Research Article

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

Mohammad Abaee Shoushtary, Hasan Hoseini Nasab, and Mohammad Bagher Fakhrzad

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

Copyright © 2014 Mohammad Abaee Shoushtary et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

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 \( t \) to \( t+1 \)) 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

\[
\begin{align*}
x_{\text{obs-or-goal-i}}(t+1) &= x_{\text{obs-or-goal-i}}(t) + v_{\text{obs-or-goal-i}}(t) \cos(\theta), \\
y_{\text{obs-or-goal-i}}(t+1) &= y_{\text{obs-or-goal-i}}(t) + v_{\text{obs-or-goal-i}}(t) \sin(\theta).
\end{align*}
\]

(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, \( \theta \) is the angle between the velocity vector and horizontal vector. The velocity vector is computed according to

\[
\begin{align*}
V_{\text{obs-i}}(t) &= \left( \frac{\sqrt{(x_{\text{obs-i}}(t) - x_{\text{obs-i}}(t-1))^2 + (y_{\text{obs-i}}(t) - y_{\text{obs-i}}(t-1))^2}}{t} \right), \\
V_{\text{goal-i}}(t) &= \left( \frac{\sqrt{(x_{\text{goal-i}}(t) - x_{\text{goal-i}}(t-1))^2 + (y_{\text{goal-i}}(t) - y_{\text{goal-i}}(t-1))^2}}{t} \right).
\end{align*}
\]

(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.
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.

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 meta-heuristic 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

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

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

$$x_i' = x_i + v_i \cos \theta_i \Delta t,$$

$$y_i' = y_i + v_i \sin \theta_i \Delta t.$$  \hspace{1cm} (3)

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 meta-heuristic methods. So, we use honey bee mating optimization algorithm for choosing the next path in each time step.

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:

$$S(t + 1) = \alpha \cdot S(t),$$

$$E(t + 1) = \gamma \cdot E(t),$$

where $\alpha$ and $\gamma$ are factors such that $\gamma, \alpha \in (0, 1)$ 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
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;

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 \((x_{\text{robot-}i(t)}, y_{\text{robot-}i(t)})\);
Initial position \((x_{\text{obstacle-}i(t)}, y_{\text{obstacle-}i(t)})\);
Initial position \((x_{\text{goal-}i(t)}, y_{\text{goal-}i(t)})\);

RealList \([i], \text{List}\_2[j];
J := 1;
Repeat
Search with Camera
If Camera see No things
begin
\[ X_{\text{obstacle-}i(t + 1)} := X_{\text{obstacle-}i(t)}; \]
\[ Y_{\text{obstacle-}i(t + 1)} := Y_{\text{obstacle-}i(t)}; \]
\[ X_{\text{goal-}i(t + 1)} := X_{\text{goal-}i(t)}; \]
\[ Y_{\text{goal-}i(t + 1)} := Y_{\text{goal-}i(t)}; \]
End
Else if see an obstacle
begin
Calculate \(X_{\text{obstacle-}i(t + 1)}\) according to equation
Calculate \(Y_{\text{obstacle-}i(t + 1)}\) according to equation
\(\text{List}_1[j] := X_{\text{obstacle-}i(t + 1)}; \)
\(\text{List}_2[j] := X_{\text{obstacle-}i(t + 1)}; \)
\(J := J + 1; \)
End
Else if see an goal
Begin
Calculate \(X_{\text{goal-}i(t + 1)}\) according to equation
Calculate \(Y_{\text{goal-}i(t + 1)}\) according to equation
end

Pseudocode 1: Pseudocode of Algorithm.

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 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 of steps.
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 $n$,

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

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

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

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


