Hindawi Publishing Corporation
Chinese Journal of Engineering
Volume 2014, Article ID 901986, 8 pages
http://dx.doi.org/10.1155/2014/901986

Research Article

Team Robot Motion Planning in Dynamics Environments Using a New Hybrid Algorithm (Honey Bee Mating Optimization-Tabu List)

Mohammad Abaee Shoushtary,[1] Hasan Hoseini Nasab,[2] and Mohammad Bagher Fakhrzad[2]

[1] *Industrial Engineering, Yazd University, Yazd 8196658899, Iran*
[2] *Faculty of Industrial Engineering Department, Yazd University, Yazd, Iran*

Correspondence should be addressed to Mohammad Abaee Shoushtary; mohamad.abaie@gmail.com

Received 4 February 2014; Revised 26 April 2014; Accepted 26 April 2014; Published 29 May 2014

Academic Editor: Jing-Shan Zhao

This paper describes a new hybrid algorithm extracted from honey bee mating optimization (HBMO) algorithm (for robot travelling distance minimization) and tabu list technique (for obstacle avoidance) for team robot system. This algorithm was implemented in a C++ programming language on a Pentium computer and simulated on simple cylindrical robots in a simulation software. The environment in this simulation was dynamic with moving obstacles and goals. The results of simulation have shown validity and reliability of new algorithm. The outcomes of simulation have shown better performance than ACO and PSO algorithm (society, nature algorithms) with respect to two well-known metrics included, ATPD (average total path deviation) and AUTD (average uncovered target distance).

1. Introduction

One of the most important issues in using robots is working automatically without human intervention. Autonomous robots are robots that self-control itself without human intervention [1]. Today, these robots could be applied in different areas such as working with dangerous materials, working in military service [2], underwater working [3], and rescue robots [4]. But using robots in these working areas is faced with some difficulties such as accuracy and constancy in operation [5]. The robots work in an unknown [6] and dynamics [2] Environments. The obstacles and goals in these environments are variable with a moving situation [7]. So, these robots must perform their tasks in such complex environment. In these environments, the robots must identify their tasks, reading and reaching decision for performing a good performance [8]. Consequently, suitable and accurate programming of robots has a positive influence on their operation. Moreover, we can use hardware capabilities of robots with a good programming of robots [9]. So, robot motion planning has been presented as an important field in robotic science [10].

Robot motion planning refers to process of robot task breakdown in the format of separated and discrete motions [7]. In robot motion planning, general strategies are educated to robots for selecting suitable motion among different motions that are available for it [11]. This helps them in doing their chores without any important problems or obstacle collision.

But the working capacity of a robot is limited and when we use it in a real world's environment, we need to be using a group of them [12]. When we use a group of robots, we faced new problems such as robot cooperation [13], robot obstacle avoidance [10], and robot avoidance deadlock [14].

Furthermore, using robot in industrial tasks causes some overall costs such as energy cost [4], robot service, and maintenance cost [11]. Moreover, when we apply a robotic system, we want to increase performance and productivity [10]. So, we need to reduce costs and increase the speed of working of robots.

Because of the following reasons, this paper tries to investigate a new hybrid algorithm for team-robot motion planning that offers a robot motion planning algorithm with an optimized robot path travelling distance and obstacle avoidance in dynamics and unknown environment with moving goals and obstacles according to real world conditions. For doing it, we use honey bee mating optimization algorithm for taking an optimized path according to robot path travelling distance and use tabu lists for obstacle avoidance. The validity and reliability of this algorithm show with regard to simulation results. All simulations has been done in Webots 7.0.1 simulation software and programming of this algorithm implemented on a C++ programming language on a Pentium PC. The simulation results compare to two other algorithms including ant colony optimization algorithm (ACO) and particle swarm (PSO) algorithm. The outcomes of simulation have shown better performance than ACO and PSO algorithm (society, nature algorithms) with respect to two well-known metrics included, ATPD (average total path deviation) and AUTD (average uncovered target distance).

In the following part, we distinguish the research methodology. Section 3 is about the results of inquiry. The last section of this research describes the conclusion of the research.

2. Methodology

In the first step of managing this research, we describe the methodology of environmental modeling. Then we identify the problem definition and the offered algorithm structure. Ultimately, we describe the simulation methods and ways of making resolutions.

2.1. Environment Modeling. In this research, the environment is dynamic and unknown. Dynamic environments are the environments with moving obstacles and goals. For the modeling of dynamic in obstacles and goals, we must apply a physical modeling. The physical modeling is based on the dynamic equations in physics science. According to physics science in one step of time (from $t-1$ to t) an obstacle or goal does a motion from situation 1 to situation 2. Figure 1 shows this motion. In this figure, if we assume that the boxes are the obstacles and the stars are goals and the cylindrical object is a robot, when the robot is in time step t, the robot must predict the obstacle and the goal position for time step $t+1$ according to previous knowledge extracted from its sensors and cameras in the previous time step (from $t-1$ to t). For calculation of the next position of obstacles and goals, robot uses physical equations of motion.

According to physics science, a solid object makes a motion with a velocity vector from its position at time t to the next position in time $t+1$. So, the robot creates a prediction according to

$$x_{\text{obs-or-goal-}i}(t+1) = x_{\text{obs-or-goal-}i}(t) + v_{\text{obs-or-goal-}i}(t)\cos(\theta),$$
$$y_{\text{obs-or-goal-}i}(t+1) = y_{\text{obs-or-goal-}i}(t) + v_{\text{obs-or-goal-}i}(t)\sin(\theta). \tag{1}$$

According to Figure 1, x and y in this equation are the position of obstacles and goals that are shown with its indexes; it is the time, θ is the angle between the velocity vector and horizontal vector. The velocity vector is computed according to

$$V_{\text{obs-}i}(t) = \left(\frac{\left(\sqrt{\left(x_{\text{obs-}i}(t) - x_{\text{obs-}i}(t-1)\right)^2 + \left(y_{\text{obs-}i}(t) - y_{\text{obs-}i}(t-1)\right)^2} \right)}{t} \right),$$
$$V_{\text{goal-}i}(t) = \left(\frac{\left(\sqrt{\left(x_{\text{goal-}i}(t) - x_{\text{goal-}i}(t-1)\right)^2 + \left(y_{\text{goal-}i}(t) - y_{\text{goal-}i}(t-1)\right)^2} \right)}{t} \right). \tag{2}$$

For the modeling of obstacle and goals in the simulation software, we use a supervised plan for making motion in time according to random position of obstacle and goals and equations (2).

This dynamic in the environment is unknown for the robot. Robots did not have any previous information about robots, obstacles, and goals. Robots must get information from their sensors and cameras.

The simulation software is working with virtual reality modeling language. So, we must use a supervisor node for making these changes in the environment.

2.2. Problem Definition. 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 multirobot path planning problem:

among a fixed set of actions in motion the robot has to select only one action at a time;

the path planning problem, hence, incurs a bit of steps until all the robots arrive at their respective address.

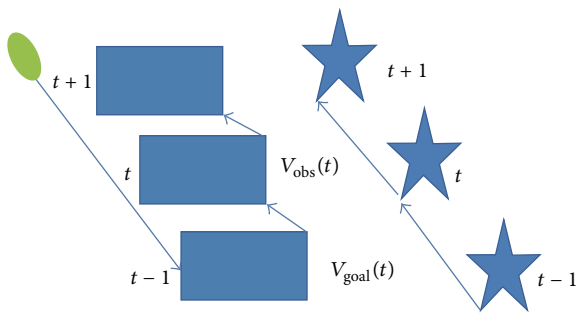

Figure 1: The motion of obstacles and goals in dynamic environment.

The following principles are used to satisfy the assumptions:

the robot first ascertains the next position in order to coordinate itself with the destination and constructs a path to that location.

Figure 2 shows the modeling of problem. In this picture, each robot show by R. The global positions of robot were calculated by information extracted from GPS device (online) mounted on robot according to equation (3).

In Figure 2, the robot R_i is available in time t in position 1 with (X_i, Y_i) coordination and wants to reach to goal that there is in (X_{ig}, Y_{ig}) coordination in time $t + 1$. In doing that, it faces an obstacle. So, it must go to a point near to obstacle with (X_i', Y_i') coordination at first and then follow the goal. So, it must go to this point by using a velocity vector named v_i in time t. According to that the (X_i', Y_i') coordination is calculated with

$$\begin{aligned} x_i' &= x_i + v_i \cos\theta_i \Delta t, \\ y_i' &= y_i + v_i \sin\theta_i \Delta t. \end{aligned} \tag{3}$$

Now, if we desire to compute the path traveling distance of all robots, we use

$$D = \sum_{i=1}^{n} \left\{ v_i + \left(\sqrt{\left(x_i + v_i \cos\theta_i - x_{ig}\right)^2 + \left(y_i + v_i \sin\theta_i - y_{ig}\right)^2} \right) \right\}. \tag{4}$$

According to (4), the path traveling distance of all robots is minimized when (4) becomes minimized. But because this problem is a NP-hard problem, solving that based on common methods is not possible and we must use metaheuristic methods. So, we use honey bee mating optimization algorithm for choosing the next path in each time step.

2.3. Honey Bee Mating Optimization Algorithm (HBMO). The honey bee is a social insect that can survive only as a member of a community, or a colony. The colony of different drone's sperm there is in her spermatheca; she can use parts of the honey bee community which 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. 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

$$\text{Prob}(D) = \exp\left(-\frac{D(f)}{S(t)}\right), \tag{5}$$

where Prob(D) is the probability of adding the sperm of drone D to the spermatheca of the queen (i.e., 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 well as the queen's. After each transition in space, the queen's speed decreases according to the following:

$$\begin{aligned} S(t+1) &= \alpha \cdot S(t), \\ E(t+1) &= \gamma \cdot E(t), \end{aligned} \tag{6}$$

where α and γ are factors such that $\gamma, \alpha\varepsilon(0, 1)$ 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 (5). 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.

By replacing the bees with points that are available around the robot, we can choose the best point with minimum traveling distance for each robot. But this point may be

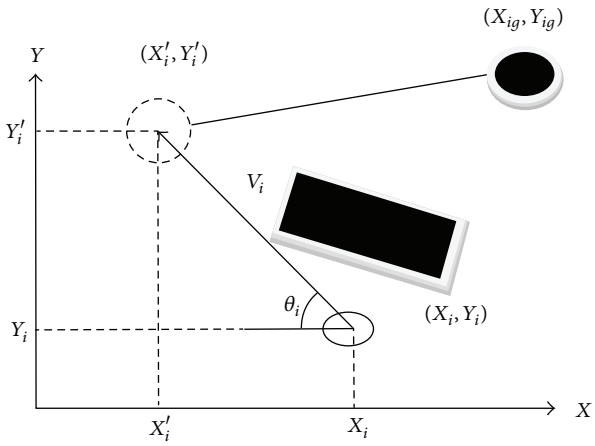

Figure 2: The problem definition.

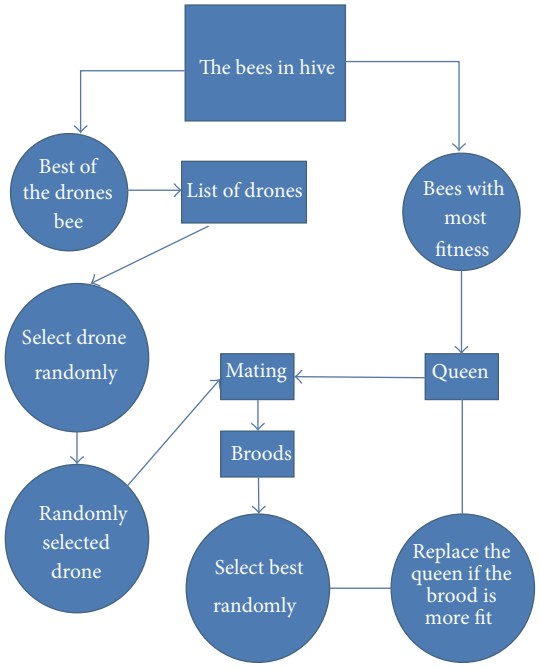

Figure 3: HBMO algorithm diagram.

available in the obstacles and robots must choose another point. So, we use tabu list technique for avoiding obstacles. The diagram of this algorithm is shown in Figure 3.

2.4. Search Strategies and Tabu List Technique. The first step in this algorithm is searching. Searching begins with camera searching in t time. According to things that camera found or forecast, we have three strategies for robots. These strategies are as follows:

if the camera did not find an object near to robot for $t + 1$ time step, the robot continues the algorithm and choose the near optimum point according to HBMO and move to it. At the end must put this calculated point in tabu list;

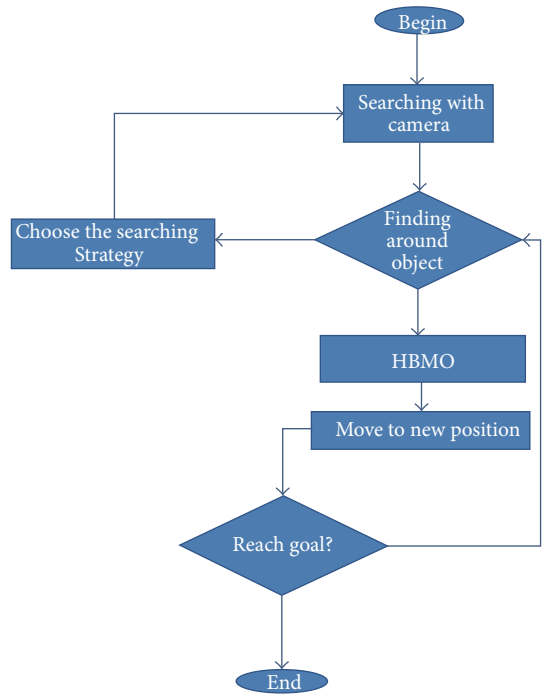

Figure 4: Searching and moving algorithm diagram.

if camera found a goal near to robot in $t + 1$ time step, then robot chooses the near to optimum point, according to HBMO move to this point and put it to tabu list;

if camera found an obstacle near to robot in $t + 1$ time step, then robot puts this point to tabu list and choose to optimum point, according to HBMO and move to that and put it to tabu list.

After doing the search, the robot runs the HBMO algorithm and chooses a near to optimum path and moves to that point. The searching and moving algorithm diagram is shown in Figure 4.

2.5. Formation of Tabu Lists. In this algorithm, there are two tabu lists. One of these is global and the other is local. Each robot has a local tabu list and there is a global list for all algorithms. The local tabu list is for robot obstacle avoidance and the global tabu list is for avoiding robot loss. The main pseudocode is shown in Pseudocode 1.

3. Results

The algorithm was implemented in a C++ programming language and on a Pentium PC. All of the simulations are done in Webots 7.0.1. The results of simulations are evaluated according to two known metrics: average total path deviation (ATPD) and average uncovered target distance (AUTD). An example of simulation is shown in Figure 5.

3.1. 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

```
Initial position (x_robot-i(t)), y_robot-i(t));
Initial position (x_obstacle-i(t)), y_obstacle-i(t));
Initial position (x_goal-i(t)), y_goal-i(t));
Real List_1[i], List_2[j];
J := 1;
Repeat
Search with Camera
    If Camera see No things
     begin
        X_obstacle-i(t + 1) := X_obstacle-i(t); Y_obstacle-i(t + 1) := Y_obstacle-i(t);
        X_goal-i(t + 1) := X_goal-i(t); Y_goal-i(t + 1) := Y_goal-i(t);
     End
    Else if see an obstacle
     begin
        Calculate X_obstacle-i(t + 1) according to equation
        Calculate Y_obstacle-i(t + 1) according to equation
        List_1[j] := X_obstacle-i(t + 1); List_2[j] := X_obstacle-i(t + 1);
     J := J + 1;
     End
    Else if see an goal
     Begin
      Calculate X_goal-i(t + 1) according to equation
      Calculate Y_goal-i(t + 1) according to equation
     end
```

Pseudocode 1: Pseudocode of Algorithm.

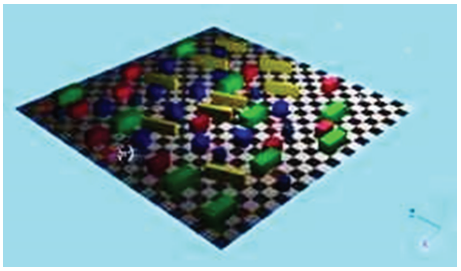

Figure 5: An example of simulation.

program for robot R_i in the kth run. If $P_{i1}, P_{i2}, \ldots, P_{ik}$ are the paths generated over k runs, then the average path traversed (APT) by robot R_i. ATPD is calculated according to

$$\sum_{i=1}^{n} p_{i\text{-ideal}} - \frac{\left(\sum_{j=1}^{k} p_{ij}\right)}{k}. \tag{7}$$

3.2. 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 $\|\cdot\|$ denotes Euclidean norm. For n robots, uncovered target distance (UTD) is the sum of $\|G_i - C_i\|$. 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 (4) as the fitness function to

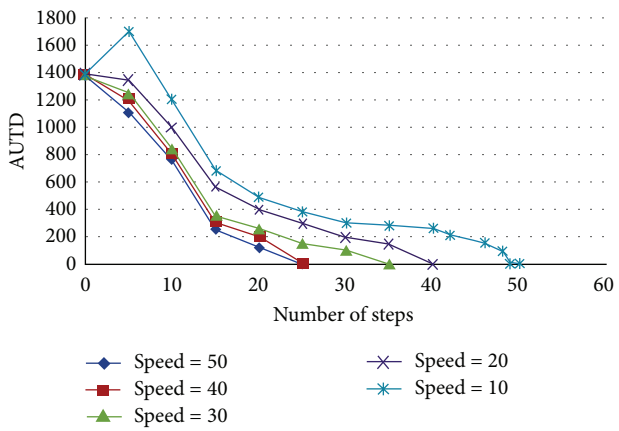

Figure 6: AUTD versus number of steps with velocity as variable for number of obstacles = 5 (constant).

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 the 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.

3.3. The Algorithm Performance. Figure 6 shows that, with the decrease in velocity, AUTD takes a longer time to attain zero value. Similar observations also follow for the number

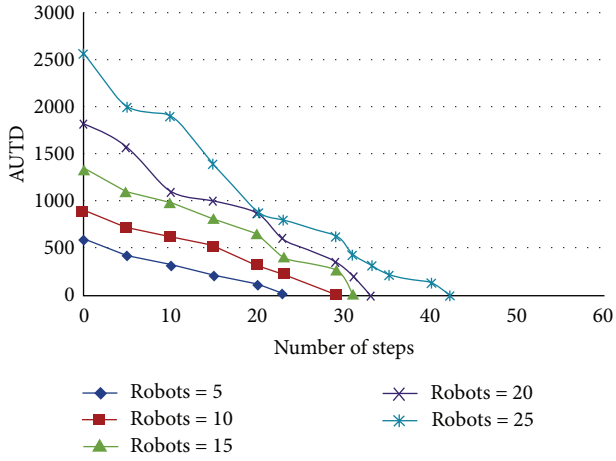

Figure 7: AUTD versus number of steps with number of robots as variables for number of obstacles = 8 (constant).

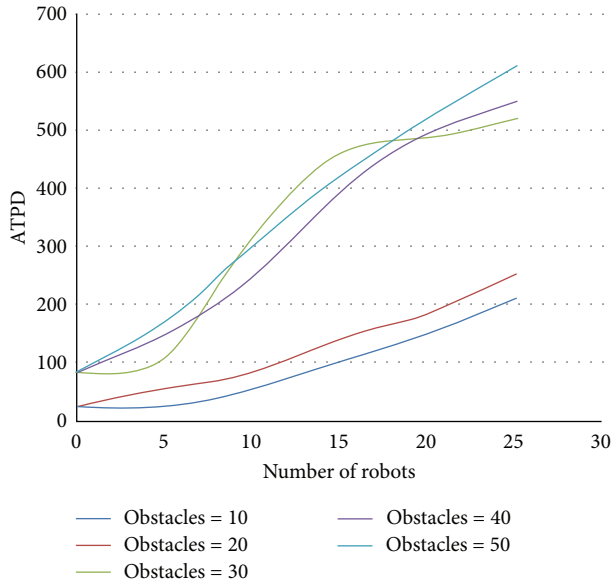

Figure 8: ATPD versus number of robots with number of obstacles as variables for velocity = 16 units (constant).

of robots n, as a variable in the AUTD versus number of steps plot (Figure 7).

Figure 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 the fall-off in the AUTD profile.

The fall-off in AUTD over program steps for a given n is demonstrated in Figure 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.

We note from Figure 8 that ATPD is a nondecreasing 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 Figure 9 that, for a constant n,

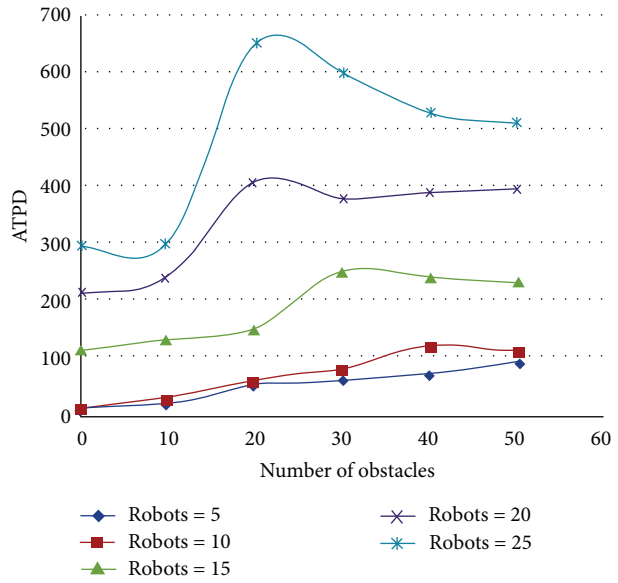

Figure 9: ATPD versus number of obstacles with number of robots as variables for velocity = 16 units (constant).

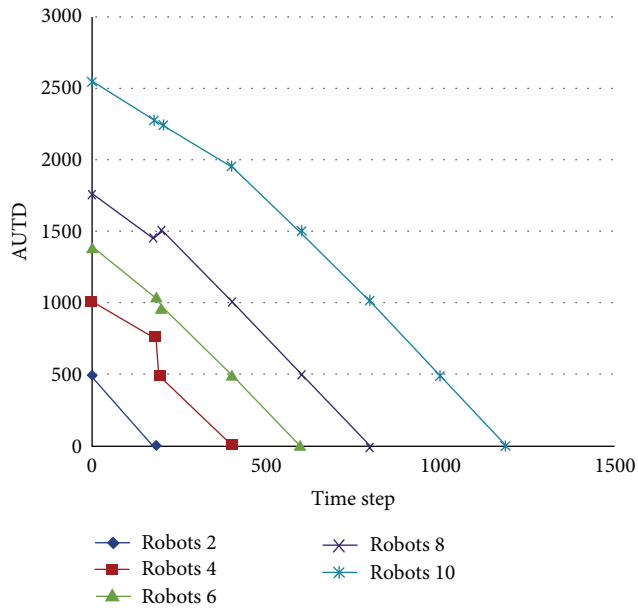

Figure 10: AUTD versus time with number of robots as variables for velocity = 16 units (constant).

an increase in m causes more spatial restrictions in trajectory planning, thereby increasing ATPD. The same observations follow from Figure 9.

The fall-off in AUTD over time for a given n is demonstrated in Figure 10 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 Figure 11 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.

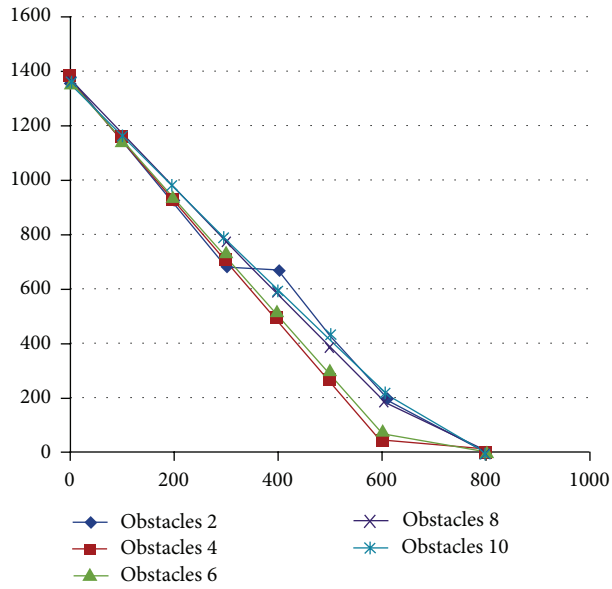

Figure 11: AUTD versus time with number of robots as obstacles for velocity = 16 units (constant).

Table 1: Comparison between new algorithm versus ACO and PSO.

Problem	Number of Obs.	Number of Robots	ATPD			AUTD		
			New algorithm	ACO	PSO	New algorithm	ACO	PSO
1	10	5	23.87	24.29	24.44	38.56	39.06	47.26
2	20	5	26.42	26.89	27.05	45.5	46.09	55.77
3	30	5	28.54	29.05	29.22	53.69	54.39	65.81
4	40	5	29.3	29.82	30.0	63.69	64.18	77.66
5	50	5	29.86	30.39	30.57	74.76	75.73	91.63
6	10	10	30.05	30.59	30.77	88.22	89.36	108.13
7	20	10	31.23	31.79	31.98	104.1	105.45	127.59
8	30	10	31.56	32.12	32.32	122.83	124.43	150.56
9	40	10	32.12	32.69	32.89	144.94	146.83	177.66
10	50	10	32.86	33.45	33.65	171.04	173.26	209.64
11	10	15	33.87	34.48	34.69	201.82	204.45	247.38
12	20	15	52.57	53.51	53.84	238.15	241.25	291.91
13	30	15	73.7	75.02	75.47	281.02	284.67	344.46
14	40	15	97.57	99.33	99.99	331.61	335.92	406.42
15	50	15	124.55	126.79	127.56	391.3	396.38	472.62
16	10	20	155.54	157.83	158.78	461.73	467.73	565.96
17	20	20	189.49	192.9	194.06	554.84	551.92	667.83
18	30	20	228.42	232.53	223.93	642.91	651.27	788.54
19	40	20	272.41	277.31	278.98	758.64	768.5	929.89
20	50	20	322.12	327.91	329.88	895.19	906.83	1097.27
21	10	25	378.29	385.1	387.41	1056.33	1070.06	1294.78
22	20	25	441.76	449.73	452.41	1246.47	1262.68	1527.84
23	30	25	513.48	605.23	525.86	1470.84	1489.96	1802.85
24	40	25	594.53	615.44	608.86	1735.59	1758.15	2127.36
25	50	25	686.12	698.47	702.66	2048	2024.62	2510.29
Average:			178.4	185.3	182.3	528.46	535.33	647.55

The simulation results of PSO and ACO algorithm show that the average performance of new algorithm is better than them (you can see the results in Table 1).

4. Conclusion

The paper introduced a new technique for team robot motion planning in dynamics and unknown environment with moving goals and obstacles to select the shortest path length of all the robots without hitting any obstacles in the world map with a hybrid algorithm extracted from HBMO and Tabu lists algorithm. Experiments reveal that the proposed scheme outperforms the PSO and ACO robot motion planning scheme at least with respect to two well-known metrics: ATPD and AUTD. These results confirm the validity and reliability of algorithm.

Conflict of Interests

The authors declare that they have no conflict of interests regarding the publication of this paper.

References

[1] N. S. V. Rao and S. S. Iyengar, "Autonomous robot navigation in unknown terrains: incidental learning and environmental exploration," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 20, no. 6, pp. 1443–1449, 1990.

[2] D. Tamilselvi and M. Shalinie, "Navigation of a robot amidst moving obstacles using DPPA (Dynamic Path Planning Agent)," *European Journal of Scientific Research*, vol. 58, no. 4, pp. 506–517, 2011.

[3] Y. Yang, S. Wang, Z. Wu, and Y. Wang, "Motion planning for multi-HUG formation in an environment with obstacles," *Ocean Engineering*, vol. 38, no. 17-18, pp. 2262–2269, 2011.

[4] H. Wei, B. Wang, Y. Wang, Z. Shao, and K. C. C. Chan, "Staying-alive path planning with energy optimization for mobile robots," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3559–3571, 2012.

[5] A. Sgorbissa and R. Zaccaria, "Planning and obstacle avoidance in mobile robotics," *Robotics and Autonomous Systems*, vol. 60, no. 4, pp. 628–638, 2012.

[6] S. S. Ge, Q. Zhang, A. T. Abraham, and B. Rebsamen, "Simultaneous path planning and topological mapping (SP2ATM) for environment exploration and goal oriented navigation," *Robotics and Autonomous Systems*, vol. 59, no. 3-4, pp. 228–242, 2011.

[7] E. Masehian and Y. Katebi, "Robot motion planning in dynamic environments with moving obstacles and target," *International Journal of Aerospace and Mechanical Engineering*, vol. 1, no. 1, pp. 20–25, 2007.

[8] J. M. Martín Ramos, D. López García, F. Gómez-Bravo, and A. Blanco Morón, "Application of multicriteria decision-making techniques to manoeuvre planning in nonholonomic robots," *Expert Systems with Applications*, vol. 37, no. 5, pp. 3962–3976, 2010.

[9] A. Dolgui and A. Pashkevich, "Manipulator motion planning for high-speed robotic laser cutting," *International Journal of Production Research*, vol. 47, no. 20, pp. 5691–5715, 2009.

[10] C. Shi, Y. Wang, and J. Yang, "A local obstacle avoidance method for mobile robots in partially known environment," *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 425–434, 2010.

[11] H. Yu, C.-J. Chi, T. Su, and Q. Bi, "Hybrid evolutionary motion planning using follow boundary repair for mobile robots," *Journal of Systems Architecture*, vol. 47, no. 6, pp. 635–647, 2001.

[12] C. M. Clark, "Probabilistic Road Map sampling strategies for multi-robot motion planning," *Robotics and Autonomous Systems*, vol. 53, no. 3-4, pp. 244–264, 2005.

[13] J. B. Kramer and L. Sabalka, "Multidimensional online motion planning for a spherical robot," *International Journal of Computational Geometry and Applications*, vol. 20, no. 6, pp. 653–684, 2010.

[14] G. A. S. Pereira, V. Kumar, and M. F. M. Campos, "Closed loop motion planning of cooperating mobile robots using graph connectivity," *Robotics and Autonomous Systems*, vol. 56, no. 4, pp. 373–384, 2008.