Improved Ant Colony Optimization Algorithm for Robot Path Planning

Syed Muhammad Fasih Hussain
Department of Computer Science
Dhanani School of Science and Engineering
Habib University – Karachi, Pakistan
sh05204@st.habib.edu.pk

Salman Muhammad Younus
Department of Computer Science
Dhanani School of Science and Engineering
Habib University – Karachi, Pakistan
sy04351@st.habib.edu.pk

Muhammad Munawwar Anwar
Department of Computer Science
Dhanani School of Science and Engineering
Habib University – Karachi, Pakistan
ma04289@st.habib.edu.pk

***Abstract*—Path planning is an active research area for motion control of autonomous robots. Path planning is an NP-complete problem which means that it cannot be solved using traditional algorithms for hard instances. The ACO (Ant Colony Optimization) algorithm is an optimization technique based on swarm intelligence. This report investigates the application of an improved ACO to robot path planning in dynamic and static environments. The simulation in Python shows that improved ACO has higher efficiency for path search.**

***Index Terms*—Dijkstra Algorithm, A* search Algorithm, Ant Colony Optimization, Path Planning**

I. Introduction

In current society, with the development of computer technology and intelligent computing, human beings are moving towards automation. Robots are extensively used in different fields such as planetary exploration, deep ocean exploration, military, home, and so on. Robots use multiple sensors to perceive the machine parameters and the current environment, to avoid obstacles to reach the destination. [5]

Path planning is an important issue for the navigation and motion control of autonomous robot manipulators. In computational complexity theory, path planning is classified as an NP-complete problem. That is, the computational time that is required to solve such a problem increases at an exponential rate when the size (or dimension) of the problem increases.

Let robot A be a single rigid object moving in a 2D or 3D Euclidean space denoted by W, and let obstacles B be rigid objects distributed in W. Then a path planning problem can be formulated as the following :

Given an initial position and a goal position of A in W, generate a path that specifies a sequence of positions of A avoiding contact with B, starting from the initial position and terminating at the goal position. Report failure if no such path exists. [1]

The studies of path planning started in the late '60s and many different algorithms have been proposed since. Dijkstra, A* search , ACO, etc. are commonly used for path planning algorithms. Dijkstra is a greedy algorithm, which only considers the distance from the current node to the next node, and uses the method, traversal search, to find the shortest path. The obtained path has high reliability and good robustness, but the complexity is high and it is easy to fall into local optimum. Based on Dijkstra, the A* search introduces a heuristic function to guide path searching. However, the traditional A* search gradually determines the next path by comparing the neighborhood heuristic function. When the heuristic function value has multiple minimum values, the A* search algorithm cannot guarantee the best solution for path searching. Ant colony optimization is an optimization technique based on swarm intelligence. It is a robust technique. However, it has a long searching time and easy to fall into the local optimum.Due to the shortfalls of AC0, this report presents a modified ACO algorithm. The modified ACO uses the self-adaptive adjustment method to improve the bearing coefficient ρ of pheromone and then updates the improved bearing coefficient into the pheromone update formula of the ACO. [4]

The rest of this paper is organized as follows. Section II introduces Robot Path Planning. Section III reviews the Ant colony optimization algorithm and the improvement .Section IV describes the proposed algorithm and gives the details of its components. Experimental results and relevant analysis are demonstrated in Section V. Finally, conclusions are summarized in Section VI.

II. Background

A. Robot Path Planning Problem

Path planning of autonomous robots can be categorized into four different categories based on the different combinations of environment and obstacles. The environment may be known or unknown. The obstacles can be considered static or dynamic. A dynamic environment is where an optimal path needs to be re-routed when a new obstacle appears. The ultimate goal of path planning is to reduce the distance, cost, or time. In

this report, path planning is done for a known environment with static and obstacles using a modified ACO to find best the possible route from the source to the destination avoiding obstacles. [2]

B. Grid World

This article uses the two-dimensional grid method to represent the robot environment, and the grid squares are divided into two types: free grid square, which is represented by zero, and obstacle grid square, which is represented by one. The robot can only move in the free grid. [3] The source is denoted by S and the destination is represented by D. The goal of the robot is to move from S to D avoiding the obstacles. Report failure if no such path exists

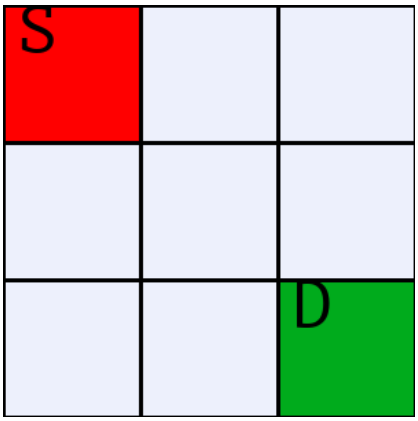


Fig. 1: Representing environment as grid

The movement of robot is restricted to eight directions is depicted in Figure 2 - North West, North, North East, West, East, South West, South East and South West.

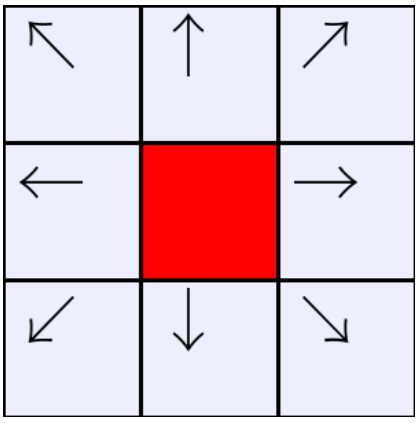


Fig. 2: Possible movements of a robot in a free grid

III. The Ant Colony Optimization Algorithm

A. The Basic Idea of Ant Colony Algorithm

The Ant Colony Optimization ACO was developed by M. Dorigo in 1992 to mimic the behavior of real ants searching for food. When searching for food, ants exhibit complex social behavior by depositing hormones called pheromones. Pheromones attract other ants and outline a path to the food source that can be followed by other ants. As more ants walk along the path, more pheromone is laid, and the likelihood that other ants will also take the path increases. The shortest path to the food source will have the highest concentration of pheromone because more ants can travel in the least amount of time. To prevent the convergence to a sub-optimal path, the pheromone also evaporates over time, thereby reducing the chances that other ants to take the path. On the other hand, the pheromone levels on the shortest path remain high, because in this case, the rate at which ants deposit pheromone is greater than the rate of evaporation. [1]

B. Mathematical Model

Consider a given network where ants can travel between different nodes.The probability that ant k located at node i will choose to go to another node j in the network is given by

$$p_{ij}^k = \begin{cases} \frac{(\tau_{ij}^t)^\alpha (\eta_{ij}^t)^\beta}{\sum_{l \in N_i^k} (\tau_{il}^t)^\alpha (\eta_{il}^t)^\beta} & \text{if } j \in N_i^k \\ 0 & otherwise \end{cases} \tag{1}$$

where τ_{ij}^t is pheromone concentration at edge (i, j), N_i^k the possible set of choices ant k has when at node i, and t represents the time step. The values of α, β, and η_{ij}^k are usually application dependent; η_{ij}^k represents the heuristic information and the value of α and β weight the importance of the pheromone and heuristic values. When $\beta = 0$, $(\eta_{ij}^t)^\beta = 1$, then the probability only depends on pheromone levels; on the other hand, when $\alpha = 0$, then the probability depends on the heuristic information.

The pheromone levels of the path from node i to node j can evaporate with a percentage ρ (also the called the evaporation rate) where $0 \leq \rho \leq 1$.

$$\tau_{ij}^t = (1 - \rho)\tau_{ij}^t \tag{2}$$

After the pheromone evaporation occurs, the new pheromone levels are updated with the additional pheromone laid by the ant(s) that just crossed the path:

$$\tau_{ij}^t = \tau_{ij}^t + \sum_{k=1}^{m} \Delta\tau_{ij}^k \tag{3}$$

$$\Delta\tau_{ij}^k = \frac{1}{C^k} \tag{4}$$

where C^k is the associated cost of ant k for choosing this path.

C. Overall structure of the Ant Colony Algorithm

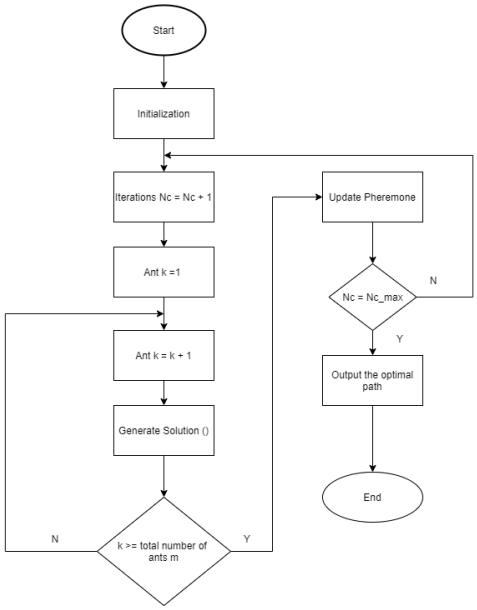


Fig. 3: Computation Flowchart of the ACO

D. Improved Ant Colony Algorithm

Due to the disadvantages that the path searching time of ant colony algorithm is long and it is easy to fall into the local optimal solution, this report presents a self-adaptive adjustment method to improve the bearing coefficient ρ of pheromone, and then updates the improved bearing coefficient into the pheromone update formula of ant colony algorithm [5]. The formula for improved ρ is as follows:

$$\rho_{ij} = e^{\frac{\tau_{ij}^{t} min}{\tau_{ij}}} + e^{-1} \tag{5}$$

where τ_{ij}^{t} min refers to the minimum value of pheromone concentration on the edge(i, j) at time t. The overall structure of the algorithm can be seen in Figure 4.

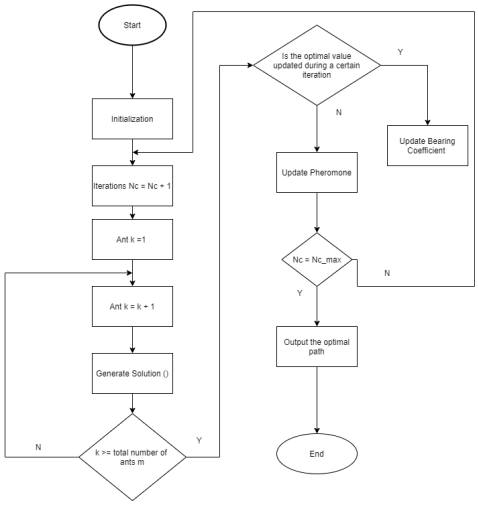


Fig. 4: Computational Flowchart of Improved ACO

IV. Robot Path Planning based on Ant Colony Algorithm

In this section, the improved ant colony optimization algorithm is applied for robot path planning in a grid network.It is assumed that the one ant can only one node and moves to one of its adjacent nodes at a time in eight different directions. All the nodes are evenly distributed in the network; and distance between any two adjacent nodes is normalized to 1. Thus, path length is represented in terms of the number of unit blocks.The envoirment is known but it can have dynamic or static obstacles.

The simulation starts with a grid which is populated with obstacles randomly. The number of obstacles added is proportional to the size of the network. All the pheromones are initialized as 0.1 The improved ant colony algorithm is then applied to find the shortest path and pheromones are deposited.

To simulate a dynamic envoirment, new barriers are added after the algorithm converges. The improved ACO must be called again in order to find the shortest path in this new network with obstacles. The flowchart is show in Figure 5.

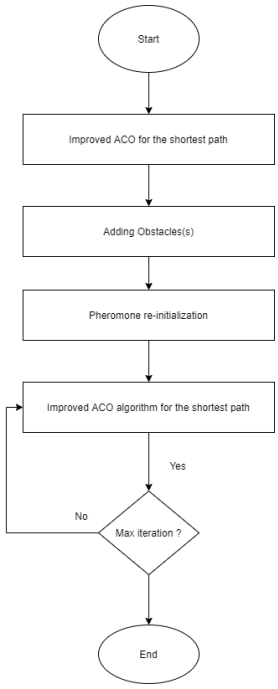


Fig. 5: The flowchart of Improved ACO in a dynamic envoirment

V. Experimental Results of Robot Path Planning

A. Experimental Setup

We simulated the four algorithms A*star search, Dijkstra, ACO and Improved ACO in Python with a $4.30GHz$ processor and $16GB$ Ram. We used the grid sizes of 30×30, 20×20,10×10. We used both static and dynamic obstacles.

B. Simulation Results

Our envoirment can be is a grid of 30×30. Each cell in the grid represents a node in the graph. In this grid, the top most

TABLE I: Parameter for the Ant Colony Algorithm

Parameters	Numerical Value
Number of Iterations , N_c	50
Number of ants , m	40
Alpha,α	1.2
Beta, β	0
Q	1
Attenuation coefficient of Pheromone, ρ	0.6
Pheromones under initial conditions, τ_0	1.5

left corner is our destination which is represented by a red dot whereas our origin is the bottom most right corner which is represented by a blue dot. The free grid where the ants can move is represented by the white cells and the obstacles are represented by the black cells. The grid can be seen in 6

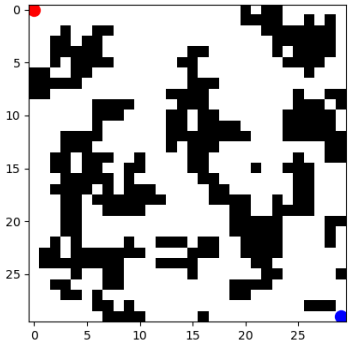


Fig. 6: Static envoirment represented by a 30×30 grid.

After initialising, the shortest path between the source and destination was found using the improved ACO. The path that was returned by the improved by the ACO can be seen in figure 7 and is represented by the blue line.

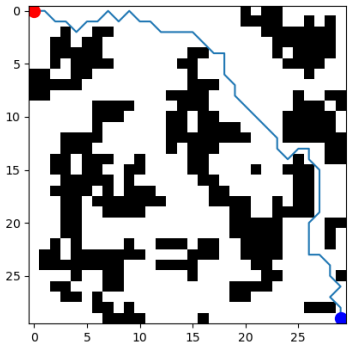


Fig. 7: Global shortest path planning results in 30×30 grid.

In order to simulate a dynamic envoirment , the dynamic obstacles are added after the algorithm converges. The new obstacles that were added are represented by red cells and can be seen in figure 8

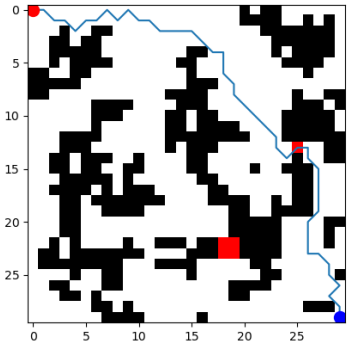


Fig. 8: Obstacles avoiding the route of the ants

The improved ACO was again called to find the new the shortest path after obstacles were added. The final path of the ants can be seen in figure 9

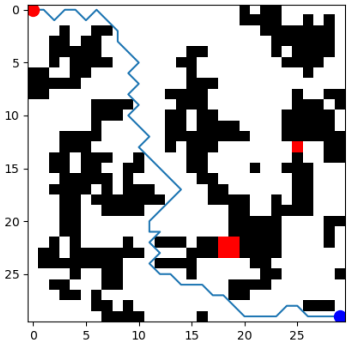


Fig. 9: Final Path found by Ants.

C. Comparison with other path planning algorithms

The results that were obtained by running four different algorithms in grid of different sizes with static obstacles are summarised in Table II,III and IV. The path measured here is measured in number of blocks and the time taken is measured in seconds.

TABLE II: STATIC GRID OF 10 BY 10

Algorithm	Path Length	Time Taken(s)
Dijkstra	14	0.0148
A* Search	15	0.0877
ACO / Improved ACO	15	0.44

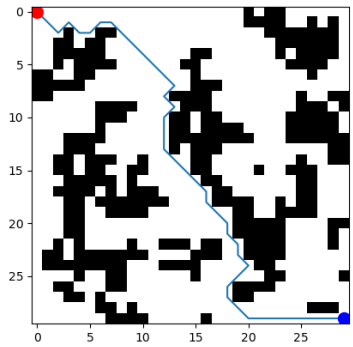
TABLE III: STATIC GRID OF 20 BY 20

Algorithm	Path Length	Time Taken
Dijkstra	38	0.0974
A* Search	38	0.0836
ACO / Improved ACO	38	2.9

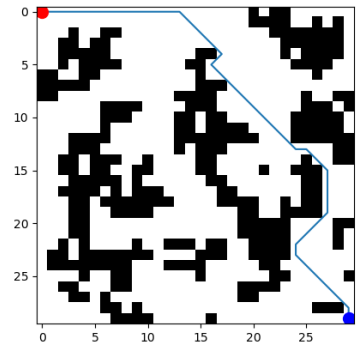
TABLE IV: STATIC GRID OF 30 BY 30

Algorithm	Path Length	Time Taken
Dijkstra	43	0.4936
A* Search	44	0.0463
ACO / Improved ACO	45	4.2

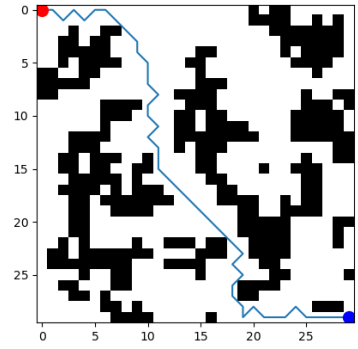
In figure 10, we can see the shortest paths obtained by Dijkstra, A* search and ACO in the 30×30 grid with static obstacles.



(a) A* Search



(b) Dijkstra



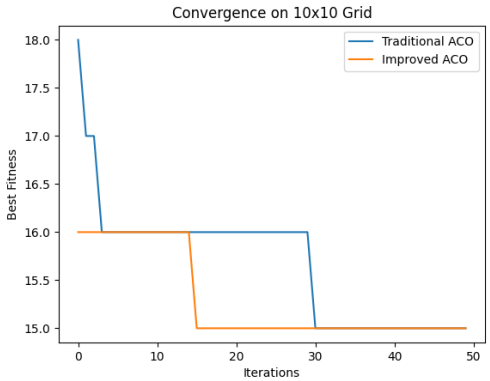
(c) ACO

Fig. 10: Paths obtained in a static grid of size 30×30

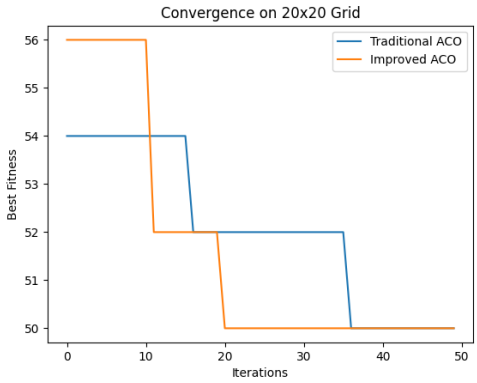
The optimal paths found by the traditional ACO were the same as the improved ACO but the improved ACO had a better convergence rate. The results that were obtained are summarised in V and the graphs can be seen in figure 11

TABLE V: TRADITIONAL ACO VS IMPROVED ACO

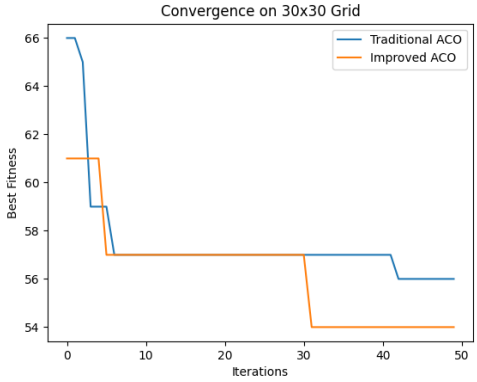
Size	10 BY 10 11a	20 BY 20	30 BY 30
Traditional ACO	15	20	31
ACO	30	36	42



(a) Static grid of 10 by 10 with static obstacles



(b) Static grid of 20 by 20 with static obstacles



(c) Static grid of 30 by 30 with static obstacles

Fig. 11: Convergence rate of ACO vs dynamic ACO

VI. Conclusion

In this paper, we applied four algorithms: Dijkstra,A* Search, ACO and improved ACO to Robot Path planning in a grid world. The improved ACO has the following characteristics: (1) it does not get trapped in a local optima (2) and has a high search efficiency. The performance of this algorithm can be further improved by improving heuristic function and the selection of the next node. In the future, we would like to focus on multi-robot path planning problem in a dynamic envoirment.

References

[1] Michael Brand, Michael Masuda, Nicole Wehner, and Xiao-Hua Yu. Ant colony optimization algorithm for robot path planning. In *2010 International Conference On Computer Design and Applications*, volume 3, pages V3–436–V3–440, 2010.

[2] Parvathy Joshy and P Supriya. Implementation of robotic path planning using ant colony optimization algorithm. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, volume 1, pages 1–6, 2016.

[3] Jianhua Liu, Jianguo Yang, Huaping Liu, Xingjun Tian, and Meng Gao. An improved ant colony algorithm for robot path planning. *Soft Comput.*, 21(19):5829–5839, October 2017.

[4] Zhen Nie and Huailin Zhao. Research on robot path planning based on dijkstra and ant colony optimization. In *2019 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, pages 222–226, 2019.

[5] Hong Zhao. Optimal path planning for robot based on ant colony algorithm. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 671–675, 2020.