Thiết kế robot ứng dụng phục vụ trong nhà hàng sử dụng định vị bằng Lidar-SLAM

Design of a service robot for restaurant applications using Lidar-Based SLAM localization

Nguyễn Văn Phúc, Nguyễn Đại Dương*

School of Electrical and Electronic Engineering, HUST
**Corresponding author E-mail: duong.nguyendai@hust.edu.vn*
DOI: https://doi.org/10.64032/mca.v29i3.319

Abstract

In this study, we propose the design of a table-running robot intended for deployment in restaurant envi-ronments, aiming to automate food service operations while enhancing safety and accuracy. The proposed robot employs a multi-sensor fusion localization method, integrating data from a LIDAR sensor, wheel en-coders, an accelerometer, and a gyroscope. Unlike systems that rely on magnetic tape for navigation, our approach utilizes a SLAM (Simultaneous Localization and Mapping) algorithm to process sensor data and construct a dynamic map of the environment. This enables the robot to navigate flexibly and reroute in re-sponse to moving obstacles. Additionally, we introduce a localization method using V-L markers, allowing the robot to autonomously locate and dock with its charging station when battery levels are low. Experi-mental results confirm the robot's accurate and reliable performance in real-world conditions.

***Keywords**: Simultaneous Localization and Mapping (SLAM), Localize, Service robot, ROS, navigation*

Tóm tắt

Trong nghiên cứu này, nhóm chúng tôi đề xuất thiết kế một robot chạy bàn để phục vụ trong các nhà hàng nhằm tự động hóa việc phục vụ, đảm bảo an toàn, chính xác. Robot đề xuất sử dụng phương pháp định vị kết hợp đa cảm biến bao gồm LIDAR, encoder bánh xe, gia tốc kế và con quay hồi chuyển. Hệ thống được phát triển không dựa trên định vị bằng băng từ mà thông tin từ các cảm biến sẽ được vào thuật toán định vị SLAM để vẽ lại bản đồ môi trường. Việc này giúp robot di chuyển linh hoạt hơn khi có thể tìm đường khác đến đích khi xuất hiện vật cản động. Chúng tôi cũng đề xuất thuật toán sử dụng V-L marker để robot có thể tự tìm về và vào đúng vị trí trạm sạc khi pin gần hết dung lượng. Thử nghiệm cho thấy hoạt động chính xác của robot trong thực tế.

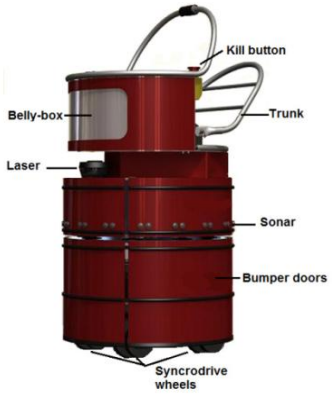
1. Giới thiệu

Cùng với sự phát triển nhanh chóng của công nghệ chế tạo, truyền thông và định vị, việc sử dụng robot để thực hiện các dịch vụ trong đời sống của con người đã và đang ngày càng phát triển, giúp giảm sức lao động và nâng cao độ chính xác khi phục vụ, đem lại sự hài lòng cho khách hàng. Ngoài ra, robot dịch vụ sẽ giúp giảm thiểu các bệnh lây nhiễm do tiếp xúc. Các nghiên cứu về robot dịch vụ đang phát triển, và nhiều định nghĩa khác nhau đã được đưa ra để mô tả loại robot này [1]. Theo liên đoàn Robot quốc tế (International Federation of Robotics), robot dịch vụ được định nghĩa như sau: "Robot dịch vụ là một loại robot hoạt động bán tự động hoặc hoàn toàn tự động để thực hiện các dịch vụ hữu ích cho sự phát triển của con người và thiết bị, loại trừ các hoạt động sản xuất." [2]. Hiện nay, nhà hàng và bệnh viện đang là hai môi trường tiềm năng nhất cho ứng dụng của robot dịch vụ [3]. Trong lĩnh vực nhà hàng, robot lễ tân là loại robot dịch vụ phổ biến nhất. Chúng được thiết kế để vận chuyển thức ăn và đồ uống từ bếp đến bàn ăn, đồng thời đảm nhận việc thu thập đĩa và vỏ trở lại bếp. Đặc biệt trong giai đoạn của đại dịch Covid-19, tốc độ lây nhiễm cực nhanh đã tạo ra một tình huống khẩn cấp thúc đẩy việc phát triển các robot dịch vụ tại các nhà hàng, bệnh viện,... [4]. Trong những tình huống như vậy, việc phục vụ không tiếp xúc là điều thực sự quan trọng. Một trong những ưu điểm chính của robot phục vụ so với con người là đảm bảo năng suất và mức độ hoạt động liên tục, đồng đều trong suốt quá trình làm việc. Chính vì thế, nghiên cứu, thiết kế và chế tạo robot lễ tân để ứng dụng vào thực tế là vô cùng cần thiết.

Dựa vào công nghệ hiện có, chúng ta có thể phân loại các dòng robot lễ tân khác nhau. Trước đây, đã có một số nghiên cứu được công bố về dòng robot lễ tân theo hệ thống dẫn đường line sử dụng vi điều khiển làm trung tâm tính toán. Một nghiên cứu đáng chú ý được thực hiện bởi M. Pakdaman Kazi và đồng nghiệp đã bắt đầu quá trình thiết kế và phát triển dòng robot này [5]. Tiếp theo, Mahmud Hasan và các cộng sự đã triển khai và mở rộng một số tính năng khác như đèn cảnh báo và khả năng đo lượng pin của robot [6]. Dòng robot lễ tân dò line truyền thống thường di chuyển theo đường cố định đã được định sẵn trên sàn. Tuy nhiên, khi gặp vật cản, robot gặp khó khăn hoặc mất thời gian để tiếp tục hành trình, đồng thời tốc độ di chuyển của nó cũng bị hạn chế. Nhằm cải thiện các hạn chế này, vào năm 2012, Zalama và các đồng nghiệp [7] đã ứng dụng kỹ thuật SLAM (Simultaneous Localization and Mapping) [8] vào robot lễ tân. Công nghệ này cho phép robot tạo ra bản đồ của nhà hàng bằng cách sử dụng LIDAR hoặc stereo camera, kết hợp với encoder bánh xe và cảm biến gia tốc IMU [9]. Phương pháp này mang lại một số ưu điểm đáng kể. Đầu tiên, nó làm cho robot trở nên thông minh hơn, cho phép robot di chuyển tự do đến bất kỳ vị trí trống trong nhà hàng và tiếp tục hành trình khi gặp vật cản mới bằng cách tìm con đường trống khác gần nhất. Thứ hai, hiệu suất của robot được cải thiện đáng kể, vì tốc độ di chuyển của robot có thể tăng giảm linh hoạt. Về mặt chi phí, ước tính giá thành của robot này xấp xỉ 500$. Tuy nhiên, các công bố nghiên cứu trước đây thường sử dụng theo hai hướng tiếp cận Particle

Received: 31 March 2025; Revised: 04 June 2025; Accepted: 04 June 2025.

filter [10] hoặc KF (Kalman Filter) [11] để thực hiện nhiệm vụ vẽ bản đồ và định vị. Hai phương pháp này chỉ hiệu quả khi diện tích nhà hàng nhỏ. Khi diện tích của nhà hàng lớn, chúng không thể tạo ra một bản đồ chính xác. Phương pháp Kalman Filter sử dụng phương pháp scan-to-scan matching [12] để tính toán sự thay đổi vị trí tương đối của robot so với bản đồ. Tuy nhiên, phương pháp scan-to-scan này có một hạn chế là việc cập nhật và tích lũy lỗi diễn ra nhanh chóng theo thời gian. Điều này có thể dẫn đến sự không chính xác trong việc xác định vị trí của robot sau một thời gian dài hoạt động. Để giải quyết vấn đề tích lũy lỗi theo thời gian, có hai phương pháp tiếp cận phổ biến là Particle filter và graph-based SLAM [13]. Tuy nhiên, đối với phương pháp tiếp cận Particle Filter, mỗi particle đều yêu cầu tính toán xác suất cho toàn bộ các vị trí trên bản đồ. Điều này đồng nghĩa với việc khi diện tích của môi trường tăng lên, khối lượng tính toán cũng tăng theo. Việc thực hiện khối lượng tính toán lớn, như yêu cầu của phương pháp tiếp cận Particle Filter, không phù hợp với hoạt động robot lễ tân - một loại robot di động có tài nguyên tính toán hạn chế. Áp dụng phương pháp này có thể gây quá tải, hiệu suất tính toán kém và thậm chí dẫn đến việc máy tính nhúng dừng hoạt động. Do đó, cần tìm các phương pháp tiếp cận khác phù hợp với tài nguyên hạn chế của robot lễ tân.



Hình 1: Sacarino robot thiết kế bởi nhóm nghiên cứu Zalama [7].

Hình 2: Hình ảnh của 1 robot phục vụ trong nhà hàng [14].

Trong bài báo này, chúng tôi đề xuất một thiết kế robot tiếp tân dành cho các nhà hàng. Để đảm bảo hiệu suất và độ chính xác trong việc tạo bản đồ và xác định vị trí trên chính bản đồ vừa tạo ra cho robot lễ tân với tài nguyên tính toán hạn chế, chúng tôi sử dụng phương pháp tiếp cận graph-based SLAM. Thuật toán sử dụng V-L marker cũng được đề xuất để robot có thể di chuyển về đúng vị trí sạc khi dung lượng pin gần hết. Cấu trúc các phần tiếp theo của bài báo như sau: phần 2 sẽ trình bày về phân tích yêu cầu, kiến trúc phần cứng và phần mềm của robot chạy bàn trong nhà hàng, phần 3 là các kết quả thực nghiệm, và phần 4 là kết luận.

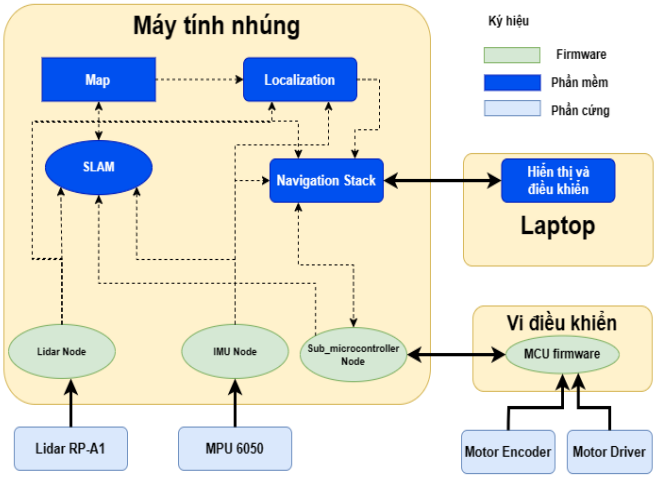
2. Thiết kế Robot chạy bàn phục vụ trong nhà hàng

2.1 Phân tích yêu cầu

Mục đích tạo ra robot chạy bàn để tự động quá trình phục vụ đồ ăn và tăng năng suất do robot có thể hoạt động liên tục không "mệt". Với nhà hàng truyền thống, khi khách vào chỗ ngồi sẽ có nhân viên tới nhận yêu cầu, sau đó vào báo cho đầu bếp chế biến rồi sau đó nhân viên sẽ mang đồ ăn tới bàn cho khách hàng. Với hệ thống nhà hàng được tự động hóa, khách khi vào nhà hàng có thể chọn món qua hệ thống máy tính bảng đặt tại các bàn, yêu cầu sẽ được chuyển tới đầu bếp qua hệ thống máy tính, sau khi chế biến xong, đầu bếp sẽ bấm gọi robot tới vị trí nhà bếp, đặt đồ ăn lên đó và chọn vị trí bàn của khách yêu cầu, robot sẽ nhận lệnh và mang đồ ăn tới tận bàn của khách. Các hệ thống nhà hàng như vậy đã trở thành những điểm mới lạ tại thành phố Thượng Hải - Trung Quốc. Dẫu vậy, ở những nhà hàng này, robot đang đi trên những đường dẫn từ dán sẵn trên sàn nhà. Hệ thống như vậy sẽ không có tính linh hoạt khi bố trí thêm bàn hoặc thay đổi bố trí bàn ăn do sẽ phải đi dán lại hệ thống đường dẫn từ dẫn tới mất thẩm mĩ. Do đó, chúng tôi nghiên cứu thiết kế robot dùng cảm biến lidar kết hợp cùng thuật toán SLAM vẽ bản đồ nhà hàng. Việc cập nhật lại bản đồ sẽ thuận tiện hơn, việc di chuyển của robot cũng linh hoạt hơn do có thể tự động tìm đường di chuyển khác khi trên đường đi có vật cản động.

2.2 Kiến trúc và tổng quan hệ thống của Robot dịch vụ

Trong phần này, chúng tôi sẽ trình bày thiết kế của robot lễ tân trong nhà hàng dựa trên các robot lễ tân đã được công bố trong phần 1. Robot này có các đặc điểm sau: trọng lượng 10 kg, khả năng tải tối đa 5 kg, được trang bị các cảm biến như LIDAR, gia tốc kế IMU, và encoder bánh xe; vận tốc tối đa 0.4 m/s; phương pháp điều hướng sử dụng là SLAM. Dựa trên các yêu cầu kỹ thuật vừa đề cập, chúng tôi đã thiết kế một sơ đồ khối chức năng để điều khiển hoạt động của robot, như được mô tả trong Hình 3.

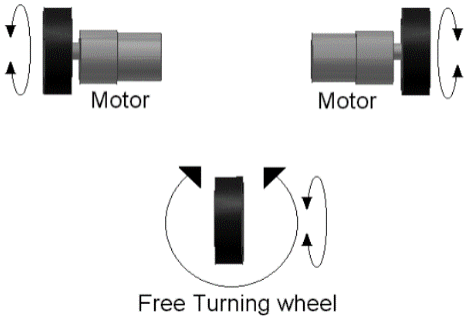


Hình 3: Sơ đồ khối chức năng của robot

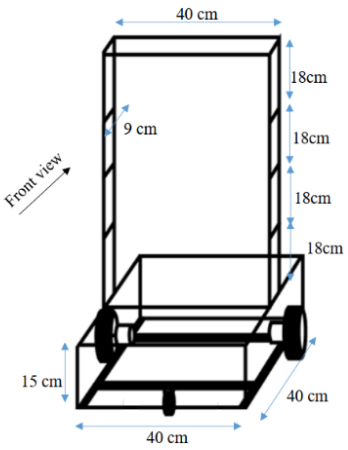
Kiến trúc điều khiển robot bao gồm các thành phần chính như SLAM (Simultaneous Localization and Mapping - định vị đồng thời và xây dựng bản đồ), kỹ thuật điều hướng và lập kế hoạch đường đi (Nav-igation).

2.3 Kiến trúc phần cứng

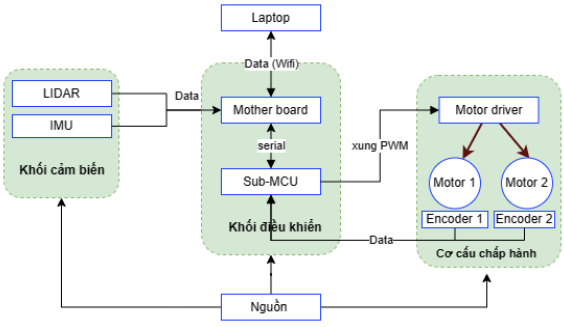
Hình 4 mô tả sơ đồ khối và mô hình phác thảo của robot. Đối với robot phục vụ trong nhà hàng hoạt động trên địa hình bằng phẳng, cơ cấu dẫn động được lựa chọn cần đơn giản, ổn định và đủ linh hoạt để di chuyển và xoay chuyển trong không gian hạn chế. Vì vậy, cơ cấu dẫn động vi sai [15] được coi là phù hợp với môi trường làm việc. Cơ cấu dẫn động này bao gồm hai bánh dẫn động được gắn trên một trục chung và một bánh đúc để đỡ xe và chống lật, như được mô tả trong hình 4(a). Mặc dù cơ cấu dẫn động này không thể di chuyển thẳng hoàn toàn, nhưng thay vào đó, nó có khả năng xoay tại chỗ để phù hợp với khu vực di chuyển hạn chế của nhà hàng. Vận tốc của từng bánh xe được điều khiển bởi một động cơ riêng biệt. Sau khi chọn được cơ cấu dẫn động cho robot, chúng tôi đã phác thảo khung robot nhìn từ phía trước, như được thể hiện trong hình 4(b). Hình phác thảo này ưa nhìn, tạo ra một sự thân thiện và hấp dẫn.



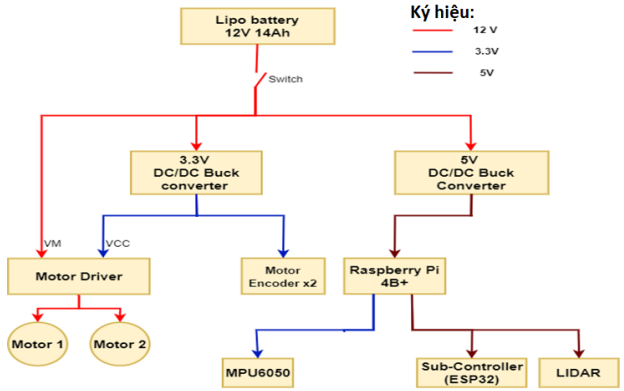
4a)



4b)



4c)



4d)

Hình 4: Sơ đồ khối robot (a) Cơ cấu dẫn động vi sai. (b) Hình phác thảo robot từ phía trước. (c) Hệ thống phần cứng. (d) Sơ đồ cấp nguồn cho thiết bị, cảm biến.

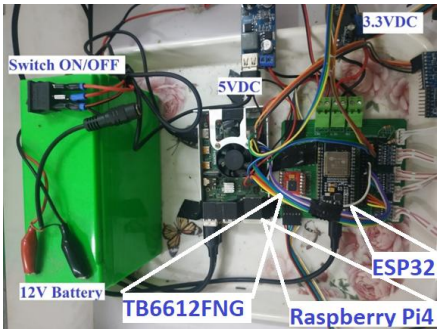
Phần cứng robot bao gồm các thành phần sau: 2 động cơ 1 chiều (12VDC); Pin Lipo 3Spack 12V/14Ah có khả năng cung cấp nguồn điện cho robot hoạt động từ 6 đến 8 giờ; LIDAR RP A1; cảm biến gia tốc kế MPU6050; Motor encoder sử dụng cảm biến Hall (480 xung mỗi vòng); Vi điều khiển Esp32 để điều khiển vận tốc 2 động cơ và máy tính nhúng Raspberry pi 4B.

Từ sơ đồ khối chức năng ở phần 2.1, chúng tôi quyết định thiết kế hệ thống phần cứng của robot bao gồm 5 khối chính: khối cảm biến, khối nguồn, khối cơ cấu chấp hành, Laptop và khối xử lý chính, được mô tả trong hình 4b). Chúng tôi chọn DC/DC Buck converter XY-3606 (12V-5V) để cung cấp mức điện áp 5VDC cho máy tính nhúng Raspberry Pi 4B, đáp ứng yêu cầu mức điện áp cần thiết và cường độ dòng điện tối đa là 3A. Ngoài ra, chúng tôi sử dụng modul DC/DC Buck converter LM2596 để chuyển đổi mức điện áp từ 12V xuống 3.3V, phục vụ cho hoạt động của các cảm biến và motor driver của robot, để giao tiếp với vi điều khiển và máy tính nhúng.

Vận tốc của robot được điều khiển bởi MCU ESP32. Mô-đun TB6612FNG được sử dụng để điều khiển hai động cơ DC thông qua tín hiệu PWM từ ESP32. ESP32 và Raspberry Pi 4B được kết nối trực tiếp với nhau thông qua một cáp USB/microUSB để thiết lập kết nối truyền tin nối tiếp (UART communication). ESP32 nhận tốc độ yêu cầu từ Raspberry Pi 4B và đọc dữ liệu từ encoder của 2 bánh xe để tính toán dữ liệu Odometry. Từ đó, dữ liệu Odometry được sử dụng để ước tính vị trí của robot. Tuy nhiên, trong quá trình di chuyển của robot, lốp có thể bị mòn và bánh xe có thể trượt, gây ra sai số tích lũy theo thời gian. Do đó, chúng tôi đã áp dụng một kênh dữ liệu khác để giảm thiểu các sai số này. Nhóm của chúng tôi đã sử dụng cảm biến MPU6050 để cung cấp dữ liệu về gia tốc tịnh tiến và gia tốc góc, từ đó tính toán dữ liệu Odometry. Trong quá trình làm việc, truyền thông giữa các thiết bị có thể xảy ra gián đoạn khi kết nối qua wifi. Để đảm bảo an toàn và đáng tin cậy, chúng tôi đã quyết định thực hiện mọi tính toán liên quan đến tốc độ đặt trên máy tính nhúng Raspberry Pi 4B. Điều này đảm bảo rằng robot hoạt động liên tục và không bị ảnh hưởng bởi sự cố khi kết nối wifi gặp lỗi không mong muốn. Máy tính cá nhân được kết nối với Raspberry Pi 4B qua wifi và sử dụng hệ điều hành ROS để quản lý toàn bộ dữ liệu vào ra của Raspberry Pi 4B. Ngoài ra, máy tính cá nhân

còn được sử dụng để hiển thị bản đồ trên màn hình và đưa ra các lệnh điều khiển thông qua công cụ trực quan 3D RVIZ.

Sau khi đã chọn các thành phần cho robot, chúng tôi đã kết hợp chúng trong một bảng mạch in để giảm số lượng dây và đảm bảo kết nối an toàn. Việc này giúp tối ưu hóa cấu trúc và giảm sự rối loạn của các dây nối, đồng thời tăng tính ổn định và đáng tin cậy của hệ thống. Sau quá trình kết hợp phần cứng và mạch điện, chúng tôi đã hoàn thiện robot theo thiết kế như hình 5.



5a)

5b)

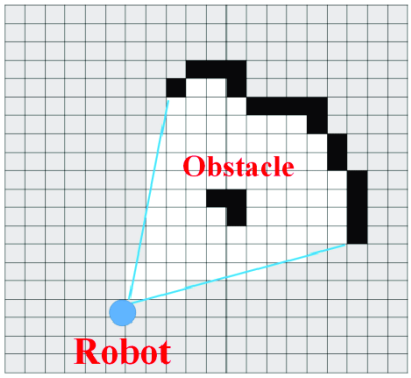
Hình 5: Phần cứng thực tế của robot (a) Mạch điện thực tế. (b) Hình ảnh robot hoàn thiện nhìn từ phía trước.

2.4 Kiến trúc phần mềm

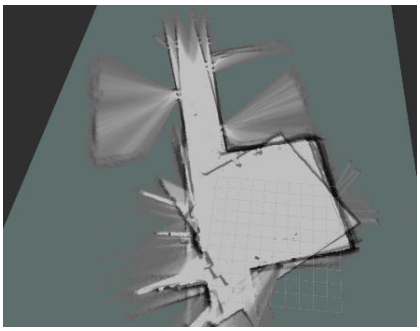
Kiến trúc của robot phục vụ được xây dựng trên nền tảng ROS (Robot Operating System). ROS là một framework được sử dụng rộng rãi trong ngành công nghiệp Robot, với sự đóng góp đa dạng từ cộng đồng phát triển lớn. Nó được coi là framework thành công nhất cho đến nay và cung cấp một loạt các công cụ và thư viện hữu ích cho các nhà phát triển robot để triển khai ý tưởng và ứng dụng của họ một cách hiệu quả.

Nhóm đề xuất phương pháp tiếp cận Graph-base SLAM cho việc xây dựng bản đồ và định vị đồng thời. Phương pháp này được coi là tiên tiến trong những năm gần đây và mang lại nhiều ưu điểm vượt trội. Khác với hai phương pháp tiếp cận khác, Particle filter hay extended Kaman filter [16], phương pháp Graph-base SLAM có thể được triển khai trên diện tích lớn với độ chính xác cao, đồng thời bản đồ thu được không bị sai lệch do tích lũy theo quỹ đạo. Để đạt được những ưu điểm này, phương pháp này sử dụng loop-closing để giải quyết vấn đề Drift, tức là khi robot quay trở lại một điểm trên quỹ đạo đã đi qua trước đó để loại bỏ sai lệch tích lũy dọc theo quỹ đạo.

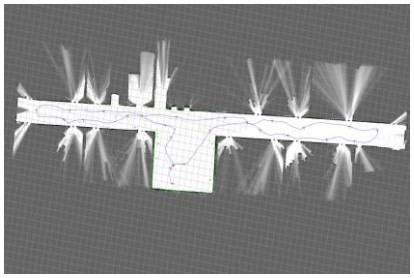
Dựa trên các cảm biến có độ chính xác tương đối: LIDAR RP A1, encoder cảm biến Hall với 16 xung mỗi vòng và gia tốc kế IMU 6050, nhóm đã xây dựng bản đồ dưới dạng grid map với độ phân giải r = 5 (cm). Bản đồ grid map cho phép chia không gian thành các ô nhỏ với kích thước đồng nhất, và mỗi ô trong bản đồ lưu trữ thông tin về trạng thái của một phần của môi trường, như có vật cản hay không. Độ phân giải r = 5 (cm) cho biết kích thước của mỗi ô trong bản đồ là 5 centimet.



Hình 6: Grid map [17]



a)



b)

Hình 7: (a) Bản đồ trong quá trình xây dựng có sai lệch tích lũy. (b) Bản đồ hoàn thiện.

Để kiểm tra tính chính xác của thuật toán SLAM nhóm chúng tôi đã thực nghiệm tại hàng lang tầng 2 toà nhà D8 Đại học Bách khoa Hà Nội, diện tích xấp xỉ 200m2 và quỹ đạo di chuyển có thể lên đến 400m và thu được kết quả như hình 7. Để thu được bản đồ hoàn thiện chính xác như hình 7(b) trên chúng tôi đã triển khai quá trình SLAM với hướng tiếp cận graph-base SLAM và sử dụng thuật toán Cartographer được đề xuất bởi các nhà nghiên cứu SLAM của Google [18]. Trong quá trình xây dựng bản đồ với hướng tiếp cận Graph-base SLAM, quá trình này bao gồm hai phần chính là frontend và

backend. Trong quá trình frontend, dữ liệu từ các cảm biến (như LIDAR) được thu thập và xử lý để tạo ra vị trí của robot và các bản đồ con, được gọi là submap. Submap là một bản đồ lưới 2D với kích thước ô cố định và độ phân giải lớn hơn so với sai số tạo ra bởi cảm biến. Các ô trong submap được cập nhật dựa trên dữ liệu "hits" (điểm phản hồi đo được từ cảm biến) và "misses" (khoảng không gian trống giữa LIDAR và các điểm phản hồi). Trong quá trình backend, các đặc trưng được trích xuất từ frontend để xây dựng bản đồ và cải thiện ước lượng vị trí chính xác hơn cho robot. Trong quá trình quét của LIDAR, sau khi không cập nhật submap mới, frontend sẽ kiểm tra xem vị trí quét hiện tại có trùng khớp với submap đã từng quét trước đó hay không. Nếu vị trí quét hiện tại tương ứng với một vị trí đã được robot đi qua trong quá khứ, điều này được gọi là phát hiện loop closure. Khi loop closure được phát hiện, frontend thực hiện quá trình Graph Op-timization [19]. Chúng tôi đã sử dụng công cụ Ceres để điều chỉnh lại vị trí của các submap dựa trên các ràng buộc đã được xây dựng giữa chúng trong quá trình frontend. Quá trình này giúp cải thiện độ chính xác vị trí và liên kết giữa các submap trong bản đồ.

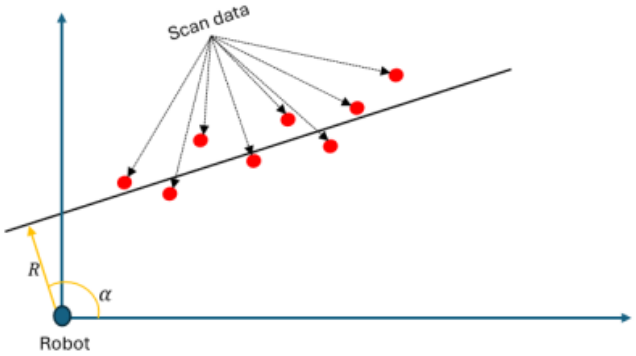
2.5 Tính năng tránh vật cản

Hình 8: Robot tránh vật cản khi hoạt động

Phần mềm của robot có tính năng tránh vật cản. Trong bài kiểm tra này, một thành viên trong nhóm đứng yên giữa con đường dẫn đến mục tiêu. Lidar có thể phát hiện được sự hiện diện của anh ấy từ xa, vì vậy một đường đi đã được lập kế hoạch đi vòng quanh anh ấy từ đầu để tránh vật cản. Hình 8 thể hiện cụ thể quá trình đó trong thực tế. Ở kịch bản vật cản động, trong quá trình robot điều hướng đến đích, một thành viên của nhóm sẽ cắt ngang qua đường đi hiện tại của nó. Lidar có thể phát hiện sự hiện diện của anh ấy, vì vậy một đường đi đã được lập kế hoạch đi vòng quanh anh ấy khi anh ấy đi ngang qua đường đi được lập kế hoạch sẵn của robot.

2.6 Tính năng tự động tìm về vị trí trạm sạc

Việc để robot hoạt động hết pin và dừng giữa đường sẽ khiến cho nhà hàng mất công sức và thời gian để kéo robot về nơi sạc pin. Do đó, nhóm nghiên cứu đề xuất thuật toán để robot tìm về điểm sạc tự động khi dung lượng pin thấp. Khi dung lượng pin dưới ngưỡng 20%, robot sẽ thực hiện nốt công việc vận chuyển đang làm (nếu có), sau đó robot sẽ tự động tìm về vị trí của trạm sạc. Ví trí trạm sạc sẽ được xác định trên bản đồ và lưu lại trong bộ nhớ của robot theo hai cách: 1) đặt robot vào vị trí sạc để lấy mẫu vị trí rồi lưu lại hoặc 2) đánh dấu bằng người vận hành bằng đánh dấu trên màn hình giao diện hiển thị bản đồ. Do mục tiêu hướng đến là robot có thể tự động vào kết nối với bộ sạc mà không cần can thiệp của con người, nên robot sẽ cần di chuyển vào chính xác vị trí. Dẫu vậy, việc di chuyển bằng bản đồ SLAM có sai số ngoài phụ thuộc vào chất lượng cảm biến LIDAR, còn có độ dung sai vị trí của thuật toán điều khiển điều hướng. Nghĩa là robot sẽ không đến đúng chính xác 100% vị trí tọa độ và hướng khi di chuyển bằng bản đồ, mà sẽ đến vị trí nằm trong khoảng của vị trí được yêu cầu cộng trừ dung sai nhỏ của tọa độ và hướng. Vì vậy, để điều khiển robot và đúng chính xác vị trí, nhóm đề xuất sử dụng V-L marker.



Hình 9: Đường thẳng dự đoán với tập điểm quét được bởi Lidar

Gói Laser line extraction là một gói mã nguồn mở trong hệ điều hành robot (Robot Operating System - ROS) do Marc Gallant viết. Gói này sẽ lấy dữ liệu từ tin nhắn LaserScan và xuất ra những đoạn đường thẳng có trong vùng quét của LI-DAR. Dựa trên thuật toán được đưa ra trong bài báo, gói Laser line extraction sẽ ước lượng xem một tập hợp các điểm dữ liệu do LIDAR trả về có phải là một đường thẳng hay không. Mỗi điểm sẽ được tính trọng số về khoảng cách và trọng số về góc, và tính hiệp phương sai vị trí của chúng so với đường thẳng ước tính. Từ những thông số trên của từng điểm do LI-DAR phát hiện ra, ta có thể ước lượng rằng tập hợp điểm đang được xét tới có phải là một đường thẳng trong không gian mà robot đang đứng hay không. Nếu như phát hiện ra được đường thẳng, gói sẽ cho ta đầu ra là khoảng cách R và góc lệch α của nó với vị trí của robot như hình 9. Ứng với từng không gian và ứng dụng khác nhau, trong gói Laser line extrac-tion cũng sẽ có những điều kiện khởi tạo ban đầu mà ta có thể điều chỉnh để kết quả có độ chính xác tốt nhất như:

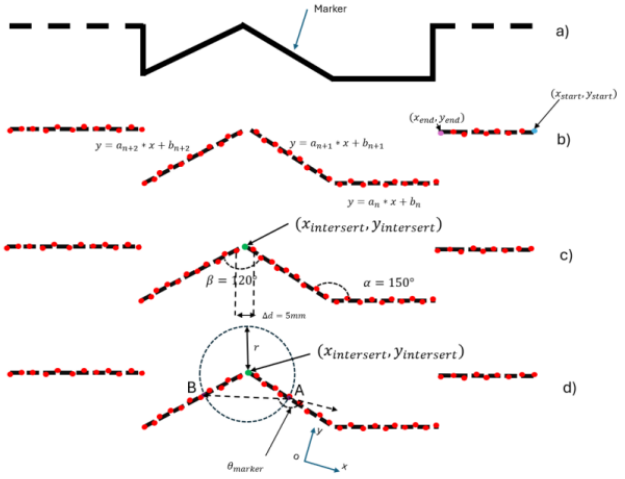
- **min_line_length**: độ dài tối thiểu mà gói sẽ coi đó là một đoạn thẳng.
- **min_line_point**: số lượng điểm tối thiểu của một đoạn thẳng.
- **max_line_gap**: khoảng cách tối đa giữa hai điểm liên tiếp trong đoạn thẳng.
- **min_range và max_range**: khoảng cách tối thiểu và tối đa giữa các điểm và robot mà gói sẽ quan tâm tới (ngoài khoảng này gói sẽ bỏ qua).

Dựa vào gói Laser line extraction, ta đề xuất một hướng làm mới để kết nối robot với trạm sạc mà không cần các trạm thu phát hồng ngoại, siêu âm hoặc sử dụng những loại camera, LIDAR đắt tiền. Ta sẽ thiết kế trạm sạc có hình dáng đặc biệt so với vật thể trong không gian làm việc của robot. Trong nghiên cứu này, trạm sạc được thiết kế có hình dạng V-L sẽ được sử dụng để robot dễ dàng nhận dạng để kết nối. Quá trình chỉ dẫn robot về trạm sạc khi robot có dấu hiệu hết pin sẽ gồm hai giai đoạn chính. Giai đoạn thứ nhất, ta sẽ sử dụng khả năng điều hướng của robot để nó có thể đi về vị trí ta đặt trạm sạc. Quá trình này sẽ có sai số do khả năng điều hướng

tới đích của robot không thể chính xác hoàn toàn. Chính vì vậy ta sẽ cần giai đoạn thứ hai để tăng độ chính xác khi kết nối robot. Khi robot đã tới được vị trí đặt trạm sạc, ta sẽ bắt đầu chạy gói Laser line extraction để phát hiện được những đoạn thẳng ở trong khu vực này.

Với những điều kiện ban đầu ứng với hình dáng V-L của trạm sạc, ta sẽ xác định được vị trí hiện tại của trạm sạc trong không gian robot đang hoạt động. Cụ thể hơn, trong nghiên cứu này, trạm sạc sẽ được thiết kế hình V-L với góc α = 120°, góc β = 150° và độ dài mỗi cạnh là 40cm như hình 10.

Quá trình xác định vị trí của trạm sạc cũng được thể hiện trong hình 11:



Hình 10: Quá trình xác định góc lệch giữa robot và trạm sạc

Với dữ liệu có được từ LIDAR, ta sẽ có được những đoạn thẳng như hình 10.b khi chạy gói Laser line extraction. Với mỗi đoạn thẳng, ta sẽ quan tâm tới tọa độ của điểm đầu ($xstart, ystart$) cũng như tọa độ của điểm cuối ($xend, yend$) thuộc đường thẳng tìm được. Lúc này ta sẽ có hai điều kiện để xác định xem các đường thẳng do robot tìm thấy có phải là trạm sạc không như sau:

- Khoảng cách giữa điểm cuối của đường thẳng thứ i và đường thẳng thứ i+1 thỏa mãn: $\Delta d \leq 5mm$.
- Góc tạo bởi đường thẳng i và i+1 là α = 150° và góc tạo bởi đường thẳng i+1 và i+2 là β = 120°.

Với tọa độ của điểm đầu và điểm cuối ta sẽ tìm được phương trình đường thẳng cũng như vecto chỉ phương của từng đường thẳng. Dựa vào vecto chỉ phương, ta có thể tính được góc tạo bởi hai đường thẳng dựa trên mối quan hệ giữa tích vô hướng và tích độ dài của hai vecto. Sau khi đã xác định được vị trí ba đường thẳng thỏa mãn hai điều kiện trên, ta sẽ cần tìm tọa độ của trạm sạc. Tọa độ của trạm sạc sẽ được đại diện bởi giao điểm của đường thẳng thứ i+1 và đường thẳng thứ i+2. Do đã biết được phương trình đường thẳng của chúng, ta có thể dễ dàng tìm được giao điểm $ainter$ đó dựa trên phương trình hoành độ giao điểm.

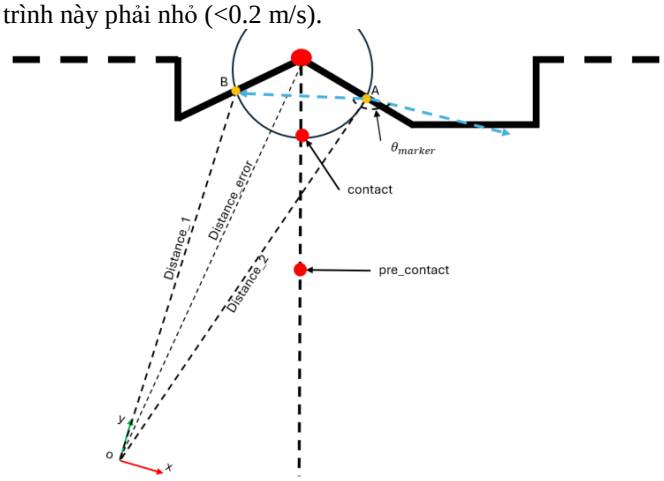
Cuối cùng, ta sẽ cần xác định được vị trí tương đối giữa robot và trạm sạc để có thể đưa ra những lệnh điều khiển cho phù hợp. Để làm được điều đó, ta cần tính khoảng cách giữa robot và trạm sạc và góc lệch giữa robot và trạm sạc. Ta cần chú ý một điều là tọa độ của các điểm sẽ được tính trong hệ trục tọa độ với gốc tọa độ chính là ví trí của cảm biến LI-DAR. Do đó, khoảng cách giữa robot và trạm sạc chính là khoảng cách giữa $ainter$ với gốc tọa độ. Để tính được góc lệch giữa robot và trạm sạc, ta sẽ cần thực hiện các bước sau:

- Bước 1: Vẽ đường tròn tâm $ainter$ có bán kính r (r = 5cm trong thực nghiệm).
- Bước 2: Tìm hai giao điểm A và B của đường tròn trên với đường thẳng thứ i+1 và i+2.
- Bước 3: Tính góc $\theta marker$ tạo bởi vecto AB và trục Ox.

Góc lệch giữa robot và trạm sạc lúc này chính là $\theta marker$. Bên cạnh đó ta sẽ tính thêm khoảng cách giữa hai điểm A và B tới robot. Với đầu ra của gói Laser line extraction, ta đã xác định được vị trí trạm sạc, khoảng cách giữa trạm sạc tới robot và góc lệch giữa trạm với robot. Việc kết nối trạm sạc, sẽ được chia ra thành ba giai đoạn dựa trên các giá trị $\theta marker$, $Distance1$, $Distance2$, $Distanceerror$:

- Giai đoạn 1: Điều khiển robot về vị trí $precontact$. Ta sẽ so sánh $Distance1$ và $Distance2$ để biết được robot đang ở phía bên trái hay phía bên phải của trạm sạc, từ đó sẽ đưa ra lệnh điều khiển robot cho phù hợp.
- Giai đoạn 2: Căn chỉnh robot và đi về vị trí $contact$. Lúc này ta sẽ dựa vào $\theta marker$ để điều khiển robot sao cho góc lệch giữa robot và trạm sạc trong khoảng sai số cho phép.
- Giai đoạn 3: Khi robot đã ở vị trí $contact$, ta sẽ chỉ cần điều khiển robot đi thẳng về phía trước cho tới khi $Distanceerror$ đạt ngưỡng cho phép thì sẽ dừng lại.

Việc tính toán $\theta marker$, $Distance1$, $Distance2$, $Distanceerror$ sẽ diễn ra liên tục sau khi robot bắt đầu việc kết nối trạm sạc. Việc gửi tín hiệu điều khiển tới động cơ của robot phụ thuộc vào việc có quét được vị trí của trạm sạc hay không nên để giảm được sai số do không xác định được trạm sạc đang ở đâu, ta sẽ cần thiết lập tốc độ di chuyển trong quá trình này phải nhỏ (<0.2 m/s).



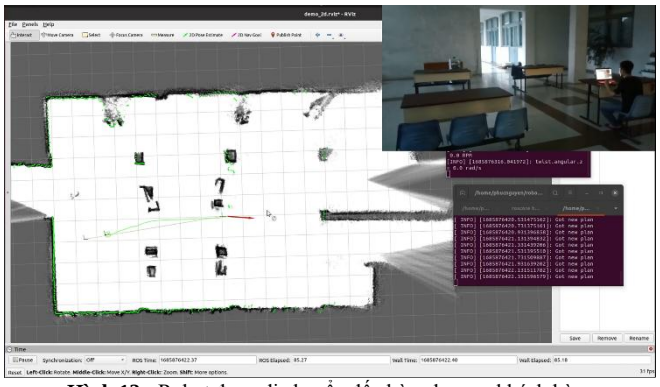
Hình 11: Cách tìm vị trí tương đối giữa Robot và trạm sạc

3. Kết quả thực nghiệm

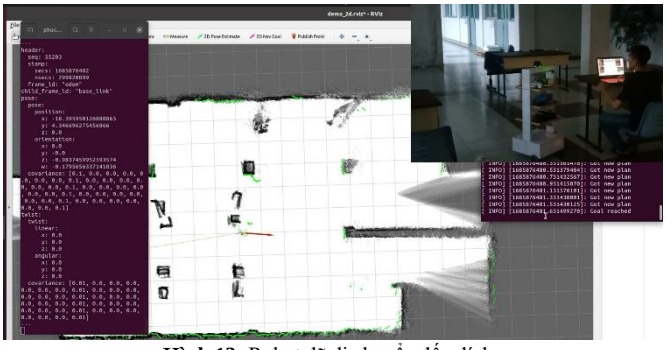
3.1 Thử nghiệm tính năng xây dựng bản đồ và di chuyển tự động

Để thử nghiệm khả năng hoạt động của robot lễ tân trong một nhà hàng, nhóm chúng tôi đã thiết lập một môi trường nhà hàng giả định và tiến hành thử nghiệm tại tầng 3 của tòa nhà C1 trong khuôn viên Đại học Bách khoa Hà Nội. Trong thử nghiệm này, thành viên trong nhóm đảm nhận vai trò là khách hàng trong nhà hàng. Sử dụng bản đồ đã được vẽ trước đó kết hợp với dữ liệu quét từ LIDAR, thuật toán Localize sẽ tìm ra submap tốt nhất khớp với dữ liệu từ LIDAR, từ đó robot sẽ biết vị trí của mình trên bản đồ. Sau khi khách hàng gọi

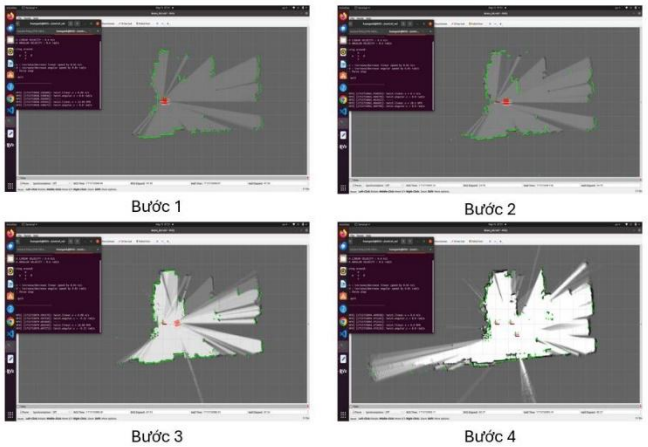
món, robot sẽ bắt đầu tìm đường đi đến bàn của khách hàng. Môi trường nhà hàng đặc trưng bởi sự hiện diện của nhiều vật cản như bàn, ghế, hộp, người đi lại, vì vậy thuật toán Dijkstra được lựa chọn là phương pháp tìm kiếm đường đi tối ưu hơn thuật toán A* trong môi trường này. Sau khi xác định được quỹ đạo ổn định, thuật toán Dynamic Window Approach sẽ được áp dụng để điều khiển robot bám sát quỹ đạo đã tính toán, đồng thời tránh các vật cản động. Với mục tiêu đảm bảo sự mượt mà trong việc di chuyển và an toàn cho khách hàng, chúng tôi đã giới hạn tốc độ di chuyển của robot là 0.1 m/s. Các thông số khác của thuật toán bám quỹ đạo và tránh vật cản đã được cấu hình dựa trên quá trình thực nghiệm và các công bố trước đây [20]. Hình 12 ghi lại quá trình robot đang bám theo đường đi màu xanh đã được tính toán, để đến đích ở vị trí được chỉ ra bởi mũi tên màu đỏ. Hình 13 thể hiện hình ảnh robot đã đến đích an toàn.



Hình 12: Robot đang di chuyển đến bàn phục vụ khách hàng



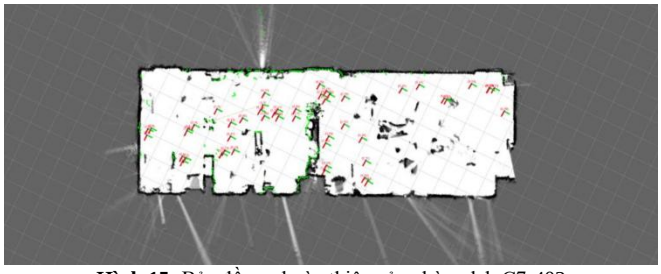
Hình 13: Robot đã di chuyển đến đích.



Hình 14: Quá trình xây dựng bản đồ trong môi trường thực tế

Phòng 402 toà C7 Đại học Bách khoa Hà Nội đã được lựa chọn làm nơi thực nghiệm thứ 2. Khu vực này có diện tích mặt sàn ước tính 120 m2. Hình 14 miêu tả quá trình xây dựng bản đồ như sau: Trong bước đầu tiên, robot bắt đầu di chuyển và bắt đầu quét các vùng xung quanh. Các điểm dữ liệu từ Li-dar được thu thập và hiển thị trên bản đồ dưới dạng các chấm xanh lá. Robot và cảm biến Lidar đang khởi động quá trình quét và bắt đầu xây dựng bản đồ từ các điểm dữ liệu ban đầu. Ở bước thứ hai, robot tiếp tục di chuyển và quét thêm nhiều vùng hơn. Bản đồ bắt đầu rõ ràng hơn khi các điểm dữ liệu mới được thêm vào. Robot tiếp tục thu thập dữ liệu, các chướng ngại vật và bức tường dần dần xuất hiện trên bản đồ. Điều này cho thấy quá trình quét và cập nhật bản đồ đang diễn ra suôn sẻ. Bước thứ ba robot tiếp tục di chuyển và thu thập dữ liệu từ nhiều góc độ khác nhau, cải thiện độ chính xác của bản đồ. Các khu vực không được quét trước đó bắt đầu hiện ra. Ở bước cuối các điểm dữ liệu cuối cùng được thu thập và thêm vào bản đồ, hoàn thiện quá trình quét. Bản đồ bây giờ đã đủ chi tiết để robot sử dụng cho việc điều hướng và tránh chướng ngại vật và bao gồm ba bản đồ con.

Trong quá trình thực nghiệm, khi vận tốc góc của robot thay đổi nhanh, đã tạo ra sai lệch về ước tính vị trí các quét của lidar. Sau khi robot đi qua vị trí ban đầu và hoàn tất quỹ đạo, thuật toán phát hiện được loop closure và điều chỉnh lại vị trí của các bản đồ con. Bản đồ hoàn chỉnh được hình thành từ 48 bản đồ con. Hình 15 thể hiện di chuyển của ro-bot để tái tạo lại bản đồ của môi trường xung quanh. Bản đồ trong hình đã hoàn thiện với độ phân giải 5cm, thể hiện rõ các chi tiết của khu vực được quét. Độ phân giải 5cm nghĩa là mỗi ô trên bản đồ đại diện cho một diện tích 5cm x 5cm trong thực tế. Điều này cho phép hiển thị các chi tiết nhỏ và cung cấp một bản đồ chính xác hơn.

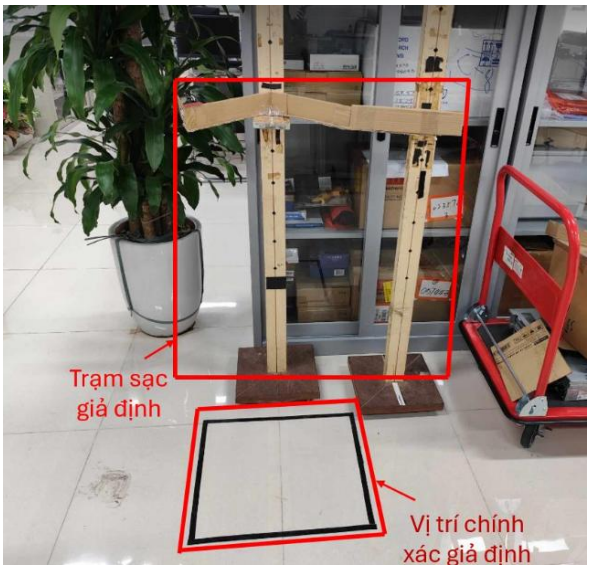


Hình 15: Bản đồ sau hoàn thiện của phòng lab C7-402

3.2 Thử nghiệm tính năng tự động về trạm sạc

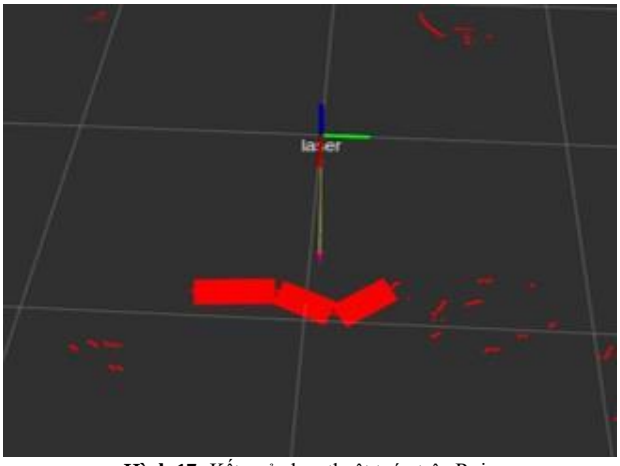
Quá trình thử nghiệm cho việc kết nối trạm sạc sẽ được tiến hành sau khi robot đã được điều hướng về vị trí đặt trạm sạc. Hình 16 mô tả việc bố trí thử nghiệm để đánh giá khả năng kết nối của robot theo phương pháp được đề xuất.

Trạm sạc được tượng trưng bởi một mảnh bìa với hình dạng V-L và đặt cách mặt đất $1m$. Chiều dài của các cạnh là $40cm$, góc tạo bởi các cạnh lần lượt là α = 150° và β = 120°. Phía trước trạm sạc mô phỏng này, có một hình chữ nhật với kích thước $48.4cm$ x $40cm$. Đây là vị trí mà robot cần nằm gọn bên trong sau khi quá trình kết nối trạm sạc kết thúc để được coi là một lần kết nối thành công. Điều này tương đương với sai số đặt của việc kết nối là $\Delta d \leq 4cm$.



Hình 16: Trạm sạc giả định

Ta sẽ đánh giá phạm vi mà gói Laser line extraction hoạt động tốt nhất. Việc đánh giá này sẽ được dựa trên phần mềm mô phỏng Rviz và vị trí đặt của robot. Nó sẽ được đặt tại các vị trí bất kì và kiểm tra xem, đâu là vị trí mà ở đó ba đoạn thẳng tạo thành trạm sạc được thể hiện rõ ràng trên Rviz.

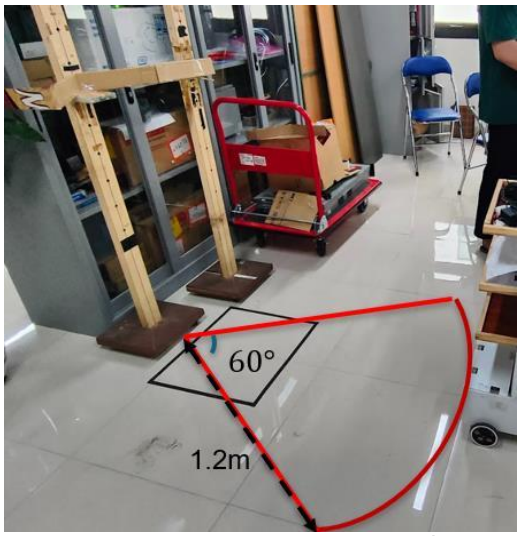


Hình 17: Kết quả chạy thuật toán trên Rviz

Như trên hình 17, ta có thể thấy được những đoạn thẳng mà gói Laser line extraction tìm thấy thông qua Rviz. Những đoạn thẳng tạo thành trạm sạc không phải lúc nào cũng được "nhìn thấy" bởi robot do robot ở càng xa trạm sạc, sai số sẽ càng lớn và số điểm của trạm sạc được phát hiện bởi LIDAR sẽ càng ít dẫn tới gói Laser line extraction không thể phát hiện được đoạn thẳng cấu tạo lên trạm sạc. Kết quả sau nhiều lần thử nghiệm, phạm vi mà khả năng "nhìn thấy" trạm sạc của robot tốt nhất là hình quạt bới bán kính $R = 1.2m$, góc mở $\delta = 60°$ như hình 18.

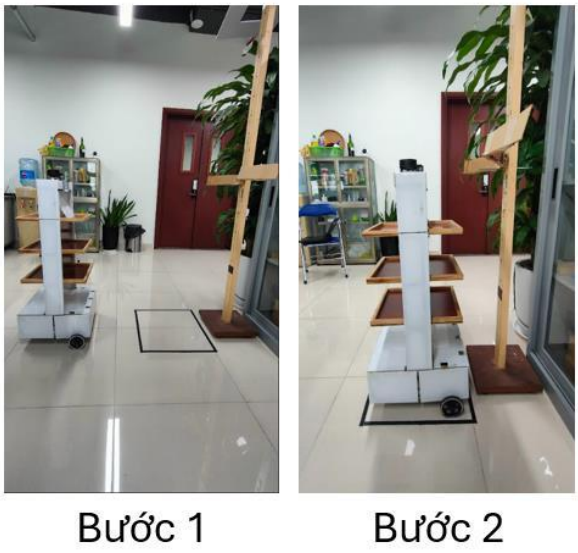
Tiếp theo ta xây dựng để thử nghiệm việc kết nối sau khi robot được điều hướng về vị trí đặt trạm sạc thì nó đã đối diện với trạm sạc. Ta sẽ thiết lập để robot kết nối với trạm sạc khi ở đối diện và cách trạm sạc $1.2m$. Lúc này quá trình kết nối sẽ đơn giản hơn do robot chỉ cần căn chỉnh để góc lệch là thỏa mãn điều kiện được đặt ra và đi thẳng về phía trạm sạc. Hai bước kết nối của trạm sạc sẽ là:

- Bước 1: Robot ở vị trí đối diện trạm sạc sẽ đi thẳng và căn chỉnh góc lệch liên tục để kết nối.
- Bước 2: Robot kết nối trạm sạc thành công.



Hình 18: Phạm vi thuật toán hoạt động tốt

Nhóm thử nghiệm trong 100 lần để đánh giá sai số trung bình và tỉ lệ thành công của robot. Kết quả thu được, sai số khoảng cách trung bình là $0.845cm$, sai số góc trung bình là 1° và tỉ lệ thành công là 100%. Hình ảnh thử nghiệm được thể hiện ở hình 19.



Hình 19: Thử nghiệm với kịch bản đi vào chỗ sạc

Sau đó, ta tiếp tục xây dựng để đánh giá khả năng kết nối trạm sạc của robot khi bắt đầu kết nối, vị trí của robot bị lệch về một phía của trạm sạc. Lúc này quá trình kết nối trạm sạc của robot sẽ có thêm một bước là di chuyển robot về vị trí đối diện trạm sạc. Quá trình thử nghiệm thể hiện ở hình 20 sẽ bao gồm các bước như sau:

- Bước 1: Đặt robot lệch so với trạm sạc.
- Bước 2: Robot căn chỉnh để về vị trí đối diện trạm sạc.
- Bước 3: Robot tiến lại gần trạm sạc và liên tục căn chỉnh góc lệch.
- Bước 4: Robot kết nối thành công khi nằm trong khung hình chữ nhật.

Thử nghiệm được tiến hành 100 lần để đánh giá sai số. Kết quả thu được là sai số khoảng cách trung bình là $1.9cm$, sai số góc trung bình là 1.95° và tỉ lệ thành công là 95%.

Hình 20: Các bước thử nghiệm với kịch bản 3

Với thử nghiệm như trên, quy trình robot tìm về trạm sạc sẽ như sau:

- Robot hoạt động tới khi hệ thống báo pin yếu (dưới 20%).
- Robot sẽ đi tự động đi về một vị trí gần trạm sạc theo thuật toán di chuyển tự động dùng SLAM (SLAM navigation), vị trí này sẽ được người dùng chọn một cách tương đối trên bản đồ sau khi bản đồ đã được xây dựng xong, gần với vị trí sạc.
- Sau khi đến vị trị đó, robot chuyển qua chế độ di chuyển vào vị trí trạm sạc theo thuật toán sử dụng V-L marker để đạt độ chính xác cao.

Về mặt tốc độ tính toán, vì sử dụng máy tính nhúng Pi, nếu chạy bằng thuật toán cartographer gốc sẽ tốn nhiều thời gian xử lý (200ms - 500ms) cho mỗi bản tin quét của LIDAR khi chạy ở chế độ định vị sau khi có bản đồ. Lí do là vì thuật toán gốc lưu lại thông tin của nhiều bản đồ con và tạo tiến trình chạy tính định vị song song trên nhiều bản đồ con đó. Để tối ưu tốc độ xử lý, sau khi xây dựng được bản đồ, nhóm sẽ phát triển chương trình để ghép các bản đồ con thành một bản đồ lớn cho toàn môi trường. Do đó khi chạy định vị trong bản đồ, số lượng tiến trình song song luôn cố định ở 3 tiến trình giúp cho máy tính nhúng Pi4 có thể chạy với thời gian 70ms cho mỗi bản tin LIDAR.

4. Kết luận

Với những thử nghiệm thực tế đã được trình bày bên trên, ta có thể thấy được rằng robot đã đạt được những kết quả theo yêu cầu ban đầu đề ra về việc phát triển robot tiếp tân phục vụ trong nhà hàng với chi phí thấp. Robot đã có thể vượt qua các thử nghiệm được đề ra. Việc chỉ sử dụng loại cảm biến LIDAR nhưng quá trình tránh vật cản, xây dựng bản đồ cũng như kết nối trạm sạc đạt kết quả khá tốt, với sai số khi kết nối trạm sạc từ $0.845 \div 1.9\ cm$, sai số khoảng cách và sai số về góc từ $1 \div 1.95°$. Tuy nhiên, từ kết quả của thử nghiệm, ta cũng thấy được nguyên nhân gây ra lỗi trong quá trình vận hành chủ yếu do bánh xe bị trượt trong quá trình di chuyển, sai số của cảm biến LIDAR.

Tài liệu tham khảo

[1] S. R.-G. F. &. A.-S. R. Molinillo, "Exploring the antecedents of customers' willingness to use service robots in restaurants," *Service Business*, pp. 17(1), 167-193, 2023.

[2] V. S. O. V. Y. a. A. P. F. McLeay, "Replaced by a Robot: Service Implications in the Age of the Machine," *J. Serv. Res.*, vol. vol. 24, no. no. 1, p. pp. 104–121, 2021.

[3] O. C. a. C. C. K. Berezina, "Robots, Artificial Intelligence, and Service Automation in Restaurants," *Robots, Artificial Intelligence, and Service Automation in Travel, Tourism and Hospitality*, vol. Emerald Publishing Limited, pp. pp. 185-219, 2019.

[4] O. &. A. H. S. El-Said, "Are customers happy with robot service? Investigating satisfaction with robot service restaurants during the COVID-19 pandemic," *Heliyon*, vol. 8(3), 2022.

[5] M. P. a. M. M. Sanaatiyan, "Design and Implementation of Line Follower Robot," in *2009 Second International Conference on Computer and Electrical Engineering*, Oct. 2009.

[6] A.-A.-N. a. A. A. M. K. M. Hasan, "Implementation of autonomous line follower robot," in *International Conference on Informatics, Electronics & Vision (ICIEV)*, pp. 865–869, May 2012.

[7] E. Z. e. al., "Sacarino, a Service Robot in a Hotel Environment," in Sacarino, a service robot in a hotel environment. In ROBOT2013: First Iberian Robotics Conference: Advances in Robotics, , Vol. 2 (pp. 3-14). Springer International Publishing., 2013.

[8] H. D.-W. a. T. Bailey, "Simultaneous localization and mapping: part I,," *IEEE Robot. Autom. Mag.*, vol. vol. 13, no. no. 2, p. pp. 99–110, Jun. 2006.

[9] Z. L. J. H. Y. D. a. H. Z. J. Cheng, "Application of Simultaneous Location and Map Construction Algorithms Based on Lidar in the Intelligent Robot Food Runner," *J. Phys. Conf. Ser.*, vol. vol. 1972, no. no. 1, p. p. 012010, Jul. 2021.

[10] C. S. a. W. B. G. Grisetti, "Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation,*, pp. 2432–2437, Apr. 2005.

[11] M. P. eBooks, Principles of Robot Motion: Theory, Algorithms, and Implementations, Apr. 17: IEEE Xplore, 2023.

[12] E. Olson, "M3RSM: Many-to-many multi-resolution scan matching," in *IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA, May 2015, pp. 5815–5821. .

[13] R. K. C. S. a. W. B. G. Grisetti, "A Tutorial on Graph-Based SLAM," *IEEE Intell. Transp. Syst. Mag.*, vol. vol. 2, no. no. 4, p. pp. 31–43, 2010.

[14] K. M. S. P. O. A. &. K. M. S. Kapoor, "Design of Restaurant Service Robot for Contact less and Hygienic Eating Experience," 2020.

[15] Y. Sugahara, "On the Drive System of Robots Using a Differential Mechanism," in *IFToMM Asian conference on Mechanism and Machine Science (pp. 10-21)*, Cham: Springer International Publishing, 2021, December.

[16] P. Z. Z. S. S. Z. X. &. T. M. Shi, "Invariant extended Kalman filtering for tightly coupled LiDAR-inertial odometry and mapping," *IEEE/ASME Transactions on Mechatronics*, Vols. 28(4), 2213-2224., 2023.

[17] S. J. C. Y. Nam TH, "A 2.5D Map-Based Mobile Robot Localization via Cooperation of Aerial and Ground Robots," *Sensors (Basel)*, vol. PMID: 29186843, no. PMCID: PMC5750701, p. 17(12):2730, 2017 Nov.

[18] D. K. H. R. a. D. A. W. Hess, "Real-Time Loop Closure in 2D LIDAR SLAM," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1271–1278.

[19] R. a. G. G. a. S. H. a. K. K. a. B. W. Kümmerle, "G2o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*, 3607-3613, 2011.

[20] K. Zheng, "Ros navigation tuning guide," *Robot Operating System (ROS) The Complete Reference*, vol. Volume 6, pp. 197-226, 2021.