# A Simple Local Pathway Planning Algorithm for Self-directed Mobile Robots

[1]Suresh Balaji.S, [2]Ragunathan. S,[3]Kisorekumar.M,[4]Makesh Raj.S,5Madhan.M, [6]Tamil Kumar.J

[1]Assistant Professor, Department of Mechanical Engineering, Knowledge Institute of Technology, Salem-637504, India.
[2] Principal , AVS Engineering College, Salem-636003, India.
[3,4,5,6]UG Scholar,Department of Mechanical Engineering, Knowledge Institute of Technology,Salem-637504, India.

[1]ssbmech@kiot.ac.in

## ABSTRACT

This paper deals in screening an impression of pathway planningalgorithms for independent robots. The paper also focuses on the bug algorithm family which plays a major role in local pathway planning algorithm. Bug algorithms always use a sensors to sense the forthcoming hindrance as a mobile robot rolls towards the target with imperfect data about the surroundings. The algorithm uses obstacle margin as guidance toward its goal as the robot orbits the obstacle till it discoveries certain condition to accomplish the algorithm standards to leave the hindrance toward target point.

In accumulation, this paper presents a method utilizing a new algorithm named PointBug. This algorithm efforts to minimalize the usage of outer perimeter of an problem (obstacle border) by looking for a few significant points on the outer perimeter of hindrance area as a turning point to board and finally make a complete pathway from source to goal. The less use of outer perimeter of obstacle area yields smaller total pathway span taken by a moveable robot. Further this method is then associated with other existing selected local pathway development algorithm for total distance and a guarantee to influence the target.

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

## I. INTRODUCTION

Pathway planning is one of the most important components formobile robot. Pathway planning is the selection of a pathway that a robot must take in order to pass over each point in an environment [1-4] and pathway is a plan of geometric locus of the points in a given space where the robot is supposed to pass through[5]. Generally, the problem of pathway planning is about finding pathway s by connecting different locations in an environment such as graph, maze and road. Pathway planning which "enables" mobile robots to see the obstacle and generate an optimum pathway so as toavoid it.The general problem of pathway planning for mobile robots is defined as the search for a

pathwaywhich a robot (with specified geometry) has to follow in a described environment, in order to reach a particular location 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. PATHWAY PLANNING ALGORITHMS

### *A. Pathway Planning Approaches*

Various methods have been introduced to implement pathway 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 pathway 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 pathway that should be free of collision and satisfies certain optimization criteria such as shortest pathway . This definition is correct if the objective of solving pathway planning problem is for the shortest pathway because most new approaches are introduced toward shorter pathway . Looking for the shorter pathway does not guarantee the time taken is shorter because sometime the shorter pathway needs complex algorithm making the calculation to generate output is longer.

### *B. Properties of Pathway Planning*

Mobile robot pathway planning has a few chief 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 pathway planningcorrespondss to environment which contains no moving objects or obstacles other than a navigating robot and dynamic pathway planning refers to environment which contains dynamic moving and changing object such as moving obstacle. Meanwhile the local and global pathway planning depend on algorithm where the information about the environment is a priori or not to the algorithm. Suppose if the pathway planning is a global, information regarding the environment already known based on map, cells, grid or etc and if the pathway planning is a local, the robot has got no information about the environment and robot has to sense the environment before it determines to move for obstacle evasion and generate trajectory planning toward target.Mobile robot navigation problem can be splited into three subtask fixed subtasks because the navigation problem varies according to the methods 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 the contact sensor.

## C .Evolution of Robot Environment Modeling

Various types of known static environment models have been imposed by previous robot pathway 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 reproduce 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 pathway to goal. Although the 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 pathway because of the complexity of the process and within a year, another alternative method known as roadmap approach was introduced.

The Roadmap method 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 pathway . Although the approach was successfully implemented in certain 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 nearer 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 instance, Galvaski, who is now investigating its performance [11]. Now, other modeling approaches based on grid have been proposed, such as Quadtree and Framed Quadtree [12-14], to increase the accuracy of the pathway 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 by choosing an appropriate environment model is very significant in robot PP research as it will influence the PP algorithm search process to find the pathway to goal position.

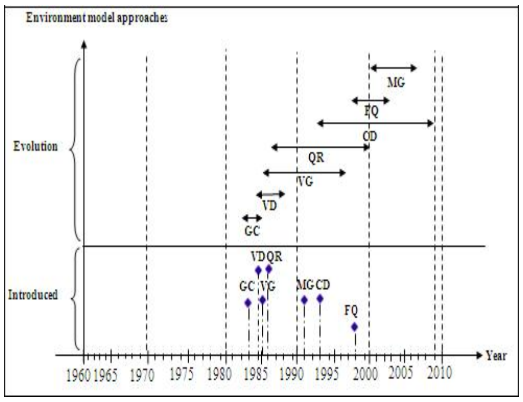

Fig. 1 Evolution of environment modeling approaches from selected sources from early 1980s until 2009.

Indicator:

GC = Generalized Cones(5papers)
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 Pathway Planning*

Global pathway planning is a pathway planning that requires robot to move with priori information of environment. The information about the environment first loaded into the robot pathway planning program before determining the pathway to take from starting point to a target point. In this method the algorithm generates a complete pathway from the start point to the destination point before the robot starts its motion [1]. Global pathway 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 pathway planning, the decision of moving robot from a starting point to a goal is already decided and then robot is released into the specified environment.One of the initial global pathway planning prototypes that extensively studied is Piano’s Mover problem where entire information is assumed to be available on the geometry, positions of the obstacles and the moving object [13]. methods involving moving rigid or hinged bodies into two or three dimensional space have been considered [19]]. Certain common approaches are used in global pathway 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 [10], Neural Network [10] and Ant Colony Optimisation (ACO)[11].

### E. *Local Pathway Planning*

Local pathway planning is pathway planning that requires robot to move in unknown environment or dynamic environment where the algorithm is used for the pathway planning will response to the obstacle and the change of environment. Local pathway 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 [10].

In the local pathway planning, usually, the robot is guided with one straight line from starting point to the target point which is the shortest pathway 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. Wherein this type of pathway 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 [14] is the one of the well known local pathway planning technique. This pathway planningmethodology,where the robot is considered to be a particle moving under influence of an artificial potential reproduced 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 impressive potential and the obstacles are all repulsive potential.

This paper introduces and describes a new local pathway planning algorithm based on the Bug Family of pathway 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 triangulation method for local pathway way planning with minimum sensor and simple process [3]. James Ng and Thomas Bräunl listed the eleven types of bug algorithms [2]. The most commonly used and referred in mobile robot pathway planning are Bug1 and Bug2 [20], DistBug [2], VisBug [13] and TangentBug [14]. Others bug algorithms are Alg1 and Alg2 [14], Class1 [14], Rev and Rev2 [13]; OneBug and LeaveBug [13].

The differences of bug algorithms showed the effort toward shorter pathway planning, shorter timing, simpler algorithm and better performance. Bug1 moves from starting point toward target point by hitting and circumnavigating the obstacle then leaving the leave point. Bug2 has similar performance 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 [13] which calculates shortcuts relative to the pathway generated by the Bug2 algorithm from or to m-line. Alg1 [12] improved Bug2 weakness is that it could trace the same pathway twice by storing the sequence of hit points occurring within an actual pathway to the goal. These storing data are used to generate shorter pathway s 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 look a opposite procedure problem where after come across a visited point that causes loop, a portable robot follows an uncertain obstacle by an opposite direction until it can leave the obstacle. Horiuchi who 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 pathway length and named the algorithm as Rev1 and Rev2 [8]. Alternative methodology which is defined as independently, if a robot always changes its direction following an uncertain obstacle consecutively, the robot arrives at the destination earlier on average and that will decrease the probability for the robot to join a loop around the particular destination.

Various other bug algorithms which also incorporate range sensors are DistBug algorithm and TangentBug Algorithm. The DistBug algorithm is a local pathway planning algorithm that guarantees convergence and will find a pathway if one exists. It requires its own position by utilising 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.

Meanwhile, the TangentBug is another variations of DistBug that improves the Bug2 algorithm in that it determines a shorter pathway 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 hurdle is detected, the robot will start touching around the obstacle and will continue its motion-toward target point monotonous 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 leakage from a local least 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 following nod. 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 [10] 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 pathway s taken among four bug algorithms.

An extension to classical Bug based algorithms called Sensbug was introduced which can produce an effective pathway in an unknown environment with both stationary and movable obstacles [11]. CBug applies the Bug1 behaviour in its algorithm with online navigation algorithm for a size D disc robot moving in general planar environments The algorithm searches a series of expanding ellipses with focal starting point and target point, and its total pathway length is at most quadratic in length of the shortest offline pathway [4]. K-Bug algorithm [4] consumes global information such as obstacles geometry and position to select the waypoint among the vertex of obstacles that caused collisions. This procedure does not use device to get the atmosphere information and the information needs to change every time the situation 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.

The robot is capable to probe the environment using array 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 spin direction of the robot is according to position of traditional 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 nonstop distance in one honest line between sudden point and mark point and its rate 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 quick point found within a single 3600 rotation, the goal is considered inaccessible and the robot halts immediately.

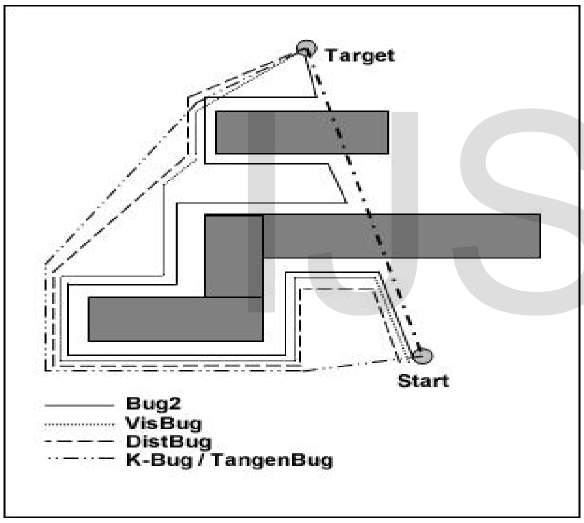

Fig. 2 Trajectories generated by a few bug algorithms

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.

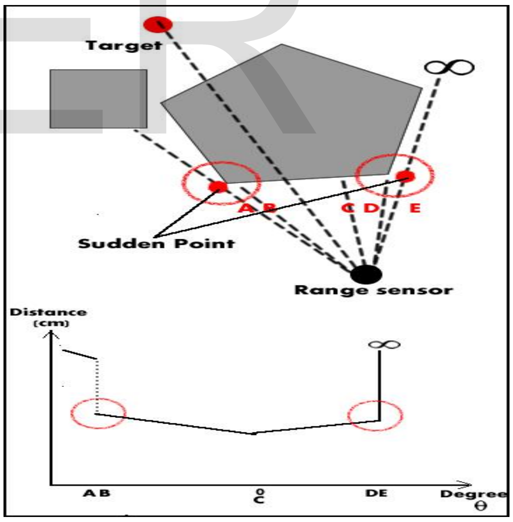

Fig. 3 Range sensor is detecting an obstacle from left to right and right to left.

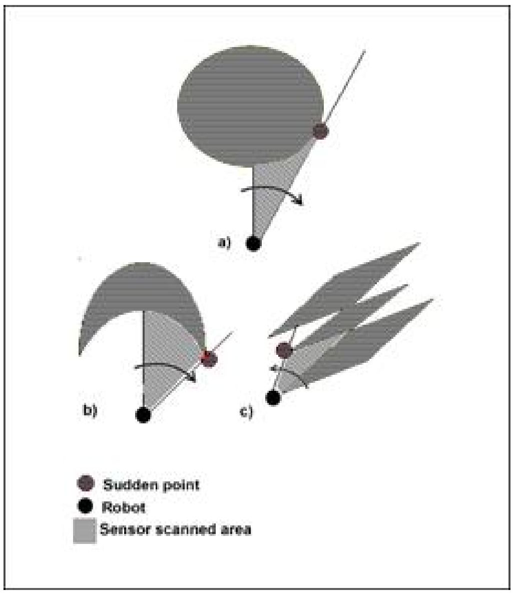

Fig. 4 Sudden points on different surfaces detected by range sensor.

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.

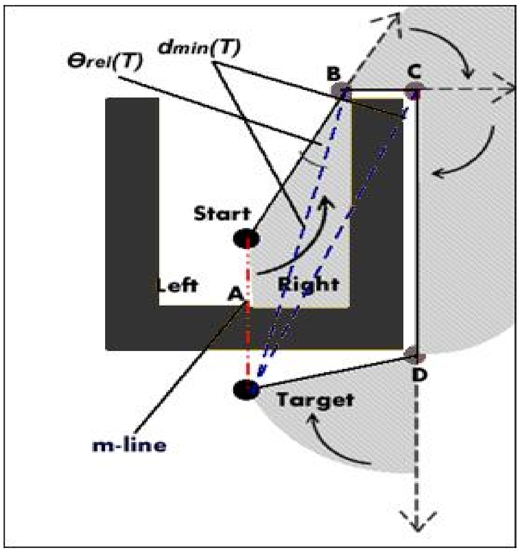

Fig. 5 Trajectory generated by the PointBug to solve local minima problem. The shadowed area is scanned area.

| 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 |

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

## V. POINT BUG ALGORITHM ANALYSIS

The main goal of the algorithm is to generate a continuous pathway 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 pathway 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.

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

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].

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

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

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

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] .The value of оA is obtained from rotation of the robot from current direction to next direction if the robot located on the starting point, otherwise:where is оAdj Adjacent angle of triangle and оRot is the robot rotation angle. оAdj value is obtained from sine rule.

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;

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

$$(d_{\min}[n])^2 = 2b^2 - 2b^2$$

$$(d_{\min}[n]) = 0$$

## 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 pathway from starting point to target point.

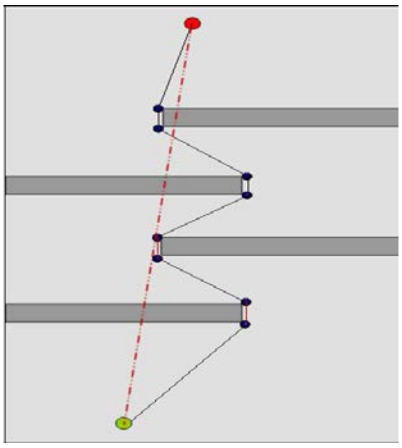

Fig. 6 Trajectory generated using PointBug algorithm in simple maze like type environment.

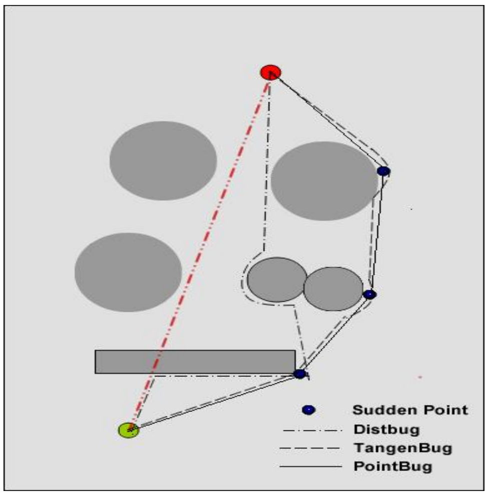

Fig. 7 Trajectory generated using Distbug, TangenBug and PointBug algorithm in Free Environment.

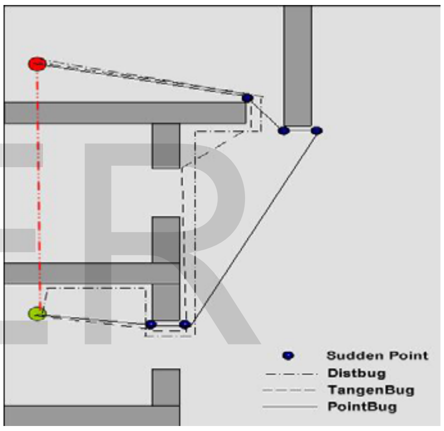

Fig. 8 Trajectory generated using Distbug, TangenBug and PointBugalgorithm in simple Office like Environment.

A Figure 7 and Figure 8 show comparison of three algorithm namely Distbug, TangenBug and PointBug with each robotequipped with unlimited sensor range. TangentBug and PointBug produced almost the same result but TangentBug makes a little obstacle following increases the total pathway 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.

## VII. DISCUSSION AND CONCLUSION

This paper compares 2 main pathway way planning algorithms from the bug algorithm family.This new approach of pathway planning, The Point to Point Sensor Based Pathway Planning algorithm is a new approach that behaves similar to other algorithms in the bug family. However, the Point to Point Sensor Based Pathway Planning Algorithm needs very minimal amount of prior information namely dmin(T) and Ɵrel(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 situation as well because data about the situation 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. Thus, whether it outperforms other bug procedures depends on hurdles 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 hurdle disturbs the total unexpected points.

## REFERENCES

[1] Choset, H., P. Pignon. Coverage Pathway Planning: The Boustrophedon Decomposition. in Int. Conf. on Field and Service Robotics. 1997.

[2] Sariff, N., Buniyamin N. An Overview of Autonomous Mobile Robot Pathway 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 Pathway 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, MihaiTimar,TudorStanciu, RaduBanabic, Octavian Cret. Mobile Robot Pathway - planning Implementation in Software and Hardware. in 6th WSEAS International Conference on Signal Processing, Robotics and Automation. 2007. Corfu Island, Greece.

[5] Haklıdır, M., TaşdelenModelıng And Sımulatıon Of An Anthropomorphıc Robot Arm By Usıng Dymola. in 5th Int. Symp.

[6] Yahja, A., Sanjiv Singh, Barry L. Brumitt. Framed-QuadtreePathway Planning for Mobile Robots Operating in Sparse Environments. in IEEE Conference on Robotics and Automation (ICRA). 1998. Leuven, Belgium.

[7] Yahja, A., Singh S., Stentz A., An efficient on-line pathway planner for outdoor mobile robot. Journal of Robotics and Autonomous Systems 2000. 32: p. 129-143.

[8] 1Stentz, A. Optimal and Efficient Pathway Planning for Partially-Known Environments. in IEEE International Conference on Robotics and Automation. 1994.

[9] 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.

[10] 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.

[11] Sedighi, K., Ashenay H.,Manikas K. , Wainwright T. W., Tai R. L. H.-M. Autonomous local pathway planning for a mobile robot using a genetic algorithm. in Congress on Evolutionary Computation (CEC2004). 2004: IEEEXplore.

[12] 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.

[13] Lumelski, V.J., A. A. Stepanov,, Dynamic Pathway Planning for a Mobile Automaton with Limited Information on the Environment. IEEE Transactions On Automatic Control,, 1986. 11.

[14] Sariff, N., Buniyamin N. Genetic Algorithm versus Ant Colony Optimization Algorithm: Comparison of Performances in Robot Pathway Planning Application. in 7th International Conference on informatics in control, automation and robotics (ICICINCO 2010), 15th- 20th June. 2010. Madeira, Portuga