ISSN: 0067-2904

Collision Avoidance Using Cat Swarm Algorithm for Multi Mobile Robot Path Planning in Dynamic Environment

Themar Ali Jaleel*, Alia Karim Abdul Hassan
Computer Science Department, University of Technology, Baghdad, Iraq

Abstract

In this paper, we proposed a hybrid control methodology using improved artificial potential field with modify cat swarm algorithm to path planning of decoupled multi-mobile robot in dynamic environment. The proposed method consists of two phase: in the first phase, Artificial Potential Field method (APF) is used to generate path for each one of robots and avoided static obstacles in environment, and improved this method to solve the local minimum problem by using A* algorithm with B-Spline curve while in the second phase, modify Cat Swarm Algorithm (CSA) is used to control collision that occurs among robots or between robot with movable obstacles by using two behaviour modes: seek mode and track mode. Experimental results show that the proposed method success to find a complete, optimal, and collision free path for all robot.

Keywords: Decoupled planning, Artificial potential field method (APF), A* algorithm, B-Spline curve (B-S), Cat swarm algorithm (CSA).

تجنب التصادم بااستخدام خوازمية سرب القطط من اجل تخطيط مسار للعديد من الروبوت الناقل في بيئه متحركه

ثمار علي جليل*، علياء كريم عبد الحسن
قسم علوم الحاسوب، الجامعة التكنولوجية، بغداد، العراق

الخلاصة

في هذه الورقة، اقترحنا منهجية تحكم الهجين باستخدام تحسين Artificial potential field مع تعديل خوارزمية سرب القطط لتخطيط مسار لعديد من الروبوت المتحرك الغير المزدوجه في بيئة ديناميكية. تتكون الطريقة المقترحة من مرحلتين: في المرحلة الأولى، يتم استخدام طريقه Artificial potential field(APF) لتوليد مسار لكل واحد من الروبوتات وتجنب العقبات ثابتة في البيئة، وتحسن هذه الطريقة في حل الحد الأدنى من مشكلة محلية باستخدام خوازميه A* مع منحنياتB–Spline بينما في المرحلة الثانية، يستخدم تعديل خوازمية سراب القطط (CSA) للسيطرة على الاصطدام الذي يحدث بين الروبوتات أو بين الروبوت مع العقبات المتحركة عن طريق استخدام وضعين من السلوك: وضع السعي ووضع التتبع. وقد اظهرت النتائج نجاح الطريقة المقترحة لإيجاد مسار كامل، والأمثل وخالي من التصادم لجميع الروبوت.

الكلمات الدالة: تخطيط الغير مزدوج، طريقه Artificial potential field، خوازميه A*، منحنيات B-Spline ، خوارزميه سرب القطط

*Email: hassanalia2000@yahoo.com

I. Introduction

In many applications, for example baggage treating systems at airports, stock-pilling systems in manufactory and moving containers in pots, where the path planning to multiple mobile robots has been widely scrutinized because of its possibility important in these applications. The path planning to these mobile robots should travel from initial location to the goal location with shortest path, optimal path, and avoid any collision that may occur in their walk, for example, collision with walls or people or another robot [1].

There are two ways for generating path of a group of robots in environment containing static and dynamic obstacles: coupled planning method and decoupled planning method. In coupled planning method, dealing with all the robots as a single system. This method provides optimal and complete path for all robots, but time consuming [2-4]. In decoupled planning method, each one of the robots in the environment that consider one independent system from the rest of the robots located within the environment. It means that, path is generated for each one of the robots independently. The advantage of this method is that, it takes less execution time upon the first method, but it has some problems such as a collision between robotics and also deadlock problem [2-4]. Therefore, to overcome this drawback, another method is used after planning such as: path coordination [5,6], priority planning [5,6], fuzzy system[7], particle swarm optimization [8], ant colony optimization [9] and fish swarm optimization [10].

Cat swarm algorithm (CSA) is one of the fields in swarm algorithm. It is a new algorithm that fosters a based learning rule on the conduct of cats. CSA consists of two behaviour modes: seek mode and track mode. The two behaviours are depended on the mixture ratio value, where the mixture ratio defines the ratio of the number of tracing mode cats to the number of seeking mode cats. Cat swarm algorithm (CSA) achieved better performance than PSO. Similar to PSO, CSO also belongs to the swarm intelligence based on population. The cat corresponding to a particle used in PSO is used as an agent. The behaviour of a cat is modelled to solve the optimization problem with employing a new learning rule [11]. In this paper, CSA uses two behaviours to control collision that may occur between robot and object in environment.

In most of the problem solving approaches there are measures that specify the quality and goodness of the approach. In robot path planning, these measures as defined by [12] are:

1. Completeness: the planner should find the solution (path from start to goal) if there is one, in some situations the planner can't guarantee to find the solution even if it exists; this is due to some problems like dead-lock.
2. Optimality: the solution or the path that is found should be the shortest between all the potential solutions exists.
3. Uncertainty: in some situations the robot may have little or no information about the environment or its work space, so how the planner can deal with such situation to find the path.

The remainder of this paper is organized as follows: Section II gives an overview of previous publications covering multi-robot using Artificial Potential Field (APF) method for path planning. Section III discusses the proposed method. Section IV discusses experimental result and discussions. A finally, some conclusions are given in Section V.

II. Related work

Many researchers suggested several proposals to solve collision problems in multi robot system in two dimensional environments cluttered with static and dynamic obstacles. They were focused their effort to improve the proposed work on the following measured: *completeness, optimality, and uncertainty path.* In 2004, Leng-Feng Lee proposed method to decentralize multi robot system in static environment, the navigation potential field function is used as path planning to group of robots and used the geometric shape to communication between these robots [13]. In 2013, Muhammad Abdullah proposed method to decupled multi robot system, where the potential field method is used to generate path to all robots. This method is improved by using fuzzy rules to solve local minimum problem. Then, the traffic rules were used to deal with situation when two robots are passageway each other. Finally, Market Based Optimization (MBO) is used to weaken or strengthen repulsive function generated by other robots based on their importance [14].

In this paper, many attempts have been made to achieve the three measures above. This is through proposing a new method, for finding a complete path to each one of robots based on improved

potential function to plan a free collision path between the initial position and the target position, while collision is avoided in the environment by using Cat Swarm Algorithm (CSA).

III. Proposed Method

The proposed method for multi robot motion planning in two dimension environment cluttered with static and dynamic obstacles consists of two phases: offline and online phase. The combination of these two phases represents the whole process of the proposed method.

1. Offline Phase

The offline phase is responsible for: prepared work space, Artificial Potential Field (APF) creation, APF path generation and improved APF using A* algorithm. The overall processes in the phase are described in algorithm-1, where all equations in this section are from [15, 16].

A. Prepared Work Space

The first step in algorithm-1 represents the prepared work space to mobile robot, where the environment is mapped to spherical work space [17]. Spherical work space is allowed to create a complete classical environment and also allow the environment and each obstacles inside it that take the one shape is circle shape, this lead to accurate and speed computation to both of the two functions: attractive and repulsive over the environment. To map the environment to spherical work space: firstly, the environment is put inside the large circle by giving the center point and radius. Secondly, each one of static obstacles inside the sphere space is assigned one small circle surround it. Therefore, all obstacles inside environment are represented by the center point and radius value of one circle that covers its space. Figure-1 show 2D environments with static obstacles (obs.) mapping spherical work space.

B. Artificial Potential Field (APF) Creation

The second step in algorithm-1, the Artificial Potential Field (APF) is created. APF will constructed by using two functions: attractive potential (equation- 1) and repulsive potential function (equation-2). By applying equation-1 on each point in the map, where each cell in the map will have a vector that helped to the gradient to the target.

$$\bigcup_{att}(q) = \begin{cases} \frac{1}{2}\zeta\, d^2(q, q_{goal}), & d(q, q_{goal}) \le d^*_{goal} \\ d^*_{goal}\,\zeta\, d(q, q_{goal}) - \frac{1}{2}\zeta\,(d^*_{goal})^2, & d(q, q_{goal}) > d^*_{goal} \end{cases} \quad (1)$$

Where ζ the attractive gradient scalar, q is cell in grid, q_{goal} is the goal point position in grid, $d^2(q,q_{goal})$ is Euclidean distance between each cell in grid to the goal point, d^*_{goal} is the threshold value that defines the distance between each cell in grid and the goal which will be compared for the choice between quadratic potential and conic potential. By applying equation-2 on each cell surroundings obstacle in the map, where each cell in the map will have vectored that help to the gradient away from obstacles.

$$U_{rep_i}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{d_i(q)} - \frac{1}{Q^*_i}\right)^2, & \text{if } d_i(q) \le Q^*_i, \\ 0, & \text{if } d_i(q) > Q^*_i, \end{cases} \quad (2)$$

where

$$d_i(q) = \min_{c \in \mathcal{Q}\mathcal{O}_i} d(q, c). \quad (3)$$

Where η a repulsive gradient scalar, q is a cell in grid, $d_i(q)$ is Euclidean distance to the closest point on obstacles, c is the obstacles, Q^* is a threshold that defined the influence range of the obstacle. The total potential field is obtained by adding the repulsive function resulting from all obstacles in the workspace and the attractive function from the target by using equation-4, this lead to construct (APF).

$$U_{total}(q) = U_{att}(q) + \sum_i U_{rep_i}(q) \quad (4)$$

Where *i* is the number of obstacles in work space. Figure-2 depicts APF creation over spherical workspace in figure1 with 12 starts point S, 12 goals point G, the scale factor for attractive ς=0.5, the threshold for attractive d^*_{goal} =15, the scale factor for repulsive η=1, and the threshold for repulsive Q* =1.

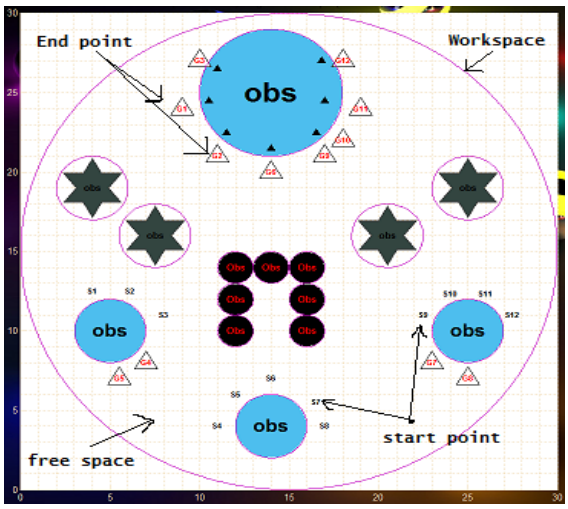


Figure 1- Show 2D environments with static obstacles mapping spherical work space

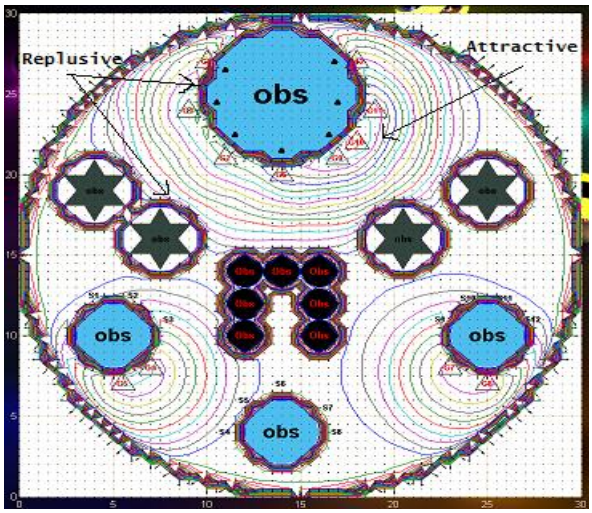


Figure 2- APF creation

C. APF Path Generated

In previous step compute both attractive and repulsive function for all workspace where each cell in a grid has vector force. The vector force has magnitude (m) and direction (d). At step5, in algorithm-1, the path finding for group of robots will found using the APF path generator $Path_{APF}$. Now, we need to generate a path for each one of mobile robot from the start point q(x,y) to the goal point $q_{goal}(x,y)$ and avoid any collision occurs with the static obstacles inside the workspace. Equation-5 is represented gradient attractive force. It is used to make the mobile robot to find the path from initial position to the target position.

$$\nabla \bigcup_{att}(q) = \begin{cases} \zeta\,(q - q_{goal}), & d(q, q_{goal}) \le d^{*}_{goal} \\ \dfrac{d^{*}_{goal}\,\zeta(q - q_{goal})}{d(q, q_{goal})}, & d(q, q_{goal}) > d^{*}_{goal} \end{cases} \quad (5)$$

Where q is location of the robot in workspace, the result of $\nabla U_{att}(q)$ decreased until equal to zero when the robot arrived to the goal. Equation-6 represents gradient repulsive force. It is used to make the robot to find a path away from the boundary of obstacles.

$$\nabla \bigcup_{rep}(q) = \begin{cases} \eta \left(\dfrac{1}{Q^{*}} - \dfrac{1}{d_i(q)}\right)^{2} \dfrac{1}{d_i^{2}(q)}\ \nabla d_i(q), & d_i(q) \le Q^{*} \\ 0, & d_i(q) > Q^{*} \end{cases} \quad (6)$$

where

$$\nabla d_i(q) = \frac{q - c}{d(q, c)}. \quad (7)$$

After that, need to compute the total negative gradient artificial potential field is obtained by adding the negative gradient repulsive potential function $\nabla U_{rep}(q)$ resulting from all obstacles in the workspace and the negative gradient attractive potential function $\nabla U_{att}(q)$ from the target by equation-8. Figure-3 depicts the path generation to 12 robots and explain local minimum situation that occurs to the robots.

$$\nabla \bigcup_{total} q = -\nabla \bigcup_{att}(q) - \sum_{i} \nabla \bigcup_{rep}(q) \quad (8)$$

D. Improved APF Using A* algorithm

The APF path planning method have local minimum problem [16], such as: when robot cannot passage between two closely obstacles, when the goal of the robot is located the effect the scope of repulsive function, when robot, obstacles, and goal are aligned, and when the workspace have U-shape.

In this phase, improved the APF by using A* algorithm with B-Spline used for finding a complete path for each one of robots in the environment. So, at step6, the generated path at previous step will be tracked. In step 7, path of mobile robot is tested, if the first point of the generated path is the initial point S and the final point of the planned path is the goal point G, if so the return path as the final planned path $Path_i = Path_{APF}$ and go to step 8 . Else, this means that the robot didn't reach the goal; here the last point of the $Path_{APF}$ will be saved as start position (SS) and A* algorithm[18] worked start and generate path from stop position of robot (SS) position in $Path_{APF}$ to goal position (G) and the generated path as $Path_{A*}$ and this path pass the B-Spline Curve [19] for making a smooth $Path_{smooth}$, because the path generated from A* was not smoothed. Finally, the path generation is $Path_i = Path_{APF} + Path_{smooth}$ and go to step 8. Repeated the steps from 4 to 9 until generate paths to all robots and this paths is returned in step 10. Figure4 depicts Path generation using an improved APF to 12 robots.

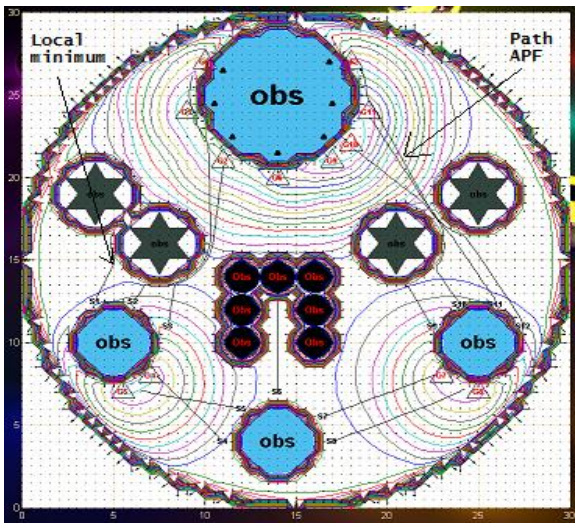


Figure 3- Path generation using APF to 12 robots, with local minimum

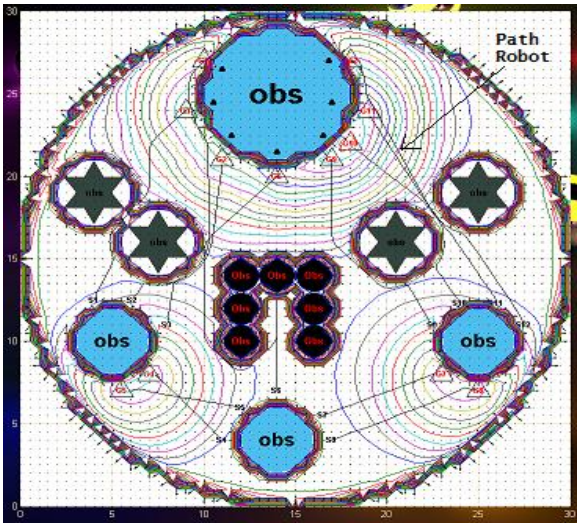


Figure 4- Path generation using Improved APF

Algorithm-1: Proposed Algorithm to Generate a Complete Path

Input: Two dimension work space, robots start position S, robots goal position G and obstacles position.
Output: Planned path each one of robot if found.

Step 1: Prepared work space
Step 2: APF creation
Step 3: i← 1; k← no.of robots
Step 4: Repeat Step 4 thought 9 until i=k
Step 5: Find path from S_i to G_i using APF path generation
$Path_{APFi} = APF(S_i, G_i)$
Step 6: Track the paths until it reaches last point in the path.
Step 7: Check the initial point and last point in the $Path_{APF}$.
If Initial point is the start point (S_i) and last point is the goal point (G_i)
Then $Path_i = Path_{APFi}$ **,** go to **Step 8**
Else robot don't reach the goal, save the last point (SS_i), and A* algorithm begin worked and generate path from stop position of robot (SS_i) position in $Path_{APFi}$ to goal position (G_i).
$Path_{A*i} = A*(SS_i, G_i)$;
$Path_{smoothi} = B\text{-}Spline(Path_{A*i})$;
$Path_i = Path_{APFi} + Path_{smoothi}$;
End
Step 8: $Final_{pathi} = Path_i$
Step 9: i← i+1;
Step 10: return ($Final_{path}$)
Step 11: End

2. Online phase

In this phase, planner is responsible for avoiding collisions that occurs between robots, collision between robots and moveable obstacles, and also solved all deadlock. So, this phase consists of three sections: firstly, a dynamic obstacle is added then the collision is detected. Finally, the collision is avoided by using cat swarm algorithm.

A. Moving obstacles

In this work, the moving obstacles represented as a block with its reference point in the centres of the block in the free part of the free space (see Figure-5). The moving obstacles move in eight directions and not change its direction when hits a robot, the obstacle will move continuously in its direction until it hits an edge of the configuration space or another obstacle then it changes its direction. And by assuming, each movable obstacle has flag equalled to 'o'. In section B and section C, the planner is responsible to change the direction of the robot when any collision might occur with another robot or move obstacle.

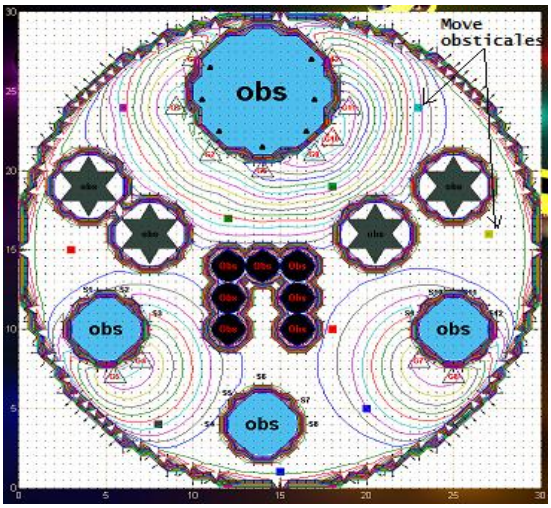


Figure 5- Moveable obstacles

B. Collision Detection

Algorithm-2 and algorithm-3 describe the proposed collision detection process. In this work, it is assumed there are a number of sensors that distributed in some way which make the robot able to cover all direction. But in the simulation design, the planner feed the collision detection algorithm with a block resides in all direction of the robot to check if a collision occurs. This block consists of sectors that represent the distance of the possible collide object to the robot. Where the distance can be classified as Short, Medium, Long, or Very Long depending on which sector the object located. At this moment, in each step move for each robot in the configuration space the planner will run the collision detection algorithm to check the block in all direction of the robot and if it finds an object it will return its distance from the robot. When we saying an object it means that it can be either movable obstacle or another robot. The values of each variable distance between robot and objects are computed according to (equation -9 [15]):

$$d = \sqrt{(obj_x - rob_x)^2 + (obj_y - rob_y)^2} \quad (9)$$

Algorithm-2: Proposed Collision Detection Algorithm

Input: Current robot position $R_c(x, y)$, next robot position $R_n(x, y)$, current objects position C-Object (x, y), next objects position N-Object (x, y), object short distance O-short-d, object medium distance O-medium-d **,** object long distance O-long-d, object very distance O-very-d.
Output: Collision flag (if found true else false).

Step 1: m=size(C-Object), n=size(C-Object), a=1, b=1, c=1,d=1;
Step 2: Classify distance algorithm (R_c (x, y), C-Object, m, a, b, c, d, O-short-d, O-medium-d **,** O-long-d, O-very-d)
Step 3: Classify distance algorithm (R_c (x, y), N-Object, n, a, b, c, d, O-short-d, O-medium-d **,** O-long-d, O-very-d)
Step 4: Classify distance algorithm (R_n (x, y), C-Object, m, a, b, c, d, O-short-d, O-medium-d **,** O-long-d, O-very-d)
Step 5: Classify distance algorithm (R_n (x, y), N-Object, n, a, b, c, d, O-short-d, O-medium-d **,** O-long-d, O-very-d)
Step 6: If O-short-d is not empty **and** O-medium-d is not empty
Then collision=1
Else collision=0
End
Step 7: Return (collision)
Step 8: End

Algorithm-3: Proposed Classify Distance Algorithm

Input: R_{move}, object, $size_{object}$, a, b, c, d, O-short-d, O-medium-d**,** O-long-d, O-very-d.
Output: Classify distance

Step 1: i=1;
Step 2: Repeat Step 2 thought 4 until i= $size_{object}$
Step 3: D(i)= Distance (R_{move}, Object (i,:)) // by using equation (9)
If D (i) >0 **and** D(i) <=2 **then** O-short-d (a, :) = Object (i, :); a=a+1;
Else If D(i)>1 **and** D(i)<=3 **then** O-medium-d(b,:)= Object(i,:); b=b+1;
Else If D (i) >2 **and** D(i) <=4 **then** O-long-d (c, :)= Object (i, :); c=c+1;
Else If D (i) >3 **then** O-very-d (d, :) = Object (i, :); d=d+1;
End
Step 4: i←i+1
Step 5: Return (a, b, c, d, O-short-d, O-medium-d**,** O-long-d, O-very-d)
Step 6: End

C. Collision avoidance

This step is worked when the collision detection algorithm return collision equal to one. Collision avoidance algorithm is used for avoiding the collision that occurs between robot and objects. The whole process of proposed collision avoidance is explained in algorithm-4. In algorithm-4, the modify cat swarm algorithm is utilized to avoid collision occurs.

Firstly, the planner must determine the flag for each one of robots in the workspace. The flag is determined depending on the mixture ratio value. The planner is depending on the mixture ratio to determine the number of robots in the seeking mode or in the tracking mode by using equation-10 and equation-11:

$$\text{No of robots}_{\text{seek mode}} = \text{round}\,(\text{mixture ratio} * \text{No of robots}) \quad (10)$$

$$\text{No of robots}_{\text{tack mode}} = \text{No of robot} - \text{No of robots}_{\text{seek mode}} \quad (11)$$

Secondly, the planner is computed the path length to each one of robots from current position to goal position. The largest length is assigned to the seek mode by giving flag equal to one while the smallest length is assigned to the tracking mode by giving flag equal to zero. If the all robot have the same length, in this case randomly select the behaviour of robots. If MR=0.4 and number of robot=12 then No.of $robots_{seek\ mode}$ = 5 then five robots that have the large length is assigned to seek mode while

No.of $robots_{track\ mode}$ = 7 then seven robots that have the small length is assigned to track mode. Now, the planner is checked the behaviour of robots if is it a seek mode or track depend on the value of flag.

When flag equals to one, the seek mode process is started the work. This mode has three main parameters which are as follow: The Seeking memory pool (SMP) is the number of copies robots that spread in the workspace, where each robot has eight directions this to lead SMP=8. The seeking range of the selected dimension (SRD) is the distance between the original location of robot and the location of robots copies, SDR is 1, 0, -1 depending on direction of robots, and the counts of dimension to change (CDC) is the number of dimensions to be mutated where proposed method is worked in 2D environment. Figure 6 shows the seek mode main parameters.

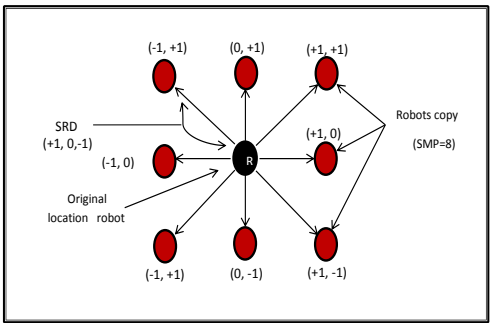


Figure 6- The seek mode parameters

Each one of eight copies has fitness value, and direction. Now, all copies are put in list and each one of copies must check. Firstly, if the copy is spread on the static obstacle. In this case, the copy is collided with the static obstacles and this copy is deleted. Secondly, the distance between each one of copy and objects (robot or movable obstacles) are checked, if the distance is the less than one. In this case, this copy is deleted. After, the list of copies is checked; if list is empty that means robot is waiting in location through this iteration otherwise the fitness value is computed for all the reset copies.

The planner is needed to plan path from each one of location copies to specific goal to original robot, after that, the planner is computed the fitness value by using equation-12:

$$Fitness_v = min\left(Length\left(path_{copy-I}\right)\right) \quad where\ I = 1,2,\ldots,no.of\ copies \quad (12)$$

Length is the number of point in $path_{copy_I}$ from each one of copies location to the target location. Then, the small fitness value is selected. Then, the copy has bested fitness value (best fitness value means the copy has been the small fitness value) is picked, and take as the next of robot location.

If fitness values are equal in all of copies. In this case, the planner is used the equation-13 to determine the best fitness value because the planner is needed to choose the copy have the small number of gradient and also makes the robot is moved forward but not backward.

$$F_v = min\left(\frac{D(path_{copy-I})}{L(path_{copy-I})}\right) \quad where\ I = 1,2,\ldots,no.of\ copies \quad (13)$$

$$where \quad D(path_{copy-I}) = \sum_{z=1}^{N*N} d\left(p_{ij}\right) \quad where\ N = no.of\ point\ in\ path_{copy-I} \quad (14)$$

$$where \quad d\left(p_{ij}\right) = \begin{cases} if\ i > j, & p_{ij} = \sqrt{\sum_{k=1}^{dim} \|x_{ik} - x_{jk}\|^2} \\ otherwised, & p_{ij} = 0 \end{cases} \quad (15)$$

Where $D(path_{copy\text{-}I})$ is the summation of computes the Euclidean distance between pairs of points in *m*-by-*n* data path, $path_{copy\text{-}I}$ is the number of coordination point $[p_1,p_2,\ldots\ldots p_N]$ that represents the location robot from each one of copies to goal, and $L(path_{copy\text{-}I})$ is number of point in the $path_{copy\text{-}I}$, $d(p_{ii})$ is the computes the Euclidean distance between pairs of points in *m*-by-*n* data path, dim is the dimensionality of data path, and x_{ik}, x_{jk} are, respectively, the k^{th} components of the i^{th} and j^{th} paths, x_i,x_j. The planner is changed the robot direction and moved to the next location (copy location that has the best fitness value (F_v)). After, the planner is replanning path to the robot and update path to the next iteration and then, re-computing the behaviour of robots in the workspace.

When flag equals to zero, the track mode process is started the work. In this mode, the planner is made the robot either wait in the location through this iteration when the distance to object is the shortest distance or move to the next location in the path, in this case, the next location in the path have been best fitness function and each new position and new velocity used the equation-16 and equation-17:

$$V_{x/d} = \text{Same the velocity gradient} \quad (16)$$

$$X_{k/d} = \text{Next location in the path in this iteration} \quad (17)$$

Algorithm-4: Proposed Collision Avoidance Algorithm

Input: Number of robot R(x, y), Current position of robot R_{move} (x,y), Goals position G_{Rmove}, O short-d, O-medium-d, mixture ratio value.

Output: New location for robot to avoid collision.

Step 1: Determine the number of robots in the seek- mode or in track- mode depending on the mixture ratio value by applying equation (10) and equation (11)

Step 2: Flag is assigned to each one of robots depending on **step 1**

Step 3: Determine the flag of R_{move} (x,y)

Step 4: Cat swarm algorithm begins to work, check the flag of robot

If flag=1 **then** go to **step 6**

Else if flag=0 then go to **step 5**

End

Step 5: Check the list of O-median-d and O-short-d

If one of object has the flag equal to 'o' **Then** flag=1 **then** go to **Step 6**

Else flag=0 **then** go to **Step 7**

End

Step 6: Seek-Mode behavior begins to work

6. 1: Make j copies of the R(x,y), where j = SMP

If the value of SPC is true, Let j = (SMP −1),**Then** return the present position as one of the candidates

End

6. 2: For each copy, according to CDC, randomly plus or minus SRD percent's the present values and replace the old ones.

6. 3: Each one of copies are checked

IF copies are spread on static obstacles **then** delete these copies **End**

IF 0>distance between ((copies and O-short-d)or (copies and O-medium-d)) <1 **Then** delete these copies **End**

6. 4: Check the list of copies

IF list of copies is empty **then** the robot is waited in the location and don't move to the next location in the path

Else compute fitness function to each one of copies using equation (12)

End

6. 5: If all fitness functions is equal to all copies **Then** equation(13) is used to choose the copy that have the best fitness function

6. 6: Robot moves to the copies that have the best fitness function and replanning path to robot and Go to **step 2.**

Step 7: Track-mode behavior begins to work, compute distance between

R(x, y) and O-medium-d and also distance between

R(x, y) and short-d by using equation (9)

If distance < 1 **then** the robot is waited in the location go to **step 2**

Else Robot move the next location in the path by apply the equation(16) and Equation (17) and go to **step 2**

End

Step 8:End

IV. Experimental Results and Discussions

The proposed method is implemented using Matlab R2010b version, on a computer (HP) with Intel I Core™ i5-2430M CPU @ 2.40GHz. The environment size (30*30), workspace is a circle (center (15, 15), radius 15), robot is point q(x, y). The parameters setting to artificial potential field: The positive scaling factor ς of attractive field is 0.5. In order to make the path of robot far away from obstacles, we set the positive coefficient d^*_{goal} is 15. The coefficient η of repulsive field is 1. The largest impact distance of obstacles Q* is 1. The parameters setting to cat swarm algorithm: SMP is 8 because robot have eight direction, SRD is (-1, 0, 1) the distance between original robot location and copies location, and MR is 0.4 because when environment have two robot the set of MR is 0.4 make at least one robot is seek mode and another is a track mode.

Table -1 lists the starting and goal points of the 12 robots in Figure-4. It can be noticed that, the total execution time to path generation for 12 robots in same environment is 0.935330 sec.

Table 1- Execution time for 12 robots in offline phase

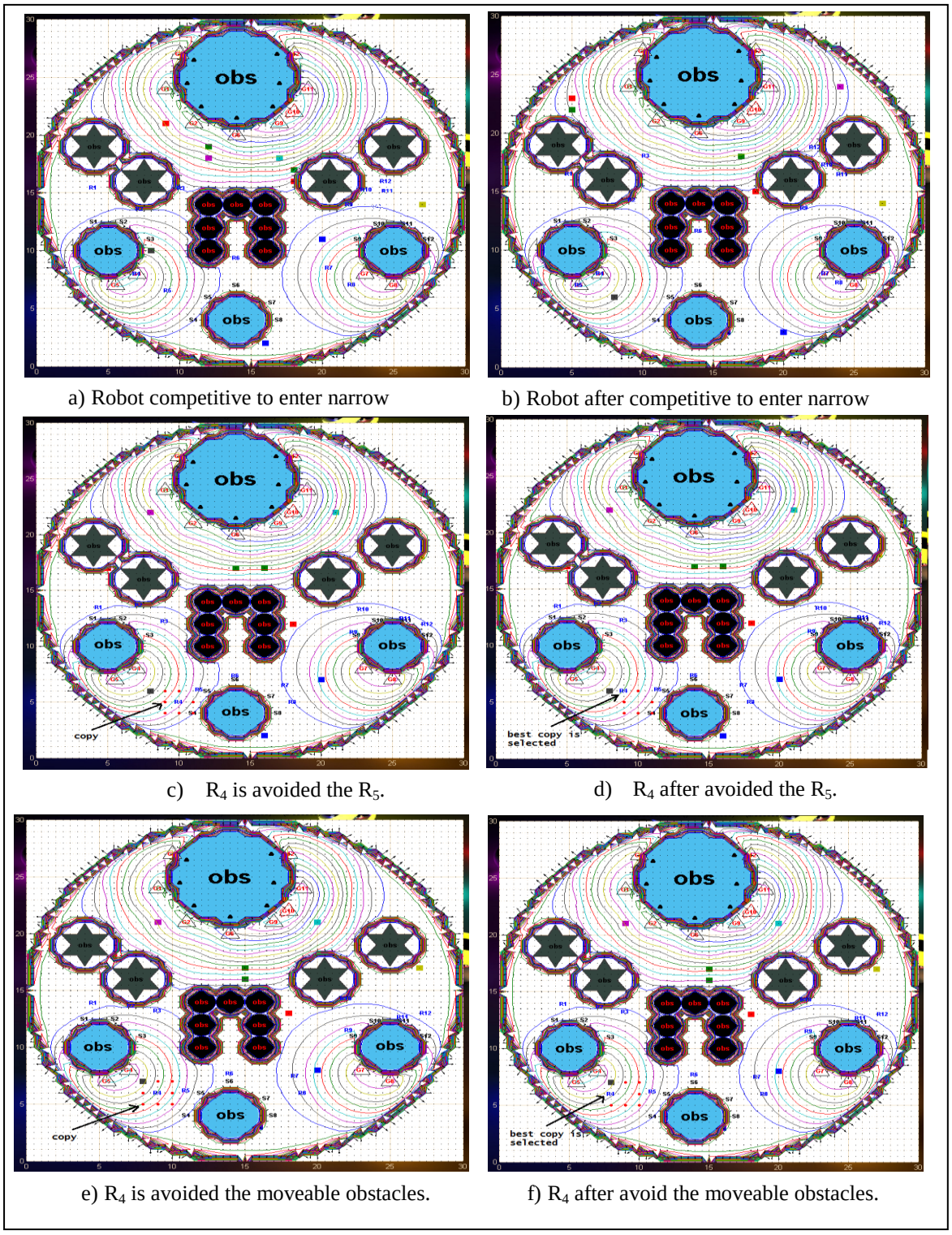
Robot No.	Start Point	Goal Point	Path Length before Path Following (grid) (offline phase)	Path Length after Path Following (grid) (online phase)
R1	[4,12.5]	[9,24]	82	83
R2	[6.1,12.5]	[11,21]	38	39
R3	[8,11]	[10,27]	58	60
R4	[11,4]	[7,8]	6	6
R5	[12,6]	[5.5,7]	15	15
R6	[14,7]	[14,20]	65	65
R7	[16.5,5.5]	[23,8]	11	10
R8	[17,4]	[25,7]	13	13
R9	[22.5,11]	[17,21]	18	20
R10	[24,12.3]	[18,22]	29	31
R11	[26,12.3]	[19,29]	70	74
R12	[27.5,11]	[18,27]	64	66
			Total Elapsed time (offline phase)= 0.935330 sec	

Also, Table-1 show the list paths length before path follows and the path lengths after the path following. As it can be seen, there is some difference in path lengths for robots, where R1, R2, R3 and R9 the path lengths is increased, this increased in path length is caused by seek mode when it is used as behavior in the collision avoidance with the moving obstacles or another robots, while the path length is increased in R10, R11 and R12 because R10, R11 and R12 are competing to entry between narrow passages, see Figure-7.a, 7.b.

Although, the R4 is the seek mode behavior and avoid the R5 is the track mode behavior (see Figure-7.c and 7.d) and also avoid the movable obstacle (see Figure-7.e and 7.f), but R4 is preserved the length of the path because R4 is always selected the forward copy. In the same time, R7 is seek mode and it is avoided the move obstacle but the length of the path is decreased because R7 is select the copy that led to forward and this copy have the short distance to goal.

So, we can notice, the cat swarm in the seek mode behavior is gave either the length path is kept the same path length or the path length is decreased or path length is little increased while the cat swarm in the track mode behavior is preserved the path length such as R5, R6 and R8 because the R5, R6 and R8 are kept the behavior through path following.

The execution time needed to seek mode process is 1.3082 sec. The time is used to generate the 8-copy, the fitness function is computed, and the best copy is selected. The time is decreased when the number of copies is generated the less than 8. Finally, table-1 shows when all robots were reached their goals without collision with each other or with static and moving obstacles. The proposed multi robot path planning in dynamic environment using cat swarm is a collision free and dead lock free algorithm.



a) Robot competitive to enter narrow

b) Robot after competitive to enter narrow

c) R_4 is avoided the R_5.

d) R_4 after avoided the R_5.

e) R_4 is avoided the moveable obstacles.

f) R_4 after avoid the moveable obstacles.

Figure 7- cat swarm algorithm behavior to avoid collision

V. Conclusion

In this paper, we have developed an algorithm for stable path planning to decouple multi mobile robot are moving in two dimensional environment (which is cluttered with static and dynamic obstacles). The simulation results show that our algorithm converges in such a manner that multi mobile robot can move easily in narrow corridors with avoid any collision with object (another robot or movable obstacles), dead lock free, and also guarantee to reach a target if path is found. So, Improved APF by using A* with B-Spline are used to find path for each one of the robots in environment and CSA are used to control the collision and deadlock occurrence. This proposed is

guaranteed complete, free collision, free deadlock, smooth and near to optimal. As a future work, the proposed algorithm can be improved by extending the work to three-dimensional environment.

References

1. Karim, A. and Shwail, S.H. **2014**. Probabilistic roadmap, A*, and GA for proposed decoupled multi-robot path planning, *Iraqi Journal of Applied Physics*, 10(2), pp:3-9.
2. Velagapudi, P., Sycara, K. and Scerri, P. **2010.** Decentralized prioritized planning in large multirobot teams, IROS, Taipei, Taiwan, October 18-22.
3. Latombe, J. C. **1991**. *Robot Motion Planning*, second addition, Kluwer Academic Publishers. Norwell, Massachusetts, USA.
4. Schwartz, J. T. and Sharir, M. **1983**. on the piano movers' problem: Iii. Coordinating the motion of several independent bodies: the special case of circular bodies moving amidst polygonal barriers, *The International Journal of Robotics Research*, 2(3), pp: 46–75.
5. Berg, J. P. and Overmars, M. H. **2005**. Prioritized motion planning for multiple robots, Taipei, RSJ International Conference on Intelligent Robots and Systems, August 2-3.
6. Bennewitz, M., Burgard, W. and Thrun, S. **2002**. Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots path planning techniques for teams of mobile robots, *Robotics and Autonomous Systems*, 41(2), pp:89–99.
7. Ching Lu, H. and Ying Chuang C. **2005**. The Implementation of Fuzzy-Based Path Planning for Car-Like Mobile Robot, *Taipei , MEMS, NANO and Smart Systems*, July 24-27.
8. Eberhart, R. and Kennedy, J. **1995**. A new optimizer using particle swarm theory, IEEE, MHS '95, Oct. 4-6, Nagoya.
9. Reshamwala, A. and Vinchurkar, D.P. **2013**. Robot Path Planning using An Ant Colony Optimization Approach: A Survey, *International Journal of Advanced Research in Artificial Intelligence*, 2(3), pp: 65-71.
10. Karim, A. **2014**. Path Planning Method for Single Mobile Robot in Dynamic Environment Based on Artificial Fish Swarm Algorithm, *Engendering and Technology Journal*, 32(2), pp: 251-257.
11. So, J. and Jenkins, W. K. **2013**. Comparison of cat swarm optimization with particle swarm optimization for IIR system identification, proceedings of the forty-seventh annual asilomar conference on signals, systems, and computers, Nov. 3-6, pacific grove, CA, pp: 903-910.
12. Vishnu R. Desaraju and Jonathan P. How. **2012**. Decentralized path planning for multi-agent teams with complex constraints**,** *Autonomous Robots*, 32(4), pp: 385-403.
13. Feng L. L. **2004**. Decentralized Motion Planning Within An Artificial Potential Framework (Apf) For Cooperative Payload Transport By Multi-Robot Collectives, M.Sc. Thesis, Department of Mechanical and Aerospace Engineering, State University of New York, New York.
14. Abdullah, M.**2013**.Mobile Robot Navigation using potential fields and market based optimization, M.Sc. Thesis, Department of Technology, Örebro University.
15. Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L. and Thrun, S. **2005**. *Principles of robot motion: Theory, Algorithms, and Implementation*, A Bradford Book, The MIT Press, Cambridge, England.
16. Montiel R. O. and Sepúlveda C. R. **2014**. *High Performance Programming for Soft Computing*, A Science Publishers Book, CRC press.
17. Koditchek, D. E. and Rimon, E. **1990.** Robot navigation functions on manifolds with boundary, *Advances in Applied Mathematics*, 11, pp: 412-442.
18. Nandkishor J. H. and Shinde, J. P. **2014**. An Image Based Path Planning Using A – Star Algorithm, *International Journal of Emerging Research in Management &Technology*, 3(5).
19. Shan, E., Dai, B., Song, J. and Sun, Z. **2009**. An Dynamic RRT Path Planning Algorithm Based On B-Spline, Second International Symposium on Computational Intelligence and Design, Dec.12-14, Changsha, Hunan, China, 2.