American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)

ISSN (Print) 2313-4410, ISSN (Online) 2313-4402

http://asrjetsjournal.org/

Path Planning Algorithm using D* Heuristic Method Based on PSO in Dynamic Environment

Firas A. Raheem[a]*, Umniah I. Hameed[b]

[a]*Assistant Prof. Dr. ,University of Technology , Baghdad , Iraq*

[b]*Engineer , University of Technology , Baghdad , Iraq*

[a]*Email: 60124@uotechnology.edu.iq*

[b]*Email: 61126@uotechnology.edu.iq*

Abstract

This paper is devoted to find a short and safe path for robot in environment with moving obstacles such as different objects, humans, animals or other robots. A mixing approach of robot path planning using the heuristic method D star (D*) algorithm based on optimization technique is used. The heuristic D* method is chosen for finding the shortest path. Furthermore, to insure the path length optimality and for enhancing the final path, it has been utilized the Particle Swarm Optimization (PSO) technique. This paper focuses on computational part of motion planning in completely changing dynamic environment at every motion sample domains. Simulation results are given to show the effectiveness of the proposed method.

Keywords: D* Algorithm; Particle Swarm Optimization (PSO); Path Planning; known Dynamic Environment.

1. Introduction

The words “path planning” in robotics where the process is totally or sub automated. It is used to refer to this type of computational process for moving an object from one place to another with respect to obstacles. In other word it is searching for initial feasible path and this is the first step in the planning process for the robot path.The path planning problem is that of finding a path for a robot that must move from initial point, which is given as the start position to the goal point which is given as the destination position, in an environment that contains a fully defined set of obstacles. Therefore, the robot does not collide with any of the obstacles [1-4]. Generally, obstacles cannot always be static. When the object stays motionless in the environment, it calls static, i.e., fixed constantly at a location, but when the object changes its location over time, it calls dynamic.

---

* Corresponding author.

Dynamic environment represents a revolution and important side in modern automation, it redound in many important diverse applications like, air and sea traffic control negotiating, freeway traffic, intelligent vehicles, automated assembly and automated wheel chairs. Because of existence the moving objects, the robot needs to make decision quickly for avoiding possibility collisions. This represents the mutual part among these applications. The ability to deal with moving objects is needful for the navigation of any factual robot. The range of applications of intelligent robots could be widely increased since it’s able to handle moving objects [5-7]. There are three types of dynamic environment, where all information is regarding to the obstacles, i.e., motions, sizes, shapes, locations, etc., is completely known, the obstacle is said to be in a known dynamic environment. As example for known environment the automated production line. Partially known Dynamic Environment is the second type when not the all information about the obstacle are exist at the planning time. In this state, it needs to calculate the robot motion with according to insufficient information about the environment. It is important to know when the robot's mission is to complete a map of the environment by scanning the area or to scan a certain area to make sure if there is a new object is found in the area. Finally when there is nothing to know about the obstacle it’s called the totally unknown dynamic environment and this need intelligent ways to be used. As example for unknown and partially known environment the other robots and humans in workspace [7].

To overcome the dynamic environment challenges, many path planning algorithms have been used in the literature. Here we started with Anthony Stentz [8] whom used The D* algorithm which is represented the dynamic A*, it aims to create a short path in real-time by incrementally reforming paths for the robot’s state as a new information is found. The D* extension which concentrate the repairs of significantly reduction the whole time required to subsequent and calculate the initial path re-planning operations which this paper aims to. Followed by Enric Cervera I Mateu [9] who suggested using robot programming architecture to learn fine motion tasks such as autonomous process of experience repetition could carry out the goal in minimized steps. This involves uncertainty in location, and the robot will be guided with sensory information by using force sensor but this work involves uncertainty in location.

Many other path planning algorithms were applied for solving dynamic environment problems such as deriving the non-linear v-obstacle for general planar obstacles to be useful in analytic expressions. Considering obstacles moving on arbitrarily trajectories is represents the concept of counting velocity obstacles. The nonlinear v-obstacle has been consisted of a deformed cone i.e. a time-scaled map of the obstacles that extend along their trajectories. Avoiding the obstacle during the time interval for which the v-obstacle has been generated could be achieved by selecting a single velocity vector outside the nonlinear v-obstacle. This method works accurate only with mobile robot [10]. As well as this, finding out a reasonable relation among parameters that used in the path planning strategies in a platform, where a robot will be able to move freely from the start location in a dynamic environment with map and make an optimal path to a set target location without collision with the static and moving objects. Researchers used a new optimal type algorithm of Rapidly-exploring Random Tree RRT algorithm, named RRT*. The mechanism of this strategy is based on incremental sampling that covers the all space and will act quickly. It can be used in more than single dimension environments and will be computationally efficient but they have found that a high number of sampling doesn’t give the best solution in addition to the higher computational time that was required for finding the path [11].

From a point of view, algorithm for specify Cartesian constraints in a single and dual arm operation is proposed and became a library for solving manipulator motion planning problems that has been developed. The main algorithm is based on RRT. One parameter is used to be the common in all RRT based on planers is the range. The range represents a threshold for the samples drawn newly and their small distance values after that those will add to the tree. Range value will be affective on the planning time which means that it should be selected carefully since the higher values causing short time planning which is advantage but as a result it also cause a tensive motions and the smaller values give a long time planning and it composed more points in the tree. Also more checking will be needed for the collision that may be occur in the space, therefore it needs extra collision checking [12].

Most of the existing literature in dynamic environment considers only the traditional path planning algorithms and there are only a few studies that consider the heuristic algorithms like Anirudh Vemula [13] whom uses a heuristic based graph search planning algorithm that uses adaptive dimensionality consider time dimension corresponding to region where potential dynamic obstacle collisions can occur. This reduction in search space supply the approach in great speed-ups in planning time over traditional approaches such as heuristic-based A*, will act without losing safety, dynamic feasibility and bounded sub-optimality of the solution This approach can only model simple interactions and fail to generalize for complex crowded settings.

Inspired by the above discussion, the contribution of the proposed method in this paper is represented by finding path solution utilizing the heuristic D* algorithm based on the PSO optimization technique in known dynamic environment. This work can solve the problem of finding an optimal path in dangerous, harsh, toxic environments and inaccessible areas.

2.The D* Theory

The D* algorithm, dynamic A*, is an algorithm that is generate short path in real-time by a graph with update or change arc costs. The utility of D* is that the most of arc cost corrections occurs in the robot's neighborhood. From this corrections, the only part of the path is going to out to the robot's current location is need for replanting. The partial optimal cost map are keep limited for the locations that the robot may use. Furthermore, the states that is probable to produce a new optimal path is used to reform of the map's cost which will be limited and re-entrant partially. D* sets the conditions for determining when repair will be stood, because of finding a new path or the previous path has been found is still optimal. So, the D* strategy is used to say that it is very memory active, works in unbounded environments and computationally as well. The mechanism of the D* algorithm is designed repeat finding optimal path to the directed graph, which means that the arc costs are assorted with transference cost values that could be ranged over a continuation. The corrections of the arc cost that came from the sensors it could be made in any time, also the estimated and measured arc costs that known is comprised to the environment map. This strategy could be used in any planning representation, including grid cell structures and visibility graphs. [14]

D* algorithm is the dynamic A*

A* algorithm equation:

$f(n) = g(n) + h(n)$

D* algorithm equation:

$f(n) = h(n)$

It could be propagated cost changes to its neighbors as shown in figure (1).

N(x1,x2)=1

N(x1,x3)=1.414

N(x1,x4)=100000, if x4 has an obstacle and x1 is a free cell.

N(x1,x5)=100000.4, if x5 has an obstacle and x1 is a free cell

The arc cost of the initial node = (0.0)

Where the arc cost of vertical and horizontal nodes are calculated as:

From x1 to x2 = $\sqrt{(\mathbf{0}-\mathbf{0})^{2}+(\mathbf{0}-\mathbf{1})^{2}} = \mathbf{1}$

From x1 to x4 = $\sqrt{(\mathbf{0}-\mathbf{1})^{2}+(\mathbf{0}-\mathbf{0})^{2}} = \mathbf{1}$

But the arc cost of diagonal nodes can be calculated as:

From x1 to x3 = $\sqrt{(\mathbf{0}-\mathbf{1})^{2}+(\mathbf{0}-\mathbf{1})^{2}} = \sqrt{\mathbf{2}} = \mathbf{1.414}$

And so on.

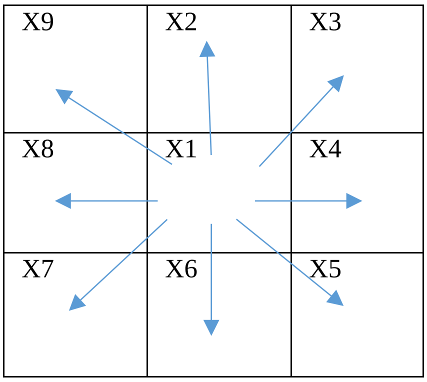


Figure 1: Node expansion

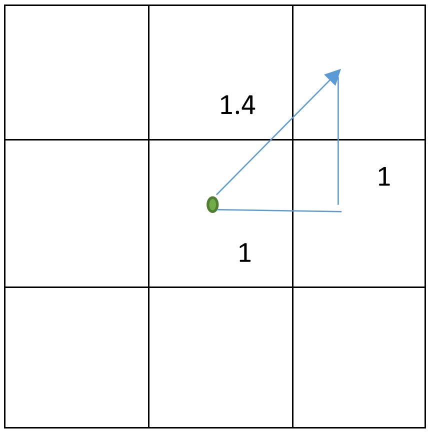


Figure 2: Calculation of node expansion

3.Particle Swarm Optimization Technique Theory (PSO)

PSO is a laborious stochastic optimization technique, it depends on the intelligence and movement of birds swarms. Inspired by the behavior of nature social and dynamic motions with communications of fish schooling, insects or birds flocking. A number of basic variations have been developed due to improve speed of convergence and quality of solution found by the PSO [15].

The principle of PSO technique depends on generated specified number of particles in random positions in the specified workspace, also the velocity of those particles is nominated randomly, each particle has a memory that stores all the best position have been visited before, in addition to the fitness in that position which has been improved over time [16]. In each iteration of PSO technique, the pos_{ij} and $vel_{i,j}^{t}$ vectors of particle i is modified in each dimension j in order to lead the particle i toward either the personal best vector ($pbest_{ij}$) or the swarm's best vector ($gbest_{ij}$)

$$vel_{i,j}^{t+1} = w \times vel_{i,j}^{t} + c_1 \times r_{1,j} \times \left(pbest_{ij} - pos_{ij}\right) + c_2 \times r_{2,j} \times \left(gbest_{ij} - pos_{ij}\right)$$

The position of each particle (bird) is updated by using the new velocity for that particle, according to equation below:

$$pos_{ij}^{t+1} = pos_{ij}^{t} + vel_{ij}^{t+1}$$

Where c_1 and c_2 are the cognitive coefficients (c_1+c_2<=4), and $r_{1,j}$ and $r_{2,j}$ are random real numbers between [0, 1], the inertia weight w controls the particle momentum [17-18].

The value of vel_{ij} is clamped to the range [-vel_{max}, vel_{max}] to reduce the probability of leaving the search space by the particle. If the search space is defined by the bounds

[-pos_{max}, pos_{max}], then the value of vel_{max} is typically set so that $vel_{max} = k * pos_{max}$, where $0.1 \leq k \leq 1.0$.

A large inertia weight (w) eases the local when its value is large and eases the global search when its value is small. To summarized PSO technique in steps it can be consist of just three steps:

1. Calculate fitness for all particles.

2. Update local and global bests.

3. Update velocity and position for all particles. [19-20]

5. Proposed Method: D* Algorithm Based On PSO

The path planning algorithm which has been used, is the heuristic D* algorithm that depends expansion nodes, in an environment with moving objects. Constructing the environment includes putting the map limitation, start and target locations and the obstacles locations and dimensions.

After constructing the environment, the algorithm will start its work. The robot firstly stands on the target node and start the initial expand to its 8-connected neighbor. Each node will have an initial cost function where the robot will choose the least cost function node to move on and put it in the closed list while the other nodes will go to the open list. Taking into consideration the obstacles having a huge cost function and sorting in the closed list from the beginning and the process is completed like that until reaching the start node. When the search is completed the robot will move from the start on the nodes that have been saved in the closed list to the target node (therefore it is called backward searching method) avoiding the obstacles nodes, the current node, and the next step movement node.

The velocity of the moving obstacle is constant and the velocity of the mass point is changing over time ($\mathrm{MS} \times T_s$) and it is calculated by:

$$v_x(t) = \frac{\Delta x\ (t)}{\mathrm{MS} \times T_s}$$

$$v_y(t) = \frac{\Delta y(t)}{\mathrm{MS} \times T_s}$$

Where Δx represents the change of distance in X-axis, Δy represents the change of distance in Y-axis, MS represents the motion sample and T_s represents the sample time which assumed to be 0.01 sec. After finding the D* path, the PSO optimization technique is used for enhancing the path, removing sharp edges and making it the shortest because the D* path has stairs shape sometimes. The proposed planning for the mass point in dynamic environment is shown in figurers (3). The first step to operate the PSO, is to set the parameters of PSO (c_1=1, c_2=3, w=1) which are found by a trial and error technique to reach the best results. Now, the D* path enters the PSO work, which will represent the D* path as the search space and choose the optimal points from it, taking into account the general shape for the D* path and without distorting the original form. After determining the PSO search space which is the D* path matrix, the search process begins with defining the number of via points (birds number) that search for the new points according to the search space dimension and the PSO parameters. These via points are chosen randomly from the search area (D* path) and will represent the position parameters in the PSO equation.

The cubic spline interpolation equation has been used to connect the best chosen points from the PSO optimization technique to generate corresponding smoothed path. The cost function of PSO which is the path length is computed. Then the elementary velocity is set, and updating the local and global parameters is the next step. The procedure is reiterated until the given points that are looked over the path are finished. The next step is updating the velocity and position equations, for converging the via points into the best cost, and also updating the local and global parameters, the flowchart in figure (4) explains the entire process.

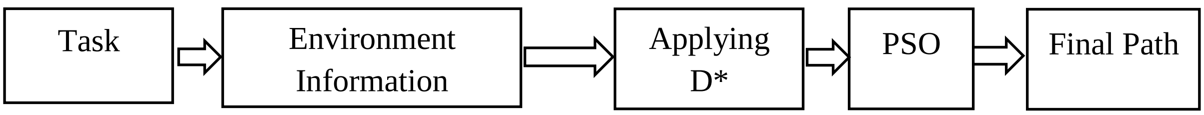


Figure 3: Proposed planning for the mass point in dynamic environment

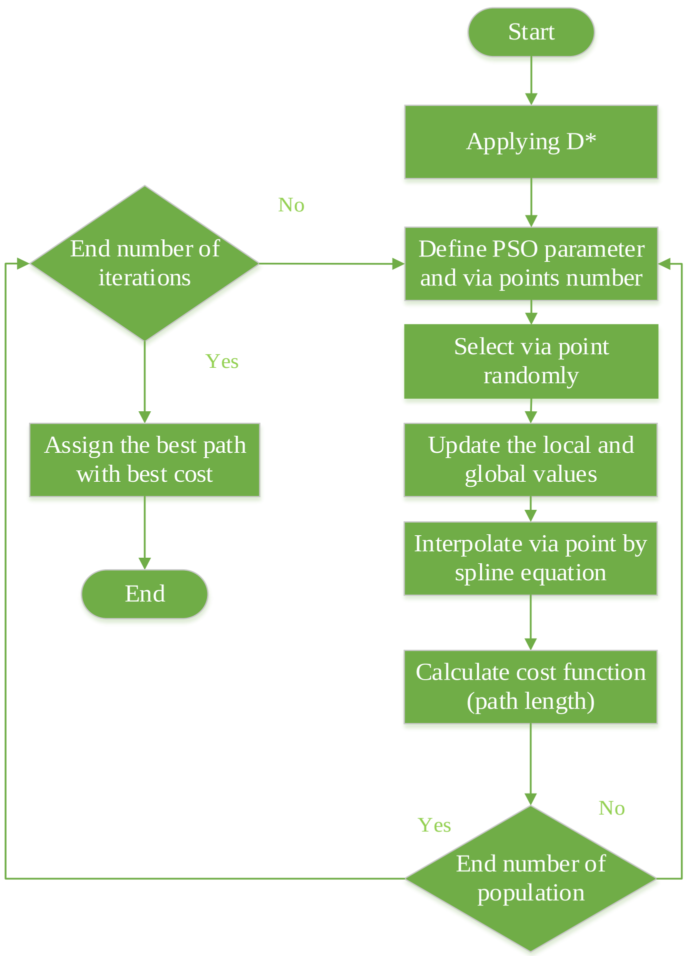
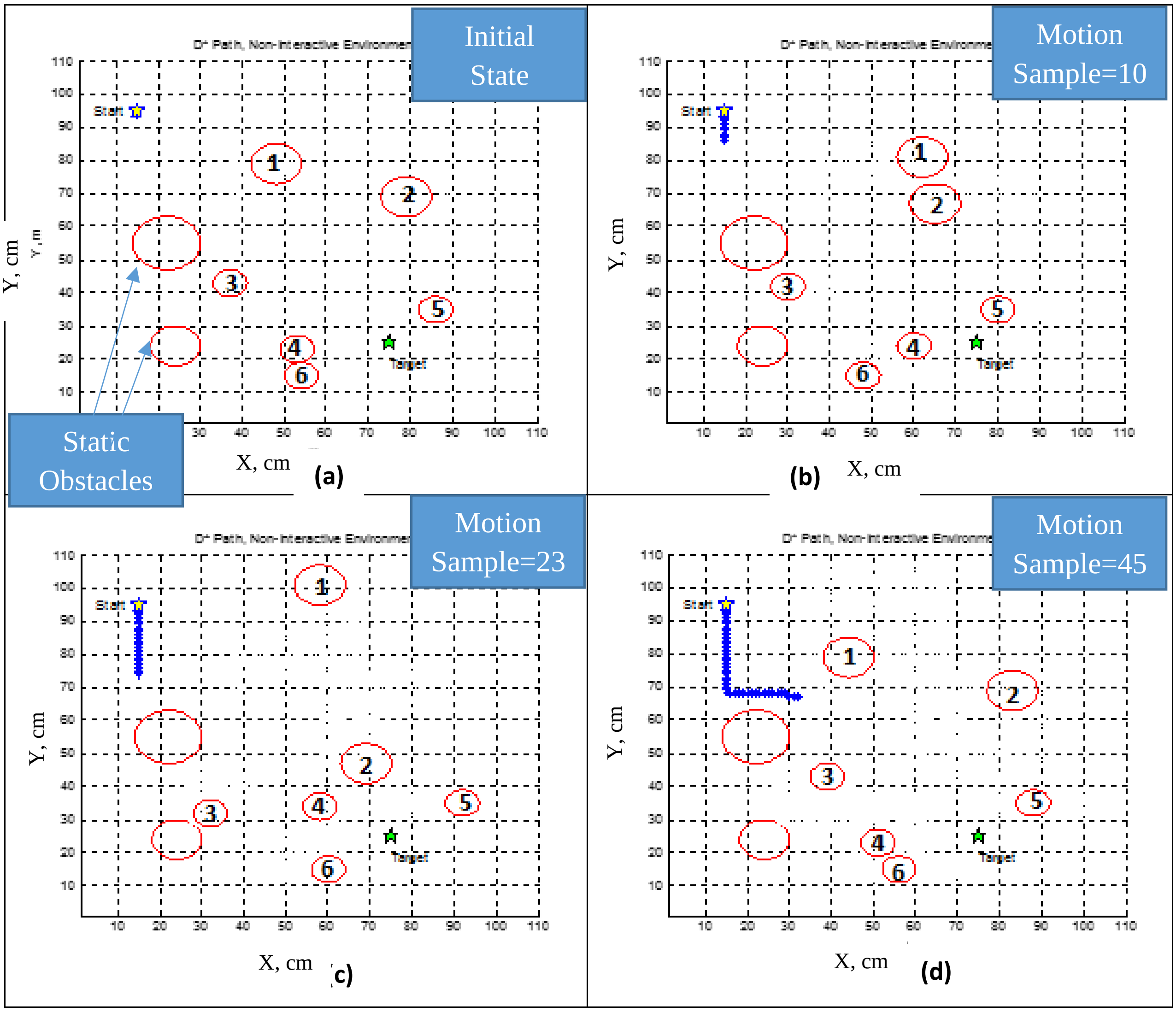


Figure 4: The heuristic D* Via Point flowchart for known obstacle position dynamic environment

6.Simulation Result

6.1 First Environment

This environment represents the non-interactive path solution among the robot and the moving obstacles in dynamic environment to find the shortest safe path using D* algorithm based on PSO. This environment features multi obstacles with different sizes and different motion styles. It have eight obstacles, tow are static and six are moving with different motion behavior. Also support critical paths only few possible paths could be found which choose the shortest. Figure (5) shows the scenario of dynamic result in case of D* path only which represents motion steps from, where the figure (a) to figure (h) according to different motion sample from initial state until achieving the goal with 106.5685 cm path long. Using D* based on PSO guarantees a collision free path as shown in figure (6.a). Mixing PSO with the D* path enhance the last to 102.6672 cm path long as shown in figure (6.b). The best coast after 300 iteration reduce the cost function with 3.9013 cm. The figure (7) shows the same motion scenario but in PSO case from initial state figure (a) to the goal figure (d).



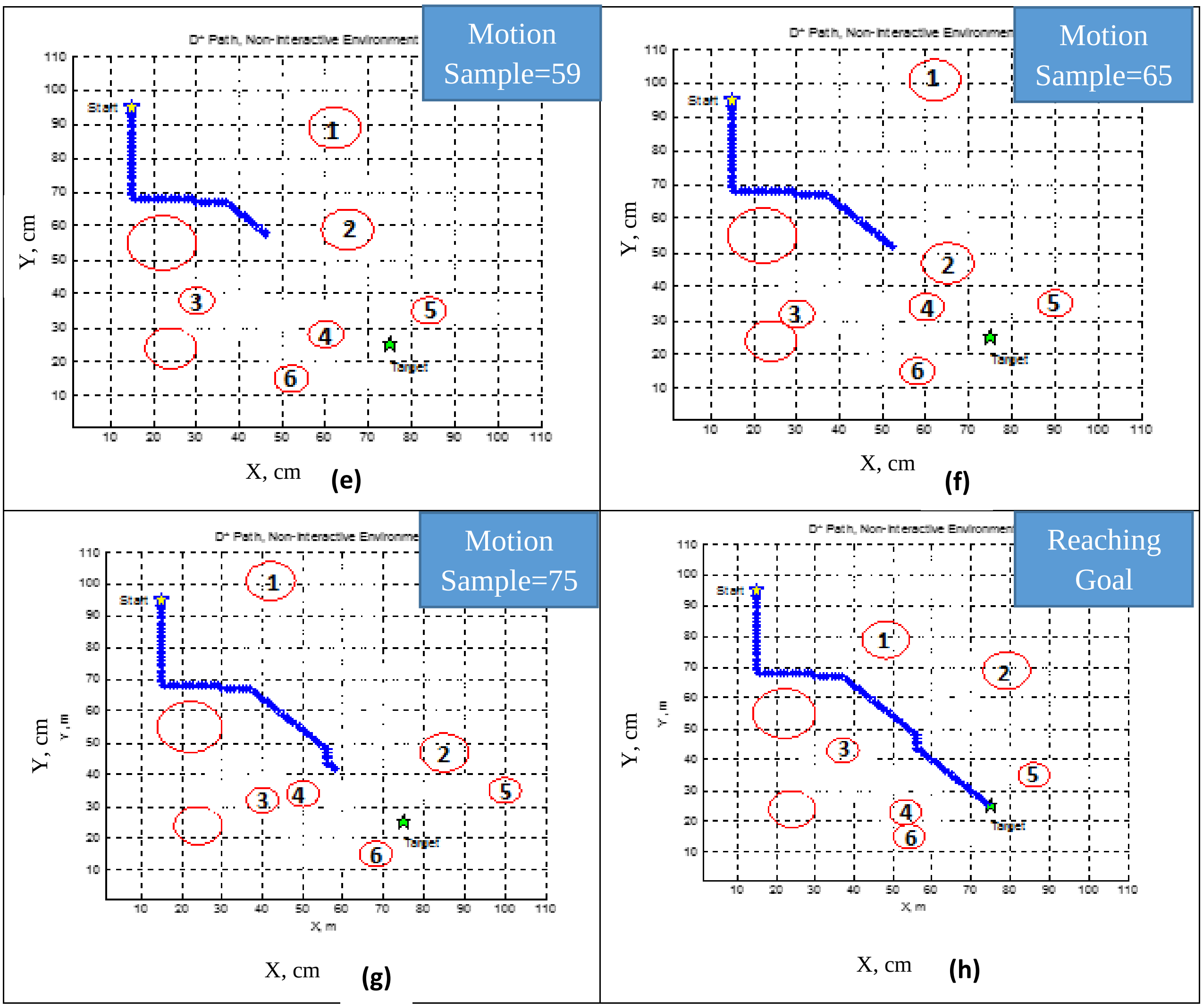


Figure 5: The heuristic D* path planning for the first environment

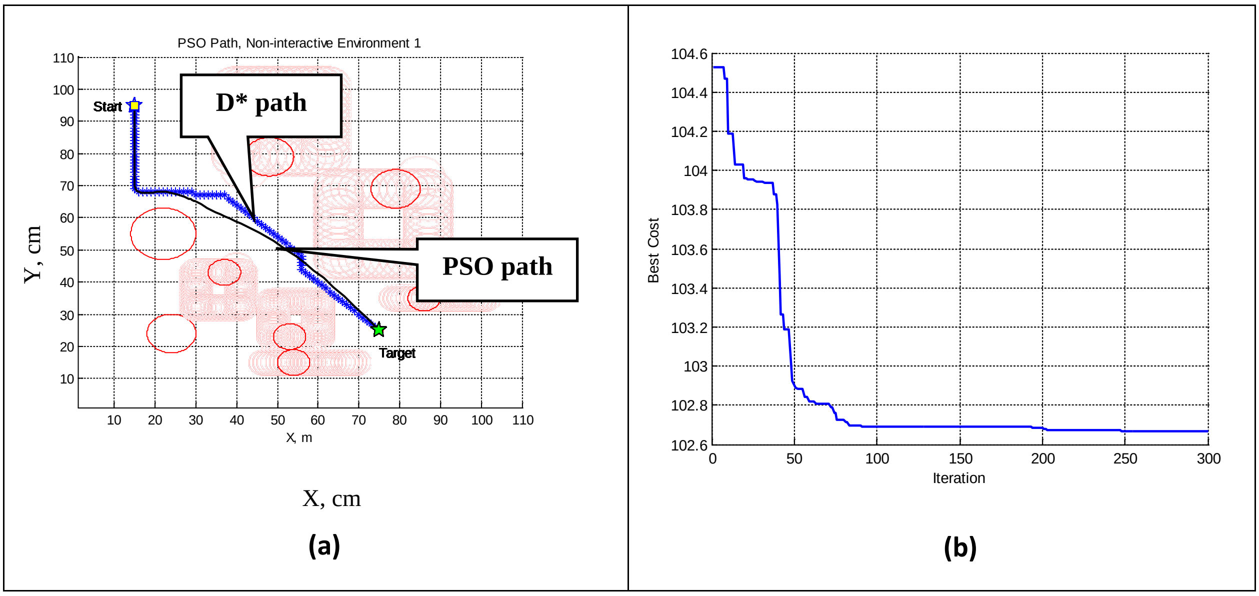


Figure 6: (a) represents D* path and PSO path, (b) represents the best cost of PSO path

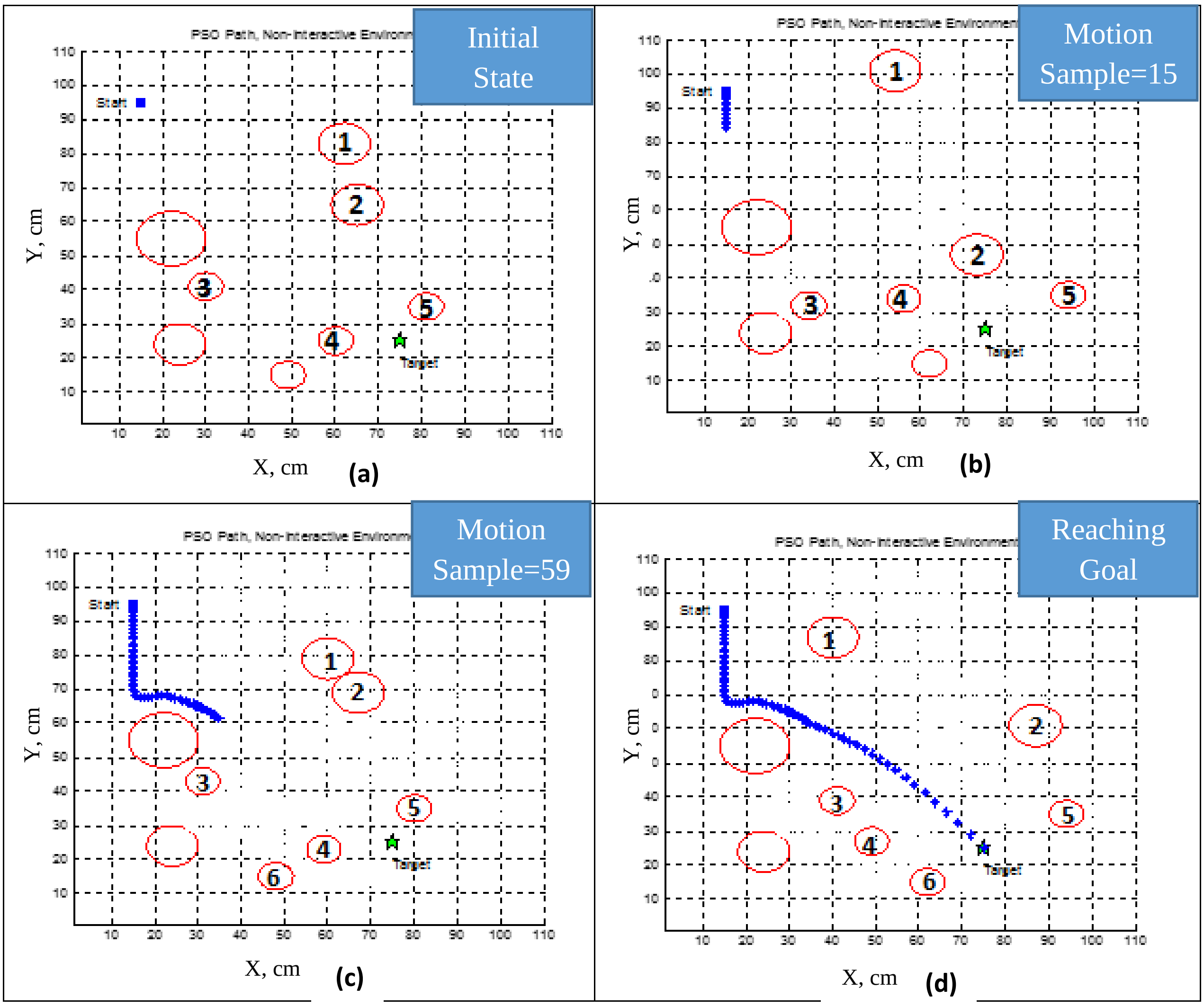
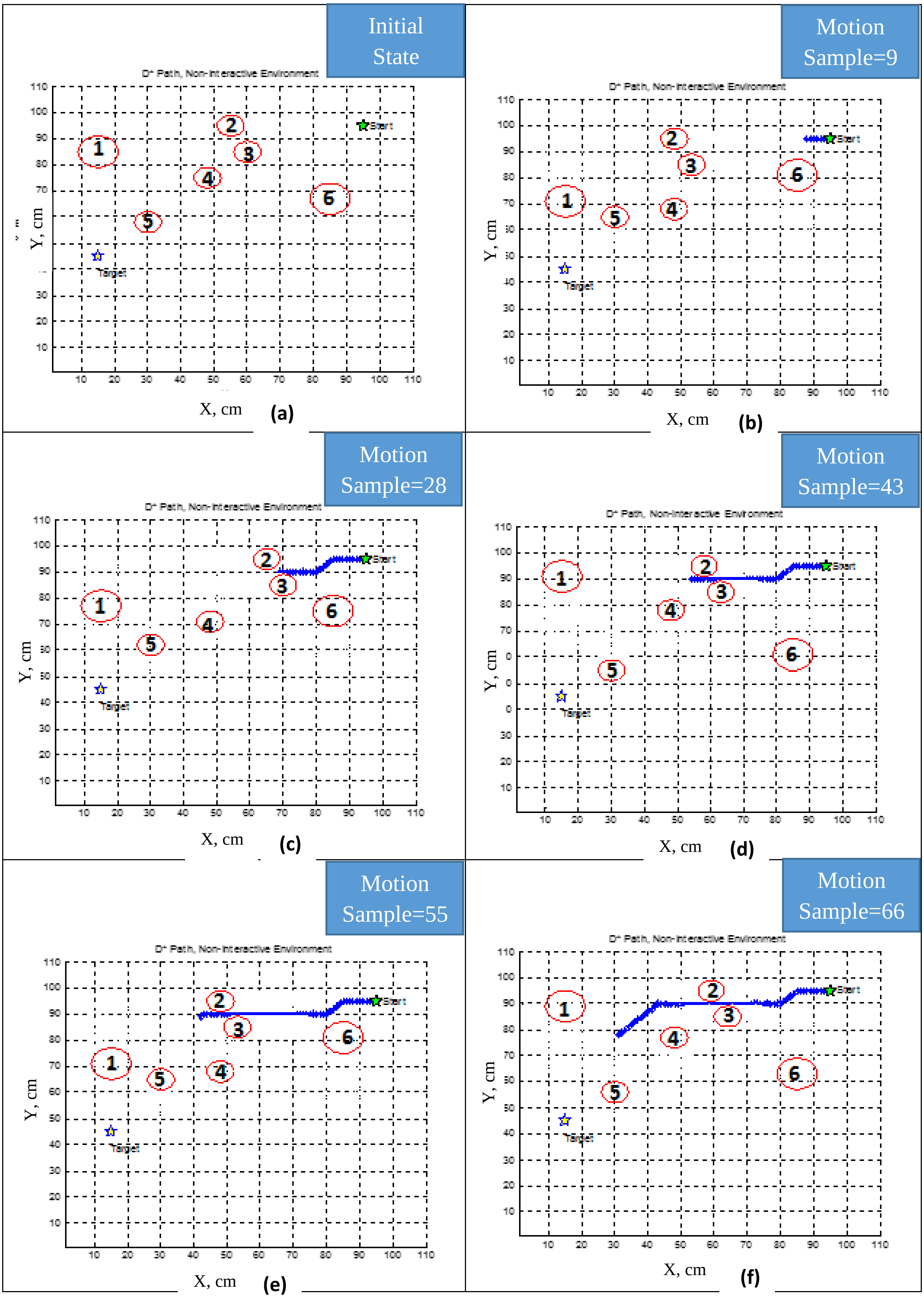


Figure 7: The heuristic D* based on PSO final path

6.2 Second Environment

This environment represents the non-interactive path solution among the robot and the moving obstacles in dynamic environment to find the shortest safe path using D* algorithm based on PSO. This environment features obstacles with different sizes and different motion styles. It have six moving obstacles with different motion behavior. It supports safe short path in narrow area as shown in figure (8) where the scenario of dynamic result in case of D* path only which represents motion steps from figure (a) to figure (h) according to different motion samples from initial state until achieving the goal with 106.5685 cm path long. Using D* based on PSO guarantees a collision free path as shown in figure (9.a). Mixing PSO with the D* path enhance the last to 106.6039 cm path long as shown in figure (9.b). The best coast after 300 iteration reduce the cost function with 4.0651cm. The figure (10) shows the same motion scenario but in PSO case from initial state figure (a) to the goal figure (d).


Initial State
D* Path, Non-interactive Environment
Start
Target
Y, cm
X, cm
(a)
Motion Sample=9
D* Path, Non-interactive Environment
Start
Target
Y, cm
X, cm
(b)
Motion Sample=28
D* Path, Non-interactive Environment
Start
Target
Y, cm
X, cm
(c)
Motion Sample=43
D* Path, Non-interactive Environment
Start
Target
Y, cm
X, cm
(d)
Motion Sample=55
D* Path, Non-interactive Environment
Start
Target
Y, cm
X, cm
(e)
Motion Sample=66
D* Path, Non-interactive Environment
Start
Target
Y, cm
X, cm
(f)

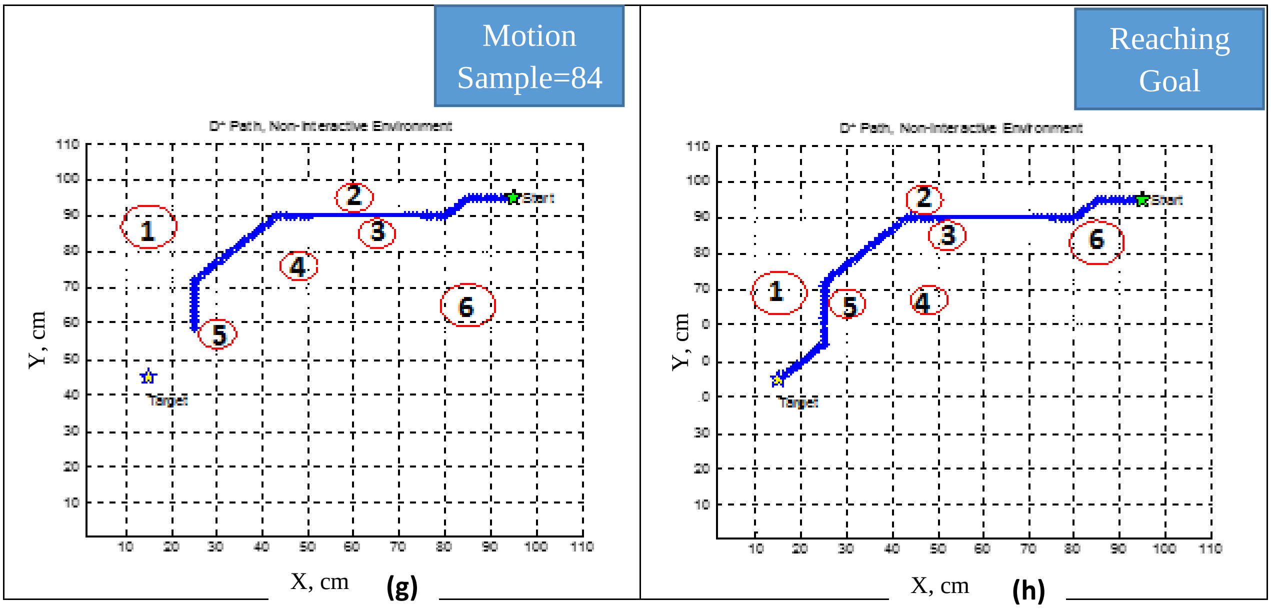


Figure 8: The heuristic D* path planning for the second environment

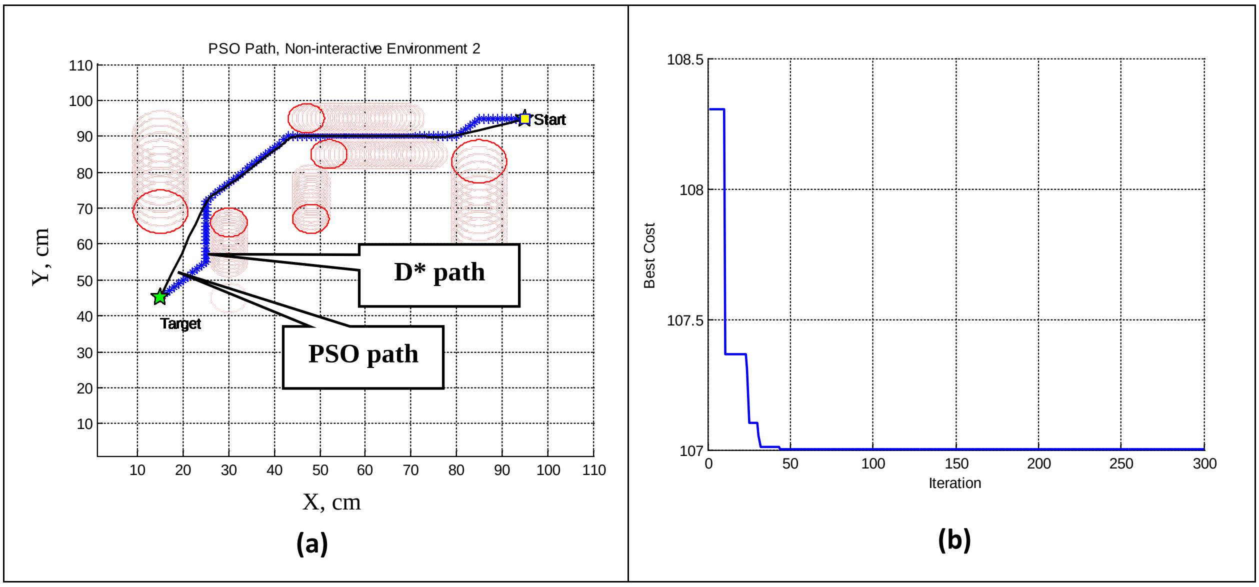


Figure 9: (a) represents D* path and PSO path, (b) represents the best cost of PSO path

Table 1: Environments information

environment	Population	Iteration	D* path	PSO path
1	50	300	106.5685 cm	102.6672 cm
2	50	300	110.6690 cm	106.6039 cm

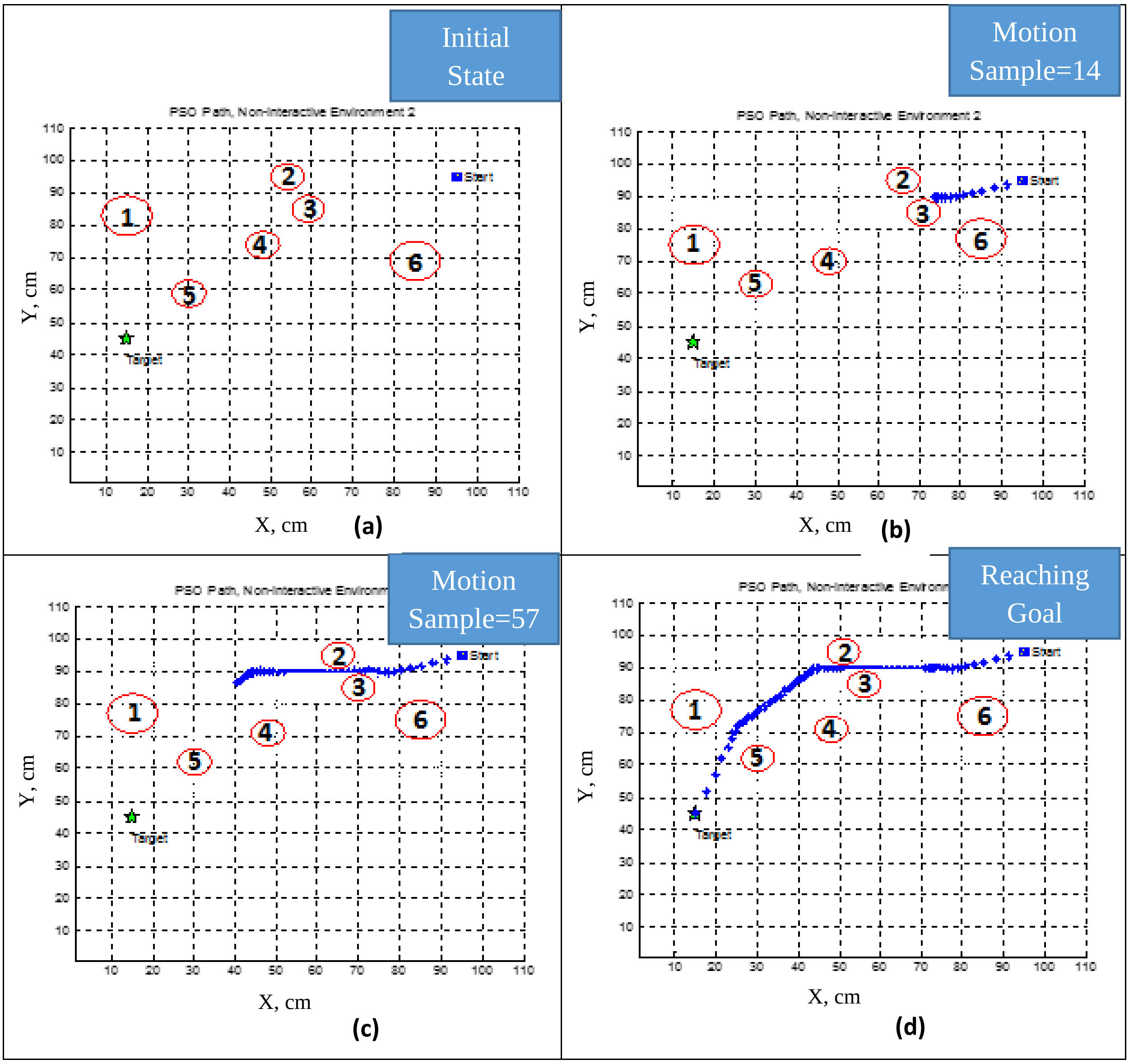


Figure 10: The heuristic D* based on PSO final path for the second environment

7. Conclusions

This paper proposed a methodology that considers solving the path planning problem in known position changes dynamic environment utilizing the use of heuristic D* algorithm for finding the best path and the use of PSO optimization technique for getting the final optimal path. At first, the D* algorithm is applied to find the shortest path then the Particle Swarm Optimization method is used to enhance the final path by removing the sharp edges till finding optimal path. The result clearly proves the advantage of using the proposed mixed approach. The length of the new path produced by the optimization technique was smooth, safe, optimal and shortest which traditional D* method could not find such a path.

References

[1] Hui Miao, “Robot Path Planning in Dynamic Environments using a Simulated Annealing Based

Approach", Faculty of Science and Technology Queensland University of Technology.

[2] F. A. Raheem and Asmaa A. Hussain, "MODIFIED C-SPACE ANALYSIS AND CONSTRUCTION FOR TWO-LINK ROBOT ARM," no. December, 2017.

[3] F. A. Raheem and Asmaa A. Hussain, "Applying A* Path Planning Algorithm Based on Modified C-Space Analysis",Al-Khwarizmi Engineering Journal,Vol. 13, No. 4, P.P. 124- 136 (2017).

[4] F. A. Raheem and and Mustafa M. Bader, "Development of Modified Path Planning Algorithm Using Artificial Potential Field (APF) Based on PSO for Factors Optimization," no. December, 2017.

[5] A. T. Sadiq F. A. Raheem, and N. A. F. Abbas, "Robot Arm Path Planning Using Modified Particle Swarm Optimization based on D * algorithm problems of local minimum stacking and has," vol. 13, no. 3, pp. 27–37, 2017.

[6] A. T. Sadiq, F. A. Raheem, and N. A. F. Abbas, "Optimal Trajectory Planning of 2-DOF Robot Arm Using the Integration of PSO Based on D * Algorithm and Cubic Polynomial Equation," First Int. Conf. Eng. Res., no. March, pp. 458–467, 2017.

[7] Kikuo Fujimura, "Motion Planning in Dynamic Environments", Springer-Verlag Tokyo, 1991.

[8] Anthony Stentz, "The Focused D* Algorithm for Real-Time Replanning", Robotics Institute Carnegie Mellon University Pittsburgh, Pennsylvania 15213 U. S. A. In Proceedings of the International Joint Conference on Artificial Intelligence, August 1995.

[9] Enric Cervera I Mateu, "Perception –Based Learning for fine motin planning in robot manipulation" , 1997

[10] Zvi Shiller, Frederic Large and Sepanta Sekhavat, "Motion Planning in Dynamic Environments: Obstacles Moving Along Arbitrary Trajectories", Mechanical and Aerospace Engineering,University of California, Proceedings of the 2001 IEEE.

[11] Roudabe Seif, Mohammadreza Asghari and Oskoei,Allameh Tabataba , "Mobile Robot Path Planning by RRT* in Dynamic Environments", April 2015 in MECS

[12] Behnam Asadi, "Single and Dual Arm Manipulator Motion Planning Library", 2015

[13] Anirudh Vemula, "Safe and Efficient Navigation in Dynamic Environments," no. July, 2017

[14] Anthony Stentz , "The D* Algorithm for Real-Time Planning of Optimal Traverses" , The Robotics Institute Carnegie Mellon University Pittsburgh, Pennsylvania 15213 September 1994.

[15] P. Boonyaritdachochai, et al., "Optimal congestion management in an electricity market using particle

swarm optimization with time-varying acceleration coefficients," Computers & Mathematics with Applications, vol. In Press, Corrected Proof, 2010.

[16] Riccardo Poli, James Kennedy and Tim Blackwell "Particle swarm optimization An overview " Springer Science 1 August 2007

[17] Enric Cervera and I Mateu, Perception –Based Learning for fine motin planning in robot manipulation, 1997.

[18] Behnam Asadi, Single and Dual Arm Manipulator Motion Planning Library, 2015.

[19] James Kennedy and Russell Eberhart Purdue School of Engineering and Technology (1995 IEEE).

[20] K. T. Chaturvedi, et al., "Particle swarm optimization with time varying acceleration coefficients for non-convex economic power dispatch," International Journal of Electrical Power & Energy Systems, vol. 31, pp. 249-257, 2009.