NGHIÊN CỨU THUẬT TOÁN TÌM ĐƯỜNG ĐI TỐI ƯU CHO ROBOT CÓ KHẢ NĂNG TRÁNH VẬT CẢN TRONG MÔI TRƯỜNG BIẾT TRƯỚC

MOBILE ROBOT FAST PATH PLANNING ALGORITHM FOR OBSTACLE AVOIDING IN KNOWN ENVIRONMEN

Nguyễn Văn Hưởng, TS. **Nguyễn Trọng Doanh**
Trường Cơ khí, Đại học Bách khoa Hà Nội

TÓM TẮT

Thiết lập đường đi cho robot di động có khả năng tránh vật cản là một trong những chủ đề thú vị của các nhà nghiên cứu trong quá khứ, cũng như ngày nay. Tuy nhiên, các công trình nghiên cứu trước đây chủ yếu tập trung vào việc tìm ra quãng đường ngắn nhất giữa vị trí bạn đầu của robot và vị trí mục tiêu của robot trong môi trường biết trước. Con đường tối ưu của robot không chỉ phụ thuộc vào quãng đường đi ngắn nhất mà còn phụ thuộc vào tốc độ di chuyển của robot để đến được vị trí mục tiêu. Lập kế hoạch tìm đường đi tối ưu cho robot với một tốc độ giới hạn có thể cho phép robot di động đến mục tiêu trong khoảng thời gian nhanh nhất. Bài báo này đề xuất một thuật toán lập kế hoạch đường đi với thời gian di chuyển đến mục tiêu nhanh nhất.

Từ khóa: *Robot di động; Thuật toán; Tránh vật cản; Lập kế hoạch đường đi.*

ABSTRACT

The mobile robot path planning for avoiding obstacles is one of the most interesting researching topics. Actuel research works concentrate to find the shortest path between the initial robot position and target position in known or dynamic environment. The optimal path depends not only on the shortest path, but also in speed of mobile robot to goal achieving. To find fast path planning with speed contrain can allow a mobile robot achieve target position in the shortest time. This paper proposed a new fast path planning algorithm for avoiding obstacles with the shortest time target achieving.

Keywords: *Mobile Robot; Algorithm; Path Planning; Obstacle Avoidance.*

1. TỔNG QUAN

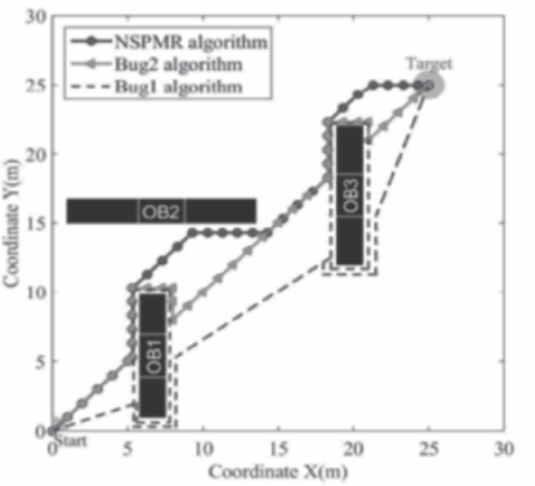
Tìm đường đi ngắn nhất và tối ưu nhất cho robot di động luôn là vấn đề nóng trong lĩnh vực điều khiển robot. Bài toán lập quỹ đạo di chuyển cho robot tự hành có thể chia làm hai dạng: bài toán toàn cục (global) và bài toán cục bộ (local). Trong bài toán toàn cục, môi trường làm việc của robot hoàn toàn đã biết, robot dựa vào đó để tìm quãng đường di chuyển sao cho đúng mục đích của người nghiên cứu. Trong bài toán cục bộ, môi trường làm việc của robot hoàn toàn chưa biết, đối với trường hợp này cần sử dụng đến các cảm biến gắn trên robot để tránh vật cản. Do vậy, thường có hai sự lựa chọn cho đề tài này:

- Một là, robot di chuyển trong môi trường đã biết.

- Hai là, robot di chuyển trong môi trường động hay môi trường không xác định.

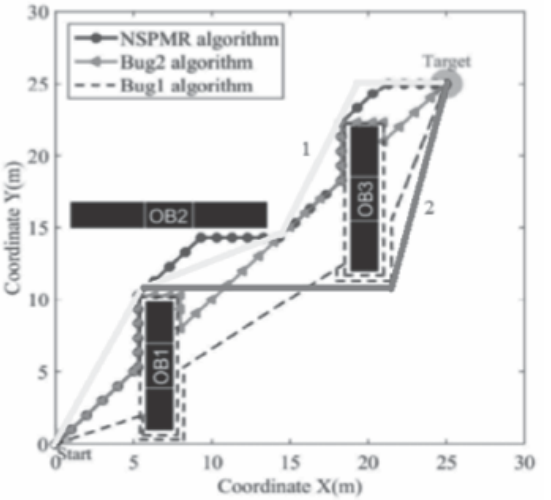
Đã có rất nhiều các nghiên cứu liên quan đến thuật toán tìm đường đi tối ưu cho robot như: Thuật toán Dikstra [7], Bug 1 [5, 7, 9], Bug 2 [5, 7, 9], Thuật toán theo dõi khoảng cách FGA [8], Thuật toán theo dõi khoảng cách thông minh IFGM [3], Thuật toán tìm đường ngẫu nhiên RRT [5, 11], Thuật toán A* [5, 8, 12, 14]. Các công trình nghiên cứu đều tập trung tìm đường đi ngắn nhất cho robot mà không chú ý đến tốc độ di chuyển. Các chướng ngại vật có rất nhiều các hình dạng khác nhau, một số trường hợp rất dễ tìm đường như chữ U, chữ H, hình chữ nhật, tam giác. Tuy nhiên, việc tìm đường mất rất nhiều thời gian và không phải con đường tối ưu.

Dựa trên thuật toán phân tích đường đi toàn cục cho robot [9], thuật toán của tác giả Lê Xuân Hải cho kết quả tốt hơn so với hai thuật toán cổ điển Bug1 và Bug2.



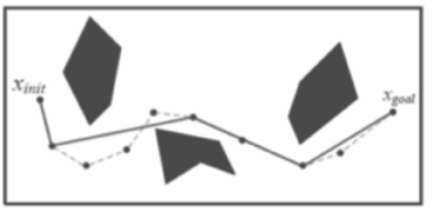
Hình 1.1. Kết quả đường đi của thuật toán NSPMR

Tuy nhiên đây vẫn chưa phải con đường đi ngắn nhất và thời gian đến đích ngắn nhất. Vẫn còn con đường đi tối ưu hơn khi xét đến vận tốc của robot. Đường số 1 màu vàng gồm 4 đoạn thẳng và 3 đoạn cua, đường số 2 gồm 3 đoạn thẳng và 2 đoạn cua. Có thể thấy rằng đường màu vàng có quãng đường di chuyển ngắn hơn tuy nhiên khi xét đến vận tốc chạy của robot thì đường màu đỏ lại có thời gian di chuyển đến đích là nhanh hơn.



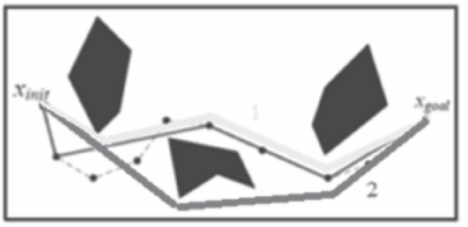
Hình 1.2. Đề xuất đường đi mới 1

Dựa vào thuật toán RRT*, Qi và các cộng sự [11] đã thực hiện một số sửa đổi và đã đạt được một kết quả tốt hơn.



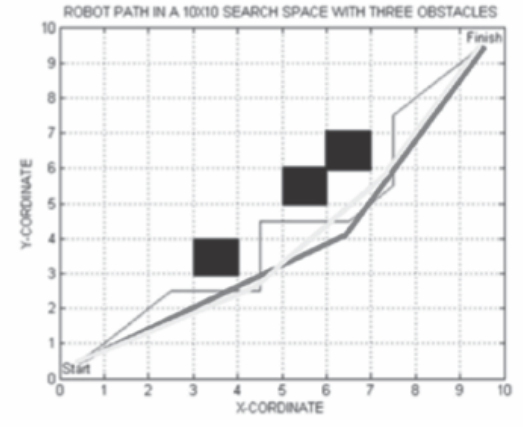
*Hình 1.3. Kết quả đường đi của thuật toán RRT**

Việc tìm kiếm đường đi được tối ưu hóa bằng thuật toán MOD-RRT* ngắn hơn so với thuật toán RRT ban đầu. Tuy nhiên vẫn còn các giải pháp tốt hơn như đường màu vàng và đỏ trong hình dưới.



Hình 1.4. Đề xuất đường đi mới 2

Tương tự đối với thuật toán PSO của Adamu và các cộng sự [2], ta có thể tìm được con đường đi tối ưu hơn.



Hình 1.5. Đề xuất đường đi mới 3

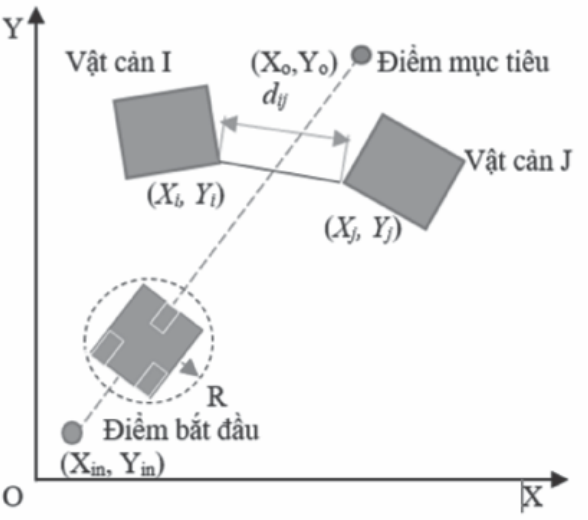
Việc xét đến tốc độ của robot là một yếu tố quan trọng để giảm thiểu thời gian di chuyển tới đích. Bài báo này sẽ xây dựng một thuật toán mới giúp robot di chuyển đến đích nhanh nhất, robot sẽ di chuyển với tốc độ tối đa ở các cung đường thẳng và sẽ di chuyển với vận tốc bằng 70% vận tốc tối đa ở các cung đường cua.

2. ĐỀ XUẤT THUẬT TOÁN TÌM ĐƯỜNG ĐI TỐI ƯU MỚI

2.1. Cơ sở lý thuyết

Thuật toán tránh vật cản với thời gian di chuyển nhanh nhất được giả thuyết như sau:

- Bề mặt phẳng, không trượt;
- Môi trường làm việc đã biết;
- Vị trí, hình dạng của chướng ngại vật xác định;
- Kích thước của robot được xác định bằng hình tròn có bán kính đã biết;
- Tốc độ di chuyển tối đa và tốc độ góc đã biết.



Hình 2.1. Mô hình hóa hệ thống

Đường đi tối ưu nhất là đường thẳng nối điểm ban đầu và điểm mục tiêu. Do đó,

khoảng cách giữa hai vật cản là yếu tố quan trọng quyết định xem robot có thể đi qua giữa các chướng ngại vật hay không. Nếu khoảng cách giữa hai chướng ngại vật lớn hơn hai lần bán kính của robot thì cho phép robot di chuyển qua các chướng ngại vật.

Khoảng cách hay khoảng cách nhỏ nhất giữa hai chướng ngại vật được xác định bằng công thức:

$$d_{ij} = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2} \quad (1)$$

$$2R < d_{ij} \quad (2)$$

Phương trình đường mục tiêu (là đường thẳng nối từ điểm bắt đầu đến điểm kết thúc):

$$Y = \frac{Y_o - Y_{in}}{X_o - X_{in}} X + b \quad (3)$$

Phương trình tiếp tuyến từ một điểm đến các điểm đầu mút của vật cản được bao bởi đường tròn có bán kính R tính từ tâm của robot:

$$Y_i = \frac{Y_o - Y_{iobs}}{X_o - X_{iobs}} X_i + b_i \quad (4)$$

2.2. Sơ đồ khối thuật toán

Thuật toán tìm đường di chuyển tối ưu được thực hiện như sau:

● Bước 1: Thiết lập bản đồ, xác định số vật cản trong môi trường làm việc (N_0), xác định điểm bắt đầu (init_point) và điểm kết thúc (final_point).

● Bước 2: Xác định phương trình đường thẳng từ điểm bắt đầu và điểm kết thúc (đường mục tiêu Y).

● Bước 3: Giải quyết bài toán:

▸ Trường hợp 1: Không có vật cản (N_0 = 0) => Đường tối ưu nhất là đường thẳng đi qua điểm bắt đầu và điểm kết thúc.

▸ Trường hợp 2: Có N là số vật cản (N_0 = N).

Kiểm tra xem đường mục tiêu có cắt vật cản không và tính khoảng cách nhỏ nhất từ vật cản đến đường mục tiêu. Nếu đường mục tiêu không cắt vật cản và khoảng cách nhỏ nhất từ vật cản đến đường mục tiêu lớn hơn R thì đường đi tối ưu là đường mục tiêu Y. Ngược lại, nếu đường mục tiêu cắt vật cản hoặc khoảng cách nhỏ nhất từ vật cản đến đường mục tiêu:

- Xác định các đường tròn tâm I bán kính (R + 0.1R); trong đó I là đỉnh các đầu mút của vật cản, R là bán kính của robot; 0.1R là hệ số an toàn.

- Tạo danh sách các tiếp điểm; trong đó ls_init là danh sách các tiếp điểm kẻ từ điểm bắt đầu đến đường tròn tâm I; ls_final là danh sách các tiếp điểm kẻ từ điểm kết thúc đến đường tròn tâm I.

- Lưu các tiếp điểm p_1 vào ls_init với p_1 thỏa mãn điều kiện: Đoạn thẳng (init_point; p_1) không cắt bất kỳ vật cản nào và đường thẳng này không cắt bất kỳ cạnh nào đối với vật cản đang xét.

- Lưu các tiếp điểm p_2 vào ls_final với p_2 thỏa mãn điều kiện: Đoạn thẳng (final_point; p_2) không cắt bất kỳ vật cản nào và đường thẳng này không cắt bất kỳ cạnh nào đối với vật cản đang xét.

- Xác định giao điểm (intersec_point) giữa hai đường thẳng (init_point; p_1) với (final_point; p_2).

+ Nếu tồn tại giao điểm (intersec_point) thỏa mãn điều kiện: Đoạn thẳng (init_point; intersec_point) không cắt vật cản nào và khoảng cách từ đoạn thẳng này đến vật cản lớn hơn R; đồng thời đoạn thẳng (final_point; intersec_point) không cắt vật cản nào và khoảng cách từ đoạn thẳng này đến vật cản lớn hơn R.

+ Tính khoảng cách từ init_point đến điểm giao và điểm final_point (distance_path).

+ Tính khoảng cách từ điểm init_point tới P_1 + khoảng cách p_1 tới p_2 + khoảng cách từ p_2 tới final_point (distance_over_tangent_point).

+ Nếu distance_path - distance_over_tangent_point ≤ 6R thì thêm intersec_point vào danh sách ls_tangent_p và ngược lại.

- ls_tangent > 0 thì:

+ Tính toán quãng đường di chuyển.

+ Tính thời gian di chuyển với giả thiết đến khúc cua vận tốc di chuyển của robot bằng 70% vận tốc tối đa khi di chuyển trên đoạn thẳng.

$$v = v_{max}(1 - 0.7\cos\alpha)$$

Trong đó, α là góc quay của robot.

+ So sánh thời gian di chuyển và chọn quãng đường di chuyển có thời gian ngắn nhất.

- Nếu ls_tangent = 0 thì:

+ Xác định p_1_tangent, p_2_tangent lần lượt là tiếp điểm giữa hai đường tròn có tiếp điểm trong danh sách của ls_init và ls_final.

+ Xác định p_1_new là giao điểm của hai đường thẳng (init_point; p_1) và (p_1_tangent, p_2_tangent).

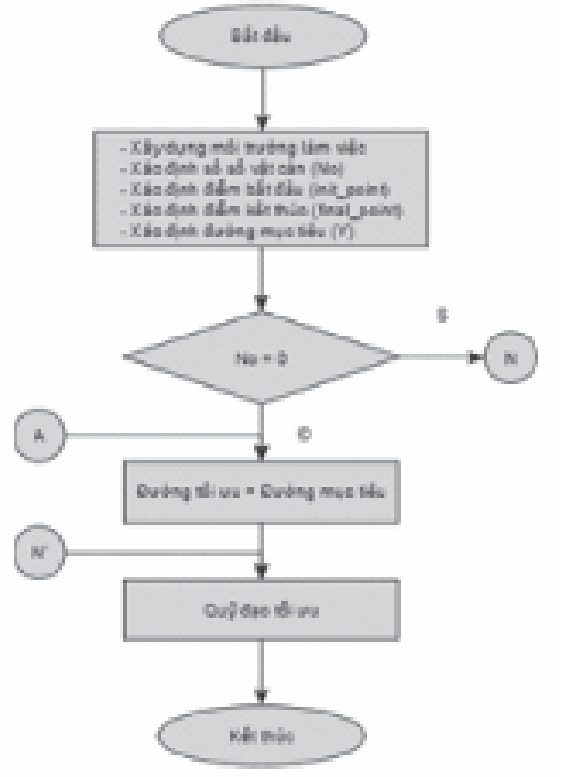
+ Xác định p_2_new là giao điểm của hai đường thẳng (final_point; p_2) và (p_1_tangent, p_2_tangent).

Đoạn thẳng (p_1_new; p_2_new) thỏa mãn điều kiện: Không cắt bất kỳ vật cản nào và khoảng cách giữa đoạn thẳng này so với vật cản phải lớn hơn R).

+ Tính khoảng cách từ điểm init point tới p_1_new + khoảng cách p_1_new tới p_2_new + khoảng cách từ p_2_new tới final point.

+ Tính thời gian di chuyển.

+ So sánh thời gian di chuyển và chọn quãng đường di chuyển có thời gian ngắn nhất.



ISSN 2615 - 9910 (bản in), ISSN 2815 - 5505 (online)
TẠP CHÍ CƠ KHÍ VIỆT NAM, Số 309, tháng 11 năm 2023
cokhivietnam.vn / tapchicokhi.com.vn

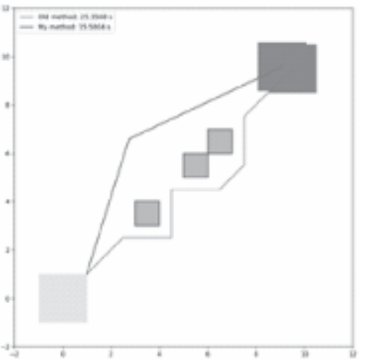
Hình 2.2. Sơ đồ khối của thuật toán

3. KẾT QUẢ VÀ THẢO LUẬN

Tiến hành mô phỏng thuật toán với các thông số đầu vào như sau:

- Bán kính của robot: R = 1;
- Vận tốc max: v_{max} = 1;
- Vận tốc khúc cua v = 70%v_{max}.

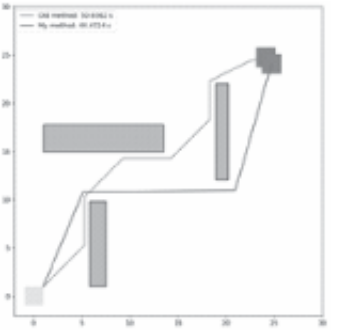
❖ Đối với thuật toán PSO của Adamu và các cộng sự:



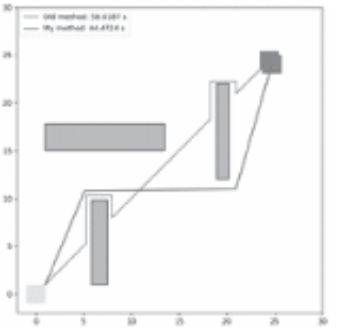
Hình 2.3. Kết quả đường đi so với PSO

⇨ Thuật toán đưa ra quãng đường đi chỉ với 1 khúc cua, trong khi thuật toán PSO đưa ra quãng đường di chuyển với 6 khúc cua, điều đó dẫn đến thời gian di chuyển của thuật toán mới là 15.6s còn thuật toán PSO di chuyển với thời gian 25.4s.

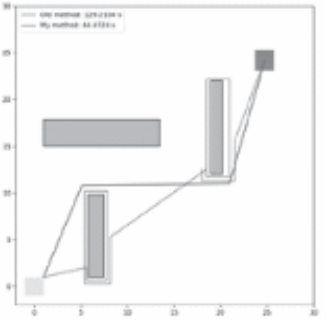
Đối với thuật toán NSPRM của Lê Xuân Hải và các cộng sự:



Thuật toán NSPRM



Thuật toán Bug1



Thuật toán Bug2

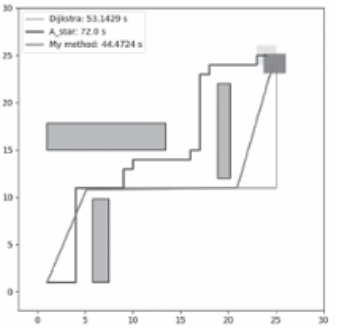
Hình 2.4. Kết quả đường đi so với NSPRM, Bug1, Bug2

Ta có bảng so sánh các thuật toán như sau:

Bảng 1.1. Bảng so sánh thời gian di chuyển giữa các thuật toán

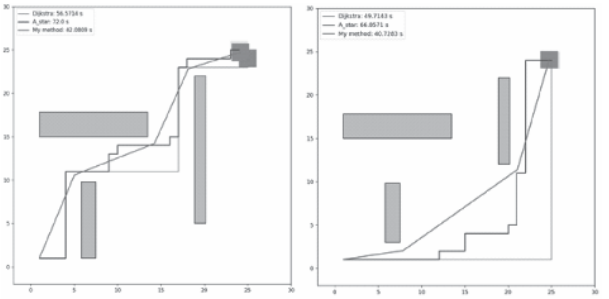
Thuật toán	My method	NSPRM	Bug2	Bug1
Thời gian (s)	45.5	50.7	58	129.2

Bên cạnh đó, tôi cũng thực hiện so sánh thuật toán của mình với các thuật toán cổ điển như A start, Dijkstra và đều thu được kết quả tốt hơn:



Hình 2.5. Kết quả đường đi so với thuật toán A và Dijsktra*

Một số trường hợp khác của thuật toán:



Hình 2.6. Một số trường hợp tìm đường khác

4. KẾT LUẬN

Thuật toán mới tìm đường đi tối ưu cho robot mang một ý tưởng mới mẻ, sử dụng hình học để giải quyết bài toán. Và kết quả của thuật toán này luôn cho thời gian di chuyển tốt hơn so với các thuật toán cổ điển trước đây như A*, RRT, Bug1, Bug2…. Ta có thể áp dụng thuật toán này vào các công việc của đời sống hàng ngày rất hiệu quả, đặc biệt là trong chiến đấu và tác chiến điện tử khi yêu cầu cần thời gian di chuyển đến đích nhanh nhất từ các đơn vị chỉ huy.

Lời cảm ơn:

Nghiên cứu này được tài trợ bởi Trường Cơ khí. Tác giả xin cảm ơn Trường Cơ khí, Đại học Bách Khoa Hà Nội đã hỗ trợ thời gian và cơ sở vật chất cho nghiên cứu này.❖

Ngày nhận bài: **15/9/2023**
Ngày phản biện: **09/10/2023**

Tài liệu tham khảo:

[1]. P.I. Adamu, H.I. Okabue, P.E. Oguntunde, *Fast and Optimal Path Planning Algorithm for a Mobile Robot, Wiless Personal Communications,* Springer online published 14 february 2019 44.

[2]. M. Basavanna, M. Shivakumar, *An Overview of Path Planning and Obstacle Avoidance Algorithm in Mobole Robot,* IJERT 81S120252, Vo. 8, I. 12, 2019, p 478-482.

[3]. R. Tirumalapudi, R. Vedaraj, *A New Shortest Path Planning Algorithm For Mobile Robot Navigation In Known Terain,* IJSTR, Vo 9, Is.03, 2020, pp 1044-1047.

[4]. A. Janis, A. Bade, Path Planning Algorithm in Complex Environnemt: *A Survey, Transaction on Science and Technology*, 2016, Vo. 3, Is.1, pp31-40.

[5]. Hoc Thai Nguyen, Hai Xuan Le, *Path Planning and Obstacle Avoidance Approaches for Mobile Robot,* IJCSI, Vo.13, Is. 4, July 2016, p 1-10.

[6]. J. Qi, H. Yang, H. Sun, MOD_RRT*: *A Sampling-based algorithm for robot path planning in dynamic environnment,* IEEE Transaction on Industrial Electronics.

[7]. F. A Raheem, M. I. Abdulkareem, *Development of A* Algorithm for robot path planning based on modified probabbilistic roadmap and artificial potential field,* Journal of Engineering Science and Technology, Taylor's University, Vo. 15.No. 5, 2020, p. 3034-3054.