A Simple Local Path Planning Algorithm for Autonomous Mobile Robots

Buniyamin N., Wan Ngah W.A.J., Sariff N., Mohamad Z.

***Abstract*—**This paper presents an overview of path planning algorithms for autonomous robots. The paper then focuses on the bug algorithm family which is a local path planning algorithm. Bug algorithms use sensors to detect the nearest obstacle as a mobile robot moves towards a target with limited information about the environment. The algorithm uses obstacle border as guidance toward the target as the robot circumnavigates the obstacle till it finds certain condition to fulfill the algorithm criteria to leave the obstacle toward target point.

In addition, this paper introduces an approach utilizing a new algorithm called PointBug. This algorithm attempts to minimize the use of outer perimeter of an obstacle (obstacle border) by looking for a few important points on the outer perimeter of obstacle area as a turning point to target and finally generate a complete path from source to target. The less use of outer perimeter of obstacle area produces shorter total path length taken by a mobile robot. Further this approach is then compared with other existing selected local path planning algorithm for total distance and a guarantee to reach the target.

***Keywords*—**Path Planning, Bug algorithm, Autonomous robot, Sensor based, Mobile robot.

I. Introduction

Path planning is one of the most important elements for mobile robot. Path planning is the determination of a path that a robot must take in order to pass over each point in an environment [1-4] and path is a plan of geometric locus of the points in a given space where the robot has to pass through[5]. Generally, the problem of path planning is about finding paths by connecting different locations in an environment such as graph, maze and road. Path planning "enables" mobile robots to see the obstacle and generate an optimum path so as to avoid them.

The general problem of path planning for mobile robots is defined as the search for a path which a robot (with specified geometry) has to follow in a described environment, in order to reach a particular position and orientation B, given an initial position and orientation. As mobile robot is not a point in space, it has to determine the correct direction or perform a proper movement to reach destination and this is called maneuvering planning.

II. Path Planning Algorithms

A. Path Planning Approaches

Various approaches have been introduced to implement path planning for a mobile robot [6]. The approaches are according to environment, type of sensor, robot capabilities and etc, and these approaches are gradually toward better performance in term of time, distance, cost and complexity. Al-Taharwa [7] for example, categorized path planning as an optimization problem according to definition that, in a given mobile robot and a description of an environment, plan is needed between start and end point to create a path that should be free of collision and satisfies certain optimization criteria such as shortest path. This definition is correct if the purpose of solving path planning problem is for the shortest path because most new approaches are introduced toward shorter path. Looking for the shorter path does not guarantee the time taken is shorter because sometime the shorter path needs complex algorithm making the calculation to generate output is longer.

B. Properties of Path Planning

Mobile robot path planning has a few main properties according to type of environment, algorithm and completeness. The properties are whether it is static or dynamic, local or global and complete or heuristic. The static path planning refers to environment which contains no moving objects or obstacles other than a navigating robot and dynamic path planning refers to environment which contains dynamic moving and changing object such as moving obstacle. Meanwhile the local and global path planning depend on algorithm where the information about the environment is a priori or not to the algorithm. If the path planning is a global, information about the environment already known based of map, cells, grid or etc and if the path planning is a local, the robot has no information about the environment and robot has to sense the environment before decides to move for obstacle avoidance and generate trajectory planning toward target.

Mobile robot navigation problem can be divided into three subtask namely mapping and modeling the environment, path planning and path traversal with collision avoidance [8]. Mobile robot navigation problems cannot be decomposed into

fixed subtasks because the navigation problem varies according to approach used to solve the problem. As an example, the bug algorithm solves the navigation problem without need to map and model the environment and only response to the output from contact sensor.

C .Evolution of Robot Environment Modeling

Various types of known static environment models have been proposed by previous robot path planning (PP) researchers. A few have created models of environments by tracing the obstacles (i.e Visibility graph), utilized the free space area (i.e MAKLINK graph), and utilized free space and obstacles area (i.e cell decomposition) to produce the connectivity of graph for PP input.

In early 1980s, Rodney Brooks introduced Generalized Cones [9] that represent the global free space area with cones before generating the optimal path to goal. Although this approach was proven to work in a simple environment within this time, it had limitations when it faced with a complex environment. It needed more time to find a path because of the complexity of the process and within a year, another alternative method known as roadmap approach was introduced.

The Roadmap approach also known as Visibility graph and Voronoi diagram are models in the categories of graph search technique [10]. The connectivity of free space graph will be generated before the PP algorithms work to find the path. Although this approach was successfully implemented in a few robot PP systems in simple environments, its limitation was that it requires more time to create the Voronoi diagram because the robot needs to create the spatial points in the initial process. For the Visibility graph, the vertices are close to the obstacles and the possibility to collide with obstacles is high [10]. In addition, it is also a complex approach for robot applications in a complex, extremely cluttered and changing environment.

Cell decomposition approach, a graph technique which is more efficient was introduced in early 1990s. This approach was widely used in robot PP systems in both static and dynamic environments as the implementation is easier, accurate and easy to be updated. It is also one of the most popular representations of environment. Its limitation is that it will work much slower than other approaches especially with older computers that have slow processors as it needs more storage to store the cells. Although it was a problem when it was first introduced, the current computing power with new generations of computers that can solve this problem has generated new interest in this approach. For example, Galvaski, who is now investigating its performance [11]. Now, other modeling approaches based on grid have been proposed, such as Quadtree and Framed Quadtree [12-15], to increase the accuracy of the path found. In addition, another graph for free space modeling known as MAKLINK graph [16] was also introduced in the year 2000.

Environment modeling has been improving from year to year. Figure 1 shows the evolution of environment modeling approaches from selected sources from early 1980s until 2009. Compared to the older, traditional approaches such as generalized cones and roadmap approaches, current modeling approach such as grid is much safer, precise, accurate and more adaptable to be used in a static or dynamic environment.

It can be concluded that choosing an appropriate environment model is very important in robot PP research as it will influence the PP algorithm search process to find the path to goal position.

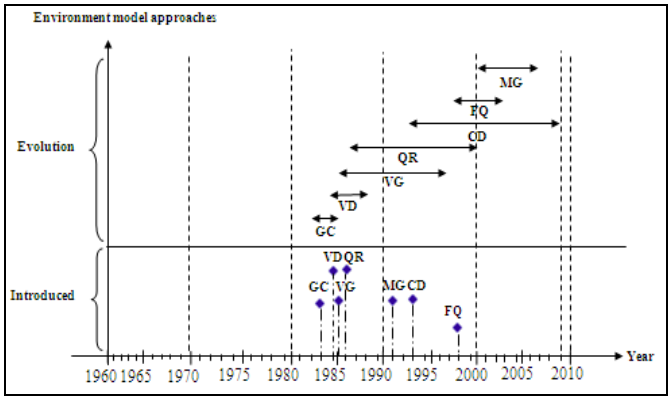

Fig. 1 Evolution of environment modeling approaches from selected sources from early 1980s until 2009

Indicator:
GC=Generalized Cones (5 papers)
VD=Voronoi Diagram (3 papers)
VG=Visibility Graph (4 papers)
QR=Quadtree Representation (4 papers)
CD=Cell Decomposition (Regular grid) (25 papers)
FQ= Framed Quadtree (3 papers)
MG=MAKLINK Graph (2 papers)

D .Global Path Planning

Global path planning is a path planning that requires robot to move with priori information of environment. The information about the environment first loaded into the robot path planning program before determining the path to take from starting point to a target point. In this approach the algorithm generates a complete path from the start point to the destination point before the robot starts its motion [17]. Global path planning is the process of deliberatively deciding the best way to move the robot from a start location to a goal location. Thus for global path planning, the decision of moving robot from a starting point to a goal is already made and then robot is released into the specified environment.

One of the early global path planning models that extensively studied is Piano's Mover problem where full information is assumed to be available on the geometry, positions of the obstacles and the moving object [18]. In this model, the full complexity of the path generation problem has been investigated, and a number of heuristic and non-heuristic

approaches involving moving rigid or hinged bodies in two or three dimensional space have been considered [19]]. A few common approaches are used in global path planning are Roadmap such as Visibility Graph, Voronoi Graph and Silhouette, Cell Decomposition such as Exact Decomposition, Approximate decomposition and Hierarchical Decomposition and also new modern approaches such as Genetic Algorithm [20-22], Neural Network [23] and Ant Colony Optimisation (ACO)[24, 25].

E. *Local Path Planning*

Local path planning is path planning that requires robot to move in unknown environment or dynamic environment where the algorithm is used for the path planning will response to the obstacle and the change of environment. Local path planning also can be defined as real time obstacle avoidance by using sensory based information regarding contingency measures that affect the save navigation of the robot [26].

In local path planning, normally, a robot is guided with one straight line from starting point to the target point which is the shortest path and robot follows the line till it sense obstacle. Then the robot performs obstacle avoidance by deviating from the line and in the same time update some important information such as new distance from current position to the target point, obstacle leaving point and etc. In this type of path planning, the robot must always know the position of target point from its current position to ensure that robot can reach the destination accurately.

Potential field method [27] is the one of the well known local path planning technique. In this path planning method, the robot is considered as a particle moving under influence of an artificial potential produced by the goal configuration and the obstacles. The value of a potential function can be viewed as energy and the gradient of the potential is force. The goal configuration is an attractive potential and the obstacles are all repulsive potential.

This paper introduces and describes a new local path planning algorithm based on the Bug Family of path planning algorithms. PointBug Algorithm tries to reduce the usage of outer parameter of obstacle as implemented in Bug Algorithm. As an example, in Bug1, the coverage of circumnavigating of obstacle is more than the size of perimeter of the obstacle and meanwhile for Bug2, the maximum coverage is equal to the total perimeter size of obstacle. By avoiding circumnavigating the obstacle, the problem PointBug is to find next point to go toward target point. It has to determine where the next point should be located on the outer parameter of obstacle.

In PointBug Algorithm robot is assumed equipped with an infinite range sensor, odometer and digital compass with ideal positioning. The range sensor gives a reading to controller for interval period and action is executed based on the difference in reading of two sequences of times. Then robot moves to the new sudden point according to the angle of robot rotation and limited by the odometer.

III. BUG ALGORITHMS

Bug algorithms are well known mobile robot navigation method for local path planning with minimum sensor and simple algorithm [3, 28]. James Ng and Thomas Bräunl listed about eleven types of bug algorithms [29]. The most commonly used and referred in mobile robot path planning are Bug1 and Bug2 [19], DistBug [30], VisBug [31] and TangentBug [32]. Others bug algorithms are Alg1 and Alg2 [33], Class1 [34], Rev and Rev2 [35]]; OneBug and LeaveBug [29].

The variations of bug algorithms showed the effort toward shorter path planning, shorter timing, simpler algorithm and better performance. Bug1 moves from start point toward target point by hitting and circumnavigating the obstacle then leaving the leave point. Bug2 has similar behaviour except it is guided by m-line where m-line is used as leaving point and hitting point. Bug1 is considered overcautious and having coverage more than the full perimeter of the obstacle yet effective meanwhile Bug2 is inefficient in some cases such as local loops but shorter coverage compared to Bug1.

The first bug family algorithm that incorporates a range sensor is Visbug [31] which calculates shortcuts relative to the path generated by the Bug2 algorithm from or to m-line. Alg1 [36, 37] improved Bug2 weakness is that it can trace the same path twice by storing the sequence of hit points occurring within an actual path to the goal. These storing data are used to generate shorter paths by choosing opposite direction to follow an obstacle boundary when a hit point is encountered for the second time. The same researcher introduced the Alg2 to improve Alg1 by ignoring the m-line of Bug2 with new leaving condition. The Alg1 and Alg2 still face a reverse procedure problem where after encountering a visited point that causes loop, a mobile robot follows an uncertain obstacle by an opposite direction until it can leave the obstacle. Horiuchi solved the reverse procedure problem in Alg1 and Alg2 by introducing a mixing reverse procedure with alternating following method to create shorter average bound of path length and named the algorithm as Rev1 and Rev2 [38]. Alternating following method is defined as independently, if a robot always changes a direction following an uncertain obstacle alternatively, the robot arrives at a destination earlier on average and there will decrease probability for the robot to join a loop around a destination.

Other bug algorithms that also incorporate range sensors are DistBug algorithm and TangentBug Algorithm. The DistBug algorithm is a local path planning algorithm that guarantees convergence and will find a path if one exists. It requires its own position by using odometry, goal position and range sensor data. To guarantee convergence to the target, the DistBug algorithm needs a small amount of global information for updating dmin(T) and for determining that the robot completed a loop around an obstacle. The value of dmin(T) can be extracted directly from the visual information. This guarantee convergence using updating dmin(T) value makes problem in determining accuracy because the value of

dmin(T) is taken from direct global visual information.

Meanwhile, the TangentBug is another variations of DistBug that improves the Bug2 algorithm in that it determines a shorter path to the goal using a range sensor with a 360 degree infinite orientation resolution [39]. Tangent Bug incorporates range sensors from zero to infinity to detect obstacles. When an obstacle is detected, the robot will start moving around the obstacle and will continue its motion-toward target point routine as soon as it has cleared the obstacle. During following boundary, it records the minimal distance to target dmin(T) which determines obstacle leaving and reaching condition. While the robot is moving towards target, d(x,T) decreases monotonically and boundary following attempts to escape from a local minimum of d(x,T). The robot constructs a local tangent graph (LTG) based on its sensors' immediate readings. The LTG is constantly updated and it is used by the robot to decide the next motion. The disadvantage of this algorithm is requiring robot to scan 3600 before making decision to move to the next target. The latest variant of TangetBug is LadyBug [40] which incorporates bio-inspired heuristics to improve the robot trajectory in real time based on Ladybugs hunt for aphids for a group of networked mobile robots. Figure 2 shows the different of paths taken among four bug algorithms.

An extension to classical Bug based algorithms called Sensbug was introduced which can produce an effective path in an unknown environment with both stationary and movable obstacles [41]]. CBug applies the Bug1 behaviour in its algorithm with online navigation algorithm for a size D disc robot moving in general planar environments [42, 43]. The algorithm searches a series of expanding ellipses with focal starting point and target point, and its total path length is at most quadratic in length of the shortest offline path [44]. K-Bug algorithm [45] consumes global information such as obstacles geometry and position to select the waypoint among the vertex of obstacles that caused collisions. This algorithm does not use sensor to get the environment information and the information needs to change every time the environment is changed.

IV. POINTBUG ALGORITHM

PointBug, recently developed navigates a point of robot in planar of unknown environment which is filled with stationary obstacles of any shape. It determines where the next point to move toward target from a starting point. The next point is determined by output of range sensor which detects the sudden change in distance from sensor to the nearest obstacle. The sudden change of range sensor output is considered inconstant reading of distance either it is increasing or decreasing. It can be from infinity to certain value or certain value to infinity or certain value to a certain value where the difference value, Δd is defined. If value of Δd is defined for 1cm, any reading from range sensor from interval time, tn to tn+1 which detects the different in range for 1cm and above is considered a sudden point.

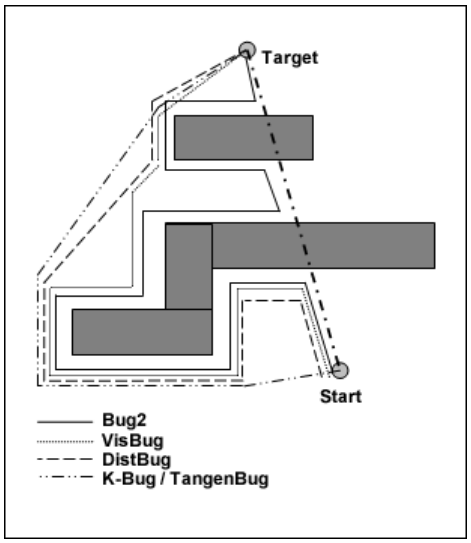

Fig. 2 Trajectories generated by a few bug algorithms

The robot is capable to scan the environment using range sensor by rotating itself from 00 up to 3600at a constant speed. The initial position of robot is facing straight to the target point and then the robot rotates left or right searching for sudden point. After the first sudden point is found, the rotation direction of the robot is according to position of straight line between current sudden point and target point or dmin line. The rotation direction of robot is always toward position of dmin line. The value of dmin is the shortest distance in one straight line between sudden point and target point and its value is always recorded every time the robot reaches new sudden point. The robot always ignores the sensor reading at rotation of 1800 to avoid detection of previous sudden point making the robot return to previous sudden point from its current point. If there is no sudden point found within a single 3600 rotation, the target is considered unavailable and the robot stops immediately.

The pseudo code of the algorithm as follows:

```
While Not Target
  If robot rotation <= 360
  Robot rotates right of left according to position of dmin
   If sudden point
     If 180 degree rotation
        Ignore reading   /* to avoid robot return to previous
point */
     Else
        Get distance from current sudden point to next sudden
point
        Get angle of robot rotation
        Move to new point according to distance and rotation
angle
        Record New dmin value
        Reset rotation
     End if
  End if
```

Else
Robot Stop / No sudden point and exit loop */*
End if
While end
Robot Stop / Robot successfully reaches target */*

Figure 3 shows a range sensor scanning a pentagon shaped obstacle from A to E with a graph showing the distance produced from range sensor in cm from A to E. The C line is perpendicular to the surface of obstacle which is the shortest distance detected to the obstacle. The value of distance increases constantly from C to B and from C to D. From point B to A from the graph, the value of distance is suddenly increased almost twice and from point D to E the value of distance is suddenly increased from a few centimeters to infinity. The point A and E are the sudden points and considered the points where the robot will move for the next point. Figure 4 shows the sudden points are detected on different shape of obstacles.

Figure 5 shows how the algorithm is working in an environment to solve local minima problem by detecting sudden points from a starting point to target point. The robot first faces the target point at the starting point and then rotates from point A until it finds a sudden point at point B. Robot then move to point B and at point B; it rotates to the right direction to find next sudden point because the dmin line is located right side of current robot direction and finds new sudden point at C. Robot rotates to the right again at point C and finds new sudden point at D. At point D, the robot still rotates to the right and finds last sudden point and stop at target point.

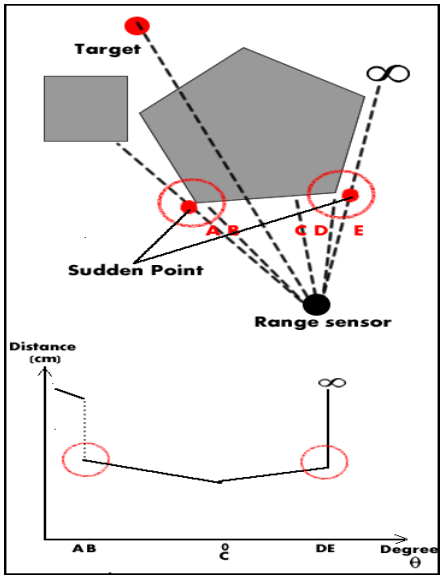

Fig. 3 Range sensor is detecting an obstacle from left to right and right to left.

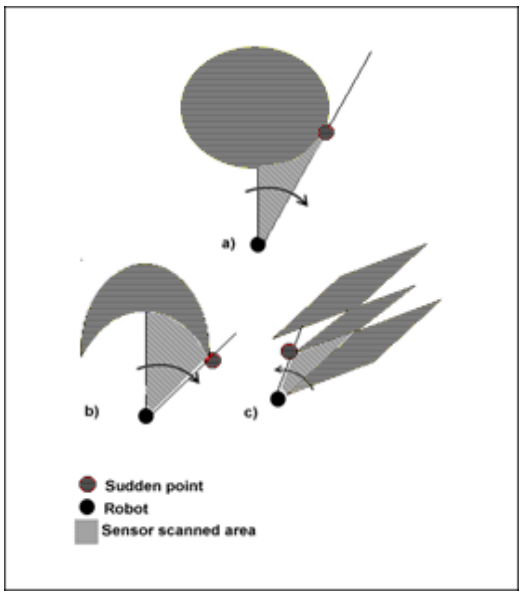

Fig. 4 Sudden points on different surfaces detected by range sensor.

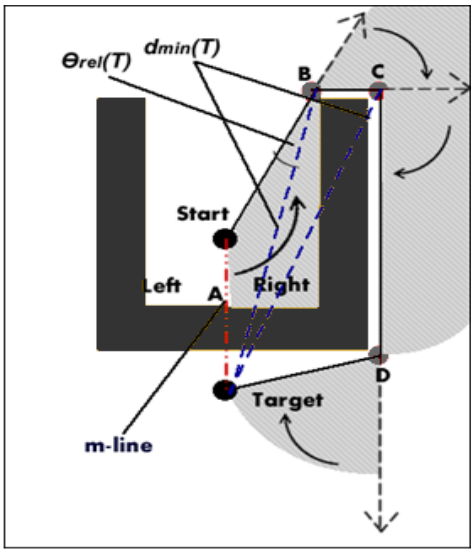

Fig. 5 Trajectory generated by the PointBug to solve local minima problem. The shadowed area is scanned area.

Table 1: Explanation on how sudden points are found from Starting point to Target from the Figure 4.

Point of Movement	Sudden point Description
A to B	B (from certain value to infinity)
B to C	C (from infinity to certain value)
C to D	D (from infinity to certain value)
D to target	Infinity to target

V. Point Bug Algorithm Analysis

The main goal of the algorithm is to generate a continuous path from start (S) to the point target (T) and S and T is fixed.

The distance between two points is denoted as d(A, B) and for this case specifically, d(S, T) = D, where D is a constant. d(An, B) signifies that point A is located at nth sudden point on the way to T, and P is total length of connected sudden points from S to T. The line (S, T) is called the main lain or m-line.

As all path generated by the algorithm are straight lines, robot position is measured by d(x, y) and the total distance can be calculate by totaling all straight lines distance those connect sudden points.

$$P = \sum_{n=1}^{s+1} (A_{n-1}, A_n) \qquad (1)$$

In PointBug algorithm, every sudden point found will produce a logical triangle which is formed from three points namely target point, current sudden point and previous sudden point. The line between target point and current sudden point is dmin line and its values are accumulated in an array starting from 0 which is distance from starting point and target point up to last sudden point before meeting target point. Value dmin[0] is assigned manually and it is the initial value required to run the algorithm. The values of dmin[1] to dmin[n] are obtained from cosines rule except dmin[0].

$$a^2 = b^2 + c^2 - 2bc \cos A \qquad (2)$$

if *a* is if a is d_{min} then;

$$d_{\min} = \sqrt{b^2 + c^2 - 2bc\cos A} \qquad (3)$$

In equation (3), the value of b is distance between current sudden point and previous sudden point which is obtained from range sensor and the value of c is previous value of dmin, then dmin[n] is;

$$d_{\min}[n] = \sqrt{b^2 + d_{\min}[n-1]^2 - 2b \times d_{\min}[n-1] \times \cos A} \qquad (4)$$

The value of ∠A is obtained from rotation of the robot from current direction to next direction if the robot located on the starting point, otherwise:

$$\angle A = 180 - \angle Adj - \angle Rot \quad \text{if } \angle A \leq 90 \qquad (5)$$
$$\angle A = \angle Adj + \angle Rot \text{ if } \angle A > 90$$

(5)

where is ∠Adj Adjacent angle of triangle and ∠Rot is the robot rotation angle. ∠Adj value is obtained from sine rule;

(6)

If sin B is ∠Adj then

$$\angle Adj = \sin^{-1} \frac{b \sin A}{a} \qquad (7)$$

where the b is previous dmin value and a is current dmin value.

Lemma 1: if dmin[n] = 0, the robot currently is on the target point.

Proof: dmin[n] is the minimum distance between sudden point and target point. If its value is zero means the sudden point is on the target point, the value of □A is zero and the value of previous dmin[n]is equal to distance between current sudden point and previous sudden point or c. Let's say value of previous dmin[n] is b, and from equation (3), the value of dmin[n] is;

$$d_{\min}[n] = \sqrt{b^2 + b^2 - 2bb\cos 0}$$
$$d_{\min}[n] = \sqrt{2b^2 - 2b^2}$$
$$d_{\min}[n] = 0$$

VI. Simulation and Results

The simulation of point to point bug algorithm is carried out using ActionSript 2.0 on Adobe Flash CS3. The algorithm is simulated on three types of environments namely free environment, maze based environment and office like environment.

Fig.6 shows that sudden points are generated at every vertex of the rectangle. In this environment, the algorithm generates the shortest path from starting point to target point.

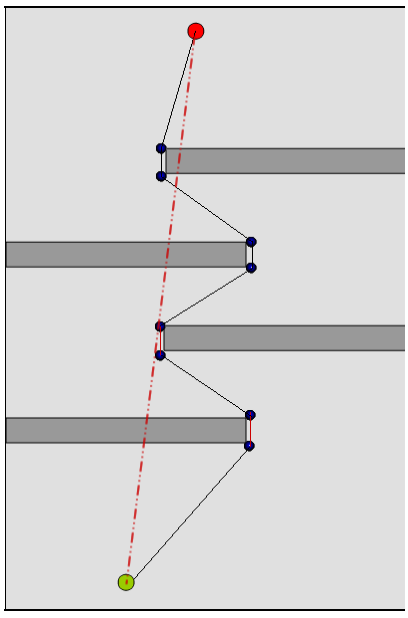

Fig. 6 Trajectory generated using PointBug algorithm in simple maze like type environment.

Figure 7 and Figure 8 show comparison of three algorithm namely Distbug, TangenBug and PointBug with each robot

equipped with unlimited sensor range. TangentBug and PointBug produced almost the same result but TangentBug makes a little obstacle following increases the total path distance taken compared to PointBug algorithm. In an office like environment, tangentBug algorithm outperforms other algorithms. The performance of pointBug varies according to types of environment and position of obstacle.

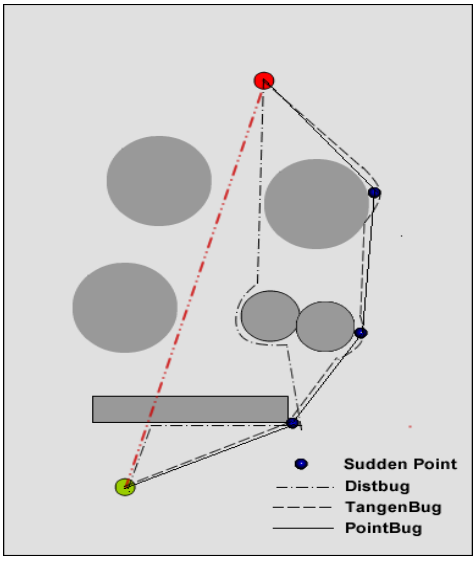

Fig. 7 Trajectory generated using Distbug, TangenBug and PointBug algorithm in Free Environment.

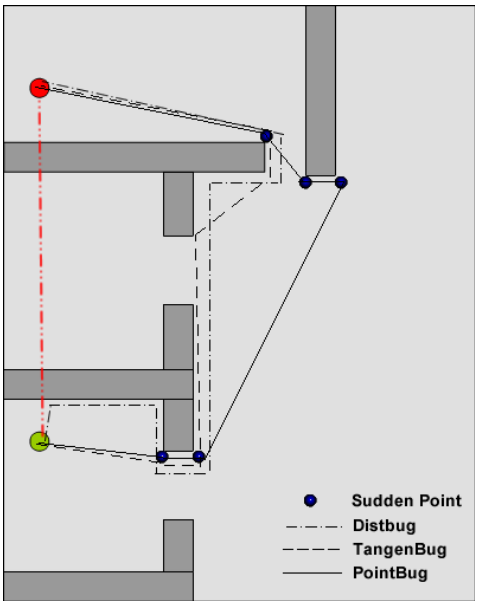

Fig. 8 Trajectory generated using Distbug, TangenBug and PointBugalgorithm in simple Office like Environment.

VII. Discussion and Conclusion

This paper compares 2 main path planning algorithms from the bug algorithm family. This new approach of path planning, The Point to Point Sensor Based Path Planning algorithm is a new approach that behaves similar to other algorithms in the bug family. However, the Point to Point Sensor Based Path Planning Algorithm needs very minimal amount of prior information namely dmin(T) and Θrel(T) compared to other bug algorithms such as DistBug and VisBug algorithm which need global information to update value of dmin(T) during the boundary following and determine completion of a loop of robot to ensure convergence to target.

The algorithm can operate in dynamic environment as well because information about the environment can be obtained immediately from the range sensor during the movement of the robot. The performance of the algorithm depends on total sudden points detected. The less number of sudden points detected is better. Thus, whether it outperforms other bug algorithms depends on obstacles in the environment as if the obstacle is a circle, it will produce less sudden point since a circle has no vertex. The total vertex in obstacle affects the total sudden points.

Acknowledgment

The authors gratefully acknowledge and thank University Technology MARA, Malaysia (UiTM) for providing a grant for this research. [600-RMI/ST/DANA 5/3/Dst (234/2009)]

References

[1] Choset, H., P. Pignon. Coverage Path Planning: The Boustrophedon Decomposition. in Int. Conf. on Field and Service Robotics. 1997.
[2] Sariff, N., Buniyamin N. An Overview of Autonomous Mobile Robot Path Planning Algorithms. in Proceedings of 4th Student Conference on Research and Development (SCORED 2006), 27th -28th June. 2006. Shah Alam, Malaysia.
[3] Wan Ngah, W.A.J., Buniyamin N, Mohamad Z. Point to Point Sensor Based Path Planning Algorithm for Mobile Robots. in 9th WSEAS International Conference on System Science and Simulation in Engineering (ICOSSSE'10). 2010. Iwate, Japan: WSEAS.
[4] Vacariu, L., Flaviu Roman, Mihai Timar,Tudor Stanciu, Radu Banabic, Octavian Cret. Mobile Robot Path-planning Implementation in Software and Hardware. in 6th WSEAS International Conference on Signal Processing, Robotics and Automation. 2007. Corfu Island, Greece.
[5] Haklıdır, M., Taşdelen Modelıng And Sımulatıon Of An Anthropomorphıc Robot Arm By Usıng Dymola. in 5th Int. Symp. on Intelligent Manufacturing Systems. . 2006.
[6] Sariff, N., Buniyamin N. 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),. 2010. 4th -6th October, Iwate, Japan: WSEAS Press.
[7] Al-Taharwa, I., A. Sheta, M. Al-Weshah, , A Mobile Robot Path Planning Using Genetic Algorithm in Static Environment. Journal of Computer Science, 2008. 4: p. 341-344.
[8] Behnke, S. Local Multiresolution Path Planning. in 7th RoboCup Int. Symposium. 2004. Padua, Italy Springer.
[9] A.Brooks, R., Solving the find path problem by good representation of free space. IEEE Transactions on Systems, Man, and Cybernetics, 1983. 13(2): p. 190-197.
[10] Eberhart, Y., Shi,. Particle swarm optimization: developments, applications and resources. in Evolutionary Computation. Proceedings of the 2001 Congress on Evolutionary Computation. 2001.
[11] Glavaski, D., Volf M., Bonkovic M., Mobile robot path planning using exact cell decomposition and potential field methods. WSEAS Transactions on Circuits and Systems 2009. 8(9).

[12] Yahja, A., Sanjiv Singh, Barry L. Brumitt. Framed-Quadtree Path Planning for Mobile Robots Operating in Sparse Environments. in IEEE Conference on Robotics and Automation (ICRA). 1998. Leuven, Belgium.
[13] Yahja, A., Singh S., Stentz A., An efficient on-line path planner for outdoor mobile robot. Journal of Robotics and Autonomous Systems 2000. 32: p. 129-143.
[14] 1Stentz, A. Optimal and Efficient Path Planning for Partially-Known Environments. in IEEE International Conference on Robotics and Automation. 1994.
[15] Singh, S., Simmons R, Smith T, Stentz A, Verma V, Yahja A, Schwehr K, . Recent Progress in Local and Global Traversability for Planetary Rovers,. in IEEE Conference on Robotics and Automation,. 2000.
[16] Hua-Qing, M., Z. Jin-Hui, Z. Xi-Jing. Obstacle avoidance with multi-objective optimization by PSO in dynamic environment. in International Conference on Machine Learning and Cybernetic. 2005.
[17] Sedighi, K., Ashenay H.,Manikas K. , Wainwright T. W., Tai R. L. H.-M. Autonomous local path planning for a mobile robot using a genetic algorithm. in Congress on Evolutionary Computation (CEC2004). 2004: IEEEXplore.
[18] Schwartz, J., M. Sharir, On the piano mover's problem II: General techniques for computing topological properties of real algebraic manifolds. Adv. Appl. Math.,, 1983. 4: p. 298-351.
[19] Lumelski, V.J., A. A. Stepanov,, Dynamic Path Planning for a Mobile Automaton with Limited Information on the Environment. IEEE Transactions On Automatic Control,, 1986. 11.
[20] Sariff, N., Buniyamin N. Genetic Algorithm versus Ant Colony Optimization Algorithm: Comparison of Performances in Robot Path Planning Application. in 7th International Conference on informatics in control, automation and robotics (ICICINCO 2010), 15th- 20th June. 2010. Madeira, Portugal.
[21] Xiong, L., F. Xiao-ping, Y. Sheng, Z. Heng, A Novel Genetic Algorithm for Robot Path Planning in Environment Containing Large Numbers of Irregular Obstacles. ROBOT,, 2004. 26.
[22] Xiao-Guang, G., Xiao-Wei Fu,Da-Qing Chen. Genetic-Algorithm-Based Approach to UAV Path Planning Problem. in 5th WSEAS Int. Conf. on SIMULATION, MODELING AND OPTIMIZATION, 2005. Corfu, Greece,
[23] N. Bin, C.X., Z. Liming, X. Wendong,, Recurrent Neural Network for Robot Path Planning. Parallel and Distributed Computing: Applications and Technologies, 2004. 3320/2005:(Springer Berlin / Heidelberg).
[24] Bella, J.E., P. R. McMullenb, Ant colony optimization techniques for the vehicle routing problem. Advanced Engineering Informatics,, 2004. 18: p. 41-48, .
[25] Sariff, N., Buniyamin N. Comparative Study of Genetic Algorithm and Ant Colony Optimization Algorithm Performances for Robot Path Planning in Global Static Environments of Different Complexities. in 8th IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA2009) , December 15-18. 2009. Daejon, Korea.
[26] Kumar, E.V., M. Aneja, D. Deodhare,. Solving a Path Planning Problem in a Partially Known Environment using a Swarm Algorithm,. in IEEE International Symposium on Measurements and Control in Robotics. 2008. Bangalore, India.
[27] Khatib, O., Real-Time Obstacle Avoidance for Manipulators and Mobile Robots,. 5, 1986. The International Journal of Robotics Research: p. 90-98.
[28] Susnea, I., Viorel Minzu, Grigore Vasiliu. Simple, real-time obstacle avoidance algorithm for mobile robots. in 8th WSEAS International Conference on Computational intelligence, man-machine systems and cybernetics (CIMMACS'09) 2009.
[29] Ng, J., T. Bräunl Performance Comparison of Bug Navigation Algorithms. Journal of Intelligent and Robotic Systems, 2007. 50: p. 73-84.
[30] Kamon, I., E. Rivlin, Sensory-Based Motion Planning with Global Proofs. IEE Transactions on Robotics and Automation, 1997. 13((6)).
[31] Lumelsky, V., T. Skewis, Incorporating Range Sensing in the Robot Navigation Function. Transactions On Systems, Man, And Cybernetics, 1990. 20((5)): p. 12.
[32] Kamon, I., E. Rimon, E . Rivlin,, TangentBug: A Range-Sensor-Based Navigation Algorithm. The Int. J. of Robotics Research, 1998. 17(9): p. 934-953.
[33] Sankaranarayanan, A., M. Vidyasagar, Path planning for moving a point object amidst unknown obstacles in a plane: the universal lower bound on the worst path lengths and a classification of algorithms. . IEEE International Conference on Robotics and Automation, 1991: p. 1734 - 1741.
[34] Noborio, H., T. Yoshioka. . An On-line and Deadlock-free Path-planning Algorithm Based on World Topology. in IEEE/RSJ Int. Conf. on Intelligent Robots and Systems. 1993. Japan.
[35] Noborio, H., K. Fhjimura, Y. Horiuchi. . A comparative study of sensor-based path-planning algorithms in an unknown maze. in Int. Conf. on Intelligent Robots and Systems, . 2000. Japan.: IEEE,RSJ, .
[36] Sankaranarayanar, A., M. Vidyasagar. . Path planning for moving a point object amidst unknown obstacles in a plane: a new algorithm and a general theory for algorithm development in Decision and Control, . in 29th IEEE Conference on Decision and Control. 1990.
[37] Sankaranarayanar, A., M. Vidyasagar. . A new path planning algorithm for moving a point object amidst unknown obstacles in a plane. in IEEE Conf. on Robotics and Automation (ICRA),. 1990.
[38] Horiuchi, Y.a.H.N. Evaluation of Path Length Made in Sensor-Based Path-Planning with the Alternative Following. in IEEE Int. Conf. on Robotics & Automation. 2001. . Seoul, Korea.
[39] Choset, H., et al.,, Principles of Robot Motion: Theory, Algorithms, and Implementations. . 2005: The MIT Press.
[40] Schwager, M., et al. A Ladybug Exploration Strategy for Distributed Adaptive Coverage Control. in IEEE Int. Conf. on Robotics and Automation. 2008. Pasadena, CA, USA.
[41] S.-K. Kim, J.S.R., and K.-J. Koo, , Construction Robot Path-Planning for Earthwork Operations. Journal of Computing in Civil Engineering, 2003. 17: p. 97-104.
[42] Gabriely, Y.a.E.R. CBUG: A Quadratically Competitive Mobile Robot Navigation Algorithm. in IEEE Int. Conf. on Robotics and Automation 2005.
[43] Gabriely, Y.a.E.R. CBUG: A Quadratically Competitive Mobile Robot Navigation Algorithm. in IEEE Transactions on Robotics, . 2008.
[44] Gabriely, Y.a.E.R., Competitive Disconnection Detection in On-Line Mobile Robot Navigation. Algorithmic Foundation of Robotics (Springer Berlin / Heidelberg), 2008. VII: p. 253-267.
[45] Langer, R.A., L.S. Coelho, G.H.C. Oliveira,. K-Bug, A New Bug Approach for Mobile Robot's Path Planning. in IEEE Int. Conf. on Control Applications. 2007.

Norlida Buniyamin graduated from the University of Adelaide, Australia with a Bachelor Degree in Electrical and Electronic Engineering (Hons.). She was a Research Fellow with the Malaysian Institute of Microelectronic Systems (MIMOS) before joining University Teknologi MARA as a lecturer in 1988. She then obtained a M.Sc. in Industrial Control System from the University of Salford, U.K in 1993 and a PHD in the area of Knowledge Management for Manufacturing Enterprises in 2004 from the University of Manchester, Institute of Science and Technology (UMIST), United Kingdom. She is now an Associate Professor at UiTM and a Fellow of the Institution of Engineers, Malaysia (IEM). Her current research interest is in Industrial Automation and Robotics, Knowledge Management and Engineering Education.

Wan Ahmad Jailani Wan Ngah graduated from the University Teknologi Malaysia with a Bachelor Degree in Computer Engineering. He is the programme Coordinator of Electronic Learning and Multimedia at the Centre for Instructor & Advanced Skill Training (CIAST) in Shah Alam, Malaysia. Jailani is currently pursuing a Masters Degree in Electrical Engineering at the University Teknologi Mara (UiTM), Malaysia in the areas of Mechatronics and Robotics.

Nohaidda Sariff graduated from the Tun Hussein Onn University, Malaysia with a Bachelor Degree in Electrical and Electronic Engineering (Automation and Robotics) in 2003. In 2008, she joined University Teknologi Mara as a lecturer of the Faculty of Electrical Engineering. She is currently pursuing a Masters Degree in Electrical Engineering at the University Teknologi Mara (UiTM), Malaysia in the areas of Mechatronics and Robotics. Her current research interests are Autonomous Mobile Robot, Navigation, and Path Planning Algorithms.

Zainuddin Mohamad graduated with a Bachelor Degree in Mechanical Engineering in 1984 from Northrop University in the United States.

Subsequently, he obtained a Degree in Aerospace Engineering followed by an MSc. in Operational Research from University of London in 1994. He is currently a lecturer of the Faculty of Mechanical Engineering, University Teknologi MARA. He has previously worked as an engineer in the Public Works Department in Malaysia as well as a development and maintenance officer in the development and maintenance department of University Teknologi Mara. Zainuddin is a member of the Institution of Engineers Malaysia and his current research interest is in Robotics, and Engineering Education.