ON-LINE PATH PLANNING FOR MOBILE ROBOTS IN DYNAMIC ENVIRONMENTS

*P. Raja, S. Pugazhenthi**

Abstract: Motion planning of mobile robots is a complex problem. The complexity further increases when it comes to path planning in dynamic environments. This paper presents an algorithm for on-line path planning of mobile robots in unknown environments with moving obstacles. A mathematical model is established which considers all the current on-line information of robot as well as nearing obstacles. Particle Swarm Optimization technique is used to optimize the velocity parameters of the robot, to arrive at the shortest collision-free trajectory, satisfying dynamic constraints. Simulation results show that the proposed algorithm is computationally efficient and effective.

Key words: *Mobile robot, on-line path planning, dynamic constraints and particle swarm optimization.*

Received: March 15, 2011
Revised and accepted: February 20, 2012

1. Introduction

Obstacle avoidance is a basic requirement in path planning of mobile robots [1]. Path planning is to determine a collision-free path from a starting point to a goal point, optimizing a performance criterion such as distance, time or energy. Based on the availability of information about environment, there are two types of path planners, namely off-line and on-line. In off-line path planning, complete information about trajectory of obstacles is known in advance. Grid [2-4], meadow map [5], Voronoi diagram [6], visibility graph [7] etc., are some of the techniques used for off-line path planning. For environments where obstacle motions cannot be predicted in advance, a different approach is needed.

In recent times, on-line path planning has received more attention from researchers [8]. For on-line path planning, complete information about environment is not available to robot in advance. The robot gets information through sensors, as

*P. Raja, S. Pugazhenthi
School of Mechanical Engineering, SASTRA University, Thanjavur – 613 401, Tamilnadu, India,
E-mail: raja@mech.sastra.edu, pugazhenthi@mech.sastra.edu

it moves through the environment. Artificial potential field concept [9] proposed by Khatib is a popular on-line path planning approach. However, this classic approach suffers from the drawback of robot getting trapped. Also, this approach deals with limited information of the local scene, referring only to the distance between robot and obstacles, ignoring their relative velocities.

Another widely used on-line path planning approach is based on collision cone concept [10, 11]. Collision of robot can be avoided if the relative velocity of the robot with respect to an obstacle falls outside the collision cone. Fiorini and Shiller [12] presented a velocity obstacle approach, which is similar to the collision cone approach. It consists of selecting avoidance maneuvers to avoid static and moving obstacles in the velocity space by using the basic heuristic strategy designed for prioritized objectives such as avoiding collisions, reaching the goal, maximizing speed or achieving trajectories with desired topologies. However, finding the collision-free velocity is fully dependent on the basic heuristic strategy without any mathematical modeling of environment.

Today, there are many advanced evolutionary techniques such as Genetic Algorithm, Ant Colony Optimization and Particle Swarm Optimization (PSO). PSO is an evolutionary computation technique inspired by social behavior of bird flocking or fish schooling. Compared to other evolutionary techniques, PSO is easier to implement, and there are fewer parameters to be adjusted [13].

Lu and Gong [14] proposed an on-line path planning algorithm using PSO technique for unknown environments. Their algorithm is entirely based on distance information of the environment without any mathematical model featuring velocity of nearing obstacle. Min et al. [15] used a mathematical model through collision cone approach and PSO technique for on-line path planning. In order to reduce computational burden, they neglected instantaneous changes in obstacle velocities in the motion model. Therefore, their algorithm is applicable to sparse environment containing obstacles with slower velocities. Further, PSO, in combination with binary coded genetic operators such as crossover and mutation, is used as a tool to find optimum collision-free direction without satisfying dynamic constraints. However, recent studies show that real coded evolutionary algorithms such as PSO perform better than the partially binary coded [4].

This paper presents an on-line path planning algorithm for mobile robots to plan a sequence of smooth collision-free motions. The algorithm employs a mathematical model based on information such as position and velocity of robot as well as obstacles. At every instant, velocity of robot is optimized using PSO by considering the shortest Euclidean distance to target and maximum achievable velocity of robot satisfying both kinematic and dynamic constraints.

The remaining part of the paper is organized as follows: Section 2 describes the proposed algorithm. Section 3 illustrates the functioning of the algorithm through simulation results. Section 4 presents the comparative performance of the algorithm, and conclusions are given in Section 5.

2. Proposed Algorithm

This section explains the internal description of the algorithm such as modeling of environment, establishment of mathematical model, dynamic constraints of robot,

collision-avoidance for multiple obstacles and implementation of PSO to determine the optimal velocity of robot at every instant.

2.1 Modeling of dynamic environment

Robot is assumed to be a point which can move at a prescribed velocity. The obstacles may be of any shape, static or dynamic, which can move with arbitrary velocities with or without rotation about their own axes. Obstacles are modeled by enclosing circles [10-12, 15-19], amplified by a value so as to take care of physical dimensions of the robot and a safe maneuverable distance from the obstacles without collision. Approximating irregularly shaped obstacles to circles is not a severe limitation since general polygons can be represented by a collection of circles [10, 20, 21]. It is assumed that robot travels towards its target and gets information of nearing obstacles' instantaneous position and velocity on-line, by refreshing sensory data.

2.2 Mathematical model for collision-avoidance

Fig. 1 shows the motion model for collision-avoidance, in which the point R represents robot and the point O represents the centre of the circle enclosing a moving obstacle. A coordinate frame XY is attached at R with its X axis along the line joining the current position of the robot R and the current position of the target T. Let α be the angle made by the velocity of robot V_R with respect to the X axis and β be the angle made by the velocity of obstacle V_O with respect to the X axis of the same frame. Relative velocity between the robot and the obstacle V_{OR} makes an angle γ with the line joining R and O. Two extreme tangents (RP and RQ respectively) are drawn from the robot to the obstacle forming a collision cone with μ as semi collision cone angle. For the mobile robot to avoid collision with obstacle, the relative velocity V_{OR} should lie outside the collision cone, i.e.,

$$\mu \leq \gamma \leq 2\pi - \mu, \tag{1}$$

where

$$\gamma = \tan^{-1}\left(\frac{V_{ORn}}{V_{ORa}}\right) = \tan^{-1}\left(\frac{V_R \sin(\alpha-\theta) - V_O \sin(\beta-\theta)}{V_R \cos(\alpha-\theta) - V_O \cos(\beta-\theta)}\right). \tag{2}$$

V_{ORa} and V_{ORn} are the relative velocities along and normal to the line joining R and O respectively while θ being the angle between X axis and the line joining R and O.

For tan function as in (2), the first derivative can be written as:

$$\frac{d\gamma}{df} = \frac{1}{1+f^2} \text{ where } f = \left(\frac{V_R \sin(\alpha-\theta) - V_O \sin(\beta-\theta)}{V_R \cos(\alpha-\theta) - V_O \cos(\beta-\theta)}\right). \tag{3}$$

Simplifying $\frac{1}{1+f^2}$ yields:

$$\frac{1}{1+f^2} = \frac{k^2}{V_R^2 + V_O^2 - 2V_R V_O \cos\left[(\alpha-\theta)-(\beta-\theta)\right]} \text{ where}$$
$$k^2 = \left[V_R \cos(\alpha-\theta) - V_O \cos(\beta-\theta)\right]^2. \tag{4}$$

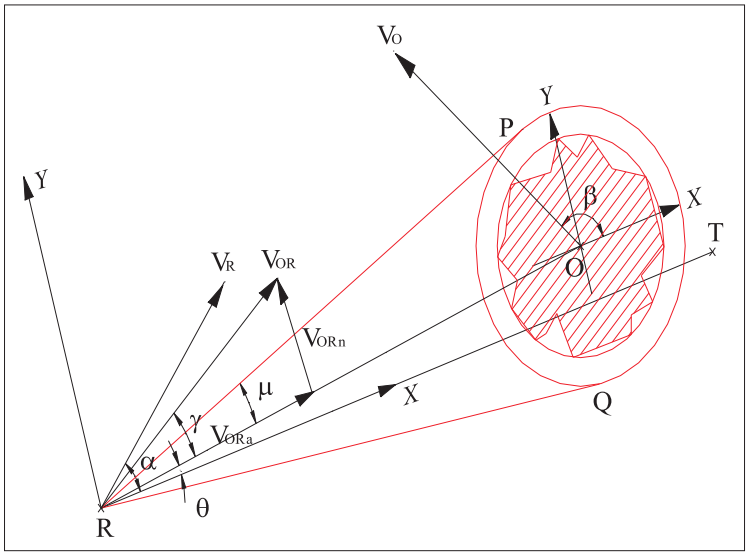


Fig. 1 *Model for collision-avoidance.*

Equation (3) can also be written as:

$$d\gamma = \left[\frac{1}{1+f^2}\right] df. \tag{5}$$

We know that γ is a function of V_R and V_O, i.e., $\gamma = tan^{-1} f\left(V_R, \alpha, V_O, \beta\right)$. By total differentiation, df can be written as:

$$df = \frac{\partial f}{\partial V_R} dV_R + \frac{\partial f}{\partial \alpha} d\alpha + \frac{\partial f}{\partial V_O} dV_O + \frac{\partial f}{\partial \beta} d\beta, \tag{6}$$

where

$$\frac{\partial f}{\partial V_R} dV_R = \frac{V_O \sin\left(\beta - \alpha\right)}{k^2} dV_R \tag{7}$$

$$\frac{\partial f}{\partial \alpha} d\alpha = \frac{V_R \left[V_R - V_O \cos\left(\alpha - \beta\right)\right]}{k^2} d\alpha \tag{8}$$

$$\frac{\partial f}{\partial V_O} dV_O = \frac{V_R \sin\left(\alpha - \beta\right)}{k^2} dV_O \tag{9}$$

$$\frac{\partial f}{\partial \beta} d\beta = \frac{V_O^2 - V_R V_O \cos\left(\alpha - \beta\right)}{k^2} d\beta. \tag{10}$$

Substituting (7-10) in (6), we get:

$$df = \frac{V_O \sin\left(\beta - \alpha\right) dV_R + V_R \left[V_R - V_O \cos\left(\alpha - \beta\right)\right] d\alpha}{k^2} + \\ + \frac{V_R \sin(\alpha - \beta) dV_O + V_O \left[V_O - V_R \cos\left(\alpha - \beta\right)\right] d\beta}{k^2} \tag{11}$$

Substituting (4) and (11), in (5) and further simplifying:

$$\Delta\gamma = \frac{V_O \sin(\beta-\alpha)\,\Delta V_R + V_R\,[V_R - V_O\cos(\alpha-\beta)]\,\Delta\alpha}{V_R^2 + V_O^2 - 2V_R V_O\cos(\alpha-\beta)} + \\ + \frac{V_R \sin(\alpha-\beta)\,\Delta V_O + V_O\,[V_O - V_R\cos(\alpha-\beta)]\,\Delta\beta}{V_R^2 + V_O^2 - 2V_R V_O\cos(\alpha-\beta)} \tag{12}$$

In order to reduce the complexity of the above expression, the relationship between V_R and V_O is modeled in Fig. 2.

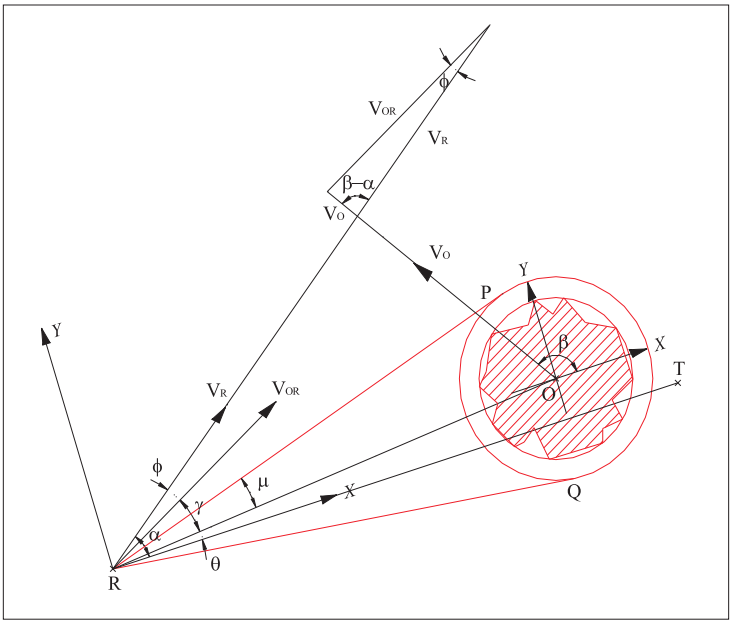


Fig. 2 *Relationship between V_R and V_O.*

From Fig. 2, using sine theorem,

$$V_O \sin(\beta-\alpha) = V_{OR}\sin\varphi, \tag{13}$$

$$V_R \sin(\alpha-\beta) = -V_{OR}\sin[180-(\beta-\alpha+\varphi)] \tag{14}$$

where φ is the angle between V_R and V_{OR}.

Similarly, through geometrical relations:

$$V_R - V_O\cos(\alpha-\beta) = -V_{OR}\cos\varphi \tag{15}$$

$$V_O - V_R\cos(\alpha-\beta) = V_{OR}\sin\left[\varphi - \frac{\pi}{2} - (\beta-\alpha)\right] \tag{16}$$

$$V_R^2 + V_O^2 - 2V_R V_O\cos(\alpha-\beta) = V_{OR}^2. \tag{17}$$

Substituting (13-17) in (12):

$$\Delta\gamma = \frac{1}{V_{OR}}\left[\sin\varphi\Delta V_R - V_R\cos\varphi\Delta\alpha - \sin\left[180-(\beta-\alpha+\varphi)\right]\Delta V_O + \right.$$
$$\left. + V_O\left(\sin\left[\varphi-\frac{\pi}{2}-(\beta-\alpha)\right]\right)\Delta\beta\right]. \qquad (18)$$

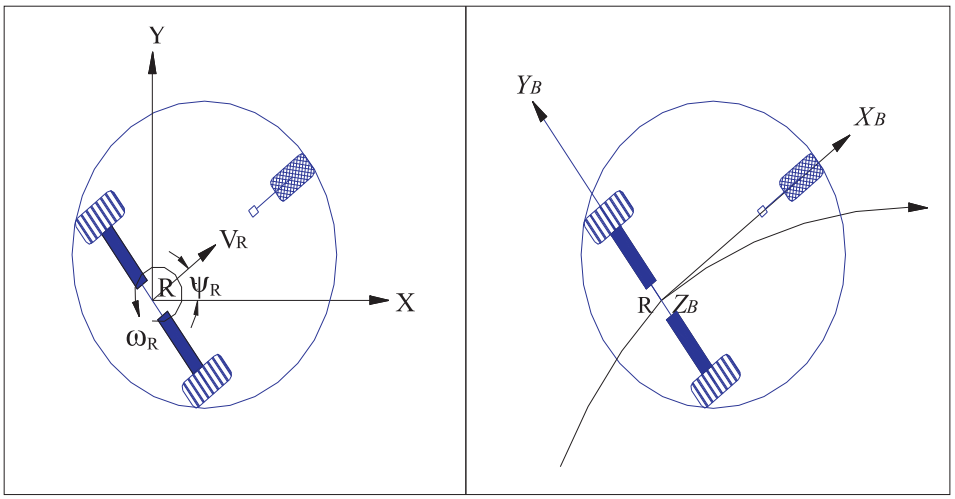
The above expression establishes $\Delta\gamma$ as a function of velocity parameters of robot as well as obstacle, namely (V_R, α) and (V_O, β). Among these, only the parameters of robot $(\Delta V_R, \Delta\alpha)$ can be adjusted. Obstacle velocity parameters $(\Delta V_O, \Delta\beta)$ cannot be adjusted but can be measured using sensors. Therefore, for the robot to move from its current location to next location, $\gamma_{new} = \gamma_{current} + \Delta\gamma$ subject to satisfying (1). However, there can be numerous combinations of (V_R, α) which result in V_{OR} lying outside the collision zone.

2.3 Dynamic constraints of robot

In order to arrive at the governing equations of velocity and acceleration bounds, a differential drive nonholonomic mobile robot is considered, as shown in Fig. 3(a). The two rear wheels are driven by two independent actuators and the front one is a self-aligning wheel. Let V_R and ω_R be the linear and angular velocities of the center point R of the rear axle. Let ψ_R be the heading angle of the robot with respect to the global frame XY. The posture of the robot is defined by the triplet $(X_R, Y_R, \psi_R)^T$. The velocity components of robot [22-24] are given by

$$X_R' = V_R\cos\psi_R, \quad Y_R' = V_R\sin\psi_R, \psi_R' = \omega_R, \qquad (19)$$

where prime (superscript) refers to derivative with respect to time t.



(a) Nonholonomic differential drive robot; (b) Motion along a curved path.

Fig. 3 *Model for kinematic and dynamic constraints.*

Fig. 3(b) shows motion of the robot along a curved path. For a plane curve expressed parametrically in terms of $X(t)$and $Y(t)$, the signed curvature (κ) is given by

$$\kappa = \frac{X'_R Y''_R - X''_R Y'_R}{(X'^2 + Y'^2)^{\frac{3}{2}}}. \tag{20}$$

The translation and rotation velocities ($V_R \& \omega_R$) are related to κ as $\kappa = \frac{\omega_R}{V_R}$. A local coordinate frame is attached to the robot body with X_B axis coinciding with the vector tangent to the curved path at R, as shown in Fig. 3(b). Let F_{XB}, F_{YB} and F_{ZB} be the forces acting along X_B, Y_B and Z_B directions at R. The motion of the robot along the path must obey Newtonian dynamics, i.e.

$$F_{XB} = ma_R, \quad F_{YB} = m\frac{V_R^2}{r} = m\kappa V_R^2, \quad F_{ZB} = mg, \tag{21}$$

where m is the mass of the robot, a_R is the acceleration of the robot, r is the radius of path followed by the robot and g is the acceleration due to gravity. For the robot to avoid sliding,

$$\sqrt{F_{XB}^2 + F_{YB}^2} \leq \mu_f F_{ZB}, \tag{22}$$

where μ_f is the coefficient of friction between the wheels and the floor.

The bound on the admissible acceleration a_R, is obtained from (21) and (22). Also considering the maximum torque or force F_{max} applied on the wheels, the admissible acceleration [25] is written as

$$a_R \leq \min\left(\sqrt{\mu_f^2 g^2 - \kappa^2 V_R^4}, \frac{F_{\max}}{m}\right). \tag{23}$$

The bound on the admissible velocity V_R, is obtained from the requirement that $\sqrt{\mu_f^2 g^2 - \kappa^2 V_R^4}$ in (23) should be non-negative. Hence,

$$V_R \leq \min\left(\sqrt{\frac{\mu_f g}{\kappa}}, V_{R\max}\right). \tag{24}$$

The bound on the angular acceleration and angular velocity are constrained by actuator specifications as given below:

$$\alpha_R \leq \alpha_{R\max} \text{ and } \omega_R \leq \omega_{R\max}. \tag{25}$$

2.4 Collision-avoidance for multiple obstacles

Mathematical model for collision-avoidance of a single obstacle was discussed in Section 2.2. In case of environment with multiple obstacles, they can be negotiated according to priority such that the most imminent collision is avoided first. At the same time, it is also necessary to consider other obstacles which may necessitate the robot to deviate further away.

2.4.1 Identification of the most imminent obstacle

The most imminent obstacle is the one which has the smallest collision distance index [17], as given below:

$$\delta = \frac{d_{R,O_J}}{V_J T_S}, \quad J = 1, 2, \ldots, N_O, \tag{26}$$

where d_{R,O_J} is the Euclidean distance between the robot R and the J^{th} obstacle, V_J is the velocity of J^{th} obstacle and T_S is sampling rate or refreshment time.

2.4.2 Effect of constraining obstacles

In an environment with multiple obstacles it is not enough to negotiate the most imminent obstacle alone, as it may not be the most constraining obstacle. If a small or slow moving obstacle is close to the robot and a much larger one is further away, it may be that the robot has to deviate more for the farther obstacle. Therefore, it is necessary to identify constraining obstacles and negotiate the most imminent one taking into account further deviation required, if any. An obstacle may be considered to be constraining obstacle if its collision cone overlaps with that of the most imminent obstacle. One such case is presented in Fig. 4. The center of the most imminent obstacle is represented by O_1 and the constraining obstacle by O_2. There may also be more than one constraining obstacle in a given situation. For the mobile robot to avoid collision with the most imminent obstacle as well as to take required deviation considering constraining obstacles, the relative velocity $V_{O_1 R}$ should lie outside the extreme tangents of all collision cones which overlap

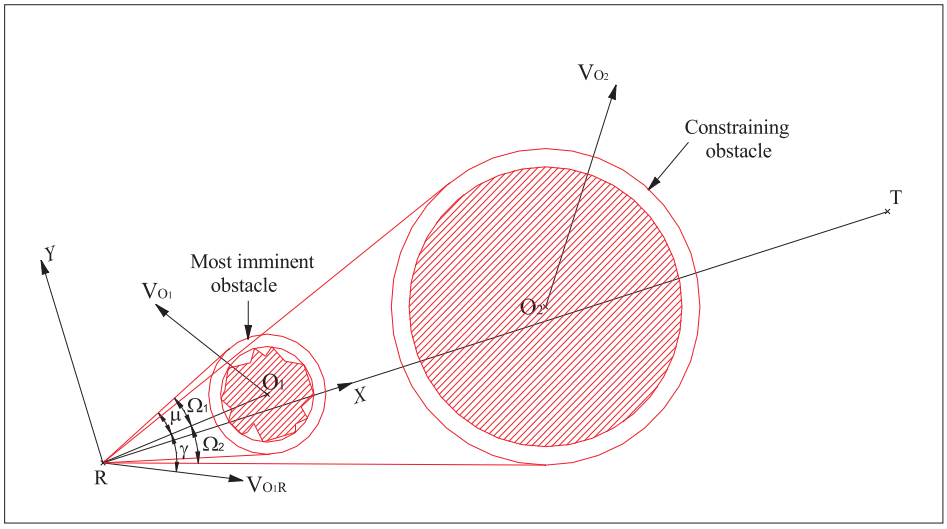


Fig. 4 *Negotiation of the most imminent obstacle considering the effect of constraining obstacle.*

with that of the most imminent one. Hence, (1) can be modified as:

$$\max(\mu, \Omega_1) \leq \gamma \leq \min(2\pi - \mu, 2\pi - \Omega_2), \tag{27}$$

where Ω_1 and Ω_2 are the angles made by the extreme tangents of the overlapping collision cones with respect to the line joining robot R and the center of the most imminent obstacle O_1.

2.5 Working of the proposed on-line algorithm

For the mobile robot to move towards its target, at every instant the algorithm receives information about its environment by refreshing sensory data. Then the algorithm checks for the existence of any nearing obstacle. If there is no obstacle, the robot travels along the straight line towards the target satisfying kinematic and dynamic constraints. In the event of any obstacle being present, the algorithm checks whether the relative velocity between the robot and the obstacle V_{OR} lies within the collision cone. In case of multiple obstacles, collision distance index δ is computed to identify the most imminent threat. Also, the algorithm checks for the influence of constraining obstacles, as discussed in Section 2.4.2. The overlapping collision cones of obstacles result in a collision zone. In order to steer the robot out of the collision zone, the algorithm generates numerous collision-free combinations of V_R and α randomly, however satisfying the dynamic constraints. PSO technique is used to quickly select the best combination to have the shortest collision-free trajectory of the robot. This is demonstrated by means of a flowchart, as shown in Fig. 5.

2.5.1 Implementation of particle swarm optimization

The velocity of the robot at any instant can be within the range between 0 m/s and the maximum (specified). Incremental velocity ΔV_R at the next instant can be such that the velocity and acceleration constraints V_R, a_R, ω_R and α_R as in (23-25) are not violated. Similarly, the angle α, made by V_R with the X axis, as shown in Fig. 1, can lie between 0^o and 360^o. Different combinations of V_R and α are obtained by random pairing. For the first iteration, the combinations are randomly chosen, and for every combination $\Delta\gamma$ is determined as per (18). Then, $\gamma_{new} = \gamma_{current} + \Delta\gamma$ is verified for satisfying (27). Such combinations of V_R and α that result in γ_{new} satisfying (23-25) & (27) alone are considered as valid particles. Fifty such valid particles constitute the first generation. Out of these fifty particles, one that has the least Euclidean distance between the new position of the robot and the target and having the maximum possible velocity of the robot is identified as 'pbest'. The same is repeated for 100 iterations.

For the subsequent iterations, particles are updated by using the following governing equations:

$$v[\,] = c_1(rand\,())(pbest[\,] - present[\,]) + c_2(rand\,())(gbest[\,] - present[\,]) \tag{28}$$

$$present[\,] = present[\,] + v[\,] \tag{29}$$

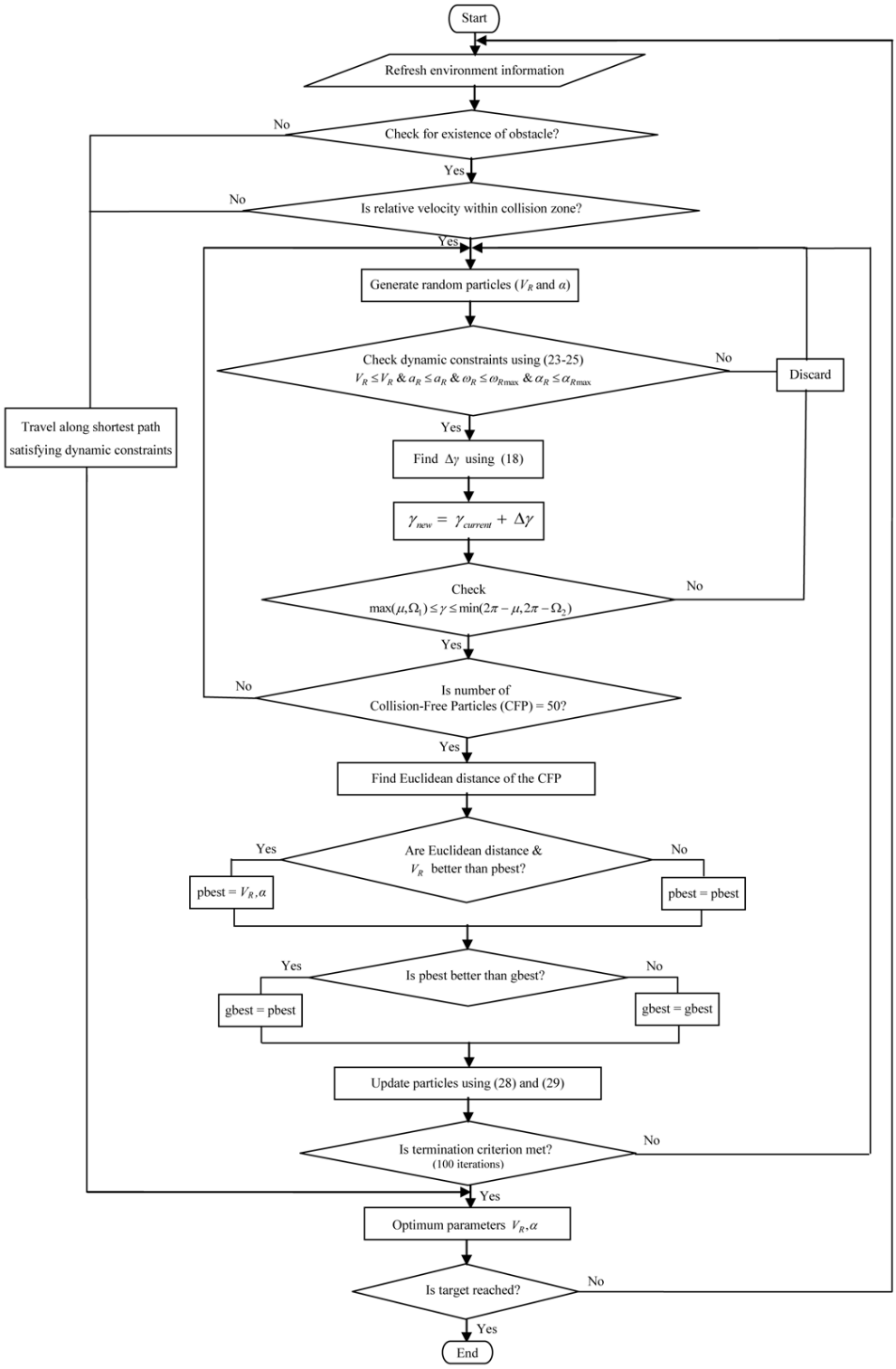


Fig. 5 *Flowchart of the PSO implementation.*

where $v\,[\,]$ is particle velocity, $present\,[\,]$ is current particle (solution), $rand\,()$ is random number between 0 and 1, 'gbest[]' is the best particle identified by comparing 'pbest[]' of every iteration and c_1 & c_2 are learning factors. For quick convergence c_1 and c_2 are tuned to be 2 based on sample trials. Thus, at the end of 100 iterations the algorithm offers an optimal set of V_R and α which has the shortest Euclidean distance to target and the maximum possible V_R. Until the robot reaches its target, sensory data are refreshed at the prescribed sampling rate. The last maneuvered point is taken as the new start point and using the algorithm described earlier, the incremental motion of robot is planned. The procedure is repeated till the robot reaches its target.

3. Simulation Results

In order to study the performance of the proposed algorithm, simulation is performed in MATLAB 7.0 on an Intel(R) Core(TM) 2 to 2.4 GHz processor. Velocity and acceleration constraints of mobile robot are v_{Rmax}=0.7 m/s, a_{Rmax}=0.5 m/s^2, ω_{Rmax}=2 rad/s and α_{Rmax}=2 rad/s^2. Other parameters used in the simulation are the mass of the robot including its payload, m=10 kg and the coefficient of friction between the wheels of the robot and the floor, $\mu_f = 0.3$. The obstacle detection range of the sensor is assumed to be 150 m. The sampling time, T_S is 50 ms.

3.1 Illustration of collision-avoidance

In order to illustrate the functioning of the algorithm, an environment consisting of two static obstacles and a moving obstacle is considered, as shown in Fig. 6. Static obstacles are represented by dark shading and moving obstacles by hatching. Even though the mobile robot is modeled as a point and the obstacles are enlarged, for showing the trajectory of the robot it is represented by small circles. R and T are the start and target locations of the robot and O is the start location of

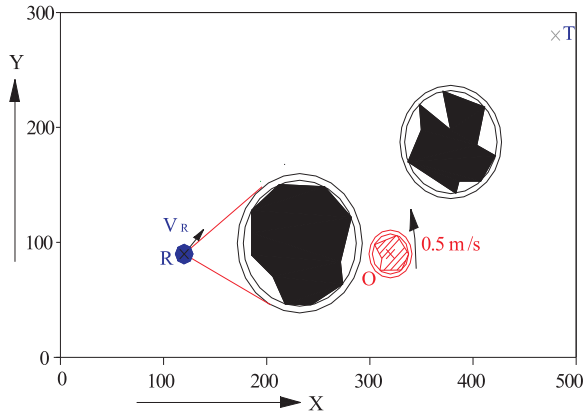
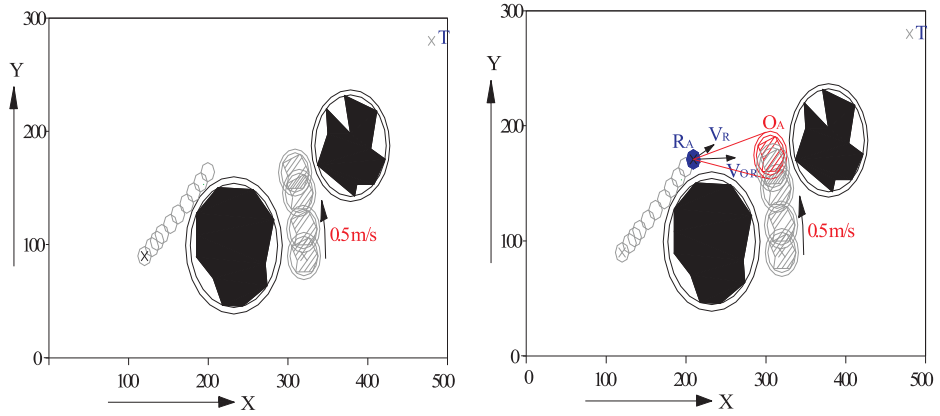


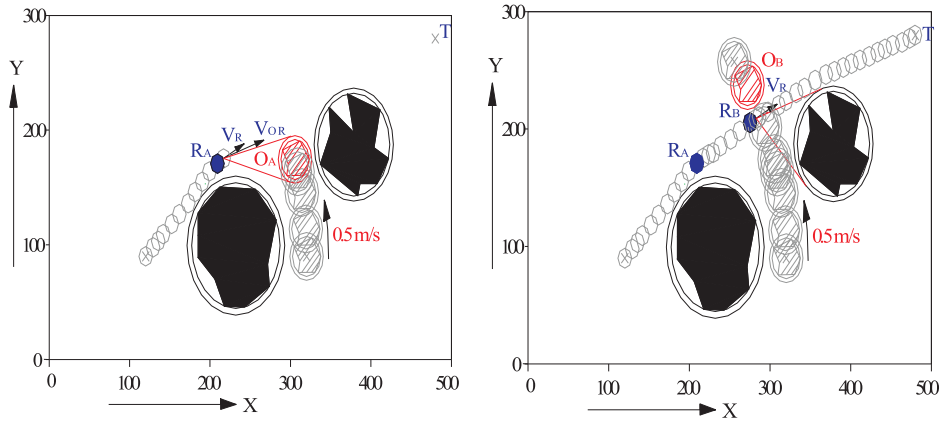
Fig. 6 *Illustrative environment.*

the moving obstacle. The path of the moving obstacle is assumed to be along a circular path. The velocity of the obstacle is considered to be 0.5 m/s, which is maintained during cruise and at the start and end, the velocity is governed by the acceleration/deceleration of 0.3 m/s^2. Fig. 7 explains different situations at different instances along the trajectory of the robot from its start to the target.

Fig. 7(a) demonstrates the negotiation of the first static obstacle and the robot moves tangential to the obstacle towards its target. The negotiation of moving obstacle is demonstrated in Fig. 7(b) and 7(c). At the instant 'A', the algorithm predicts possible collision as the relative velocity V_{OR} lies within collision cone, as shown in Fig. 7(b). The algorithm steers the robot keeping V_{OR} outside the collision cone, as shown in Fig. 7(c). The robot is driven with a slower velocity till the instant 'B' by which time the obstacle has crossed the path of the robot.



(a) Negotiation of static obstacle; (b) Prediction of collision with a moving obstacle



(c) Steering out of the collision cone; (d) Reaching the target

Fig. 7 *Illustration of collision-avoidance.*

Subsequently, the second static obstacle is negotiated to reach the target, as shown in Fig. 7(d).

In order to illustrate the effectiveness of the algorithm in negotiating multiple moving obstacles, an environment having three moving obstacles O_1, O_2 and O_3 with velocities of 0.3 m/s, 1.2 m/s and 0.1 m/s respectively is considered, as shown in Fig. 8. Obstacles O_1 and O_2 pose the threat of collision as their relative velocities V_{O_1R} and V_{O_2R} lie within their collision cones respectively (Fig. 8(a)). The collision distance indices of the obstacles O_1 and O_2 are computed as $\delta_1 = 6$ and $\delta_2 = 1.66$. Therefore, the algorithm identifies O_2 as the most imminent threat. Further, the algorithm also identifies O_3 as a constraining obstacle as its collision cone overlaps with that of O_2. The algorithm negotiates obstacle O_2 first steering V_{O_2R} not just outside its collision cone but outside the extreme tangents of overlapping collision cones of O_2 and O_3, ahead of O_1, as shown in Fig. 8(b). Fig. 8(c) shows the negotiation of obstacle O_1 by steering V_{O_1R} outside its collision cone.

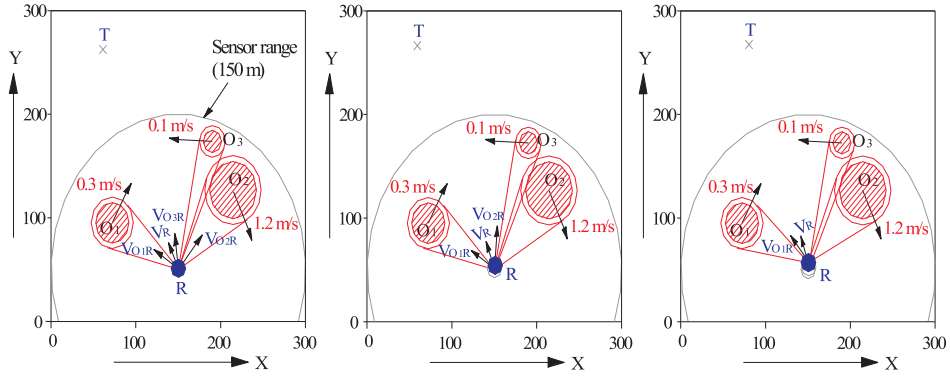


Fig. 8 *Negotiation of multiple moving obstacles.*

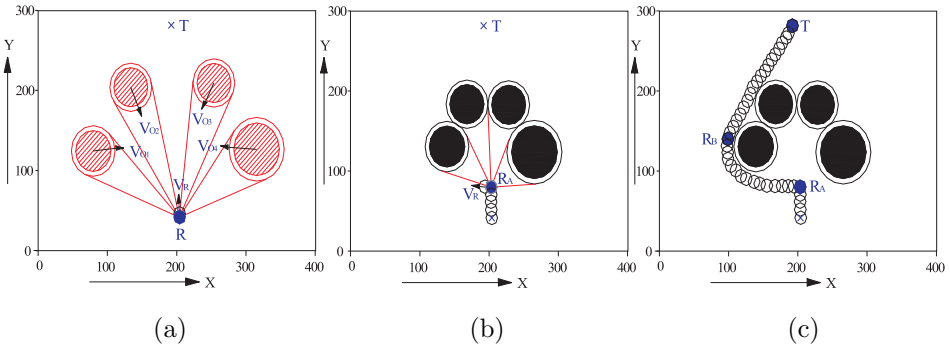


Fig. 9 *Trajectory of robot escaping from U-trap situation.*

3.2 Illustration of trap avoidance

Fig. 9 presents a trap situation where multiple moving obstacles form a trap by hindering the robot's path to reach its target. Fig. 9(a) shows the situation where the obstacles are moving and robot is directed towards its target. Robot is trapped at the instant 'A' as the moving obstacles become stagnant and form a U-shaped trap situation, as shown in Fig. 9(b). At this instant, the algorithm generates a path tangential to the outermost collision cone. After following the contour till the instant 'B', the algorithm drives the robot towards its target, as shown in Fig. 9(c). Thus, by the inherent nature of the proposed algorithm, it does not require any special trap recovery procedure like the ones employed in [26-28].

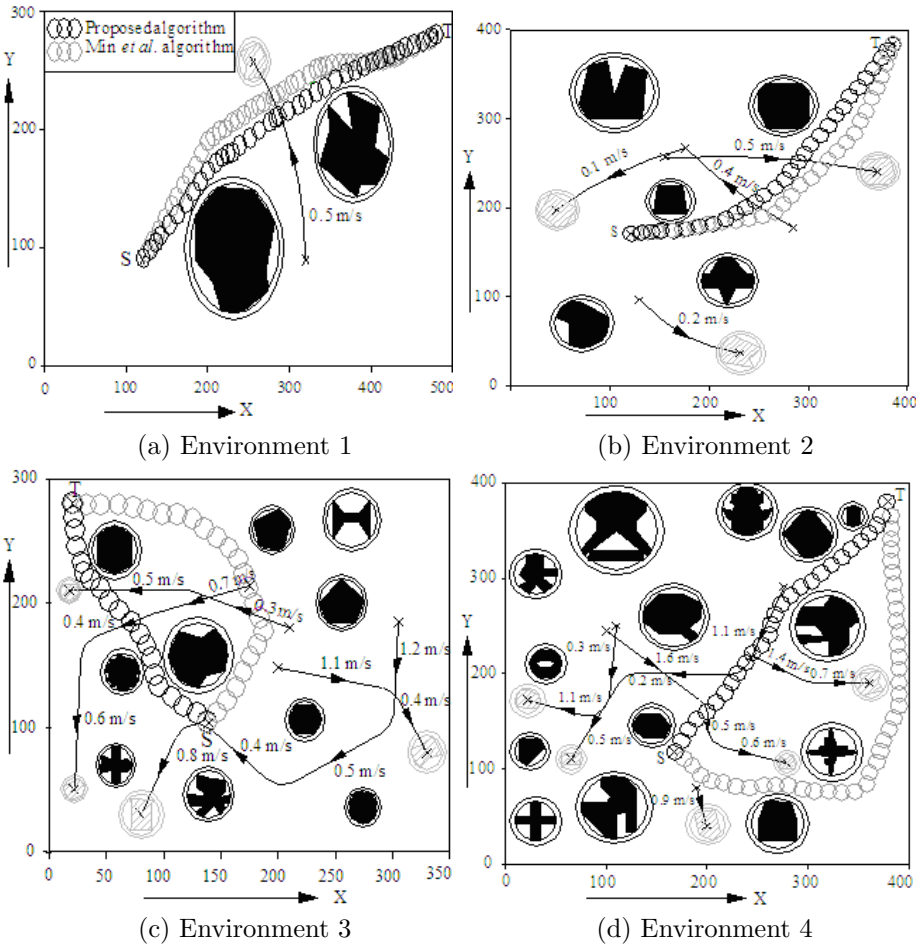
4. Comparative Performance of the Algorithm

The performance of the proposed algorithm is studied using four different simulated environments by comparing with that of Min et al. [15]. Fig. 10 shows the four simulated dynamic environments. The first two environments are sparse and the other two are cluttered environments. Lines with arrowheads show the path of moving obstacles. The corresponding obstacle velocities are written alongside. Simple curvilinear paths for the moving obstacles are designed for the first two sparse environments, as shown in Fig. 10(a) and 10(b). More cluttered environments are presented in Fig. 10(c) and 10(d), with obstacles having different velocities when they move through a series of linear and curvilinear path segments. The optimal path generated by the proposed algorithm as well as that of Min et al. are shown in the figures where S and T are the start and target points of the mobile robot.

Environment	Path length (m)		Improvement in %	Computation time for each instant of the robot (ms)		Improvement in %
	Proposed on-line	Min et al. [15] on-line		Proposed on-line	Min et al. [15] on-line	
1	413	431	4.18	12	52	76.92
2	358	388	7.73	16	64	75
3	229	316	27.53	24	89	73.03
4	340	505	32.67	33	119	72.26

Tab. I *Comparison of results with Min et al. algorithm.*

Tab. I compares the results of the proposed algorithm for the above dynamic environments with that of Min et al. algorithm. The data presented are of the result of an average of 100 trials. The algorithm achieves a minimum of 4.18% improvement in path length for sparse dynamic environments while a maximum of 32.67% improvement in path length for the cluttered environments over Min et



(a) Environment 1 (b) Environment 2

(c) Environment 3 (d) Environment 4

Fig. 10 *Comparison of path with Min et al. algorithm.*

al. algorithm. This improvement can be attributed to the use of a mathematical model, which considers the complete environment information such as velocity parameters of both robot and nearing obstacles, whereas the model used by Min et al. neglected the instantaneous changes in obstacle velocity and robot dynamics. Further, due to real encoding and elimination of invalid particles from population [29-31], an improvement of over 70% in computation time per instant is achieved.

5. Conclusions

An effective on-line path planning algorithm for mobile robots in dynamic environments has been developed. A mathematical model is established which considers all the current on-line information of robot as well as nearing obstacles. The proposed algorithm combines a mathematical model for collision-avoidance and evolutionary

PSO technique to plan an optimal collision-free path satisfying both kinematic and dynamic constraints of robot. The effect of constraining obstacles is also taken into consideration while negotiating the most imminent obstacle. The algorithm does not require any separate recovery mode approach to escape from trap situations. Simulation results show that the proposed algorithm is computationally efficient and effective in reaching the target along the shortest possible path. The algorithm is also applicable to environments having moving targets. This work can also further be extended to multi-robot applications.

References

[1] Latombe J. C.: Robot motion planning, Boston, Kluwer Academic Publisher, 1991.

[2] Payton D. W., Rosenblatt J. K., Keirsey D. M.: Grid-based mapping for autonomous mobile robot, Robotics and Autonomous Systems, **11**, 1, 1993, pp. 13-21.

[3] Likhachev M., Ferguson D., Gordon G., Stentz A., Thrun S.: Anytime Dynamic A*: An Anytime, Replanning Algorithm, Proc. 2005 Int. Conf. on Automated Planning and Scheduling, 2005, pp. 262-271.

[4] Raja P., Pugazhenthi S.: Path planning for a mobile robot using real coded genetic algorithm, International Journal of Assistive Robotics and Systems, **10**, 1, 2009, pp. 27-39.

[5] Arkin R. C.: Navigational path planning for a vision-based mobile robot, Robotica, 7, 1989, pp. 49-63.

[6] Dunlaing C. O., Yap C. K.: A retraction method for planning the motion of a disc, Journal of Algorithms, 6, 1982, pp. 104-111.

[7] Mitchell H.: An algorithmic approach to some problems in terrain navigation, Artificial Intelligence, 37, 1988, pp. 171-201.

[8] Masehian E., Katebi Y.: Robot motion planning in dynamic environments with moving obstacles and target, International Journal of Mechanical Systems Science and Engineering, 2007, pp. 20-25.

[9] Khatib O.: Real-time obstacle avoidance for manipulators and mobile robots, International Journal of Robotics Research, **5**, 1, 1986, pp. 90-98.

[10] Chakravarthy A., Ghose D.: Obstacle avoidance in a dynamic environment: a collision cone approach, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, **28**, 5, 1998, pp. 562-574.

[11] Qu Z., Wang J., Plaisted C. E.: A new analytical solution to mobile robot trajectory generation in the presence of moving obstacles, IEEE Transactions on Robotics, **20**, 6, 2004, pp. 978-993.

[12] Fiorini P., Shiller Z.: Motion planning in dynamic environments using velocity obstacles, International Journal of Robotics Research, **17**, 7, 1998, pp. 760-772.

[13] Kennedy J., Eberhart R. C.: Particle swarm optimization, Proc. IEEE Int. Conf. on Neural Networks, Perth, Australia, 1995, pp. 1942-1948.

[14] Lu L., Gong D.: Robot path planning in unknown environments using particle swarm optimization, Proc. 4th Int. Conf. on Natural Computation, Jinan, October 2008, pp. 422-426.

[15] Min H. Q., Zhu J. H., Zheng X. J.: Obstacle Avoidance with Multi-Objective Optimization by PSO in Dynamic Environment, Proc. IEEE Int. Conf. on Machine Learning and Cybernetics, Guangzhou, August 2005, pp. 2950-2956.

[16] Yu Z., Meng G., Liu H., Deng X., Liu J., Wu Q., Liu Y.: Dynamic obstacle avoidance in polar coordinates for mobile robot based on laser radar, Proc. 2008 IEEE Pacific Asia workshop on computational intelligence and industrial applications, Wuhan, China, PACIIA 2008, pp. 334-338.

[17] Luh G. C., Liu W. W.: Motion planning for mobile robots in dynamic environments using a potential field immune network, IMechE Journal of Systems and Control Engineering, 221, 2007, pp. 1033-1046.

[18] Hui-zhong Z., Shu-xin D., Tie-jun W.: On-line real-time path planning of mobile robots in dynamic uncertain environment, Journal of Zhejiang University SCIENCE A, **7**, 4, 2006, pp. 516-524.

[19] Qingquan W., Bi Z.: Real-time globally optimized path planning in a dynamic environment, Proc. Second Int. Conf. on Intelligent Computation Technology and Automation, Hunan, China, 2009, pp. 261-264.

[20] O'Rourke J., Badler N.: Decomposition of three-dimensional objects into spheres, IEEE Transactions on Pattern Analysis and Machine Intelligence, **1**, 3, 1979, pp. 295-305.

[21] Chazelle B.: Approximation and decomposition of shapes, Advances in Robotics Vol I: Algorithmic and Geometric Aspects of Robotics, Schwartz J. T. and Yap C. K., Eds. Hillsdale, NJ: Lawrence Erlbaum, 1987, pp. 145-185.

[22] Benayad M. A., Campion G., Wertz V., Achhab M. E.: Steering a mobile robot: selection of a velocity profile satisfying dynamical constraints, Asian Journal of Control, **2**, 4, 2000, pp. 219-229.

[23] Samuel S., Keerthi S. S.: Numerical determination of optimal non-holonomic paths in the presence of obstacles, Proc. IEEE Int. Conf. on Robotics and Automation, Georgia, 1993, pp. 826-831.

[24] Soueres P., Boissonnat J. D.: Robot motion planning and control, New York, Springer, 1998.

[25] Ge S. S., Lai X. C., Mamun A. A.: Sensor-based path planning for nonholonomic mobile robots subject to dynamic constraints, Robotics and Autonomous Systems, 55, 2007, pp. 513-526.

[26] Ge S. S., Cui Y. J.: Dynamic motion planning for mobile robots using potential field method, International Journal of Autonomous Robots, 13, 2002, pp. 207-222.

[27] Borenstein J., Koren Y.: Real-time obstacle avoidance for fast mobile robots, IEEE Transactions on Systems, Man and Cybernetics, **19**, 5, 1989, pp. 1179-1187.

[28] Luh G. C., Liu W. W.: An immunological approach to mobile robot reactive navigation, International Journal of Applied Soft Computing, **8**, 1, 2008, pp. 30-45.

[29] Raja P., Pugazhenthi S.: Path planning for mobile robots in dynamic environments using particle swarm optimization, Proc. IEEE Int. Conf. on Advances in Recent Technologies in Communication and Computing (ARTCom 2009), Kottayam, India, 2009, pp. 401-405.

[30] Raja P., Pugazhenthi S.: Path planning for a mobile robot in dynamic environments, International Journal of the Physical Sciences, **6**, 20, 2011, pp. 4721-4731.

[31] Raja P., Pugazhenthi S.: Optimal path planning of mobile robots: A review. Accepted for publication, International Journal of the Physical Sciences. 2012.