Optimal Path Planning of a Mobile Robot using Quadrant Based Random Particle Optimization Method

Conference Paper · August 2014

5 authors, including:

Hamda Khan
National University of Computer and Emerging Sciences

Intsar Saeed
Technische Universität Hamburg

Optimal Path Planning of a Mobile Robot using Quadrant Based Random Particle Optimization Method

*Hamda Khan
Department of Mechatronics Engineering
Air University
E9, Islamabad, Pakistan
110883@students.au.edu.pk
Hamda,khan@mail.au.edu.pk

Intsar Saeed
Department of Mechatronics Engineering
Air University
E9, Islamabad, Pakistan
100420@students.au.edu.pk

Haider Saif Agha
Department of Mechatronics Engineering
Air University
E9, Islamabad, Pakistan
100439@students.au.edu.pk

Liaquat Ali Khan
Department of Mechatronics Engineering
Air University
E9, Islamabad, Pakistan
liaquat@mail.au.edu.pk

***Abstract* – The Random Particle Optimization (RPO) method, used for determining the optimal path from a specified initial pose of a mobile robot in a static or dynamic environment, models obstacles and target as repulsive and attractive Coulomb potentials respectively. It is a sensor-based method drawing its methodology from bacterial foraging just as Genetic Algorithms (GA) are based on Darwin's genetic evolution of species. The RPO, like GA and random search methods such as the Particle Swarm optimization (PSO) and Ant Colony Optimization (ACO) are effective in obtaining global optima for large Non-Polynomial NP-hard path-planning environments. It has the advantage however that it is not restricted to 'link points' which are usually incorporated in GA algorithms.**

This study aims to investigate a computational acceleration scheme in RPO for obtaining the optimal path for deterministic NP time hard problems. The use of bacterial foraging, or 'artificial points' on the next time step of a mobile robot are randomly biased towards the quadrant of interest, in line with the Euclidean path of shortest distance in a 2D workspace, thus reducing the computational processing. Thus, the sample size is reduced with the advantage of less on-line processing for an FPGA-based autonomous robot. The resulting optimal path is compared with the standard RPO as well as an arbitrary-swing RPO to show the computational advantage.

Keywords: optimal path, mobile robots, random particle optimization

I. INTRODUCTION

Path planning of mobile robots [2, 5, 6, 10, 11] is an active area of research in mechatronics extending to articulated robotic manipulators as well as wheeled robots. Early work included geometrical methods such as Voronoi diagrams, Dubin's car and Reeds and Shepp [12] models for idealized mobile robots. Variational methods [13] were developed and used to determine optimal trajectories. With increasing complexity, deterministic methods were found to be inefficient for such optimization methods classified as NP-hard problems. Thus, stochastic methods such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Random Particle Optimization (RPO) were developed to carry out efficient random searches for optimality. A widely used stochastic method is Genetic Algorithms [1, 3, 4] based on biological evolution processes in nature. Such heuristic methods have also been implemented on mobile robots using Field Programmable Gate Arrays (FPGA's) and demonstrated to be capable of determination of optimal paths in both static and dynamic environments [8].

This paper considers the determination of an optimal path using the Random Particle Optimization method in a static environment. The efficiency of the computational scheme in standard RPO is altered by selecting artificial points only in the 'quadrant of interest' i.e. in the quadrant that focuses towards the target. The robot, obstacles as well as target are considered as points rather than of finite dimension. This implies that the robot has no turning radius as incorporated in the Dubin's car representation. Such 'finite' effects can easily be incorporated by requiring a minimum width of the repulsive potentials.

In robotics, optimal control implies minimum time, minimum energy, minimum fuel, minimum off-track error from a desired prescribed trajectory or a combination of these performance indicators [7].

This work is of relevance to optimal path planning of mobile robots and for implementation through appropriate controllers.

II. THEORY

The Random Particle Optimization Method [9] has recently been proposed for the path planning of mobile robots in dynamic environments. Such environments are versatile as they incorporate sensor-based updates which enable path planning in the presence of moving obstacles. The obstacles and the target are expressed as repulsive and attractive Coulomb barriers, respectively, represented by Gaussian potentials of specified intensity and width (see fig. 1).

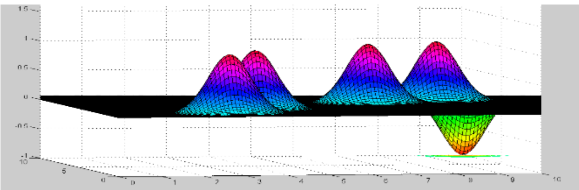

Fig. 1. Obstacles and target shown as interaction potentials

In a static environment, the locations of the obstacles and target are known a priori and thus an off-line implementation is possible. In the case of a dynamic environment where obstacles and target, or either of the two, are moving, the RPO method is required to be performed on-line with information gathered by on-board sensors.

Fig. 1 shows the four obstacles and a target represented by the potentials

$$J^{(obs)}(\bar{r}) = \alpha_o \exp(-\mu_o(\bar{r} - \bar{r}_o)^2) \qquad (1)$$

and

$$J^{(T)}(\bar{r}) = -\alpha_T \exp(-\mu_T(\bar{r} - \bar{r}_T)^2) \qquad (2)$$

With

$\alpha^{(obs)} = 1, \mu_o = 1$ and $\alpha^{(T)} = 1, \mu_T = 1$.

The optimal path from a start point to an end point avoiding the obstacles consists of two parts *viz*:

(i) the glide phase which is along a straight line when no obstacles are sensed by a sensor placed on the robot, and
(ii) a maneuver phase when the robot encounters an obstacle.

In the guide phase, the robot sets out in a direction obtained by calculated the shortest path i.e. a straight line from its starting point to its intended target. When an obstacle is detected in the path of the robot, the maneuver phase is used to determine the next point in the path. Flow chart is shown in fig. 2.

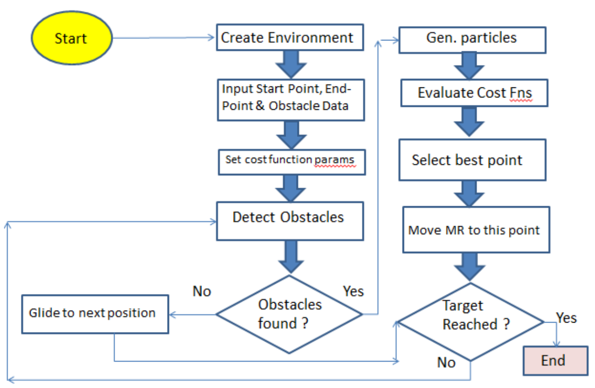

Fig. 2. Flowchart for the RPO program

The algorithm, following the principles of bacterial foraging works as follows:

Artificial particles are placed, at equal angle intervals, on a circle of radius C_t

i- At each particle location, the distance error: $e_s^d(t+dt) = d_s(t+dt) - d_s(t)$ is calculated. Here time t refers to the present position of the robot, time $t+dt$ refers to the artificial particles $s = 1,2,3,\cdots n$, and $d_s(t+dt)$ is the distance from the artificial particle to the target. Clearly $e_s^d(t+dt) < 0$ for a particle closer to the target and $e_s^d(t+dt) > 0$ for a particle further away from the target than the present position. The 'best' new point based on this metric would thus have the lowest negative value of the distance error.

ii- The second metric calculated at each new point is the cost function error $e_s^J(t+dt) = J(\theta_s(t+dt)) - J(\theta_s(t))$ where the cost function is a repellant-attractive Gaussian cost function for the obstacle $J^{(obs)}$ and target $J^{(T)}$ respectively and $J = J^{(obs)} + J^{(T)}$.

iii- The procedure is then to move the robot to the best point found *viz* one for which the distance error is smallest and negative, and the cost function is also negative.

III. Results

We consider the effect of a number of model parameters on the optimality of the path based on the 'best' point selected from the artificially generated points.

A. *Effect of Increasing Sample Size (N)*

The effect of simulating a larger number of points (N=100) is shown in Fig. 3 together with the resulting optimal path to the target. The total distance travelled to the target in this case is 2.6545 which, for a larger sample size (N=500) decreases to 2.6388 which remains stable to as far as N~10,000.

Fig. 3 shows the Robot, Obstacle and Target on a 2-D grid with robot position R at t=0 and N=100 artificial generated points on a circle of radius Ct . The 'optimal path' is shown in red color. It was found that (i) there is no optimal path, for N<10, that satisfies both criteria of distance and cost function, (ii) that optimality is a weakly dependent function of the sample size beyond N=10, and that (iii) the angle found can vary from 'one side' to the 'other side' of the obstacle without any effect on the total distance travelled to the target. This model, of selecting points based on the distance as well as repulsive-attractive potentials, is thus found to be stable.

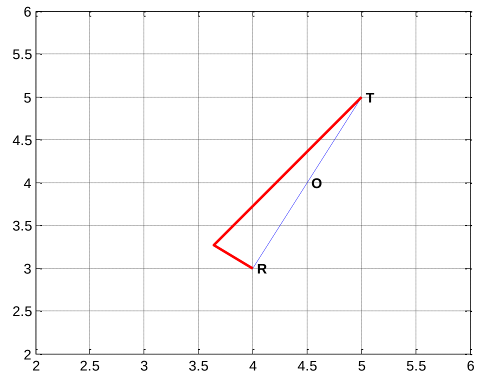

Fig. 3. A Simplified Optimal Path

Fig. 4 shows the Robot, Obstacle and Target on a 2-D grid with robot position R at *t=0* and *N=500* artificial generated points on a circle of radius C_t . The 'optimal path' is shown in red color.

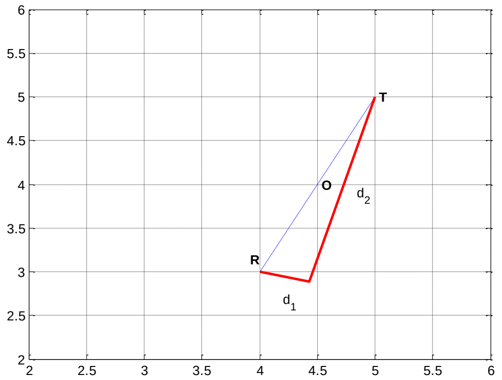

Fig. 4. A simplified optimal path with 500 generated points

B. *Effect of Varying Potential Parameters*

The effect of varying potential parameters α and μ for the obstacle (repulsive) and target (attractive) potentials $J^{(obs)}(\bar{r})$ and $J^{(T)}(\bar{r})$ was also investigated. For the sample size, we fix $N = 100$ which should be large enough to show any effect, on the optimal distance, of varying the parameters. It was observed that there is no effect of varying the magnitude of the repulsive potential; this is due to the close proximity of the robot to the obstacle and due to its relatively larger distance from the goal. The latter contribution is of a very small magnitude and is insufficient to have an effect on the overall cost function J. We now consider the case of a strong attractor, even though it is far from the robot, to observe the effect on the cost function with a small sample size. It was observed that for *N<10*, no feasible point could be selected; this was due to the large difference observed in the magnitudes of the cost functions. The target attractive cost function was very small in magnitude due to the Gaussian exponential reduction making it impossible for *J* to have an overall (repulsive + attractive) favorable (-ve) value.

We thus considered a very small sample size, *N=10* and found the first feasible solution with $\alpha^{(T)} = 10^6$. It was found that a solution is found only because of the comparable increase in magnitude of J_{goal} and hence J values. The optimal distance to target is found to be 2.2371.

Fig. 5 shows the arbitrary swing path and the optimal path for a mobile robot in a 2D environment initially at the point (0, 0) with seven obstacles required to go to the target point (10, 10).

Coordinates of the seven obstacles are given below in Table I

.

Table I. Obstacle and respective x, y coordinates

Obstacle	x-coordinate	y-coordinate
1	3.0	3.0
2	8.0	8.5
3	7.2	7.0
4	4.05	4.2
5	2.10	2.2
6	7.50	7.4
7	3.20	6.0

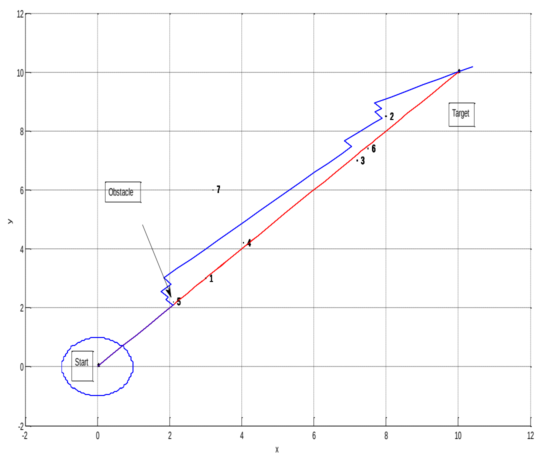

Fig. 5. Swing and RPO from start to target

As stated before, the arbitrary swing is taken without regard to the finite turning radius of a robot as incorporated in the Dubin's car representation. This, however, is a realistic 'swing' that must be taken into account when implementing the RPO algorithm for a robot. It can result in a significant increase in the distance travelled as shown in Fig. 6.

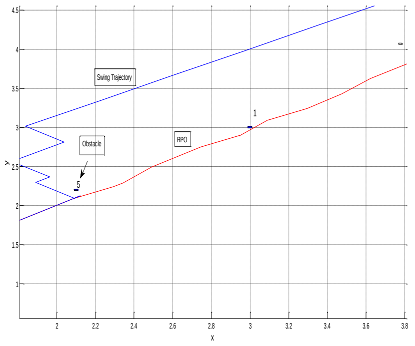

Fig. 6. Maneuver phase near an obstacle

C. *Quadrant-based RPO*

We consider generating artificial points in the quadrant of interest. Fig. 7 shows the 2D reference trajectory taken for the problem. It is the trajectory which the robot will follow and discard any other available path.

Fig. 8 depicts the arbitrary swing which the robot might take at any time when it detects an obstacle in its path while following the reference trajectory.

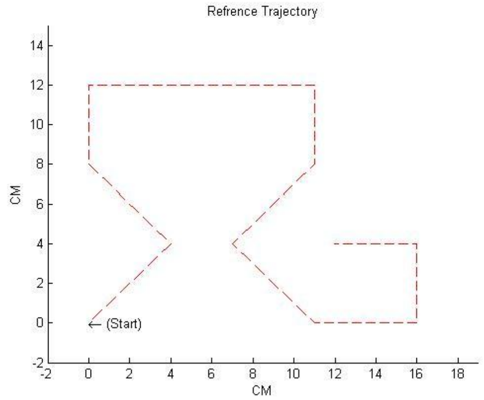

Fig. 7 Reference Trajectory

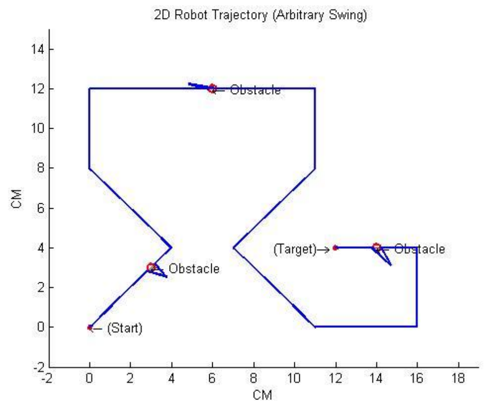

Fig. 8 Arbitrary Swing

Fig. 9 shows the trajectory taken by the robot while following the reference path, the trajectory was determined using the quadrant based RPO method.

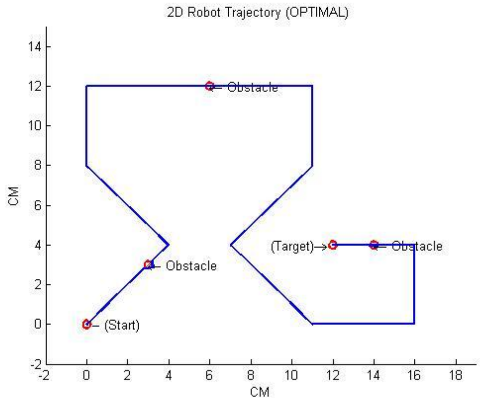

Fig. 9 Optimal path with obstacles on reference trajectory

Fig. 10 shows that the error converged after 200 points generation using quadrant based RPO method.

The complete results are given in Annex A.

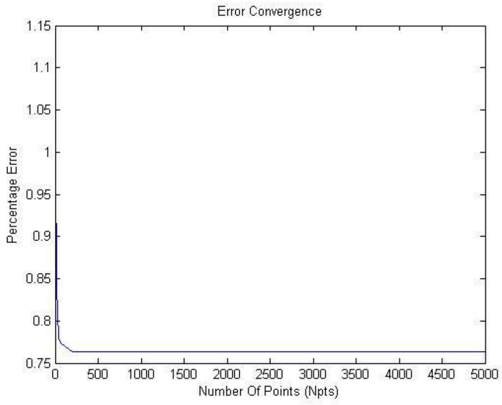

Fig. 10 Error Convergence Plot

IV. Conclusions

The Random Particle Optimization method was used to obtain the optimal path *i.e* the shortest path for a mobile robot from a starting point to a target point in the presence of static obstacles. The method was found to be computationally efficient as a small number of points N is sufficient for determining optimality and is thus useful as an onboard algorithm.

It was also found that:

i. the magnitude of the repulsive barrier $\alpha^{(obs)}$ does not contribute significantly when the robot is in close proximity to an obstacle,
ii. the width of the repulsive barrier $\mu^{(obs)}$can be of use for obstacles of large size, and
iii. that the magnitude of the attractive potential $\alpha^{(T)}$ should be orders of magnitude higher than that for the repulsive barrier due to the Gaussian exponential, especially for large distance of the robot from the target, to effectively 'pull' the robot towards the target.

The resulting RPO algorithm, especially in case of a static environment, can be input into the onboard computer of the robot as a reference trajectory for which an optimal control strategy can readily be computed for mechanical actuation.

When quadrant based RPO method was used, the computational time was significantly reduced as the area in which points were generated was reduced to a quadrant from a full circle thus reducing the total number of generated points.

REFERENCES

[1] H. Burchardt, H. and R. Salomon, *Implementation of Path Planning using Genetic Algorithms on Mobile Robots*, Institute of Applied Microelectronics and Computer Engineering, University of Rostock, Germany, 2004. Cruz-Bernal, A., Meta-Heuristic Optimization Techniques and Its Applications in Robotics, InTech 2013.
[2] A. ElShamli, *Mobile Robots Path Planning Optimization in Static and Dynamic Environments*, MSc Thesis, University of Guelph, 2004
[3] R.L.Haupt and S.E., Haupt, *Practical Genetic Algorithms*, Second Edition, John Wiley & Sons Inc., 2004.
[4] J.H. Holland. *Adaptation in Natural and Artificial Systems*, Ann Arbor: University of Michigan Press, 1975.
[5] J.P.Laumond, Editor, *Robot Motion Planning and Control*, Springer-Verlag London Limited 1998.
[6] S.M.LaValle., *Planning Algorithms*, Cambridge University Press, 2006.
[7] F.Lewis, ,D.Vrabie, and V. Syrmos, *Optimal Control*, John Wiley and Sons, Inc., 2012.
[8] I. Mahmoud, M. Salama, and A. Abd El Tawab," Implementation of Hardware Genetic Algorithm", Proceedings of ICEENG, 2008, Cairo, Egypt.
[9] K. Mohajer, E. Kiani, E. Samiei and M. Sharifi, "A new online random particles optimization algorithm for mobile robot path planning in dynamic environments", *Mathematical Problems in Engineering*, vol 2013, Article ID 491346, January 2013.
[10] G. Nagib, and W. Gharieb., 2004, Path planning for a mobile robot using genetic algorithms, Proceedings of the International Conference on Electrical, Electronic and Computer Engineering (ICEEC'04), Cairo, Egypt, pp.185-189.
[11] P. Raja and S. Pugazhenthi, "Optimal path planning of mobile robots: A review", *International Journal of Physical Sciences*, Vol. 7(9), pp. 1314-1320, 23 February, 2012.
[12] J.A.Reeds and L. A.Shepp, "Optimal Paths for a Car that Goes Both Forwards and Backwards", *Pacific Journal of Mathematics*, Vol. 145, No. 2, 1990.
[13] G.C.Walsh, R. Montgomer, and S.S.Sastry, "Optimal Path Planning on Matrix Lie Groups", WP-11, *Proceedings of the 33rd Conference on Decision and Control*, Lake Buena Vista, FL, December 1994
[14] H. Khan, Optimal Path Planning of Mobile Robots using Computational Algorithms, MS Thesis, Air University, Islamabad, Pakistan, 2013.

Annex A

Npts	Ө˚(obs.)	Ө˚(Mn.)	X (Mn.)	y (Mn.)	errJ	errD	Distance (no obs.)	Distance (Obs.)	Error	% Error	CPU Time (s)
3	45	0	3.2472	2.8	-0.058793	-0.44325	54.627417	55.2462585	0.619	1.13284	0.833
	0	15	6.232	12.115744	-0.087857	-0.16458					
	180	165	13.768	4.1157439	-0.08786	-0.42818					
10	45	0	3.2472	2.8	-0.058793	-0.44325	54.627417	55.16898657	0.542	0.99139	0.83
	0	9	6.2416942	12.069957	-0.075863	-0.20752					
	180	171	13.758306	4.0699575	-0.075867	-0.4403					
20	45	4.5	3.2458214	2.8350869	-0.021812	-0.44659	54.627417	55.08653461	0.459	0.84045	0.882
	0	4.5	6.2458214	12.035087	-0.07072	-0.23829					
	180	175.5	13.754179	4.0350869	-0.070724	-0.44547					
40	45	6.75	3.2441002	2.8525627	-0.003832	-0.44717	54.627417	55.05362966	0.426	0.78022	0.82
	0	7.75	6.2468552	11.982443	-0.069426	-0.28165					
	180	177.75	13.753145	4.017557	-0.069431	-0.44677					
80	45	6.75	3.2441002	2.8525627	-0.003832	-0.44717	54.627417	55.0494064	0.422	0.77249	0.91
	0	8.875	6.2471138	11.99122	-0.069102	-0.27468					
	180	181.13	13.752886	3.9912198	-0.069107	-0.44709					
200	45	7.2	3.2436737	2.856049	-0.000287	-0.4472	54.627417	55.04467772	0.417	0.76383	0.912
	0	359.55	6.2471862	11.996488	-0.069011	-0.27044					
	180	179.55	13.752814	4.0035123	-0.069016	-0.44718					
500	45	7.2	3.2436737	2.856049	-0.000287	-0.4472	54.627417	55.04448163	0.417	0.76347	0.968
	0	359.82	6.2471978	11.998595	-0.068997	-0.26874					
	180	179.82	13.752802	4.0014049	-0.069001	-0.4472					
1000	45	7.2	3.2436737	2.856049	-0.000287	-0.4472	54.627417	55.04445358	0.417	0.76342	0.914
	0	359.91	6.2471994	11.999298	-0.068995	-0.26817					
	180	179.91	13.752801	4.0007025	-0.068999	-0.4472					
5000	45	7.236	3.2436384	2.8563278	-3.99E-06	-0.4472	54.627417	55.04416362	0.417	0.76289	1.33
	0	359.98	6.2472	11.99986	-0.068994	-0.26771					
	180	179.98	13.7528	4.0001405	-0.068999	-0.4472					