Navigational Path Planning of Multi-Robot using Honey Bee Mating Optimization Algorithm (HBMO)

Rashmi Ranjan Sahoo
Department of ETCE
Jadavpur University
Kolkata-700032, India

Pratyusha Rakshit
Department of ETCE
Jadavpur University
Kolkata-700032, India

Md Taslim Haidar
Department of EE
Jadavpur University
Kolkata-700032, India

Sujata Swarnalipi
Department of ETCE
Jadavpur University
Kolkata-700032, India

Bunil k. Balabantaray
Department of CSE
SYIT, BBSR-752101, India

Sharmilla Mohapatra
Department of CSE
VSSUT, Burla-768018, India

ABSTRACT
Over the last decade, evolutionary and meta-heuristic algorithms have been extensively developed and used as search and optimization tools in various problem domains, including science, commerce, and engineering. Their broad applicability, ease of use, and global perspective may be considered as the primary reason for their success. The honey-bees mating process may also be considered as a typical swarm-based approach to optimization, in which the search algorithm is inspired by the process of real honey-bees mating. In this paper we present an alternative approach for navigational path plan of multi robot using HBMO algorithm. We reveal that this proposed optimization scheme outperforms other Evolutionary algorithms like Particle swarm optimization, Differential Evolutionary algorithm in the task of navigation.

Keywords
Multi Robot Path Planning, Honey Bee Mating Optimization algorithm, Centralized Planning.

1. INTRODUCTION
Robot navigational path planning has been cited as a vital promenade of research in the field of Robotics. There are numerous scenarios in which large groups of robots are required to navigate around shared environment. Some specific examples in this era could be delivery robots in an office, a warehouse, a shipping yard, or a mines, or even virtual armies in a computer war game [1]. In all the cases, there are several robots with independent goals that must decussate the shared environment without colliding with other static or dynamic members present in its runway. Robot navigation has been accustomed to be solution to three fundamental questions: (1) Where am I? (2) Where are other places in affinity to me? (3) How can I locomote from one place to another? [2] In multi-robot scenario, with insinuation to the given world map, the path planning problem ascertains the trajectory motion of the robot from a given starting point to a given destination in addition it also avoids the collisions with obstacles as well as the other robots that comes along its runway. The basic path planning problem has several extensions and classifications as indicated below. One common classification of the problem includes *local* and *global* planning [3], [4]. In a local path planning a robot *navigates* through the world map with obstacles in steps and delineate it's next position towards the goal, satisfying one or more predefined constraints on path-, time-, energy-optimality [5], [6]. In global planning, the entire navigation path is planned by the robot prior to its movement towards the goal. This type of global planning is referred to as offline planning in literature [7]. Whereas Local path planning includes navigation and online planning, this is referred to as navigation only in literature. The phrase, 'motion planning' deals with the location of the robot on a planned trajectory in a given workspace. Motion planning thus takes care of planning the path with some resource management or constraints over time.

Over the last three decades, significant progress has been attained on single robot motion planning in mobile robotics. Fuzzy obstacle avoidance and motion planning algorithms , evolutionary algorithms and some classical approaches such as quad-tree [5],[6], graph based algorithms, heuristic algorithms such as real time A* [8], neural algorithm [9], are some of the well known techniques for the path planning as evident from the literature. In a Multi-robot path planning problem each robot has a predefined starting and destination locale in the given world map and the robots owe to plan their itinerary either locally or globally without hitting any of its teammates or obstacles that come across its runway and also attempt to minimize the traversal of the robots. The encumbrance that comes in the path of the robot can be stationary or dynamic. However this paper deals with stationary obstacles given in the world map for the robots. Robot path planning is part of a larger class of problems pertaining to scheduling and routing, and is known to be NP-hard (NP-complete) [10]. The path planning problem is known to be PSPACE hard [11]. This means that the complexity of the path planning problem increases exponentially with the dimension of the configuration space. The configuration space is the space of all complete specifications of the position of every point of a robot system.

The concept of swarm intelligence as an optimization technique is projected for finding collision free paths in work space containing differently shaped and distributed and centralized encumbrance. Therefore the problem of path planning is hence assumed to be an optimization technique, where the paths free from collisions receive higher fitness value in correspondence to

those problems resulting in collision with an obstacle path planning problem. Honey Bees Mating Optimization Algorithm (HBMO) can be accustomed to be a typical swarm based approach of optimization. The algorithm is stimulated by the behavior of social insects, which are characterized by three main features: cooperation among adults in brood care and nest construction, overlapping of at least two generations and reproductive division of labor.

In this paper, we realize the multi-agent motion-planning problem by the HBMO algorithm. Naturally, for the centralized approach we need to construct a fitness function for the HBMO to determine the next position of all the robots that lie on optimal trajectories leading towards respective goals. The fitness function has two main components: 1) the objective function describing the selection of next position on an optimal trajectory and 2) the constant representing collision avoidance with peers and static obstacles.

Rest of the article is organized as follows: in section 2 we describe the formulation of the multi-robot motion planning problem Section 3 describes structure of Honey Bees colony, as they are in nature and HBMO. The pseudo-code for solving the given constrained optimization function is scripted in Section 4. Experimental results and the computer simulation are depicted in Section 5. Finally we conclude with section 6 by comparing other swarm intelligence techniques like DE and PSO and further possibilities of improvement.

2. PROBLEM FORMULATION

The conceptualization speculates the evaluation of the next position of the robot in its workspace thereby avoiding collision with other robots and the static hindrances in its runway from the current locality of the robot in the workspace. The following are the presuppositions made to validate the multi-robot path planning problem:

- The current and the destination locale of each robot is known prior with respect to a given reference co-ordinate system.
- Among a fixed set of actions for motion the robot has to select only one action at a time.
- The path planning problem hence incurs a number of steps until all the robots reach their respective destination.

The following principles are used satisfying the assumptions.

- The robot first ascertains the next position in order to co-ordinate itself with the destination and constructs a path to that location.
- If more than one robot occupies the same next locale then this calibration may result in possible collision with its teammates. This collision may occur with a static obstacle as the position is occupied by a static hindrance. To avoid such collision a new next position is to be resolved for which the robot has to be rotated left or right by certain angle.
- The robot will move to the calculated next position if it can align itself towards its goal location without any collision.
- If turning left or right requires the same angle of rotation of the robot about z-axis, the tie towards goal is arbitrarily broken.

According to principle (1) each robot R_i first determines its next position towards its goal.

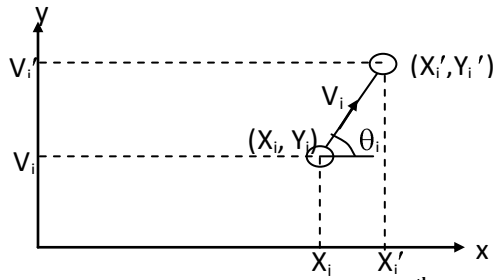

Fig 1: Current and next position of the i[th] robot

Let,

- At time instant t the current position of robot R_i is (x_i, y_i)
- At time instant $(t+1)$ the next position of robot R_i is (x_i', y_i')
- The goal position of the robot R_i (x_{ig}, y_{ig})
- The angle of rotation of robot R_i is θ_i to align itself towards its goal position
- The velocity of robot R_i is v_i

So from Fig. 1, we have

$$x_i' = x_i + v_i cos\theta_i \Delta t \quad (1)$$
$$y_i' = y_i + v_i sin\theta_i \Delta t \quad (2)$$

For Δt=1 sec, above equations are reduced to

$$x_i' = x_i + v_i cos\theta_i \quad (3)$$
$$y_i' = y_i + v_i sin\theta_i \quad (4)$$

According to principle (3), if the determined next position of robot R_i, (x_i', y_i') is not occupied by any other robot or static obstacle, R_i should move to (x_i', y_i') and then (x_i', y_i') will become its current position. According to principle (2) if determined (x_i', y_i') results in a collision, this (x_i', y_i') has to be abandoned and new (x_i', y_i') is calculated so that the line joining (x_i, y_i), (x_i', y_i') and (x_i', y_i'), (x_{ig}, y_{ig}) do not touch the static obstacle as shown in Fig. 2.

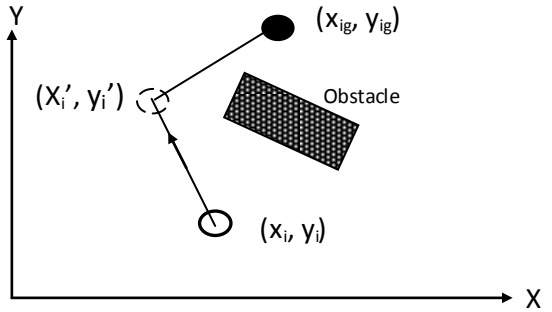

Fig 2: Selection of (x_i', y_i') from (x_i, y_i) to avoid collision

In order to reach the goal position (x_{ig}, y_{ig}) the parameters which must be taken care are given as follows.

Total distance traversed by robot R_i from current position (x_i, y_i) to next position (x_i', y_i') and from next position (x_i', y_i') to goal position (x_{ig}, y_{ig}), which is given as

$$Dist = \sqrt{((x_i - x_i^{'})^2 + (y_i - y_i^{'})^2)} + \sqrt{((x_i^{'} - x_{ig})^2 + (y_i^{'} - y_{ig})^2)} \quad (5)$$

Substituting the values of $x_i^{'}$ and $y_i^{'}$ from (3) and (4) we have for all *n* robots

$$Dist = \sum_{i=1}^{n} \{v_i + \sqrt{((x_i + v_i \cos\theta_i - x_{ig})^2 + (y_i + v_i \sin\theta_i - y_{ig})^2)}\} \quad (6)$$

Minimization of *Dist* confirms that the robots will follow the shortest paths. The distance between next position of robot R_i and all other teammates is given as d_{ij}. In order to avoid collision of i-th robot with j-th robot we have to consider the constraint $(d_{ij} - 2r) > 0$, where r is radius of the robot R_i.

Let the distance between next position of robot R_i and static obstacle is given as d_{i-obs}. The optimization problem here includes an objective function f, concerning minimization of Euclidian distance between the current positions of the robots with their respective goal positions, constrained by obstacles and teammates on the path. The objective function for the proposed optimization problem is given by

$$f = \sum_{i=1}^{n} \{v_i + \sqrt{((x_i + v_i \cos\theta_i - x_{ig})^2 + (y_i + v_i \sin\theta_i - y_{ig})^2)}\} + f_{dp} \sum_{i'j'=1}^{n(n-1)/2} (\min(0, (d_{i'j'} - 2r)))^2 + f_{st} / d_{i-obs} \quad (7)$$

Where f_{dp}(>0) and f_{st}(>0) are scale factors. In our experiments, we used f_{st}=5000 and f_{dp}=100.

3. HONEY BEE MATING OPTIMIZATION ALGORITHM

The honey bee is a social insect that can survive only as a member of a community, or colony. The colony inhabits an of different drone's sperm in her spermatheca, she can use parts of the honey bee community consists of three structurally different forms: The queen (reproductive female), the drones (male) and the workers (no reproductive female). These castes are associated with different functions in the colony; each caste possesses its own special instincts geared to the needs of the colony. The HBMO Algorithm combines a number of different steps and the main steps of HBMO are depicted in figure 3.

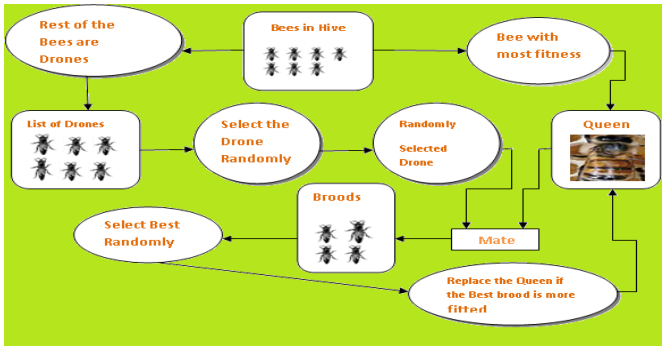

Fig 3: Steps of HBMO.

Each of them corresponds to a different phase of the mating process of the honey bee. A drone mates with a queen probabilistically using an annealing function as:

$$Prob(D) = \exp(-D(f)/S(t)) \quad (8)$$

where *Prob (D)* is the probability of adding the sperm of drone *D* to the spermatheca of the queen (that is, the probability of a successful mating), *D(f)* is the absolute difference between the fitness of *D* and the fitness of the queen (for complete description of the calculation of the fitness function see below) and *S(t)* is the speed of the queen at time *t*. The probability of mating is high when the queen is with the high speed level, or when the fitness of the drone is as good as the queen's. After each transition in space, the queen's speed decreases according to the following equations:

$$S(t+1) = \alpha . S(t) \quad (9)$$

$$E(t+1) = \gamma . E(t) \quad (10)$$

Where α and γ are a factors such that $\alpha, \gamma \in (0, 1)$ and are the amount of speed and energy reduction after each transition and each step respectively. Initially, the speed of the queen is generated at random. A number of mating flights are realized. At the start of a mating flight drones are generated randomly and the queen selects a drone using the probabilistic rule in Eq. (8). If the mating is successful (i.e., the drone passes the probabilistic decision rule), the drone's sperm is stored in the Queen's spermatheca. By using the crossover of the drone's and the queen's genotypes, a new brood (trial solution) is generated, which can be improved later by employing workers to conduct local search. In real life, the role of the workers is restricted to brood care and for this reason the workers are not separate members of the population and they are used as local search procedures in order to improve the broods produced by the mating flight of the queen. If the new brood is better than the current queen, it takes the place of the queen. If the brood fails to replace the queen, then in the next mating flight of the queen this brood will be one of the drones.

4. SOLVING CONSTRAINED OPTIMIZATION USING HBMO

In this section we propose a solution to the centralized version of the multi-robot motion planning problem using HBMO. The proposed scheme presumes current position of n-robots and their speed, and determines next position of each robot by optimizing the given constrained single-objective function. Here angles of rotation of n robots are considered to be parameters of each solution. An algorithm outlining the scheme is discussed below:

***Pseudo Code*:**

Input: Initial position (x_i, y_i), goal position (x_{ig}, y_{ig}) and velocity v_i for n robots where $1 \le i \le n$ and a threshold value ε.

Output: Trajectory of motion P_i for each robot R_i from (x_i, y_i) to $((x_{ig}, y_{ig})$

Begin

Set for all robot i

$x_{curr-i} \leftarrow x_i$; $y_{curr-i} \leftarrow y_i$;

For robot $i = 1$ to N

Repeat

Call **HBMO** (x_{curr-i} , y_{curr-i} , pos-vector)

// pos-vector denotes current position of all robots //

Move-to (x_{curr-i} , y_{curr-i});

Until $\|curr-i-G_i\| \leq \varepsilon$

// $curr-i$ = (x_{curr-i} , y_{curr-i}), G_i = (x_{ig} , y_{ig})//

End For;

End.

Procedure HBMO (x_{curr-i}, y_{curr-i}, pos-vector)

Begin

Initialize all the bees (initial population);

Initialize problem parameters as well as algorithm parameters like

- D←No. of Drones;
- B←No. of Broods;
- W←No. of worker bees;
- C←Capacity of Spermatheca;
- S_{max}←Maximum speed of Queen at starting of mating flight;
- S_{min}←Minimum speed of Queen;
- E_{max}←Maximum Energy of Queen at the starting of mating flight;
- γ←Energy reduction schema;
- α ←Speed reduction schema;

Evaluate the fitness (fit_i) of the population;

Set the bee with highest fitness as Queen and set $fit_Q = fit_{best\ bee}$;

For Iter=1 to Maxiter do

Begin

While $Speed > S_{min}$ *OR* ***Spermatheca*** **is *NOT full*** do

Begin

Select a drone depending on $Drone_i$ depending on *Prob* (D_i) as in equation (8);

Store the drone $Drone_i$ in queen's spermatheca;

Update $Speed$as in equation (9);

End While.

For i=1 to B do

Begin

$Brood_i = Drone_i + rand(0,1)(Queen - Drone_i)$;

End For.

For i=1 to W do

Begin

Improve the broods by local search;

End For.

Select the $best\_brood$ with highest fitness;

If $fit_{best\ brood} > fit_Q$

Then $Q \leftarrow best_{brood}$

$fit_Q = fit_{best\ brood}$;

End If.

End For.

Update:

$x_{curr-i} \leftarrow x_{curr-i} + v_i \cos\theta$;

$y_{curr-i} \leftarrow y_{curr-i} + v_i \sin\theta$;

Return;

End

5. EXPERIMENT AND COMPUTER SIMULATION

The multi-robot path planning was implemented in C on a Pentium processor. The experiment was performed with input parameters as shown inTable1, alongside 10 similar soft-bots of circular cross section. The radius of robot was 6 pixels. For each robot the starting and the goal points are pre-defined prior to initiating the experiment. The experiments were performed with 2,4,6,8 and 10 differently shaped obstacles.

Table 1.Input parameters values

Input Parameters	Initialized values
No. of population	10
D	9
B	9
W	5
C	9
S_{max}	1
S_{min}	0.2
E_{max}	1
α	0.93
γ	0.93

While performing the experiments, old obstacles were retained and new obstacles were added. The experiments were conducted with equal velocities for all the robots in a given run of the program; however, the velocities were adjusted over different runs of the same program.

One of our experimental world-maps is shown in Fig. 4. Fig. 4(a) demonstrates an initial configuration of the world map with 4 dark obstacles, and the starting and the goal positions of 6 circular soft-bots. The steps of movement of the robots are shown in Fig. 4(b).

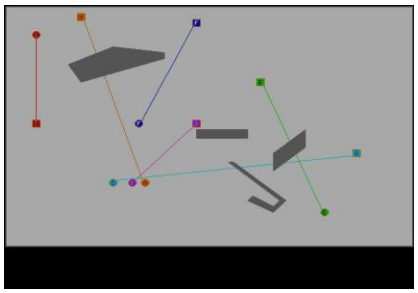

Fig4 (a): Initial configuration of the world map with 4 obstacles.

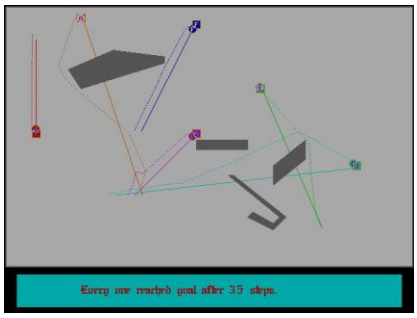

Fig 4(b): Final configuration of the world map.

To analyze the performance of the proposed multi-robot motion-planning problem, we measured the following two parameters.

5.1 Average total path deviation (ATPD)

Let P_{ik} be a path from the starting point S_i to the goal point G_i generated by the program for robot R_i in the kth run. If P_{i1}, $Pi2,\ldots,$ Pik are the paths generated over k runs then the average path traversed (APT) by robot R_i is given by $\sum_{j=1}^{k} P_{ij}/k$ and the average path deviation for this robot is evaluated by measuring the difference between APT and the ideal shortest path between S_i to G_i (with minimum threshold spacing with each obstacle). The threshold in our experiment was considered to be one pixel.

If the ideal path for robot R_i obtained geometrically is $P_{i\text{-}ideal}$, then the average path deviation is given by $P_{i\text{-}ideal} - \sum_{j=1}^{k} P_{ij}/k$. Therefore for n robots in the workspace the average total path deviation (ATPD) is $\sum_{i=1}^{n}(Pi - ideal - \sum_{j=1}^{k} P_{ij}/k)$.

5.2 Average Uncovered Target Distance

Given a goal position G_i and the current position C_i of a robot on a 2-dimensional workspace, where G_i and C_i are 2-dimensional vectors, the uncovered distance of robot i is
$\| G_i - C_i \|$, where $\|.\|$ denotes Euclidean norm.
For n robots, uncovered target distance (UTD) is the sum of $\| G_i - C_i \|$ i.e., UTD=$\sum_{i=1}^{n} \| Gi - Ci \|$
Now, for k runs of the program, we evaluate the average of UTDs and call it the average uncovered target distance (AUTD). For all experiments conducted in this study, we considered $k=5$.

The experiment was conducted using the centralized version of the algorithm, where we used (8) as the fitness function to determine the next position of each robot from the current position. The algorithm is iterated until all the robots reach their respective goal positions. Let the number of robots be n and the number of obstacles m. the experiment was performed by setting same velocity for all the robots in a given program run and AUTD readings versus the number of steps were noted for each run. The experiment was then repeated by changing velocities of the robots in each run.

Fig. 5 shows that with decrease in velocity, AUTD takes a longer time to attain zero value. Similar observations also follow for the number of robots n, as a variable in the AUTD versus number of steps plot (Fig. 6).

Fig. 5 also shows that the AUTD gradually diminishes with iterations. Further, it is noted that the larger the velocity settings of the robots in program run, the faster is the fall off in the AUTD profile.

The fall-off in AUTD over program steps for a given n is demonstrated in Figs. 6 where we see that the larger the number of robots, the slower the convergence. Slower convergence, in turn, causes a delayed fall-off in AUTD.

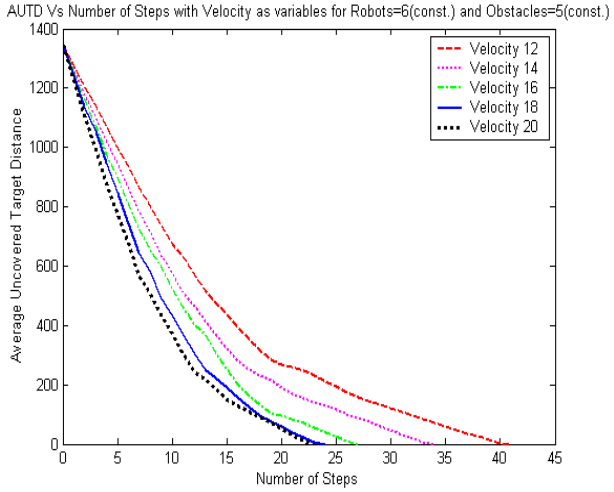

Fig 5: AUTD vs. Number of steps with velocity as variable for number of obstacle=5 (constant).

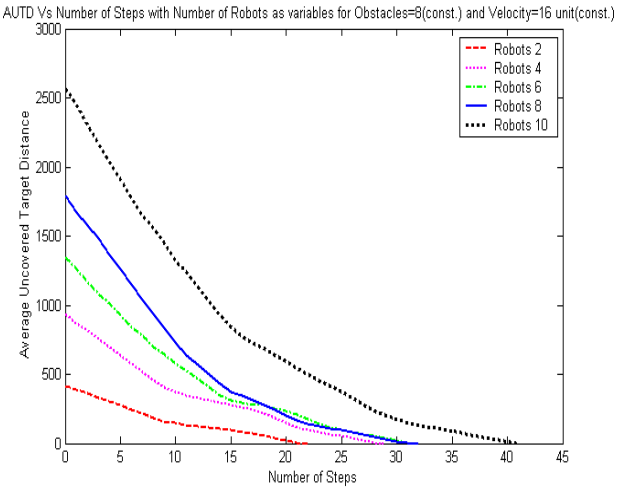

Fig. 6. AUTD vs. Number of steps with number of robots as variables for number of obstacle=8 (constant).

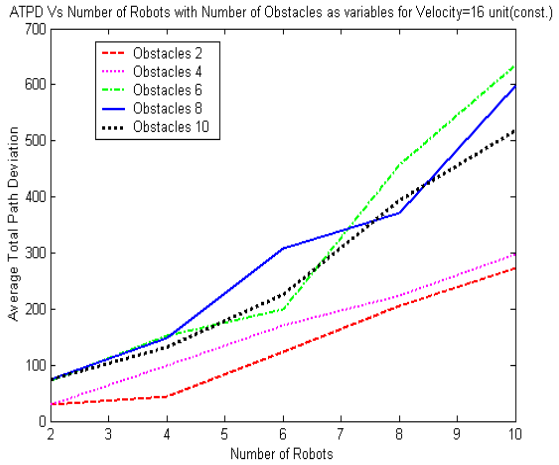

Fig. 7(a). ATPD vs. Number of Robots with number of obstacles as variables for velocity =16 unit (constant).

We note from Fig. 7(a) that ATPD is a non-decreasing function of *n* for a constant *m*. An intuitive interpretation of this phenomenon is that with increase in *n*, robots face more constraints to plan local trajectories, thereby increasing ATPD. It is also noted from Fig. 7(a) that for a constant *n*, an increase in *m* causes more spatial restrictions in trajectory planning, thereby increasing ATPD. The same observations follow from Figure 7(b).

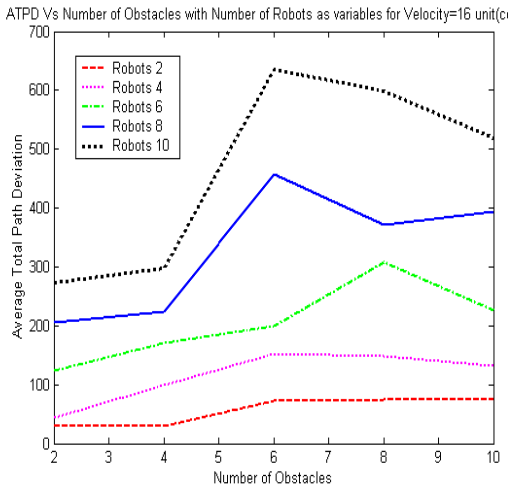

Fig 7(b): ATPD vs. number of obstacles with no. of robots as variables for velocity=16 unit (constant).

The fall-off in AUTD over time for a given *n* is demonstrated in Fig 8(a) where we see that the larger the number of robots, the slower the convergence. Slower convergence, in turn, causes a delayed fall-off in AUTD. Also we note from Fig. 8(b) that for a constant *n*, an increase in *m* causes more spatial restrictions in trajectory planning, thereby increasing time required to reach the goal position.

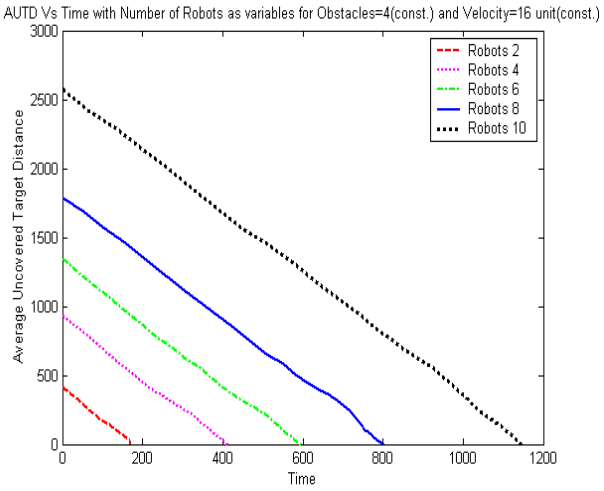

Fig 8(a): AUTD vs. time with number of robots as variables for velocity=16 unit (constant).

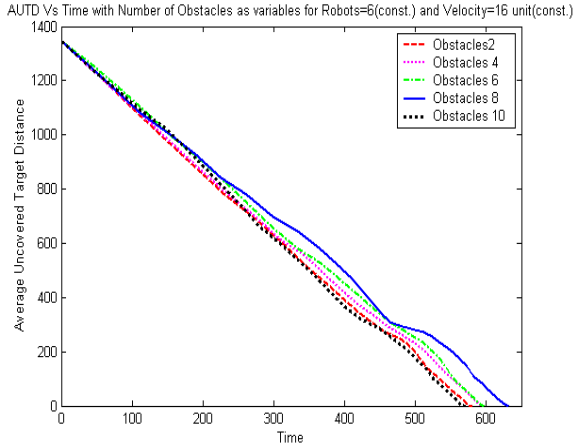

Fig 8(b): AUTD vs. time with number of robots as obstacles for velocity=16 unit (constant).

The relative performance of DE, PSO and HBMO can be studied through error estimation as indicated in Fig. 9(a)-(c). In these figures, we plotted the average of total path traversed (ATPT) obtained from classical DE-, PSO- and HBMO - based experiments, corresponding to each value of *n*. We also evaluated the error in ATPT by taking the difference of ATPT values obtained from DE and HBMO as shown in Fig. 9(b) and also from PSO and HBMO as shown in Fig. 9(c). Let *Ei* be the error for the *i*-th sample data. Since the errors for different sample data are all positive, indicating a superiority of HBMO over DE and PSO, a measure of the relative goodness of HBMO over DE and PSO can be defined as the root mean square error *Er.m.s*= 28.58386384 and *Er.m.s*= 107.3231872 respectively. This shows HBMO as having an advantage over DE and PSO for the multi-robot motion planning problem. Of course, the root mean square error (28.58386384 and 107.3231872 respectively) at the sample points being insignificantly less than the root mean square value (1091.754129 and 1136.364117 respectively) of the averaged ATPT profiles for DE, PSO and HBMO - based simulations.

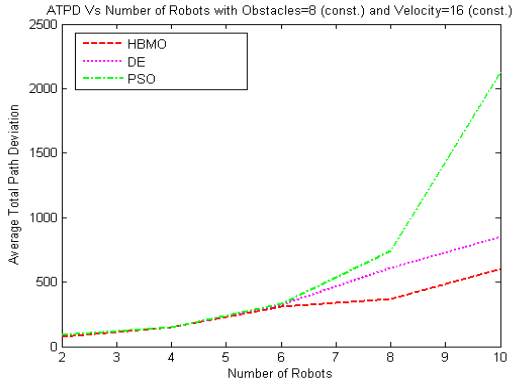

Fig 9(a): Average total path traversed vs. Number of robots.

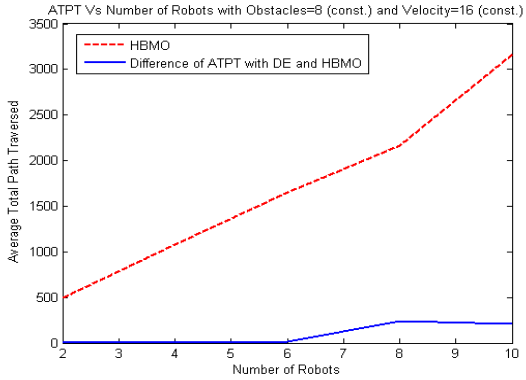

Fig 9(b): Average (dotted line) and the difference (solid line) of ATPT vs. Number of robots obtained from the Fig. 9(a).

In Fig. 10, we plotted the average of total path deviation (ATPD) obtained from classical DE-, PSO- and HBMO - based experiments, corresponding to each value of *n*. From the figure it has been noted that path deviation incurred in case of HBMO-based simulation is less than that of classical DE and PSO-based simulations.

From Fig. 11, it has been noted that AUTD takes more time to attain a zero value in case of classical DE and PSO- based simulations than HBMO-based simulation

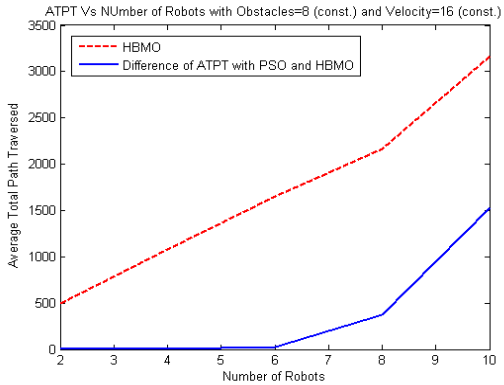

Fig 9(c): Average (dotted line) and the difference (solid line) of ATPT vs. Number of robots obtained from the Fig. 9(a).

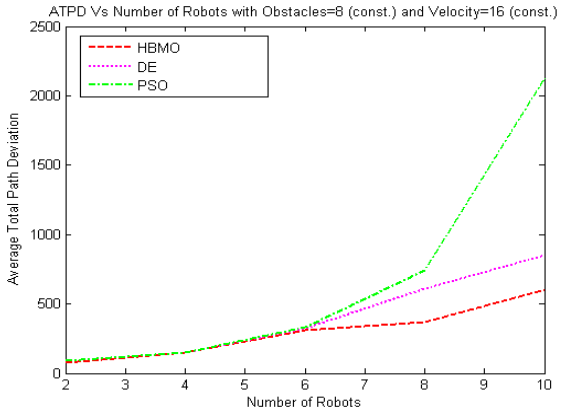

Fig 10: Average total path deviation vs. Number of robots .

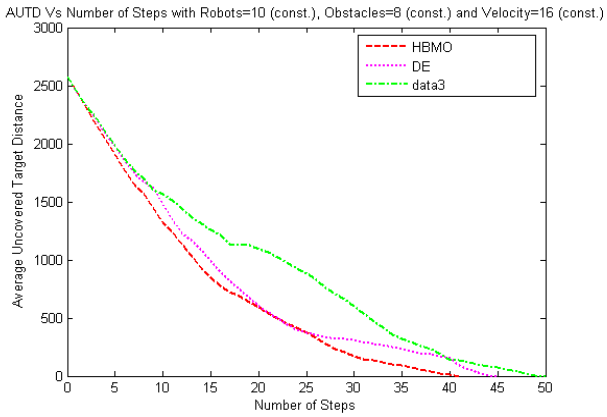

Fig 11: Average uncovered target distance vs. Number of steps.

The relative performance of HBMO, DE and PSO-based can be studied through the plot of average values of uncovered target distance (AUTD), total path traversed (ATPT) and total path deviation (ATPD) obtained from HBMO-, DE- and PSO-based experiments and also by observing the number of steps required for the robots to reach their goals with HBMO, DE and PSO based algorithm as shown in Fig. 12(a)-12(c). HBMO seems to have marginally outperformed classical DE and PSO considering all the cases.

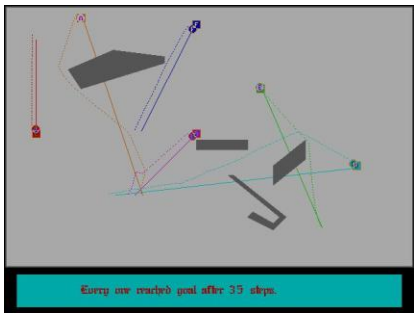

Fig 12(a): Final configuration of the world map after execution of the HBMO- based simulation with 6 robots and 4 obstacles requiring 35 steps.

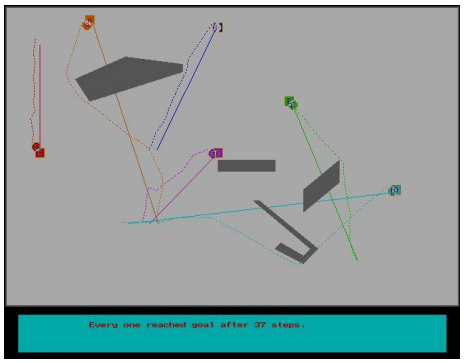

Fig 12(b): Final configuration of the world map after execution of DE- based simulation for 6 robots and 4 obstacles requiring 37 steps.

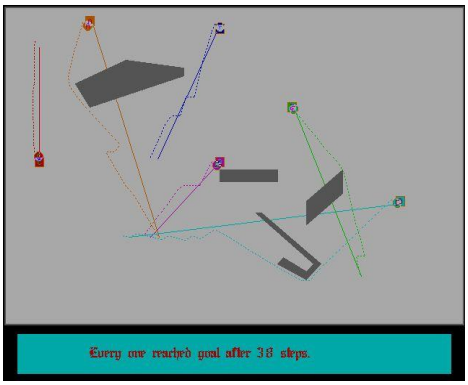

Fig 12(c): Final configuration of the world map after execution of PSO based- simulation for 6 robots and 4 obstacles requiring 38 steps.

6. CONCLUSION

The paper introduced a new technique for multi-robot path-planning in a given environment with an ultimate objective to select the shortest path length of all the robots without hitting any obstacles in the world map. The HBMO algorithm has been employed here for local path-planning of the individual robots. Experiments reveal that the proposed scheme outperforms the PSO- and DE-based path-planning scheme at least with respect to two well-known metrics: ATPT and AUTD.

7. REFERENCES

[1] Malcolm R. K. Ryan, *"Exploiting Sub graph Structure in Multi-Robot Path Planning"*, in Journal of Artificial Intelligence Research 31 (2008) 497-542.

[2] Laura M. Grabowski, *"Robot Navigation: A Developmental Approach"*, in Michigan Celebration of Women in Computing, 2007.

[3] R Regele and P Levi, *"Cooperative Multi-Robot Path Planning by Heuristic Priority Adjustment"*, in Proceedings of the IEEE/RSJ International Conf on Intelligent Robots and Systems, 2006.

[4] K.H. Sedighi, K. Ashenayi, T.W. Manikas, R.L. Wainwright and H. Tai, *"Autonomous Local Path Planning for a Mobile Robot Using a Genetic Algorithm"*, in Proceedings of the IEEE International Conference on Robotics and Automation, 2004, 1338–1345.

[5] Jayasree chakraborty, Amit Konar, L.C Jain and Uday K. Chakraborty, *"A Distributed Cooperative Multi-Robot Path Planning Using Differential Evolution"*, in Journal of Intelligent & Fuzzy Systems 19, 2008, 1–15.

[6] J. Xiao, Z. Michalewicz, L. Zhang, and K. Trojanowski, *"Adaptive Evolutionary Planner/Navigator for Mobile Robots"*, in IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, April, 1997.

[7] R. Smierzchalski and Z. Michalewicz, *"Path planning in Dynamic Environments"*, in *Innovations in Robot Mobility and Control*, S. Patnaik (Ed.), Springer-Verlag, Berlin Heidelberg, 2005.

[8] M.Bennewitz, W.Burgard and S.Thrun, "*Optimizing schedules for prioritized path planning of multi-robot systems*", in Proceedings of the IEEE International Conference on Robotics and Automation, 2001, 271–276.

[9] H. Meng and P. D. Picton, *"Neural Network for Local Guidance of Mobile Robots"*, in Proc. of the third Int. Conf. on Automation, Robotics and Computer Vision (ICARCV' 94), pp. 1238-1242, Singapore, Nov. 1994.

[10] Amir Hosseinzadeh and Habib Izadkhah, *" Evolutionary Approach for Mobile Robot Path Planning in Complex environment"* in IJCSI Vol. 7, Issue 4, No 8, July 2010.

[11] Bhaduri, A., *"A mobile robot path planning using Genetic Artificial Immune Network algorithm ",* in IEEE transaction on Nature & Biologically Inspired computing, 2009, 1536 – 1539.

[12] Bozorg Haddad, O., and Afshar, A. (2004), *"MBO (Marriage Bees Optimization), A New Heuristic Approach in Hydro systems Design and Operation"*, in 1st International Conference on Managing Rivers In the 21st Century: Issues and Challenges, Penang, Malaysia, Sep. 2004