مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (39) العدد (3) 2017
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (39) No. (3) 2017

تخطيط حركة روبوت متنقل باستخدام خوارزميات Bug

د. محسن داود*
د. نزار عبد الرحمن**
هبه حليوه***

(تاريخ الإيداع 19 / 1 / 2017. قُبِل للنشر في 25/ 5 / 2017)

□ ملخّص □

يعتبر تخطيط الحركة من القضايا الهامة والملحة لما يعطيه للروبوت من قدرة على الوصول إلى الهدف بشكل آلي ومنع الاصطدام بأي عائق، مما يرفع من أداء الروبوت ويقلل من كلفته التشغيلية. ويقسم تخطيط حركة الروبوت عادةً إلى قسمين: يقوم الأولبإيجاد المسار المناسب ويضمن الثاني تتبع الروبوت لهذا المسار وصولاً لهدفه.

يعتمد هذا البحث على التقنيات المستخدمة في أحد أشهر خوارزميات تجنب العوائق (خوارزميات Bug) من أجل إيجاد مسار عام للروبوت. وتطبق تلك المسارات الناتجة على أرض الواقع باستخدام روبوت Boe–Bot تفاضلي القيادة.

الكلمات المفتاحية: تخطيط الحركة، تخطيط المسار، خوارزميات Bug، روبوت تفاضلي الحركة.

* أستاذ – قسم هندسة الميكاترونيك – كلية الهندسة الميكانيكية والكهربائية – جامعة تشرين – اللاذقية –سورية.
** أستاذ مساعد – قسم هندسة الميكاترونيك – كلية الهندسة الميكانيكية والكهربائية – جامعة تشرين – اللاذقية –سورية.
*** طالبة دكتوراه – قسم هندسة الميكاترونيك – كلية الهندسة الميكانيكية والكهربائية – جامعة تشرين – اللاذقية –سورية.

مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (39) العدد (3) 2017
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (39) No. (3) 2017

Mobile Robot Motion Planning using Bug Algorithms

Dr. Mohsen Daoud*
Dr. Nizar Abdulrahman**
HibaHliwa***

(Received 19 / 1 / 2017. Accepted 25 / 5 / 2017)

□ ABSTRACT □

Motion Planning is an important and potential issue in Robotics, because it gives the robot the ability to reach its target automatically with collision free, which increase the robot performance and reduce its operational cost. The robot motion planning is commonly divide into two approaches: finding the appropriate path and making the robot tracking this path until it reaches its goal.

This research depends on one of the most common obstacles avoidance path planningtechniques (Bug algorithms) to find global path of robot. And make the Boe-Bot robot (differential drive robot) tracking the paths generated by these algorithms in a certain environment.

Keywords: motion planning, path planning, Bug algorithms, differential drive robot

*** Professor, Department of Mechatronics, Faculty of Mechanical and Electrical Engineering, Tishreen University, Lattakia, Syria.**
**** Professor Assistant, Department of Mechatronics, Faculty of Mechanical and Electrical Engineering, Tishreen University, Lattakia, Syria.**
***** Postgraduate Student, Teacher Assistant, Department of Mechatronics, Faculty of Mechanical and Electrical Engineering, Tishreen University, Lattakia.**

مقدمة:

ينتمي مصطلح تخطيط المسار path planning إلى مجالات علمية مختلفة كالروبوتات والذكاء الصنعي ونظرية التحكم. وهذا ما جعل تعريفه يختلف من علم إلى آخر [1]. فيرتبط تخطيط المسار في علم الروبوتات بمشكلة تحريك الروبوت من مكان إلى آخر والتي ترتبط بمجموعة من التعقيدات كالتحكم بعدة روبوتات في آن واحد [2]ووجود الروبوت في بيئة متغيرة [3]. أما في مجال الذكاء الصنعي فيعبر تخطيط المسار عن البحث عن سلسلة من الأفعال المنطقية التي تنقل الروبوت من حالة ابتدائية initial state إلى الهدف المرغوب desired goal state. قد يتضمن هذا النوع من تخطيط المسار بعض أفكار نظرية اتخاذ القرار decision–theoretic كعمليات ماركوف لاتخاذ القرار Markov decision processes[4] وأساليب التعلم learning methods[5]. في حين تُعنى نظرية التحكم بالمسائل المتعلقة بالاستقرار Stability[6]والتغذية العكسية Feedback[7] والأمثلة Optimality[8].وهذا ما يجعل تخطيط مسار روبوت متنقل مسألة واسعة ومتشعبة.

يُقسم تخطيط الحركة motion planningفي علم الروبوتات عادة إلى تخطيط المسار path planning وتخطيط الطريق trajectory planning واللذان يعرَّفان وفق [1] بأنهما الأسس اللازمة لتحويل المهمة التي يحتاجها الانسان من الروبوت إلى أوامر حركية مفهومة من قبل الروبوت. ففي الوقت الذي يهمل فيه تخطيط المسار ديناميكية الروبوت والقيود الأخرى المفروضة على الحركة ويركز على إيجاد الانتقالات الانسحابية والدورانية اللازمة للكائن المتحكم به لإنجاز الحركة المطلوبة منه. يقوم تخطيط الطريق بالانطلاق من الاعتبارات والقيود الميكانيكية للروبوت في تحديد كيفية حركته وفق المسار الموضوع من قبل خوارزميات تخطيط المسار.

تقسم خوارزميات تخطيط المسار من جهة أخرى إلى قسمين أساسيين [9] هما تخطيط المسار العام Global Path Planning وتخطيط المسار المحلي Local Path Planning. يتطلب تخطيط المسار العام معرفة تامة بالبيئة المحيطة بالروبوت والعوائق الثابتة الموجودة فيها. وتقوم تلك الخوارزميات بإيجاد كامل المسار بين نقطة تواجد الروبوت والنقطة الهدف قبل أن يبدأ الروبوت بالحركة. أما عندما تكون بيئة عمل الروبوت غير معروفة بشكل كامل فعلى الروبوت في هذه الحالة جمع معلومات عن البيئة المحيطة به في الزمن الحقيقي ومن ثم تحديث آلية التحكم الخاصة به بما يسمح بتحقيق هدفه المطلوب منه وهذا ما يُعرف بتخطيط المسار المحلي[10].

تعد خوارزميات Bug من الخوارزميات التي تجمع بين التخطيط العام (من حيث امتلاك الروبوت لهدف يسعى للوصول إليه) والمحلي للمسار (البيئة غير معروفه بشكل كامل للروبوت). فهي تسمح للروبوت بالوصول لغايته المنشودة وتسمح في نفس الوقت للحساسات الموجودة على الروبوت بتحسس البيئة المحيطة واتخاذ القرار المناسب لتجنب العوائق.

أهمية البحث وأهدافه:

تعددت استخدامات الروبوتات المتنقلة المستقلة بشكل كبير في الآونة الأخيرة. وتنوعت بين الاستخدامات العسكرية والصناعية وغيرها. كما أنها تستخدم بشكل واسع في إنجاز المهام الخطرة على حياة الإنسان كإصلاح المنشآت النووية وعمليات البحث والإنقاذ بعد الكوارث وغيرها [11].ويُعدّ تخطيط مسار الروبوتات أحد الأمور الأساسية اللازمة لتمكين الروبوت من القيام بالمهمة الموكلة إليه.

يهدف هذا البحث إلى الاستفادة من خصائص خوارزمية Bugالتي تسمح للروبوت بالقيادة في بيئة غير معلومة بشكل كامل وتمكنه من الوصول إلى الهدف مع تجنب العوائق التي قد يصادفها واختبار أدائها كخوارزمية تخطيط مسار عام (إيجاد المسار في بيئة معروفة بشكل كامل قبل أن يبدأ الروبوت بحركته). ومن ثم تطبيق هذه الخوارزمية على روبوت متنقل واختبار أدائها على أرض الواقع.

طرائق البحث ومواده:

يعتمد هذا البحث على Raspberry pi (وهو حاسوب آلي متكامل مصنوع من شريحة الكترونية واحدة تحتوي على مكونات الحاسب التقليدي من معالج CPU بسرعة 700ميجاهرتز، وذاكرة عشوائية RAM تصل إلى 512 ميجابايت، بالإضافة لمخارج تحكم رقمية General Purpose Input Output(GPIO) يمكنها التحكم بالقطع الالكترونية والكهربائية. ويتم تشغيل هذا الحاسوب الصغير باستخدام نظام Raspbian Wheezy وهو أحد أنظمة لينكس Linux مفتوحة المصدر) . وقد أنجزت خوارزمية Bug باستخدام برنامج Octave المنصب داخل الـRaspberry pi.

ينقسم العمل في هذا البحث إلى عدة أقسام تبدأ باستعراض خوارزميات Bugكأحد خوارزميات تخطيط المسار المطروحة ومن ثم تنطلق من تخطيط الطريق المعتمدة على آلية الحركة لدى روبوت Boe-Bot(وهو من انتاج شركة Parallax والموجود ضمن مخابر قسم الميكاترونيك) لتطبيق تلك الخوارزميات واختبارها على أرض الواقع.

خوارزميات تخطيط مسار روبوت:

تنوعت خوارزميات تخطيط مسار الروبوت المتنقل تبعاً لطبيعة البيئة المحيطة بالروبوت ونوع الحساسات وقدرات الروبوت وغيرها [12]. وسعت كل الخوارزميات للوصول إلى أداء أفضل فيما يتعلق بالزمن والمسافة والكلفة ودرجة التعقيد. صممت خوارزميات Bug كخوارزميات تخطيط مسار محلي تعتمد على الحساسات المثبتة على الروبوت في تحسس البيئة المحيطة عندما تكون المعلومات حول البيئة قليلة جداً. وتطورت تلك الخوارزميات بعدها لتستخدم في تخطيط المسار العام.

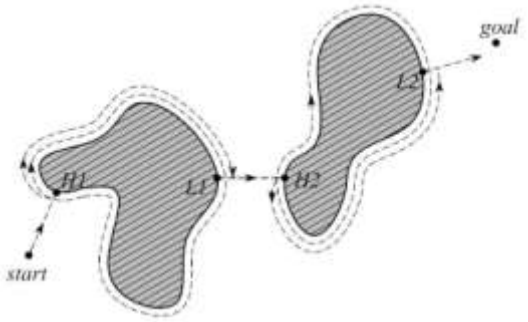
خوارزمياتBug:

تعتبر خوارزميات Bug من الخوارزميات الأساسية المتكاملة [12]، لأنها تسمح للروبوت بالوصول إلى غايته. وتعتمد في ذلك على سلوكين مختلفين هما السعي نحو الهدف goal pursuing mode وتتبع الحدود boundary following mode. تشكل مرحلة السعي نحو الهدف الحالة العامة للروبوت والتي يتحرك فيها بشكل مباشر ومستقيم نحو الهدف ما لم يصادف في طريقه عائقاً. عندما يواجه الروبوت عقبةً ما فإنه ينتقل إلى مرحلة تتبع الحدود ولا يعود لمرحلة السعي نحو الهدف حتى تتحقق شروط تبديل معينة. وفي حال لم يكن للروبوت أي معرفة بالبيئة المحيطة (ولو كانت معرفة جزئية) يمكن من خلالها إيجاد صلة تربط نقطة البداية عن الهدف فإن الخوارزمية ستتعرف على تلك الحالة وتُنهي عملية البحث عن مسار وتُقرر أن الهدف لا يمكن الوصول إليه [13].وتنوعت خوارزميات Bugحتى وصلت إلى 11 طريقة[14]، إلا أن أشهر تلك الخوارزميات المستخدمة في تخطيط مسار روبوت متنقل هما خوارزميتا Bug1 وBug2.

خوارزمية Bug1:

تُعدّ خوارزمية Bug1 من أوائل خوارزميات تجنب العوائق obstacle avoidance وبرغم بساطة تلك الطريقة إلا أنها قد تذهب بالروبوت بعيداً عن الهدف في بعض الحالات [15]. ففي هذه الخوارزمية وعند تحسس الروبوت لعائق فإنه يقوم بتتبع حدود هذا العائق حتى يصل إلى النقطة التي بدأ منها وخلال عملية التتبع يقوم بحساب المسافة بين النقطة الحالية والهدف ومن ثم يخزن النقطة ذات المسافة الأقل وذلك بعد دورة كاملة للروبوت حول العائق. يعود الروبوت بعدها لتتبع حواف العائق حتى يصل إلى النقطة المختارة وينتقل بعدها من مرحلة تتبع الحدود إلى مرحلة السعي نحو الهدف عن طريق رسم مسار مستقيم انطلاقاً من النقطة المختارةووصولاً إلى الهدف المرغوب . ويتبع الروبوتبعدها هذا الخط المستقيم حتى يصل لهدفه أو يصادف عائقاً آخر في طريقهكما يظهر في الشكل(1).

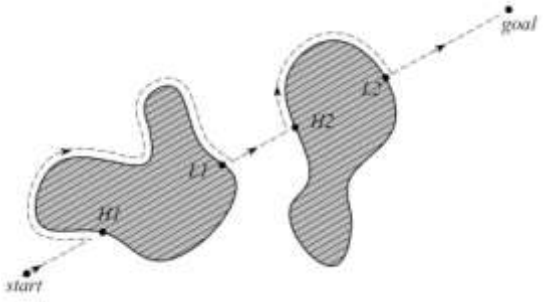
يسلك الروبوت باستخدام هذه الخوارزمية طريقاً طويلاً جداً مقارنة بالمسار المطلوب إلا أنه يستطيع دائماً الوصول إلى هدفه طالما أن هناك مساراً ولو وحيداً للوصل إليه.



الشكل (1) خوارزمية Bug1[16].

خوارزمية Bug2:

أتت خوارزمية Bug2 كتحسين للخوارزمية السابقة وذلك من خلال توليد مسار ابتدائي بين نقطة البداية والهدف وتخزين ميل ذلك المسار كأساس لمرحلة السعي نحو الهدف. ويتغير سلوك الروبوت إلى مرحلة تتبع الحدود عندما يصادف عائقاً في طريقه، عندها يبدأ بتتبع حواف العائق مع حساب مستمر لميل النقطة التي يتواجد فيها الروبوت مع الهدف. وعندما يتساوى ذلك الميل مع ميل المسار الابتدائي عندها يعود الروبوت إلى مرحلة السعي نحو الهدف كما يظهر في الشكل(2).يقوم الروبوت في تلك الخوارزمية بسلوك المسار مرة واحدة فقط وهذا ما يجعل خوارزمية Bug2 أفضل وأسرع من خوارزمية Bug1في أغلب الأحيان.



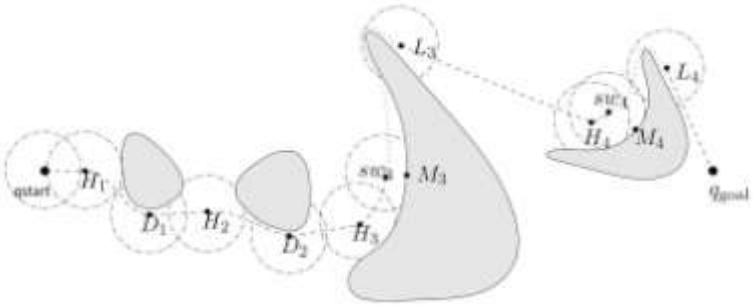
الشكل (2) خوارزمية Bug2[16].

يعد تحديد نقطة تحول الروبوت من مرحلة تتبع العائق إلى مرحلة السعي وراء الهدف "leave point"من الأمور الهامة جداً في تقييم خوارزمية[17]Bug2. فإذا لم تتوضع تلك النقطة في مكانها الأمثل فإن ذلك سيؤثر بشكل واضح على طول المسار. كما أن الاختيار السيء لنقطة المغادرة يمكن أن يدخل الروبوت في حلقات لا نهائية.

خوارزمية TangentBug:

تعتمد خوارزمية TangentBug على حساسات المسافة في بناء مخطط للمحيط الحالي للروبوت واستخدام تلك المعلومات في تقليل طول المسار قدر الإمكان [18]. تحاكي هذه الخوارزمية روبوتاً مثبتاً عليه حساساً Range Sensorقادراً على تحسس كل العوائق المحيطة بالروبوت.

يتجه الروبوت في هذه الخوارزمية نحو الهدف ما لم يتحسس عائقاً، وعندما يصادف في طريقه عائقاً فإنه يبحث عن النقطة التي تقلل من الكلفة الارشادية Heuristic Cost (المسافة بين الروبوت والنقطة التي تم تحسسها مجموعاً له المسافة بين تلك النقطة والهدف). عندما تبدأ الكلفة بالتزايد فإن الروبوت يبدأ بتتبع حواف العائق. يتحول الروبوت إلى نمط السعي إلى الهدف عندما تصبح كلفة السعي إلى الهدف أقل من كلفة تتبع الحدود كما يظهر في الشكل (3).



الشكل (3) خوارزمية [16]TangentBug.

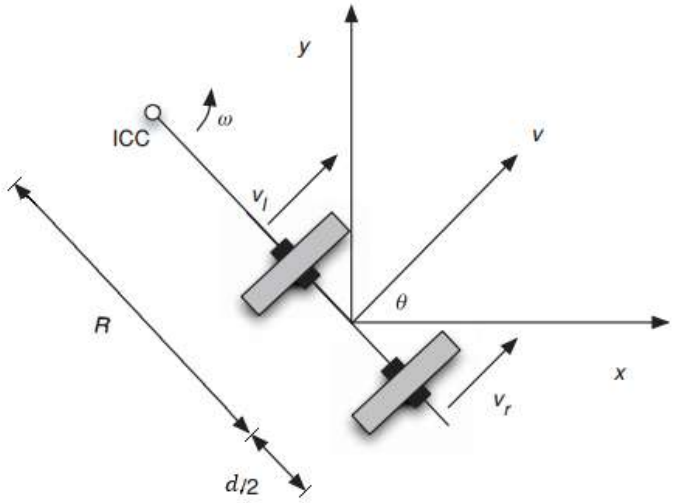
تخطيط الطريق

يهدف تخطيط الطريق إلى توليد الدخل المرجعي لمنظومة التحكم بالروبوت بما يضمن تحقيق الحركة المطلوبة منه. تعتمد خوارزميات تخطيط الطريق على المسار المولد من قبل خوارزميات تخطيط المسار بالإضافة للاعتبارات الديناميكية والحركية للروبوت للوصول إلى سلسلة من قيم الموضع والسرعات والتسارعات الخاصة بكل مشغل في الروبوت[20].ويتعامل هذا البحث مع روبوت تفاضلي القيادة differential drive robot.

القيادة التفاضلية للروبوت المتنقل:

تعد العجلات التفاضلية أحد أكثر طرق الحركة شيوعاً للروبوتات المتنقلة المستقلة. ويعتبر التحكم بتلك العجلات من أهم القضايا المطروحة عند التحكم بالروبوت المتنقل لتتبع مسار ما موضوع من قبل خوارزميات تخطيط المسار.

يمتلك الروبوت تفاضلي القيادة عجلتين يمنى ويسرى مستقلتين بعضهما عن بعض في الحركة. وغالباً ما تكون هناك عجلة إضافية غير فعالة passive wheel في مقدمة الروبوت لجعل حركة الروبوت سلسةكما يبدو في الشكل(4).



الشكل (4) نموذج روبوت تفاضلي القيادة[22].

وللتحكم بهذا النوع من الروبوتات يجب التحكم بسرعة كل من العجلتين الخلفيتين اليمنى واليسرى V_R و V_L على حدة[21]. وبما أن كلتا عجلتي القيادة تتوضعان على المحور نفسه لذلك فإن الروبوت يدور حول نقطة واقعة على المحور الواصل بين العجلتين تدعى بمركز الانحناء اللحظي Instantaneous Center Of Curvature(ICC). وبتغيير السرعة النسبية بين العجلتين قد يتغير موقع مركز الانحناء اللحظي مما يتولد عنه مسارات مختلفة للروبوت. فمن أجل أي لحظة يدور بها الروبوت لا بد أن تتحرك العجلتان اليمنى واليسرى حول ICC بنفس السرعة الزاوية ω وفق المعادلتين (1) و(2)

$$V_R = \omega(R + \frac{d}{2}) \qquad (1)$$

$$V_L = \omega(R - \frac{d}{2}) \qquad (2)$$

تعبر R عن نصف قطر الدوران (المسافة بين مركز الانحناء اللحظي ومنتصف المسافة بين العجلتين) و d هي المسافة بين مركزي العجلتين. كما يمكن إيجاد قيم R و ω وفق المعادلتين (3) و(4):

$$R = \frac{d}{2}.\frac{V_L + V_R}{V_L - V_R} \qquad (3)$$

$$\omega = \frac{V_L - V_R}{d} \qquad (4)$$

هناك حالتان خاصتان لقيمة نصف قطر الانحناء R. فعندما تكون $V_L = V_R$ تكون قيمة نصف القطر لا نهائية وهذا يعني أن الروبوت يتحرك بخط مستقيم أما عندما تكون $V_L = -V_R$ يكون عندها نصف القطر مساوٍ للصفر ويدور الروبوت في هذه الحالة حول النقطة الواقعة في منتصف المسافة الواصلة بين العجلتين أي أنه يدور في مكانه.

أما من أجل القيم الأخرى ل V_L و V_R فإن الروبوت لا يسير بمسار مستقيم بل يتبع مساراً منحنياً حول نقطة تبعد مسافة R عن مركز الروبوت مما يغير من موقع واتجاه الروبوت في آن واحد. وبإحصاء عدد دورات عجلات

الروبوت مع معرفة حجم العجلات وكينماتيكية الروبوت يمكن معرفة مقدار تغير مكان الروبوت كما أن نسب هذه القياسات إلى إحداثيات مطلقة يجعل بالإمكان إجراء القياسات المختلفة كرصد تغيرات الموضع والسرعة والاتجاه.

وبما أن كل من R وω و V_R و V_L هي توابع للزمن. فإذا كانت إحداثيات الروبوت (x, y, θ) من أجل اللحظة t بالإضافة لسرعات كلتا العجلتين اليمنى واليسرى V_L و V_Rعلى التوالي، يمكن بسهولة وفق الشكل(5) تحديد موقع مركز الانحناء اللحظي كما يلي:

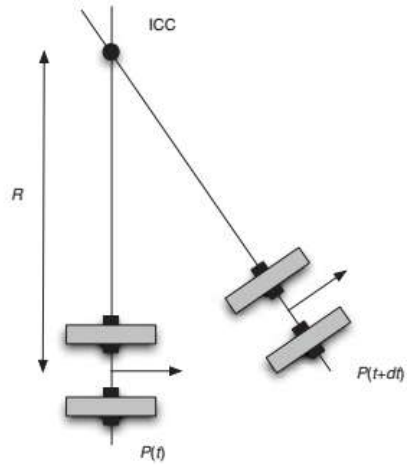
$$ICC = \big(x - Rsin(\theta), y + Rcos(\theta)\big) \qquad (5$$

كما يمكن تحديد أين يمكن أن يتواجد الروبوت في أي لحظة t اعتماداً على بارامترات التحكم $V_L(t)$ و $V_R(t)$ وعلى فرض أن الروبوت يسير وفق اتجاه معين $\theta(t)$ وبسرعة محددة $V(t)$:

$$x(t) = \int_0^t V(t)\cos\big(\theta(t)\big)dt \qquad (6$$

$$y(t) = \int_0^t V(t)\sin\big(\theta(t)\big)dt \qquad (7$$

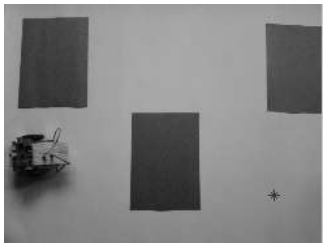
$$\theta(t) = \int_0^t \omega(t)dt \qquad (8$$



الشكل (5) حساب موقع مركز الانحناء اللحظي [22].

النتائج والمناقشة:

يقوم هذا البحث بتطبيق خوارزميات Bug المذكورة سابقاً على روبوت Boe–Botوفق بيئة العمل المبينة في الشكل (6) والمثبت عليها مكان وجود الروبوت بالإضافة لنقطة الهدف



خريطة العمل

روبوت Boe–Bot

الشكل (6) الروبوت وبيئة العمل

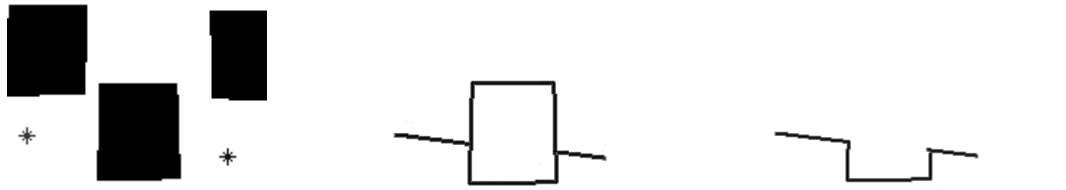
خوارزميات Bug:

اعتمد هذا البحث في تطبيقه لخوارزميات Bug الثلاث (bug1,bug2,Tangentbug) على الأكواد الوهمية pseudocodes المستعرضة في[23]وفق ما يلي:

خوارزمية Bug1:

Input: A point robot with a tactile sensor
Output: A path to the q_{goal} or a conclusion no such path exists
1: **while** Forever **do**
2: **repeat**
3: From q_{i-1}^{L}, move toward q_{goal}
4: **until** q_{goal} is reached **or** an obstacle is encountered at q_i^H
5: **if** Goal is reached **then**
6: **Exit**.
7: **endif**
8: **repeat**
9: Follow the obstacle boundary.
10: **until** q_{goal} is reached **or** q_i^H is re-encountered.
11: Determine the point q_i^H on the perimeter that has the shortest distance to the goal.
12: Go to q_i^L.
13: **if** the robot were to move toward the goal **then**
14: Conclude q_{goal} is not reachable and exit.
15: **end if**
16: **end while**

وبما أن الخوارزميات هنا هي خوارزميات عامة وليست محلية لذلك يمكن الاكتفاء بالحل النهائي الذي يتم الحصول عليه دون الحاجة لأن يتتبع الروبوت كامل حواف العائق قبل إيجاد المسار والتوجه للهدف وتكون نتيجة تطبيق الخوارزمية كما في الشكل (7)



بيئة عمل الروبوت — نتيجة تطبيق خوارزمية Bug1 — المسار الناتج عن تطبيق Bug1†.

الشكل (7) تطبيق خوارزمية Bug1.

خوارزمية Bug2:

Input:A point robot with a tactile sensor
Output: A path to q_{goal} or a conclusion no such path exists
1: **While** True **do**
2: **repeat**
3: From q_{i-1}^{H}, move toward q_{goal} along m-line.
4: **until** q_{goal} is reached **or** an obstacle is encountered at hit point q_i^H
5: Turn left (or right).

†تم رسم المسار الناتج عن تطبيق الخوارزمية وحده بدون رسم خريطة العمل لأن المسار في هذه الخوارزمية يكون ملاصقاً تماماً للعائق فلا يظهر بشكل جيد إذا رسم العائق معه

6: **repeat**
7: Follow boundary
8: **until**
9: q_{goal} is reached **or**
10: q_i^H is re-encountered **or**
11: m-line is re-encountered at a point m such that
12: $m \neq q_i^H$ (robot did not reach the hit point),
13: $d(m, q_{goal}) < d(m, q_i^H)$ (robot is closer), and
14: If robot moves toward goal, it would not hit the obstacle
15: Let $q_{i+1}^L = m$
16: Increment i
17: **end while**

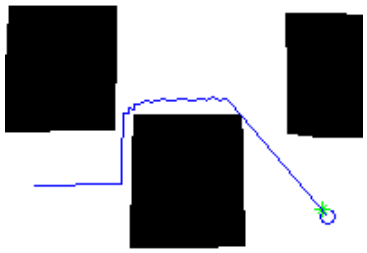
وتظهر نتائج تطبيق خوارزمية Bug2 في الشكل (8):

الشكل (8) المسار الناتج عن تطبيق خوارزمية Bug2.

1.1.1 خوارزمية TangentBug:

Input:A point robot with a range sensor
Output:A path to the q_{goal} or a conclusion no such path exists
1: **while**True **do**
2: **repeat**
3: Continuously move toward the point $n \in \{T, O_i\}$ which minimizes $d(x, n) + d(n, q_{goal})$
4: **until**
-the goal is encountered **or**
-The direction that minimizes $d(x, n) + d(n, q_{goal})$ begins to increase $d(x, q_{goal})$ i.e., the robot detects a "local minimum" of
5: Chose a boundary following direction which continues in the same direction as the most recent motion-to-goal direction.
6: **repeat**
7: Continuously update d_{reach}, $d_{followed}$, and $\{O_i\}$
8: Continuously moves toward $n \in \{O_i\}$ that is in the chosen boundary direction.
9: **until**
- The goal is reached.
- The robot completes a cycle around the obstacle in which case the goal cannot be achieved.
- $d_{reach} < d_{followed}$
10: **end while**

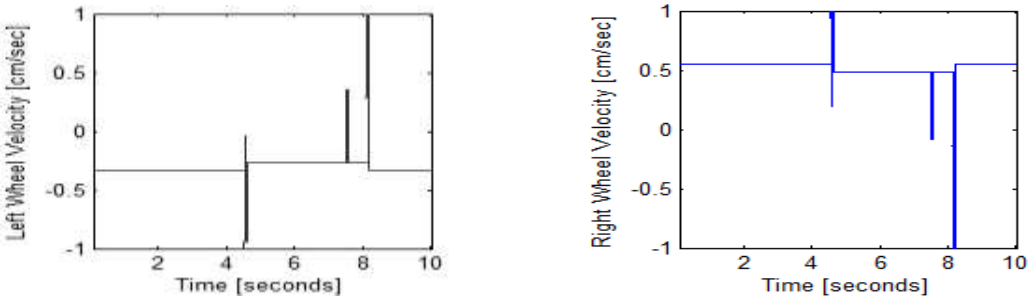
ويبين الشكل(9) نتيجة تطبيق خوارزمية TangentBug



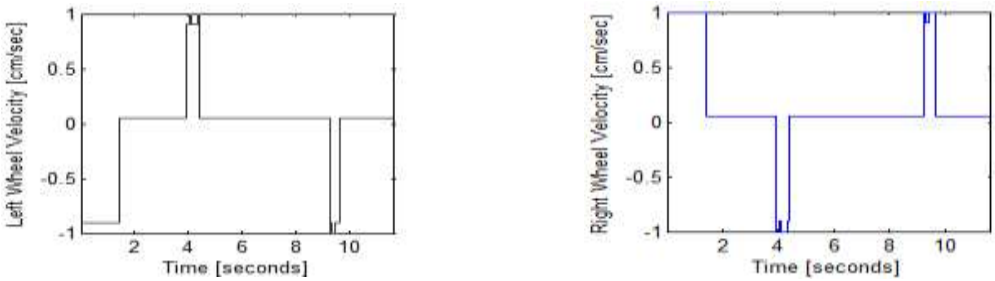
الشكل (*9*) تطبيق خوارزمية TangentBug.

قيادة الروبوت التفاضلي:

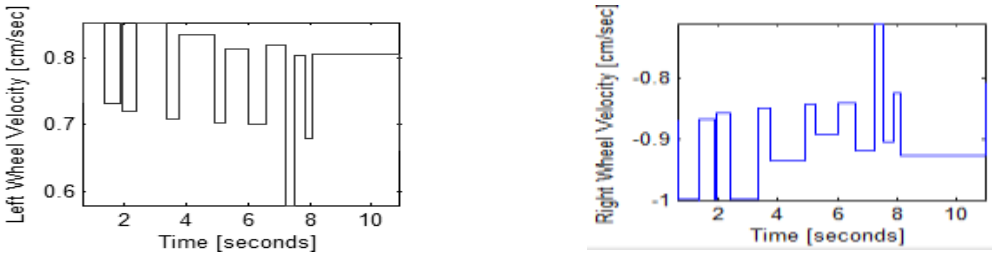
لإنجاز الحركة المطلوبة من الروبوت لابد من تحويل المسار الناتج عن الخوارزميات السابقة وفق المعادلات المذكورة سابقاً إلى قيم تمثل سرعتي العجلتين القائدتين للروبوت.



خوارزمية Bug1



خوارزمية Bug2

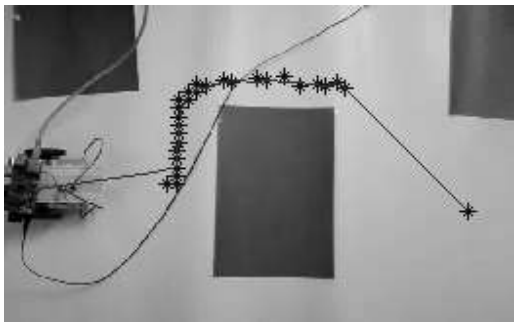


خوارزمية TangentBug

الشكل (*10*) مخططات سرعتي العجلتين في الخوارزميات الثلاث.

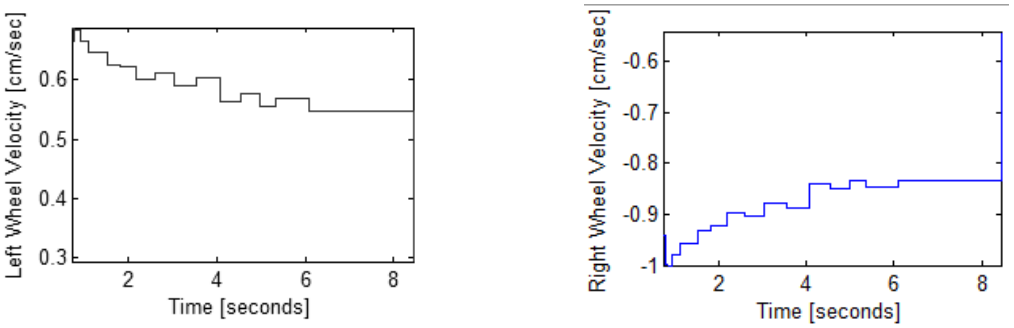
ويظهر الشكل (10) مخططي سرعة العجلتين لكل من الخوارزميات الثلاث السابقة بالنسبة للزمن(الذي تم الحصول عليه من إيجاد قيمة الانتقال الخطي أو الزاوي للروبوت وتقسيمه على السرعة الأعظمية للروبوت والتي تساوي $V_{robot} = 15\ cm/sec$ و$\omega_{robot} = 1.42\ rad/sec$).

يمكن بسهولة ملاحظة مقدار التغير الكبير المفاجئ في قيم سرعة الروبوت وخصوصاً في خوارزمية TangentBug و يمكن تنعيم المسار للتخلص من التغيرات الصغيرة الحاصلة فيه. وقد اعتمد هذا البحث على مرشح القيمة المتوسطة لإنجاز عملية التنعيم والتي تبدو في الشكل (11).



الشكل (*11*) مسار الروبوت بعد التنعيم.

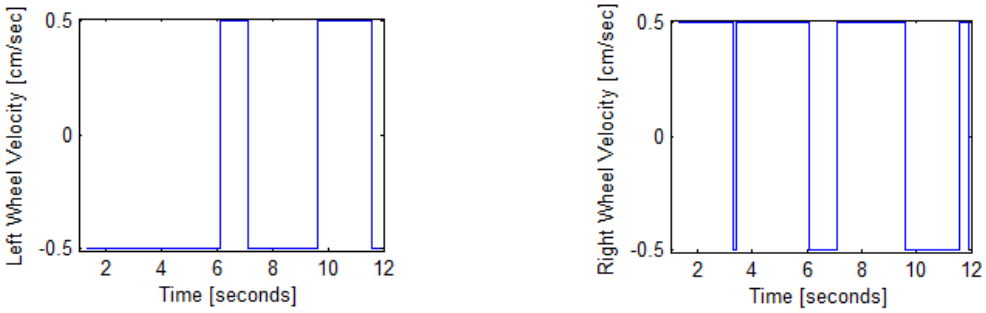
تصبح تغيرات سرعة العجلتين أكثر سلاسة بعد تنعيم المسار كما يبدو في الشكل(12).



الشكل (*12*) مخططا سرعتي العجلتين بعد تنعيم المسار.

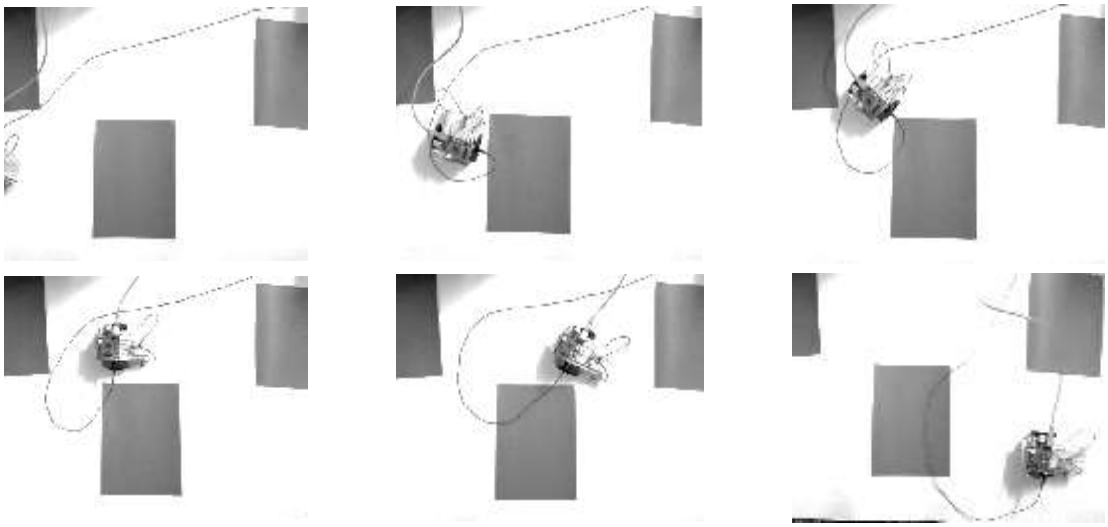
حالة خاصة:

يمكن الاعتماد على الحالتين الخاصتين لعمل الروبوت (سير الروبوت بخط مستقيم أو دورانه حول نفسه)في إنجاز الحركة المطلوبة وهنا تكون سرعتا العجلتين متساويتين دائماً والاختلاف الوحيد هو في اتجاه الدوران.يتم في هذه الحالة تحويل مسار الروبوت إلى الشكل القطبي والاستعانة بالطويلة والزاوية في تحريك الروبوت، بحيث يدور الروبوت أولاً بقيمة الزاوية الناتجة ثم يتجه بخط مستقيم إلى النقطة التالية. ويبين الشكل (13) مخطط سرعتي العجلتين اللازمتين لتحريك الروبوت وفق المسار الناتج عن خوارزمية TangentBug في هذه الحالة.



الشكل (*13*) مخططا سرعتي العجلتين باستخدام الحالة الخاصة.

وعند تطبيق مخططات سرعتي العجلتين على عجلات الروبوت تحرك وفق الشكل(14).



الشكل (*14*)تسلسل حركة الروبوت وصولاً إلى الهدف.

يبين الجدول التالي طول وزمن تنفيذ المسارات الناتجة عن الخوارزميات الثلاث.

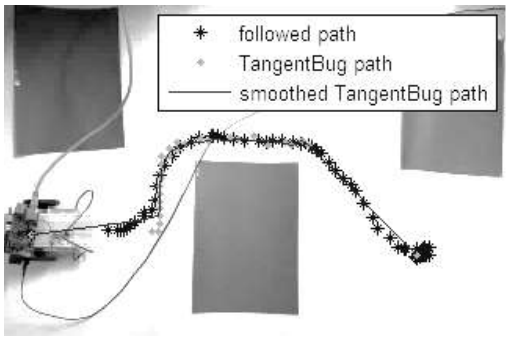
جدول(1) نتائج تنفيذ مسارات الروبوت الناتجة عن الخوارزميات الثلاثة في بيئة العمل.

نوع الخوارزمية	طول المسار [cm]	طول المسار بعد التنعيم[cm]	زمن تنفيذ المسار[sec]	زمن تنفيذ المسار بعد التنعيم[sec]	زمن تنفيذ المسار وفق الحالة الخاصة[sec]
Bug1	207.2742	190.1250	10.25	9.82	10.25
Bug2	277.7696	240.5321	17.37	13.94	17.37
TangentBug	238.2520	193.2435	17.51	11.53	11.9

وبالاعتماد على الجدول السابق يمكن استنتاج ما يلي:

-المسار الناتج عن تطبيق خوارزمية Bug1 هو أقصر المسارات التي تم الحصول عليها، برغم كون السيئة الأساسية لتلك الخوارزمية عند استخدامها كخوارزمية تجنب عوائق هو طول المسار الناتج. ويعود السبب وراء هذا الاختلاف في النتائج أن تلك الخوارزمية تفترض التجول الكامل حول العائق قبل اختيار النقطة المناسبة للانتقال نحو تتبع الهدف. أما عند استخدامها كخوارزمية تخطيط عام للمسار فعندها تؤخذ تلك النقطة الناتجة دون أن يضطر الروبوت لسلوك المسار الطويل، مما يجعل نتائج هذه الخوارزمية تبدو الأفضل بين نظيراتها.

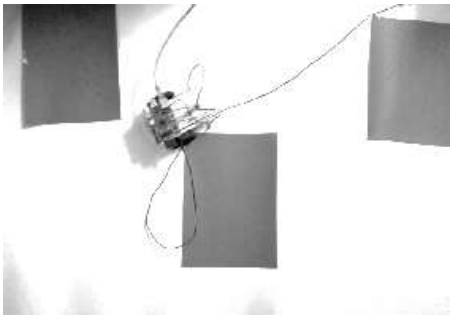
-طول المسار لا يعني زمن تنفيذ أقل، ويظهر هذا جلياً في الجدول السابقفالمسار الناتج عن خوارزمية TangentBugبالرغم من أنه ليس أطول المسارات إلا أنه أخذ الوقت الأكبر في التنفيذ، ويرجع ذلك إلىاحتوائه على تعرجات صغيرة متعددة يحتاج تنفيذها إلى مجموعة كبيرة من التغييرات الحادة في سرعتي المحركين كما يظهر في الشكل (10)، وهذا ما يجعل زمن تنفيذه أطول من البقية.أما عند تطبيق عملية التنعيم للتخلص من تلك التعرجات الصغيرة فقد قل زمن التنفيذ بشكل كبير. ويظهر الشكل(15) الفرق في التنفيذ بين المسار العام والمسار بعد تنعيمه



الشكل (*15*) الفرق بين المسارات والناتجة والمسار الذي اتبعه الروبوت.

إذا كان المسار ذو حواف حادة كما في خوارزميتي Bug1 و Bug2 فلن يكون هناك أي فرق في تنفيذ المسار بين الحالتين العامة والخاصة، لأنه في كلتا الحالتين سيضطر الروبوت إلى السير بخط مستقيم ومن ثم الدوران بما يحقق الزاوية المطلوبة وإكمال مساره وفق خط مستقيم وهكذا. ويظهر هذا جلياً من تساوي زمن تنفيذ المسار وفق الحالتين العامة والخاصة كما يظهر في الجدول السابق.

قد يؤديالمسار الناتج عن عملية التنعيم للاصطدام بعائق وذلك نظراً لأن أساس خوارزميات Bug تعتمد على السير بمحاذاة العائق عند تحسسه. ويؤدي هذا الاقتراب الشديد إلى إمكانية عالية لحدوث أي اصطدام فيما لو تم تنعيم هذا المسار بشكل مباشر دون الأخذ بعين الاعتبار مقدار اقتراب الروبوت أساساً من هذا العائق كما يبدو في الشكل(16).



الشكل (*16*) اصطدام الروبوت بالعائق أثناء تنفيذه للمسار الناتج عن التنعيم.

الاستنتاجات والتوصيات:

استطاعت خوارزميات Bug(عند استخدامها كخوارزميات تخطيط مسار عام)إيجاد مسار قريب من المثالي يمكن للروبوت أن يسلكه بسهولة وخصوصاً في خوارزميتي Bug1وBug2 . فقد استطاعت خوارزمية Bug1 أن تستفيد من الدوران حول العائق في اختيار نقطة الانتقال الأنسب وهذا ما أدى لمسار قصير بين نقطتي البداية والهدف. كما أظهر البحث أهمية تنعيم المسار في زيادة كفاءة تنفيذ الروبوت لحركته وسرعة سلوكه للمسار الناتج "خاصةً إذا كان هذا المسار يمتلك مجموعة من التعرجات الصغيرة كما في مسار TangentBug"، مع الأخذ بعين الاعتبار زيادة مسافة الأمان اللازمة حول العوائق لضمان ألا يصطدم الروبوت بالعائق أثناء سلوكه للمسار المنعم Smoothed path.

رغم بساطة آلية عمل خوارزميات Bug إلا أنه من الممكن الاستفادة من التحسن الكبير الذي طرأ على خوارزمية Bug1 عندما استطاعت إيجاد النقطة المناسبة للانتقال بين تتبع المسار والاتجاه نحو الهدف والعمل على

زيادة كفاءة الخوارزميات المتبقية في إيجاد المسار الأمثل عن طريق تحسين آلية الانتقال بين مرحلتي تتبع الحدود والاتجاه نحو الهدف واختيار الاتجاه الذي سيسلكه الروبوت في بداية تتبعه للحدود بالإضافة لتقسيم المسار إلى عدد من الأهداف الجزئية التي تضمن اجتياز العوائق بشكل أفضل والتخلص من التعرجات الصغيرة التي قد تحصل في المسار.

المراجع

[1] S. M. LAVALLE, *Planning Algorithms*, New York, USA. : Cambridge University Press, 2006.

[2] P. K. DAS, H. S. BEHERA, P. K. JENA AND B. K. PANIGRAHI, "*Multi-robot path planning in a dynamic environment using improved gravitational search algorithm*," Journal of Electrical Systems and Information Technology, 2016,vol. 3, pp. 295-313.

[3] L. FENG-LI, "*Tracking and Following Algorithms of Mobile Robots for Service Activities in Dynamic Environments*," International Journal of Automation and Smart Technology, 2015,vol. 4, no. 4, pp. 39-48.

[4] W. LOU AND X. CHUNRUI, "*Mobile Robot Path Planning based on Probabilistic Model Checking under Uncertainties*," in 3rd International Conference on Machinery, Materials and Information Technology Applications (ICMMITA 2015), 2015.

[5] S. CHENG'EN AND H. JUN, "*A Hybrid Path Planning Algorthim For Indoor Mobile Robot Using Hierarchy Reinforcement Learning*," International Journal of Control and Automation, 2016,vol. 9, no. 5, pp. 351-362.

[6] C. Y. CHEN, B. Y. SHIH, C. H. SHIH AND L. H. WANG, "*Enhancing robust and stability control of a humanoid biped robot: system identification approach.*," Journal of Vibration and Control, 2013, vol. 19, no. 8, pp. 1199-1207.

[7] M. BARKAOUI, J. BERGER AND A. BOUKHTOUTA, "*An information-theoretic-based evolutionary approach for the dynamic search path planning problem.*," in Advanced Logistics and Transport (ICALT), 2014 IEEE International Conference, 2014.

[8] K. SOLOVEY, J. YU, O. ZAMIR AND D. HALPERIN, "*Motion Planning for Unlabeled Discs with Optimality Guarantees*," in *arXiv* , 2015.

[9] G. KALARANI AND R. RANIHEMAMALINI, "*A Survey of the Various Path Planning Techniques Used in the Navigation of Autonomous Mobile Robot*," INDIAN JOURNAL OF APPLIED RESEARCH, 2014, vol. 4, no. 10, pp. 442-444.

[10] N. LEENA AND K. K. SAJU, "*A survey on path planning techniques for autonomous mobile robots*," IOSR J ournal of Mechanical and Civil Engineering (IOSR - JMCE), pp. 76-79, 2014.

[11] L. YANG, J. QI, S. DALEI, X. JIZHONG, H. JIANDA AND Y. XIA, "*Survey of Robot 3D Path Planning Algorithms*," Journal of Control Science and Engineering, 2016.

[12] N. B. N. SARIFF, "*Ant Colony System For Robot Path Planning In Global Static Environment*.," in 9th WSEAS International Conference on System Science and Simulation in Engineering *(ICOSSSE'10)*, Iwate, Japan, 2010.

[13] V. G. M. SEZER, "A Novel Obstacle Avoidance Algorithm: Follow the Gap Method.," *Robotics and Autonomous Systems*, 2012,vol. 60, no. 9, pp. 1123-1134.

[14] C. H. CHIANG, J.-S. LIU AND Y.-S. CHOU, "*Comparing path length by boundary following fast matching method and bug algorithms for path planning*," in Opportunities and Challenges for Next-Generation Applied Intelligence, Springer, 2009, p. 303–309.

[15] N. CHOUDHURY, R. MANDAL AND S. K. KAR, "*Bioinspired Robot Path Planning using PointBug Algorithm*," in International Conference on El ectrical, Electronics, and Optimization Techni ques *(ICEEOT)*, 2016.

[16] V. SEZER AND M. GOKASAN, "*A novel obstacle avoidance algorithm: "Follow the Gap Method"*," Robotics and Autonomous Systems, vol. 60, no. 9, pp. 1123-1134, 2012.

[17] W. NYANKO, "*Bug1, Bug2, Tangent Bug algorithm*," 12 7 2011. [Online]. Available: Http://blog.daum.net/pg365/115 . [Accessed 5 12 2016].

[18] H. T. Nguyen and H. X. Le, "Path planning and Obstacle avoidance approaches for Mobile robot," International Journal of Computer Science Issues, 2016,vol. 13, no. 4, pp. 1-10.

[19] N. JAMES, *An analysis of mobile robot navigation algorithms in unknown environments*, Perth: Univ. of Western Australia, 2010.

[20] G. CARBONE AND F. GOMEZ-BARVO, "*Motion and Operation Planning of Robotic Systems: Background and Practical Approaches*," in Springer International Publishing, Switzerland, 2015.

[21] G. DUDEK and M. JENKIN, "*Mobile robot hardware:Locomotion*," in *Computational Principles of Mobile Robotics 2nd edition*, CAMBRIDGE UNIVERSITY PRESS, 2010, pp. 31-69.

[22] R. MATHEW and S. S. HIREMATH, "Trajectory Tracking and Control of Differential Drive Robot for Predefined Regular Geometrical Path," Procedia Technology, 2016,vol. 25, pp. 1273-1280.

[23] H. CHOSET, K. LYNCH, S. HUTCHINSON, G. KANTOR, W. BURGARD, L. KAVRAKI and S. THRUN, "Chapter2: Bug Algorithms," in *Principles of robot motion: theory, algorithms, and implementation*, The MIT Press, 2005, pp. 25-43.