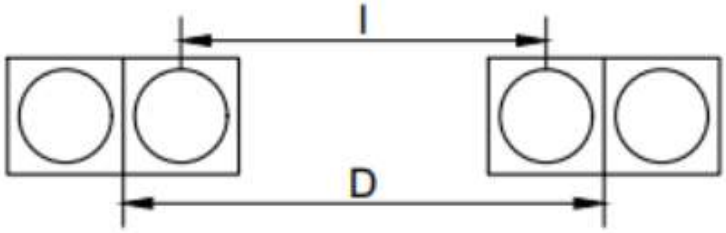
Từ thực nghiệm tiến hành kiểm tra, khoảng cách tối ưu với sàn cho cảm biến được chọn: h = 10 mm. (thể hiện trong bản vẽ 2D)

Khoảng cách giữa các cảm biến: Để các cảm biến hoạt động tốt, cùng hoạt động của 2 cảm biến phải tách biệt nhau, không được chồng lên nhau. Với khoảng cách từ cảm biến và đường line h = 10 mm, bán kính của đường tròn trên mặt sàn được tính toán như sau:

Led phát: $r_{phát}$ = (□ + a) × tan$\frac{\alpha}{2}$ (4.3)

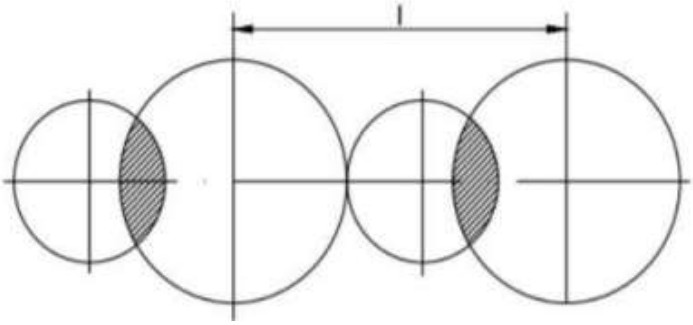
Led thu: R_{thu} = (□ + a) × tan$\frac{\beta}{2}$

- Led phát: $r_{phát}$ = (□ + a) × tan8 = (10 + 0,7) × tan 8 = 1,5 (mm). (4.3)
- Led thu: R_{thu} = (□ + a) × tan15 = (10 + 0,7) × tan 15 = 2,87(mm). (4.4)



Hình 4.28 Khoảng cách D giữa hai cảm biến

Phạm vi quét của 2 cảm biến liền kề được mô tả:



Hình 4.29 Phạm vi quét của 2 cảm biến liền kề

Từ (H4.29), khoảng cách l_1 được xác định:

$$D = r_{phát} + R_{thu} = (1{,}5 + 2{,}87) = 4{,}37(m\,m). \quad (4.5)$$

Khoảng cách tối thiểu D giữa 2 cảm biến để không bị nhiễu: D > 4,37 ($m\,m$).

4.3. Thiết kế hệ thống đo khoảng cách

4.3.1. Đề xuất phương án lựa chọn cảm biến đo khoảng cách

Phương án	Hình ảnh	Ưu điểm	Nhược điểm
Cảm biến khoảng cách laser		- Đo khoảng cách chính xác từ 10mm đến 1.8m với sai số nhỏ (~2-3mm). - Sử dụng công nghệ TOF nên thời gian phản hồi rất nhanh (~30ms). - Không bị ảnh hưởng nhiều bởi ánh sáng môi trường xung quanh. - Hỗ trợ I2C và UART, có thể dễ dàng kết nối với Arduino, ESP32, STM32...	- Các vật liệu hấp thụ ánh sáng (màu đen, bề mặt mờ) có thể làm giảm độ chính xác. - Cảm biến chỉ có thể đo chính xác trong một vùng nhỏ (~3°). - Vì dùng tia laser, bụi hoặc sương có thể làm nhiễu tín hiệu đo, dẫn đến sai số.
Cảm biến khoảng cách siêu âm		- Khoảng cách đo rộng từ 2cm đến 4m. - Hoạt động ổn định trong nhiều điều kiện ánh sáng môi trường. Sai số thấp (~3mm)	- Bị nhiễu trong môi trường có nhiều tiếng ồn siêu âm. - Không hoạt động tốt trong môi trường có gió mạnh hoặc thay đổi áp suất không khí đột ngột.

		- Dễ dàng giao tiếp với vi điều khiển. - Thuật toán điều khiển đơn giản. - Giá thành rẻ.	- Sóng siêu âm tỏa theo hình nón (~15°), nên nếu có nhiều vật thể trong vùng quét, cảm biến có thể đo sai. - Mỗi lần đo mất khoảng 60ms, chậm hơn so với cảm biến laser (TOF).

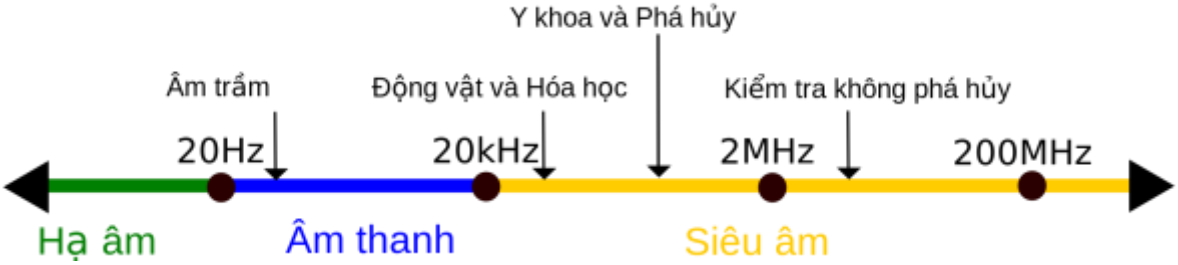
Để phù hợp với giá thành rẻ, đo xa (4m), không bị ảnh hưởng bởi ánh sáng, dễ dùng với vi điều khiển, thuật toán điều khiển dễ dàng nên chọn **phương án 2: Cảm biến siêu âm HC-SR04.**

4.3.2. Tổng quan về sóng âm

Trong vật lý, sóng âm là những dao động được lan truyền trong môi trường vật chất theo thời gian. Cụ thể, trong quá trình dao động, các phân tử thuộc môi trường bao quanh sẽ được nhận một phần năng lượng từ nguồn gây ra dao động và bắt đầu lệch khỏi vị trí cân bằng; dưới tác dụng của lực liên kết đàn hồi giữa các phần tử, chúng sẽ quay trở lại vị trí cân bằng ban đầu nhưng sau đó tiếp tục chuyển động theo quán tính...

Nói cách khác, sự dịch chuyển của mỗi phần tử làm cho những phân tử bên cạnh lệch khỏi vị trí cân bằng và thực hiện dao động. Kết quả của hiện tượng này đó là dao động đã được truyền lan từ nguồn phát dao động cho đến các phần tử ở xa. Quá trình lan truyền sóng của sóng âm và sóng điện từ đó là hoàn toàn khác nhau: sự truyền sóng âm chỉ thực hiện được trong môi trường có xuất hiện lực liên kết đàn hồi hay nói cách khác môi trường lan truyền là môi trường vật chất đàn hồi trong khi sóng điện từ có thể truyền được trong chân không. Căn cứ vào tần số sóng âm phát ra, sóng âm được chia làm ba loại như hạ âm, sóng âm gây cảm giác âm và siêu âm. Sóng siêu âm là sóng âm với tần số từ 16KHz trở lên nằm ngoài khả năng nghe được của tai người. Giới hạn trên của tần số sóng siêu âm thường là 5 MHz đối với chất khí và 500 MHz đối với chất lỏng hay chất rắn. Trong thực tế, sóng siêu âm được chia ra thành sóng siêu âm tần số thấp, năng lượng cao (20kHz-100kHz) và sóng siêu âm tần số cao, năng lượng thấp (2MHz-10MHz).

Siêu âm là một loại âm thanh mà vượt qua khỏi giới hạn nghe của tai người. Thông thường tai người có thể nghe được tần số trong giới hạn 20.000 Hz trở xuống. Còn siêu âm thì thường có tần số từ 20.000Hz trở lên.



Hình 4.30 Các biên dạng tần số của âm thanh

Ngoài siêu âm thì ta còn có 1 khái niệm khác là hạ âm. Hạ âm là loại âm thanh có tần số rất thấp, chỉ vào khoảng dưới 20 Hz. Vậy thì cảm biến siêu âm chính là 1 loại cảm biến hoạt động dựa trên sóng siêu âm. Ta có thể hiểu đơn giản là vậy.

4.3.3. Giới thiệu cảm biến siêu âm 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.

Cảm biến siêu âm HC-SR04 thường được kết hợp với các bộ arduino, PIC, AVR,… để chạy một số ứng dụng như : phát hiện vật cản trên xe robot, đo khoảng cách vật,…

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.

Hình 4.31 Cảm biến siêu âm HC-SR04

Thông số kỹ thuật của cảm biến:

- Nguồn điện: +5V DC
- Điện áp làm việc: 5VDC
- Dòng điện làm việc: 15mA
- Góc hiệu dụng: <15º
- Khoảng cách dao động: 2cm - 400 cm
- Độ phân giải: 0.3 cm
- Độ rộng xung đầu vào kích hoạt: 10uS
- Kích thước: 45mm x 20mm x 15mm
- Trọng lượng xấp xỉ: 10 g

4.3.4. Cấu tạo và chức năng cảm biến siêu âm HC-SR04

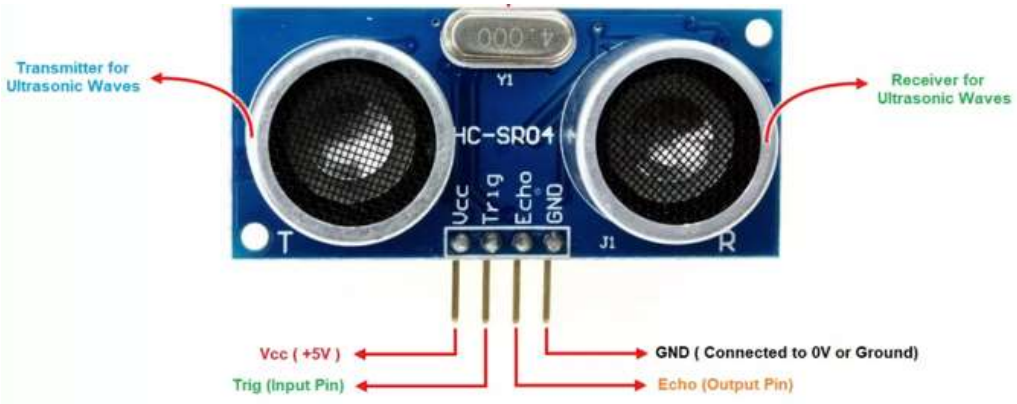
* Cấu tạo của cảm biến siêu âm HC-SR04 gồm 2 phần:

Bộ phận phát sóng siêu âm

- Phần phát (chân TRIG) của cảm biến siêu âm chứa một bộ dao động điện,tạo ra sóng siêu âm với tần số cao qua một tấm màng lọc trước khi tiếp xúc với môi trường.

Bộ phận thu sóng siêu âm

- Khi sóng siêu âm tương tác đối tượng thì nó sẽ phản xạ lại về phần thu. Sóng phản xạ thu được và chuyển đổi thành tín hiệu điện

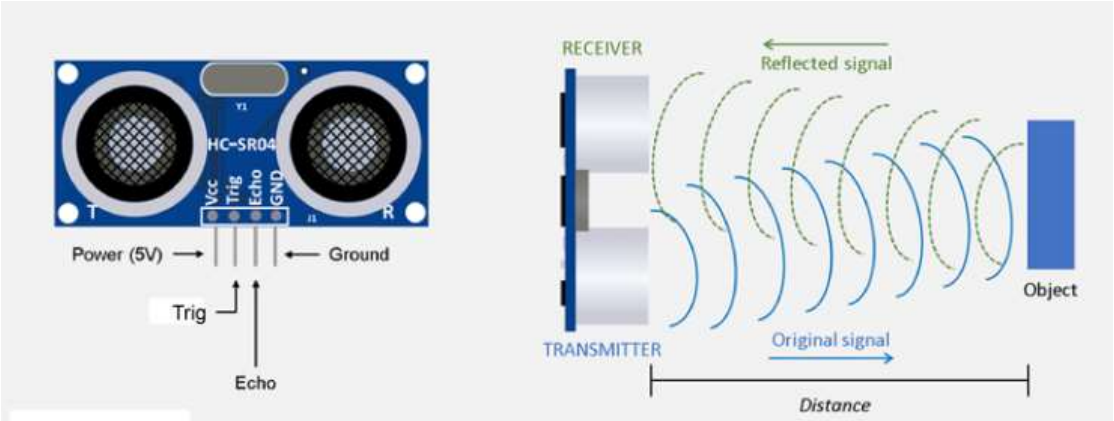


Hình 4.32 Phát thu sóng trên cảm biến siêu âm HC-SR04

Tất cả các chân này đều có chức năng riêng biệt.

- Chân VCC sẽ được dùng dể nối với nguồn nuôi cho cảm biến, mức điện áp tối đa theo nhà sản xuất là 5V.
- Chân GND là chân nối đất, dùng để kết nối với nguồn điện.
- Chân TRIG chịu trách nhiệm gửi xung siêu âm. Để tạo ra xung siêu âm, chân này phải được đặt ở mức HIGH trong 10 µs. HCSR04 khi đó sẽ phát ra một đợt sóng âm 8 chu kỳ ở tần số 40 kHz. Sau khi xung được gửi đi, chân ECHO sẽ chuyển sang mức HIGH.
- Chân ECHO là chân dữ liệu được sử dụng để đo khoảng cách. Nó sẽ duy trì ở mức HIGH cho đến khi xung quay trở lại cảm biến, và ngay tại thời điểm đó, nó sẽ chuyển về mức LOW. Nếu những xung này không được phản xạ trở lại, chân ECHO sẽ chuyển sang trạng thái LOW sau 38ms (38 mili giây).

4.3.5. Nguyên lý hoạt động cảm biến siêu âm HC-SR04



Hình 4.33 Nguyên lý hoạt động của cảm biến siêu âm HC-SR04

Cảm biến siêu âm HC-SR04 có bốn chân là chân Vcc, chân kích hoạt, chân Echo và chân nối đất. Cảm biến này dùng để đo khoảng cách chính xác giữa mục tiêu và cảm biến. Cảm biến này chủ yếu hoạt động trên sóng âm thanh.

Khi nguồn điện được cung cấp cho mô-đun này, nó sẽ tạo ra sóng âm thanh truyền trong không khí để đập vào vật thể cần thiết. Những sóng này tấn công và quay trở lại từ vật thể, sau đó thu thập bởi mô-đun máy thu.

Ở đây cả khoảng cách cũng như thời gian đã đi đều tỷ lệ thuận với nhau vì thời gian đi cho quãng đường nhiều hơn là cao. Nếu chân kích hoạt được giữ ở mức cao trong 10 μs, thì sóng siêu âm sẽ được tạo ra và sẽ truyền đi với tốc độ âm thanh. Vì vậy, nó tạo ra tám chu kỳ bùng nổ âm thanh sẽ được tập hợp trong chân Echo. Cảm biến siêu âm này được giao tiếp với Arduino để đo khoảng cách cần thiết giữa cảm biến và đối tượng. Khoảng cách có thể được tính bằng công thức sau.

Độ rộng của xung nhận được được sử dụng để tính khoảng cách từ vật thể phản xạ. Điều này có thể được giải quyết bằng cách sử dụng phương trình liên hệ giữa 3 biến khoảng cách-tốc độ-thời gian đơn giản như sau:

Phương trình 1:

$$S = \frac{v.t}{2} \tag{4.6}$$

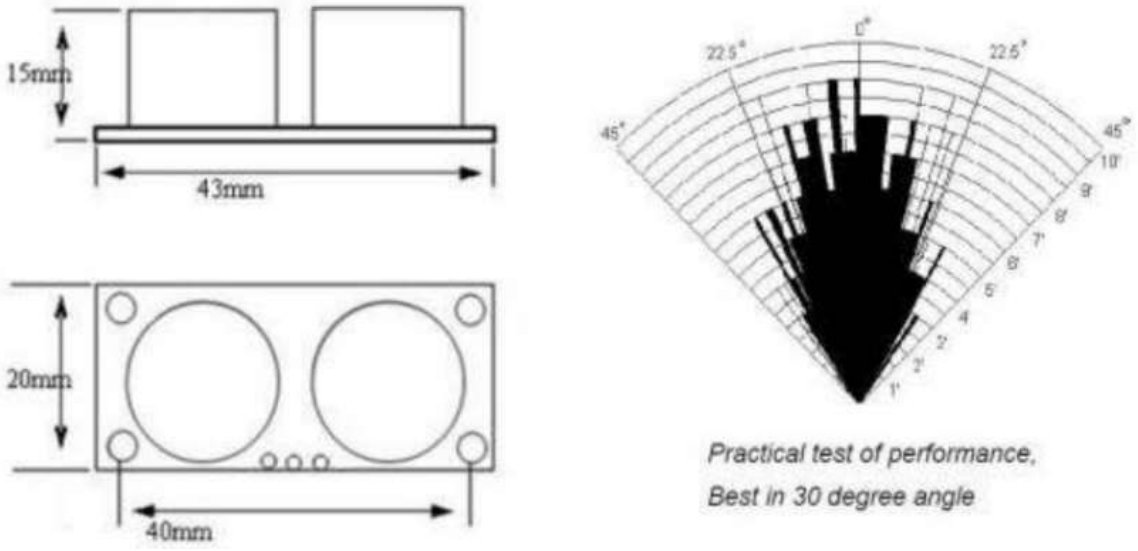
Trong đó chữ ‘S’ là khoảng cách bắt buộc

- v là tốc độ âm thanh (343m/s)
- t là thời gian cần thiết để sóng âm quay trở lại sau khi đập vào vật.
- Khoảng cách bị chia 2 vì sóng âm phải đi từ cảm biến đến vật và quay ngược lại.
- Khoảng cách thực tế có thể được tính bằng cách chia giá trị của nó cho 2 vì thời gian sẽ gấp đôi khi sóng truyền đi và nhận lại từ cảm biến.

Cụ thể, để đo khoảng cách bằng cảm biến siêu âm HC-SR04, ta sẽ phát 1 xung rất ngắn (5 microSeconds) từ chân Trig. Tiếp theo, 1 xung HIGH ở chân Echo sẽ được cảm biến tạo ra và phát đi cho đến khi nhận lại được sóng phản xạ ở chân này. Lúc này, độ rộng của xung sẽ bằng với thời gian sóng siêu âm được phát từ cảm biến và phản xạ lại.

Trong không khí, tốc độ âm thanh đạt mức 340 m/s (hằng số), tương đương với 29,412 microSeconds/cm (106 / (340*100)).

Khi đã tính được thời gian, ta sẽ chia cho 29,412 để ra giá trị khoảng cách.



Hình 4.34 Kích thước và song cảm biến siêu âm 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.

4.3.6. Các yếu tố dẫn đến lỗi và sai số cho cảm biến

Sóng âm là những dao động được lan truyền trong môi trường vật chất theo thời gian, vì vậy các kết quả cho ra bởi cảm biến sẽ bị ảnh hưởng bởi những yếu tố ngoại quan. Yếu tố đầu tiên là môi trường. Nhiệt độ không khí có ảnh hưởng lớn nhất đến độ chính xác đo của cảm biến siêu âm. Sau khi đã đo được thời gian truyền của xung siêu âm phản xạ, cảm biến sẽ tính toán khoảng cách đến vật thể bằng cách sử dụng tốc độ của âm thanh. Tuy nhiên, khi nhiệt độ không khí thay đổi. Kết quả đo cũng phụ thuộc nhiều vào đối tượng được thực nghiệm. Với những vật có kích thước bề mặt nhỏ hơn $0.5m^2$ hoặc phản xạ không tốt, cảm biến sẽ cho ra kết quả không ổn định và chính xác.

Khi xảy ra vấn đề nhiệt độ cao thì lượng hơi nước bay lên bám vào đầu phát sóng gây ảnh hưởng đến kết quả đo; độ chính xác.

4.3.7. Thiết kế thực nghiệm đo khoảng cách trên cảm biến

*** Quy trình thiết kế thực nghiệm:**

Để tra các đặc tính của cảm biến ta thực hiện lần lượt những thí nghiệm cụ thể để kiểm định và đưa ra những thông số kĩ thuật, những yếu tố ảnh hưởng đến sự hoạt động bình thường của cảm biên siêu âm SR04. Gồm các đặc trưng về nhiệt độ môi trường,

đặc trưng vật lý của cảm biến, từ đó đưa ra các sai số của phép đo, các cách hiệu chỉnh cảm biến để hướng đến kết quả đo gần giá trị chuẩn nhất. Quy trình thiết kế và phân tích thực nghiệm gồm các bước:

Bước 1: Chọn lựa kế hoạch thực nghiệm.

Bước 2: Thực hiện thực nghiệm.

Bước 3: Phân tích thống kê dữ liệu thực nghiệm.

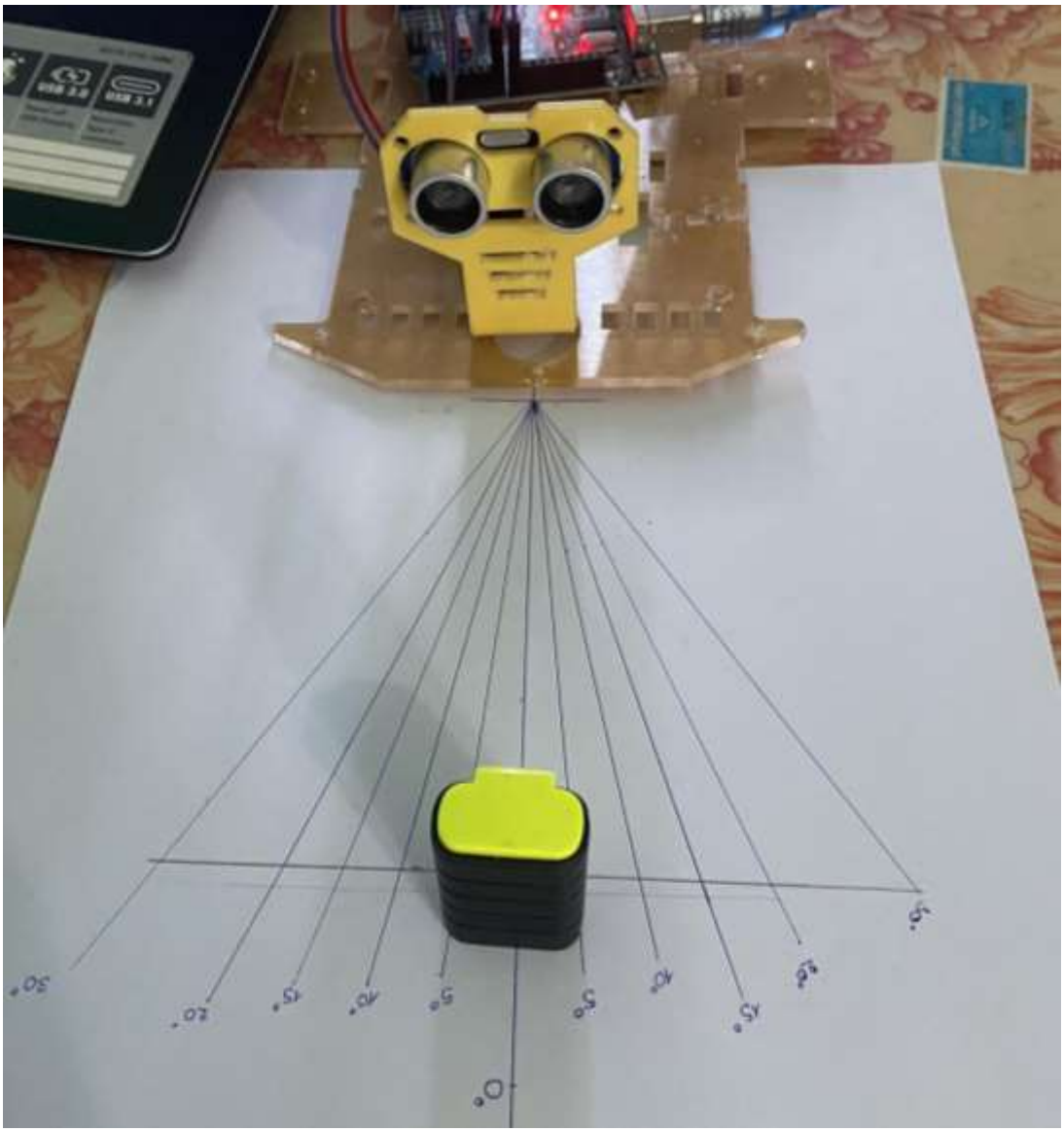
Kiểm tra các đặc trưng vật lý của cảm biến HC-SR04:

*** Xác định góc mở của cảm biến siêu âm:**

Đầu tiên ta để cảm biến trên một giá đỡ và đặt trên một mặt phẳng có chia góc như trong hình bên dưới. Sau đó ta đưa 1 vật có kích thước vừa đủ đi từ điểm chính giữa so với cảm biến dần ra xa về hai phía rồi thu lại kết quả đo vào bảng.

Bảng 4.5 Xác định góc mở của cảm biến

Góc	0°	5°	10°	15°	16°	20°
Giá trị đo được	11.96 11.96 11.87 12.04 11.97 12.00 11.73 12.10 12.00 11.97	12.07 12.10 12.10 11.83 12.07 12.14 12.21 11.97 12.14 12.14	12.38 12.04 11.93 12.38 12.38 11.70 11.93 12.04 12.04 12.38	12.99 13.12 12.99 13.09 13.19 12.75 12.72 13.09 12.58 12.55	13.40 13.8 13.84 14.69 14.01 13.94 14.69 14.69 13.84 13.84	14.28 2263.43 2263.33 251.09 16.05 15.88 17.20 354.18
Giá trị trung bình	11.963	12.077	12.12	12.907	14,074	522.703
Sai số tuyệt đối trung bình	0.037	0.077	0.12	0.907	2.074	510.703



Hình 4.35 Xác định góc mở cảm biến

Kết luận:

Ta thấy khi đặt vật trong khoảng 0° đến 15° cảm biến đưa ra kết quả đo ổn định với sai số nhỏ.

Từ 16° bắt đầu có sự chênh lệch khoảng cách đáng kể và đặc biệt là khi vật ở 20° thì cảm biến không đo chính xác được nữa.

Xác định khoảng cách làm việc tối đa và tối thiểu của cảm biến:

Bảng 4.6 Kiểm tra khoảng cách làm việc của cảm biến

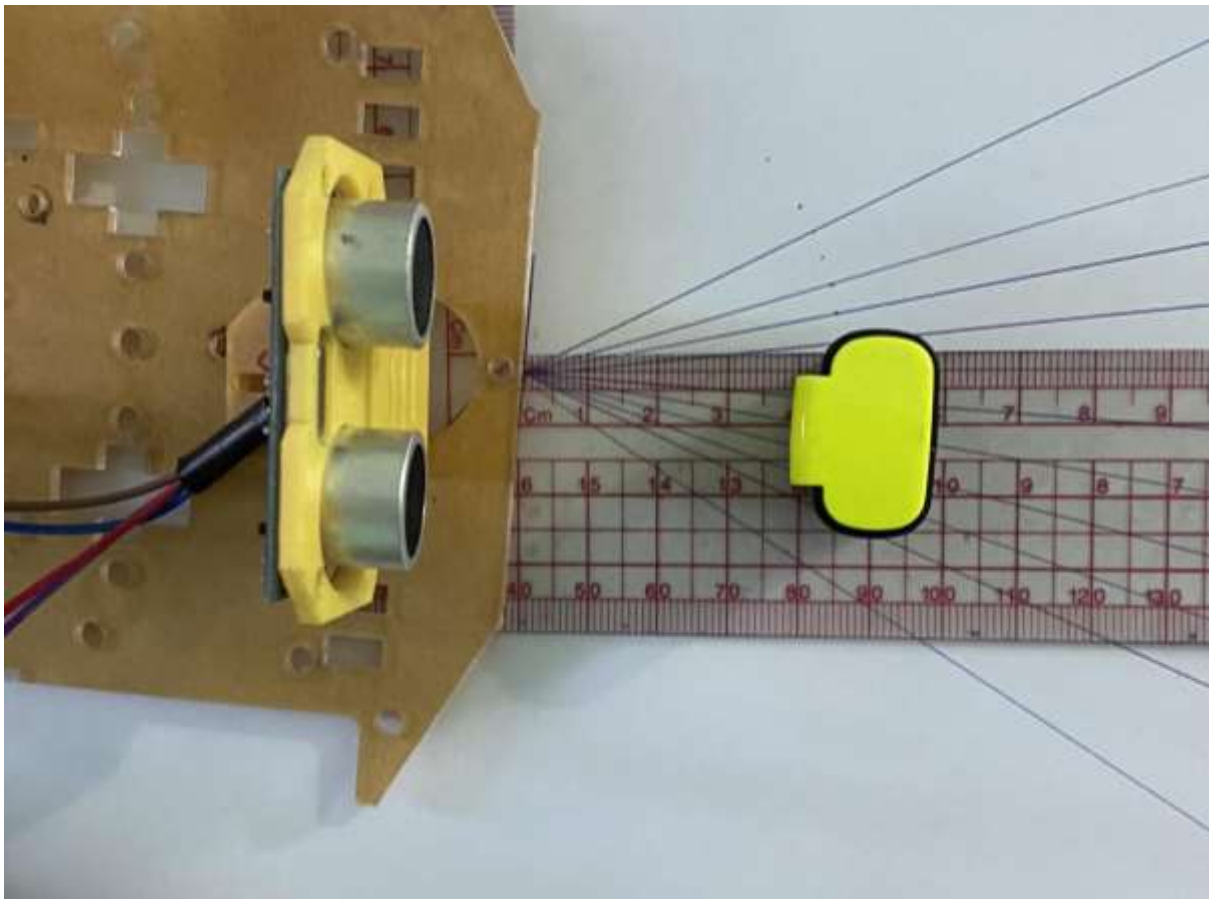
	1cm	2cm	3 cm	4 cm	100cm	200cm	250cm	300cm	400cm	410cm
Giá trị đo được	2265.2	6.6	3.21	4.11	99.45	199.48	245.95	293.11	392.15	2248.0
	2265.2	3.60	3.21	4.11	99.89	197.91	245.24	293.86	392.32	2247.9
	2265.2	4.28	3.14	4.08	98.91	200.19	245.34	293.86	393.17	404.90
	2266.2	4.59	2.86	4.08	98.91	197.67	245.34	293.4	392.42	2248.9
	2265.6	3.09	3.24	4.05	98.94	198.93	246.2	294.2	392.42	403.64
	2267.7	3.2	3.24	3.76	100.1	197.95	245.27	294.06	391.91	403.15

	2265.4 2265.7 2266.8 2266.1 2265.8 2266.8 2265.9 2266.3 2267.4	3.20 3.20 3.09 3.60 3.60 3.13 3.43 2.96 3.13	3.21 3.24 3.54 2.60 2.94 2.84 3.64 3.20 3.33	4.05 3.64 4.05 4.05 4.05 3.94 3.64 3.94 3.76	98.91 100.16 99.89 99.01 98.94 99.42 99.86 99.76 98.94	198.05 199.41 197.91 199.88 198.42 199.03 198.08 198.46 199.88	245.4 245.68 245.48 245.89 245.41 245.95 245.44 245.51 246.43	293.59 293.55 294.10 294.40 293.93 294.37 293.4 293.52 293.86	392.87 392.80 394.06 391.95 393.65 393.72 394.12 392.91 392.42	2248.7 2248.6 2250.5 2249.6 2250.6 2251.2 2250.2 2249.1 2249.2
Giá trị trung bình	2266.0	3.65	3.16	3.954	99.406	198.75	245.64	293.81	392.86	1880.2
Sai số tuyệt đối trung bình	2265	1.65	0.16	0.046	0.594	1.25	4.365	6.19	7.14	1470.2

Kết luận:

Dựa vào dữ liệu bảng trên ta có thể thấy:

- Trong khoảng từ (0; 2cm) và 400 cm trở lên thì cảm biến đưa ra kết quả đo sai.
- Từ 3cm đến 200cm, cảm biến đưa ra kết quả tương đối chính xác với sai số nhỏ.
- Từ 200cm đến 400cm, cảm biến đưa ra kết quả với sai số khá lớn.



Hình 4.36 Hình ảnh đo vật ở khoảng cách 4cm

Xác định vùng mù của cảm biến siêu âm:

Cảm biến siêu âm HC-SR04 là một cảm biến khoảng cách sử dụng sóng siêu âm để đo khoảng cách từ cảm biến đến vật thể. Tuy nhiên, như mọi cảm biến, nó cũng có một số hạn chế, trong đó có khái niệm vùng mù hay được gọi là "dead zone".

Vùng mù của cảm biến siêu âm HC-SR04 là khoảng không gian gần cảm biến mà nó không thể đo được đúng khoảng cách. Điều này xảy ra do thời gian mà sóng siêu âm mất để đi và trở lại từ vật thể. Cảm biến HC-SR04 sử dụng nguyên lý đo khoảng cách bằng cách gửi một tín hiệu siêu âm và theo dõi thời gian mà sóng siêu âm mất để quay trở lại. Tuy nhiên, trong thời gian này, cảm biến không thể nhận biết được những sóng siêu âm phản xạ từ vật thể nằm trong vùng mù.

Kết luận:

Dựa vào số liệu bảng 2 thì ta có thể thấy vùng mù của cảm biến HC-SR04 chính là ở khoảng cách rất gần cảm biến (dưới 2 cm) và ở khoảng cách xa (trên 4 mét). Trong khoảng cách này, cảm biến không thể đo khoảng cách chính xác và có thể cho kết quả không chính xác

Để tránh vùng mù của cảm biến HC-SR04, bạn có thể xem xét sử dụng các phương pháp như lọc dữ liệu đầu ra, đặt ngưỡng khoảng cách hợp lý hoặc sử dụng các cảm biến khác để bổ sung thông tin.

*** Kiểm tra sự hội tụ của phép đo khi ở cùng một khoảng cách:**

Như ta được biết, máy móc cũng giống như con người và cũng có khoảng thời gian khởi động khi mới bật nguồn, thời gian mà năng suất làm việc kém hiệu quả. Trong một hệ thống máy móc khi bắt đầu vận hành khoảng thời gian không hiệu quả đó người

ta gọi là khoảng thời gian không ổn định, máy móc sẽ hoạt động với những sai số rất lớn đạt độ chính xác không cao. Tuy nhiên, hầu hết các hệ thống sau một khoảng thời gian nhất định thì các sai số sẽ giảm dần và đạt độ ổn định nhất định khi đến ngưỡng thời gian đó. Thời gian đó gọi là thời gian để hệ thống hoạt động ổn định. Vây thời gian ổn định đó được xác định như thế nào. Trong toán học có một khái niệm về độ lệch chuẩn và nó là một đại lượng thống kê dùng để đo mức độ phân tán của một tập dữ liệu.

Độ lệch chuẩn của một tập dữ liệu tính theo công thức:

$$S = \sqrt{\frac{\sum (x - \bar{x})^2}{n-1}} \tag{4.7}$$

Trong đó:

S: Độ lệch chuẩn của mẫu

$\bar{X}$: Trung bình của mẫu

X_i: Thành phần thứ i của mẫu

n: Tổng số thành phần của mẫu

Để minh hoạ cho điều đó, dưới đây là bảng dữ liệu (150 giá trị) về khoảng cách 10 cm khi đo cùng một điều kiện:

Bảng 4.7 Dữ liệu về đo khoảng cách 10cm

9.83	9.83	9.93	10.27	10.37	9.83	9.83	9.83
9.83	10.27	9.83	10.27	9.93	9.93	9.93	9.83
9.93	9.79	9.86	10.27	9.83	9.86	9.93	9.93
9.93	9.89	9.83	10.37	9.83	9.83	9.83	9.93
9.83	9.93	9.93	10.37	9.93	9.83	9.83	9.83
9.83	9.83	10.37	10.27	9.93	9.93	9.86	9.83
9.93	9.83	9.86	10.27	9.83	9.93	9.93	9.93
9.93	9.93	9.86	10.27	9.83	9.83	9.83	9.93
9.79	9.93	10.37	10.27	9.93	9.83	9.83	9.83
9.83	10.27	10.37	10.37	9.93	9.89	9.83	9.76
9.93	9.86	9.96	10.37	9.83	9.93	9.93	
9.93	9.86	9.86	10.27	9.83	9.83	9.83	
9.83	9.96	10.27	10.27	9.83	9.83	9.83	
9.83	9.79	10.37	10.27	9.93	9.93	9.83	
9.83	9.79	9.96	10.37	9.79	9.93	9.93	
9.93	9.79	9.86	9.96	9.83	9.83	9.93	

9.83	9.93	9.86	10.27	9.79	9.83	9.93	
9.83	9.83	9.86	10.27	9.93	9.93	9.83	
9.83	9.83	9.96	10.27	9.83	9.93	9.83	
9.93	9.83	10.27	10.27	9.83	9.83	9.93	

Dữ liệu bảng trên, được em thể hiện qua biểu đồ bên dưới, để có cái nhìn khái quát nhất về dữ liệu.

Hình 4.37 Biểu đồ biểu diễn sự thay đổi của độ lệch chuẩn

Ta có thể thấy được, khoảng 70 giá trị đầu, độ lệch chuẩn của dữ liệu có sự thay đổi rõ rệt, đó cũng chính là khoảng thời gian làm việc không ổn định của cảm biến. Sau đó, cảm biến chuyển sang giai doạn 2. Lúc này các giá trị của độ lệch chuẩn có những sự giao động nhẹ và gần như đi thẳng. Vì sai số giữa các lần đo thu được là rất nhỏ. Đó chính là thời điểm ổn định cần tìm.

*** Xác định sai số của cảm biến:**

Bảng 4.8 Dữ liệu xác định sai số

Số lần đo	5	10	15	20	25	30
Các giá trị đo	14.55	14.75	14.79	15.10	14.93	14.95
	14.93	14.93	15.03	15.03	14.89	14.95
	14.93	14.93	15.03	15.03	14.99	14.99
	14.52	14.62	15.06	15.03	15.03	14.99
	15.03	15.03	14.99	15.03	14.99	14.96
		14.62	15.03	15.03	15.03	15.05
		15.03	15.03	15.03	14.79	14.93
		15.03	14.99	14.62	15.44	15.03
		14.72	15.03	14.79	15.03	14.99
		14.72	15.04	15.03	15.03	14.93
			15.03	15.03	14.99	14.93
			14.59	15.03	15.03	14.93
			14.35	15.03	15.44	14.96

			14.65 15.13	15.03 14.62 15.03 15.03 14.76 14.99 15.03	14.99 15.03 14.89 14.89 14.89 14.69 14.93 14.93 14.89 14.89 14.89 14.93	14.93 15.05 15.10 14.93 14.93 14.96 14.93 14.93 15.01 14.93 14.95 14.89 14.93 15.93 15.05 15.03 15.03
Trung bình	14.792	14.838	14.918	14.965	14.9667	14.97
Sai số trung bình	0.208	0.162	0.082	0.035	0.0332	0.03
Độ lệch chuẩn	0.2384	0.1694	0.2210	0.1430	0.1614	0.1823

Kết luận:

Từ bảng trên ta có thể thấy độ lệch chuẩn của phép đo: nhỏ nhất khi số lần đo là nhiều nhất, và có xu hướng tăng khi số lần đo giảm. Độ tin cậy của phép đo 30 lần khoảng cách 15cm cũng lớn hơn.

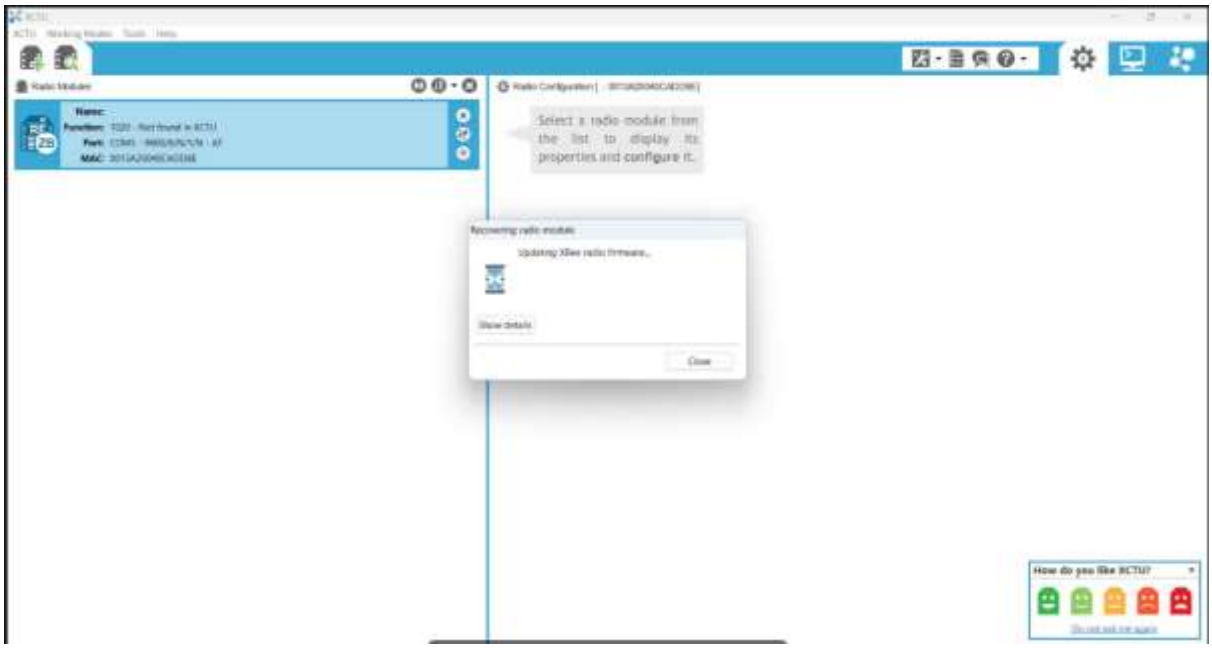
4.4 Thiết kế hệ thống giao tiếp

4.4.1. Phương án lựa chọn hệ thống giao tiếp

Tiêu chí	Lora (Long Range)	Xbee (Zigbee)
Hình ảnh minh hoạ		
Dải tần	433 MHz, 868 MHz, 915 Mhz (tuỳ khu vực)	2.4 GHz hoặc 900 MHz (tuỳ phiên bản)
Phạm vi truyền	Lên đến 3 – 4 km (môi trường lý tưởng)	30-100m (trong nhà), tối đa 1 -2 km (phiên bản tần số thấp)

Tốc độ truyền dữ liệu	Thấp (Khoảng vài Kbps)	Cao hơn (vài chục đến vài trăm Kbps)
Tiêu thụ năng lượng	Rất thấp (phù hợp với IoT, cảm biến xa)	Trung bình (thấp hơn WiFi nhưng cao hơn Lora)
Kiểu mạng	Point-to-Point, Star	Mesh, Point-to-Point, Star
Ứng dụng chính	IoT, giám sát từ xa, cảm biến nông nghiệp, đo lường thông minh	Nhà thông minh, tự động hoá công nghiệp, truyền dữ liệu cảm biến tầm gần
Hỗ trợ giao tiếp	Không (chủ yếu là mạng Star hoặc P2P)	Có (hỗ trợ mesh networking)
Chi phí phần cứng	Thường rẻ hơn XBee	Cao hơn, nhất là module Zigbee Pro

XBee là lựa chọn phù hợp để giao tiếp giữa hai xe vì nó có tốc độ truyền dữ liệu cao hơn LoRa, giúp đảm bảo tín hiệu được gửi gần như theo thời gian thực, phù hợp với các ứng dụng cần phản hồi nhanh như điều khiển xe hoặc trao đổi thông tin vị trí. Ngoài ra, XBee có độ trễ thấp và hỗ trợ cả chế độ Point-to-Point (P2P) lẫn mạng Mesh, giúp linh hoạt hơn nếu muốn mở rộng hệ thống sau này. Với phạm vi hoạt động từ 30-100m (hoặc lên đến 1-2km với phiên bản tần số thấp), XBee đáp ứng tốt nhu cầu giao tiếp giữa hai xe trong môi trường đô thị hoặc khu vực mở. Hơn nữa, việc thiết lập và cấu hình XBee khá đơn giản, có thể giao tiếp qua UART (Serial) với chế độ AT Command hoặc API Mode. Trong khi đó, LoRa có thể truyền xa hơn (trên 10 km) nhưng tốc độ chậm, độ trễ cao và chỉ phù hợp cho các ứng dụng cần gửi dữ liệu không thường xuyên, chẳng hạn như theo dõi vị trí theo khoảng thời gian dài. Vì vậy, nếu cần một giải pháp nhanh, ổn định và dễ triển khai trong phạm vi gần hoặc trung bình, XBee là lựa chọn tối ưu.



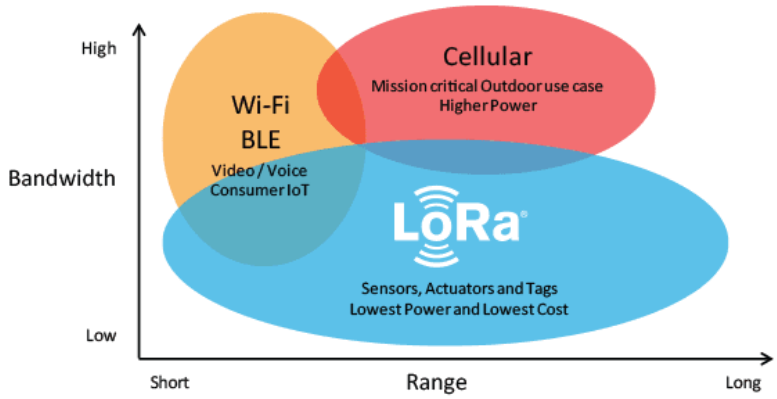
Hình 4.38 Lỗi không tìm thấy XBee

Mặc dù XBee là lựa chọn phù hợp cho giao tiếp giữa hai xe nhờ tốc độ cao và độ trễ thấp, nhưng trong quá trình thực hiện, em gặp khó khăn khi phần mềm XCTU không tìm thấy được XBee và không thể thiết lập các thông số giao tiếp phù hợp. Do đó, em quyết định chuyển sang sử dụng LoRa để tiết kiệm chi phí và tận dụng phạm vi hoạt động xa hơn, giúp đảm bảo kết nối ổn định ngay cả khi khoảng cách giữa hai xe lớn.

4.4.2. Hình thức giao tiếp Lora

a, Tổng quan

LoRa là viết tắt của Long Range Radio là một công nghệ truyền thông dữ liệu không dây sử dụng kỹ thuật điều chế vô tuyến có thể được tạo ra bởi các chip thu phát Semtech LoRa.



Hình 4.39 Tổng quan cấu trúc mạng

Kỹ thuật điều chế này cho phép giao tiếp tầm xa với một lượng nhỏ dữ liệu (có nghĩa là băng thông thấp), khả năng chống nhiễu cao, đồng thời giảm thiểu mức tiêu thụ điện năng. Vì vậy, nó cho phép giao tiếp đường dài với các yêu cầu công suất thấp. Điều này phù hợp với các thiết bị IoT (internet of things) với dung lượng pin hạn chế. Mỗi gateway LoRa có thể xử lý hàng triệu node. Điều đó, cộng với thực tế là các tín hiệu có thể kéo dài khoảng cách đáng kể, có nghĩa là cần ít cơ sở hạ tầng mạng hơn, do đó làm cho việc xây dựng mạng LoRa rẻ hơn. Các mạng LoRa có thể được đặt cùng với các thiết bị liên lạc khác, như các tháp điện thoại di động, làm giảm đáng kể các hạn chế xây dựng.

Như vậy, đặc điểm nổi bật nhất của LoRa đó là khả năng truyền nhận dữ liệu ở khoảng cách rất xa (có thể lên đến 10km tùy thiết kế anten và vật cản) và tiết kiệm pin. Đây là những đặc điểm mà kết nối Wifi hay 3G/4G không có và giá thành của các chipset Lora cũng khá rẻ.

Sự phát triển của Internet of Things bị giới hạn bởi dung lượng của mạng, bởi khả năng hoạt động của thiết bị mà không cần thay pin và bởi khả năng mã hóa truyền dẫn bí mật. Các tính năng được tích hợp trong LoRa cung cấp tất cả các khả năng này và sẽ cho phép sự phát triển rộng rãi của IoT.

Với công nghệ Lora , chúng ta có thể truyền dữ liệu với khoảng cách lên hàng km mà không cần các mạch khuếch đại công suất; từ đó giúp tiết kiệm năng lượng tiêu thụ

khi truyền/nhận dữ liệu. Do đó, LoRa có thể được áp dụng rộng rãi trong các ứng dụng thu thập dữ liệu như sensor network trong đó các sensor node có thể gửi giá trị đo đạc về trung tâm cách xa hàng km và có thể hoạt động với battery trong thời gian dài trước khi cần thay pin.

Hình 4.40 Ứng dụng công nghệ LoRa

b, Nguyên lý hoạt động của LoRa

LoRa sử dụng kỹ thuật điều chế gọi là Chirp Spread Spectrum. Có thể hiểu nôm na nguyên lý này là dữ liệu sẽ được băm bằng các xung cao tần để tạo ra tín hiệu có dãy tần số cao hơn tần số của dữ liệu gốc (cái này gọi là chipped); sau đó tín hiệu cao tần này tiếp tục được mã hoá theo các chuỗi chirp signal (là các tín hiệu hình sin có tần số thay đổi theo thời gian; có 2 loại chirp signal là up-chirp có tần số tăng theo thời gian và down-chirp có tần số giảm theo thời gian; và việc mã hoá theo nguyên tắc bit 1 sẽ sử dụng up-chirp, và bit 0 sẽ sử dụng down-chirp) trước khi truyền ra anten để gửi đi.

Theo Semtech công bố thì nguyên lý này giúp giảm độ phức tạp và độ chính xác cần thiết của mạch nhận để có thể giải mã và điều chế lại dữ liệu; hơn nữa LoRa không cần công suất phát lớn mà vẫn có thể truyền xa vì tín hiệu Lora có thể được nhận ở khoảng cách xa ngay cả độ mạnh tín hiệu thấp hơn cả nhiễu môi trường xung quanh. Nhờ sử dụng chirp signal mà các tín hiệu LoRa với các chirp rate khác nhau có thể hoạt động trong cùng 1 khu vực mà không gây nhiễu cho nhau. Điều này cho phép nhiều thiết bị LoRa có thể trao đổi dữ liệu trên nhiều kênh đồng thời (mỗi kênh cho 1 chirprate). LoRa sử dụng tần số không cần giấy phép có sẵn trên toàn thế giới, Băng tần làm việc của LoRa từ 430 MHz đến 915 MHz cho từng khu vực khác nhau trên thế giới:

- EU là 863 MHz – 870 MHz
- US là 902 MHz – 928 MHz
- AS là 920 MHz – 923 MHz

c, Ứng dụng LoRa trong điều khiển xe thông minh

* Hệ thống điều khiển xe đoàn tự động

Trong mô hình xe đoàn, một xe dẫn đầu (leader) có nhiệm vụ xác định đường đi và truyền thông tin điều hướng đến các xe phía sau. LoRa giúp đảm bảo dữ liệu được truyền đi một cách ổn định mà không bị gián đoạn, ngay cả khi khoảng cách giữa các xe là vài km. Cảm biến khoảng cách siêu âm hoặc radar có thể kết hợp với LoRa để điều chỉnh tốc độ của từng xe, đảm bảo khoảng cách an toàn.

* Hệ thống định vị và giám sát từ xa

LoRa có thể được sử dụng để truyền dữ liệu vị trí GPS từ xe về trung tâm điều khiển, giúp giám sát tình trạng hoạt động của xe theo thời gian thực. Điều này đặc biệt hữu ích trong các ứng dụng xe thông minh như:

- Xe bus thông minh.
- Xe điện công cộng.
- Hệ thống vận chuyển hàng hóa tự động.

* Điều khiển và giám sát cảm biến trong xe

Xe thông minh sử dụng nhiều cảm biến như:

- Cảm biến đo tốc độ.
- Cảm biến khoảng cách.
- Cảm biến phát hiện chướng ngại vật.

Các cảm biến này có thể gửi dữ liệu về bộ điều khiển trung tâm thông qua LoRa, giúp xe tự động điều chỉnh tốc độ, hướng di chuyển và đưa ra các quyết định an toàn.

d, Lợi ích, hạn chế và thách thức của LoRa trong hệ thống điều khiển xe thông minh

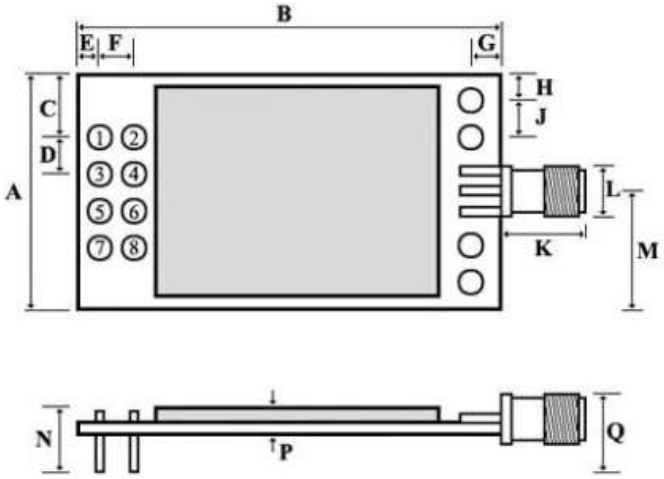
- Tiết kiệm năng lượng: Do đặc điểm tiêu thụ năng lượng thấp, LoRa phù hợp cho các hệ thống xe tự hành hoặc xe IoT cần hoạt động lâu dài.
- Phạm vi phủ sóng rộng: Có thể hoạt động hiệu quả ngay cả trong khu vực đô thị đông đúc hoặc vùng nông thôn xa xôi.
- Chi phí thấp: So với các công nghệ truyền thông khác như 4G/5G, LoRa có chi phí đầu tư và vận hành rẻ hơn.
- Tính ổn định cao: Có khả năng chống nhiễu tốt, hoạt động hiệu quả trong nhiều điều kiện môi trường khác nhau.

Mặc dù có nhiều lợi ích, nhưng LoRa cũng có một số hạn chế như:

- Băng thông thấp: Không phù hợp cho các ứng dụng yêu cầu truyền dữ liệu tốc độ cao như video streaming.
- Độ trễ cao: Không thích hợp cho các ứng dụng điều khiển thời gian thực yêu cầu phản hồi tức thì.
- Phụ thuộc vào môi trường: Khoảng cách truyền có thể bị ảnh hưởng bởi địa hình và vật cản.

4.4.3. Cấu tạo và chế độ vận hành của module LoRa

a, Cấu tạo của module LoRa



	MIN	MAX
A	18.0	18.1
B	33.4	33.5
C	5.14	5.16
D	2.54	2.54
E	1.44	1.46
F	2.54	2.54
G	2.59	2.61
H	1.72	1.74
J	2.54	2.54
K	9.60	9.60
L	6.20	6.20
M	9.00	9.05
N	11.2	11.3
P	3.80	3.90
Q	12.8	12.9

Hình 4.41 Cấu tạo LoRa

4.4.4. Quy trình cấu hình module LoRa

a, Cấu tạo mạch giao tiếp AS15-USB-T2

Hình 4.42 Mạch Giao Tiếp USB UART Lora SX1278 EBYTE E15-USB-T2

Mạch chuyển giao tiếp USB UART Lora SX1278 EBYTE E15-USB-T2 được sử dụng với Mạch thu phát RF UART Lora SX1278 để có thể giao tiếp giữa mạch và máy tính qua cổng USB, giúp cấu hình với Software hoặc truyền nhận dữ liệu trực tiếp với máy tính dễ dàng.

Thiết kế nhỏ gọn bao gồm IC chuyển USB UART CH340X có khả năng nhận Driver trên hầu hết các hệ điều hành, mạch được thiết kế thêm Jumper M0, M1 để thiếp lập các chế độ truyền nhận, cấu hình cho Mạch thu phát RF UART Lora SX1278, mạch còn có Led hiển thị trạng thái truyền nhận.

b, Chế độ vận hành

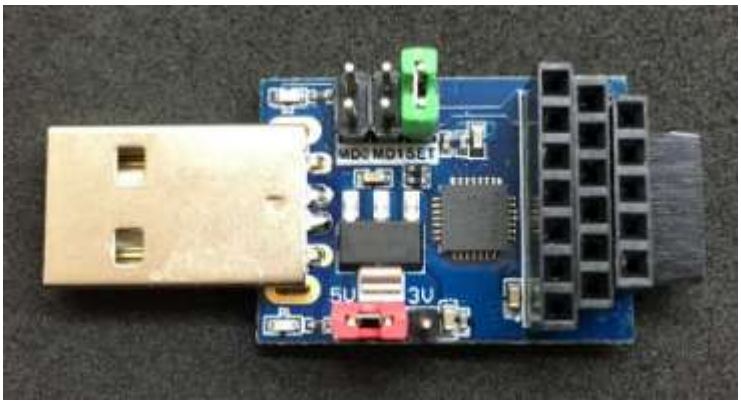
Hình 4.43 Chế độ vận hành

Bảng 4.9 Chi tiết về chế độ vận hành

Tên chế độ	M1	M0	Mô tả	Ghi chú
Mode 0 (normal)	0	0	Cổng nối tiếp mở, kênh truyền không dây mở. Cho phép truyền thông.	Bộ nhận phải làm việc ở mode 0 hoặc mode 1
Mode 1 (wake-up)	0	1	Cổng nối tiếp mở, kênh truyền thông không dây mở. Điều khác biết duy nhất với mode 0 là ở mode 1, dữ liệu tự động được thêm wake-up code. Cho phép đánh thức bộ nhận ở chế độ 2.	Bộ nhận phải làm việc ở mode 0,1 hoặc 2.
Mode 2 (power-saving)	1	0	Cổng nối tiếp đóng, kênh truyền làm việc ở chế độ wake-up. Và cổng nối tiếp sẽ mở khi nhận được tín hiệu từ kênh truyền thông không dây.	Bộ truyền nên hoạt động ở mode 1. Không thể truyền ở chế độ này.
Mode 3 (sleep-mode)	1	1	Chế độ ngủ, có thể thiết lập các thông số cho module	

c, Phần mềm cấu hình LoRa

Bước 1: Kết nối



*Cài jump chốt tại vị trí **"SET"***

Cắm module E32-TTL-1W lên module AS15-USB-T2

Kết nối module AS15-USB-T2 với cổng USB máy tính

Bước 2: Thực hiện cài đặt thông số cho module E32-TTL-1W.

Hình 4.44 Phần mềm RF_Setting_V3.35

- B1: Chọn cổng Com
- B2: OpenPort để mở cổng giao tiếp
- B3: GetPram để đọc dữ liệu của module
- B4: Tùy chỉnh các giá trị phù hợp
- B5: SetParam để lưu giá trị cài đặt vào module

Hình 4.45 Thiết lập thông số khởi tạo LoRa

Trên giao diện chúng ta có thể thiết lập lại các thông số của module như tốc độ truyền, công suất, kênh truyền, địa chỉ,...

Chú ý: Module có 32 kênh truyền nhận khác nhau, ứng với mỗi kênh là một tần số.

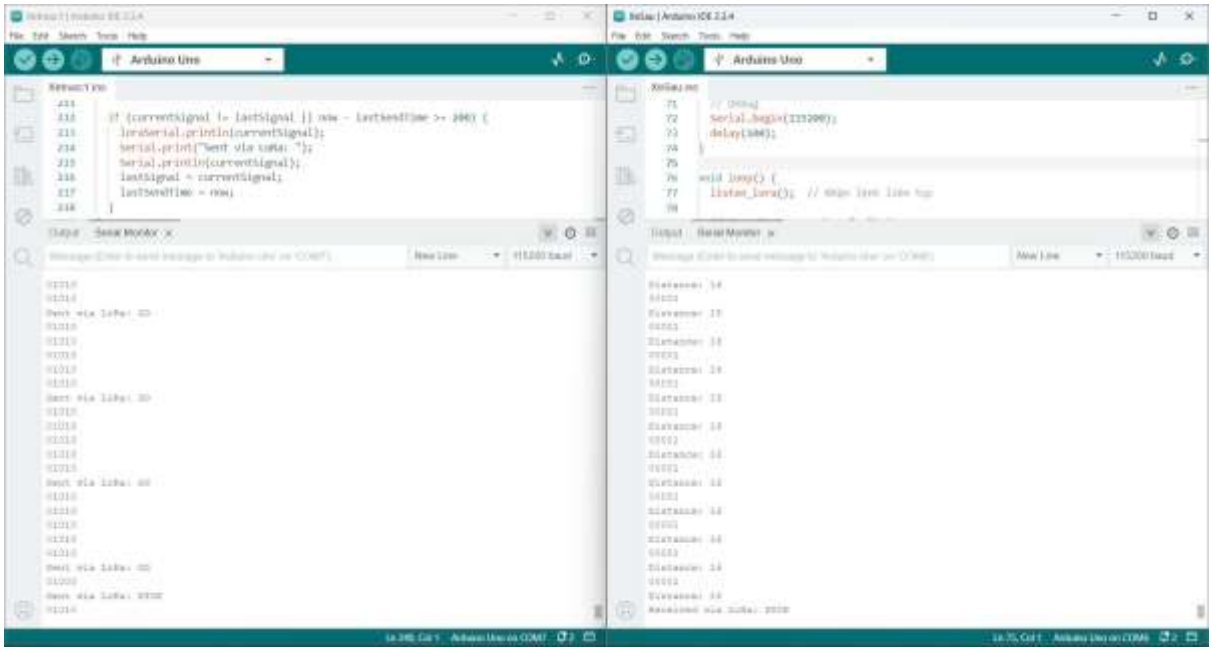
Công thức tính tần số cho mỗi kênh truyền như sau:

Channel: 0-31 Frequency = 410MHz + Channel * 1MHz

Ví dụ ở kênh 23 thì tần số sẽ là: Frequency = 410MHz + 23 * 1MHz = 433MHz

Sau khi thiết lập các thông số, chọn **SetParam → ClosePort**

4.4.5. Hình thức giao tiếp của LoRa trên xe



Hình 4.46 Giao diện kiểm tra giao tiếp giữa hai xe

a, Tổng quan hình thức giao tiếp

Trong các hệ thống xe tự hành hiện đại, đặc biệt là những hệ thống gồm nhiều phương tiện di chuyển theo đội hình (vehicle-to-vehicle – V2V), khả năng giao tiếp giữa các xe đóng vai trò cực kỳ quan trọng. Việc trao đổi thông tin giữa các xe giúp tăng cường độ an toàn, đồng bộ hoá chuyển động và tối ưu hóa khả năng phản ứng với các tình huống bất ngờ trong môi trường thực tế.

Trong đề tài này đã xây dựng thành công một mô hình gồm hai xe robot di chuyển theo vạch (line-following) có khả năng tránh vật cản nhờ cảm biến siêu âm, đồng thời tích hợp mô-đun LoRa để giao tiếp không dây liên tục giữa hai xe. Hệ thống cho phép xe sau phản ứng kịp thời với các tình huống mà xe trước gặp phải, đặc biệt là những tình huống bị che khuất tầm nhìn, chẳng hạn như khi rẽ cua gắt.

b, Mô tả hệ thống

- Xe đi đầu (Leader)

Xe trước được lập trình để tự hành theo vạch bằng cách sử dụng hệ thống cảm biến dò line gồm 5 cảm biến hồng ngoại. Khi di chuyển, xe liên tục quét mặt đất để giữ cho lộ trình luôn bám sát vạch hướng dẫn.

Song song với việc dò line, một cảm biến siêu âm được tích hợp phía trước xe để phát hiện vật cản. Khi xe phát hiện có vật cản nằm trong ngưỡng nguy hiểm (ví dụ: dưới 20 cm), nó sẽ tự động dừng lại và gửi tín hiệu "STOP" qua mô-đun LoRa đến xe sau. Khi vật cản biến mất và khoảng cách an toàn được đảm bảo, xe sẽ tiếp tục hành trình và gửi tín hiệu "GO" để thông báo cho xe sau tiếp tục di chuyển.

Để tránh tình trạng gửi trùng lặp và giảm nhiễu tín hiệu, hệ thống kiểm tra nếu tín hiệu hiện tại khác tín hiệu trước đó và đã đủ thời gian cách quãng tối thiểu (ví dụ 200ms) thì mới thực hiện gửi lệnh mới.

- Xe đi sau (Follower)

Xe sau liên tục lắng nghe tín hiệu LoRa từ xe trước. Khi nhận được lệnh "STOP", xe ngay lập tức dừng lại. Ngược lại, khi nhận lệnh "GO", xe tiếp tục di chuyển theo vạch như bình thường. Nhờ vậy, xe sau có thể phản ứng tức thì trước các tình huống mà cảm biến của nó không trực tiếp quan sát được, chẳng hạn như vật cản xuất hiện ở khúc cua gắt, nơi mà tầm quét siêu âm bị giới hạn.

Việc truyền lệnh liên tục và theo thời gian thực giúp đảm bảo rằng xe sau luôn đồng bộ hoá được trạng thái hoạt động với xe trước – điều cực kỳ quan trọng để duy trì an toàn và hiệu quả khi di chuyển theo nhóm.

c, Kiểm thử và đánh giá

Thông qua quá trình kiểm thử thực tế, hệ thống đã cho thấy khả năng hoạt động ổn định và hiệu quả. Khi xe trước gặp vật cản và dừng lại, tín hiệu "STOP" được gửi đi ngay lập tức, và xe sau cũng dừng lại gần như đồng thời. Khi vật cản được loại bỏ, tín hiệu "GO" được phát đi, giúp xe sau tiếp tục hành trình mà không cần bất kỳ xử lý phức tạp nào từ người điều khiển.

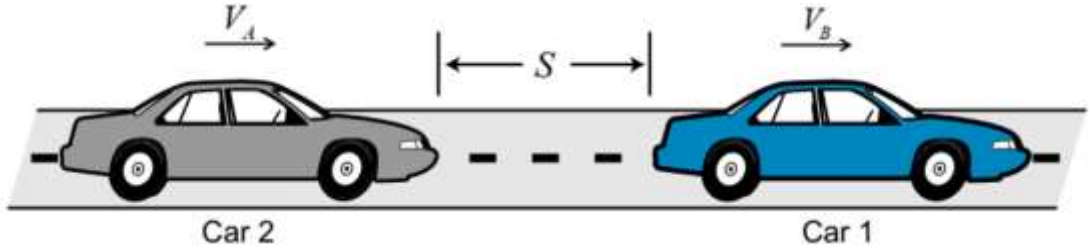
Dữ liệu trên Serial Monitor ghi nhận chi tiết quá trình truyền và nhận lệnh giữa hai xe. Tín hiệu line như "01110", "01010" được in ra thường xuyên cùng với trạng thái "Sent via LoRa: GO" hoặc "STOP". Ở phía nhận, tín hiệu "Received via LoRa: GO" hoặc "STOP" phản ánh chính xác trạng thái tương ứng.

Chương 5: THIẾT KẾ PHẦN LẬP TRÌNH VÀ THUẬT TOÁN HỆ THỐNG ĐIỀU KHIỂN KẾT HỢP MÔ PHỎNG QUÁ TRÌNH DI CHUYỂN CỦA XE

5.1. Thiết kế phần lập trình

5.1.1. Lý thuyết cơ sở hệ thống điều khiển

a, Xử lý tín hiệu trên đường thẳng



Hình 5.1 Sơ đồ bố trí hai xe đang di chuyển trên đường

Giả sử hai xe đang chuyển động với tốc độ V_A, V_B. Khoảng cách giữa 2 xe là S(m). Tại thời điểm t_1: tín hiệu về S_1, V_1. Tại thời điểm t_2: tín hiệu về S_2, V_2. Với S_1, S_2 là khoảng cách giữa 2 xe đo ở 2 thời điểm khác nhau và V_1, V_2 là vận tốc của xe ta đang chạy ứng với 2 thời điểm đo liên tiếp nhau. Và t_1, t_2: là thời điểm đo liên tiếp giữa 2 lần nhận tín hiệu.

* Tính vận tốc tương đối giữa 2 xe [6]:

$$\Delta V = \frac{S_2 - S_1}{\Delta t} = \frac{S_2 - S_1}{t_2 - t_1}(m/s) \qquad (5.1)$$

Δt : là biến thiên thời gian giữa 2 lần nhận tín hiệu liên tiếp, phụ thuộc vào thời gian đưa tín hiệu về của cảm biến khoảng cách.

* Xét điều kiện:

S_{min} (m): là khoảng cách nhỏ nhất ứng với vận tốc chuyển động của xe V (m/s) mà lúc đó cần phải điều khiển giảm tốc độ hoặc dừng để đảm bảo điều kiện chuyển động an toàn của xe.

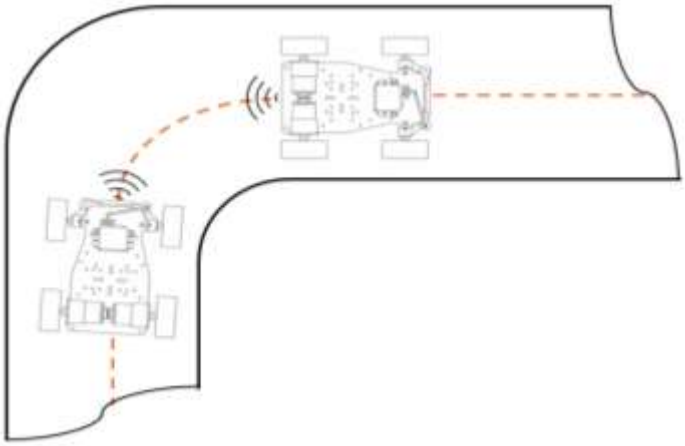
- Nếu $\Delta V < 0$ (tức là $S_2 < S_1$) thì ta xét:
  + Nếu $S_2 < S_{min}$ thì cần điều khiển motor để giảm tốc độ.
  + Nếu $S_2 > S_{min}$ để xe duy trì với tốc độ đặt ổn định.
- Nếu $\Delta V > 0$ (tức là $S_2 > S_1$) thì xe sẽ duy với tốc độ đặt ổn định.

b, Xử lý tín hiệu nhập làn vào vòng cua

Trong bài báo cáo này, hình thức xây dựng một mô hình điều khiển phương tiện trên đường liên tục, tập trung vào việc xác định khoảng cách với xe phía trước khi vào vòng cua. Mục tiêu là đảm bảo xe phía sau có thể duy trì quỹ đạo cung tròn một cách mượt mà, đồng thời quyết định khi nào cần tăng tốc hoặc giảm tốc để thích ứng với điều kiện vào cua. Nếu khoảng cách giữa hai xe đủ lớn, xe phía sau có thể chủ động vào cua mà không ảnh hưởng đến dòng giao thông. Tuy nhiên, khi khoảng cách không đảm bảo, các phương tiện buộc phải giảm tốc độ hoặc thậm chí dừng lại, gây gián đoạn luồng di

chuyển. Ngược lại, nếu xe phía sau ưu tiên giữ khoảng cách hơn là cân nhắc đến mức độ an toàn khi vào cua, nguy cơ va chạm trên làn đường vào cua sẽ gia tăng.

Để giải quyết vấn đề này, bài báo đề xuất một phương pháp điều khiển giao tiếp tự động giữa hai phương tiện, dựa trên hệ thống đường được thiết lập. Phương pháp này tính toán và cung cấp tốc độ tham chiếu phù hợp, giúp điều chỉnh quỹ đạo của xe khi vào cua một cách an toàn và hiệu quả.



Hình 5.2 Sơ đồ thời điểm hai xe đang di chuyển vào đường quỹ đạo cong

Trong mô hình thực tế, chúng đề xuất sử dụng RFID hoặc Hệ thống định vị toàn cầu (GPS) để phát hiện vị trí của phương tiện giao thông cá nhân. Xe di chuyển trên đường sẽ nhận được tín hiệu này để xử lý và sau đó truyền tín hiệu đến xe khác [7].

Từ công thức 5.2, ta suy ra công thức 5.3. Tiếp tục tích hợp công thức 5.3, ta thu được công thức 5.4. Cuối cùng, công thức 5.6 được thiết lập, cho thấy rằng với thông tin về tốc độ ban đầu an toàn, vị trí tức thời của xe, khoảng cách an toàn và gia tốc, ta có thể dễ dàng tính toán vận tốc cuối cùng mà xe cần đạt được.

$$a = \frac{v_{safe}^{truoc} - v_{safe}^{sau}}{t_{truoc} - t_{sau}} \tag{5.2}$$

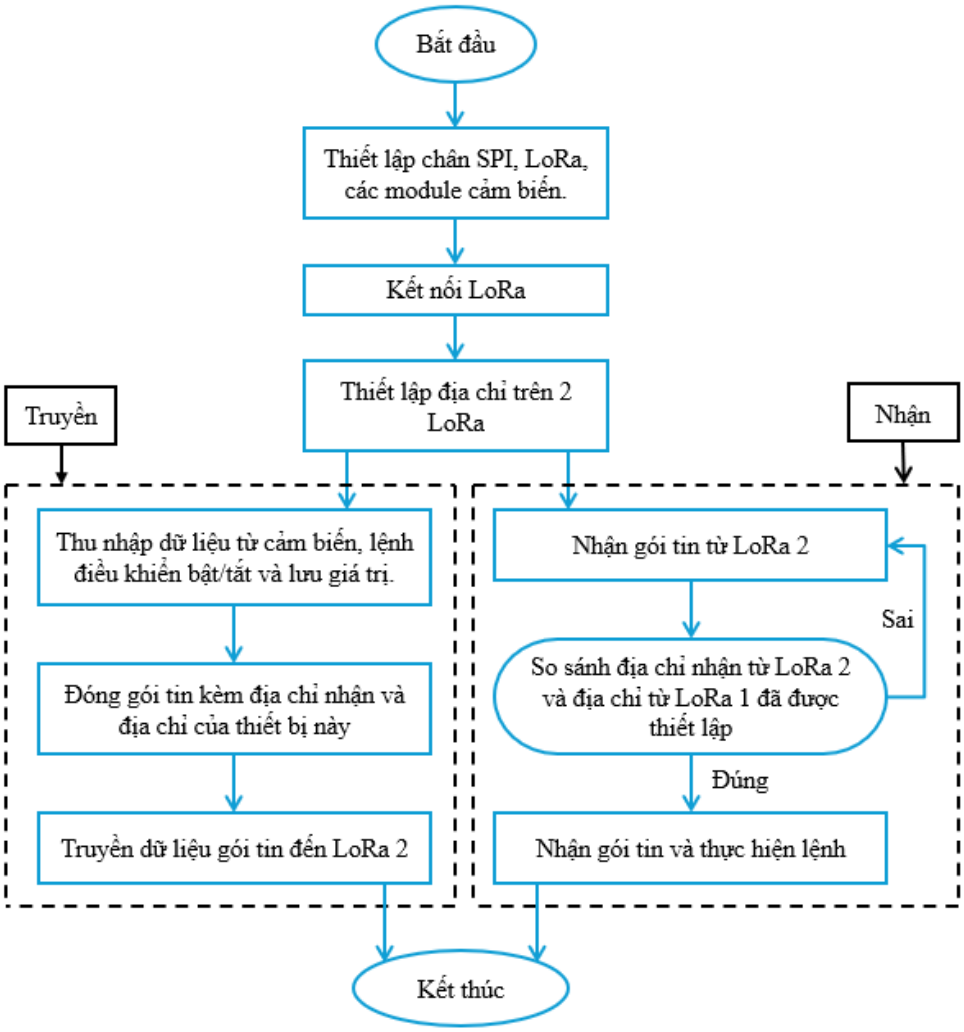
$$v_{safe}^{truoc} = v_{safe}^{sau} + a.\Delta t \tag{5.3}$$

$$s_{safe} = \int v_{safe}^{truoc}.\Delta t = v_{safe}^{sau}.\Delta t + \frac{1}{2}.a.\Delta t^2 \tag{5.4}$$

$$s_{safe} = \frac{(v_{safe}^{truoc})^2 - (v_{safe}^{sau})^2}{2a} \tag{5.5}$$

$$v_{safe}^{truoc} = \sqrt{2a.s_{safe} + (v_{safe}^{sau})^2} \tag{5.6}$$

c, Nguyên lí tín hiệu truyền và nhận giữa hai xe



Hình 5.3 Sơ đồ truyền – nhận tín hiệu giữa hai xe

Khi xe bắt đầu di chuyển, hệ thống LoRa được kích hoạt để chia sẻ dữ liệu giữa hai xe nhằm đảm bảo duy trì tốc độ và khoảng cách an toàn. Quá trình này bắt đầu bằng việc thiết lập kết nối LoRa và cài đặt địa chỉ cho từng xe để xác định thông tin gửi – nhận.

Xe phía trước thu thập dữ liệu từ các cảm biến, bao gồm khoảng cách đo được và lệnh điều khiển, sau đó đóng gói dữ liệu cùng với địa chỉ nhận và gửi đến xe phía sau thông qua LoRa. Khi xe phía sau nhận được gói tin, hệ thống sẽ kiểm tra địa chỉ để xác minh tính hợp lệ. Nếu địa chỉ khớp với thiết lập ban đầu, xe sẽ thực thi lệnh, điều chỉnh tốc độ hoặc dừng lại khi cần thiết. Nếu địa chỉ không khớp, gói tin sẽ bị bỏ qua để tránh nhầm lẫn. Cơ chế này giúp hai xe phối hợp nhịp nhàng, duy trì khoảng cách hợp lý và tránh va chạm. Nhờ đó, hệ thống đảm bảo sự an toàn và ổn định trong quá trình di chuyển, đặc biệt khi vào các đoạn đường cong hoặc điều kiện giao thông phức tạp.

d, Nguyên lí điều khiển theo đường line

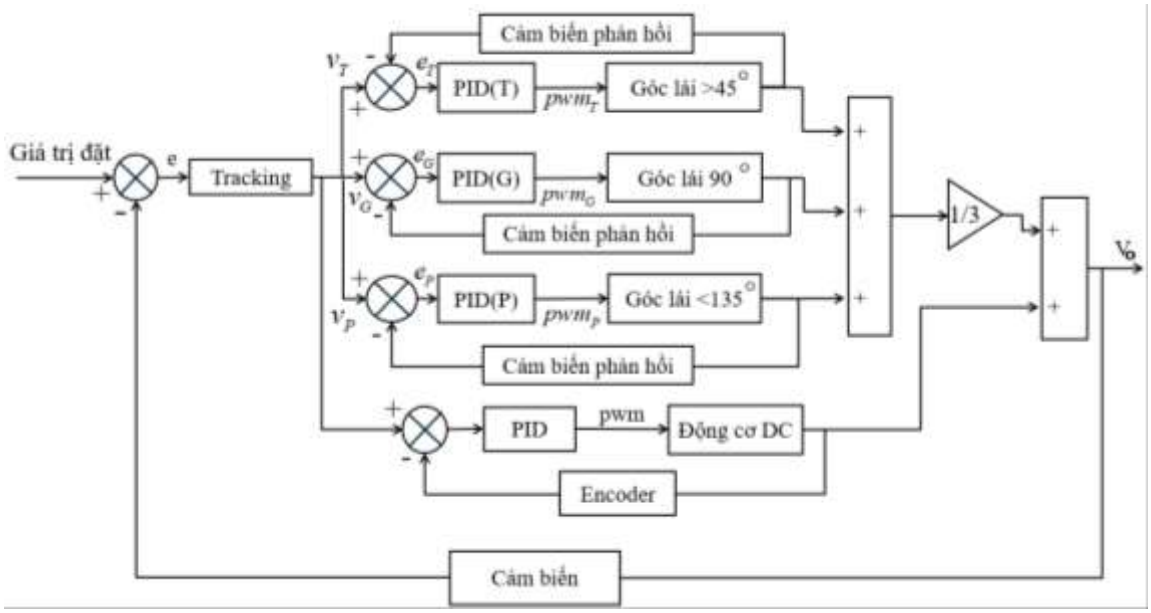
Do cấu trúc đơn giản và dễ triển khai nên bộ điều khiển PID được sử dụng rộng rãi trong lĩnh vực kỹ thuật điều khiển [9]. Cấu trúc chung của bộ điều khiển PID là

$$u_{PID}(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} \qquad (5.7)$$

Với: $u_{PID}(t)$ là tín hiệu điều khiển.

$e(t)$ = Giá trị đặt – Giá trị thực tế.

K_p, K_i, K_d là các hệ số điều chỉnh.



Hình 5.4 Sơ đồ bộ điều khiển PID

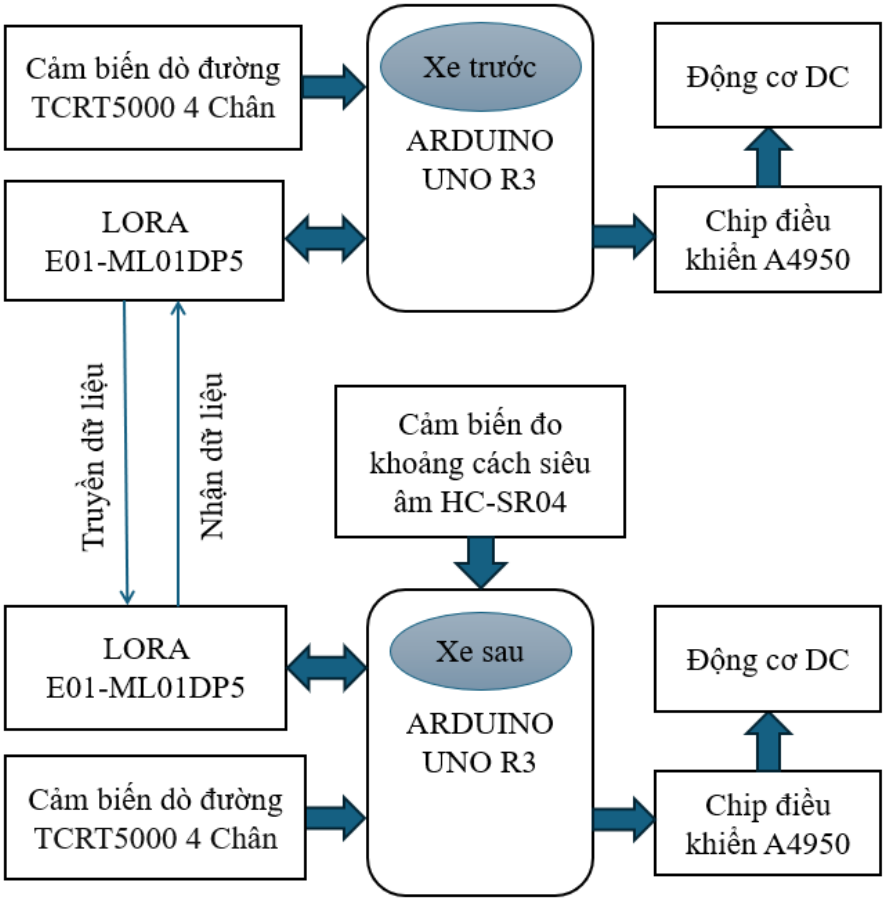
Phạm vi của các biến chưa biết tức là k_p, k_i và k_d đã được quyết định sau một số lần chạy thử và được cung cấp trong Bảng 1 nhằm tối ưu hóa thông qua u_{PID} để tìm giá trị tham số bộ điều khiển PID.

Khoảng cách xe trước không được xử lý trực tiếp nhưng có thể ảnh hưởng gián tiếp qua sai số vận tốc. Sử dụng thuật toán PID điều khiển tốc độ dựa trên sai số giữa tốc độ mong muốn và thực tế, đồng thời có thể được thiết lập để kiểm soát tốc độ phù hợp khi di chuyển.

Bảng 5.1 Thông số thiết lập xe

Thông số	k_p	k_i	k_d
Phạm vi thực nghiệm	12	0,0002	8
Vận tốc xe	1,08 m/s		
Khoảng cách với xe trước	> 30 cm		
Tần số giao tiếp	433MHz		
Phạm vi giao tiếp	~ 3 km		

5.1.2. Sơ đồ khối hệ thống điều khiển



Hình 5.5 Sơ đồ khối hệ thống điều khiển

Tại thời điểm xuất phát, đặt hai xe trên cùng một đường thẳng, nằm trong quỹ đạo của một cung tròn. Ban đầu, nhờ tích hợp cảm biến dò đường và đo khoảng cách, cả hai xe duy trì tốc độ ổn định thông qua thuật toán PID, được lập trình bằng ngôn ngữ C và nạp vào mô-đun Arduino Uno R3. Sau khi di chuyển trên đoạn đường thẳng, xe bắt đầu vào quỹ đạo cung tròn, kích hoạt hệ thống tạo mạng lưới để chia sẻ dữ liệu. Hệ thống này sử dụng công nghệ LoRa, kết nối với Arduino để theo dõi chuyển động của xe. Nếu bất kỳ xe nào tiến gần một xe khác, Arduino sẽ truyền cảnh báo đến phương tiện lân cận. Quá trình phát hiện được thực hiện nhờ cảm biến siêu âm, giúp đo khoảng cách và xác định vị trí xe xung quanh. Cảm biến này thu thập thông tin từ máy thu, truyền qua máy phát và gửi dữ liệu đến LoRa.

LoRa, với khả năng liên lạc tầm xa, giúp xác định các phương tiện trong phạm vi của nó. Khi một xe tiếp cận xe khác ở khoảng cách ngắn, hệ thống sẽ gửi cảnh báo đến Arduino, kích hoạt thông báo bằng văn bản trên màn hình OLED được cài đặt trên xe. Hơn nữa, mỗi xe không chỉ nhận dữ liệu từ chính nó mà còn từ các xe khác trong mạng lưới, bao gồm thông tin về vị trí và vận tốc của phương tiện xung quanh. Dựa trên dữ liệu này, xe phía sau có thể tính toán thời điểm vào cua và điều chỉnh tốc độ phù hợp để đảm bảo hành trình an toàn và hiệu quả.

5.1.3. Lưu đồ giải thuật

Diễn giải quy trình tổng quát với sơ đồ trên được chia thành hai phần: phần đầu là bảng mô tả bốn bước chính khi các xe tham gia vào khu vực có vòng cua, và phần thứ hai là lưu đồ thuật toán cho toàn bộ quá trình nhận diện, phân tích và ra quyết định điều khiển hành trình.

Bước 1: Tiếp cận và phát tín hiệu

Xe đầu tiên di chuyển trên đoạn đường thẳng và duy trì khoảng cách ổn định. Khi nó tiến vào vùng có vòng cua, xe này bắt đầu phát tín hiệu – một dấu hiệu cho thấy nó đã đi vào khu vực cần kiểm soát tốc độ và hướng lái chặt chẽ hơn. Đây là thời điểm khởi đầu của chuỗi phối hợp giữa các xe.

Bước 2: Các xe khác cùng vào cua

Xe khác khi tiếp cận khu vực nhập làn sẽ bắt đầu quá trình di chuyển vào vòng cua. Trong giai đoạn này, việc chia sẻ dữ liệu vị trí và tốc độ là cực kỳ quan trọng để đảm bảo các xe duy trì hành trình một cách an toàn.

Bước 3: Tự động kết nối và chia sẻ thông tin

Khi xe tiếp cận phạm vi chuyển động theo quỹ đạo cong, chúng sẽ được kết nối với nhau thông qua mạng Lora và chia sẻ thông tin với nhau. Mỗi xe có khả năng truyền và nhận dữ liệu như vị trí (ký hiệu là px) và tốc độ từ các xe khác. Đây là bước tạo nền tảng cho việc điều khiển đồng bộ giữa nhiều phương tiện.

Bước 4: Ra quyết định điều khiển

Từ ba biến số chính – tốc độ xe, khoảng cách từ xe nhập làn đến các xe khác, và vị trí tương đối – xe phía sau sẽ điều chỉnh tốc độ để duy trì hành trình theo xe phía trước. Một thuật toán điều khiển tự động (PID hoặc tương đương) sẽ được kích hoạt để xử lý dữ liệu nhận được và tính toán hành vi điều khiển phù hợp.

Bảng 5.2 Quy trình giải thuật điều khiển

Bước 1	Khi xe đầu tiên di chuyển trên đường thẳng và kiểm soát duy trì khoảng cách ổn định, nó sẽ tiếp tục dần dần di chuyển phạm vi vào vòng cua của đường. Sau đó, nó sẽ gửi lệnh phát sóng trong vùng đang có vòng cua.
Bước 2	Xe khác đang di chuyển đến khu vực nhập làn trên đường để bắt đầu di chuyển vào vòng cua.
Bước 3	Tự động kết nối mạng và chia sẽ thông tin.
Bước 4	Dựa trên 3 biến số (tốc độ xe, khoảng cách từ xe nhập làn vào cua, vị trí của xe) xe phía sau vẫn duy trì tốc độ so với xe phía trước dựa trên dữ liệu vị trí.

Lưu đồ trong hình 5.2 thể hiện chi tiết quá trình điều khiển hành trình theo luồng xử lý tín hiệu:

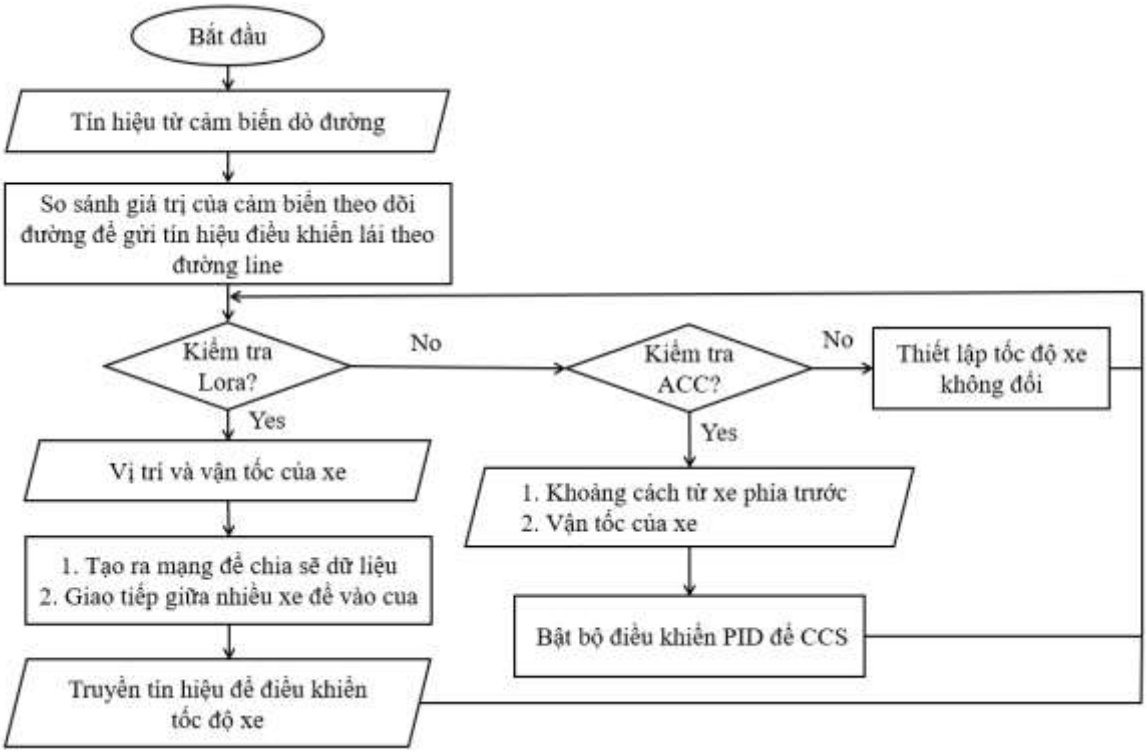
- Hệ thống bắt đầu từ việc nhận tín hiệu từ cảm biến dò đường.

- Sau đó, các cảm biến theo dõi đường sẽ đánh giá và gửi tín hiệu điều khiển để xe bám theo làn đường định sẵn.

- Tiếp theo, hệ thống kiểm tra kết nối Lora. Nếu có, xe sẽ nhận được thông tin vị trí và vận tốc của các xe khác, tạo ra mảng dữ liệu chia sẻ để hỗ trợ điều hướng vào cua.

- Nếu không có Lora, hệ thống sẽ kiểm tra xem ACC có được kích hoạt không. Nếu có, xe sẽ sử dụng khoảng cách đến xe phía trước và vận tốc hiện tại để bật bộ điều khiển PID cho hệ thống kiểm soát hành trình (CCS).

- Nếu cả hai điều kiện trên đều không được thỏa mãn, hệ thống sẽ duy trì tốc độ không đổi – đây là phương án dự phòng đơn giản nhất.



Hình 5.6 Lưu đồ giải thuật điều khiển

- Vai trò của thuật toán điều khiển tự động

Khi thông tin từ các xe khác được chia sẻ qua Lora, bộ điều khiển sẽ xử lý thông tin dựa trên thuật toán điều khiển tự động. Mỗi xe lưu trữ dữ liệu vị trí px của chính mình và của các xe khác. Trong quá trình xử lý, xe sẽ tìm giá trị nhỏ nhất nhưng lớn hơn px (nghĩa là xe liền phía trước trên cùng làn), và định danh đó là py. Dựa vào py, xe sẽ xác định được mục tiêu theo dõi để thực hiện điều khiển hành trình thích ứng, đảm bảo không va chạm và giữ khoảng cách an toàn.

- Thuật toán cụ thể như sau:

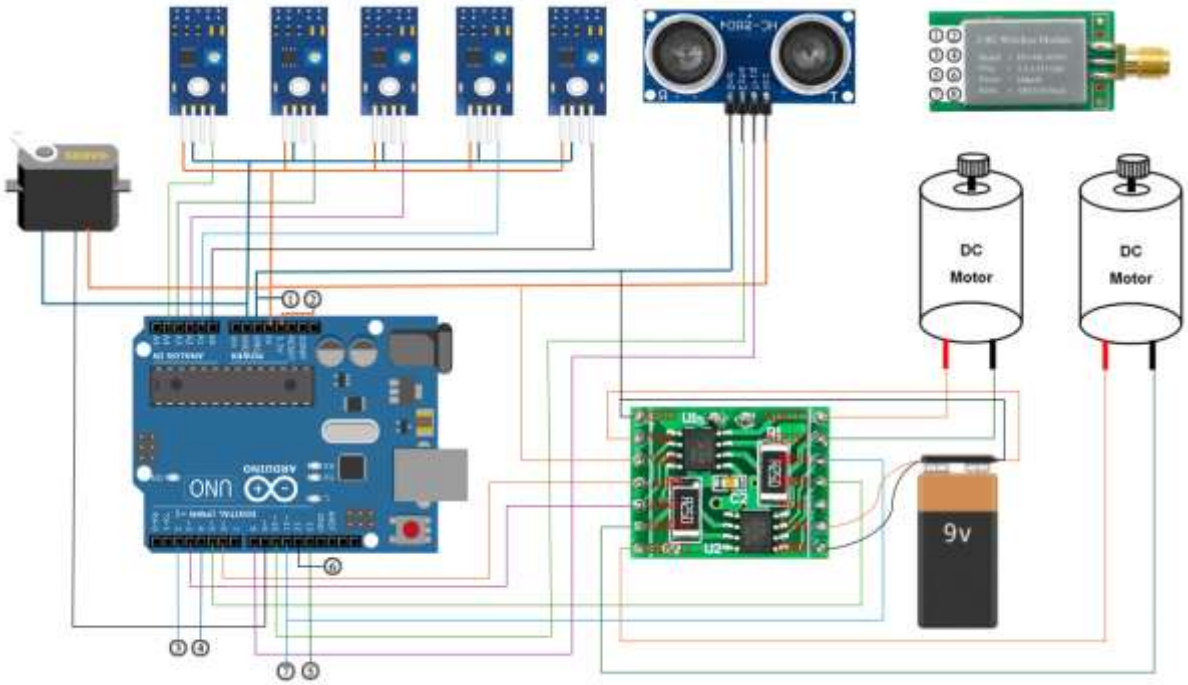
1. Lấy dữ liệu px của xe hiện tại và các px của các xe khác.
2. Lọc các giá trị px lớn hơn px hiện tại.
3. Tìm giá trị nhỏ nhất trong tập hợp này – chính là py.

4. Sử dụng py để tính khoảng cách dọc, từ đó áp dụng điều khiển PID nhằm điều chỉnh tốc độ xe sao cho phù hợp.

5.2. Thiết kế các hệ thống kết hợp và thử nghiệm mã hoá thuật toán ứng dụng trên mô hình xe

5.2.1. Tổng quan thiết kế

Trong hệ thống điều khiển xe tự hành, việc xây dựng mạch điện là một trong những bước quan trọng để đảm bảo các thành phần phần cứng phối hợp hoạt động hiệu quả. Mô hình sử dụng bộ điều khiển trung tâm là Arduino Uno, kết nối với các cảm biến dò line, cảm biến siêu âm, module giao tiếp không dây LoRa, servo điều hướng, động cơ DC và mạch điều khiển động cơ A4950.



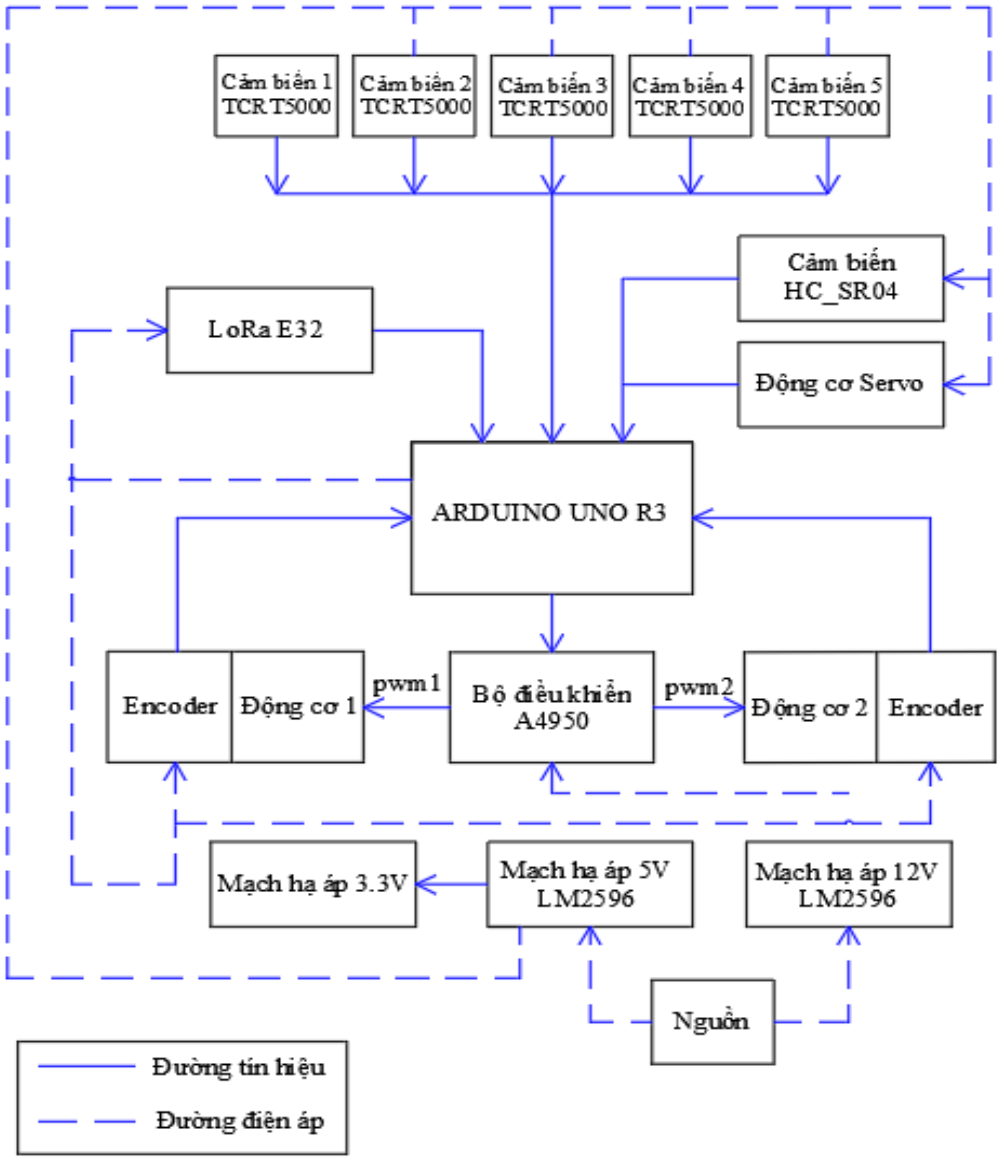
Hình 5.7 Thiết kế hình hoạ hệ thống tích hợp

5.2.2. Cơ sở lý thuyết nối mạch

Hệ thống được thiết kế dựa trên các nguyên lý điều khiển tự động và cảm biến. Xe tự hành thực hiện việc dò đường thông qua 5 cảm biến dò line hồng ngoại (IR) gắn phía trước xe. Các cảm biến này giúp phát hiện vạch màu đen-trắng để điều chỉnh hướng đi phù hợp. Đồng thời, cảm biến siêu âm HC-SR04 được lắp ở phía trước để phát hiện và tránh vật cản bằng cách đo khoảng cách từ xe đến chướng ngại vật. Module LoRa SX1278 cho phép giao tiếp không dây giữa các xe hoặc với trạm điều khiển trung tâm để truyền nhận dữ liệu thời gian thực. Việc điều khiển hướng di chuyển được hỗ trợ bởi servo motor, trong khi hai động cơ DC chịu trách nhiệm cho việc di chuyển tiến/lùi.

Mạch điều khiển động cơ A4950 đóng vai trò trung gian giữa vi điều khiển và động cơ, cung cấp dòng điện phù hợp và điều chỉnh tốc độ, chiều quay thông qua tín hiệu PWM.

5.2.3. Sơ đồ kết nối và nguyên lý hoạt động



Hình 5.8 Sơ đồ khối hệ thống tích hợp

Dựa trên sơ đồ mạch điện như hình trên:

- **Arduino Uno** là trung tâm điều khiển, nhận tín hiệu từ các cảm biến và điều khiển động cơ thông qua mạch A4950.

- **5 cảm biến dò line IR** được kết nối với các chân digital để phát hiện vạch kẻ đường. Mỗi cảm biến đưa ra tín hiệu HIGH hoặc LOW, tùy vào nền đen hoặc trắng, từ đó xác định vị trí xe so với vạch.

- **Cảm biến siêu âm HC-SR04** sử dụng 2 chân Trigger và Echo kết nối với Arduino để đo khoảng cách bằng xung siêu âm.

- **Module LoRa** kết nối với cổng TX/RX của Arduino, cho phép giao tiếp không dây. Trong dự án, nó có thể truyền dữ liệu trạng thái, khoảng cách, hoặc tín hiệu điều khiển từ xa.

- **Servo motor** điều khiển hướng đầu xe, được kết nối với một chân PWM để thay đổi góc quay.

- **Hai động cơ DC** nối với mạch A4950. Mạch này nhận tín hiệu điều khiển từ Arduino (chân IN1–IN4) và nguồn nuôi riêng từ pin 9V.

- **Nguồn cấp** cho toàn hệ thống bao gồm nguồn từ cổng USB hoặc pin 9V cho động cơ.

5.2.4. Quy trình vận hành

1. Khởi động hệ thống, Arduino khởi tạo và kiểm tra các thiết bị ngoại vi.

2. Các cảm biến IR liên tục đọc giá trị bề mặt bên dưới xe. Dữ liệu từ cảm biến sẽ được Arduino xử lý để xác định hướng cần điều chỉnh.

3. Nếu phát hiện vật cản phía trước, cảm biến siêu âm sẽ gửi tín hiệu về Arduino. Vi điều khiển sẽ xử lý và ra lệnh dừng hoặc đổi hướng tùy theo khoảng cách đo được.

4. Dữ liệu vị trí hoặc trạng thái xe được truyền đi qua module LoRa để giao tiếp với các xe khác hoặc với trung tâm điều khiển.

5. Dựa vào thuật toán điều khiển, Arduino gửi lệnh tới mạch điều khiển động cơ A4950 để điều chỉnh tốc độ và hướng quay của động cơ nhằm điều khiển chuyển động của xe.

5.3. Mô phỏng chuyển động của mô hình xe

5.3.1. Giới thiệu công cụ mô phỏng

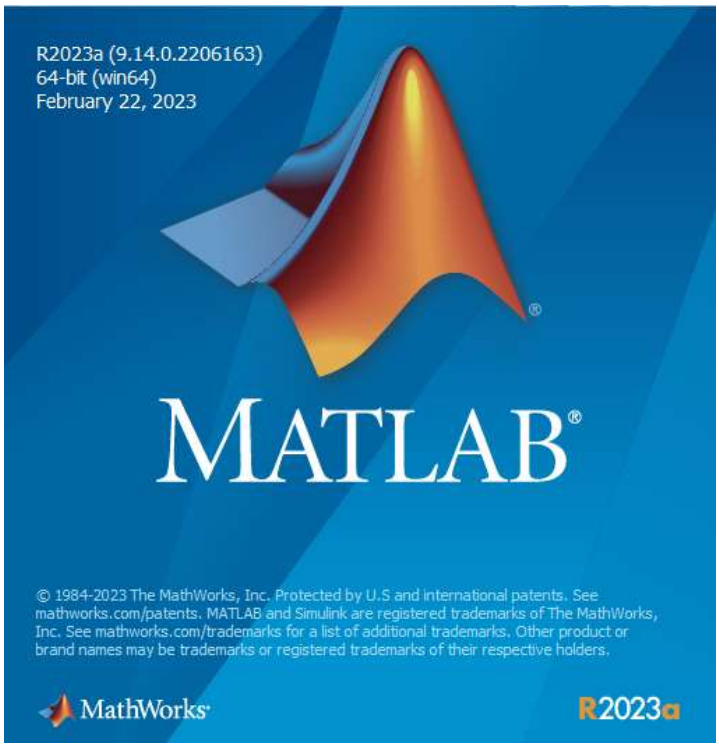
Matlab Simulink

Chương trình Matlab là một chương trình viết cho máy tính PC nhằm hỗ trợ cho các tính toán khoa học và kỹ thuật với các phần tử cơ bản là ma trận trên máy tính cá nhân do công ty "The MATHWORKS" viết ra. Thuật ngữ Matlab có được do hai từ MATRIX và LABORATORY ghép lại. Chương trình này hiện đang được sử dụng nhiều trong nghiên cứu các vấn đề tính toán của các bài kỹ thuật như: Lý thuyết điều khiển tự động, kỹ thuật thống kê sắc suất, xử lý tín hiệu, .v.v..

Chương trình Matlab có thể chạy liên kết với các chương trình ngôn ngữ cấp cao như C++,C,v.v.. Việc cài đặt Matlab thật dễ dàng và ta cần chú ý việc dùng thêm các thư viện trợ giúp hay muốn liên kết phần mềm này với một vài ngôn ngữ cấp cao.

TOOL BOX SIMULINK là phần mở rộng của Matlab, sử dụng mô phỏng các hệ thống động học một cách nhanh chóng và tiện lợi. Thông thường dùng để thiết kế hệ thống điều khiển, thiết kế DSP, hệ thống thông tin và các ứng dụng mô phỏng khác.

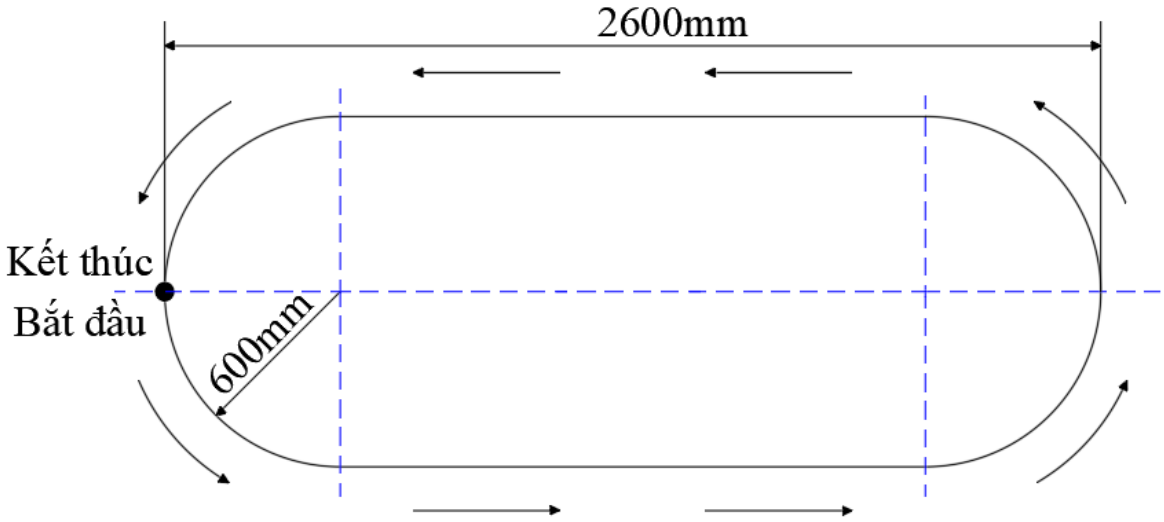
Simulink là thuật ngữ mô phỏng dễ nhớ được ghép bởi hai từ Simulation và Link. Simulink cho phép mô tả hệ thống tuyến tính, hệ phi tuyến, các mô hình trong miền thời gian liên tục, hay gián đoạn hoặc một hệ cả liên tục hay gián đoạn.



Hình 5.9 Phần mềm MATLAB

5.3.2. Mô phỏng mô hình xe tích hợp hệ thống dò line

Thiết kế sa bàn cho mô hình xe di chuyển bám line



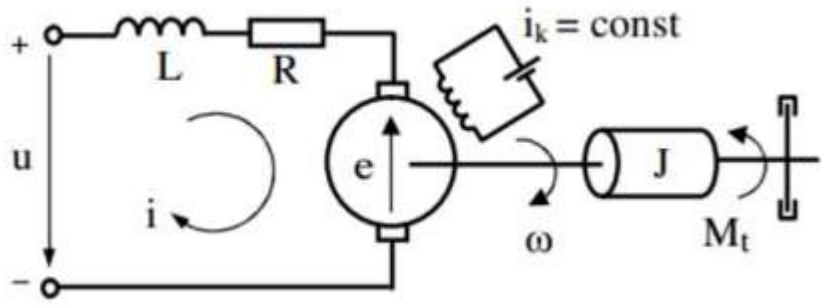
Hình 5.10 Biên dạng hoạt động của robot

- *Mô hình hoá hệ thống robot dò line*

Mô hình hoá động cơ DC:

Cấu tạo của động cơ DC được mô ta như hình dưới:

Động cơ điện một chiều (DC) kích từ độc lập, được điều khiển bằng điện áp phần ứng. Sơ đồ nguyên lý của loại động cơ này được thể hiện trên hình, trong đó dòng kích từ i_k được giữ không đổi.



Hình 5.11 Sơ đồ nguyên lý động cơ điện DC

Với: + Tín hiệu vào là điện áp u đặt vào phần ứng [Volt;V].

+ Tính hiệu ra là vận tốc góc ω của động cơ [rad/s].

Sử dụng 3 phương trình cơ bản như sau:

* Phương trình mạch điện phần ứng

$$U = L\frac{di}{dt} + Ri + K_e\omega \tag{5.1}$$

Trong đó: + R : Điện trở phần ứng, [Ω].

+ L : Điện cảm phần ứng, [H].

+ I : Dòng điện phần ứng, [A].

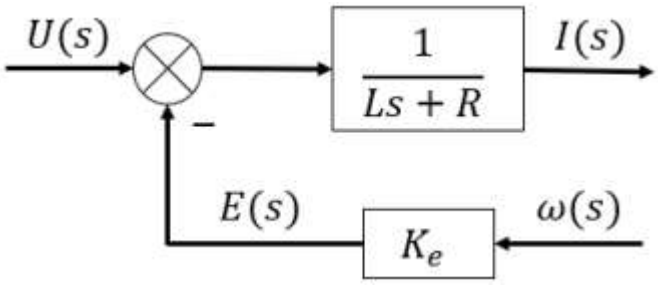
+ K_e : Hằng số sức điện động [Vs/rad].

+ $K_e\omega$: Sức điện động phần ứng [V].

Biến đổi Laplace 2 vế phương trình, ta được:

$$U(s) = LsI(s) + RI(s) + K_e\omega(s) \tag{5.2}$$

Sơ đồ khối tương ứng:



Hình 5.12 Sơ đồ khối mạch điện phần ứng

- *Phương trình momen điện từ của động cơ*

Với dòng kích từ i_k không đổi thì từ thông khe khí $\Phi = k_2 i_k$ là không đổi và momen điện từ M của động cơ tỉ lệ với dòng điện phần ứng:

$$M = K_m i \tag{5.3}$$

Trong đó: $K_m = k_1\Phi = k_1 k_2 i_k$ là hằng số momen của động cơ, [N.m/A]

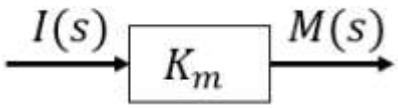
Với: + k_1 : Hằng số phụ thuộc vào kết cấu động cơ.

+ k_2 : Hằng số đặc trung đoạn tuyến tính của từ thông thay đổi theo i_k

Biến đổi Laplace hai vế ta được:

$$M(s) = K_m I(s) \tag{5.4}$$

Sơ đồ tương đương:



Hình 5.13 Sơ đồ khối phần momen điện tử

- *Phương trình cân bằng momen trên trục động cơ*

$$M = J\frac{d\omega}{dt} + B\omega + M_t \tag{5.5}$$

Trong đó:

+ J : Momen quán tính của động cơ và tải quy về trục động cơ, [kg.m^2]

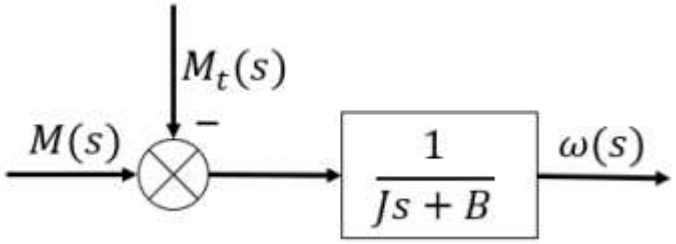
+ B : Hệ số ma sát nhớt của động cơ và tải quy về trục động cơ, [kg.m^2]

+ M_t : Momen phụ tải (nhiễu), [Nm]

Biến đổi Laplace 2 vế phương trình:

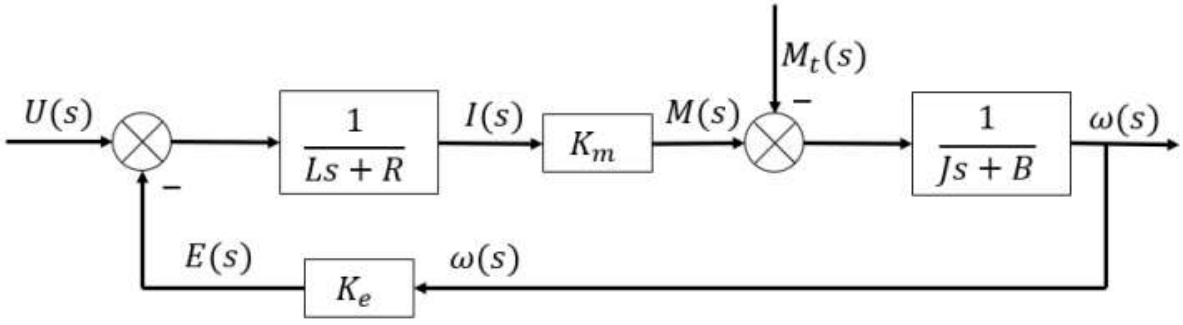
$$M(s) = Js\omega(s) + B\omega(s) + M_t(s)$$

Sơ đồ khối tương ứng:



Hình 5.14 Sơ đồ khối phần cân bằng momen trên động cơ

Kết hợp 3 sơ đồ khối thành phần, ta thu được sơ đồ khối của động cơ:



Hình 5.15 Sơ đồ khối động cơ với tín hiệu vào điện áp và tín hiệu ra vận tốc

Hàm truyền của động cơ DC với tín hiệu vào là điện áp và tín hiệu ra là vận tốc:

$$G_{dc}(s)=\frac{\omega(s)}{U(s)}=\frac{\frac{K_m}{(Ls+R)(Js+B)}}{1+\frac{K_mK_e}{(Ls+R)(Js+B)}}=\frac{K_m}{LJs^2+(LB+RJ)s+(K_mK_e+RB)} \quad (5.6)$$

Trong đồ án này, ta quan tâm đến điều khiển vị trí của robot, tín hiệu ra phải là đại lượng dịch chuyển của góc.

Gọi θ_m - góc quay của động cơ. Do đó đạo hàm của góc quay chính là vận tốc góc:

$$\Theta_m(t)=\int \omega(t)dt \quad (5.7)$$

Chuyển sang miền tần số, ta được:

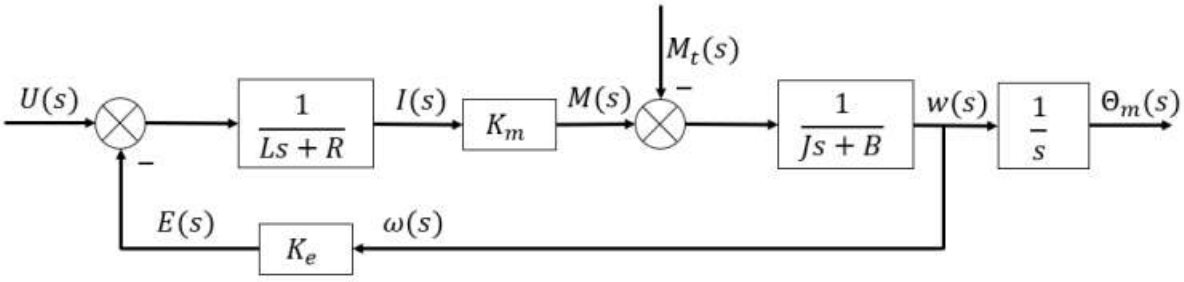
$$\Theta_m(s)=\frac{\omega(s)}{s} \quad (5.8)$$

Lúc này, hàm truyền của động cơ trở thành:

$$G_{dc}(s)=\frac{\Theta_m(s)}{U(s)}=\frac{K_{td}}{T_1s^3+T_2s^2+s} \quad (5.9)$$

Trong đó: $T_1=\frac{\tau_t\tau_cRB_{td}}{K_mK_e+RB_{td}}; T_2=\frac{(\tau_t+\tau_c)RB_{td}}{K_mK_e+RB_{td}}; K_{td}=\frac{K_m}{K_mK_e+RB_{td}}$

Sơ đồ khối của động cơ DC với tín hiệu vào là điện áp và tín hiệu ra là góc quay:

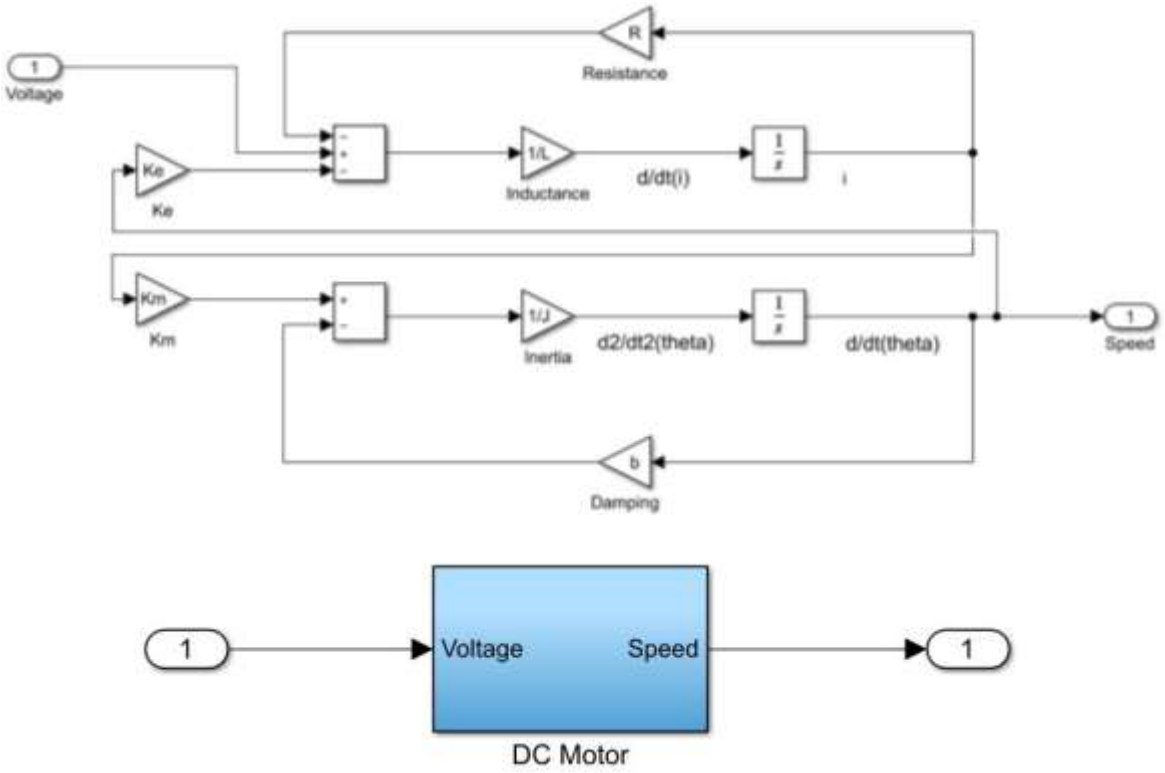


Hình 5.16 Sơ đồ khối động cơ với tín hiệu vào điện áp và tín hiệu ra góc quay

Bảng 5.3 Thông số động cơ servo DC sử dụng trong robot

Hằng số sức điện động $K_e(Vs/rad)$	0.01
Hệ số momen xoắn $K_m(Nm/A)$	0.01
Quán tính của rotor $J_m(kg.m^2)$	0.01
Điện trở R (Ω)	1
Độ tự cảm L (H)	0.5

Mô hình hoá động cơ trên matlab với đầu vào điện áp và đầu ra vận tốc:



Mô hình hoá cảm biến:

Cảm biến dò line sử dụng mắt thu hồng ngoại, giá trị trả về khi gặp line là 1, khi lệch khỏi line là 0. Do đó cảm biến dò line là loại cảm biến logic, giá trị đầu ra sẽ là tín hiệu digital ngẫu nhiên theo vạch line.

Sử dựng cảm biến dò line 5 mắt thu phát, ta có thể có bằng trạng thái như sau:

Error sẽ là phần quan trọng số cho mỗi loại lỗi khi cảm biến ở đúng vị trí như bằng trạng thái.

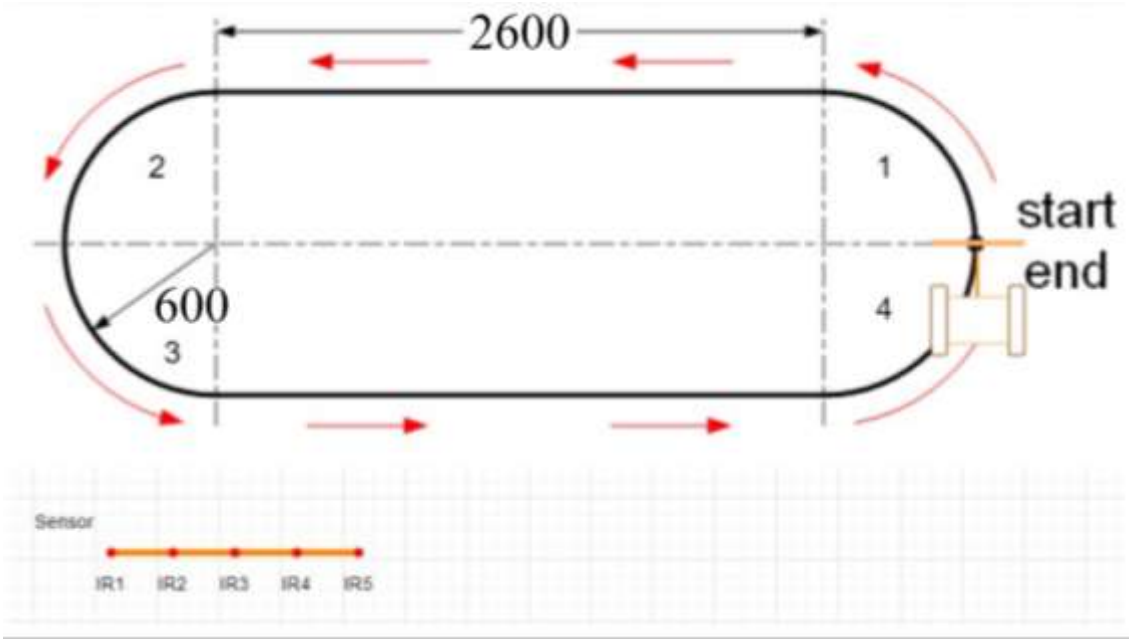
Bảng 5.4 Thiết lập mã hóa lỗi từ cảm biến dò line

	IR1	IR2	IR3	IR4	IR5	Error
Trạng thái	0	0	0	0	1	5
	0	0	1	1	1	4
	0	0	0	1	1	3
	0	0	0	1	0	2
	0	0	1	1	0	1
	0	0	1	0	0	0
	0	1	1	1	0	0

	0	1	1	0	0	-1
	0	1	0	0	0	-2
	1	1	0	0	0	-3
	1	1	1	0	0	-4
	1	0	0	0	0	-5

Để giả lập tín hiệu của bộ cảm biến line, ta sẽ xét tới map của bài toán đặt ra:

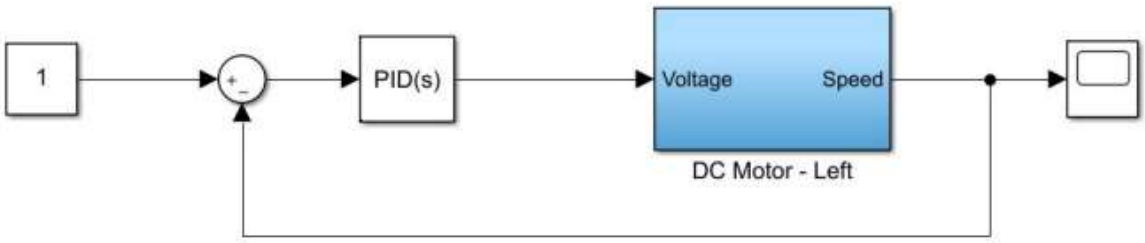
Khi chạy đến những vị trí cong thì giá trị Error sẽ tăng lên.



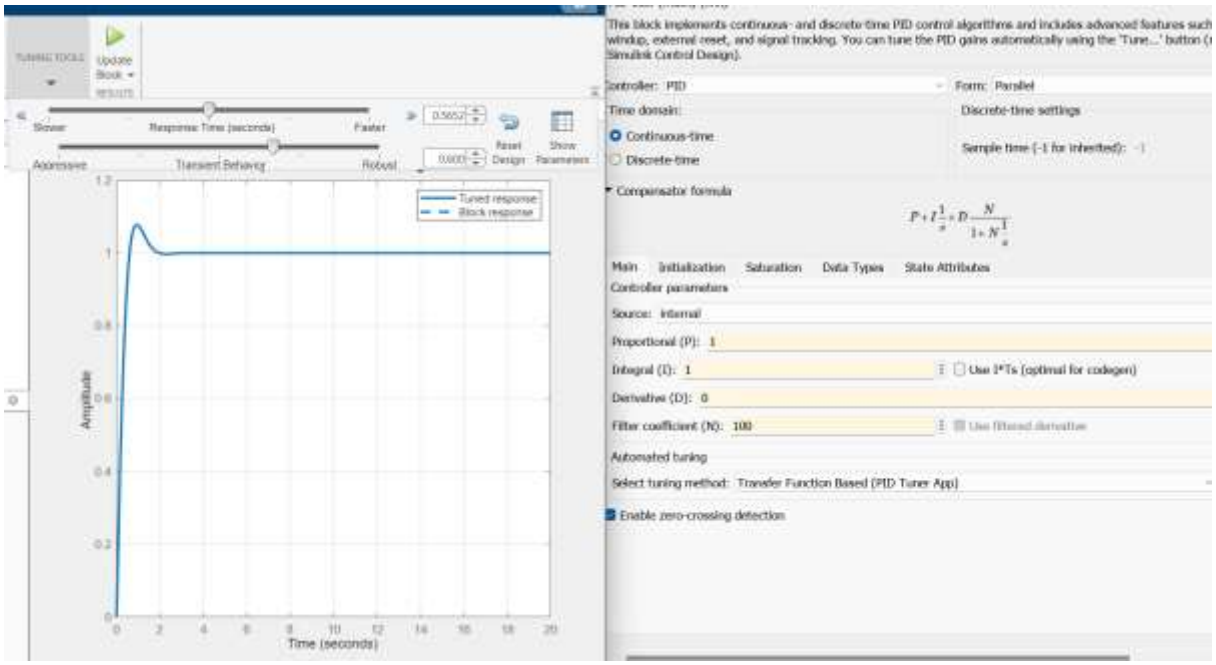
Ở đây là mô phỏng nên ta chọn đầu vào của các cảm biến là tín hiệu ngẫu nhiên 0 hoặc 1.

- *Mô hình hoá bộ điều khiển:*

Bộ điều khiển PID cho từng động cơ:

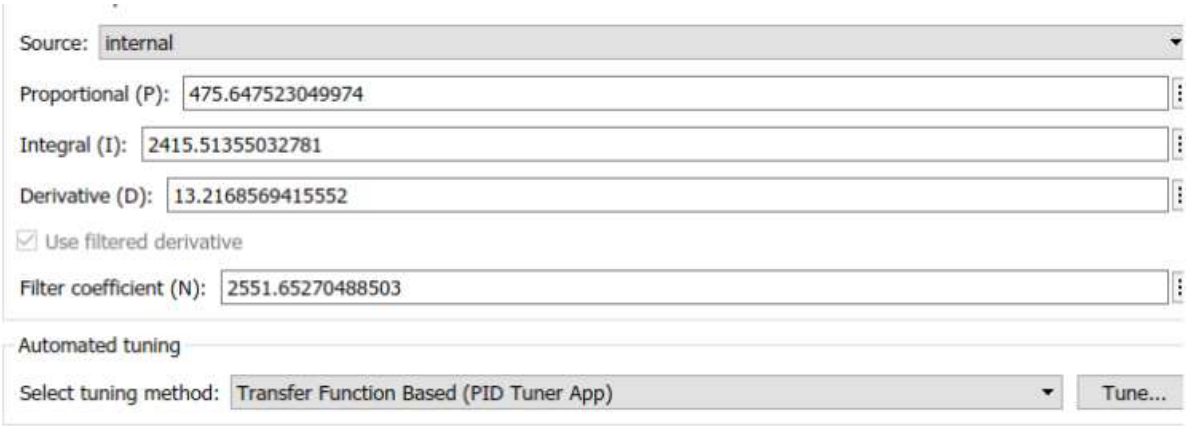


Vào mục Tunning PID để điều chỉnh đáp ứng cho động cơ:

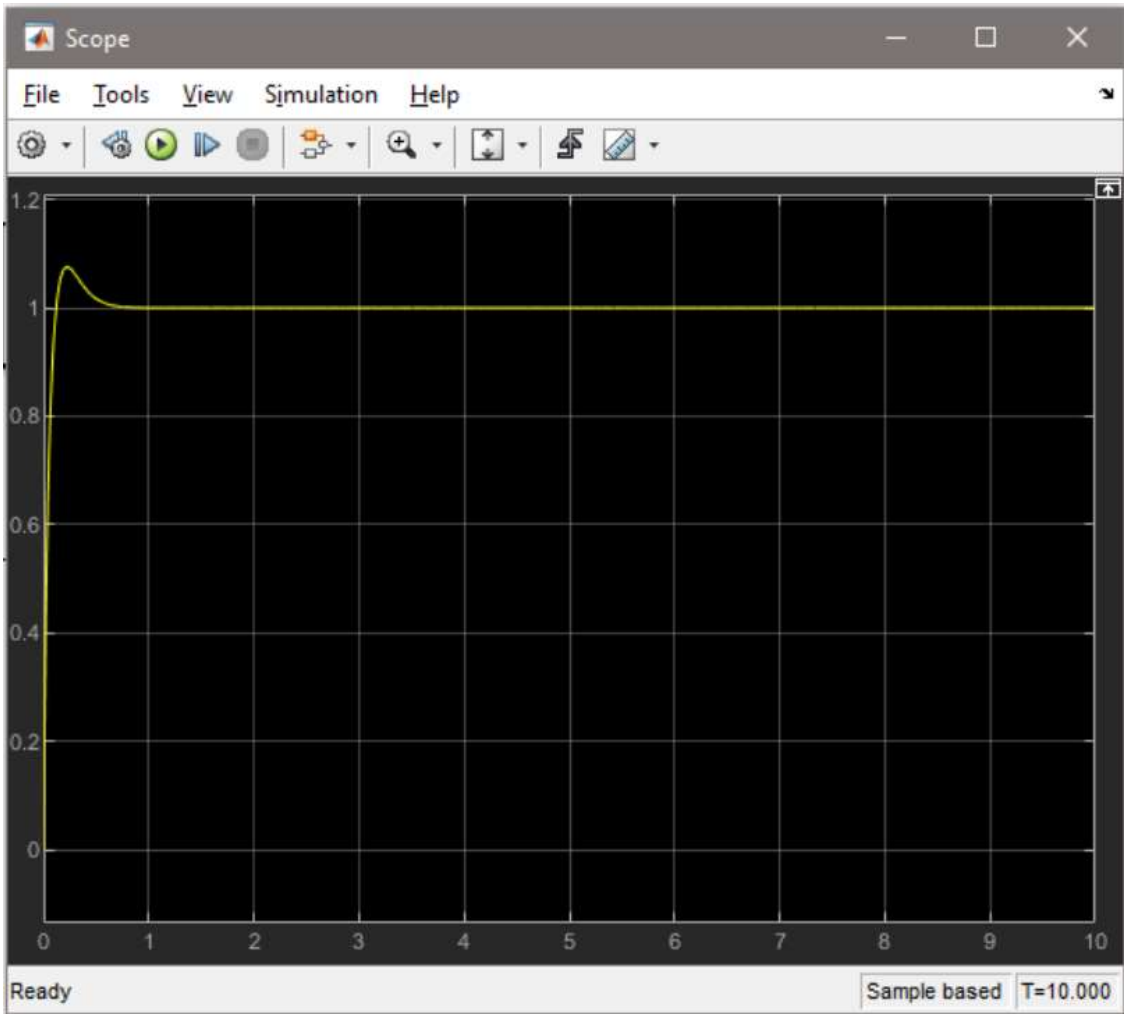


Hình 5.17 Thiết lập thông số tunning PID

Các thông số sau khi tunning PID:

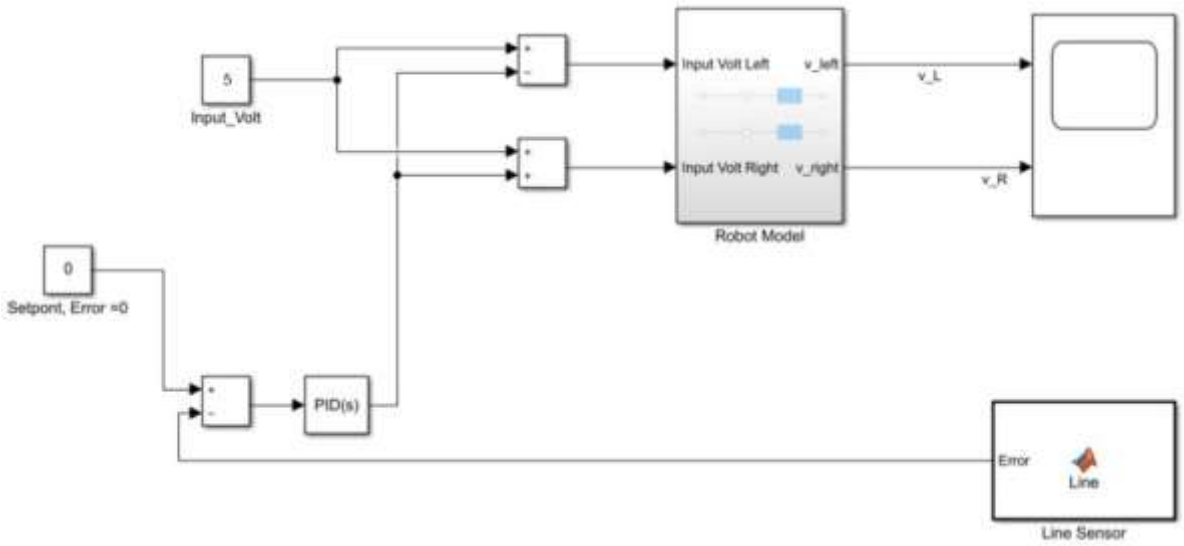


Tốc độ đặt là 1, qua bộ điều khiển PID, mô hình về mối quan hệ giữa vẫn tốc và điện áp ta sẽ có kết quả:



Hình 5.18 Kết quả điều chỉnh tunning PID

Áp dụng cho hệ thống toàn robot:

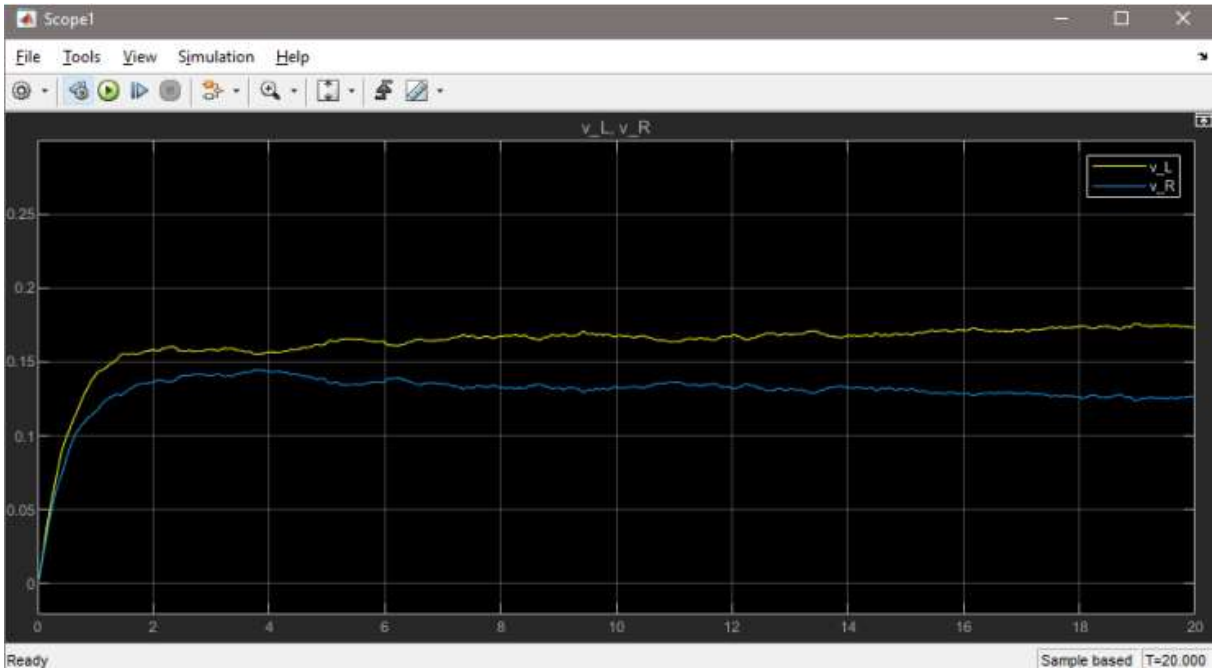


Hình 5.19 Thiết lập các khối mô phỏng quá trình di chuyển vào đường cua

Bộ điều khiển nhận tín hiệu phản hồi từ cảm biến (ở đây tín hiệu được chọn ngẫu nhiên theo bảng Error ở mục 2 từ -5 đến 5);

Qua bộ điều khiển PID để đáp ứng vị trí đầu vào setpoint =0; Điện áp cấp ban đầu sẽ là 5V (có thể tuỳ chỉnh theo người đặt), nhưng phải <12V do động cơ chọn có điện áp max là 12V.

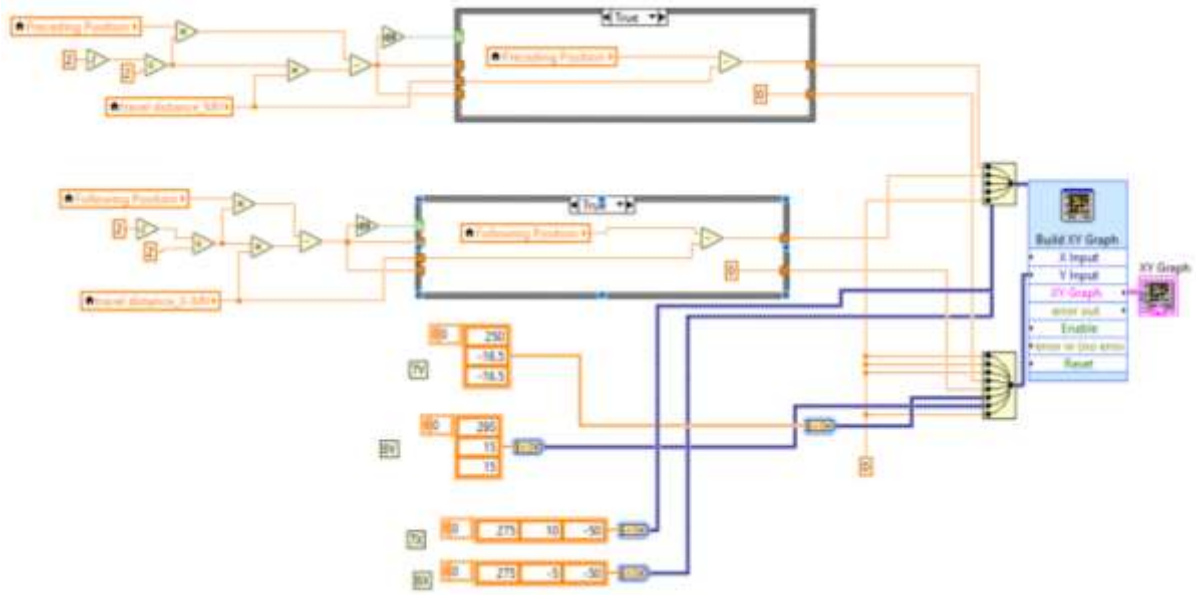
Đáp ứng đầu ra của vận tốc:



Hình 5.20 Kết quả mô phỏng sự chênh lệch tốc độ giữa hai bánh xe khi vào cua

5.3.3. Mô phỏng chuyển động của hai xe

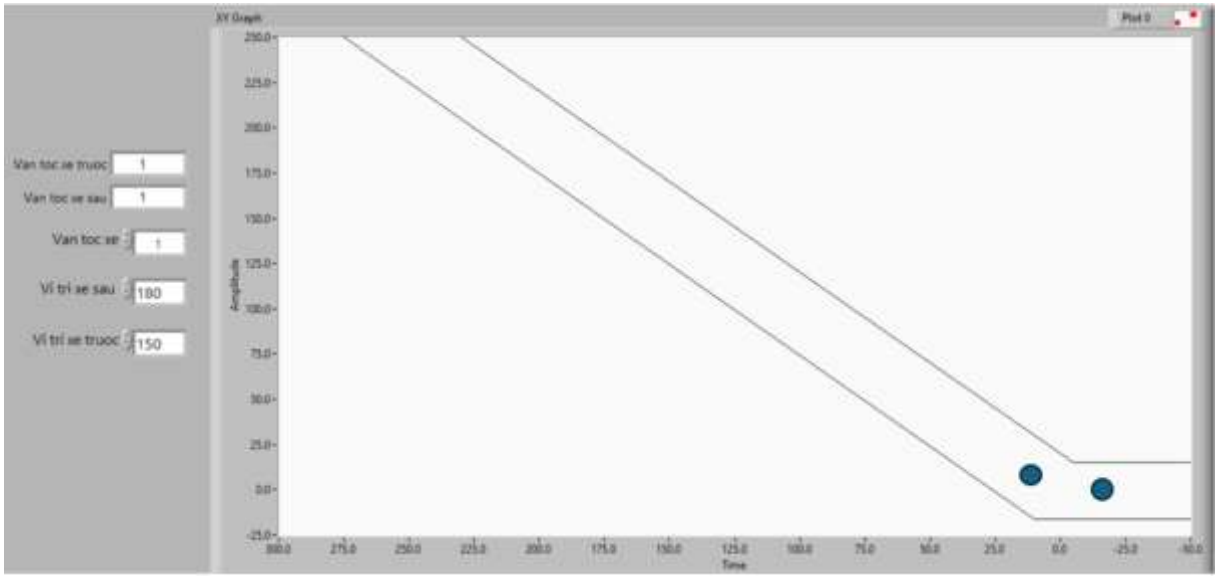
Trong phạm vi nghiên cứu này, phần mềm **LabVIEW** đã được sử dụng như một công cụ mô phỏng trực quan nhằm đánh giá hiệu suất của hệ thống điều khiển vận tốc theo quỹ đạo cong khi hai xe điện cùng di chuyển. LabVIEW là một nền tảng lập trình đồ họa mạnh mẽ, thích hợp để mô hình hóa, mô phỏng và trực quan hóa các hệ thống điều khiển nhúng và tự động. Nhờ khả năng tương tác trực tiếp và khả năng kết nối dữ liệu mô phỏng thời gian thực, LabVIEW giúp kiểm tra các thuật toán điều khiển phức tạp một cách hiệu quả, nhanh chóng và linh hoạt hơn so với các phương pháp lập trình tuyến tính truyền thống.



Hình 5.21 Thiết lập mô hình điều khiển hai xe trên đường cua

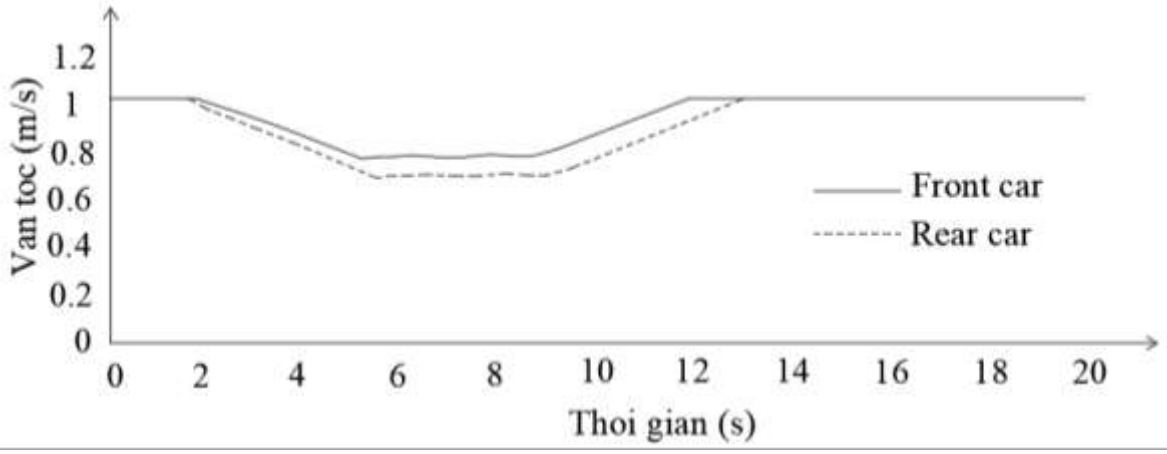
Trong mô hình mô phỏng được thiết kế, hai xe điện thông minh lần lượt di chuyển vào một đoạn đường cong có bán kính nhất định, mô phỏng tình huống thực tế khi các xe đang tham gia giao thông trong môi trường đô thị hoặc khu vực có nhiều khúc cua. Để đảm bảo an toàn và ổn định khi vận hành, mỗi xe được điều khiển bởi một thuật toán PID (Proportional – Integral – Derivative) nhằm giữ vận tốc phù hợp với độ cong của quỹ đạo, đồng thời duy trì khoảng cách an toàn với xe phía trước thông qua thông tin truyền từ hệ thống giao tiếp V2V.

Hình 5.22 thể hiện giao diện mô phỏng của hệ thống điều khiển trong LabVIEW, trong đó quỹ đạo cong được lập trình như một vùng có giới hạn tốc độ nghiêm ngặt, buộc xe phải giảm vận tốc khi đi vào và tăng tốc trở lại khi ra khỏi vùng cong. Giao diện trực quan để theo dõi quá trình thay đổi vận tốc và vị trí của mỗi xe trong thời gian thực.



Hình 5.22 Mô hình mô phỏng giao diện vùng chuyển động theo quỹ đạo cong

Hình 5.23 tiếp tục làm rõ sự khác biệt trong biểu đồ vận tốc theo thời gian giữa hai xe khi di chuyển trên quỹ đạo cong. Kết quả cho thấy vận tốc của xe A giảm nhẹ khi vào cua và nhanh chóng ổn định trở lại sau khi ra khỏi vùng cong. Ngược lại, vận tốc của xe B có sự suy giảm đột ngột hơn nhằm giữ khoảng cách an toàn với xe A – hành vi điều khiển này hoàn toàn nằm trong kỳ vọng của hệ thống Cruise Control kết hợp V2V. Sau khi cả hai xe rời khỏi đoạn cua, tốc độ được điều chỉnh tăng dần trở lại và ổn định theo vận tốc cài đặt ban đầu.



Hình 5.23 Khác biệt giữa vận tốc của mỗi xe di chuyển trên quỹ đạo đường cong

Kết quả mô phỏng cho thấy hiệu quả của thuật toán điều khiển PID trong việc điều chỉnh vận tốc một cách trơn tru, đồng thời cho thấy sự phối hợp nhịp nhàng giữa các phương tiện khi có sự hỗ trợ của công nghệ giao tiếp V2V. Điều này góp phần chứng minh tính khả thi của hệ thống khi ứng dụng trong thực tế để nâng cao tính an toàn, hiệu quả vận hành và giảm thiểu rủi ro trong các tình huống giao thông có mật độ cao hoặc địa hình phức tạp.

5.4. Kết luận chương

Trong chương này, hệ thống điều khiển của mô hình xe điện đã được trình bày chi tiết với sơ đồ kết nối phần cứng trực quan và rõ ràng. Việc kết hợp giữa vi điều khiển Arduino UNO và các cảm biến như dò line IR, siêu âm HC-SR04, cùng với module truyền thông LoRa, đã giúp mô hình xe điện có thể hoạt động một cách linh hoạt, tự động bám theo vạch đường và phát hiện, tránh chướng ngại vật hiệu quả. Bên cạnh đó, mạch điều khiển động cơ A4950 và hệ thống cấp nguồn được thiết kế hợp lý, đảm bảo khả năng cấp điện ổn định cho toàn bộ hệ thống.

Qua đó, mô hình không chỉ đáp ứng được yêu cầu về chức năng điều khiển cơ bản mà còn có tiềm năng mở rộng để nghiên cứu các hệ thống thông minh hơn, như điều khiển hành trình thích ứng (ACC), giao tiếp giữa các xe (V2V), hay áp dụng thuật toán điều khiển PID nhằm tối ưu hoá hành vi di chuyển. Đây là bước đệm quan trọng trong việc phát triển các mô hình xe tự hành thông minh ở quy mô nhỏ, phục vụ cho học tập, nghiên cứu và ứng dụng thực tiễn trong tương lai.

Chương 6: THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ

6.1. Kết quả thực nghiệm

Hình 6.1 Chạy thực nghiệm robot dò line

Kết thúc việc nghiên cứu, thiết kế mô hình xe thông minh tích hợp hệ thống dò đường, hệ thống phát hiện vật cản và hệ thống giao tiếp. Sau một thời gian nghiên cứu thì em đã hoàn tất việc thiết kế, chế tạo mô hình xe giao tiếp tránh va chạm và vận hành thực tế (hình 6.1). Giao tiếp giữa hai xe thực tế khi có mang tải hoàn thành việc bám sa bàn, đo được khoảng cách chính xác và giao tiếp để nâng cao an toàn. Số lần chạy thử đúng theo yêu cầu với thời gian nhanh nhất xấp xỉ 12s, đạt vận tốc xấp xỉ 1,08 m/s như (bảng 6.1)

Bảng 6.1 Kết quả thực nghiệm

Ngày chạy thử	Số lần chạy	Số lần chạy được hết sa bàn	Thời gian nhỏ nhất đạt được (s)	Hệ thống	Sửa đổi bổ sung thêm	Lưu ý
21/3	5	3	30	(1)	Thêm cảm biến dò đường hồng ngoại	Cảm biến bị nhiễu, điều chỉnh tốc độ; Điều chỉnh biến trở trên cảm biến hồng ngoại;

23/3	5	4	18	(1)+(2)	Thêm cảm biến đo khoảng cách	Điều chỉnh thông số PID phù hợp với tốc độ; Nối dây giữa cảm biến khoảng cách và arduino uno càng ngắn càng hoạt động tốt
25/3	5	5	12	(1)+(2)	In 3d bộ vỏ xe, giúp xe ổn định hơn	Tải trọng của vỏ vừa đủ để xe có thể vận hành, không làm giảm tốc độ quá nhiều, không để ảnh hưởng đến quá trình dò đường
4/4	5	0	0	(1)+(2)+(3)	Thêm module LoRa	LoRa chỉ hoạt động với điện áp 3,3V
6/4	5	0	0	(1)+(2)+(3)	Thêm ăng-ten cho LoRa (test ở trạng thái tĩnh)	LoRa đã hoạt động ở trạng thái tĩnh
7/4	5	3	23	(1)+(2)+(3)	Thêm module hạ áp 3,3V cho LoRa để nâng dòng điện	Vì kết hợp 3 hệ thống trên 1 chiếc xe nên xảy ra hiện tưởng nhiễu và cấp dòng điện yếu
19/4	5	5	23	(1)+(2)+(3)		Điều chỉnh PID để dò line bám tốt và phù hợp với tốc độ xung thiết lập

21/4	5	5	12	(1)+(2)+(3)		
Chú thích: (1) Cảm biến dò đường hồng ngoại; (2) Cảm biến khoảng cách siêu âm; (3) LoRa						

6.2. Đánh giá kết quả thực nghiệm

Xe chạy ổn định khi sử dụng bộ điều khiển PID, bám line khá tốt và còn có thể cải tiến để đạt tốc độ lớn hơn.

Duy trì tốc độ ổn đỉnh khi đi trên đường thẳng và tự động điều chỉnh khoảng cách khi vào vòng cua khá tốt, mượt mà.

Trong quá trình thực hiện điều khiển mô hình còn nhiều sai sót về mặt vật lý cũng như thuật ngữ điều khiển nên phải cần nhiều thời gian để tinh chỉnh phù hợp với chức năng của từng hệ thống.

Mô hình điều chỉnh góc cua bằng thước lái nên đôi khi vẫn xảy ra hiện tượng lệch ra ngoài line.

Hai động cơ có sai số về tốc độ với nhau nên làm giảm tính ổn định của xe.

Cảm biến của xe đôi lúc bị nhiễu do điều kiện môi trường.

Sự giao tiếp giữa hai xe khá tốt nên tránh được sự va chạm trong nhiều tình huống khác nhau.

6.3. Đề xuất phương án hiệu chỉnh thiết kế

Để khắc phục các sai số này, các giải pháp được đề ra bao gồm:

***Tăng độ chính xác khi gia công cơ khí**:*

- Sử dụng máy in 3D có độ phân giải cao để in vỏ xe và giá đỡ linh kiện, hạn chế sai số lắp ghép.

- Tối ưu kết cấu khung xe để giảm rung lắc khi xe vận hành ở tốc độ cao hoặc vào cua.

***Hiệu chỉnh lại thuật toán PID**:*

- Tinh chỉnh các thông số P, I, D nhằm tăng khả năng ổn định khi dò line và giảm độ lệch khi cua gấp.

- Kết hợp PID với thuật toán dự đoán quỹ đạo để xử lý tốt hơn tại các đoạn đường cong phức tạp.

***Cân bằng tốc độ hai động cơ**:*

- Sử dụng encoder để đo tốc độ thực tế từng bánh xe, từ đó điều chỉnh PWM phù hợp cho từng động cơ nhằm đảm bảo xe di chuyển thẳng và ổn định hơn.

***Cải thiện nguồn cấp và lọc nhiễu**:*

- Tăng cường module nguồn ổn áp 3.3V riêng biệt cho LoRa để tránh sụt áp.

- Bổ sung tụ lọc và chống nhiễu cho mạch điều khiển, đặc biệt tại các điểm giao thoa giữa cảm biến siêu âm, LoRa và vi điều khiển.

***Tối ưu vị trí và độ nhạy cảm biến**:*

- Hiệu chỉnh lại khoảng cách giữa các mắt cảm biến dò line để nhận diện tốt hơn các đoạn line cong hoặc bị đứt quãng.

- Gắn cảm biến siêu âm ở vị trí cố định, tránh bị rung lắc, tăng độ chính xác trong phát hiện vật cản.

***Nâng cấp giao tiếp V2V**:*

- Thiết lập lại cấu trúc gói tin LoRa để đảm bảo độ chính xác và giảm độ trễ.

- Kiểm tra và tối ưu thuật toán truyền/nhận, tránh xung đột khi truyền đồng thời giữa hai xe.

***Bổ sung cơ chế phản hồi lỗi**:*

- Tích hợp LED báo trạng thái hoặc hiển thị thông tin trên màn hình OLED để dễ theo dõi lỗi.

- Thiết lập chế độ khôi phục khi hệ thống mất tín hiệu line hoặc LoRa, giúp xe dừng an toàn và chờ tái khởi động.

KẾT LUẬN

Trên đây là bản báo cáo **Capstone Project** với đề tài *"**Nghiên cứu, thiết kế mô hình xe thông minh sử dụng công nghệ giao tiếp V2V**"*. Đây là một đề tài mang tính thực tiễn cao, phù hợp với xu thế phát triển của thời đại công nghiệp hóa, hiện đại hóa đất nước, trong bối cảnh cuộc cách mạng công nghiệp lần thứ tư đang diễn ra mạnh mẽ. Sự cạnh tranh toàn cầu đòi hỏi việc nâng cao năng suất và chất lượng sản phẩm thông qua ứng dụng các công nghệ tự động và hệ thống thông minh.

Thông qua đề tài này, em đã có cơ hội tiếp cận, nghiên cứu và xây dựng một mô hình xe thông minh tích hợp ba hệ thống chính: hệ thống dò đường, hệ thống đo khoảng cách để tránh vật cản, và đặc biệt là công nghệ giao tiếp V2V (Vehicle-to-Vehicle). Đây là những nền tảng quan trọng trong việc phát triển các hệ thống giao thông thông minh và xe tự hành trong tương lai. Cụ thể, các công việc đã được thực hiện bao gồm:

- Nghiên cứu tổng quan về phương tiện thông minh và công nghệ giao tiếp V2V;
- Thiết kế mạch điện cho hệ thống mô hình xe thông minh bằng phần mềm AutoCAD;
- Dựng mô hình 3D cơ khí bằng phần mềm Inventor;
- Tính toán các thông số cơ khí phù hợp với thiết kế thực tế;
- Tìm hiểu nguyên lý hoạt động và tích hợp các linh kiện điện tử cần thiết;
- Xây dựng sơ đồ khối chức năng và sơ đồ nguyên lý của hệ thống điều khiển;
- Lập trình điều khiển để xe có thể tự dò line, tránh vật cản và nhận/gửi tín hiệu V2V.

Thông qua quá trình thực hiện Capstone Project, em không chỉ vận dụng được các kiến thức chuyên môn đã được đào tạo tại **Trường Đại học Bách Khoa – Đại học Đà Nẵng**, mà còn tích lũy thêm nhiều kỹ năng quan trọng như: giải quyết vấn đề, tra cứu tài liệu khoa học, viết báo cáo và trình bày kết quả nghiên cứu.

Em xin gửi lời cảm ơn chân thành đến thầy **TS. Hoàng Thắng**, cùng quý thầy cô trong bộ môn **Kĩ thuật ô tô**, đã luôn tận tình hướng dẫn, hỗ trợ và tạo điều kiện thuận lợi cho chúng em trong suốt quá trình thực hiện đề tài.

Tuy nhiên, do giới hạn về thời gian cũng như kiến thức, đề tài hiện tại mới chỉ giải quyết được một số khía cạnh cơ bản trong thiết kế và triển khai mô hình xe thông minh. Vẫn còn nhiều vấn đề cần nghiên cứu sâu hơn để hoàn thiện và có thể áp dụng rộng rãi trong thực tiễn, đặc biệt là trong lĩnh vực giao thông và nông nghiệp thông minh. Chúng em rất mong nhận được sự góp ý, phản biện từ quý thầy cô và các bạn để có thể hoàn thiện đề tài tốt hơn trong tương lai.

Một lần nữa, em xin chân thành cảm ơn!

TÀI LIỆU THAM KHẢO

[1]. TS.Phan Minh Đức. ***"Lý thuyết điều khiển tự động"***. Sách lưu hành nội bộ trường Đại học Bách Khoa – Đại học Đà Nẵng, 2017.
[2]. National Instruments, ***"PID Control Toolset User Manual"***, National Instruments Corporation, USA, 2009.
[3]. Nguyễn Trọng Hiệp, ***"Truyền động điện ô tô hiện đại"***, Nhà xuất bản Giao thông Vận tải, Hà Nội, 2019.
[4]. Warren, J.-D., Adams, J., & Molle, H. (2011). ***"Arduino Robotics"***. Apress.
[5]. William B.Ribbens. "***Understanding Automotive Electronics***". Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA.
[6]. TS. Trần Quang Thọ. ***"Truyền động điện tự động"***. Nhà xuất bản đại học quốc gia TP Hồ Chí Minh, 2020.
[7]. Giảng viên Khoa Điện – Điện tử, ĐH SPKT Vĩnh Long. ***"Giáo trình Vi điều khiển và Ứng dụng"***. Tài liệu lưu hành nội bộ Vĩnh Long, 2017.
[8]. Richard Zurawski, "***Automotive Embedded System Handbook***", Edited by Nicolas Navet and Françoise Simonot-Lion
[9]. ThS. Dương Thế Anh. ***"Nghiên cứu thiết kế chế tạo bộ thiết bị truyền thông tin giữa các xe ô tô tham gia giao thông trên đường"***. *Bộ Khoa học và Công nghệ, Cục thông tin khoa học và công nghệ Quốc Gia, 2023.*
[10]. Bo Bernhardsson and Karl Johan Åström .***"Control System Design - PID Control"***. *Department of Automatic Control LTH, Lund University*
[11]. Daddanala, Ramya, et al. "Vehicle to vehicle (V2V) Communication Protocol: components, benefits, challenges, safety and machine learning applications" *arXiv preprint arXiv:2102.07306* (2021)
[12]. Jahnavi, Maudhoo, et al. "Vehicle to vehicle communication for collision avoidance" *International Journal for Research in Applied Science and Engineering Technology* 6.5 (2018): 1380-1386
[13]. Cai, Lin, et al. "A neural-fuzzy controller for intelligent cruise control of vehicle in highways" *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems*. Vol. 2. IEEE, 2003.
[14]. Takai, Isamu, et al. "Optical vehicle-to-vehicle communication system using LED transmitter and camera receiver." *IEEE photonics journal* 6.5 (2014): 1-14
[15]. Velvizhi, V. A., et al. "Communication Between Two Vehicles Using LoRa." *2021 4th International Conference on Computing and Communications Technologies (ICCCT)*. IEEE, 2021.
[16]. Thai, Pham Quoc, et al. "Design and Demonstration of Intelligent Chain Control System for Personal Rapid Transit" *International Conference on Engineering Research and Applications*. Cham: Springer Nature Switzerland, 2023.
[17]. Cortes, Filipe, Rafael Carvalho, and Luis Miguel Pires. "Vehicle to Vehicle Communication with LoRa network." (2024).

[18]. Farivar, Faezeh, et al. "On the security of networked control systems in smart vehicle and its adaptive cruise control." *IEEE Transactions on Intelligent Transportation Systems* 22.6 (2021): 3824-3831.
[19]. Rout, M. K., et al. "PID controller design for cruise control system using genetic algorithm." *2016 international conference on electrical, electronics, and optimization techniques (ICEEOT)*. IEEE, 2016.
[20] Monolithic Power Systems, "Analog vs. Digital Signals: Uses, Advantages and Disadvantages," *MPS*, [Online]. Available: https://www.monolithicpower.com/en/analog-vs-digital-signals. [Accessed: 22-May-2025]

PHỤ LỤC

Code xe trước:

```
#include <SoftwareSerial.h>
// Định nghĩa chân động cơ
#define MOTOR_L_1 7
#define MOTOR_L_2 6
#define MOTOR_R_1 5
#define MOTOR_R_2 4
// Định nghĩa chân cảm biến line
const uint8_t SENSORS_PIN[] = {A0, A1, A2, A3, A4};
// Cảm biến siêu âm
const int trigPin = 8;
const int echoPin = 9;
// LoRa E32
SoftwareSerial loraSerial(2, 3); // RX, TX
const int M0 = 10;
const int M1 = 11;
// PID parameters
float Kp = 20;
float Ki = 0.0002;
float Kd = 12;
int P, I, D, PIDValue = 0;
float error = 0;
int lastError = 0;
bool isRunning = true;
bool isNoLine = false;
bool loraBlocked = false;  // true nếu nhận được STOP từ LoRa
// Tốc độ động cơ
const uint8_t maxspeeda = 220;
const uint8_t maxspeedb = 220;
const uint8_t minspeeda = 20;
const uint8_t minspeedb = 20;
```

```
int lastSpeedA = 0;
int lastSpeedB = 0;
// Khoảng cách an toàn
const int safeDistance = 20;
// Gửi LoRa không lặp lại liên tục
unsigned long lastSendTime = 0;
String lastSignal = "";
// ==== SETUP ====
void setup() {
  pinMode(MOTOR_L_1, OUTPUT);
  pinMode(MOTOR_L_2, OUTPUT);
  pinMode(MOTOR_R_1, OUTPUT);
  pinMode(MOTOR_R_2, OUTPUT);
  digitalWrite(MOTOR_L_1, HIGH);
  digitalWrite(MOTOR_L_2, HIGH);
  digitalWrite(MOTOR_R_1, HIGH);
  digitalWrite(MOTOR_R_2, HIGH);
  for (auto pin : SENSORS_PIN) pinMode(pin, INPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  pinMode(M0, OUTPUT);
  pinMode(M1, OUTPUT);
  digitalWrite(M0, LOW);
  digitalWrite(M1, LOW); // Normal mode

  loraSerial.begin(9600);
  Serial.begin(115200);
  delay(100);
}
// ==== LOOP ====
void loop() {
  receive_lora();  // Luôn kiểm tra tín hiệu từ robot khác
```

```
  if (loraBlocked) {
    stop_motors();
    return; // Bỏ qua toàn bộ nếu bị chặn từ xa
  }
  if (isRunning) {
    read_sensors();       // Đọc cảm biến
    caculate_pid();       // PID
    motor_control();      // Điều khiển
    send_lora_signal();   // Gửi tín hiệu
    delay(10);
  } else {
    stop_motors();
    if (check_restart_condition()) {
      isRunning = true;
    }
  }
}
// ==== ĐỌC CẢM BIẾN LINE ====
void read_sensors() {
  String sensorArray = "";
  int sumSensor = 0;

  for (auto pin : SENSORS_PIN) {
    int val = digitalRead(pin);
    sensorArray += (char)(val + '0');
    sumSensor += val;
  }
  isNoLine = (sumSensor == 0);
  Serial.println(sensorArray);
  if (sensorArray == "00001") error = 5;
  else if (sensorArray == "00111") error = 4;
  else if (sensorArray == "00011") error = 3;
  else if (sensorArray == "00010") error = 2;
```

```
  else if (sensorArray == "00110") error = 1;
  else if (sensorArray == "00100" || sensorArray == "01110") error = 0;
  else if (sensorArray == "01100") error = -1;
  else if (sensorArray == "01000") error = -2;
  else if (sensorArray == "11000") error = -3;
  else if (sensorArray == "11100") error = -4;
  else if (sensorArray == "10000") error = -5;
  else if (sensorArray == "00000") error = lastError;
  else if (sensorArray == "11111" || sensorArray == "10101") isRunning = false;

  float distance = read_ultrasonic();
  if (distance < safeDistance) isRunning = false;
}
// ==== ĐỌC CẢM BIẾN SIÊU ÂM ====
float read_ultrasonic() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  long duration = pulseIn(echoPin, HIGH);
  float distance = (duration * 0.034) / 2;
  return distance;
}
// ==== PID ====
void caculate_pid() {
  P = error;
  if (abs(error) < 6) I += Ki * error;
  D = error - lastError;
  PIDValue = Kp * P + I + Kd * D;
  lastError = error;
}
// ==== ĐIỀU KHIỂN ĐỘNG CƠ ====
```

```
void set_motor(int speedA, int speedB) {
  speedA = -speedA;
  if (speedA > 0) {
    speedA = constrain(speedA, minspeeda, maxspeeda);
    digitalWrite(MOTOR_R_2, LOW);
    analogWrite(MOTOR_R_1, speedA);
  } else {
    speedA = constrain(speedA, -maxspeeda, -minspeeda);
    digitalWrite(MOTOR_R_2, HIGH);
    analogWrite(MOTOR_R_1, 255 + speedA);
  }

  if (speedB > 0) {
    speedB = constrain(speedB, minspeedb, maxspeedb);
    digitalWrite(MOTOR_L_1, LOW);
    analogWrite(MOTOR_L_2, speedB);
  } else {
    speedB = constrain(speedB, -maxspeedb, -minspeedb);
    digitalWrite(MOTOR_L_1, HIGH);
    analogWrite(MOTOR_L_2, 255 + speedB);
  }
}

void motor_control() {
  if (isNoLine) {
    set_motor(lastSpeedA, lastSpeedB);
  } else {
    if (error < 0)
      set_motor(maxspeeda, maxspeedb + PIDValue);
    else
      set_motor(maxspeeda - PIDValue, maxspeedb);
    lastSpeedA = maxspeeda;
    lastSpeedB = maxspeedb;
```

```
  }
}
// ==== DỪNG ĐỘNG CƠ ====
void stop_motors() {
  digitalWrite(MOTOR_R_2, HIGH);
  digitalWrite(MOTOR_R_1, HIGH);
  digitalWrite(MOTOR_L_1, HIGH);
  digitalWrite(MOTOR_L_2, HIGH);
  delay(100);
  digitalWrite(MOTOR_L_1, LOW);
  digitalWrite(MOTOR_L_2, LOW);
  digitalWrite(MOTOR_R_1, LOW);
  digitalWrite(MOTOR_R_2, LOW);
}

// ==== ĐIỀU KIỆN KHỞI ĐỘNG LẠI ====
bool check_restart_condition() {
  float distance = read_ultrasonic();
  return (distance >= safeDistance);
}

// ==== GỬI TÍN HIỆU QUA LoRa ====
void send_lora_signal() {
  float distance = read_ultrasonic();
  unsigned long now = millis();
  String currentSignal = (distance < safeDistance) ? "STOP" : "GO";

  if (currentSignal != lastSignal || now - lastSendTime >= 200) {
    loraSerial.println(currentSignal);
    Serial.print("Sent via LoRa: ");
    Serial.println(currentSignal);
    lastSignal = currentSignal;
    lastSendTime = now;
```

```
  }
}

// ==== NHẬN TÍN HIỆU LoRa ====
void receive_lora() {
  if (loraSerial.available()) {
    String message = loraSerial.readStringUntil('\n');
    message.trim();
    message.toUpperCase();

    if (message == "STOP") {
      loraBlocked = true;
      Serial.println("Received STOP via LoRa → BLOCKED");
    } else if (message == "GO") {
      loraBlocked = false;
      Serial.println("Received GO via LoRa → UNBLOCKED");
    } else {
      Serial.print("Received unknown: ");
      Serial.println(message);
    }
  }
}
```

Code xe sau:

```
#include <Servo.h>
#include <SoftwareSerial.h>
// ==== Định nghĩa chân ====
#define MOTOR_L_1 3
#define MOTOR_L_2 6
#define MOTOR_R_1 11
#define MOTOR_R_2 5
#define TRIG_PIN 10
#define ECHO_PIN 12
```

```
#define LORA_RX 2
#define LORA_TX 4
#define LORA_M0 8
#define LORA_M1 7
SoftwareSerial loraSerial(LORA_RX, LORA_TX);
const uint8_t SENSORS_PIN[] = {A0, A1, A2, A3, A4};
// ==== PID ====
float Kp = 14;
float Ki = 0.00005;
float Kd = 12;
int P, I, D;
int PIDValue = 0;
float error = 0;
int lastError = 0;
int isRunning = 1;
bool isNoLine = false;
// ==== Tốc độ ====
const uint8_t maxspeedA = 160;
const uint8_t maxspeedB = 160;
const uint8_t minspeedA = 50;
const uint8_t minspeedB = 50;
Servo steeringServo;
int lastPIDValue = 0;
int lastSteeringAngle = 90;
String receivedCommand = "STOP"; // Mặc định dừng khi chưa có tín hiệu

void setup() {
  // Động cơ
  pinMode(MOTOR_L_1, OUTPUT);
  pinMode(MOTOR_L_2, OUTPUT);
  pinMode(MOTOR_R_1, OUTPUT);
  pinMode(MOTOR_R_2, OUTPUT);
  stop_motor();
```

```
  // Cảm biến dò line
  for (auto sensorPin : SENSORS_PIN) {
    pinMode(sensorPin, INPUT);
  }
  // Cảm biến siêu âm
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  // Servo
  steeringServo.attach(9);
  steeringServo.write(90);
  // LoRa
  pinMode(LORA_M0, OUTPUT);
  pinMode(LORA_M1, OUTPUT);
  digitalWrite(LORA_M0, LOW);
  digitalWrite(LORA_M1, LOW); // Normal mode
  loraSerial.begin(9600);
  // Debug
  Serial.begin(115200);
  delay(100);
}

void loop() {
  listen_lora();  // Nhận lệnh liên tục
  if (receivedCommand == "GO") {
    read_sensors();
    caculate_pid();
    steering_control();
    motor_control();
  } else {
    stop_motor();
  }

  delay(10);
```

```
}
// ==== Nhận tín hiệu từ xe trước qua LoRa ====
void listen_lora() {
  if (loraSerial.available()) {
    receivedCommand = loraSerial.readStringUntil('\n');
    receivedCommand.trim();
    Serial.print("Received via LoRa: ");
    Serial.println(receivedCommand);
  }
}
void read_sensors() {
  String sensorArray = "";
  int sumSensor = 0;
  for (auto sensorPin : SENSORS_PIN) {
    sensorArray += (char)(digitalRead(sensorPin) + 48);
    sumSensor += digitalRead(sensorPin);
  }
  isNoLine = sumSensor == 0;
  Serial.println(sensorArray);
  if (sensorArray == "00001") error = 5;
  else if (sensorArray == "00111") error = 4;
  else if (sensorArray == "00011") error = 3;
  else if (sensorArray == "00010") error = 2;
  else if (sensorArray == "00110") error = 1;
  else if (sensorArray == "00100") error = 0;
  else if (sensorArray == "01110") error = 0;
  else if (sensorArray == "01100") error = -1;
  else if (sensorArray == "01000") error = -2;
  else if (sensorArray == "11000") error = -3;
  else if (sensorArray == "11100") error = -4;
  else if (sensorArray == "10000") error = -5;
  else if (sensorArray == "00000") {
    if (isNoLine) {
```

```
      PIDValue = lastPIDValue;
      return;
    }
  } else if (sensorArray == "11111" || sensorArray == "10101") {
    isRunning = 0;
  }
}
void caculate_pid() {
  P = error;
  if (abs(error) < 6) I += Ki * error;
  D = error - lastError;
  PIDValue = Kp * P + I + Kd * D;
  lastError = error;
  lastPIDValue = PIDValue;
}
void steering_control() {
  int steeringAngle = 90 + PIDValue * 0.8;
  steeringAngle = constrain(steeringAngle, 53, 127);
  steeringServo.write(180 - steeringAngle);
  lastSteeringAngle = steeringAngle;
}
void motor_control() {
  long distance = read_ultrasonic();
  Serial.print("Distance: ");
  Serial.println(distance);
  if (distance < 20) {
    stop_motor();
    return;
  }
  int speedLeft = maxspeedA;
  int speedRight = maxspeedB;
  if (error < 0) {
    speedRight += PIDValue;
```

```
    speedLeft -= PIDValue;
  } else {
    speedLeft -= PIDValue;
    speedRight += PIDValue;
  }
  speedLeft = constrain(speedLeft, minspeedA, maxspeedA);
  speedRight = constrain(speedRight, minspeedB, maxspeedB);

  set_motor(speedLeft, speedRight);
}
void set_motor(int speedA, int speedB) {
  speedA = -speedA;
  if (speedA > 0) {
    speedA = constrain(speedA, minspeedA, maxspeedA);
    digitalWrite(MOTOR_R_2, LOW);
    analogWrite(MOTOR_R_1, speedA);
  } else {
    speedA = constrain(speedA, -maxspeedA, -minspeedA);
    digitalWrite(MOTOR_R_2, HIGH);
    analogWrite(MOTOR_R_1, 255 + speedA);
  }
  if (speedB > 0) {
    speedB = constrain(speedB, minspeedB, maxspeedB);
    digitalWrite(MOTOR_L_1, LOW);
    analogWrite(MOTOR_L_2, speedB);
  } else {
    speedB = constrain(speedB, -maxspeedB, -minspeedB);
    digitalWrite(MOTOR_L_1, HIGH);
    analogWrite(MOTOR_L_2, 255 + speedB);
  }
}
void stop_motor() {
  digitalWrite(MOTOR_R_1, LOW);
```

```
  digitalWrite(MOTOR_R_2, LOW);
  digitalWrite(MOTOR_L_1, LOW);
  digitalWrite(MOTOR_L_2, LOW);
}

long read_ultrasonic() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  long duration = pulseIn(ECHO_PIN, HIGH);
  long distance = duration * 0.034 / 2;
  return distance;
}
```