Optimization Techniques for Multi Cooperative Systems MCTR 1021
Mechatronics Engineering
Faculty of Engineering and Materials Science
German University in Cairo

Task Allocation and Cooperative Path Planning of Multi Mobile Manipulators

By

Team-12

Youssef Waleed Abdelshafy
Seif Hany Seif
Zeiad Hossam Othman
Mohanad Ahmed Azzam
Abdelrahman Mohamed Elsisi

Course Team

Dr.Eng. Omar Shehata
M.Sc Catherine Elias
M.Sc Shaimaa Elbaklish

December 1, 2022

This is to certify that:

(i) the report comprises only my original work toward the course project,

(ii) due acknowledgment has been made in the text to all other material used

Team-12
December 1, 2022

Abstract

Contents

List of Abbreviations

MRS	Multi-robot systems
MRTA	Multi-robot Task Allocation
MAPD	Multi-Agent Pickup and Delivery
GA	Genetic Algorithm
UAVs	Unmanned Aerial Vehicles
PSO	Particle Swarm Optimization
ACO	Ant Colony Optimization
SA	Simulated Annealing

List of Figures

List of Tables

Chapter 1

Introduction

Multi-robot systems (MRS) are a group of robots that are designed aiming to perform some collective behavior. By this collective behavior, some goals that are impossible for a single robot to achieve become feasible and attainable. One of the reasons that the topic has become more popular is the various foreseen benefits of MRS compared to single robot systems. Resolving task complexity, increasing the performance, increasing the performance and simplicity in design are from these benefits.

Multi-robot Task Allocation (MRTA) problem addresses the question of finding the task-to-robot assignments in order to achieve the overall system goals. MRTA problem aims to find an optimal or near optimal solution for the following question: "Which robot will be responsible to handle which task such that the overall cost of the team is to be minimized?".

Path planning is an essential issue in mobile robotics, which is to find a suitable collision-free path for a mobile robot to move from a start location to a target location. Very often this path is highly desirable to be optimal or near optimal with respect to time, distance or energy. Distance is a commonly adopted criterion. One of the difficulties of path planning is the undesired local minima such as trapping in some deadlock situations.

Chapter 2

Literature Review

The problem of Multi-Agent Pickup and Delivery (MAPD) Which is Commonly has two parts (i) task allocation, where the agent receives the appropriate task, and (ii) path planning, where the best path for the agent to perform its task, without colliding with other agents [1]. In this work, the approach proposed was an integer-encoded genetic algorithm for solving the task allocation part of the MAPD problem combined with two-path planning algorithms already known: the Prioritized Planning and the Improved Conflict-Based Search (ICBS). Performed computational experiments to solve MAPD simulating a real-world situation, to evaluate the two proposed representations, to analyze the performance of the GA when a heuristic is used to create its initial population The environment is a 21×35 grid, For each problem, 500 tasks must be concluded These tasks are randomly created with their task-endpoints (pickup and delivery locations). They proposed to extend the application of the proposed techniques to situations with a larger number of agents, different (and even more realistic) environments and considering additional constraints in the tasks.

In [2], they have designed two algorithms MCA and RMCA to solve the Multi-agent Pickup and Delivery problem where each robot can carry multiple packages simultaneously. MCA and RMCA successfully perform task assignment and path planning simultaneously. This is achieved by using the real collision-free costs to guide the multi task multi-robot assignment process. Further, we observe that the newly introduced anytime improvement strategy improves solutions substantially. The main ingredients of approach are a marginal-cost assignment heuristic and a meta-heuristic improvement strategy based on Large Neighborhood Search. As a further contribution, they also consider a variant of the MAPD problem where each robot can carry multiple tasks instead of just one.

This paper proposed a new hybrid meta-heuristic technique which is the Hybrid Most Valuable Player algorithm in order to solve combinatorial optimization problems [3]. The Most Valuable Player algorithm is hybridized with the GA by integrating the crossover operator into the original algorithm. A comparative study was performed between the proposed algorithm and two well-known algorithms, the GA and the Ant Colony Optimization, in solving the Multi-depot Vehicle Routing Problem. Multi-depot Vehicle Routing Problem is a variant from the Vehicle Routing Problem, where each vehicle starts from and returns to a separate depot from the others, without violating the maximum capacity of each. This problem can be used in various applications such as the routing of busses through known stations. The algorithms were compared by solving 10 problems, in terms of the solution quality, the computational time and the convergence speed. The shortages in this paper were min number of constraints to the problem and try to apply the solution of the problem on real life scenarios should be the focus.

In [4], path planning for multi cooperative robots was discussed using communication between them in the local area, so where any of them finds an obstacles it shares its position with other robots. This algorithm was implemented and optimized using ant colony technique in order to find the optimal path of swarm robots in a Manhattan grid.

A comparative study of Meta-heuristic algorithms was implemented for solving Unmanned Aerial Vehicles (UAVs) path planning in [5]. It was assumed that the flight environment is fixed and all obstacles and threats are known. The main aim was to find the shortest path length for the UAVs while flying into a high-threat area as safely as possible. The objective function in this problem was the distance travelled by the UAVs from the starting point to the end point taking into consideration the distance moved to avoid any obstacles . This function was subjected to constraint that the distance between the current position of UAVs and the obstacle should be greater that the radius of the UAVs.

As mentioned in paper [6], The Multi-Agent Path-Finding problem that seeks both task assignments and collision-free paths for a set of agents navigating on a graph, this approach based on two key ideas: (i) operate on a search forest rather than a search tree; and (ii) create the forest on demand, avoiding a factorial explosion of all possible task assignments. They provided extensive empirical results comparing CBS-TA to task assignment followed by Conflict-Based Min-Cost-Flow, and an integer linear program solution, demonstrating the advantages of our algorithm. Their results highlight a significant advantage in jointly optimizing the task assignment and path planning for very dense cases compared to the traditional method of solving those two problems independently. For large environments with many robots we show that the traditional approach is reasonable, but that we can achieve similar results with the same runtime but stronger sub optimality guarantees.

This paper [7] presented two novel offline MAPD algorithms that improve the MAPD problem online MAPD algorithm with respect to task planning, path planning, and deadlock avoidance for the offline MAPD problem, MAPD algorithms first compute one task sequence for each agent by solving a special traveling salesman problem and then plan paths according to these task sequences, also introduced an effective deadlock avoidance method, called "reserving dummy paths." Theoretically, their MAPD algorithms are complete for well-formed MAPD instances, a realistic subclass of all MAPD instances. Experimentally, they produce solutions of smaller makespans and scale better than the online MAPD algorithm in simulated warehouses with hundreds of robots and thousands of tasks. also may extend them to more real-world scenarios ,also consider agents with different velocities and tasks with different deadlines.

In a paper by Amar et al [8],three approaches for determining the optimal pathway of a robot in a dynamic environment were proposed. These approaches are; the Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and hybrid PSO and ACO. These used to carry out the path planning tasks effectively. A set of certain constraints must be met simultaneously to achieve the goals; the shortest path, the least time, and free from collisions. The results are calculated for the two algorithms separately and then that of the hybrid algorithm is calculated. The proposed algorithm has been proven the ability through a concatenation from simulation experiments. Empirical validation has been performed to validate the suggested navigation algorithm. Some comparative results are presented on the basis of simulation results to illustrate the efficiency and the feasibility of the proposed algorithm.This approach supplies the optimal path with minimum number of iterations.Comparing the obtained results from the PSO, ACO, and hybrid PSO & ACO shows that the best minimum path length and minimum consumption time achieved by applying the hybrid algorithm. While, the PSO algorithm perform better than the ACO algorithm.

A paper by Shareef et al[9], demonstrates a method for global path planning in a static setting using the grasshopper optimization algorithm (GOA). The bias factor is used to enhance this algorithm's performance and the path that results. An updated version of the multinomial logistic regression algorithm is used to better improve the path that is produced by this approach (MLR). Three distinct, sizable scenarios with varied levels of complexity were used to assess the algorithms. Using the identical surroundings, the ACO and the GOA algorithm have been compared.The results of the studies demonstrated that their algorithm enhanced cost and time convergence. The experiments have also shown a shorter path than ACO using regression. The new global GOA path planning version may be used in dynamic contexts in the future.

This Paper[10] studied the lifelong variant of MAPF, where agents are constantly engaged with new goal locations, such as in large-scale automated warehouses. They propose a new framework Rolling-Horizon Collision Resolution (RHCR) for solving lifelong MAPF by decomposing the problem into a sequence of Windowed MAPF instances, where a Windowed MAPF solver resolves collisions among the paths of the agents only within a bounded time horizon and ignores collisions beyond it. RHCR is particularly well suited to generating pliable plans that adapt to continually arriving new goal locations, showed how to transform several regular MAPF solvers to Windowed MAPF solvers. They demonstrated its scalability up to 1,000 agents while also producing solutions of high throughput. Compared to other Method RHCR not only applies to general graphs but also yields better throughput. Overall, RHCR applies to general graphs, invokes replanting at a user specified frequency, and is able to generate pliable plans that cannot only adapt to continually arriving new goal locations but also avoids wasting computational effort in anticipating a distant future.

Chapter 3

Methodology

3.1 Problem Formulation

Consider that multiple mobile manipulators need to finish a set of tasks, where the position of each task is known. The mobile manipulators need to execute these tasks without collisions between each other. The problem is divided into two sub-problems, first assigning the tasks to each robot,then finding the near-optimal path for each robot in terms of shortest distance from the start to the position of the final task passing through the set of tasks assigned for each robot in it's way.

3.1.1 Mathematical Formulation

We use $P = \{1, 2, ..., n\}$ where P denotes the set of indices of n randomly distributed tasks that need to be executed according to their corresponding locations. Each task $i \in P$ is associated to a given tuple(x_i, y_i) which represents the location of a task. $R = \{1, 2, ..., m\}$ where R denotes set of indices of $m > 1$ robots that are initially located at dispersed depots in form of tuple (x_k, y_k).It is assumed that the robot can be assigned to a max of C tasks. Let $n_k <= C$ be the number of tasks assigned to the $robot_k \in R$ at time instant t and $p_k(t)$ be the position of $robot_k$ at time instant t. Let σ_{ik} be the path planning mapping that maps the index $i \in P$ of the task to index $k \in R$ of the k_{th} robot to a binary value which equals 1 if and only if if task *i* is assigned to robot and 0 otherwise.

Regarding the task allocation problem, tasks will be assigned to their corresponding robots with objective being that the total time of travel(estimated) and execution time of the tasks is approximately equally shared among all robots.The travel time is estimated from the distance travelled where$Time = \frac{TravelledDistance}{V_{Avg}}$ and average velocity of all the robots is assumed to be constant and equals to $1 cm/sec$ so we can conclude that R$Time = TravelledDistance$.This is evaluated by calculating the smoothing index formula relating the total execution time of each robot to the maximum time executed by a robot. During task allocation the robots are constrained to have a maximum number of tasks C and each task must be executed by one robot only. Decision variable obtained from this step is the set of tasks assigned to each robot with their order of execution(as they appear in the array).the problem is explained mathematically as follows.

$$SI = (\sqrt{\frac{\sum_{k=1}^{m}(RL_k - RL_{max})^2}{m}})(\alpha) + (d_{estimated})(1-\alpha) \tag{3.1}$$

Where $d_{estimated}$ is the sum of estimated distances covered by all robots and α is factor determining the ratio of consideration between the smoothing index and the estimated distance Equation 3.1 is subjected to:

$$n_k <= C \tag{3.2}$$

$$\sum_{k \in R} \sigma_{ik} = 1, \forall i \in P \tag{3.3}$$

Where Equation 3.1 represents smoothing index discussed before to be our objective function of which should be minimized .RL_k represents the kth robot working load in terms of estimated time of travel and execution of tasks, and RL_{max} represent the maximum work load across all robots .m represent total number of robots. equation(1.2) states that the number of tasks assigned to k_{th} robot should not exceed the maximum number of tasks determined for each robot .Finally , equation (1.3) ensures that each task is assigned to only 1 robot to be executed by.

Regarding the cooperative path planning problem, our objective is to get the near optimal path in terms of distance for each robot to get to the required tasks while avoiding collisions with other robots.This objective can be formulated by the following equation.

$$d_{total} = \sum_{i=1}^{k}\sum_{i=1}^{N}\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \tag{3.4}$$

constrained to:

$$p_k(t) \neq p_w(t), \forall k, w \in R \tag{3.5}$$

Where equation (1.4) represents our objective(cost) function that should be minimized in order to obtain the shortest path taken by the robot to execute all tasks assigned to it , where N represents the number of points in the trajectory of the robot. Equation (1.5) ensures that collisions between robots is avoided where positions of robots *k* and *w* can not be equal at any instance of time.

3.2 Algorithms

3.2.1 Simulated Annealing (SA)

Simulated Annealing (SA) is an optimization method which mimics the slow cooling of metals. It is a probabilistic approach for estimating a function's global optimum. It is a trajectory based meta-heuristic optimization algorithm that is used to approximate global optimization in a vast search space for an optimization issue. In the context of optimization, the minimum of the objective function represents the minimum-energy state of the system. Transition Probability is the probability of moving from the current solution to a new solution.
Illustration of SA algorithm is as follows

Initially

- Initialize the optimization problem parameters:
  - No. Of Robots/Tasks and Execution time of each task
  - Max no. of tasks for each robot
  - Initial/Current/Final Temperatures
  - Beta for linear cooling and Alpha for geometric cooling
  - In **Path Planning** problem, we will get the best Task Allocation solution from the TA algorithm as an input to the Path Planning problem
- Generate initial solution
- Check feasibility:

  If not feasible keep generating a new solution till finding a feasible one
- Assign best cost, current solution, and initial best solution from that initial solution

Repeat till the maximum number of iterations is reached

- Modify the current solution
  - In case of Task Allocation, modify the solution by randomly selecting a task then assign to that task a randomly selected robot
  - In case of Path Planning, modifying the solution is done by adding or subtracting a random noise to the randomly generated points between tasks
- Check feasibility:

  If not feasible keep generating a new solution till finding a feasible one
- If new solution is better than current solution, directly accept the solution
- If not better, randomly generate a number [0,1] and compare it to the probability
- If $p > r$, accept the solution
- Update current sol and current temperature
- Update the best cost and best solution based on the current solution

3.2.2 Genetic Algorithm (GA)

GA is a population based meta-heuristic search algorithm. It is based on the ideas of natural selection and genetics. It is commonly used to generate high-quality solutions for optimization problems and search problems. GA simulates the process of natural selection. It simulates "survival of the fittest" among individual of consecutive generation for solving a problem. Illustration of GA algorithm is as follows

Initially

- Initialize the optimization problem parameters:
  - No. Of Robots/Tasks and Execution time of each task
  - Max no. of tasks for each robot
  - Population size
  - Percentages of Elite, CrossOver, and Mutation members
  - Alpha used only in CrossOver in **Path Planning** problem to generate new feasible children
  - In **Path Planning** problem, we will get the best Task Allocation solution from the TA algorithm as an input to the Path Planning problem
- Generate initial generation
- Check feasibility:

  If not feasible keep generating a new solution till finding a feasible one
- Sort the population based on the cost of each chromosome
- Get the initial best cost and initial best solution from the first element of the population (first elite element)

Repeat till the maximum number of iterations is reached

- Modify the population using CrossOver and Mutation
  - CrossOver in Task allocation problem is based on 1-point CrossOver while in Path Planning is based on Whole Arithmetic Recombination using Alpha
  - Mutation in Task allocation problem is based on swap mutation while in Path Planning is based on modifying the randomly generated points by adding or subtracting a random noise to the points
- Calculate required number of Crossover Children and Mutation Children
  - Choose 1 parent from the elite list and the other parent is randomly selected from the rest of the population and apply CrossOver
  - Check feasibility until the two children are feasible solutions
  - For mutation, keep randomly modifying the solution and append the solution to the mutation list when it is feasible
- Append the solutions obtained from the above step to the CrossOver and Mutation lists
- Sort the current population based on the cost to update the elite, medium (neither elite nor mutation members), and mutation lists
- Update the best cost and best Task Allocation solution based on the updated generation

Chapter 4

Results

The general goal of the algorithms was to get the optimum task allocation between multiple robots and find optimum path planning between tasks by considering obstacles.

4.1 Simulated Annealing

Initially, a simple experiment is conducted where 4 tasks are to be executed using 2 mobile robots with maximum number of tasks for each robot is 2. Regarding the task allocation problem, linear cooling is chosen with $i_{max} = 1000$ and $T_{initial} = 500$
Figure 4.1 shows the linear decaying of temperature, Figure 4.2 shows the decay of the best cost of the task allocation problem. Since it is a simple experiment, the cost immediately decays and converges.

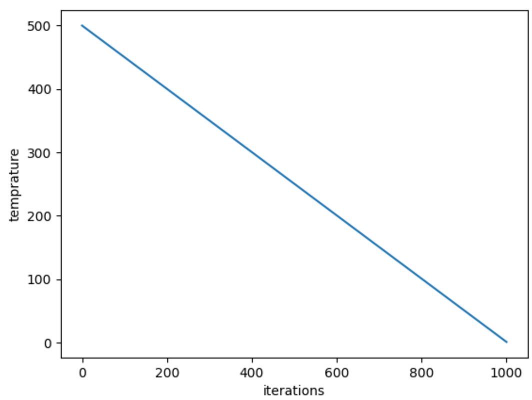

Figure 4.1: Temperature decay using linear cooling in Task Allocation

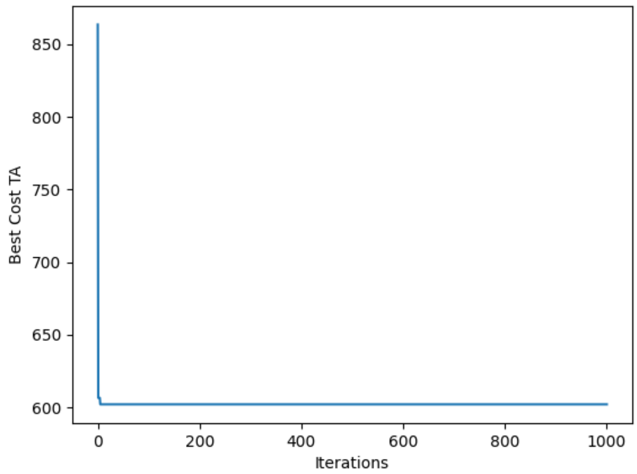

Figure 4.2: Best Cost Task Allocation

For the path planning problem, geometric cooling is chosen with $i_{max} = 800$, $i_{sub-iteration} = 10$, $\alpha = 0.95$,$T_{initial} = 100000$ and number of randomly generated points between tasks is 2 $(N = 2)$
Figure 4.3 shows the geometric decaying of temperature, Figure 4.6 shows the decay of the best cost of the path planning problem with obstacles.

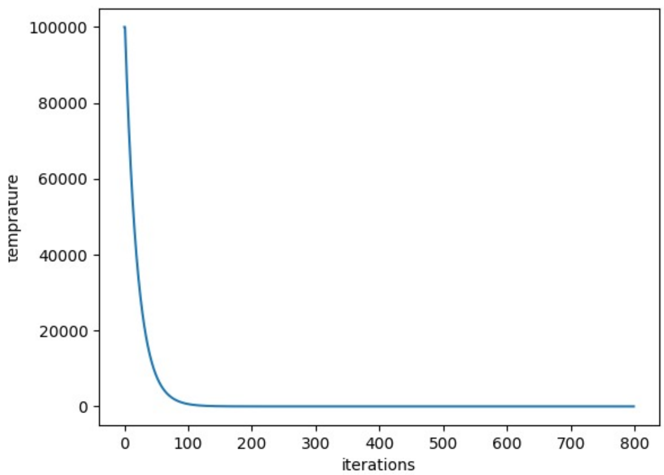

Figure 4.3: Temperature decay using geometric cooling in Path Planning

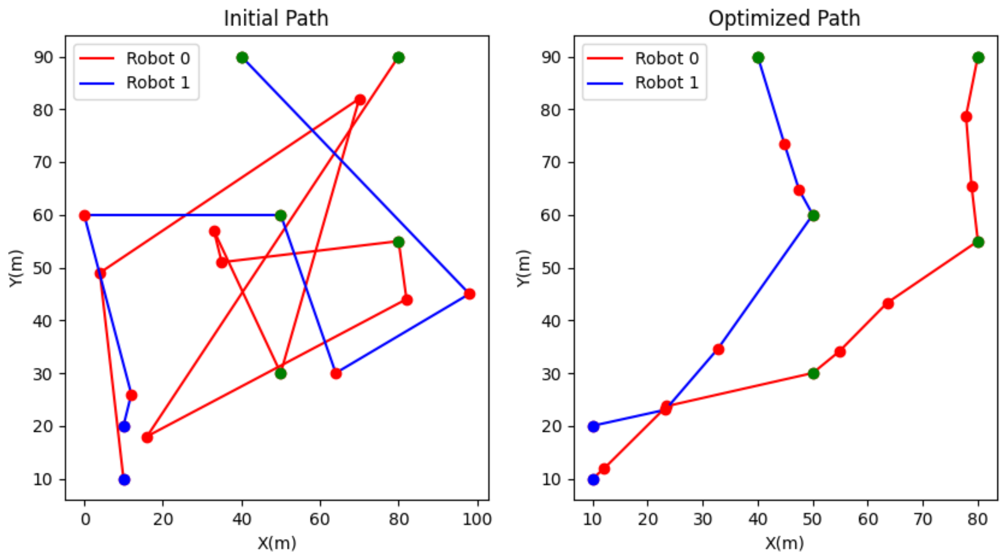

Figure 4.4: First experiment in Path Planning

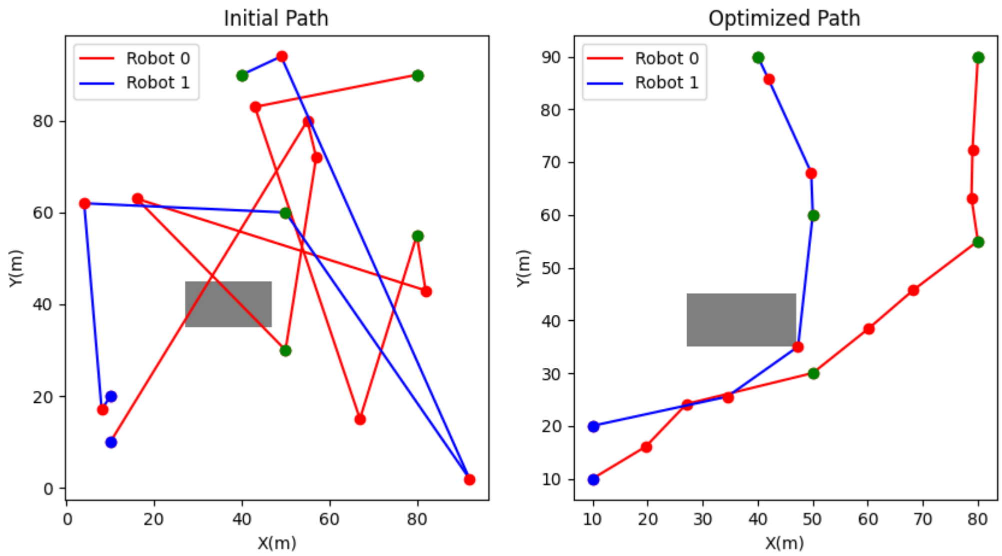

Figure 4.5: First experiment in Path Planning with obstacles

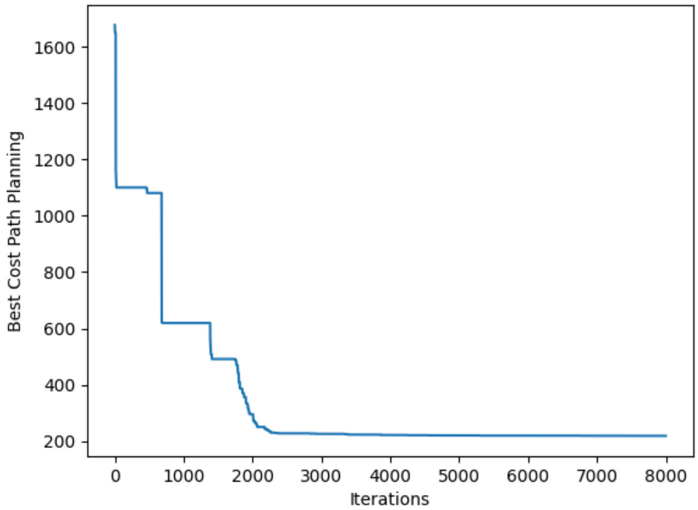

Figure 4.6: Best Cost of first experiment in Path Planning

In another experiment, we have to execute 9 tasks using 2 mobile robots with and without obstacles. Regarding the task allocation problem, linear cooling is chosen with $i_{max} = 1000$ and $T_{initial} = 500$ similar to the previous experiment.

For the path planning problem, geometric cooling is chosen with $i_{max} = 800$, $i_{sub-iteration} = 10$ and $T_{initial} = 100000$ also similar to the previous one.

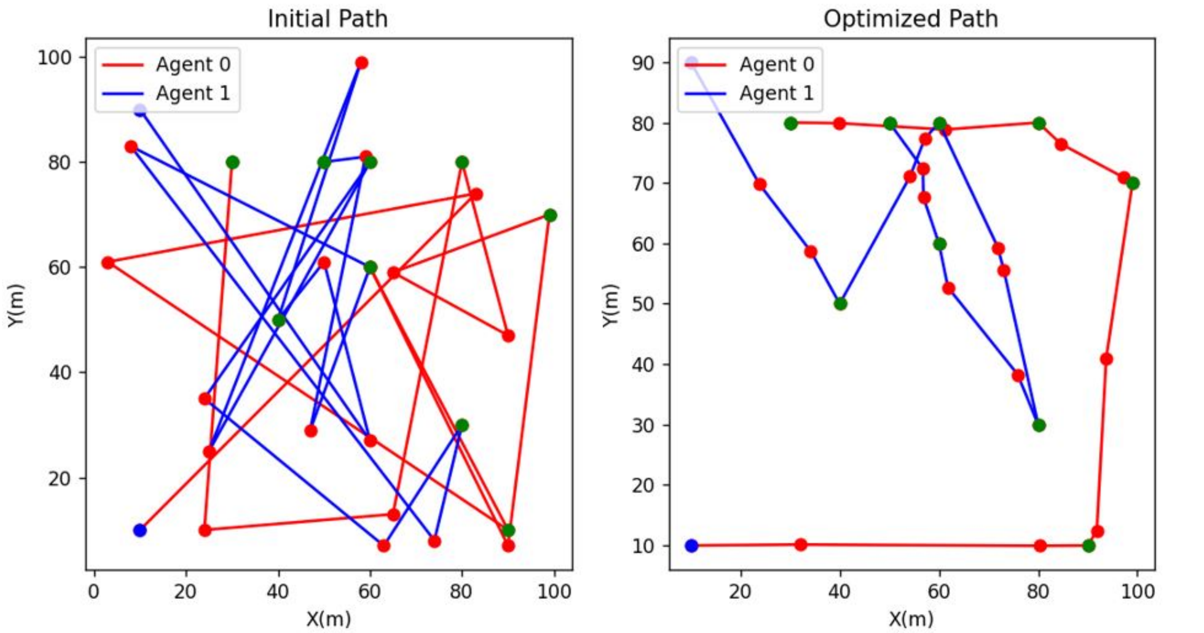

Figure 4.7: Path Planning without obstacles

In Figure 4.7, the second robot takes the task at (80,30) however it is in the path of the first robot due to the smoothing index.
The same experiment was conducted but with an obstacle. The results of the path planning was as shown in Figure 4.8

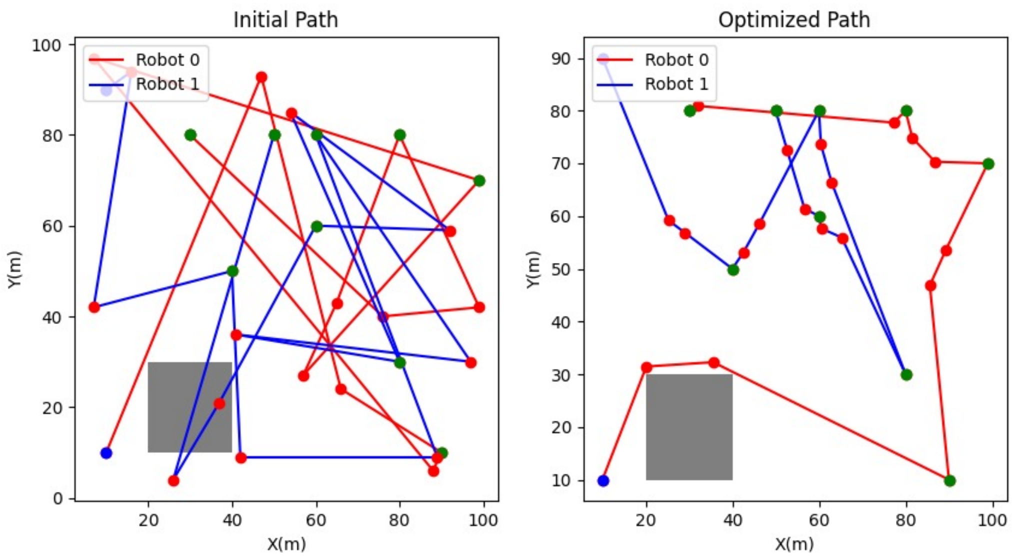

Figure 4.8: Path Planning with obstacles

The decay of the best cost of this path planning experiment with obstacles is as shown in Figure 4.9.

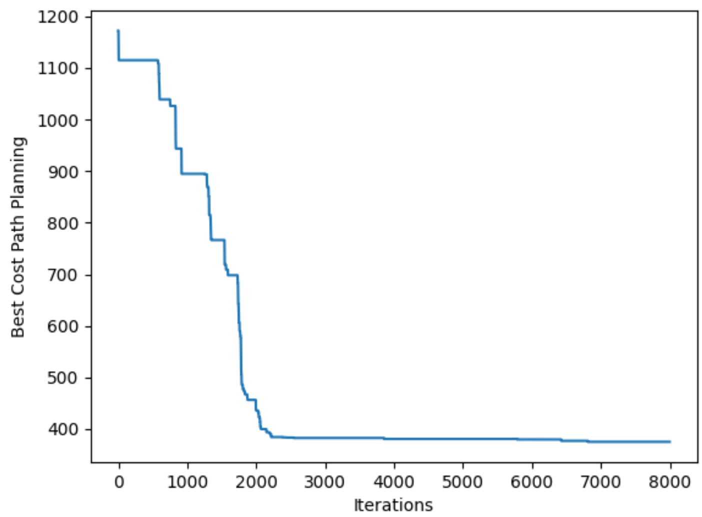

Figure 4.9: Best Cost Path Planning

4.2 Genetic Algorithm

Conducting the same problem in Figure 4.8 using GA with population size = 3000 , $i_{max} = 10$ and Pecentages of elite,crossover and mutation are 0.1,0.8 and 0.1 respectively.

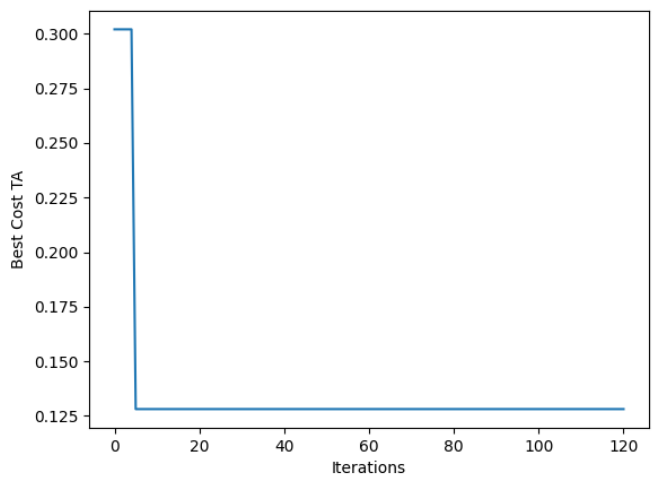

Figure 4.10: Best cost for Task allocation using GA

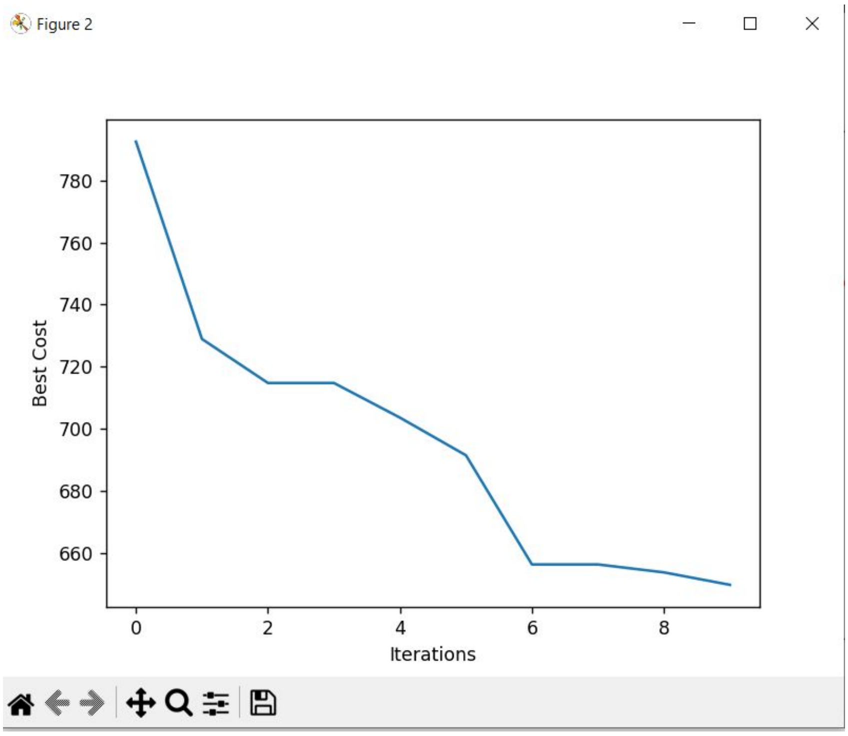

Figure 4.11: Path Planning Best Cost with obstacles

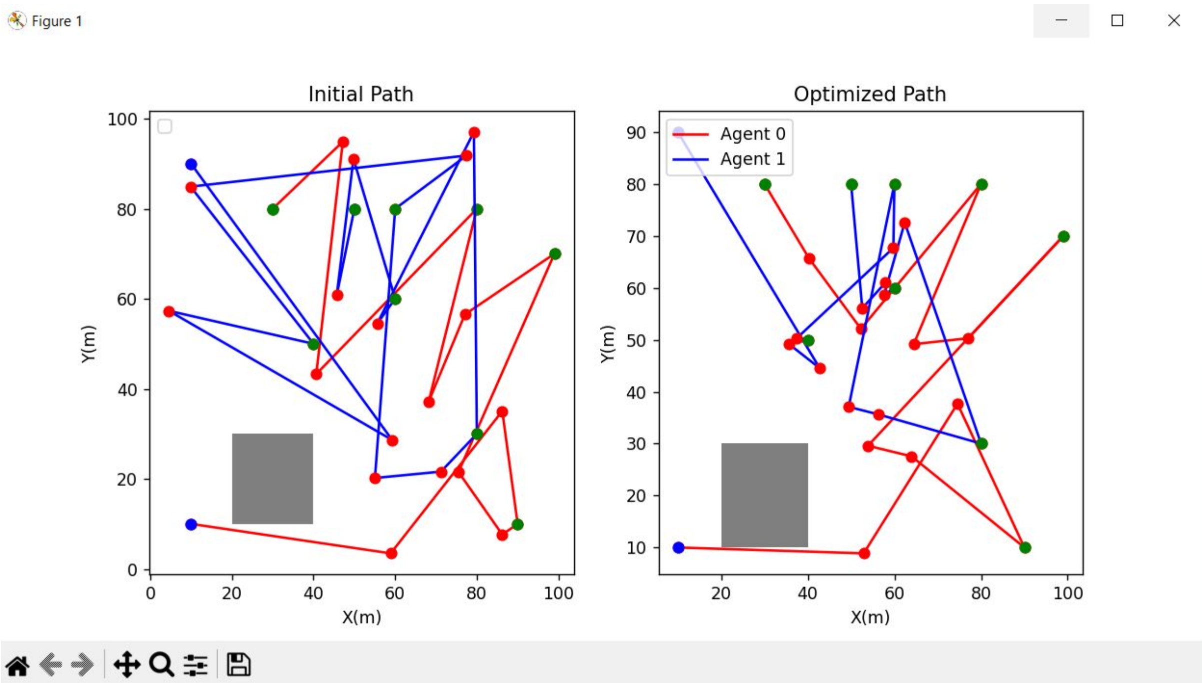

Figure 4.12: Path Planning with obstacles

Where Initial cost = 1200 and Final cost = 650

4.3 Performance Metrics

Key Performance Indicators (KPIs) are used to measure and compare between different algorithms in terms of convergence and indices. Here we calculated the average,standard deviation and coefficient of variation values for the path planning problem for both algorithms (SA,GA). The following results are obtained after running the first experiment shown in Figure 4.5 many times using SA.

Table 4.1: Simulated Annealing Performance

Run	Best Cost
1	269.8201173509318
2	243.80566806578867
3	247.73388452452633
4	249.20875877197187
5	248.87050678190695

The average,standard deviation and coefficient of variation values of SA algorithm are as shown in Figure 4.13

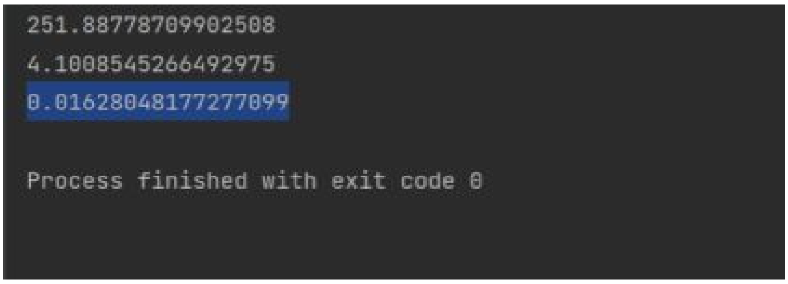

Figure 4.13: Average,Standard deviation and Coefficient of variation values

Regarding the GA performance evaluation, it needs so much time for a single run for the GA algorithm (Task allocation and Path planning) and also the GA algorithm still needs some improvements, so it will be provided in the next milestone

Bibliography

[1] Ana Carolina LC Queiroz, Heder S Bernardino, Alex B Vieira, and Helio JC Barbosa. Solving multi-agent pickup and delivery problems using a genetic algorithm. ., pages 140–153, 2020.

[2] Zhe Chen, Javier Alonso-Mora, Xiaoshan Bai, Daniel D Harabor, and Peter J Stuckey. Integrated task assignment and path planning for capacitated multi-agent pickup and delivery. *IEEE Robotics and Automation Letters*, 6(3):5816–5823, 2021.

[3] Youmna Magdy, Omar M Shehata, Mohamed AbdelAziz, Maged Ghoneima, and Farid Tolbah. Metaheuristic optimization in path planning of autonomous vehicles under the atom framework. ., pages 32–37, 2017.

[4] Li Yong, Li Yu, Guo Yipei, and Cai Kejie. Cooperative path planning of robot swarm based on aco. In *2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pages 1428–1432. IEEE, 2017.

[5] Soheila Ghambari, Julien Lepagnot, Laetitia Jourdan, and Lhassane Idoumghar. A comparative study of meta-heuristic algorithms for solving uav path planning. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 174–181. IEEE, 2018.

[6] Wolfgang Hönig, Scott Kiesel, Andrew Tinka, Joseph Durham, and Nora Ayanian. Conflict-based search with optimal task assignment. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*, 2018.

[7] Minghua Liu, Hang Ma, Jiaoyang Li, and Sven Koenig. Task and path planning for multi-agent pickup and delivery. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2019.

[8] Lina Basem Amar and Wesam M Jasim. Hybrid metaheuristic approach for robot path planning in dynamic environment. *Bulletin of Electrical Engineering and Informatics*, 10(4):2152–2162, 2021.

[9] Asmaa Shareef and Salah Al-Darraji. Grasshopper optimization algorithm based path planning for autonomous mobile robot. *Bulletin of Electrical Engineering and Informatics*, 11(6):3551–3561, 2022.

[10] Jiaoyang Li, Andrew Tinka, Scott Kiesel, Joseph W Durham, TK Satish Kumar, and Sven Koenig. Lifelong multi-agent path finding in large-scale warehouses. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11272–11281, 2021.