(12) **BẢN MÔ TẢ GIẢI PHÁP HỮU ÍCH THUỘC BẰNG ĐỘC QUYỀN GIẢI PHÁP HỮU ÍCH**

(19) **Cộng hòa xã hội chủ nghĩa Việt Nam (VN)**
CỤC SỞ HỮU TRÍ TUỆ

(11) **2-0004226**

(51) **G01B 11/00**; G06T 7/60; G06N 3/00
2024.01

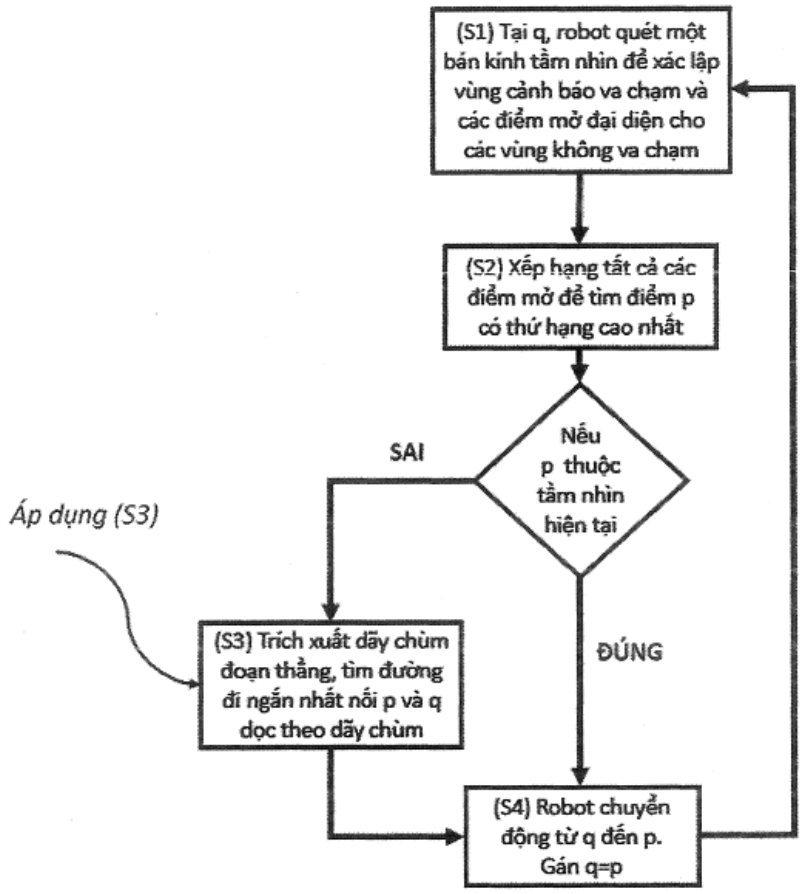
(13) **Y**

(21) 2-2024-00798
(22) 02/12/2024
(45) 25/07/2025 448
(43) 25/02/2025 443A1
(73) Trường Đại học Bách Khoa - Đại học Quốc gia Thành phố Hồ Chí Minh (VN)
268 Lý Thường Kiệt, Phường 14, Quận 10, Thành phố Hồ Chí Minh
(72) Phan Thành An (VN); Trần Văn Hoài (VN); Phạm Hoàng Anh (VN); Trần Thanh Bình (VN); Nguyễn Thị Lê (VN).

(54) PHƯƠNG PHÁP TÌM ĐƯỜNG QUAY LẠI CHO ROBOT TỰ HÀNH DỌC THEO DÃY CÁC CHÙM ĐOẠN THẲNG ĐÃ XÁC ĐỊNH TRƯỚC

(21) 2-2024-00798

(57) Giải pháp hữu ích đề xuất phương pháp tìm đường quay lại cho robot tự hành dọc theo dãy các chùm đoạn thẳng đã xác định trước. Ba yếu tố của phương pháp bao gồm phân chia chùm, điều kiện thẳng hàng, và cập nhật các điểm bắn, trong đó nếu điều kiện thẳng hàng được thỏa mãn, đường đi ngắn nhất chính xác của bài toán sẽ được xác định; ngược lại, độ dài của các dãy đường đi tìm được qua quá trình cập nhật của phương pháp sẽ hội tụ dần. Việc trích xuất dãy chùm đoạn thẳng và sử dụng phương pháp bắn nhiều lần MMS (Method of Multiple Shooting); cho việc tìm đường đi dọc theo dãy chùm đoạn thẳng, bao gồm: bước (S3.1), robot xác định được điểm *p* có thứ hạng cao nhất cần trở lại mà *p* không nằm trong tầm nhìn hiện tại, việc trích xuất được các dãy chùm đoạn thẳng thuộc phần không gian robot sẽ trở lại được tiến hành; đường xương cá và dãy chùm đoạn thẳng gắn với đường xương cá được tạo ra, tại mỗi đỉnh, chỉ cần xét chùm đoạn thẳng nằm trong cung tròn có tâm tại đỉnh chùm và góc của cung không vượt quá 180^0; bước (S3.2), tối ưu đường quay lại bằng cách tìm đường đi ngắn nhất hình học dọc theo một dãy các chùm đoạn thẳng bằng phương pháp bắn nhiều lần MMS; bước (S3.3), robot di chuyển theo đường ngắn hơn đã tìm được ở bước trên.



Hình 1

Lĩnh vực kỹ thuật được đề cập

Giải pháp hữu ích thuộc lĩnh vực Tự động hóa và Toán ứng dụng. Trong quá trình xác định các hướng đi ưu tiên, một robot có tầm nhìn chính xác bị hạn chế có thể cần quay lại một số vị trí đã được đánh dấu trước đó, các vùng mà robot trở lại được mô hình hóa dưới dạng dãy các chùm các đoạn thẳng. Giải pháp hữu ích đề xuất phương pháp tìm đường quay lại dọc theo dãy các chùm đoạn thẳng nhằm giảm thiểu độ dài của các đường được robot đi qua.

Tình trạng kỹ thuật của giải pháp hữu ích

Bài toán robot tự hành tránh các vật cản đa giác trong môi trường không xác định là tìm một con đường từ điểm ban đầu đến điểm mục tiêu mà tránh được chướng ngại vật. Ngoài mục tiêu chạm đến đích, việc tiết kiệm thời gian và giảm thiểu độ dài của quãng đường chuyển động cũng rất quan trọng.

Xung quanh vấn đề tìm đường trở lại này có các tài liệu sáng chế liên quan sau:

(1) Sáng chế số CN116483083A "*Shortest path planning method for automatic cleaning robot*" (link: https://patents.google.com/patent/CN116483083A/en?oq=CN116483083A), sáng chế này đề xuất ra phương pháp lập kế hoạch đường đi ngắn nhất của robot vệ sinh tự động, bao gồm các bước xây dựng bản đồ môi trường thành các hình đa giác, là ranh giới của một khu vực vệ sinh, tìm đường đi ngắn nhất từ điểm bắt đầu đến điểm kết thúc bằng cách đưa ra một nhóm đa giác được xác định bởi đỉnh, điểm bắt đầu và điểm kết thúc, và tìm đường đi ngắn nhất từ điểm bắt đầu đến điểm kết thúc mà không đi qua bất kỳ đa giác nào.

(2) Sáng chế số CN113867353A "*Automatic return path planning method of cleaning robot*" (link: https://patents.google.com/patent/CN113867353A/en?oq=CN113867353A) Sáng chế này đề xuất ra phương pháp lập kế hoạch đường trở về tự động bao gồm các bước sau: lập kế hoạch cho tất cả các đường có thể có của đường trở về tự động; thực hiện đánh giá khả thi dựa trên tất cả các đường có thể và sàng lọc các đường khả thi; đánh giá xem đường tốt hay xấu dựa trên đường khả thi và sàng lọc một đường tối ưu.

(3) Sáng chế số WO2017161632A1, *"Cleaning robot optimal target path planning method based on model learning"*. (link:

https://patents.google.com/patent/WO2017161632A1/en?oq=WO2017161632A1), sáng chế này đề xuất ra phương pháp lập kế hoạch đường đi mục tiêu tối ưu của robot vệ sinh dựa trên mô hình học, sử dụng kiến trúc học tăng cường và thuật toán Dyna-H, thuật toán R-MAX.

(4) Sáng chế số US8903589B2 *"Method and apparatus for simultaneous localization and mapping of mobile robot environment"* (link:

https://patents.google.com/patent/US8903589B2/en?oq=US8903589B2), sáng chế này đề xuất ra kỹ thuật tối ưu hóa hiệu suất của quy trình định vị và lập bản đồ đồng thời (SLAM) cho robot di động.

(5) Sáng chế số CN109839927A *"Method and device for robot path planning"*. (link: https://patents.google.com/patent/CN109839927A/en?oq=CN109839927A), sáng chế này đề xuất ra phương pháp lập kế hoạch đường đi của robot: thu thập điểm xuất phát và điểm đến, lập bản đồ robot; tìm đường đi ngắn nhất dùng thuật toán theo dõi ngược.

(6) Bài báo *Trajectory optimization and obstacle avoidance of autonomous robot using Robust and Efficient Rapidly Exploring Random Tree, (link:*

*https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0311179), s*ử dụng thuật toán RRT dựa vào lấy mẫu để tìm đường đi tối ưu cho robot dựa trên dữ liệu hành trình trước.

(7) Bài báo: *An, P. T. and Le, N. T. Multiple shooting approach for finding approximately shortest paths for autonomous robots in unknown environments in 2D,* được công bố dưới dạng tiền ấn phẩm trên Arxiv [bản V2 ngày 16/12/2023], (link:

https://arxiv.org/abs/2208.10386), và bản chính thức được xuất bản trên tạp chí Journal of Combinatorial Optimization năm 2024 (xem [3]).

Các sáng chế và bài báo (6) kể trên cung cấp thuật toán lập kế hoạch đường đi cho robot bằng cách sử dụng hoặc cải tiến các thuật toán tìm kiếm trong đồ thị, các thuật toán trong học máy... Từ đó robot trở lại bằng một quỹ đạo có sẵn. Quỹ đạo này chính là đường đi trong một đồ thị có các nút thu được từ thông tin cục bộ về các vị trí mà robot đã đi qua trong quá khứ. Cách tiếp cận này cũng xuất hiện trong các thuật toán lập kế hoạch đường đi

cho robot dựa vào tìm kiếm đồ thị [5, 12], thuật toán trường lực hấp dẫn [6, 11], bản đồ hóa đường đi [8, 9] hoặc các thuật toán dựa vào sinh mẫu ngẫu nhiên (xem bài báo (6)). Trong các công bố này các tác giả đều tập trung vào việc tìm đường đi tới đích dựa trên các nút có sẵn. Giải pháp chúng tôi đưa ra hướng đến việc tìm một đường đi mới có độ dài ngắn hơn quỹ đạo có sẵn trước đó, đường đi này là kết quả của việc giải bài toán đường đi ngắn nhất hình học. Cách tiếp cận mới mẻ sử dụng các kết quả của hình học tính toán mang lại ưu thế bởi vì robot trở lại bằng đường đi tối ưu được tìm ra sẽ giúp tiết kiệm thời gian và năng lượng vận hành robot.

Bài báo (7) là tiền ấn phẩm đăng trên Arxiv, mà bản chính thức của cùng nhóm tác giả được xuất bản trên tạp chí Journal of Combinatorial Optimization năm 2024 (xem [3]) có một số khác biệt.

Cụ thể, bản chính thức [3] được bổ sung Mệnh đề về độ phức tạp tính toán (Mệnh đề 4 trong [3]); các hình vẽ minh họa một số đồ thị và việc tìm đường xương cá (skeleton paths) là đường đi ngắn nhất trong các đồ thị này (Hình 8 trong [3]); các kết quả thực nghiệm so sánh đánh giá tốc độ hội tụ của các phương pháp được đề xuất là phương pháp bắn nhiều lần, phương pháp của Li và Kelette (phần cuối Mục 6 và 15 trong [3]; phần phụ lục phân tích các hạn chế của thuật toán phễu của Lee và Preparata, 1984, thuật toán của Trang và các cộng sự, 2017, thuật toán của Li và Klette (Phụ lục B trong [3]).

Danh mục các tài liệu tham khảo được trích dẫn

[1] An, P. T., Hai, N. N., and Hoai, T. V. (2013). Direct multiple shooting method for solving approximate shortest path problems. Journal of Computational and Applied Mathematics, 244, 67–76.

[2] An, P. T. and Le, N. T. (2024). Multiple shooting approach for finding approximately shortest paths for autonomous robots in unknown environments in 2D, Journal of Combinatorial Optimization, 47(71).

[3] An, P. T. and Trang, L. H. (2018). Multiple shooting approach for computing shortest descending paths on convex terrains. Computational and Applied Mathematics, 37(4), 1–31.

[4] An, P. T. , Anh, P. H. , Binh, T. T. , Hoai, T. V. , Le, N. T. , Tam, T. M. , Truong, V. P. , and Vi , T. L. (2022). Shortest paths on sequences of bundles of line segments for

autonomous robots in unknown evironments, Hội thảo Tối ưu và Tính toán khoa học lần thứ 20, Ba Vì - Hà Nội, 21-23/4/2022.

[5] Brooks, Rodney A., and Tomas Lozano-Perez. (1985). A subdivision algorithm in configuration space for findpath with rotation. IEEE Transactions on Systems, Man, and Cybernetics 2, pp. 224–233.

[6] Glavaski, D., Volf, M., & Bonkovic, M. (2009). Mobile robot path planning using exact cell decomposition and potential field methods. WSEAS Transactions on Circuits and Systems, 8(9), pp. 789–800.

[7] Hoai, T.V., An, P. T., and Hai, N. N. (2017). Multiple shooting approach for computing approximately shortest paths on convex polytopes. Journal of Computational and Applied Mathematic, 317, 235–246.

[8] Hsu, D., Latombe, J. C., Kurniawati, H. (2006). On the probabilistic foundations of probabilistic roadmap planning. The International Journal of Robotics Research, 25(7), 627–643.

[9] Ladd, A. M., & Kavraki, L. E., (2004). Measure theoretic analysis of probabilistic path planning. IEEE Transactions on Robotics and Automation, 20(2), 229–242.

[10] Li, F., & Klette, R. (2011). Rubberband algorithms. In Euclidean Shortest Paths (53-89). Springer, London, 53–89.

[11] Rosell, J., & Iniguez, P. (2005). Path planning using harmonic functions and probabilistic cell decomposition. In Proceedings of the 2005 IEEE international conference on robotics and automation 1803–1808. IEEE.

[12] Warren, C. W. (1993). Fast path planning using modified A* method. In Proceedings IEEE International Conference on robotics and Automation, 662-667. IEEE.

Bản chất kỹ thuật của giải pháp hữu ích

Giải pháp hữu ích mô tả quy trình một robot tự hành với tầm nhìn hạn chế khám phá môi trường với các vật cản đa giác chưa xác định trước, tối ưu độ dài đường đi đến đích bằng cách tìm đường trở lại ngắn nhất (*xem quy trình trong Hình 1*). Các vùng robot trở lại được mô hình hóa dưới dạng dãy các chùm các đoạn thẳng, giúp chuyển bài toán tối ưu đường trở lại thành một bài toán hình học, giải nó bằng phương pháp bắn nhiều lần, từ đó

đưa ra giải pháp làm cho độ dài con đường quay lại ngắn hơn quỹ đạo có sẵn trước đó của robot, giúp tiết kiệm thời gian và năng lượng.

Robot bắt đầu chuyển động dựa vào việc quét một tầm nhìn là một hình tròn tâm tại vị trí của robot còn bán kính xác định trước tùy theo cấu hình của robot. Trong mỗi bước đi, robot xác định phần không gian không va chạm và phần không gian cảnh báo có va chạm. Cũng trong mỗi bước đi robot đánh giá và lưu trữ vị trí các điểm mà nó có thể chuyển động tới cho các bước tiếp theo, đảm bảo không va chạm. Sau nhiều bước di chuyển của robot, tập hợp các điểm này cũng tăng lên. Các điểm này được tính toán và xếp hạng để chọn ra điểm có xếp hạng tốt nhất nhằm ưu tiên tiếp cận tại bước tiếp theo.

Giả sử tại một thời điểm nào đó, robot đang ở vị trí q của bản đồ. Trong quá trình khám phá môi trường, robot đi đến điểm có thứ hạng ưu tiên cho chuyển động tiếp theo, gọi là p. Mặc dù p là điểm mà robot chưa đi qua, nhưng p đã được đánh dấu và xếp hạng trong quá khứ. Có 2 trường hợp xảy ra: nếu p thuộc tầm nhìn hiện tại của robot, nó chỉ việc dịch chuyển theo đoạn thẳng từ q đến p, nhưng nếu p không nằm trong tầm nhìn hiện tại việc chạm đến p có rất nhiều con đường để đi. Một đường đi của robot trong miền không va chạm vật cản, từ vị trí q đến p, không thuộc tầm nhìn hiện tại được gọi là "đường trở lại".

Mục đích của giải pháp hữu ích là giảm thiểu độ dài đường đi của robot tự hành thông qua việc giảm thiểu độ dài con đường quay trở lại của robot (*xem mô tả trong Hình 2*). Để đạt được mục đích đó, giải pháp cụ thể được đề xuất như sau:

i) Trích xuất một dãy các chùm đoạn thẳng trên con đường robot chuyển động trở lại nhằm mô hình hóa phần không gian mà robot sẽ quay trở lại, khi robot đi trên con đường dọc theo các dãy chùm đoạn thẳng này sẽ đảm bảo không chạm vào vật cản. Đồng thời khi giải một bài toán hình học quen thuộc là tìm đường đi dọc theo dãy các chùm đoạn thẳng bằng phương pháp bắn nhiều lần, ta sẽ thu được con đường ngắn hơn quỹ đạo có sẵn mà các cách tiếp cận trước đây sử dụng.

ii) Trong quá trình robot tự hành khám phá môi trường và tối ưu đường đi đến đích, robot cần tìm các đường trở lại ngắn hơn các quỹ đạo có sẵn. Theo quy trình này, miền không gian mà robot đi qua được mô hình hóa bằng dãy các chùm đoạn thẳng, sau đó sẽ áp dụng phương pháp trên (*i*) để tìm đường quay trở lại.

Việc các vùng mà robot trở lại được mô hình hóa dưới dạng dãy các chùm các đoạn thẳng, và quy bài toán về bài toán hình học tìm đường đi ngắn nhất dọc theo dãy chùm đoạn

thẳng là sự khác biệt rõ ràng so với cách tiếp cận trước đây. Cụ thể cách thức robot tự hành khám phá môi trường và tối ưu độ dài đường đi đến đích bằng cách tìm đường trở lại ngắn nhất dọc theo dãy các chùm đoạn thẳng đã xác định trước bao gồm các bước sau:

bước S1, quét tầm nhìn ($\overline{B}[a,r]$) để xác định các phần không gian không va chạm, nơi mà robot có thể di chuyển trong các bước tiếp theo, và các phần không gian va chạm, nơi mà robot sẽ gặp phải chướng ngại vật nếu chuyển động quãng đường thẳng bằng độ dài bán kính (xem *Hình 3*),

bước S2, thiết lập những điểm có thể di chuyển tiếp theo, gọi là "điểm mở", là điểm đại diện cho mỗi góc không quá 120^0 cho mỗi phần không gian không va chạm để đưa vào tập hợp các điểm mở có thể di chuyển tiếp theo, nếu phần không va chạm có góc tương ứng vượt quá 120^0 thì ta thực hiện chia thành các phần không va chạm hình quạt có góc không quá 120^0, và

trong mỗi bước đi của robot, đánh giá và lưu trữ vị trí các điểm mở mà robot có thể chuyển động tới cho các bước tiếp theo, đảm bảo không va chạm với các chướng ngại vật;

các điểm mở được tính toán và xếp hạng để chọn ra điểm có xếp hạng tốt nhất để ưu tiên tiếp cận tại bước tiếp theo;

thứ hạng các điểm này được xác định theo một quy tắc định trước và được thiết lập cẩn thận trong việc lập kế hoạch đường đi địa phương, robot đi đến điểm có thứ hạng ưu tiên cho chuyển động tiếp theo, gọi là p; trong đó, thứ hạng được tính như sau:

$$rank(t) = \begin{cases} \dfrac{\alpha}{d(t)} + \dfrac{\beta}{a(t)} \text{ nếu } d(t) \neq 0 \text{ và } a(t) \neq 0 \\ +\infty \text{ nếu } d(t) = 0 \text{ hoặc } a(t) = 0 \end{cases}$$

trong đó:

$d(t)$ là khoảng cách giữa điểm mở và mục tiêu,

$a(t)$ là góc tạo thành giữa a đến điểm mở và a đến mục tiêu,

α và β được thiết lập dựa trên mức độ quan trọng của góc và khoảng cách,

trong quá trình triển khai, lấy $\alpha = \beta = 1$, điểm mở có thứ hạng cao nhất sẽ có mức ưu tiên đầu tiên; lưu trữ đồ thị G bao gồm các nút là tâm và các lá là điểm cuối của các đoạn thẳng biểu diễn các điểm mở; và một tập OP bao gồm các điểm mở và thứ hạng

của chúng, khi lưu trữ hiệu quả các nút của đồ thị G, mỗi góc của cung xác định mỗi điểm mở không vượt quá 120^0, và mỗi điểm mở không thuộc về bất kỳ điểm mở hoặc đóng nào trước đó;

nếu p thuộc tầm nhìn hiện tại của robot, nó chỉ việc dịch chuyển theo đoạn thẳng từ q đến p như chỉ ra theo bước S4, nhưng nếu p không nằm trong tầm nhìn hiện tại việc chạm đến p thực hiện theo bước S3 tiếp theo. Chú ý rằng robot có thể trở lại theo một quỹ đạo có sẵn gọi là "đường xương cá" (skeleton path), nó là đường đi ngắn nhất nối p và q trong đồ thị G, nhưng để giảm thiểu độ dài đường đi, ở đây ta giải bài tìm đường đi ngắn nhất dọc theo dãy chùm đoạn thẳng như sau;

bước S3, trích xuất dãy chùm đoạn thẳng F và sử dụng phương pháp bắn nhiều lần MMS (Method of Multiple Shooting, MMS) cho việc tìm đường đi dọc theo dãy chùm đoạn thẳng,

trong đó, bước S3 bao gồm:

bước S3.1, robot xác định được điểm p có thứ hạng cao nhất cần trở lại mà p không nằm trong tầm nhìn hiện tại, việc trích xuất được các dãy chùm đoạn thẳng thuộc phần không gian robot sẽ trở lại được tiến hành; đường xương cá và dãy chùm đoạn thẳng gắn với đường xương cá được tạo ra, tại mỗi đỉnh, chỉ cần xét chùm đoạn thẳng nằm trong cung tròn có tâm tại đỉnh chùm và góc của cung không vượt quá π;

bước S3.2, tối ưu đường quay lại bằng cách tìm đường đi ngắn nhất hình học dọc theo một dãy các chùm đoạn thẳng (F) bằng phương pháp MMS,

bước S3.3, robot di chuyển theo đường ngắn hơn đã tìm được ở bước trên,

bước S4, sau khi xác định được đường đi từ q trở lại p, robot chuyển động tới p; gán $q=p$, nếu điểm đích không thuộc tầm nhìn hiện tại $\overline{B}[q,r]$ của robot, quá trình lại tiếp tục trở lại bước S1."

trong đó, bước S3.2 được thực hiện như sau:

chia dãy các chùm đoạn thẳng (F) thành các dãy con, chọn một tập hợp có thứ tự của các đoạn cắt, là các đoạn cuối của mỗi dãy con, và lấy một điểm bắn đầu tiên trong mỗi đoạn cắt;

xây dựng một đường dọc theo dãy các chùm đoạn thẳng) nối p và q, được tạo thành từ tập hợp các điểm bắn, đường này là sự kết hợp của các đường ngắn

nhất nối hai điểm bắn liên tiếp dọc theo dãy con tương ứng, thiết lập điều kiện dừng tại các điểm bắn;

kiểm tra điều kiện dừng ở tất cả các điểm bắn, nếu điều kiện thẳng hàng không được thỏa mãn, các điểm bắn mới sẽ cải thiện đường nối p và q dọc theo dãy các chùm đoạn thẳng (F), được tạo bởi tập hợp các điểm bắn mới

trong đó, thời gian chạy của thuật toán ở bước 3.2 phụ thuộc vào m, N, J, với: N là số lượng chùm của dãy các chùm đoạn thẳng (F), J là số lượng đoạn thẳng của dãy các chùm đoạn thẳng (F), và m là số lần lặp của thuật toán để tìm được đường đi cần thiết nối giữa p và q.

Mô tả vắn tắt các hình vẽ

Hình 1. Quy trình: robot tự hành khám phá môi trường và tối ưu độ dài đường đi đến đích bằng cách tìm đường trở lại ngắn nhất

Hình 2. Bước (S3): trích xuất dãy chùm đoạn thẳng và sử dụng phương pháp bắn nhiều lần MMS cho việc tìm đường quay lại cho robot tự hành dọc theo dãy các chùm đoạn thẳng.

Hình 3. Bước (S1) Sự giao nhau giữa tầm nhìn hiện tại của robot và các vật cản đa giác. Hình tam giác màu nhạt hơn đại diện cho phần không gian không va chạm, nơi mà robot có thể di chuyển an toàn quãng đường một bước bằng độ dài bán kính trong khi hình tam giác màu đậm hơn thể hiện phần không gian cảnh báo va chạm, nơi mà robot sẽ gặp vật cản nếu di chuyển một bước bằng độ dài bán kính.

Hình 4. Bước (S1) Cách xác định các điểm mở có thể di chuyển tiếp theo: Giả sử robot đang ở điểm a_2, xét giao điểm giữa tầm nhìn hiện tại với các vật cản để xác định phần không gian không va chạm và phần không gian cảnh báo có va chạm. Lấy một điểm gọi là điểm mở, đại diện cho mỗi góc không quá 120^0 cho mỗi phần không gian không va chạm (tương ứng màu nhạt hơn) để đưa vào tập hợp các điểm di chuyển tiếp theo (nếu phần không va chạm có góc tương ứng vượt quá 120^0 thì ta thực hiện chia thành các phần không va chạm hình quạt có góc không quá 120^0).

Hình 5. Bước (S2) Tập các điểm mở có thể di chuyển tiếp theo cho đến khi robot đến điểm $q = a_8$ gồm có s, p và t. Bằng tính toán xếp hạng, điểm có thứ tự ưu tiên cao nhất là p. Do đó robot tiến hành di chuyển từ q trở lại p.

Hình 6 và 7. Bước (S3.1) Trích xuất ra dãy chùm đoạn thẳng từ thuộc phần không gian robot sẽ trở lại.

Hình 8. Bước (S3.1) Tiền xử lý các chùm đoạn thẳng có sẵn. Dãy chùm đoạn thẳng ban đầu được đưa về dãy các chùm đoạn thẳng nằm trong cung tròn có tâm tại đỉnh chùm và góc của cung không vượt quá 180^0. Bước (S3.2) Tính toán đường quay lại tối ưu dọc theo dãy các chùm đoạn thẳng bằng phương pháp MMS. Đường đi (đường nét liền ở giữa) dọc theo dãy chùm đoạn thẳng là đường đi thay thế cho đường xương cá như Bước(S3.3) đề cập.

Hình 9. Tập bản đồ dùng để chạy thuật toán và thu kết quả.

Hình 10. Minh họa các chuyển động thu được bằng cách sử dụng phương pháp bắn nhiều lần MMS.

Hình 11. Minh họa việc trích xuất các dãy chùm đoạn thẳng thu được.

Mô tả chi tiết giải pháp hữu ích

Các phương án ưu tiên thực hiện của giải pháp hữu ích sẽ được mô tả chi tiết sau đây, có tham chiếu đến các hình vẽ kèm theo.

Tìm đường đi ngắn nhất là một bài toán tối ưu hình học tự nhiên với nhiều ứng dụng trong các lĩnh vực khác nhau như hệ thống thông tin địa lý (GIS), robot, thị giác máy tính, v.v.. Một trường hợp của bài toán đường đi ngắn nhất hình học là tính toán đường đi ngắn nhất nối hai điểm dọc theo một tập hợp các đoạn thẳng trong không gian 2D như sau:

Cho hai điểm và một tập hợp có thứ tự các đoạn thẳng trong không gian 2D, tìm đường đi ngắn nhất nối hai điểm này sao cho nó đi qua các đoạn thẳng theo thứ tự đã cho.

Nếu các đoạn thẳng rời nhau đôi một, nó được gọi là bài toán SP cho các đoạn thẳng rời nhau đôi một. Xét bài toán SP cho các chùm đoạn thẳng như sau:

(P) Cho hai điểm và một dãy các chùm đoạn thẳng trong không gian 2D trong đó các chùm này không giao nhau, tìm đường đi ngắn nhất nối hai điểm dọc theo dãy chùm đoạn thẳng.

Bài toán (P) cũng là một trường hợp của bài toán SP cho các đoạn thẳng và ngược lại. Tuy nhiên, bài toán (P) tổng quát hơn bài toán SP cho các đoạn thẳng rời nhau đôi một.

Bằng cách tiếp cận tối ưu lồi, Polishchuk và Mitchell đã sử dụng chương trình hình nón bậc hai (SOCP) để giải quyết bài toán SP cho các đoạn thẳng tùy ý, nhưng phương pháp này không phổ biến vì các tác giả đã nói rằng việc sử dụng SOCP được xem là một "hộp đen" mà không hiểu rõ bên trong hoạt động như thế nào. Hiện tại không có thuật toán hình học nào có thể giải bài toán SP cho các đoạn thẳng tùy ý. Tuy nhiên, các lời giải chính xác và gần đúng có thể đạt được dưới một số điều kiện cụ thể của các đoạn thẳng, đáng tiếc là các thuật toán này không áp dụng được cho bài toán SP với các chùm đoạn thẳng. Ví dụ, nếu các đoạn thẳng trong một dãy chùm có thể được sắp xếp thành danh sách các đường chéo của các tam giác kề nhau, thì kỹ thuật phễu của Lee và Preparata có thể được sử dụng để tìm đường đi ngắn nhất chính xác. Tuy vậy miền chứa các đường đi mà chúng ta xét không phải là một đa giác đã được tam giác phân, do đó, áp dụng kỹ thuật phễu nói trên là không thể cho bài toán SP với các chùm đoạn thẳng. Nếu để giải bài toán SP cho các đoạn thẳng rời nhau đôi một, có thể sử dụng các thuật toán lặp theo các phương pháp gần đúng của Li và Klette (*Rubberband algorithms. Euclidean Shortest Paths*) và Trang và cộng sự (*An iterative algorithm for computing shortest paths through line segments in 3D. In International Conference on Future Data and Security Engineering*). Tuy nhiên, các thuật toán này không phù hợp khi mà các đoạn thẳng không rời nhau đôi một, chẳng hạn khi tập hợp các đoạn thẳng là một dãy chùm. Câu hỏi đặt ra là "Có thuật toán hiệu quả nào cho bài toán SP với các chùm đoạn thẳng không?" Giải pháp hữu ích sử dụng phương pháp bắn nhiều lần MMS lần để trả lời câu hỏi này.

Xét một bài toán lập kế hoạch đường đi trong đó robot có phạm vi tầm nhìn hạn chế di chuyển trong một môi trước chưa xác định, trong không gian 2D và tránh các chướng ngại vật là các đa giác. Lưu ý rằng robot không có tất cả thông tin về các chướng ngại vật ngoại trừ tọa độ của điểm bắt đầu và điểm kết thúc.

Việc tránh các chướng ngại vật yêu cầu các đường đi không cắt vào bên trong của các chướng ngại vật này. Ngoài việc chạm đến đích, việc tiết kiệm thời gian và giảm độ dài đường đi cũng rất quan trọng. Hãy giả sử robot đang ở vị trí q trên bản đồ. Trong quá trình khám phá môi trường, robot có thể quay lại một số vị trí, gọi là p, mà không thuộc tầm nhìn hiện tại của robot. Mặc dù p chưa được robot đi qua, p đã được đánh dấu và xếp hạng trong quá khứ. Một cách tiếp cận phổ biến của các thuật toán lập kế hoạch đường đi cho robot là sử dụng một quỹ đạo có sẵn cho đường trở lại.

Quỹ đạo có sẵn (đường xương cá - skeleton path), nó là đường đi ngắn nhất nối p và q trong đồ thị G, là đồ thị có các nút thu được từ thông tin cục bộ của các vị trí mà robot đã đi qua trong quá khứ và có các cạnh là các đoạn thẳng nối các nút. Các cách tiếp cận sữ dụng quỹ đạo có sẵn này xuất hiện trong các thuật toán lập kế hoạch đường đi sử dụng các phương pháp tìm kiếm trên đồ thị, các thuật toán trường lực hấp dẫn và các thuật toán biểu đồ đường đi. Để làm cho robot di chuyển hiệu quả, An và cộng sự đã đề xuất một thuật toán lập kế hoạch đường đi cho robot tự hành trong đó chú trọng tìm đường đi ngắn nhất hình học thay thế cho quỹ đạo có sẵn thu được từ G. Bài toán tối ưu hóa đường đi cho các chuyển động quay trở lại được phát biểu như sau:

(P∗) Tìm một đường đi (hoặc đường đi ngắn nhất) nối p và q tránh các chướng ngại vật, mà ngắn hơn quỹ đạo có sẵn thu được từ đồ thị G.

Giải pháp hữu ích mô tả cách thức biểu diễn khu vực mà robot sẽ di chuyển từ p đến q, khu vực này được xác định là một dãy các chùm đoạn thẳng, chúng được trích xuất từ thông tin cục bộ có được khi robot khám phá môi trường, giúp cho việc chuyển Bài toán (P∗) thành một bài toán tìm đường đi ngắn nhất hình học. Thay vì giải quyết Bài toán (P∗), ta có thể tiếp cận bài toán SP cho dãy các chùm đoạn thẳng bằng phương pháp bắn nhiều lần MMS.

Trong giải pháp được nêu ra ở đây, chúng tôi tập trung vào trích xuất dãy chùm đoạn thẳng, giải gần đúng Bài toán (P) trên dãy chùm bằng phương pháp MMS và ứng dụng của nó trong việc giảm thiểu độ dài đường đi đến mục tiêu cho robot tự hành có tầm nhìn hạn chế và cần tránh các vật cản.

Nếu robot di chuyển từ q đến p, trong đó p đã được đánh dấu trong quá khứ và không nằm trong tầm nhìn hiện tại của robot, thì MMS sẽ được gọi để tìm gần đúng đường đi ngắn nhất nối p và q dọc theo dãy các chùm các đoạn thẳng theo các bước mô tả trong *Hình 2*. Đây là một đường đi khả thi và ngắn hơn quỹ đạo có sẵn là đường đi ngắn nhất nối p và q trên đồ thị G.

Giải pháp hữu ích đề xuất thuật toán lặp dựa trên phương pháp bắn nhiều lần MMS để robot tìm đường trở lại ngắn nhất nối hai điểm p và q dọc theo dãy các chùm F đoạn thẳng như sau:

(f1) Chia dãy F của các chùm đoạn thẳng thành các dãy chùm con. Chọn một tập hợp có thứ tự của các đoạn cắt, là các đoạn cuối của mỗi dãy chùm con. Lấy một điểm bắn đầu tiên trong mỗi đoạn cắt.

(f2) Xây dựng một đường đi khởi tạo dọc theo F nối p và q, được tạo thành từ tập hợp các điểm bắn. Đường này là sự kết hợp của các đường ngắn nhất nối hai điểm bắn liên tiếp dọc theo dãy con tương ứng. Một điều kiện dừng được thiết lập tại các điểm bắn. Trong đó, điều kiện dừng được xây dựng dựa vào góc có hướng tại điểm bắn đó, góc này lớn hơn hoặc bằng π thì điều kiện thẳng hàng được xem là xảy ra tại điểm bắn đó.

(f3) Thuật toán kiểm tra điều kiện dừng ở tất cả các điểm bắn. Nếu điều kiện dừng không được thỏa mãn, thì cập nhật đến các điểm bắn mới và đường đi mới được tạo bởi tập hợp các điểm bắn mới sẽ cải thiện đường nối p và q dọc theo F.

Mã giả của thuật toán MMS tìm đường đi ngắn nhất xấp xỉ nối hai điểm dọc theo một dãy các chùm đoạn thẳng được xây dựng như sau:

Bảng 1. Mã giả thuật toán MMS

Input: Hai điểm p và q, một dãy F các chùm đoạn thẳng.

Output: Một đường đi ngắn nhất xấp xỉ nối p và q dọc theo F.

1: Chia F thành F_i thỏa mãn bằng cách cắt các đoạn $[u_i,v_i]$, với $i = 0, 1, ..., K+1$, (K là số đoạn cắt)

2: Lấy một tập hợp có thứ tự $s_i i = 0,..$ K+1 của các điểm bắn ban đầu bao gồm p, v_i(i = 1, 2,..., K) và q. Gọi $\gamma^{current}$ là đường đi được tạo thành từ s_i , $i = 0$, K+1.

3: $flag \leftarrow true$, $s_i^{next} \leftarrow s_i$ và γ^{next} là đường đi được tạo thành từ $s_i^{next}, i = 0,..$ K+1.

4: Gọi Thủ tục COLLINEAR_UPDATE ($\gamma^{current}$, γ^{next}, $flag$).

5: **if** ddieufe kiện dừng thỏa mãn tại tất cả các điểm bắn, tức là, flag = true **then**

6: **stop** và **return** γ^{next} ▷*Đường đi ngắn nhất đã được tìm thấy*

7: **else** $\gamma^{current} \leftarrow \gamma^{next}$ và **return** bước 4.

8: **return** γ^{next}.

Nếu điều kiện dừng có được tại tất cả các điểm bắn, Thuật toán dùng MMS dừng lại sau một số bước lặp hữu hạn và đường đi ngắn nhất được tìm thấy.

Ngược lại, Thủ tục COLLINEAR_UPDATE($\gamma^{current}$, γ^{next}, *flag*) thực hiện việc cập nhật các điểm bắn để có được một đường đi γ^{next} tốt hơn $\gamma^{current}$

Bảng 2. Mã giả thuật toán thực hiện cập nhật các điểm bắn COLLINEAR_UPDATE

Thủ tục COLLINEAR_UPDATE($\gamma^{current}, \gamma^{next}, flag$)

Input: $\gamma^{current}$ là hợp nhất của các đường đi ngắn nhất nối các cặp điểm bắn s_i, tức là $\gamma^{current} = \cup_{i=0}^{K} \text{SP}(s_i, s_{i+1})$

Output: Xác định nếu $flag = true$ hoặc $flag = false$ và tập hợp các điểm bắn mới sao cho $l(\gamma^{next}) \leq l(\gamma^{current})$, trong đó γ^{next} là hợp nhất của các đường đi ngắn nhất nối các cặp điểm bắn mới s_i^{next}

$flag \leftarrow true$
for $i = 0, 1, \ldots, K$ **do**
    Lấy các điểm t_i được định nghĩa trong (B) sao cho
        nếu $\text{SP}(s_i, s_{i+1})$ là một đoạn thẳng, thì t_i là trung điểm của $\text{SP}(s_i, s_{i+1})$,
        nếu không, t_i là một trong các điểm đầu không trùng với s_i và s_{i+1}.
    if i=0 **then**
        continue
    if $s_i \in]v_i, u_i[$ **then**
        if $\angle_{v_i u_i}(\text{SP}(s_{i-1}, s_i), \text{SP}(s_i, s_{i+1})) = \pi$ **then** ▷ (A1-A2) được kiểm tra
            Đặt $s_i^{next} \leftarrow s_i$
        else gọi $\text{SP}(t_{i-1}, t_i)$
            Đặt s_i^{next} là điểm giao nhau của $\text{SP}(t_{i-1}, t_i)$ và $[u_i, v_i]$ ▷ *dựa vào* (B)
            $flag \leftarrow false$
    else if $s_i = v_i$ **then**
        if $\angle_{v_i u_i}(\text{SP}(s_{i-1}, s_i), \text{SP}(s_i, s_{i+1})) \leq \pi$ **then** ▷ (A1-A2) được kiểm tra
            Đặt $s_i^{next} \leftarrow s_i$
        else gọi $\text{SP}(t_{i-1}, t_i)$
            Đặt s_i^{next} là điểm giao nhau của $\text{SP}(t_{i-1}, t_i)$ và $[u_i, v_i]$ ▷ *dựa vào* (B)
            $flag \leftarrow false$
    else if $s_i = v_i$ **then**
        if $\angle_{v_i u_i}(\text{SP}(s_{i-1}, s_i), \text{SP}(s_i, s_{i+1})) \leq \pi$ **then** ▷ (A1-A2) được kiểm tra
            Đặt $s_i^{next} \leftarrow s_i$
        else gọi $\text{SP}(t_{i-1}, t_i)$
            Đặt s_i^{next} là điểm giao nhau của $\text{SP}(t_{i-1}, t_i)$ và $[u_i, v_i]$ ▷ *dựa vào* (B)
            $flag \leftarrow false$
$s_0^{next} \leftarrow p, s_{K+1}^{next} \leftarrow q$

Thuật toán nêu trên được áp dụng vào robot tự hành trong quá trình khám phá môi trường nhằm tối ưu độ dài đường đi đến đích theo quy trình mô tả trong *Hình 1*, cụ thể như nêu dưới đây:

Môi trường và các chướng ngại vật được biểu diễn bởi các miền bị chặn bởi hợp của các đường gấp khúc, chẳng hạn như đa giác được thể hiện trong *Hình 3*. Đường đi của robot phải nằm trong miền không gian không va chạm, tức là, đường đi không giao nhau với phần trong trong của bất kỳ vật cản nào.

Bước S1: giả sử tại một thời điểm nào đó, robot đang ở vị trí q của bản đồ. robot quét một tầm nhìn. Phạm vi tầm nhìn của robot được thể hiện bằng một đường tròn có tâm ở vị trí a hiện tại với bán kính r, ký hiệu $B[a,r]$. Chúng ta xét giao giữa $B[a,r]$ và phần không gian không va chạm. Nó được biểu diễn bởi các vùng màu nhạt hơn và màu đậm hơn như trong *Hình 3*. Cụ thể, các hình tam giác màu nhạt hơn minh họa các phần không gian không va chạm, nơi mà robot có thể di chuyển trong các bước tiếp theo, trong khi các hình tam giác màu đậm hơn biểu thị các phần không gian cảnh báo va chạm, nơi mà robot sẽ gặp phải chướng ngại vật nếu chuyển động quãng đường thẳng bằng độ dài bán kính. Mỗi phần không gian không va chạm, chọn ra một điểm mở có thể di chuyển tiếp theo, là điểm đại diện cho mỗi góc không quá 120^0 cho mỗi phần không gian không va chạm (tương ứng màu nhạt hơn) để đưa vào tập hợp các điểm mở có thể di chuyển tiếp theo (nếu phần không va chạm có góc tương ứng vượt quá 120^0 thì ta thực hiện chia thành các phần không va chạm hình quạt có góc không quá 120^0). Cũng trong mỗi bước đi robot đánh giá và lưu trữ vị trí các điểm mở mà nó có thể chuyển động tới cho các bước tiếp theo, đảm bảo không va chạm *(xem Hình 4)*. Sau nhiều bước di chuyển của robot, tập hợp các điểm này cũng tăng lên.

Bước S2: các điểm mở được tính toán và xếp hạng để chọn ra điểm có xếp hạng tốt nhất để ưu tiên tiếp cận tại bước tiếp theo. Thứ hạng các điểm này được xác định theo một quy tắc định trước và được thiết lập cẩn thận trong việc lập kế hoạch đường đi địa phương. Đặc biệt, có hai yếu tố bao gồm khoảng cách giữa mỗi điểm có thể di chuyển tiếp theo với mục tiêu, và góc. Trong quá trình khám phá môi trường, robot đi đến điểm có thứ hạng ưu tiên cho chuyển động tiếp theo, gọi là p.

Có 2 trường hợp xảy ra: nếu p thuộc tầm nhìn hiện tại của robot, nó chỉ việc dịch chuyển theo đoạn thẳng từ q đến p như chỉ ra theo Bước S4, nhưng nếu p không nằm trong tầm nhìn hiện tại việc chạm đến p có rất nhiều con đường để đi *(xem Hình 5)*.

Bước S3: áp dụng phương pháp chỉ ra ở phần trên (phần: *Trích xuất dãy chùm đoạn thẳng và sử dụng phương pháp bắn nhiều lần cho việc tìm đường đi dọc theo dãy chùm đoạn thẳng)*. Cụ thể, bước này bao gồm:

Bước S3.1: đang ở điểm q, robot xác định được điểm p có thứ hạng cao nhất cần trở lại mà p không nằm trong tầm nhìn hiện tại, việc trích xuất được các dãy chùm đoạn thẳng thuộc phần không gian robot sẽ trở lại được tiến hành.

Một đường đi của robot trong miền không va chạm vật cản, từ vị trí q đến p được gọi là "đường trở lại". Một cách tiếp cận phổ biến của các thuật toán lập kế hoạch đường đi cho robot trước đó chính là sử dụng ngay một quỹ đạo có sẵn cho các đường trở lại. Quỹ đạo này chính là đường đi trong một đồ thị có các nút thu được từ thông tin cục bộ về các vị trí mà robot đã đi qua trong quá khứ, gọi là đường xương cá. Từ thông tin thu được sau mỗi bước chuyển động của robot, đường xương cá và dãy chùm đoạn thẳng gắn với đường xương cá được tạo ra, xem *Hình 6* và *7*. Mỗi chùm đoạn thẳng được lấy từ một tập hợp các hình tam giác màu đậm hơn đại diện cho các tầm nhìn đóng. Tại mỗi đỉnh, ta chỉ cần xét chùm đoạn thẳng nằm trong cung tròn có tâm tại đỉnh chùm và góc của cung không vượt quá 180^0.

Bước S3.2: tuy nhiên, để tạo ra một chuyển động hiệu quả, ta sẽ tối ưu đường quay lại bằng cách giải quyết một bài toán tìm đường đi ngắn nhất hình học dọc theo một dãy các chùm đoạn thẳng bằng phương pháp bắn nhiều lần MMS và so sánh với việc áp dụng kỹ thuật của Li và Kelette.

Bước S3.3: việc trở lại bằng một đường ngắn hơn đường xương cá giúp tiết kiệm thời gian cho các robot trong thực tế (xem đường nét liền ở giữa trong *Hình 8*). Đường đi (đường nét liền ở giữa) dọc theo dãy chùm đoạn thẳng là đường đi thay thế cho đường xương cá.

Bước S4: sau khi xác định được đường đi từ q trở lại p, robot chuyển động tới p. Gán $q=p$. Nếu điểm đích không thuộc tầm nhìn hiện tại B[q,r] của robot, quá trình lại tiếp tục trở lại Bước S1.

Ví dụ thực hiện giải pháp hữu ích

Chúng tôi sử dụng các bản đồ trong *Hình 9* dưới đây với các điểm bắt đầu và mục tiêu được lấy ngẫu nhiên, bán kính tầm nhìn lần lượt là 8, 10 và 15 mét. Các kết quả về con đường mà robot tìm ra được thể hiện như *Hình 10* với các dãy chùm các đoạn thẳng được chỉ ra như *ở Hình 11 và Bảng 3* so sánh thời gian chạy của các thuật toán lập kế hoạch đường đi sử dụng các đường xương cá (skeleton paths) và thuật toán thay thế đường xương

cá bằng các đường đi ngắn hơn (thuật toán sử dụng kỹ thuật của Li và Klette và thuật toán sử dụng phương pháp bắn nhiều lần) xét cho toàn bộ quãng đường robot chuyển động và chỉ xét cho các đường đi trở lại. *Bảng 4* so sánh độ dài của các đường đi thu được bằng thuật toán lập kế hoạch đường đi sử dụng các đường xương cá và thuật toán thay thế đường xương cá bằng các đường đi ngắn hơn (thuật toán sử dụng kỹ thuật của Li và Klette và thuật toán sử dụng phương pháp bắn nhiều lần MMS).

Đầu vào				Thời gian chạy của		Thời gian chạy của thuật toán dùng đường xương cá	Thời gian chạy cho giải bài toán trên dãy chùm		
Bản đồ	Điểm đầu	Điểm cuối	Bán kính	thuật toán dùng kỹ thuật của Li & Klette	thuật toán dùng MMS		dùng kỹ thuật của Li & Klette	dùng MMS	Tỉ số tăng tốc
_map_1	(120,50)	(160,160)	8	186.3656	182.0621	178.6172	4.9058	1.6197	3.03
_map_1	(5,5)	(140,50)	10	14.2539	13.9766	13.8138	0.1175	0.0599	1.96
_map_1	(120,50)	(160,60)	15	39.42	39.2046	38.2075	0.7815	0.3748	2.09
_map_2	(100,25)	(80,145)	8	166.1297	165.8722	157.0573	4.715	3.4893	1.35
_map_2	(10,50)	(30,160)	10	46.0518	45.8075	45.0867	0.39	0.3813	1.02
_map_2	(5,5)	(160,10)	15	37.2069	36.8976	36.2428	0.5592	0.3016	1.84
_map_3	(20,60)	(70,30)	8	54.7647	53.8304	50.2951	1.8857	0.6784	2.78
_map_3	(75,60)	(35,30)	10	40.9327	39.4424	37.3475	2.233	0.3932	5.68
_map_3	(50,20)	(50,-10)	15	16.1423	15.6189	15.0918	0.6728	0.0578	11.64
_map_4	(90,70)	(0,60)	8	44.5927	44.1078	42.6152	0.6975	0.2499	2.79
_map_4	(30,70)	(60,70)	10	56.4147	53.881	51.9044	3.2104	0.2339	13.73
_map_4	(95,60)	(40,50)	15	31.3389	27.8889	26.1923	3.7107	0.188	19.74
_map_5	(40,40)	(20,20)	8	225.1577	224.7189	215.2458	3.7956	1.9902	1.91
_map_5	(60,50)	(20,45)	10	9.5332	8.9903	8.3139	0.6873	0.1059	6.49
_map_5	(20,55)	(30,40)	15	13.4351	13.1511	12.7288	0.3272	0.0446	7.33
_map_6	(80,80)	(28,42)	8	31.8206	30.1569	27.7515	0.9105	0.5241	1.73
_map_6	(20,50)	(30,60)	10	22.0036	20.6986	19.7666	1.5633	0.1715	9.12
_map_6	(20,50)	(50,50)	15	12.8503	12.6942	12.2847	0.2337	0.0506	4.62

Bảng 3: Thời gian chạy của các thuật toán lập kế hoạch đường đi sử dụng các đường xương cá và thuật toán thay thế đường xương cá bằng các đường đi ngắn hơn (thuật toán sử dụng kỹ thuật của Li và Klette và thuật toán sử dụng phương pháp bắn nhiều lần MMS) xét cho toàn bộ quãng đường robot chuyển động và chỉ xét cho các đường đi trở lại.

Các thử nghiệm số như trong B*ảng 3* cho thấy rằng thuật toán sử dụng phương pháp bắn nhiều lần trung bình giảm thời gian chạy khoảng 2,88 % so với thuật toán lập kế hoạch đường đi bằng cách sử dụng kỹ thuật của Li và Klette, mặc dù cả hai phương pháp đều có độ dài đường đi mà robot đi qua là tương đương). Hơn nữa, nếu chỉ xem xét thời gian cho việc quay lại (không phải cho toàn bộ đường đi), Thuật toán sử dụng phương pháp bắn nhiều lần giảm đáng kể khoảng 64,46 % so với thời gian sử dụng kỹ thuật Li và Klette, xem *Bảng 3*. Bên cạnh đó, *Bảng 4* cho biết độ dài đường đi của robot khi so sánh giữa hai cách

tiếp cận: thuật toán lập kế hoạch đường đi sử dụng các quỹ đạo có sẵn và các thuật toán thay thế các đường đi này bằng các phương pháp xấp xỉ (thuật toán sử dụng kỹ thuật của Li và Klette, Thuật toán sử dụng phương pháp bắn nhiều lần). Kết quả cho thấy tổng quãng đường di chuyển của robot giảm trung bình khoảng 16,57 % khi áp dụng phương pháp bắn nhiều lần để tìm các đường đi gần như dọc theo chuỗi các đoạn thẳng được sử dụng thay vì các đường trong biểu đồ. Do đó, việc quay trở lại bằng các con đường ngắn hơn quỹ đạo có sẵn thu được từ đồ thị tích lũy giúp tiết kiệm thời gian cho các robot chuyển động trong thực tế.

Đầu vào				Độ dài các đường thu được bởi thuật toán dùng dùng kỹ thuật Li & Klette	Độ dài các thu được bởi bởi thuật toán dùng MMS	Độ dài các đường đường thu được bởi thuật toán dùng các đường xương cá
Bản đồ	Điểm đầu	Điểm cuối	Bán kính tầm nhìn			
_map_1	(120,50)	(160,160)	8	11509.6869	11507.6238	14088.6523
_map_1	(5,5)	(140,50)	10	1365.673	1365.6031	1647.8448
_map_1	(120,50)	(160,60)	15	6024.8101	6024.4755	7590.9705
_map_2	(100,25)	(80,145)	8	14063.9094	14063.0613	16868.6556
_map_2	(10,50)	(30,160)	10	3908.5088	3908.3194	4692.9537
_map_2	(5,5)	(160,10)	15	4912.8289	4912.5929	5693.1111
_map_3	(20,60)	(70,30)	8	4838.9133	4838.6719	5998.563
_map_3	(75,60)	(35,30)	10	4353.7386	4353.513	5284.6136
_map_3	(50,20)	(50,-10)	15	2113.2647	2113.1411	2371.6619
_map_4	(90,70)	(0,60)	8	1633.3712	1632.9317	1849.1196
_map_4	(30,70)	(60,70)	10	2343.3567	2341.9326	2725.1261
_map_4	(95,60)	(40,50)	15	2190.4232	2189.8435	2676.0038
_map_5	(40,40)	(20,20)	8	14376.3518	14375.5194	19104.875
_map_5	(60,50)	(20,45)	10	1067.6096	1067.3361	1225.8483
_map_5	(20,55)	(30,40)	15	1381.1164	1381.0975	1565.5296
_map_6	(80,80)	(28,42)	8	3906.2147	3905.6391	4787.4772
_map_6	(20,50)	(30,60)	10	2513.966	2513.1652	3052.9031
_map_6	(20,50)	(50,50)	15	1753.6594	1753.6194	2131.6867

Bảng 4: Độ dài của các đường đi thu được bằng thuật toán lập kế hoạch đường đi sử dụng các đường xương cá (skeleton paths) và thuật toán thay thế đường xương cá bằng các đường đi ngắn hơn (thuật toán sử dụng kỹ thuật của Li và Klette và thuật toán sử dụng phương pháp bắn nhiều lần MMS).

Hiệu quả của giải pháp hữu ích/ Những lợi ích giải pháp hữu ích đạt được

Các thí nghiệm số cho thấy phương pháp của chúng tôi giảm thời gian chạy trung bình khoảng 2,88% so với thuật toán sử dụng kỹ thuật của Li và Klette (Bảng 3). Đặc biệt, khi tập trung vào bài toán đường đi ngắn nhất hình học, thuật toán của chúng tôi vượt trội đáng kể so với phương pháp khác. Nếu chỉ xem xét thời gian chạy của việc giải quyết các bài toán phụ nhằm tìm gần đúng các đường đi ngắn nhất dọc theo dãy đoạn thẳng (không

phải cho toàn bộ bài toán), thuật toán sử dụng MMS giảm khoảng 64,46% so với thuật toán sử dụng kỹ thuật của Li và Klette. Ngoài ra, tổng quãng đường robot di chuyển cũng giảm đáng kể, khoảng 16,57%, so với phương pháp không thay thế các quỹ đạo có sẵn từ đồ thị G bằng các đường đi ngắn hơn để quay lại.

Kết quả của giải pháp hữu ích có thể áp dụng vào robot tự hành. Bằng cách tối ưu độ dài đường quay lại một cách nhanh chóng, robot có thể tiếp tục hoạt động hiệu quả hơn. Điều này có thể giúp tiết kiệm năng lượng và do đó làm cho robot hoạt động trong thời gian dài hơn. Chúng tôi đã tiến hành áp dụng các thuật toán nói trên vào robot TurtleBot 3 Burger với các thông số kỹ thuật như Tầm nhìn: 120mm ~ 3,500mm, góc quay 360 độ. Chuyển động của robot được ghi lại trong các video tại https://www.youtube.com/watch?v=HYnNVsh7fpw được trích xuất từ chuyển động thực tế của robot này theo thuật toán sử dụng kỹ thuật của Li và Klete.

Chúng tôi cũng thử nghiệm thành công các thuật toán nói trên vào robot 4 bánh với các thông số kỹ thuật như dàixrộngxcao 25cmx18cmx21cm với trọng lượng 3.5 kg, bán kính bánh xe 6.33cm và vận tốc tối đa 2.1m/s. Thông số về máy tính nhúng NVIDIA Jetson Nano B01, vi điều khiển Raspberry Pico, RP Lidar A1 là 12m, ảm biến gia tốc và góc quay IMU. Chuyển động của robot được ghi lại trong các video tại https://www.youtube.com/watch?v=DdPtubqQ54Y được trích xuất từ chuyển động thực tế của robot này.

Yêu cầu bảo hộ

1. Phương pháp tìm đường quay lại cho robot tự hành dọc theo dãy các chùm đoạn thẳng đã xác định, bằng phương pháp bắn nhiều lần MMS, cụ thể cho ở bước sau đây

(S3) trích xuất dãy chùm đoạn thẳng F và sử dụng phương pháp bắn nhiều lần MMS (Method of Multiple Shooting, MMS) cho việc tìm đường đi dọc theo dãy chùm đoạn thẳng,

bao gồm:

bước (S3.1), robot xác định được điểm p có thứ hạng cao nhất cần trở lại mà p không nằm trong tầm nhìn hiện tại, việc trích xuất được các dãy chùm đoạn thẳng thuộc phần không gian robot sẽ trở lại được tiến hành; đường xương cá và dãy chùm đoạn thẳng gắn với đường xương cá được tạo ra, tại mỗi đỉnh, chỉ cần xét chùm đoạn thẳng nằm trong cung tròn có tâm tại đỉnh chùm và góc của cung không vượt quá π;

bước (S3.2), tối ưu đường quay lại bằng cách tìm đường đi ngắn nhất hình học dọc theo một dãy các chùm đoạn thẳng (F) bằng phương pháp bắn nhiều lần MMS,

bước (S3.3), robot di chuyển theo đường ngắn hơn đã tìm được ở bước trên,

bước (S4), sau khi xác định được đường đi từ q trở lại p, robot chuyển động tới p; gán q=p, nếu điểm đích không thuộc tầm nhìn hiện tại $\overline{B}[q,r]$ của robot, quá trình lại tiếp tục trở lại bước S1;

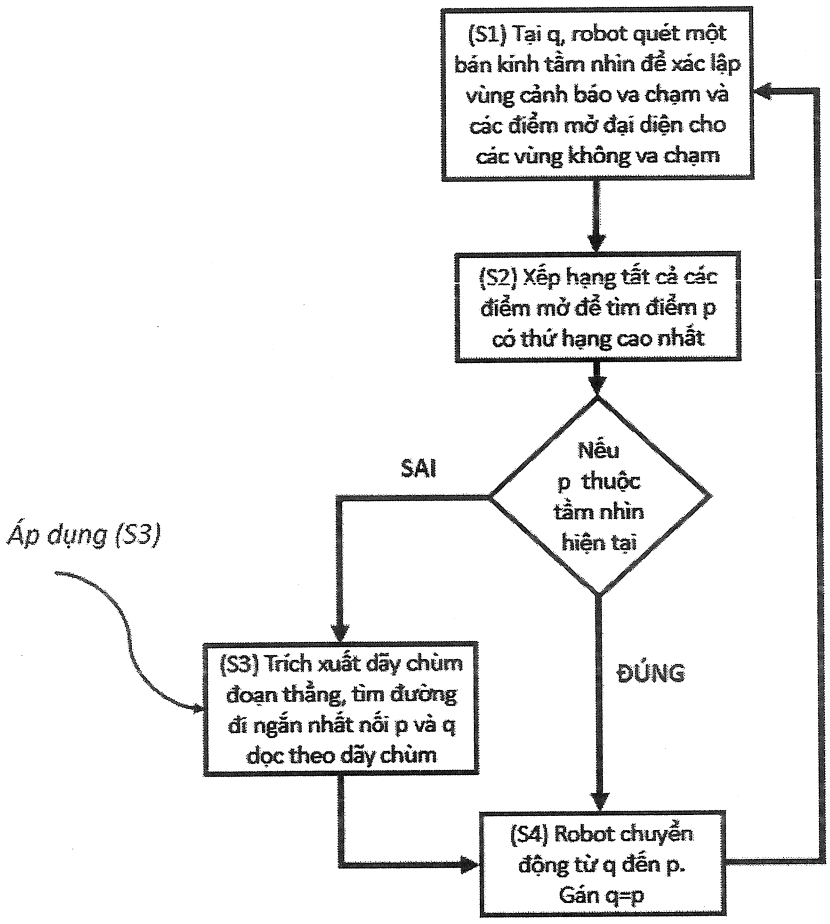
trong đó, bước (S3.2) được thực hiện như sau:

chia dãy các chùm đoạn thẳng (F) thành các dãy con, chọn một tập hợp có thứ tự của các đoạn cắt, là các đoạn cuối của mỗi dãy con, và lấy một điểm bắn đầu tiên trong mỗi đoạn cắt;

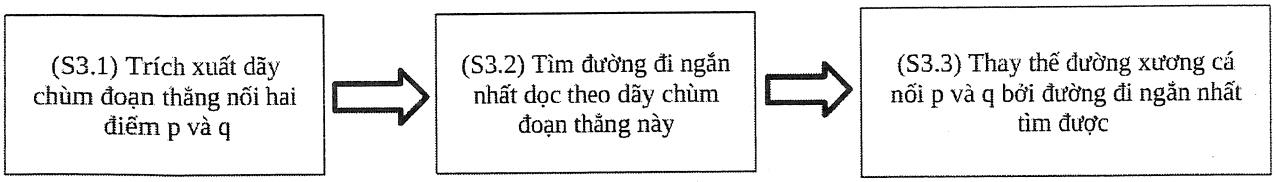
xây dựng một đường dọc theo dãy các chùm đoạn thẳng) nối p và q, được tạo thành từ tập hợp các điểm bắn, đường này là sự kết hợp của các đường ngắn nhất nối hai điểm bắn liên tiếp dọc theo dãy con tương ứng, thiết lập điều kiện dừng tại các điểm bắn;

kiểm tra điều kiện dừng ở tất cả các điểm bắn, nếu điều kiện thẳng hàng không được thỏa mãn, các điểm bắn mới sẽ cải thiện đường nối p và q dọc theo dãy các chùm đoạn thẳng (F), được tạo bởi tập hợp các điểm bắn mới;

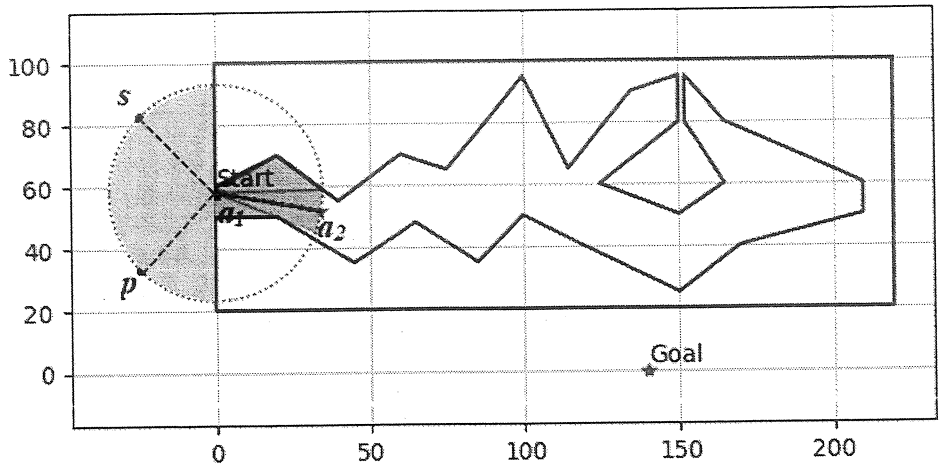
trong đó, thời gian chạy của thuật toán ở bước (S3.2) phụ thuộc vào m, N, J, với: N là số lượng chùm của dãy các chùm đoạn thẳng (F), J là số lượng đoạn thẳng của dãy các chùm đoạn thẳng (F), và m là số lần lặp của thuật toán để tìm được đường đi cần thiết nối giữa p và q.


(S1) Tại q, robot quét một bán kính tầm nhìn để xác lập vùng cảnh báo va chạm và các điểm mở đại diện cho các vùng không va chạm
(S2) Xếp hạng tất cả các điểm mở để tìm điểm p có thứ hạng cao nhất
Nếu p thuộc tầm nhìn hiện tại
SAI
ĐÚNG
Áp dụng (S3)
(S3) Trích xuất dãy chùm đoạn thẳng, tìm đường đi ngắn nhất nối p và q dọc theo dãy chùm
(S4) Robot chuyển động từ q đến p. Gán q=p

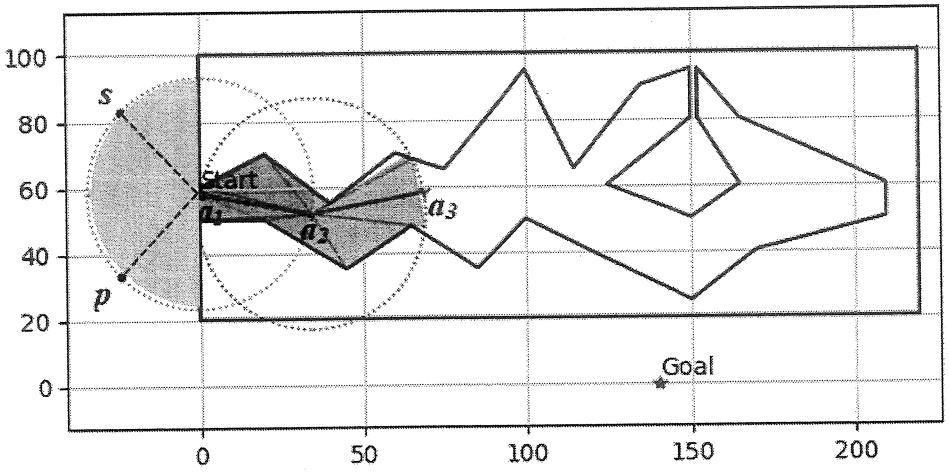
Hình 1


(S3.1) Trích xuất dãy chùm đoạn thẳng nối hai điểm p và q
(S3.2) Tìm đường đi ngắn nhất dọc theo dãy chùm đoạn thẳng này
(S3.3) Thay thế đường xương cá nối p và q bởi đường đi ngắn nhất tìm được

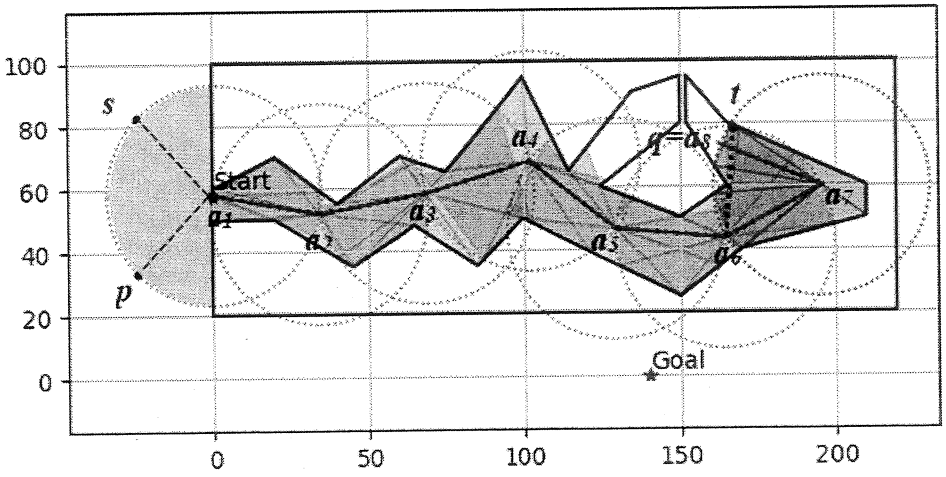
Hình 2



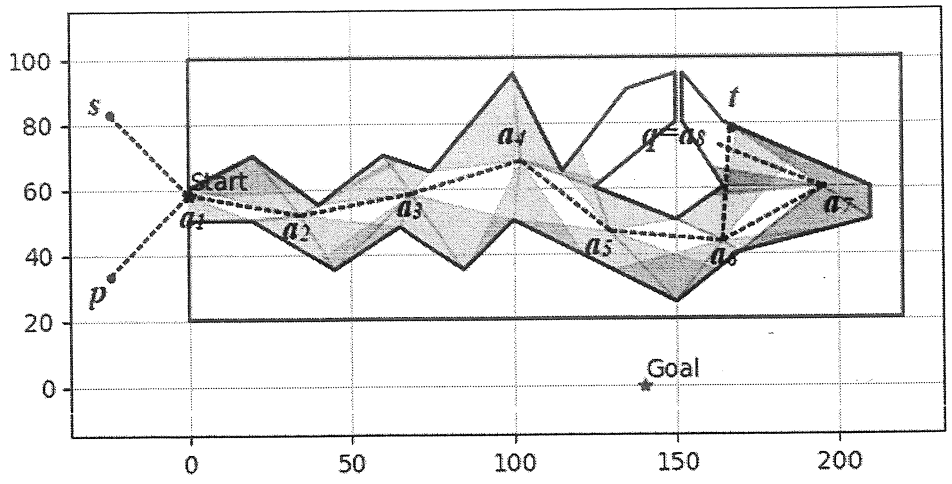
Hình 3



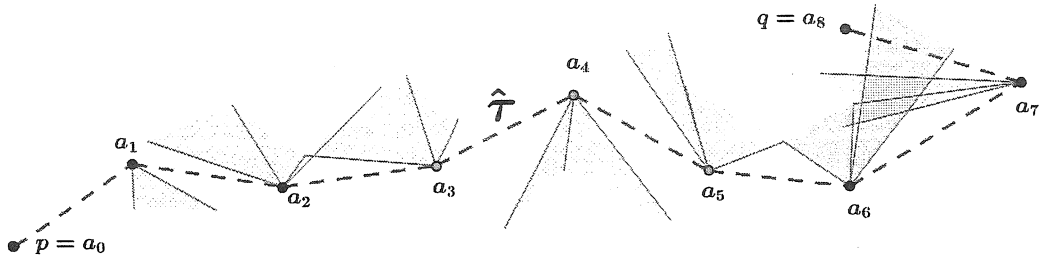
Hình 4



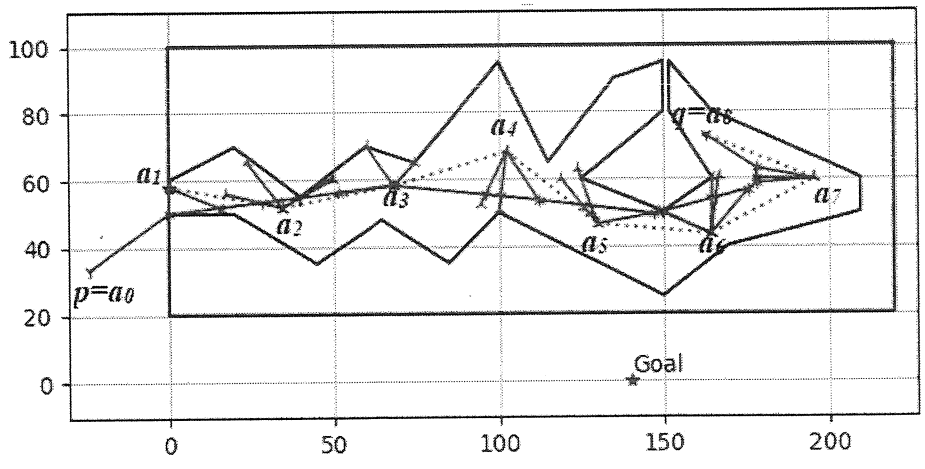
Hình 5



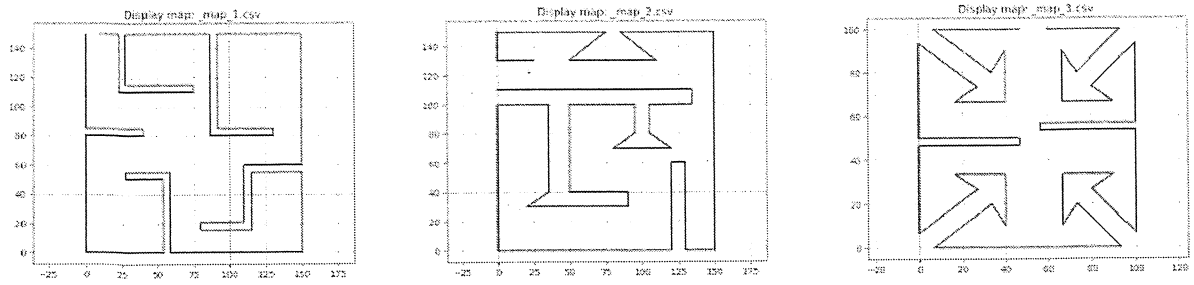
Hình 6



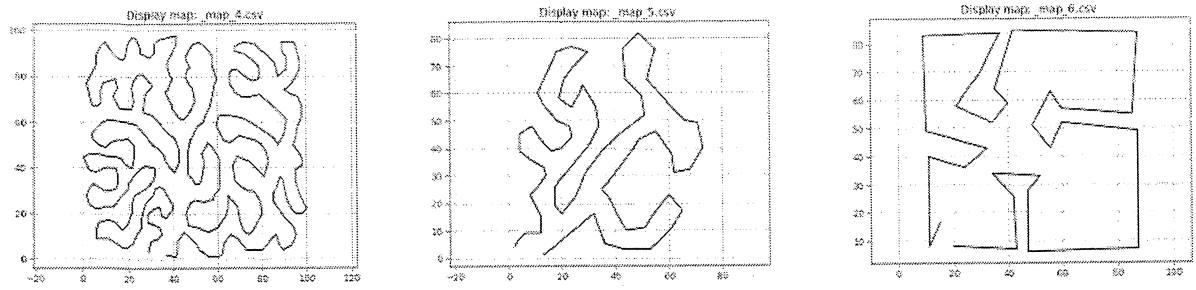
Hình 7



Hình 8

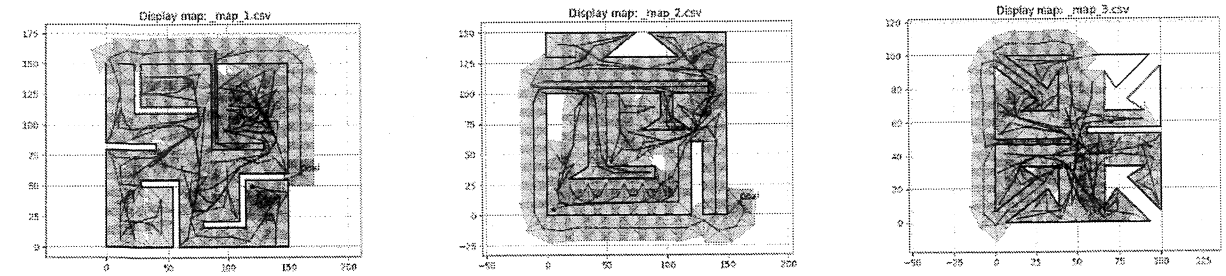


(i) (ii) (iii)



(iv) (v) (vi)

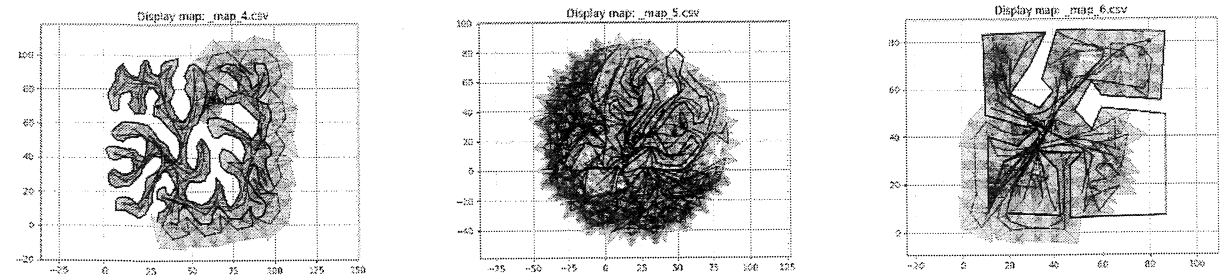
Hình 9



(i): Điểm đầu $(120, 50)$ và đích $(160, 60)$ với bán kính $r = 15$

(ii): Điểm đầu $(5, 5)$ và đích $(160, 10)$ với bán kính $r = 15$

(iii): Điểm đầu $(75, 60)$ và đích $(35, 30)$ với bán kính $r = 10$

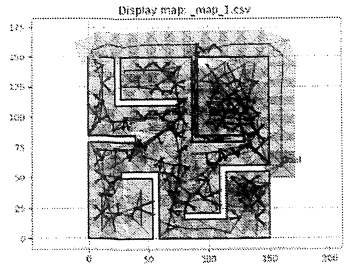


(iv): Điểm đầu $(30, 70)$ và đích $(60, 70)$ với bán kính $r = 10$

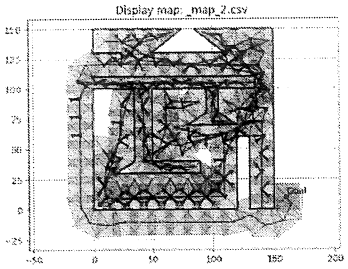
(v): Điểm đầu $(40, 40)$ và đích $(20, 20)$ với bán kính $r = 8$

(vi): Điểm đầu $(80, 80)$ và đích $(28, 42)$ với bán kính $r = 8$

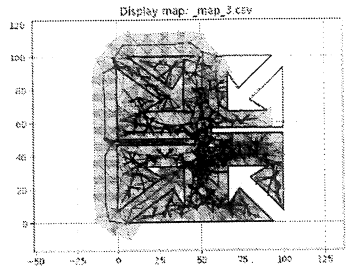
Hình 10



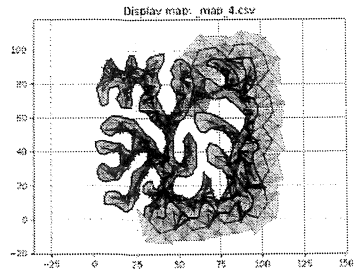
(i): Điểm đầu $(120, 50)$ và đích $(160, 60)$ với bán kính $r = 15$



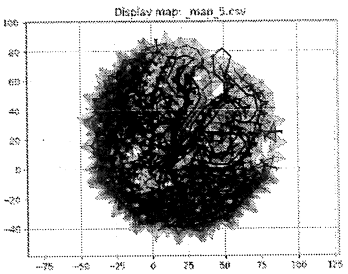
(ii): Điểm đầu $(5, 5)$ và đích $(160, 10)$ với bán kính $r = 15$



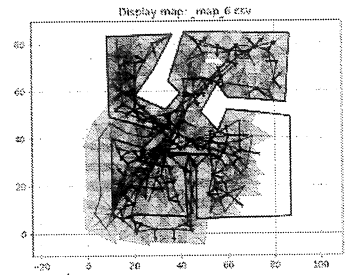
(iii): Điểm đầu $(75, 60)$ và đích $(35, 30)$ với bán kính $r = 10$



(iv): Điểm đầu $(30, 70)$ và đích $(60, 70)$ với bán kính $r = 10$



(v): Điểm đầu $(40, 40)$ và đích $(20, 20)$ với bán kính $r = 8$



(vi): Điểm đầu $(80, 80)$ và đích $(28, 42)$ với bán kính $r = 8$

Hình 11