Multi-objective optimization for Multi-Robot Path Planning on warehouse environments*

Enol García González[1], José R. Villar[1], Camelia Chira[2], Enrique de la Cal[1], Luciano Sánchez[1], and Javier Sedano[3]

[1] University of Oviedo, Computer Science Department, Oviedo, Spain
{garciaenol,villarjose,delacal,luciano}@uniovi.es
[2] University of Babes Boliay, Department of Computer Science, Cluj-Napoca, Romania
camelia.chira@ubbcluj.ro
[3] Instituto Tecnológico de Castilla y León, Burgos, Spain
javier.sedano@itcl.es

Abstract. Today, robots can be found in almost any field. Examples include robots for transporting materials in hospitals and warehouses, surveillance, intelligent laboratories and space exploration. Whatever the reason for moving the robot and whatever its location, all robot applications anywhere require path calculation. In this paper, we address the problem of collision-free path planning in multirobot environments, known as Free Multi-Robot Path Planning (MPP). In this paper we propose a novel approach to solve the MPP problem using multi-objective optimization, for which we define two functions that has to be minimized. In experimentation, it is compared with previous approaches to the problem, improving them in some scenarios. Finally, new lines of research are proposed to improve this path calculation problem using multi-objective optimization and to address new and more complex problems in warehouse environments.

Keywords: Multi-Robot Path Planning, NSGA, Multi-objective optimization

1 Introduction

Nowadays it is common to use robots for the automation of multiple transport tasks. We can find robots automating transport tasks in different environments, from hospitals [1, 13, 18] to large logistics centers [3, 9, 17, 19]. In many of these spaces, the transport tasks are not performed by a single robot, but multiple robots must collaborate in the same space and move around without colliding

* This research has been funded by the Spanish Ministry of Economics and Industry, grant PID2020-112726RB-I00, by CDTI projects CER-20211003 and CER-20211022, by Missions Science and Innovation project MIG-20211008 (INMERBOT), and by ICE (Junta de Castilla y León) under project CCTT3/20/BU/0003. Also, by Principado de Asturias, grant SV-PA-21-AYUD/2021/50994.

with each other. The problem of determining the best path for a set of robots in a shared space without collisions or human intervention is known as Free Multi-robot Path Panning (MPP).

This MPP problem has applications in multiple fields that can be very different from each other, such as surveillance [22], intelligent laboratories [21, 23], or space exploration [12]. In current work, the MPP problem is applied in the field of warehouse logistics [3, 9, 17, 19]. The objective of this study is to provide a method that calculates the route to be followed by robots within a known environment in which the location of obstacles and the start and end points of each robot are known.

In the literature, there are two main approaches to solving the MPP problem. The first of them is a heuristic approach, which is based on A*. A* [10] is a search algorithm that has been widely used for route calculation in single-robot environments, but it does not consider the existence of multiple robots in a shared space. Some authors have developed heuristic algorithms like Windowed HCA* [20], D*Lite [16], Field D* [6] or ThetA* [4] that are based on A* and extend it to solve the path planning problem when there are multiple robots. The other approach that is largely present in the literature is the use of metaheuristics. Within this approach, we find works such as the one presented in [2] that solve the problem using Differential Evolution. Other works of this approach are [8, 24], which solve the problem with other bio-inspired metaheuristics such as Ant Colony Optimization and Grey Wolf Optimizer.

In this research, we propose a new approach to this problem based on multi-objective optimization. Our goal is to improve our previous proposals to solve this problem by employing co-evolutionary algorithms [7, 15].

The structure of this document is as follows. Section 2 explains the proposed approach for solving MPP. Next, Section 3 gives details of the experimentation, while Section 4 shows the results and discussion. Finally, Section 5 draws the conclusions extracted from this study and presents future research lines on this topic.

2 Non-Dominated Genetic Algorithm approach

This proposal focuses on solving the MPP problem with a multiobjective optimization approach. To do so, it makes use of a metaheuristic well-known in the literature: NSGA-III [5, 14]. This metaheuristic is based on genetic algorithms since their behavior is very similar. Like genetic algorithms, it starts from an initial population and performs an iterative process, in which, in each iteration, new solutions will be generated using crossover and mutation operators.

The difference between NSGA-III and genetic algorithms are the number of criteria. Classical genetic algorithms minimize a single criteria called fitness, while NSGA-III have many objectives to minimize. Having many criteria changes how we determine whether one solution is better. To consider that a solution dominates another one, its value must be strictly lower in all the criteria except one in which it can be less or equal. For this work, two objectives have been defined to be minimized:

- Length of the paths. That consist on the number of movements on the path.
- Number of collisions. Taking into account that X_i^t is the position of robot i at time t. We can define that a collision occurs if:
  - Two robots i, j share a tile at the same moment of time. $X_i^t = X_j^t$
  - Two robots i, j exchange positions at two consecutive instants of time. $X_i^t = X_j^{t-1}$ and $X_i^{t-1} = X_j^t$

To minimize those two objectives using NSGA-III, its necessary to define three behaviors: how to initialize the first population and how to perform the crossover and mutation operator. To generate the initial population. It is proposed that we start from a base of k pre-generated routes –being k a parameter of the algorithm–, in a pseudo-random way and that the solutions are formed by indicating which of the already calculated routes is used. The following sections indicate how these routes are generated and how the behaviors required by NSGA-III will be implemented.

2.1 Route Generation

The generation of routes will be performed randomly by means of an iterative algorithm in which in each iteration the movement to be performed will be selected pseudo-randomly (Alg. 1).

This algorithm is separated into two parts: i) generation of the initial routes, and ii) loop elimination (Alg. 2). In the generation of routes, the movements are chosen randomly from all possible movements at that moment, but taking into account two restrictions:

- If it is possible to redo the last move, it will have a higher probability of being chosen to encourage routes to incorporate straight-line sections.
- The movement opposite to the previous one can only be chosen if there is no other possible movement to avoid the algorithm getting stuck between two points.

Once the initial path is generated, it is analyzed and two time instants are searched for in which the robot passes through the same coordinates. If the robot passes through the same coordinate in two different t instants, it is considered that there is a loop and all the movements between the two appearances of those coordinates in the path are eliminated.

To analyze the feasibility of this algorithm to generate routes, the algorithm has been run 30 times to generate 1000 routes in each run. During this test, the algorithm was able to generate an average of 930.92 solutions per second, with a standard deviation of 30.31.

Algorithm 1 Random search O(n)

Input: *origin*, *dest* points
Output: *path*: Sequence of movements to reach the point *dest* from the point *origin*.

```
current ← origin
path ← ∅
while current ≠ dest do
    if path is not empty & is_valid_movement(path[-1]) & random < 0.7 then
        Add path[-1] to path again        ▷ Copy the last movement
    else
        Calculate available_movs
        if length(available_movs) == 1 then
            Add available_movs[0] to path
        else
            Remove oposite(path[-1]) from available_movs
            next_movement ← random(available_movs)
            Add next_movement to path
        end if
    end if
    Update current position
end while
Look for loops on path and remove them using Alg. 2
return path
```

2.2 Initial population

The representation of the solutions is defined as a vector of n positions, where n is the number of robots in the problem. Each position in the vector will contain an integer representing which index of the pre-generated paths will be used for that robot. Figure 1 shows an example of this representation, where robot number 1 will use the path at position 33 in the list of pre-generated paths, the second robot will use path 15 and the last robot will use the path at position 24.

33	15	24

Fig. 1: Example of the representation of a single solution

The solutions will be generated randomly so that each position of the vector will contain a random integer generated between 0 and the number of precomputed routes.

2.3 Crossover

For the crossover operator we have chosen to use the One-Point Crossover[11]. This operator selects a random cut point in the two parent solutions and gener-

Algorithm 2 Remove loops O(n^2)

Input: *path*: sequence of movements and points for a robot that could contain loops
Output: *path*: Sequence of movements and points for a robot without loops
  for $i \leftarrow 0$ To $length(path)$ **do**
    for $j \leftarrow length(path)$ To i Step -1 **do**
      if coords_at(i) == coords_at(j) **then** ▷ If a coord appears twice in the route
        $new\_path \leftarrow path[0:i] + path[j:length(path)]$
        return remove_loops(new_path)
      end if
    end for
  end for
  return path

ates two daughter solutions in which each part will come from one of the parents. Figure 2 illustrates this crossover operator.

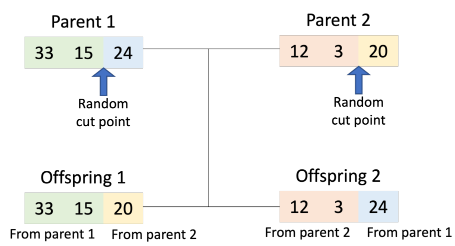


Fig. 2: Example of crossover operator

2.4 Mutation

The mutation operator will act on each of the elements of a solution. This means that each of the numbers present in the vector forming an individual will be evaluated and modified with a probability of p_m. The modification to be made in the mutation operator will be to change the value to a random one. Figure 3 illustrates this mutation operator.

3 Experimentation Setup

The objective of the experimentation is to compare with other methods already published in previous works to solve the MPP problem [15, 7]. The comparison replicates the three scenarios proposed in [7]: the first one with a single room and fixed obstacles; the second one with 4 connected rooms; and the third one that replicates a warehouse.

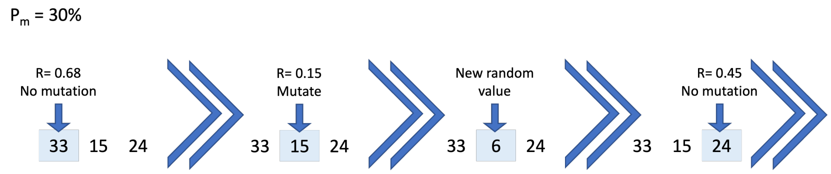
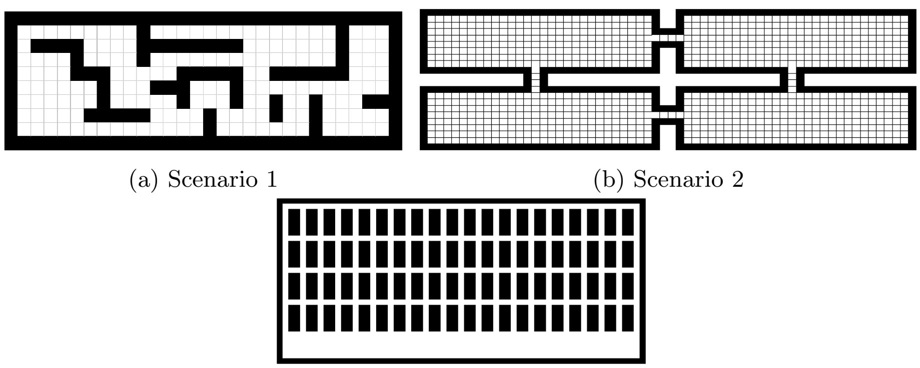


Fig. 3: Example of mutation operator



(a) Scenario 1 (b) Scenario 2

(c) Scenario 3

Fig. 4: Graphical representation of the scenarios used in the experimentation

Figure 4a shows the first scenario consisting of a closed room of size 10x30 meters. In this first scenario, there are only a few fixed objects inside the room. The objective in this scenario is for the robot to go from any random starting point to any random ending point. The number of robots used in this experiment varies between 3 and 15.

The second scenario, shown in Figure 4b, consists of 4 rooms of the same size as the first scenario. In this case, the rooms are free of obstacles, but connected by narrow corridors of length 4, in which only one robot fits. The complication of this scenario is that the robots are required to end up in a different room from the starting room, in order to analyze the behavior of the algorithm when crossing corridors. For this experiment, the number of robots varied between 3 and 6 robots per room.

Figure 4c shows the last scenario, which replicates a warehouse in which there are two clearly differentiated zones. The large area at the top with a pattern of aisles and square obstacles corresponds to the storage area of the workshop. The square obstacles would be the shelves where the products are stored. Analogous to the second scenario, only one robot is considered to fit in each aisle. The smaller area at the bottom of the map with no obstacles inside is considered a working area for the robots to move around in as needed. The goal of this experiment is for the robots to move from a random point in the work zone to a

Table 1: Parameters used during the experimentation.

Crossover probability	0.7
Mutation probability	0.3
Max number of iterations	200
Population size	50
Pre-generated routes (k)	100

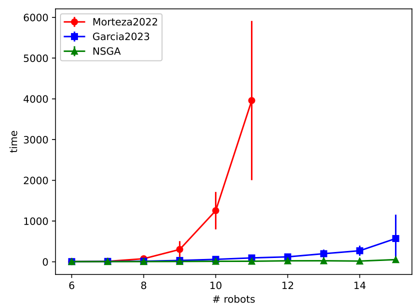


Fig. 5: Comparative graphs of the evolution of the execution time concerning the number of robots for each method for scenario 1.

random point in the shelving area. In this scenario, the experimentation started with 10 robots and was increased by 5 to reach 30 robots.

Up to 10 runs of each algorithm and scenario will be run to estimate the statistics of the performances. During the experimentation, the parameters in Table 1 were used. These parameters were chosen among the best after several runs with different parameters.

4 Results and discussion

Three metrics were used during the experimentation: the execution time, the length of the longest path and the sum of the lengths of all paths.

Table 2 shows the runtime results compared to the two previous works: [15] and [7]. The new method obtains better results in the first scenario, since for 15 robots it produces results up to ten times better. Figure 5 graphically represents the evolution in execution time as a function of the number of robots for these three methods. In the case of the last two scenarios, it is observed that the proposed method does not perform as well, since it has much longer execution times than [7]. However, it does improve on [15], as it is able to solve these scenarios with almost any number of robots.

The other two metrics corresponding to the length of the longest path and the aggregated length are shown in Table 3.

The result from the experimentation shows that the new method gives better results in both time and path length when we are in a simple environment with few robots like in scenario 1. In more complex scenarios –scenario 2 and scenario

Table 2: Mean (MN) and standard deviation (SD) of the path-planning time in seconds for each method and scenario. The gray cells represent the experiments for which the method could not find a collision-free path.

	No. robots	Morteza2022[15]		Garcia2023[7]		NSGA-III	
		MN	SD	MN	SD	MN	SD
Scenario 1	6	3.8313	1.4861	4.7479	3.5449	4.9712	0.3330
	7	11.2127	5.4879	9.6416	4.3064	5.3837	0.4022
	8	75.8119	41.5637	14.3116	8.0995	5.7193	0.3323
	9	303.3389	203.8299	32.8010	12.7689	8.3554	4.6203
	10	1255.6156	459.5965	59.7161	24.0424	12.0660	6.6266
	11	3958.5166	1954.4560	95.2412	46.1279	13.7042	8.0799
	12			121.2675	72.5481	23.9644	0.8502
	13			198.7366	101.0648	25.1462	0.2680
	14			271.5449	124.6219	27.6918	1.0077
	15			571.3999	585.3738	53.2227	0.5877
Scenario 2	3 per room	41384.1716	14851.4483	0.3939	0.0663	5.4058	0.5484
	4 per room			0.2327	0.0536	14.7859	1.5257
	5 per room			0.4368	0.0162	60.5659	1.2013
	6 per room			1.2399	0.0183		
Scenario 3	10			0.1558	0.0106	13.5677	2.2207
	15			0.1999	0.0524	23.9452	3.1946
	20			0.1730	0.0121	36.2394	4.1399
	25			0.1810	0.0160	52.8438	4.9216
	30			0.2105	0.0081	69.6164	5.5152

3–, with a larger number of robots, the new approach takes longer to find results, reaching paths that are longer than those provided by the method with which we are compared.

5 Conclusion and future work

This paper proposes a new approach to the MPP problem. In the experimentation, we compare with another work that tries to solve the same problem using co-evolutionary algorithms. The newly proposed method outperforms the previous one in simple scenarios with few robots, but it is not able to beat the benchmark method in more complicated environments with more robots.

Future work will try to improve this approach in more complex scenarios, and we will combine the MPP and the TSP problem. To improve this approach, we will consider two possible lines: i) the improvement of the path generation algorithm, and ii) the development of a local search algorithm that obtains new solutions from those with few collisions. Regarding the combination of the MPP with other known problems in the literature, we will try to solve the MPP problem in environments where there is not only one starting point and one destination point; the main idea is that the robot must visit multiple waypoints before reaching the destination.

References

1. Causse, O., Pampagnin, L.: Management of a multi-robot system in a public environment. In: Proceedings 1995 IEEE/RSJ International Conference on Intelligent

Table 3: Max and aggregated length (in meters) of the paths obtained from each method for each scenario according to the number of robots. The gray cells represent the experiments for which the method could not find a collision-free path.

	No. robots	**Garcia2023[7]**		**NSGA-III**	
		Max	Aggregated	Max	Aggregated
Scenario 1	6	55.0	363.9	32.0	120.0
	7	56.4	377.1	32.0	146.2
	8	57.2	438.5	32.0	174.6
	9	60.5	389.1	36.0	195.2
	10	59.2	444.5	50.2	248.3
	11	68.9	414.7	50.6	300.7
	12	63.1	426.1	52.1	338.8
	13	79.0	437.8	54.3	387.6
	14	78.4	469.8	62.0	412.2
	15	90.8	486.0	70.1	489.0
Scenario 2	3 per room	77.0	736	76.0	788.3
	4 per room	79.0	951.0	82.2	920.2
	5 per room	79.0	1206.0	88.0	1336.9
	6 per room	79.0	1458.0		
Scenario 3	10	62.0	399.0	70.0	415.1
	15	62.0	536.0	140.2	740.9
	20	62.0	639.0	236.7	1339.5
	25	62.0	808.0	248.6	1662.3
	30	62.0	946.0	253.3	1810.8

Robots and Systems. Human Robot Interaction and Cooperative Robots, vol. 2, pp. 246–252 vol.2 (1995). DOI 10.1109/IROS.1995.526168

2. Chakraborty, J., Konar, A., Jain, L.C., Chakraborty, U.K.: Cooperative multi-robot path planning using differential evolution. Journal of Intelligent & Fuzzy Systems **20**, 13–27 (2009). DOI 10.3233/IFS-2009-0412
3. Chen, X., Li, Y., Liu, L.: A coordinated path planning algorithm for multi-robot in intelligent warehouse. In: 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 2945–2950 (2019). DOI 10.1109/ROBIO49542.2019.8961586
4. Daniel, K., Nash, A., Koenig, S., Felner, A.: Theta∗: Any-angle path planning on grids. Journal of Artificial Intelligence Research **39**, 533–579 (2010). DOI 10.1613/jair.2994
5. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. IEEE Transactions on Evolutionary Computation **18**, 577–601 (2014). DOI 10.1109/TEVC.2013.2281535
6. Ferguson, D., Stentz, A.: Using interpolation to improve path planning: The field d* algorithm. Journal of Field Robotics **23**, 79–101 (2006). DOI 10.1002/rob.20109
7. García, E., Villar, J.R., Tan, Q., Sedano, J., Chira, C.: An efficient multi-robot path planning solution using a* and coevolutionary algorithms. Integrated Computer-Aided Engineering **30**, 41–52 (2023). DOI 10.3233/ICA-220695
8. Gul, F., Rahiman, W., Alhady, S.S.N., Ali, A., Mir, I., Jalil, A.: Meta-heuristic approach for solving multi-objective path planning for autonomous guided robot using pso–gwo optimization algorithm with evolutionary programming. Journal of Ambient Intelligence and Humanize Computing **12**, 7873–7890 (2021). DOI 10.1007/s12652-020-02514-w

9. Han, S.D., Yu, J.: Effective heuristics for multi-robot path planning in warehouse environments. In: 2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS), pp. 10–12 (2019). DOI 10.1109/MRS.2019.8901065
10. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics **4**, 100–107 (1968). DOI 10.1109/TSSC.1968.300136
11. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. MIT Press, Cambridge, MA, USA (1992)
12. Huang, D., Jiang, H., Yu, Z., Kang, C., Hu, C.: Leader-following cluster consensus in multi-agent systems with intermittence. International Journal of Control, Automation and Systems **16**, 437–451 (2018). DOI 10.1007/s12555-017-0345-2
13. Huang, X., Cao, Q., Zhu, X.: Mixed path planning for multi-robots in structured hospital environment. The Journal of Engineering **2019**(14), 512–516 (2019). DOI 10.1049/joe.2018.9409
14. Jain, H., Deb, K.: An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. IEEE Transactions on Evolutionary Computation **18**, 602–622 (2014). DOI 10.1109/TEVC.2013.2281534
15. Kiadi, M., García, E., Villar, J.R., Tan, Q.: A*-based co-evolutionary approach for multi-robot path planning with collision avoidance. Cybernetics and Systems pp. 1–16 (2022). DOI 10.1080/01969722.2022.2030009
16. Koenig, S., Likhachev, M.: Fast replanning for navigation in unknown terrain. IEEE Transactions on Robotics **21**, 354–363 (2005). DOI 10.1109/TRO.2004.838026
17. Kumar, N.V., Kumar, C.S.: Development of collision free path planning algorithm for warehouse mobile robot. Procedia Computer Science **133**, 456–463 (2018). DOI 10.1016/j.procs.2018.07.056
18. Ortiz, E.G., Andres, B., Fraile, F., Poler, R., Ángel Ortiz Bas: Fleet management system for mobile robots in healthcare environments. Journal of Industrial Engineering and Management **14**(1), 55–71 (2021). DOI 10.3926/jiem.3284
19. Sharma, K., Doriya, R.: Coordination of multi-robot path planning for warehouse application using smart approach for identifying destinations. Intelligent Service Robotics **14**, 313–325 (2021). DOI 10.1007/s11370-021-00363-w
20. Silver, D.: Cooperative pathfinding. In: Proceedings of the First AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE'05), p. 117–122 (2005). DOI 10.1609/aiide.v1i1.18726
21. Solak, S., Yakut, Ö., Dogru Bolat, E.: Design and implementation of web-based virtual mobile robot laboratory for engineering education. Symmetry **12** (2020). DOI 10.3390/sym12060906
22. Stump, E., Michael, N.: Multi-robot persistent surveillance planning as a vehicle routing problem. In: 2011 IEEE International Conference on Automation Science and Engineering, pp. 569–575 (2011). DOI 10.1109/CASE.2011.6042503
23. Tan, Q., Denojean-Mairet, M., Wang, H., Zhang, X., Pivot, F.C., Treu, R.: Toward a telepresence robot empowered smart lab. Smart Learning Environments **6**, 5 (2019). DOI 10.1186/s40561-019-0084-3
24. Zheng, Y., Luo, Q., Wang, H., Wang, C., Chen, X.: Path planning of mobile robot based on adaptive ant colony algorithm. Journal of Intelligent & Fuzzy Systems **39**, 5329–5338 (2020). DOI 10.3233/JIFS-189018