UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Visión Electrónica

https://doi.org/10.14483/issn.2248-4728

VISIÓN ELECTRÓNICA

A RESEARCH VISION

Path planning using metaheuristics

Planificación de trayectorias usando metaheurísticas

Felipe Trujillo-Romero [1]

INFORMACIÓN DEL ARTÍCULO

Historia del artículo:
Enviado: 25/11/2021
Recibido: 21/01/2022
Aceptado: 30/03/2022

Keywords:
Ant Colony Optimization
Artificial Bee Colony
Mobile Robot
Robotic simulation

ABSTRACT

In this work, a comparison between two metaheuristic methods to solve the path planning problem is presented. These methods are 1) Artificial ant colony and 2) Artificial bee colony. The following metrics are used to evaluate these implementations: 1) Path length and 2) Execution time. The comparison was tested using ten maps obtained from the University of Prague Department of Intelligent Cybernetics and the Mobil Robotics Group. Several runs were carried out to find the best algorithm parameters and get the best algorithm for the route planning task. The best algorithm was the artificial bee colony. These evaluations were visualized using the VPython package; here, a differential mobile robot was simulated to follow the trajectory calculated by the best algorithm. This simulation made it possible to observe that the robot makes the correct trajectory from the starting point to the objective point in each evaluated map.

Palabras clave:
Colonia de abejas
Colonia de hormigas
Robot Móvil
Simulación de robots

RESUMEN

En este trabajo se presenta una comparación entre dos métodos metaheurísticos para resolver problemas de planificación de rutas. Estos métodos son: 1) Colonia de hormigas artificiales y 2) Colonia de abejas artificiales. Para evaluar estas implementaciones se utilizan las siguientes métricas: 1) Longitud de ruta y 2) Tiempo de ejecución. El comparativo se probó utilizando diez mapas obtenidos del Departamento de Cibernética Inteligente y Mobil Robotics Group de la Universidad de Praga. Se realizaron varias ejecuciones con el objetivo de encontrar los mejores parámetros de los algoritmos y obtener el mejor algoritmo para la tarea de planificación de ruta. El mejor algoritmo fue la colonia de abejas artificiales. Estas evaluaciones se visualizaron utilizando el paquete VPython; aquí se simuló un robot móvil diferencial para seguir la trayectoria calculada por el mejor algoritmo. A partir de esta simulación fue posible observar que el robot realiza la trayectoria correcta desde el punto de inicio hasta el punto objetivo en cada uno de los mapas evaluados.

1BSc. in Communications and Electronics Engineering, Universidad de Guanajuato, México. PhD. in Information systems, Instituto Politécnico de Toulouse, Francia. Current position: Research Professor at Universidad de Guanajuato, México. E-mail: fdj.trujillo@ugto.mx

Cite this article as: F. Trujillo-Romero, "Path planning using metaheuristics", *Visión Electrónica*, vol. 16, no. 1, pp. 29-40, 2022. https://doi.org/10.14483/22484728.18174

1. Introduction

Planning trajectories is a problem of interest in different research fields, such as robotics. So that a robot can move without problems in an environment about which it has specific information. This information can be in different ways. Such as knowing a priori the map or how many objects, shapes, and characteristics are on the scene. All this is necessary for the planning system to find a collision-free path and ensure the shortest route. For this, the system must know the map, the objects present, the robot's size, and the starting and ending points of the route.

This problem has been approached with various approaches, from classical algorithms such as A star (A*) [1] to evolutionary algorithms such as genetic algorithms [2] or ant colony optimization [3]. In the following paragraphs, we will comment on some of the work done using so-called classical algorithms to discuss later the work that uses metaheuristic algorithms for trajectory planning in mobile robots.

It starts by mentioning the work of Guruji et al. [4], who improved the execution time of the A* algorithm for the planning of mobile robots. In [5], Almanza Ojeda et al. implemented a system for a mobile robot to avoid obstacles through probabilistic models in embedded hardware. For their part, Goez et al. [6] used a microcontroller to implement route planning optimization using the particle swarm optimization (PSO) algorithm. Another work that uses PSO is Cuchango [7], who used the approach of potential fields and active Brownian particles to calculate trajectories in mobile robots. Li et at. in [8] improved the PSO algorithm also oriented towards trajectory planning.

Marquez Sanchez et al. [9], perform the generation of trajectories for mobile robots by using Bézier polynomials. Diaz-Arango et al. [10] use a Spherical Algorithm for trajectory planning in land mobile robots. In [11], Forero-García et al. implemented the intelligent control of a mobile robot differential for assistance applications in a house. Another application of trajectory planning but a variant of the differential evolution algorithm is found in [12]. As the last work of this block, we will comment on the one developed by Campos-Archila, Pinzón-Saavedra, and Robayo-Betancourt, who was using fuzzy logic carried out the trajectory control of an aerial robot [13].

On the side of trajectory planning works using metaheuristic algorithms, we will mention Canca et al. [14]. They solved the problem of the design and planning of Seville's rapid rail transport network by developing a metaheuristic algorithm called the Adaptive Neighborhood Search algorithm. A fascinating application is developed by Kergosein et al. [15]; this uses a metaheuristic algorithm to solve the problem of routing connected vehicles in a hospital complex. In [16], Ferreira et al. compared several evolutionary algorithms to solve the problem of routing mobile robots. Liu and Kozana in [17] implemented a hybrid metaheuristic algorithm to plan trajectories of a mobile robot to transport parts on a production line.

On the other hand, Guzman and Peña [18] were different bio-inspired algorithms for planning trajectories, but manipulative robots. Genetic algorithms have also been implemented as a strategy for planning trajectories. Examples of this are the works presented in [19-20]. In [21], Wu, Du, and Zhang performed trajectory planning of a mobile robot using wavefront's generalized algorithm. Finally, in [22], we can find a comparative study of metaheuristic algorithms applied to the planning of trajectories in mobile robots. These works are just a sample of metaheuristics to solve the problem of trajectory planning in mobile robots. Now move on to discuss results where both ant colony (ACO) optimization algorithms [3] and artificial bee colony (ABC) algorithms [23] are used.

We start the analysis of developments that use ACOs like the one made by Rashid et al. [24], who directly used it for trajectory planning. While in [25], Liu et al. implemented an improvement of ACO to plan trajectories in mobile robots. Another modification of the ant colony algorithm can be reviewed at [26]. Yong et al. implemented a cooperative algorithm for trajectory planning for a set of robots using ACO [27]. Optimizing the route planning of a school bus using a distributed ACO is presented in [28]. This brief review of trajectory planning using ACO will mention two-hybrid implementations. In [29], Gigras et al. performed an ACO and PSO hybrid algorithm. While in [30], Tao et al. merged ACO with fuzzy logic in both cases with acceptable results.

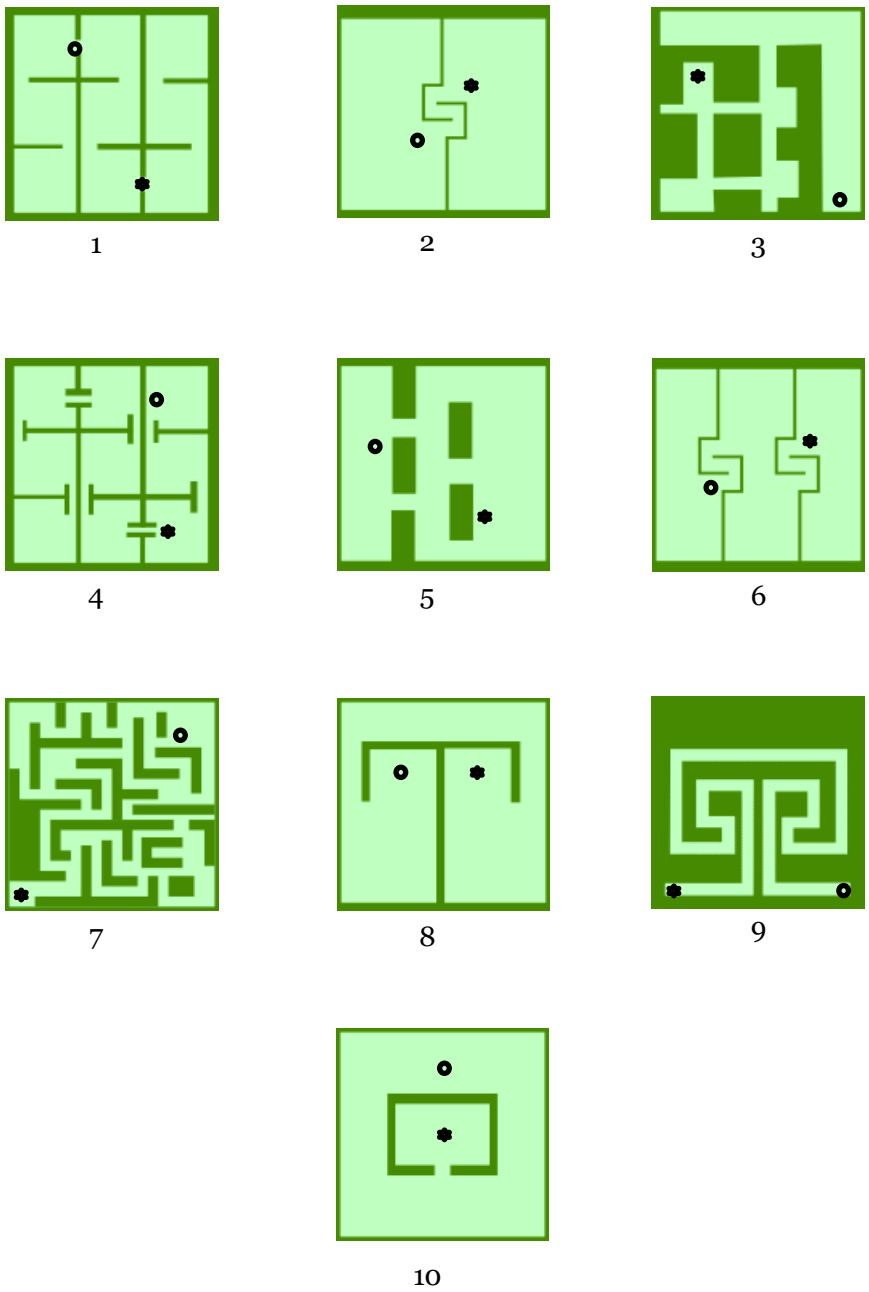
Now comment on a couple of papers that use the bee colony algorithm in trajectory planning. In this section, we start commenting on the work of Contreras-Cruz et al., who in [31] carried out the implementation of the ABC for the planning of trajectories also using evolutionary programming to soften the route obtained by ABC In [32], the best pathway for each of the robots in the scenario is tried, calculated from the robots present. While in [33], it uses an adaptive ABC approach to

estimate the trajectories of robots by evaluating their implementation in three different scenarios. Liang et al. [34] use the ABC to calculate a collision-free path between two points efficiently. Finally, Chen et al. use a hybrid neuro-diffuse approach with ABC to control a mobile robot [35].

2. System Elements

This section presents the different elements used to simulate the generation of the trajectories using the metaheuristic algorithms of the ant colony and bee colony. These elements are the maps, the robot, and the simulation environment.

Figure 1. Maps used to validate the planning algorithms [36]



2.1. Map representation

Ten maps were used to carry out the experiments presented in this paper. Figure 1 shows the maps used. These maps represent the full maps proposed by the Department of Cybernetic Intelligence and Mobile Robotics group of the University of Prague [36].

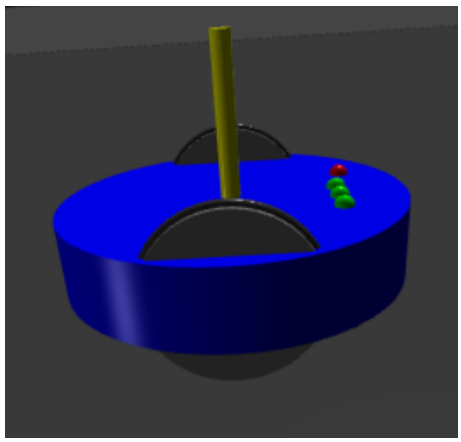
The maps used were drawn in the environment with a 5 pixels/cm ratio. The maps represent different geometries, allowing us to validate the metaheuristic algorithms in different scenarios.

Figure 1 shows the starting and target points on each map. The starting point is marked by a circumference (°), while the endpoint is represented by an asterisk (*). The trajectory calculation will be carried out using the implemented metaheuristic algorithms among these points.

2.2. Robot employee

The mobile robot used to develop this work was a robot with a differential configuration, shown in Figure 2. The diameter of the robot is a dimension of 10 cm. This robot is made using the Vturtle module [37] developed in Python using VPython [38].

Figure 2. Virtual representation of the robot [37]



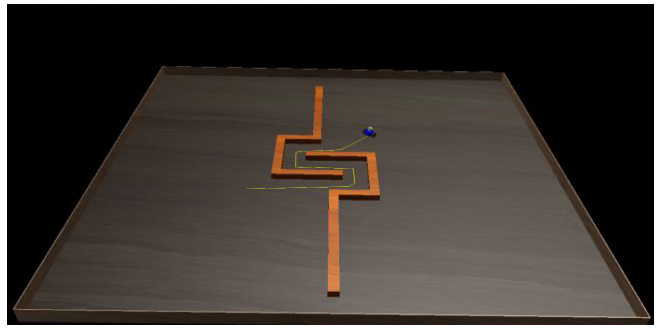
As can be seen in Figure 2, the robot has two main wheels and a freewheel. In addition, it has a set of sensors that help not collide with the obstacles present in the environment. However, given the characteristics of this work on this occasion, the sensors will not be used. Since the robot only executes the trajectory that metaheuristic algorithms have previously calculated.

2.3. Simulation environment

It was necessary to use a simulator to have a visualization that allowed us to appreciate the robot's behavior when tracking the trajectory obtained.

For that reason, this work uses the environment developed in [37]. This environment was created in Python and observed the robot's movement on the desired trajectory, Figure 3. This environment offers us several things: a three-dimensional environment, a differential robot, placing obstacles, the use of a pencil to trace the movement of the robot in the scenario.

Figure 3. Simulation environment



Source: own.

Adding different obstacles in the regions of our interest allowed us to make the maps used in a three-dimensional environment. In the case of the robot, it has several sensors that were not used for this work, but that can be very useful if it wants the robot to be reactive. The significant point is that the robot can load a tool, which is a pencil that allows observing the trajectory made by the robot in the environment.

3. Metaheuristic algorithms

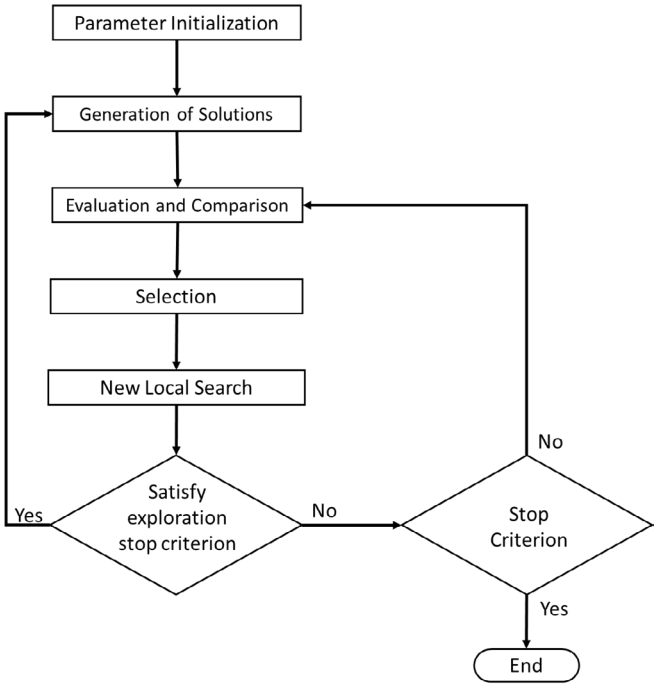
The following paragraphs will review in a general way the metaheuristic algorithms implemented, as well as the description of the parameters used for the planning of trajectories in both ABC and ACO.

3.1. Artificial Bee Colony (ABC)

In order to solve the trajectory planning problem in a given environment, it was proposed that the bees initialize in the same node. This node was called the origin. The possible paths that would connect it to

the target node would be created from this source. Therefore, the trajectory's length would improve with the algorithm's iterations, obtaining minor and less distance between the nodes. This procedure allowed eliminating the most extensive circuits to be replaced by the shortest distance. This process of eliminating the most expensive solutions, such as the longer trajectories, is known as elitism.

Figure 4. Flowchart for ABC



Source: own.

In this way, the robot's path from the initial point to the final point in the trajectory obtained is generated by the optimization obtained by the bee colony algorithm. Such optimization is performed according to the flowchart shown in Figure 4.

Following the flowchart in Figure 4, the scout bees are initialized and built different paths that connect the start node with the end node of the way. After the evaluation stage, the solutions obtained are compared, allowing us to know the best route generated. This is done by emitting to the surviving observers to select the best one by knowing the distance of the ways generated.

Subsequently, a local search is carried out, which allows improving the selected route until that moment. It is evaluated if the criterion of stop in the exploration carried out of the previously generated roads is met. The purpose of this is to stop exploring the environment and generate new paths in case of not complying with the condition. Otherwise, if the condition is satisfied, the unemployment condition of the algorithm is evaluated. Suppose this condition is not met, the algorithm iterates to continue evaluating and comparing new paths generated. In case the unemployment condition is met, the algorithm ends. This would indicate that the shortest route has been found between the start and endpoints of the desired trajectory.

Finally, from the algorithm, as mentioned earlier, we proceed to simulate the trajectory obtained in the environment made in Vpython with the differential robot in Figure 2. So, for each map shown in Figure 1 and considering the initial and objective nodes, the following steps are performed:

- Generate different trajectories without collision.
- Generate a random configuration on the map for each point configuration
- Find a collision-free path that connects the starting position to the target

Table 1 displays the values selected for the implementation of the algorithm.

Table 1. Parameters used for ABC.

Parameter	Value
Num. Cycles	N*50
Num. Bees	80, 100, 200
No. food sources	75

Source: own.

3.2. Ant Colony Optimization (ACO)

The Ant Colony optimization algorithm serves to find the shortest path in a graph. The main feature of this graph is that each node represents a place that the ant can visit, and each arc that joins these nodes has an associated cost that makes it more or less attractive.

However, a graph search does not solve trajectory planning problems since the work environment is a discrete scenario with free and forbidden regions. Being the free regions, the areas that the ants can visit and, for their part, the existing obstacles on the map will be prohibited areas for these.

Therefore, ants must find the shortest path from the nest to food. In other words, they must go from the starting point to the target point within the assigned map. In this case, there are no arches. In addition, the nodes are equivalent to spaces on the map. This allows the ant to move to any free position within the free zones.

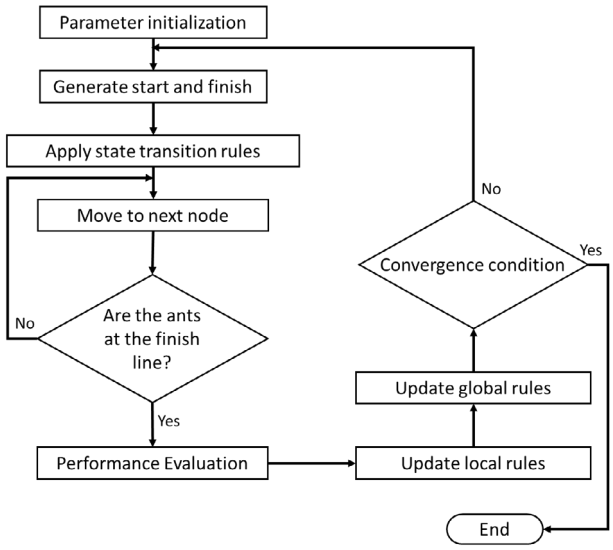
Another critical point is that the pheromone must be deposited in the areas visited by ants, thus generating a heuristic of movement preference that depends on the distance between the ant and the desired final configuration.

However, ants cannot move without having a possible route to follow. It is necessary to generate random nodes to construct a probabilistic map stored as a graph. The nodes are collision-free configurations, and the edges are feasible trajectories between these configurations.

The goal is to build a connected graph, and then the search phase responsible for finding a path through the graph that connects the initial and final configurations.

The results were obtained by planning the trajectories from an initial configuration from the algorithm shown in Figure 5.

Figure 5. Algorithm for planning trajectories using ACO



Source: own.

To determine the values of the parameters to be used in the experimentation stage, the values suggested in the literature for implementations of the ACO algorithm showed better performance was chosen. These parameters are the number of routes generated, the rate of evaporation of the pheromone, the level of the pheromone trail, and the number of ants.

The number of routes generated corresponds to the number of cycles performed by the algorithm multiplied by the number of ants in each cycle. That is the total of solutions that the algorithm calculates. We chose to use 50 cycles for each iteration of the algorithm performed.

In addition, it was determined to use three different numbers of ants to carry out the experiments carried out in this work. These were 80,100 and 200 ants.

The evaporation rate of the pheromone, represented by ρ, is the speed with which an ant fade leaves the trail. The value used was 0.1 since it showed a better result than with other values analyzed.

In the case of the value corresponding to the level of the pheromone trace, Dorigo [3] suggests using a minimal value. Therefore, it was decided to use the value of 0.0001 to implement this algorithm. This parameter is represented by t0. Table 2 is appreciated in a concentrated way, both the parameters used in the implementation of ACO and the corresponding values for each of them.

Table 2. Parameters used for ACO.

Parameter	Value
Num. Cycles	N*50
Num. ants	80, 100, 200
ρ	0.1
$\tau 0$	0.0001

Source: own.

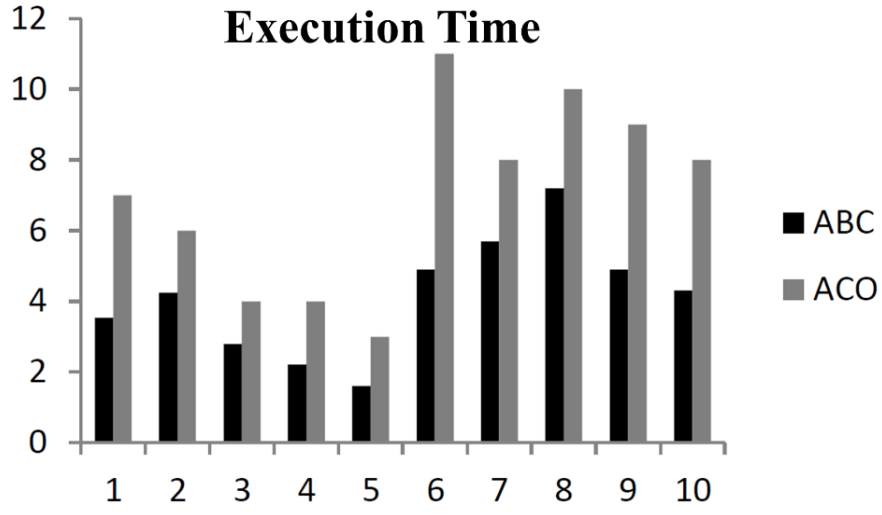
4. Results

Once the two algorithms were implemented, the relevant tests were carried out for each of them. For each map depicted in Figure 1, 50 executions were performed with each algorithm. The parameters considered to evaluate performance were runtime speed and path length. What was interesting was obtaining a fast implementation that obtained the minor travel. From these 50 executions were obtained the average value of execution time and path length performed by each of the algorithms in each map evaluated.

Figure 6 shows the execution time of the algorithms implemented for each of the ten maps. Also, in Figure 6, it is possible to see that the ACO algorithm spends a more significant amount of time finding the best route between the start and end points given for each map. It was the implementation of ABC the one that obtained better performance in time.

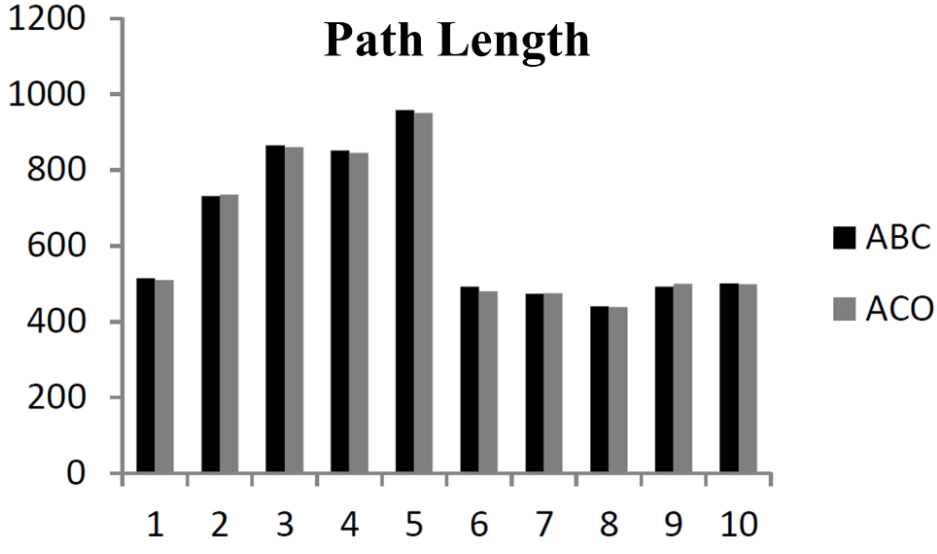
Figure 7, is shown the performance of ABC and ACO in the path length obtained for each map evaluated. In the same way, as for speed performance, 50 runs were used for each map shown in Figure 1. It is shown in Figure 7 the average values of these executions.

Figure 6. Runtime performance



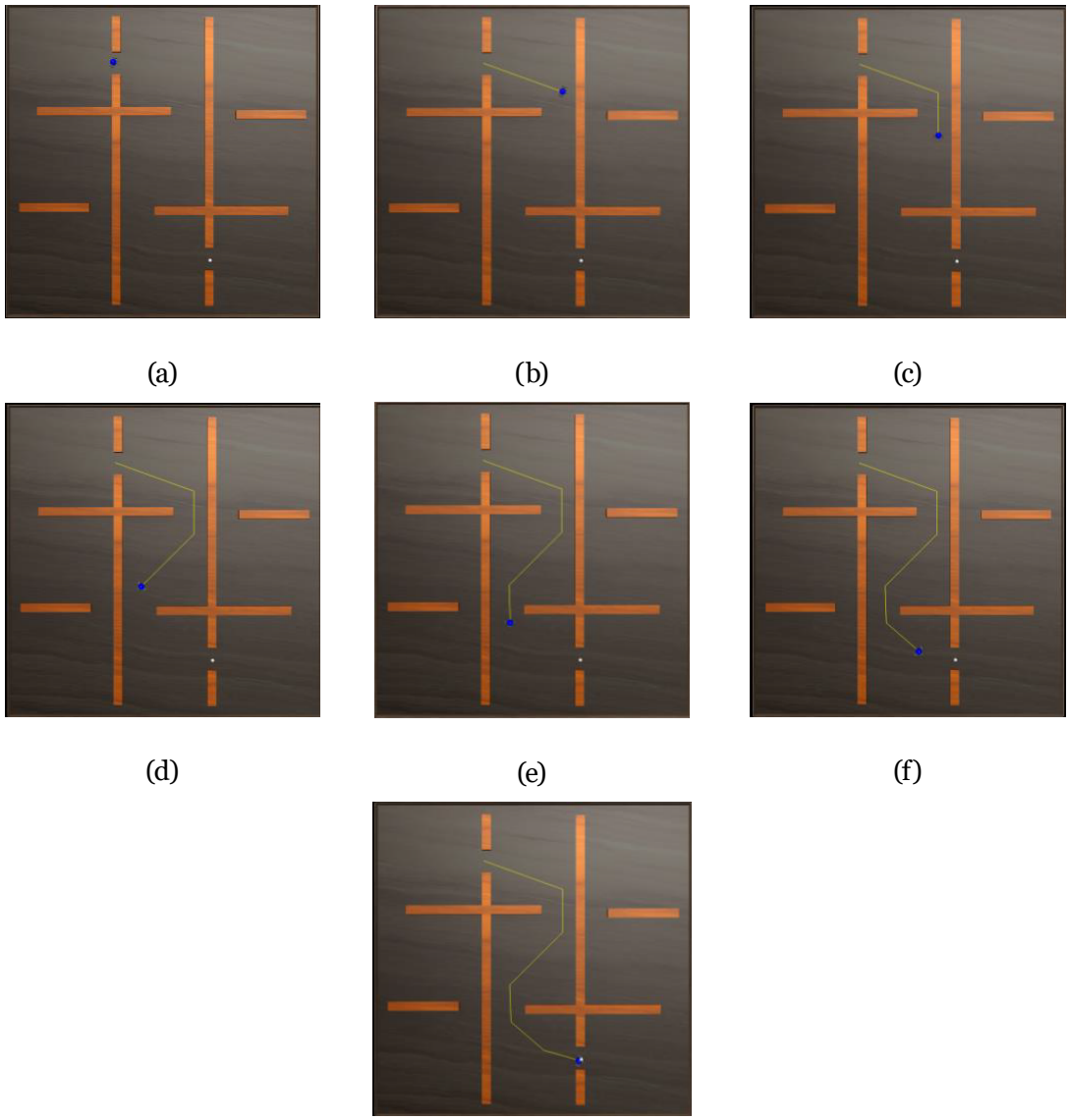
Source: own.

Figure 7. Performance of the path length obtained.



Source: own.

Figure 8. Route found by ABC for map number 1.



(a) (b) (c)

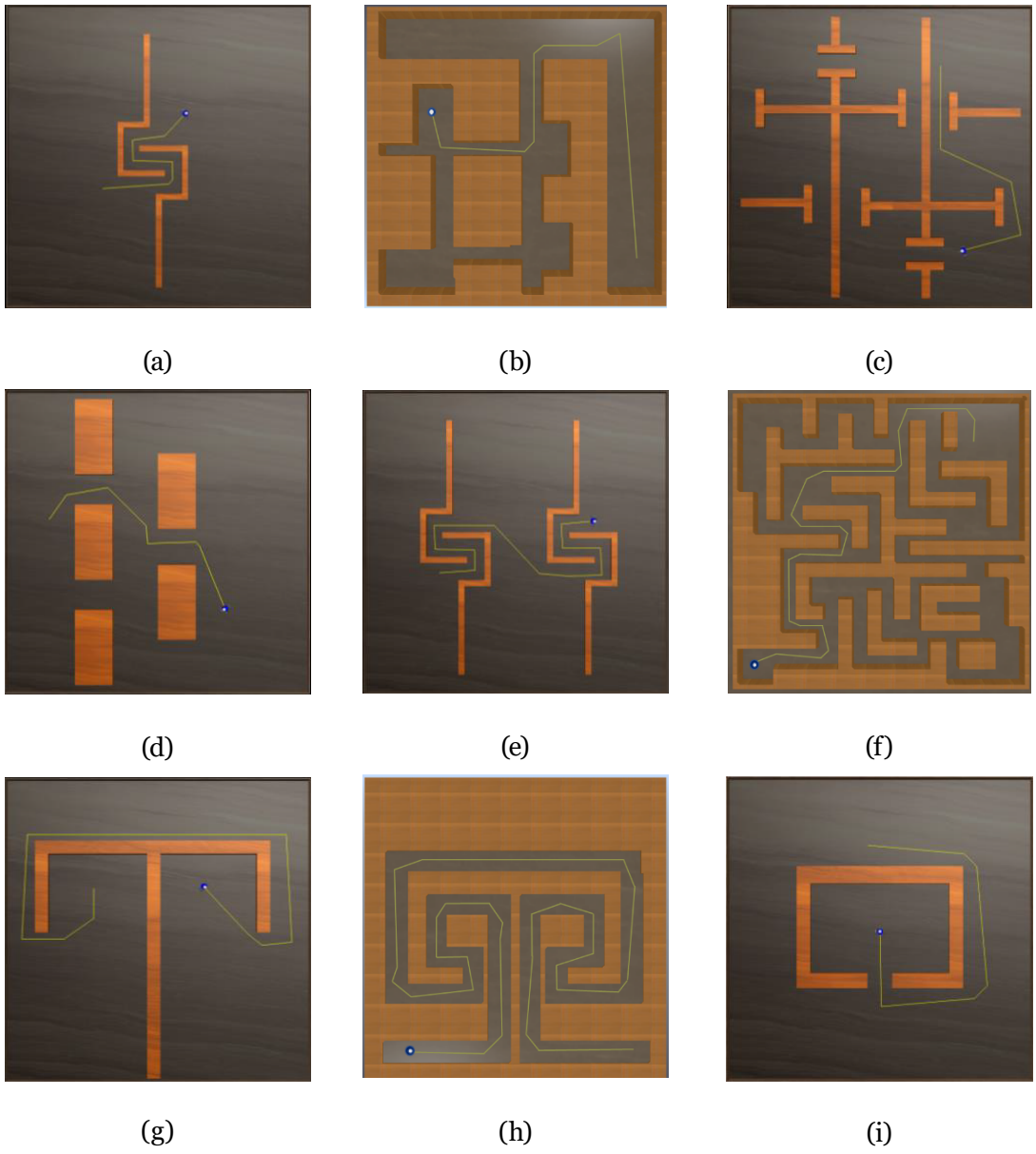
(d) (e) (f)

(g)

Source: own.

In Figure 8, we can see different captures of the trajectory that was obtained by the ABC algorithm for map number 1 shown in Figure 1. In the figure, we can see that the robot executes the planned trajectory without colliding. The stroke, in yellow color, allows us to realize how many nodes it visits between the initial and final points. In this case, there are five nodes considering the target node. Although it could have a trajectory with fewer nodes, it must be taken into account that when calculating the trajectory in each map, a margin is added to the obstacles due to the robot's size.

Figure 9. Trajectories found by ABC for other of the maps evaluated.



Source: own.

The trajectories obtained for maps 2 to 10 (see Figure 1) are shown in Figure 9. These trajectories were generated with the bee colony algorithm because it was better in the tests carried out, and what was sought was to have an efficient planner. We can highlight from the trajectories obtained that although parts on the maps could be assumed that the path found would be as direct as possible, this is not always the case. This is because the nodes initialized by the algorithm are random, allowing some edges in the path. The path found is the shortest in each of the maps.

5. Conclusions

The tests performed for the trajectory planning problem with each of the algorithms, both ACO and ABC applied to the evaluated maps, showed that the implementation of the ABC algorithm was the one that obtained a better performance both in speed and in the length of trajectory found.

Another relevant point observed while evaluating the implementations in scenarios used is the metaheuristic

algorithms. In this case, ACO and ABC are robust because a route between the starting and ending points for each map evaluated is always found as long as there was a route between those points.

Some directives to continue this work that has been considered are those listed below:

- Validate the ACO and ABC algorithms in a real robot. Mainly ABC that gave better results.
- Implement other metaheuristic algorithms for more exhaustive comparison
- Evaluate the performance of algorithms in real scenarios.
- Perform the smoothing of the obtained path

References

[1] P. E. Hart, N. J. Nilsson, B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100-107, 1968. https://doi.org/10.1109/TSSC.1968.300136

[2] J. H. Holland, "Genetic algorithms", Scientific American, vol. 267, no. 1, pp. 44-50, 1992. https://doi.org/10.1038/scientificamerican0792-66

[3] M. Dorigo, G. Di Caro. "Ant colony optimization: a new meta-heuristic", Congress on Evolutionary Computation, 1999. CEC 99. Proceedings of 1999, vol. 2, 1999.

[4] A. K. Guruji, H. Agarwal, D. K. Parsediya, "Time-efficient A* Algorithm for Robot Path Planning", *Procedia Technology*, vol. 23, pp. 144-149, 2016, https://doi.org/10.1016/j.protcy.2016.03.010

[5] D. L. Almanza Ojeda, Y. Gomar Vera, M. A. Ibarra Manzano, "Obstacle Detection and Avoidance by a Mobile Robot Using Probabilistic Models", *IEEE Latin America Transactions*, vol. 13, no. 1, pp. 69-75, 2015. https://doi.org/10.1109/TLA.2015.7040630

[6] G. D. Goez, R. A. Velasquez Velez, J. S. Botero Valencia, "UAV route planning optimization using PSO implemented on microcontrollers", *IEEE Latin America Transactions*, vol. 14, no. 4, pp. 1705-1710, 2016. https://doi.org/10.1109/TLA.2016.7483504

[7] H. E. Espitia Cuchango, J. Sofrony Esmeral, "Algoritmo para la planeación de trayectorias de robots móviles empleando enjambres de partículas brownianas", *Visión Electrónica*, vol. 5, no. 1, pp. 4–14, 2011.

[8] X. Li, D. Wu, J. He, M. Bashir, M. Liping, "An Improved Method of Particle Swarm Optimization for Path Planning of Mobile Robot", *Journal of Control Science and Engineering*, 2020. https://doi.org/10.1155/2020/3857894

[9] C. Marquez Sanchez et al., "Trajectory Generation for Wheeled Mobile Robots Via Bézier Polynomials", in *IEEE Latin America Transactions*, vol. 14, no. 11, pp. 4482-4490, 2016. https://doi.org/10.1109/TLA.2016.7795818

[10] G. Diaz-Arango, H. Vázquez-Leal, L. Hernandez-Martinez, M. T. S. Pascual and M. Sandoval-Hernandez, "Homotopy Path Planning for Terrestrial Robots Using Spherical Algorithm", in *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 567-585, 2018. https://doi.org/10.1109/TASE.2016.2638208

[11] S. G. Moctezuma Gutiérrez, A. Cruz Pazarán, R. Galicia Mejía, L. N. Oliva Moreno, "Desarrollo de plataforma para implementación de robots colaborativos", *Vis. Electron.*, vol. 12, no. 1, pp. 22–31, 2018. https://doi.org/10.14483/22484728.13308

[12] A. Rodríguez-Molina, J. Solís-Romero, M. G. Villarreal-Cervantes, O. Serrano-Pérez, and G. Flores-Caballero, "Path-Planning for Mobile Robots Using a Novel Variable-Length Differential Evolution Variant", *Mathematics*, vol. 9, no. 4, p. 357, 2021 https://doi.org/10.3390/math9040357

[13] F. Campos Archila, V. Pinzón Saavedra, F. Robayo Betancourt, "Fuzzy control of quadrotor Ar. Drone 2.0 in a controlled environment", *Vis Electron.*, vol. 13, no. 1, pp. 39–49, feb. 2019. https://doi.org/10.14483/22484728.14406

[14] D. Canca, A. De-Los-Santos, G. Laporte, J. A. Mesa, "An adaptive neighborhood search metaheuristic for the integrated railway rapid transit network design and line planning problem", *Elsevier Computers & Operations Research*, vol. 78, pp. 1–14, 2017. https://doi.org/10.1016/j.cor.2016.08.008

[15] Y. Kergosein, Ch. Lenté, J. Billaut, S. Perrin, "Metaheuristic algorithms for solving two interconnected vehicle routing problems in a hospital complex", *Elsevier Computers & Operations Research,* vol. 40, pp. 2508–2518, 2013. https://doi.org/10.1016/j.cor.2013.01.009

[16] J. C. Ferreira, M. T. Arns Steiner, and M. Siqueira Guersola, "A Vehicle Routing Problem Solved Through Some Metaheuristics Procedures: A Case Study", in *IEEE Latin America Transactions*, vol. 15, no. 5, pp. 943-949, 2017. https://doi.org/10.1109/TLA.2017.7910210

[17] S. Q. Liu, E. Kozana, "A hybrid metaheuristic algorithm to optimize a real-world robotic cell", *Elsevier Computers & Operations Research*, 2016. https://doi.org/10.1016/j.cor.2016.09.011

[18] R. M. Molano Pulido, F. Parca Acevedo, F. M. Cabrera, H. Ñungo Londoño, "Prototipo control de vehículo robot por señales EMG", *Vis. Electron.*, vol. 15, no. 2, 2021.

[19] K. Hao, J. Zhao, K. Yu, C. Li, C. Wang, "Path Planning of Mobile Robots Based on a Multi-Population Migration Genetic Algorithm", *Sensors*, vol. 20, no. 20, p. 5873, 2020. https://doi.org/10.3390/s20205873

[20] R. K. Panda, B. B. Choudhury, "An Effective Path Planning of Mobile Robot Using Genetic Algorithm", 2015 IEEE International Conference on Computational Intelligence & Communication Technology, pp. 287-291, 2015. https://doi.org/10.1109/CICT.2015.145

[21] S. Wu, Y. Du, Y. Zhang, "Mobile Robot Path Planning Based on a Generalized Wavefront Algorithm", *Mathematical Problems in Engineering*, vol. 2020, 2020. https://doi.org/10.1155/2020/6798798

[22] S. K. Pattnaik, D. Mishra, S. Panda, "A comparative study of meta-heuristics for local path planning of a mobile robot", Engineering Optimization, 2021. https://doi.org/10.1080/0305215X.2020.1858074

[23] D. Karaboga, "An idea based on honey bee swarm for numerical optimization", Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, vol. 200, 2005.

[24] R. Razif, N. Perumal, I. Elamvazuthi, M. Kamal Tageldeen, M. Ahamed Khan, S. Parasuraman, "Mobile robot path planning using Ant Colony Optimization", in Robotics and Manufacturing Automation (ROMA), 2016 2nd IEEE International Symposium on, pp. 1-6, 2016. https://doi.org/10.1109/ROMA.2016.7847836

[25] L. Jianhua, J. Yang, H. Liu, X. Tian, M. Gao. "An improved ant colony algorithm for robot path planning", Soft Computing, vol. 21, no. 19, pp. 5829-5839, 2017. https://doi.org/10.1007/s00500-016-2161-7

[26] Y. Zhongrui, Y. Houyu, H. Miaohua, "Improved Ant Colony Optimization Algorithm for Intelligent Vehicle Path Planning", 2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), Wuhan, pp. 1-4, 2017. https://doi.org/10.1109/ICIICII.2017.55

[27] L. Yong, L. Yu, G. Yipei, C. Kejie, "Cooperative path planning of robot swarm based on ACO", 2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, pp. 1428-1432, 2017. https://doi.org/10.1109/ITNEC.2017.8285033

[28] E. Mouhcine, M. Khalifa, Y. Mohamed, "Route optimization for school bus scheduling problem based on a distributed ant colony system algorithm", 2017 Intelligent Systems and Computer Vision (ISCV), Fez, pp. 1-8, 2017.

[29] Y. Gigras, K. Choudhary, K. Gupta, Vandana, "A hybrid ACO-PSO technique for path planning", 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, pp. 1616-1621, 2015.

[30] Y. Tao, H. Gao, F. Ren, C. Chen, T. Wang, H. Xiong, S. Jiang, "A Mobile Service Robot Global Path Planning Method Based on Ant Colony Optimization and Fuzzy Control", Applied Sciences, vol. 11, no. 8, p. 3605, 2021. https://doi.org/10.3390/app11083605

[31] M. Contreras-Cruz, V. Ayala-Ramirez, H. Hernandez-Belmonte. "Mobile robot path planning using artificial bee colony and evolutionary programming", Applied Soft Computing, vol. 30, pp. 319-328, 2015. https://doi.org/10.1016/j.asoc.2015.01.067

[32] P. Bhattacharjee, P. Rakshit, I. Goswami, A. Konar, A. Nagar, "Multi-robot path-planning using artificial bee colony optimization algorithm", in Nature and Biologically Inspired Computing (NaBIC), pp. 219-224, 2011. https://doi.org/10.1109/NaBIC.2011.6089601

[33] A. Nizar Hadi, J. Ahmed Abdulsaheb. "An Adaptive Multi-Objective Artificial Bee Colony Algorithm for Multi-Robot Path Planning", Association of Arab Universities Journal of Engineering Sciences, vol. 24, no. 3, pp. 168-189, 2017.

[34] L. Jun-Hao, C. Hung Lee. "Efficient collision-free path planning of multiple mobile robots system using efficient artificial bee colony algorithm", Advances in Engineering Software, vol. 79, pp. 47-56, 2015. https://doi.org/10.1016/j.advengsoft.2014.09.006

[35] C. H. Chen, S. Y. Jeng, C. J. Lin, "Using an Adaptive Fuzzy Neural Network Based on a Multi-Strategy-Based Artificial Bee Colony for Mobile Robot Control", Mathematics, vol. 8, no. 8, p. 1223, 2020. https://doi.org/10.3390/math8081223

[36] V. Vanásek, "Intelligent and Mobile Robotics Group", 2009. [online]. Available: http://imr.felk.cvut.cz/planning/maps.xml

[37] Possiblywrong, "Web simulador turtle", https://possiblywrong.wordpress.com/2010/12/04/robot-simulator-and-turtle-graphics/

[38] Python, "Web Visual Python", http://vpython.org/