INTERNATIONAL JOURNAL OF SYSTEMS APPLICATIONS, ENGINEERING & DEVELOPMENT
Issue 3, Volume 3, 2009

The use of linear parametric smoothed curve Path planning for mobile robots

O. Hachour

***Abstract*—**In this present work we present a linear parametric smoothed curve characteristic navigation approach of autonomous mobile robots. The proposed linear parametric curves of navigation are used for path planning where the smoothness of the trajectory around the control point is taken into account. The proposed method starts from an initial point to a target point establishing a control points for which connections are made to determine the form of the path. This algorithm provides the robot the possibility to move from the initial position to the final position (target). The robot moves within the unknown environment by sensing and avoiding the obstacles coming across its way towards the target. The obtained path is the shortest path from all possible free trajectories. This optimality is ensured by using the simulated annealing technique. The Simulation results show that the algorithm generates smooth, optimal path for the robot and the robot always successfully reaches the target. The proposed algorithm can deal with any shape obstacles even if it is the case of circular obstacles. This case is the hardest one in any navigation problem. The problem is solved by proposing some useful solutions for each situation. For any proposed environment, the robot succeeds to reach its target without collisions. The results are satisfactory to see the great number of environments treated. The results are satisfactory and promising for next developments and more design.

***Keywords*—** Autonomous Mobile Robot (AMR), optimal path, parametric curves, and Simulated Annealing.

I. Introduction

THE autonomous robot navigation problem has been studied thoroughly by the robotics research community over the last years. The basic feature of an autonomous mobile robot is its capability to operate independently in unknown or partially known environments. The autonomy implies that the robot is capable of reacting to static obstacles and unpredictable dynamic events that may impede the successful execution of a task. To achieve this level of robustness, methods need to be developed to provide solutions to localization, map building, planning and control. The development of such techniques for autonomous robot navigation is one of the major trends in current robotics research .

The robot has to find a collision-free trajectory between the starting configuration and the goal configuration in a static or dynamic environment containing some obstacles. To this end, the robot needs the capability to build a map of the environment, which is essentially a repetitive process of moving to a new position, sensing the environment, updating the map, and planning subsequent motion. Most of the difficulties in this process originate in the nature of the real world: unstructured environments and inherent large uncertainties. First, any prior knowledge about the environment is, in general, incomplete, uncertain, and approximate. For example, maps typically omit some details and temporary features; also, spatial relations between objects may have changed since the map was built. Second, perceptually acquired information is usually unreliable. Third, a real-world environment typically has complex and unpredictable dynamics: objects can move, other agents can modify the environment, and apparently stable features may change with time. Finally, the effects of control actions are not completely reliable, e.g. the wheels of a mobile robot may slip, resulting in accumulated zoometric errors.

Robot navigation can be defined as the combination of three basic activities:

• Map building: this is the process of constructing a map from sensor readings taken at different robot locations. The correct treatment of sensor data and the reliable localization of the robot are fundamental in the map-building process.

• Localization: this is the process of getting the actual robot's location from sensor readings and the most recent map. An accurate map and reliable sensors are crucial to achieving good localization.

• Path planning : This is the process of generating a feasible and safe trajectory from the current robot location to a goal based on the current map. In this case, it is also very important to have an accurate map and a reliable localization [4].

Moreover, when a robot moves in a specific space, it is necessary to select a most reasonable path so as to avoid collisions with obstacles. Several approaches for path planning exist for mobile robots, whose suitability depends on a particular problem in an application.

The major task for path-planning for single mobile robot is to search a collision –free path. The work in path planning has led into issues of map representation for a real world. Therefore, this problem considered as one of challenges in the field of mobile robots because of its direct effect for having a simple and computationally efficient path planning strategy. For path planning areas, it is sufficient for the robot to use a topological map that represents only the different areas without details such as office rooms. The possibility to use topological maps with different abstraction levels helps to save processing

time. The static aspect of topological maps enables rather the creation of paths without information that is relevant at runtime. The created schedule, which is based on a topological map, holds nothing about objects which occupy the path. In that case it is not possible to perform the schedule [3,9]. To get further actual information, the schedule should be enriched by the use of more up-to date plans like egocentric maps.

Moreover, when a robot moves in a specific space, it is necessary to select a most reasonable path so as to avoid collisions with obstacles. Several approaches for path planning exist for mobile robots, whose suitability depends on a particular problem in an application. For example, behavior-based reactive methods are good choice for robust collision avoidance. Path planning in spatial representation often requires the integration of several approaches. This can provide efficient, accurate, and consistent navigation of a mobile robot.

A robotic system capable of some degree of self-sufficiency is the overall objective of an autonomous mobile robot and are required in many fields . The focus is on the ability to move and on being self-sufficient to evolve in an unknown environment for example. Thus, the recent developments in autonomy requirements, intelligent components, multi-robot systems, and massively parallel computer have made the autonomous mobile robot very used, notably in the planetary explorations, mine industry, and highways [4].

Intelligent autonomous systems designers search to create dynamic systems to navigate and perform purposeful behaviours like human in real environments where conditions are laborious. However, the environment complexity is a specific problem to solve since the environments can be imprecise, vast, dynamical, and partially or not structured. Then, intelligent autonomous Systems must then be able to understand the structure of these environments. To reach the target without collisions, intelligent autonomous systems must be endowed with recognition, learning, decision-making, and actions capabilities. The ability to acquire these faculties to treat and transmit knowledge constitutes the key of a certain kind of intelligence. Building this kind of intelligence is, up to now, a human ambition in the design and development of intelligent vehicles. However, the mobile robot is an appropriate tool for investing optional artificial intelligence problems relating to world understanding and taking a suitable action, such as, planning missions, avoiding obstacles, and fusing data from many sources (12,13].

Recent research on intelligent autonomous systems has pointed out a promising direction for future research in mobile robotics where real-time, autonomy and intelligence have received considerably more weight then, for instance, optimality and completeness. Many navigation approaches have dropped the explicit knowledge representation for an implicit one based on acquisitions of intelligent behaviours that enable the robot to interact effectively with its environment, they have to orient themselves, explore their environments autonomously, recover from failure, and perform whole families of tasks in real-time.

A robot is a "device" that responds to sensory input by running a program automatically without human intervention. Typically, a robot is endowed with some artificial intelligence so that it can react to different situations it may encounter. The robot is referred to be all bodies that are modeled geometrically and are controllable via a motion plan. A robotic vehicle is an intelligent mobile machine capable of autonomous operations in structured and unstructured environment. It must be capable of sensing thinking and acting. The mobile robot is an appropriate tool for investigating optional artificial intelligence problems relating to world understanding and taking a suitable action, such as , planning missions, avoiding obstacles, and fusing data from many sources.

The goal of the navigation process of mobile robots is to move the robot to a named place in a known, unknown or partially known environment. In most practical situations, the mobile robot can not take the most direct path from the start to the goal point. So , path planning techniques must be used in this situation, and the simplified kinds of planning mission involve going from the start point to the goal point while minimizing some cost such as time spent, chance of detection, or fuel consumption.

Systems that control the navigation of a mobile robot are based on several paradigms. Biologically motivated applications, for example, adopt the assumed behavior of animals. Geometric representations use geometrical elements like rectangles, polygons, and cylinders for the modeling of an environment. Also, systems for mobile robot exist that do not use a representation of their environment. The behavior of the robot is determined by the sensor data actually taken. Further approaches were introduced which use icons to represent the environment. One of the specific characteristics of mobile robots is the complexity of their environment, therefore, one of the critical problem for the mobile robots is path planning.

Several approaches for path planning exist for mobile robots, whose suitability depends on a particular problem in an application. For example, behavior-based reactive methods are good choice for robust collision avoidance. Path planning in spatial representation often requires the integration of several approaches. This can provide efficient, accurate, and consist navigation of a mobile robot. It is sufficient for the robot to use a topological map that represents only the areas of navigation (free areas, occupied areas of obstacles). It is essential the robot has the ability to build and uses models of its environment that enable it to understand the environment's structure. This is necessary to understand orders, plan and execute paths.

To deal with a one of largely studying area of research which is now used in computer graphics in design and manufacture computer-assisted, we present the linear parametric curves for the path planning. Several approaches have been addressed to the use of the parametric curves for path planning. A path without collision is presented as a

sequence of curves connecting the initial points to the target in the workspace. The control points connected determine the form of the path for which the problem of optimal trajectory calculation is posed. In this regard, we present an algorithm for the path planning using only one point of control from which we can obtain the remaining points constituting the trajectory.

The use of parametric curves for the path planning is very developed and largely used in computer graphics in design and manufacture computer-assisted. Many researchers have addressed also the smoothness objective, in which they propose an algorithm that generates a path with a low curvature. Some linear parametric curves are used for path planning where the smoothness of the trajectory around the control point is taken into account [1, 2, 5, 6,8,10,11]. Hence, a parametric curve has an inherent directional property which enormously reduces calculations. It is also very important to clarify that the control of mobile robots will be very easy if the curve is expressed in parametric form then the position of the robot can be obtained simply by varying the curve parameter.

For the path planning we suppose that the system is equipped by vision sensors that able to provide information concerning the obstacles. These obstacles can have any complex or simple shape. Our conception treats the problem of moving of an autonomous mobile robot without collisions with any object in the workspace taking into consideration the dimension of that robot.

For our approach, we have used the principle of parametric curves. A parametric curve has an inherent directional property which enormously reduces calculations. A complete path planning algorithm guarantees that the robot can reach the target if possible, or returns a message that indicates that there is no free path when the target cannot be reached. The crucial point of this conception is that the obtained trajectory is smoothed. Furthermore, the returned path is optimal, i.e. this path is the shortest path from the possible free trajectories. This optimality is ensured by using the simulated annealing technique. This approach can be realized in efficient manner and has proved to be robust method due to the problem complexity. This approach based on intelligent computing offers to the autonomous mobile system the ability to realize these factors: recognition, learning, decision-making, and action (the principle obstacle avoidance problems). The results are promising for next development.

II. The use of A SMOOTHED curve characteristic path planning

The use of the parametric curves for path planning was very developed and largely studied [2]. The idea is to present the trajectory without collision which is regarded as a series of curves connecting the initial points to the target in the workspace (see the figure1). The control points for which connection were made determines the form of the trajectory. For this case the problem of optimal trajectory calculation is posed.

In this work, an algorithm is developed for the path planning using only one point of control from which we can obtain the remaining points constituting the trajectory. Here we suppose that the path planning is designed in a real and known environment workspace i.e., the vision or the system of autonomous sensors is able to provide information concerning the obstacles. More, the first part of our work supposes that robot is considered as a material point (the case study of moving only material point to another position without collision with any object in the workspace). The second part will be interpreted in another work where the dimensions of the robot are taken in consideration (3D conception). This work will be submitted with this article.

First we examine if there is a linear connection trajectory from the initial point to the target points. If this connection trajectory collides an obstacle, a point of control is introduced between the initial and the goal point and a point of intermediate connection is created once again. This control point can be structured in a coordinated way, and a checking of interference between the trajectory and the obstacles produces a map. We have started our work by studying the problem of control in two (02) dimensions where two parameters are necessary to define a control point, and four parameters are necessary to define two points of control. From where a certain number of control points define the dimension of the Control Point Space CPS. An interference checking produces images in the CPS and this tracing is defined by a geometrical layout. In the case of three (03) 3D dimensions, three parameters are necessary to define a control point and the problem can be reduced to a problem of two dimensions by projecting the objects on the plan containing the initial point, the goal point and the control point.

Note that the free space is a surface which is belonging to the CPS, but unoccupied by the image of the obstacles. The geometrical layout is a computing process. The complete construction of CPS is not necessary for the practical problems. The construction of the CPS depends on the manner in which the control points are defined. Two methods of control point definition are presented here: one for the control point in a circular workspace and another one in an elliptic workspace, i.e. in polar coordinates; this last method eliminates tiresome calculations for the checking of interference after smoothing.

The control point can define a trajectory without collision in the free space. The Free space (FS) is a surface belonging to the CPS, but unoccupied by the image of the obstacles. A control point is defined by a circular space has the property to have the same total length starting from the initial point to the goal point for a given value of ρ without taking into consideration Θ. That makes possible the search for optimal trajectory in a given minimum time starting from $\delta = 0$ to $\delta = \delta_{\max}$.The smoothing around the point of connection is preferable for a better robot control. Smoothing by using quadratic parametric curves is presented in section III.3, where the known field of the parameter ρ is used to obtain the

smoothing interval of the original curve. Smoothing can be carried out for the entire trajectory if the known interval of ρ satisfied the condition $\rho_{min}<0.5\rho_{max}$. For the elliptic workspace of the control point, an additional calculation is necessary to transform δ into ρ. As the definite field of ρ takes into consideration the interference with the obstacles, we will not need to repeat the checking of interference after smoothing.

The smoothing around the point of connection is very important for a better robot control, considering the physical limits of the robot actuators, safety issues and that some service robots are required to carry sensitive loads. We clarify more the problem of smoothing by using quadratic parametric curves is presented in section 3, where the known field of the parameter ρ is used to obtain the smoothing interval of the original curve. Smoothing can be carried out for the entire trajectory if the known interval of ρ satisfied the condition. As the definite field of ρ takes into consideration the interference with the obstacles, the algorithm will not need to repeat the checking of interference after smoothing. A common resolution of CPS will be able to have as a result a space completely occupied by the obstacles images; although, may be it will have an available free space. In the case that the CPS is completely occupied by the image of the obstacles we need to define and add another control points or the use of higher order degree format of a curve.

III. The Proposed Parametric Curves and Trajectories Without Collision

A. Direct trajectory and the linear parametric curve

A linear parametric curve passing through two points can be expressed as follows (see the figure 1):

$$r_{(s)} = (T-S)\times s + S \qquad (1)$$

Where S and T represent the coordinates of the starting and the target points, the trajectory ST is shown in figure 2. s represents the curve parameter and varies from zero (0) to one (1). The directional property of the parametric curve enables us to express the intersection of two segments of a line in simple terms. To determine if the line ST collides an object we must carry out a checking of the intersection of the line ST with the polygonal contours of the obstacles. In the figure 2, the line ST has the tendency to intersect with the contour of the obstacle Obs2 in two points (1) and (2), such that $S_1 \le S \le S_2$. The calculation of the intersection between the line segment ST and the obstacle contour P_iP_j can be expressed as follows :

$$(T-S)\times s + S = (P_j - P_i)\times t + P_i \qquad (2)$$

where t: represents the curve parameter; In the case of 2D the vectorial equation (2) is composed of two equations with two unknowns (s and t parameters). The parameter s and t can be solved by eliminating the other:

$$s = \frac{(P_{ix}-S_x)(P_{jy}-P_{iy}) - (P_{iy}-S_y)(P_{jx}-P_{ix})}{(T_x-S_x)(P_{jy}-P_{iy}) - (T_y-S_y)(P_{jx}-P_{ix})} \qquad (3)$$

$$t = \frac{(T_x-S_x)(S_y-P_{iy}) - (T_y-S_y)(S_x-P_{ix})}{(T_x-S_x)(P_{jy}-P_{iy}) - (T_y-S_y)(P_{jx}-P_{ix})} \qquad (4)$$

The segment ST cuts the contour P_iP_j if $0 \le S \le 1$ and $0 \le t \le 1$,else there is no intersection. An exception can take place when the denominator of the expressions (3) and (4) is null i.e.:

$$\frac{T_y - S_y}{T_x - S_x} = \frac{P_{jy} - P_{iy}}{P_{jx} - P_{ix}} \qquad (5)$$

This case gives us the coincidence of the slopes of the line ST with that of the contour P_iP_j. This explains that when the denominators of the two equations (3) and (4) are null, the two lines are in parallel. Consequently, no calculation is necessary when they are null.

In the workspace, we can define the position of the obstacles with respect to the line ST, and there exist three cases: the obstacles which cut the line ST, which are above the line ST and which are below the line ST. In the case where there is no intersection between the obstacle and the line ST, like the case of the obstacles Ob_1, Ob_2, Ob_3, and $Ob4$ which are presented in the figure2, an additional calculation is necessary in order to locate its position with respect to ST. This does not require a precise value but a relative value with the aim of making a comparison. A method which does not use the trigometrical functions consists in calculating the determinant of the equation :

$$A = \begin{vmatrix} S_x & T_x & P_{ix} \\ S_y & T_y & P_{iy} \\ 1 & 1 & 1 \end{vmatrix} \qquad (6)$$

This equation (6) gives the double of the area of the triangle STP_1 (see the figure 4), where the order (orientation) determines of the points constituting the vertices of the obstacle. The equation (6) indicates if a vertex P_1 of an obstacle is on top, below or on the line ST. Three cases arise:

1. $|STP_1|>0$, in this case S,T and P_1 are ordered in the clockwise direction, i.e. the obstacle is bellow the line ST.
2. $|STP_1|<0$¶, in this case S,T and P_1 are ordered in the counter clockwise direction, i.e. the obstacle is above the line ST.
3. $|STP_1|=0$¶, in this case S,T and P_1 are in the same line.

In order to reduce calculations this checking is limited to the cases where obstacles are in collision with the line *ST*.

B. The control point and circular workspace (CWS)

Previously we described the case of interference of a direct trajectory *ST* with an obstacle. If it is the case, a control point *Q* is suggested to be an intermediate point of connection (see figure 3). In this figure, the polar coordinates are used to define a control point with the parameters θ and ρ. The arbitrary value of θ and ρ is fixed between $0° < \theta < 360°$ (θ not equal 0, 180°) and $0 < \rho \leq 1$.

When θ =0°,180°,360° or ρ=0, Q is on the line ST and the trajectory degenerates into a direct trajectory. The extreme value of Q_p=1 is shown in figure 3 and belongs to the effective working zone. As long as the center *C* of CWS is in the middle of *ST*, the distance 'd' between C and the extreme limit of AWS is expressed as follows:

$$d = \sqrt{(W_x - C_x)^2 + (W_y - C_y)^2} \qquad (7)$$

Where (W_x,W_v) represent the coordinates of a point belong to the boundary working zone. The values of Q_{max} correspondent to the maximum values of D and are given by the expressions:

$$\begin{cases} Q_{max_x} = \frac{1}{D}\left\{(T_x - S_x)(d_{max}\cos\theta + C_x) - (T_y - S_y)(d_{max}\sin\theta + C_y)\right\} \\ Q_{max_y} = \frac{1}{D}\left\{(T_y - S_y)(d_{max}\cos\theta + C_x) - (T_x - S_x)(d_{max}\sin\theta + C_y)\right\} \end{cases} \qquad (8)$$

Where D represents the distance ST, i.e. D=|T-S| . The system of equations (8) can be used to locate the extreme positions of CWS. The edges of the contour of AWS can be treated as being the boundary of the obstacles (area work space), and the circular obstacles can be represented by a circumscribed polygon (polygon with the maximum corners) as shown in the figure 3 (Obs4). After the determination of CWS we determine the maximum radius, and the image of the obstacle in the CPS can be obtained as follows: for the discrete values of θ we determine the possible values of ρ by taking into account the phenomenon of interference and are traced in the CPS. The checking of interference with the obstacles must be carried out before the tracing of the two line segments SQ and QT.

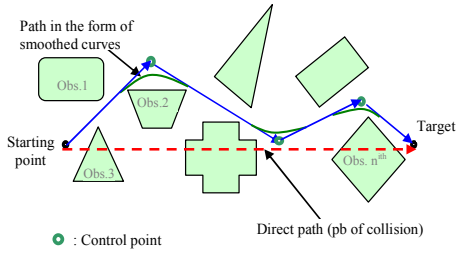

Fig.1 Path in form of a series of linear segments connected by curves

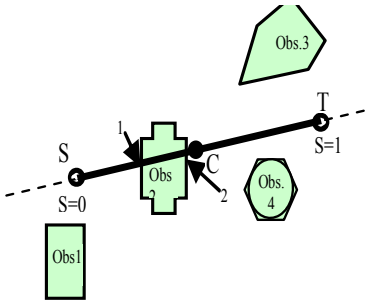

Fig. 2 Parametric linear curve with obstacles

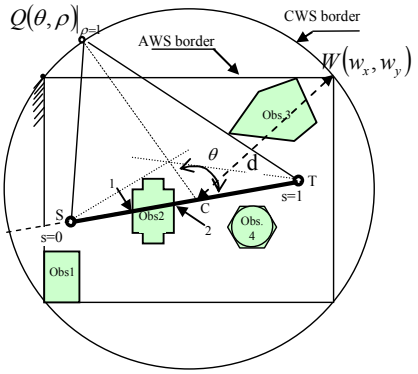

Fig. 3 the intermediate connection Points and the linear trajectory

A typical case is considered in the figure 4. Note that there exist two types of interferences between an obstacle and a trajectory:

- The trajectory passes through a point coinciding with one of the vertexes of the obstacle.
- The point of control is in contact with the edge of the obstacle (tangent).

The equations (9) express the conditions for the first case.

$$\begin{cases} \{(Q_{\max} - C)\rho + C - S\}r + S = P \\ \{(Q_{\max} - C)\rho + C - T\}s + T = P \end{cases} \quad (9)$$

In the equation (9), the parameters r and S express the distances SQ and QT respectively, and P indicates one of the vertices of the obstacle contour (see figure 4). Calculation must be carried out for all the vertices in order to carry out a complete interference checking. The *p* value can be obtained after elimination of the parameters r and s of the equations (9) as follow:

$$\begin{cases} \{(Q_{\max,x} - C_x)\rho + C_x - S_x\}r + S_x = P_x \\ \{(Q_{\max,y} - C_y)\rho + C_y - S_y\}r + S_y = P_y \end{cases} \quad (10)$$

By eliminating the parameter r we find:

$$\rho = \frac{(P_x - S_x)(C_y - S_y) - (P_y - S_y)(C_x - S_x)}{(Q_{\max,x} - C_x)(P_y - S_y) - (Q_{\max,y} - C_y)(P_x - S_x)} \quad (11)$$

And in order to eliminate the parameter s we use the equation (12):

$$\begin{cases} \{(Q_{\max,x} - C_x)\rho + C_x - T_x\}s + T_x = P_x \\ \{(Q_{\max,y} - C_y)\rho + C_y - T_y\}s + T_y = P_y \end{cases} \quad (12)$$

We get:

$$\rho = \frac{(P_x - T_x)(C_y - T_y) - (P_y - T_y)(C_x - T_x)}{(Q_{\max,x} - C_x)(P_y - T_y) - (Q_{\max,y} - C_y)(P_x - T_x)} \quad (13)$$

The equations (11) and (13) return the value of ρ for a Q position of the SQ and QT line segment. These ρ values are different except if P is on the line $CQ_{\max}$. The denominator of the equation (13) can be cancelled when the lines SP and $CQ_{\max}$ or TP and $CQ_{\max}$ are in parallel. This condition could be avoided by comparing the denominator with the numerator as long as ρ must belong to the range 0 and $\rho_{\max}$ so that the real contact takes place. As we obtain two values of ρ, additional computations are necessary to determine if the point P is located inside the triangle $SQ_{\max}T$. The equations (9) can be rearranged to eliminate ρ and to have the values of r and s.

$$r = \frac{(Q_{\max,y} - C_y)(P_x - S_x) - (Q_{\max,x} - C_x)(P_y - S_y)}{(Q_{\max,x} - C_x)(S_y - C_y) - (Q_{\max,y} - C_y)(S_x - C_x)} \quad (14)$$

$$s = \frac{(Q_{\max,y} - C_y)(P_x - T_x) - (Q_{\max,x} - C_x)(P_y - T_y)}{(Q_{\max,x} - C_x)(T_y - C_y) - (Q_{\max,y} - C_y)(T_x - C_x)} \quad (15)$$

If the vertex P is located inside the triangle $SQ_{\max}T$, the parameters r and s vary between 0 and 1. The pair of equations (11) and (14) or (13) and (15) can be used to determine if the vertex P is inside the triangle $SQ_{\max}T$ or not.

When the control point is in contact with the edge of the obstacle, the line segment $CQ_{\max}$ intersected with the contour. In this case ρ can be calculated by using the equation (17).

$$(Q_{\max} - C)\rho + C = (P_2 - P_1).t + P_1 \quad (16)$$

$$\rho = \frac{(P_{1,x} - C_x)(P_{2,x} - P_{1,x}) - (P_{1,y} - S_y)(P_{2,x} - P_{1,x})}{(Q_{\max,x} - C_x)(P_{2,y} - P_{1,y}) - (Q_{\max,y} - C_y)(P_{2,x} - P_{1,x})} \quad (17)$$

$$t = \frac{(Q_{\max,x} - C_x)(C_y - P_{1,y}) - (Q_{\max,y} - C_y)(C_x - P_{1,x})}{(Q_{\max,x} - C_x)(P_{2,y} - P_{1,y}) - (Q_{\max,y} - C_y)(P_{2,x} - P_{1,x})} \quad (18)$$

An exception is made when the line segment $CQ_{\max}$ and contour P_1P_2 are in parallel. With the combination of the two cases above, all the exceptional cases can be solved easily. On figure 4, the well defined ρ values are represented with red discontinued lines where $0 < \rho < \rho_1$ and $\rho_2 < \rho < \rho_3$. As each obstacle produced well defined ρ ranges, the union of these ranges gives us as a result a well defined ρ ranges for a given value of θ. The checking of interference between the triangle $SQ_{\max}T$ and the obstacle edge must be made using the equation (2) to determine the position of the obstacle compared with the triangle $SQ_{\max}T$. The checking of interference must be made for all the obstacles according to the above procedure. The Figures 5 and 7 show a typical case

of geometrical planning. The shading surface of the figure 4 indicates that all control points are belonging to this surface which intersect with the line segment SQ or QT. The surface defined by A_1, A_2, A_3 and A_4 of CPS. The figure 5 shows *the trajectories without collision.*

The obstacles in the Euclidian space are represented by a complex form that show only the surface for the control point Q. As the process end up when a feasible path is found, the construction of the whole map is not necessary, unless there are other constraints which must be taken on consideration. According to figure 5, it is clear that the trajectory taken from A_3 and A_4 cross behind $Ob1$ and $Ob3$ respectively. The circle C_1 and C_2 are the inscribed circle with an important diameter to be more suitable for surfaces A_3 and A_4 respectively, and C_1 is largest than C_2. The path gotten from C_1 is far from obstacles comparing to any other path can be regarded as the most certain trajectory. It is one of the useful properties of this geometric planning.

When the control point is in contact with the edge of the obstacle, the line segment $CQ_{\max}$ intersected with the contour. In this case ρ can be calculated by using the equation (17).

$$(Q_{\max} - C)\rho + C = (P_2 - P_1).t + P_1 \quad (16)$$

$$\rho = \frac{(P_{1,x} - C_x)(P_{2,x} - P_{1,x}) - (P_{1,y} - S_y)(P_{2,x} - P_{1,x})}{(Q_{\max,x} - C_x)(P_{2,y} - P_{1,y}) - (Q_{\max,y} - C_y)(P_{2,x} - P_{1,x})} \quad (17)$$

$$t = \frac{(Q_{\max,x} - C_x)(C_y - P_{1,y}) - (Q_{\max,y} - C_y)(C_x - P_{1,x})}{(Q_{\max,x} - C_x)(P_{2,y} - P_{1,y}) - (Q_{\max,y} - C_y)(P_{2,x} - P_{1,x})} \quad (18)$$

An exception is made when the line segment $CQ_{\max}$ and contour P_1P_2 are in parallel. With the combination of the two cases above, all the exceptional cases can be solved easily. On figure 4, the well defined ρ values are represented with red discontinued lines where $0 < \rho < \rho_1$ and $\rho_2 < \rho < \rho_3$. As each obstacle produced well defined ρ ranges, the union of these ranges gives us as a result a well defined ρ ranges for a given value of θ. The checking of interference between the triangle $SQ_{\max}T$ and the obstacle edge must be made using the equation (2) to determine the position of the obstacle compared with the triangle $SQ_{\max}T$. The checking of interference must be made for all the obstacles according to the above procedure. Figures 5 and 7 show a typical case of geometrical planning. The shading surface in the figure 4 indicates that all control point belonging to this surface intersect with the line segment SQ or QT. The surface defined by A_1, A_2, A_3 and A_4 of CPS in the figure 5 shows the trajectories without collision. The obstacles in the Euclidian space are represented by a complex form that show only the surface for the control point Q. As the process end up when a feasible path is found, the construction of the whole map is not necessary, unless there are other constraints which must be taken into account. For the figure 5, it is clear that the trajectory taken from A_3 and A_4 cross behind $Ob1$ and $Ob3$ respectively. The circle C_1 and C_2 are the inscribed circle with an important diameter to be more suitable for surfaces A_3 and A_4 respectively, and C_1 is largest than C_2. The path gotten from C_1 is far from obstacles comparing to any other path can be regarded as the most certain trajectory. It is one of the useful properties of this geometric planning.

C. Connection smoothing using a quadratic parametric curve

Until now, the trajectory that we considered is discontinuous in its first derived at the point $Q_{(\theta,\rho)}$ (figure 3). To have a continuous trajectory it is necessary to smooth the curve.

The trajectory connecting the point S and T must be continuous and derivable of second degree. According to figure 6, R_a describes the portion of the parametric trajectory R_0R_1, and R_b describes the other portions, R_1R_2 and R describe the parametric trajectory R_aR_b where s is the parameter of the line segments R_0R_1 and R_1R_2. R_aR_b is the valid range and $0 \le s \le 1$.

Knowing that:

$$R_a = (R_1 - R_0).s + R_0 \quad (19)$$

$$R_b = (R_2 - R_1).s + R_1 \quad (20)$$

$$R = (R_b - R_a).s + R_a \quad (21)$$

By replacing the terms by their values in the equation (21) we obtain

$$R = (R_2 - 2.R_1 + R_0).s^2 + 2.(R_1 - R_0).s + R_0 \quad (22)$$

We replace R_0, R_1 and R_2 by S, Q and T respectively; we obtain:

$$R = (T-2.Q+S).s^2 + 2.(Q-S).s + S \qquad (23)$$

Thus we obtain a quadratic curve of Bezier, i.e. a parabola; the process can be generalized for a polygon with n vertexes which correspond to a Bezier curve of $n-1$ degree.

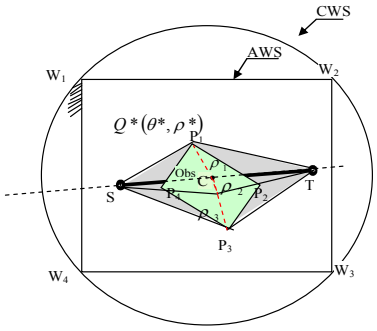

Fig. 4 The intersection computation for a given $Q^*(\theta^*, \rho^*)$ ranges.

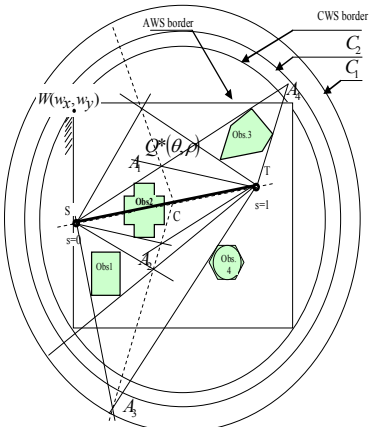

Fig. 5 Euclidian space of the obstacles

The lines SQ and QT are tangent to the curve at points S and T respectively as shown in the figure 6. The curve passes through the middle of the line CQ; thus $\rho_a = 0{,}5.\rho_b$. This relation can be confirmed using the equation (24) after the substitution of $s = 0{,}5$ and $C = 0{,}5.(S+T)$

$$\frac{(Q-R)}{(Q-C)} = \frac{1}{2} \qquad (24)$$

This fact ensures that a connection submitted to a smoothing can be made without taking into account the other considerations relating to the interferences; if a trajectory family is formed by the control point $Q = Q_{|\theta=\theta^*}$ with a range of ρ vary from $0{,}5.\rho_b$ to ρ_b, then this trajectory family does not collide with any obstacles.

The trajectory SQ^uT is the smoothing result of the whole linear trajectory SQ_2 and Q_2T as shown in figure 8. It is possible that ρ_1 is smaller than ρ^u, where $\rho^u = 0{,}5.\rho_2$ for the trajectory SQ^uT. Given that any linear trajectory between ρ_1 and ρ_2 does not contain any obstacles for a given value of θ_a, the lower plug of the quadratic trajectory can be obtained from ρ_1 by using the relation $\rho^1 = 2.\rho_1$. The trajectory SQ^1T is the smallest for quadratic trajectories family for θ_a which does not need lot of calculations to check the interference with the obstacles. For the family of curves θ_b, the smoothing of the whole trajectory is impossible as long as $\rho_1 > 0{,}5.\rho_2$ The maximum softness, the constraints points R_1 and R_2, which gives the largest softness area can be obtained from ρ_1 and ρ_2. The points R_1 and R_2 can be calculated by using the following geometrical relation; see the trajectory SPT of the figure 8.

$$\frac{P_2 - C^*}{P_2 - C} = \frac{P_2 - R_1}{P_2 - S} \qquad (25)$$

where

$$P_2 - C^* = 2.(P_2 - P_1).R_2 \qquad (26)$$

There is a removal between the trajectory SQ^1T and the obstacle $Obs2$, and between SPT and $Obs1$. This is due to the nature of the smoothing which depends on the linear trajectory. The concave polygon SW^1TQ^u provides a passage without collision for the quadratic trajectory SQT, and SP_1TP_2 provides a passage without collision for the trajectory SPT via R_1 and R_2. Smoothing can also be carried out to have a C^2 continuity at the connection points. Having two multiple of vertices at point Q and point P will

result in Bezier curve. A Bezier cubic curve formed by four points is expressed as follows by using the same notation as that of the equation (22).

$$R = \left(R_3 - 3R_2 + 3R_1 - R_0\right)\cdot s^3 + 3\cdot\left(R_2 - 2R_1 + R_0\right)\cdot s^2 + 3\cdot\left(R_1 - R_0\right)\cdot s + R_0 \quad (27)$$

Now applying two multiple of vertices, i.e., $R_1 = R_2$, with the substitution of $P_0 = R_0$, $P_1 = R_1 = R_2$ and $P_2 = R_3$ the equation (27) can be rearranged as follows:

$$R = \left(P_2 - P_0\right)\cdot s^3 + 3\cdot\left(P_1 - P_0\right)\cdot\left(s - s^2\right) + P_0 \quad (28)$$

The cubic Bezier curve is close to the linear trajectory than the quadratic Bezier curve. This cubic curve passes through the quarter of the line segment CP_1, refer to figure 4 for the notation of vertices. By substituting $s = 0.5$ in the equation (28), we get the following relation :

$$\frac{P_1 - R}{P_1 - C} = \frac{1}{2}$$

By using the relation (29), the range of minimum to the maximum field of ρ which does not interference with obstacles, $\rho_1 < \rho < \rho_2$, can be reduced to $\rho_1 = 0.75 \times \rho_2$, see trajectory SPT of the figure 8.

This doubles the interval of smoothing comparing to the quadratic curve. For example, in figure 8, the trajectory SPT can be smoothed from any point. The equation (24) can also be applied for the smoothing of a portion of the trajectory by substituting:

$$P_2 - C^* = 2^2\left(P_2 - P_1\right)$$

The degree of Bezier curve can be increased as we want to suit the original linear trajectory; however, higher is the Bezier curve degree lower is the radius of curvature beside the connection point Q or P and spends much computing time.

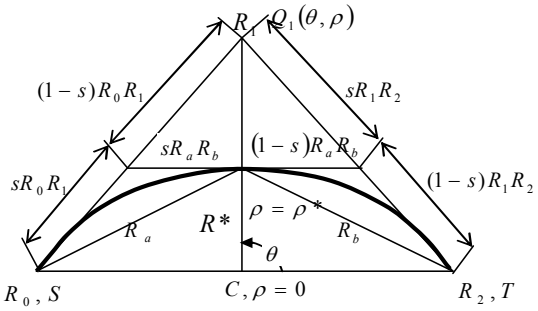

Fig. 6 Relationship between the linear and quadratic parametric curve

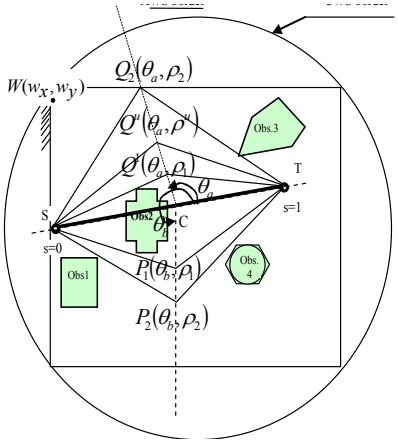

Fig. 7 The geometrical planning of the environment

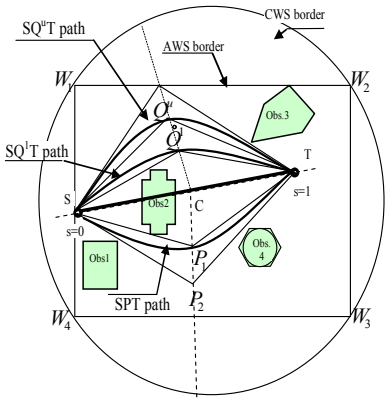

Fig. 8 The path resulted from smoothing

IV. ANNEALING SIMULATED TECHNIQUE

The annealing simulated emulates the physical process on two phases: Increase the solid temperature up to its liquid state, and then cool down slowly the liquid up to its crystallization. Its objective is to generate solids with very low energy level.

The calculation algorithm can be summarized as follows:

- let's c be a candidate solution for the problem with an associated cost c_d,
- generate a solution t in the vicinity of the first solution with a cost c_t such that $c_t < c_d$ in this case we take the modification as the best solution comparing to the previous solution and we do other iteration again.

Once $c_t > c_d$, we generate a random number x where $(0<x<1)$ and we verify if $x < \exp\left[\frac{(c_t - c_d)}{T}\right]$, on this condition we take the modifications, else we reject them and we iterate other times. This step is used to avoid the local minimum, by allowing the algorithm to explore alternatives which seem to be considered.

The T value is used as an adjustment parameter of the acceleration proportion for the solutions of the equations (14) and (15). We present bellow an algorithm that allows us to get an optimal local path starting from an initial configuration q_i and ending with target q_g (see algorithm 1):

Use the algorithm of the fourth point to find all trajectories without collision.

Chose any random trajectory and evaluate its cost $(p\_\cos t)$.

Do *{suggest another trajectory and evaluate its cost* $(trial\_\cos t)$.

If $(trial\_\cos t < p\_\cos t)$ ***then*** *{accept the trajectory* t *as the best solution till now*

Goto *13*

Else *{ let* x *a random variable varies between 0 and 1*

If $(x < (\exp(p\cos t - trial\cos t)/T))$ ***then*** *{accept the trial trajectory* $(trial\_path)$

Goto *13*

Else *{reject the trial trajectory* $(trial\_path)$

iteration = iteration + 1

}

While $(iteration < MAX\_ITER \ or \ stop <> true)$ ***Do***

Print *(the path and its cost)*

Algorithm1: The local optimal path algorithm

V. SIMULATION RESULTS

The algorithm is written under Visual Basic software, and we tested it using AutoCad software which helps us also to construct the obstacles. Simulation results show that the algorithm generates smooth, optimal path for the robot and the robot always successfully reaches the target. The obstacles and the trajectory are presented on 2D environment, the starting point and the target are shown by small spherical objects, and the linear trajectory is shown by continuous violet line segments and the red spline shows the smoothed path. In the figures 9, figure 10 and figure 11 we present one example of the algorithm when only one control point is generated. In these figures the robot moves from the start point coordinates $S(-10,147;0)$ to the target point coordinates $T(328,147;0)$, the path without collision is generated via one control point. We can start from any initial point and the process responds(i.e. the robot finds its target without collisions).

In the figures 12, figure 13 and figure 14 we present another example of the algorithm of the same environment when two control points are generated. In this present result the coordinates of source S and target T are changed (This is more important in our algorithm). For any coordinates of Source and target (This is done by using AutoCAD programming environment) the robot finds its best trajectory without collisions. More, the problem of graphical situations and recognition of environment are solved using AutoCAD tool. In these figures the robot moves from the start point $S(2,2;0)$ to the target $T(290,80;0)$, in this case the path without collision is generated via two points of control Q_1 and Q_2.

As it is clear presented, the linear trajectory of the figure 13 is more close to its smoothing shown in figure 14 than is the linear trajectory of figure 10 which is somehow far from its smoothing trajectory shown in figure 9.c. the influence of the number of control points on the smoothing of the trajectory can be more clear in next figures (figure number's 15, 16, 17, 18, 19, 20). We clarify more our method by the figures 18, figure 19, and figure 20 where the numbers of control points is increased (case for example of nine (09) control points). The increased number of control points will help us in deducing some remarks (to be explained later).

The number of control points increases by the complexity of the environment through the start point and the target, and higher is the number of control points better is the smoothing trajectory and greater is the computation time to get results. The algorithm can be applied in more complex environment as shown in the figure 21, figure 22, and figure 23. If the environment across the start point and the target is free from any obstacle, for this case the resulted path will be a line trajectory with no control point and no smooth (see the figure 21). In the worst case, when the robot is surrounded by obstacles, i.e., there is no path for the robot to reach the target,

the algorithm works as follow:

- Draw line from start point forward the target and check for collision with the obstacle.
- As the collision is true, then keep rotating with small θ angles on the reverse clockwise direction and check for collision for each θ value until $\theta = 180°$.
- As the collision is always true, initialize the θ value to small negative angle and still decreasing θ by small negative angles, i.e., rotate in the clockwise direction, and for each rotation check for collision.
- Once $\theta = 180°$, and collision is true, the algorithm return a message that indicates that there is no feasible path.

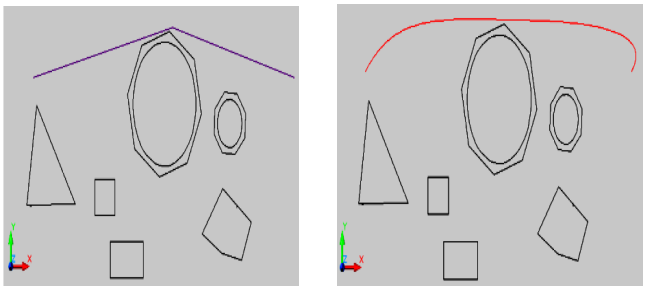

Fig. 10 The linear trajectory and the smoothed path of the simulation result of one environment where one control point is generated

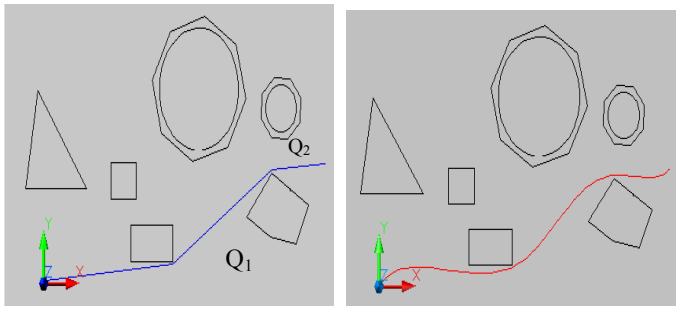

Fig. 13 The linear trajectory and the smoothed path of the simulation result of the last environment where two control points are generated

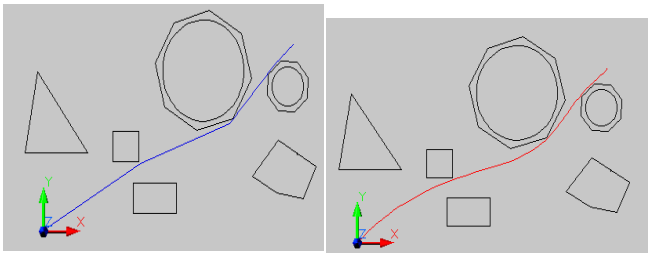

Fig. 17 the Smoothed path of this environment
The simulation result where the path is generated from three control points

The control can be done by more numbers of controls. The number of control points increases by the complexity and the format of the environment through the start point and the target. When the number is greater, the smoothness is more done and better for the trajectory. More, when the number is increased the computation time is much and hard to be implemented.

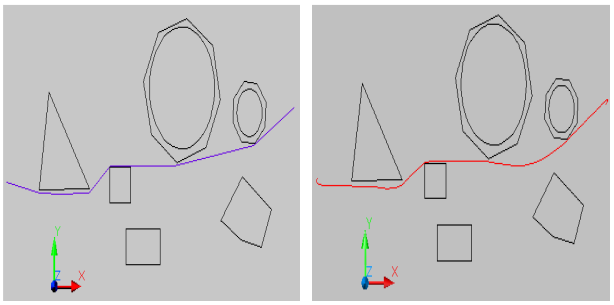

Fig. 19 The linear trajectory
The simulation result where the path is generated from nine control points

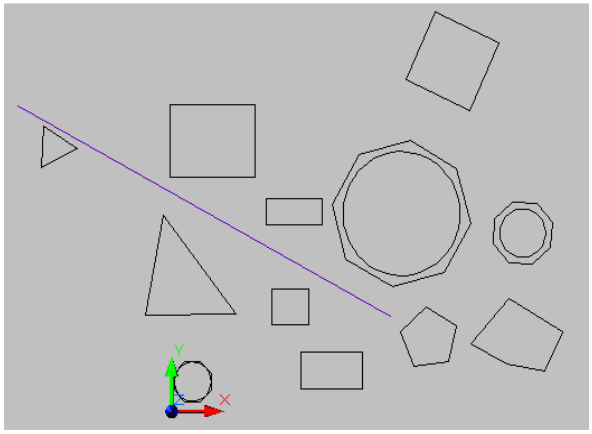

Fig. 21 Path in free environment
Path planning in congested environment

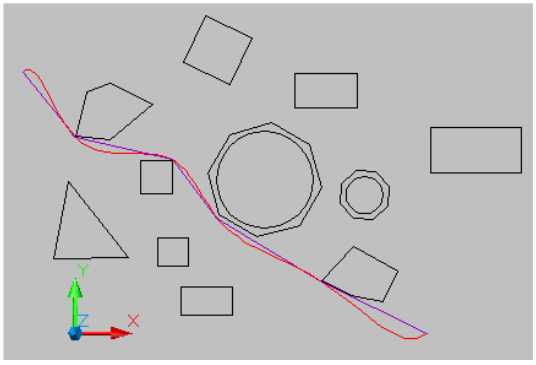

Fig. 22 Navigation in environment2
Path planning in congested environment

VI. Conclusion

In this paper we studied the path planning problem of an autonomous robot operating in a 2-dimentional surface with obstacles. A complete path planning algorithm guarantees that the robot can reach the target if possible, or returns a message that indicates that there is no free path when the target cannot be reached.

There are also a very crucial performance consideration, i.e., the trajectory was smoothed. Furthermore, the returned path is optimal, i.e. this path is the shortest path from the possible free trajectories. This optimality is ensured by using the simulated annealing technique Actually the algorithm can deal with any shape obstacles but for the circular obstacles they be surrounded by a polygon with maximum number of edges, in the simulation we supposed to be eight edges. In addition, one of the assumptions in this part was that the dimensions of the robot are not taken in consideration, i.e., the robot is considered as material point. But in the next part the robot will be considered as an object which has length, width and height where the navigation will be taken in 3-dimension environment.

References

[1] O. Hachour AND N. Mastorakis Behaviour of intelligent autonomous ROBOTIC IAR", *IASME transaction*, issue1, vol 1, ISSN 1790-031x WSEAS, January 2004,pp 76-86.

[2] Ihn Namgung and Joseph Duffy, "Two dimensional collision-free path planning using linear parametric curve", *Journal of Robotic Systems*, 1998.

[3] S.Florczyk, *Robot Vision Video-based Indoor Exploration with Autonomous and Mobile Robots*, WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim, 2005.

[4] O. Hachour and N. Mastorakis, IAV : A VHDL methodology for FPGA implementation, *WSEAS transaction on circuits and systems,* Issue5, Vol 3,ISSN 1109-2734, 2004,pp.1091-1096.

[5] D. F. Rogers and J. A. Adams, « Mathematical Elements for Computer Graphics », 2nd ed., *McGraw-Hill*, New York, 1990.

[6] Bezier, P., 1972, "Numerical Control-Mathematics and Application, A. R. Forrest and A. F. Pankhurst, Trans., John Wiley & Sons, New York, 1972.

[7] V. J. Lumelsky, "Effect of kinematics on motion planning for planar robots arms moving amidst unknown obstacles", *IEEE J. of Robotics and Automation*, RA-3(3), 1987,pp207-223,.

[8] I. D. Faux et M. J. Pratt, *« Computational Geometry for Design and Manufacture"*, Jhon Wiley & Sons, New York, 1979.

[9] S. M. LaValle, "*Planning Algorithms*" , Published by Cambridge University Press, 2006.

[10] O.Hachour, "The Proposed Genetic FPGA Implementation For Path Planning of Autonomous Mobile Robot", *International Journal of Circuits , Systems and Signal Processing,* Issue 2, vol2 ,2008,pp151-167.

[11] O. Hachour AND N. Mastorakis, Avoiding obstacles using FPGA –a new solution and application ,*5th WSEAS international conference on automation & information (ICAI 2004) , WSEAS transaction on systems* , issue9 ,vol 3 , Venice , Italy , , ISSN 1109-2777, November 2004, pp2827-2834.

[12] O. Hachour AND N. Mastorakis, "Behaviour of intelligent autonomous ROBOTIC IAR", *IASME transaction*, issue1, vol.1, ISSN 1790-031x WSEAS, January 2004,pp 76-86.

[13] O. Hachour, ,"path planning of Autonomous Mobile Robot", *International Journal of Systems Applications, Engineering & Development*, Issue4, vol.2, 2008, pp178-190.