Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 51-13 (2018) 650–655

MATERIAL DISTRIBUTION WITH MOBILE ROBOTS IN AN INDUSTRIAL ENVIRONMENT: System design and simulation.

Gabriel Grijalva, Danilo Chávez, Oscar Camacho

Departamento de Automatización y Control Industrial
Escuela Politécnica Nacional, Quito, Ecuador
{gabriel.grijalva, danilo.chavez, oscar.camacho} @epn.edu.ec

Abstract: This paper presents an approach to find the fastest route for autonomous robots in an industrial environment. The system predicts the fastest route by comparing the theoretical performance of the shortest and fastest routes. The system has been tested with the layout of the MAN's busses assembly line in Turkey, and the simulation has been made using Matlab. Simulations are used to prove the proposal functioning.

Keywords: mobile robots; path planning; intralogistics; Matlab simulation.

1. INTRODUCTION

The automated warehouse is one of automation's current objectives. In this regard, there has been plenty of research, and the search for an optimal route is one of the most studied topics (Sai-nan, 2008).

Multiple robots are able to deal with complex distribution problems with tight time restrictions. However, while working with multiple robots it is necessary to use a central coordination system. Mathew's multi-robot delivery with multiple warehouse (Mathew, Smith, & Waslander, 2015) is an example of the benefits of using one of this systems.

Path planning is a wide topic, which according to Raja (Raja & Pugazhenthi, 2012) could be divided between "on-line" and "off-line" algorithms. Off-line algorithms consider the whole area's distribution to build the routes that the robots will be able to use. For example C-space, visibility graph and cell decomposition are some of the most popular approaches (Lozano-Pérez & Wesley, 1979).

Path planning algorithms obtain optimal routes according to a certain criteria, normally the shortest routes. Dijkstra's algorithm and its improvements are widely used (Kang, Lee, & Kim, 2008), but other faster algorithms like the A-Star and some of its improvements are currently the most common path planning technique (Wang, Wang, & Qin Jian, 2015; Jia, Ren, & Chen, 2017). Meanwhile, advanced optimization techniques like the Genetic Algorithm are more adequate for a multi-objective problem (Han, Wang, Liu, & Zhao, 2017).

This paper presents an approach to path-planning and assignment for autonomous robots in an industrial environment. The system is able to build all the available routes, local evasion of static obstacles, collision avoidance between robots, determine the fastest available route, determine when to start a new delivery and coordinate all the robots.

This work is organized as follows: section 2 shows the problem description, in section 3 the methodology is described, section 4 contains the simulation results and finally conclusions are offered.

2. PROBLEM DESCRIPTION

The assembly of a car, bus or truck requires a process with a series of activities, an example of departments and activities is presented in Table 1. The sub-assembly process allows to combine the smallest material to form bigger parts that will be added to the vehicles' chassis. This materials, stored at the warehouse, must be transported to the working stations as efficient as possible.

Material distribution is normally in charge of human operators. However, since it is a highly repetitive activity the operator gets easily distracted and bored, decreasing performance. Meanwhile, material availability is critical to maintain the manufacturing process continuity, so it is imperative to optimize this activity.

Robots coordination is important to obtain the best possible results. There are several ways to approach the robotic coordination, for example the robotic formations with different objectives according to their distribution (Balch & Arkin, 1998), or a coordinated system based on interaction rules (Wang, 1995; Švestka & Overmars, 1998). In order to achieve an organized deployment and since the activities of each robot is independent from the rest of the group, a central coordination system is the selected option for this problem.

3. METODOLOGY

The main objective of this project is to determine if the proposed system to select a route could be useful. For this reason, the other elements will be designed as simple as possible just to allow the verification of the route selection system.

The first part of the system is the offline path generator. During the first startup of the program, it will generate a database with every possible path. After that the route selection system will start working. And the robots will follow their designed paths with a slight modification of the path following control proposed by Soetanto et al. (Soetanto, Lapierre, & Pascoal, 2003).

3.1. Sub-assembly layout

10.1016/j.ifacol.2018.07.354

Table 1: DEPARTMENTS AND ACTIVITIES AT MAN S.A.

Department	Type of work
Chassis Assembly	Chassis welding
Paint shop	Anti-corrosive chemical application and painting
Pre-assembly	Infrastructural systems assembly
Assembly	Accessories assembly
Final product	Paint polish and final revisions

Based on Artun Torenli's (Törenlí, 2009) similar chart and information

3.2. Layout cells and Flow points

Building the layout is the first step. The layout is based on Fig. 1 and the sub-assembly stations distribution to optimize the production process proposed by Artun Törenlí (Törenlí, 2009), shown in Fig. 2. As a result the sub-assembly layout and station's location from Fig. 3

Each layout cell (black squares from Fig. 4) has been tagged depending on their function in the sub-assembly process. The layout cells that do not take part in the sub-assembly are identified with the code PA#. While the layout cells that has a sub-assembly station are named by the code WA#. The final names for each layout cell are shown in Fig. 4.

The "flow points" represent an area located next to a layout cell where the robots can circulate. There are four flow points for each layout cell, but there could only be one flow point between two layout cells. As a result, most of them will be addressed by two different names. An example of both names for each flow point around the layout station WA1 is shown in Fig. 5.

3.3. Path fragments

Path fragments are small routes that connects two different flow points, they are the building blocks of every route used by the system. The "path fragments" are automatically generated using the layout static obstacles and the flow points coordinates. Between two adjacent flow points, two path fragments are build, this is shown in Fig. 6 and Fig. 9.

Each North/South (NS) flow point is connected to two NS flow points and four West/East (WE) flow points, and each WE flow point is connected to two WE flow points and four NS flow points.

Depending on the type of flow point that is being connected, the path fragments could be built in two different ways. Between two flow points of the same type (NS with NS or WE with WE) a "line" connection is created, and if its two flow points of different types (NS with WE and vice versa) a "quarter-circle" connection is built instead. This type of connections are very simple and smooth.

3.4. "Line" path fragments

The "line" path fragments are used to connect two flow points that maintain the same direction. "Line" path fragments are not straight connections between the flow points coordinates. They start and end as straight lines (parallel to the closest cell's border), but they are connected by two circumference fragments that provide a smooth transition, depending on the amount of space available.

A "line" connection starts with the construction of the auxiliary path fragment, like the one in Fig. 7. Each auxiliary path fragment is divided in three parts: two straight segments (segments 1 and 2) and one curved connector (curve 3).

The straight segments maintain the original direction of the initial and final flow points. To build the curved connector it is necessary to analyze Fig. 8, where:

- The blue arrows represent the direction and position of the last fragments of both straight segments previously constructed.

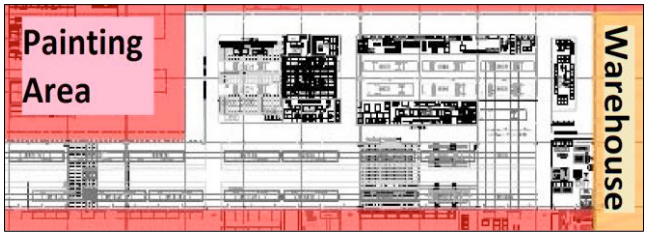

Fig. 1 Sub-assembly layout.

Fig. 2 Sub-assembly stations' recommended distribution.

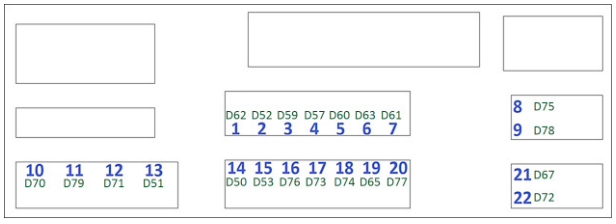

Fig. 3 Simulation's sub-assembly layout and stations' locations.

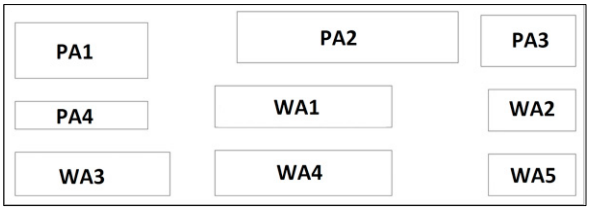

Fig. 4 Layout areas' names and locations.

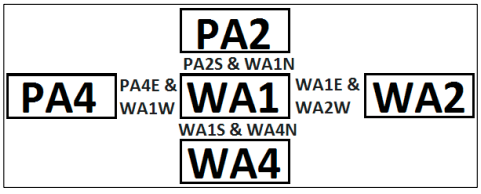

Fig. 5 Flow points' tagging example.

- "B" is the end of the straight segment that is going to the left, "E" is the end of the straight segment that is going to the right, and "D" is the center between "B" and "E".
- "A" is the location of the center of the circle that would connect "B" with "D".
- "F" is the center of the circle that would connect "E" with "D".
- "r" is the radius of both circles and also the distance of the repeated side from both isosceles triangles (ΔABD *and* ΔDEF).
- "$\angle s$" represents the circumference fragment that would be used to connect both straight segments ("B" and "E") with "D" and the unequal angle from the isosceles triangles.
- "$\angle q$" is the repeated angle of the isosceles triangles, "c" is the unequal side of the isosceles triangles (which is easily measurable).
- "∠diff" is the angle between the straight segment and the segment "$\overline{EB}$" (which is also easily measurable).

Applying the law of cosines to the isosceles triangles we get (1):

$$c^2 = r^2 + r^2 - 2 * r * r * \cos(\hat{s}) \qquad (1)$$

With the sum of internal angles to the isosceles triangles we get (2):

$$\hat{s} + 2 * \hat{q} = 180° \qquad (2)$$

Using complementary angles between "∠q" and "∠diff" in "B" we get (3):

$$\hat{q} = 90° - \widehat{diff} \qquad (3)$$

Using the first three equations we are able to simplify the value of "r" as function of "∠diff" and "$\overline{c}$" expressed in (4):

$$r = \frac{c}{\sqrt{2 * (1 - \cos(2 * \widehat{diff}))}} \qquad (4)$$

Since we know that the center should be located in a perpendicular line to each straight path, and that "r" is the distance "$\overline{AB}$" and "$\overline{EF}$", then we already know the location of the center of the circles and their radius, which allows the construction of the auxiliary path fragment for a "line" connection.

Once that the auxiliary fragment has been created (green line from Fig. 7) the main fragments are created by adding and subtracting a fixed distance (horizontally or vertically) to every coordinate in the auxiliary fragment, and adjusting the directions that each path fragment should have, obtaining the two path fragments like the ones in Fig. 9.

The "line" connections are proposed as the simplest way to connect two points with a smooth transition and maintain the original direction.

3.5. *"Quarter-circle" path fragments*

The "Quarter-circle" path fragments are used when the direction of the flow points that would be connected is different.

A "Quarter-circle" connection's auxiliary path fragment, as it is shown in Fig. 10, is formed by four parts. They are two straight segments until the border of the obstacles (segments 1 and 2 which will be identified as "obstacle segments"). Another straight segment to equalize the distance with the end of the other obstacle segment (segment 3, identified as "equalizing segment"). The last one (curve 4, identified as "quarter circle") is used to link the obstacle straight segment and the extension segment.

The building process is the same than the last one for the obstacle segments. The equalizing segment makes sure that the horizontal distance and the vertical distance between the end of one of the obstacle segments and the end of the equalizing segment is the same. Finally, the quarter circle is used to link

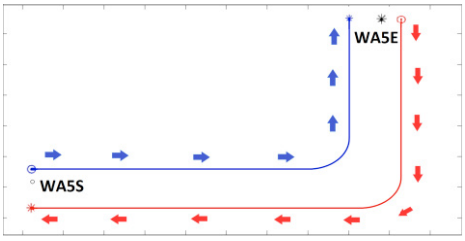

Fig. 6 "Quarter-circle" main path fragments between WA5E and WA5S.

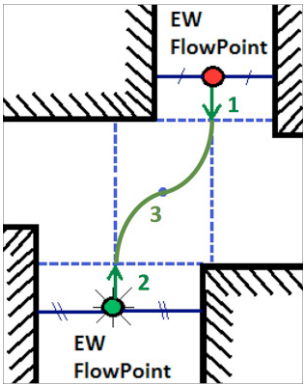

Fig. 7 Auxiliary fragment between two EW flow points to build the "Line" type of path fragments.

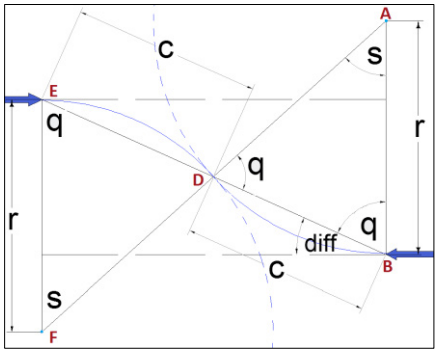

Fig. 8 Line connection auxiliary angles and distances.

the previous segments, and its radius is equal to the horizontal (or vertical) distance between them.

When the auxiliary path fragment is completed, the main path fragments are made in two phases. The first part of the process is to reposition the curved segment to the inner and the outer part of the curve, as shown in Fig. 11. To achieve it is required to notice that the auxiliary fragment's curved segment ($\widetilde{BD}$) is surrounded by a square (■$ABCD$). A square of the same dimensions (■$EFGH$) is moved along the radial diagonal ($\overline{EK}$) a distance previously calculated "d", obtaining the same vertical and horizontal distance (k). Then the curved segment of one of the main path fragments is constructed ($\widetilde{FH}$). The process is repeated for the square IJKL, but it is moved in the opposite direction of the diagonal radial diagonal ($\overline{EK}$).

The last part would be to extend the curved segment to reach the level of the flow point. The results are two path fragments with opposite directions like the ones in Fig. 6.

The quarter-circle is the simplest way to execute a smooth turn of ninety degrees. The system does not consider the robots-s turning radius, but it is possible to select a minimum radius for the circumference fragment.

3.6. Multipath

Once that every connection between all the available flow points has been made, the result is the web of path fragments that is displayed on Fig. 12.

The last part of the startup process is to create all the usable chains of path fragments ("multipath") that go from WA2E to a reachable flow point that follows certain conditions ("going" set of multipaths), and from every flow point back to WA5E ("returning" set of multipaths).

To create a set of "going" multipaths the next process is followed:

First a set of initial flow point and a direction is selected, for example [WA2E-south] or [WA2E-north]. Then for each flow point connected to the previous [flow point-given direction], two new multipaths are added to the set of multipaths. For example for [WA2E-north] the multipaths [WA2E-north, WA2N-west] and [WA2E-north, PA3E-north] will be generated two times each. Both equal multipaths are needed since one is considered a complete multipath while the other's possibility to continue has to be verified. A multipath is completed if it could not be connected to another flow point without repeating any path fragment that has already been used in that multipath, or if the number of connected flow points has exceeded a certain number (11 is used on the simulation). If a multipath is not completed the process of creating two new multipaths for each connected [flow point-direction] is repeated, and if it is completed it is deleted and the next incomplete multipath growing capability is verified.

Once all the possible multipaths have been created, a matrix will contain the corresponding initial [flow point - direction] and destination [flow point - direction] to find the required multipath whenever they are needed.

The process to create a "returning" set of multipaths is almost the same, the only difference is that when analyzing which [Flow point-direction] are linked to our current [Flow point-direction], the ones that would be selected are those that would reach the current [Flow point-direction] instead of the ones that would be able to go from our current [Flow point-direction]. For example staring with [WA5E-north] the resulting multipath will be [WA5E-north, WA5S-east] two times, since it is the only way to reach [WA5E-north].

3.7. Send a robot to a station or back to the warehouse

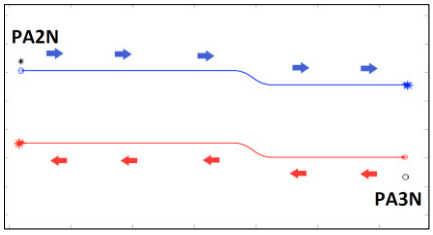

Fig. 9 "Line" main path fragments between PA2N and PA3N

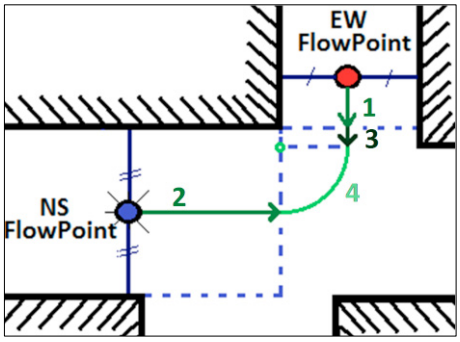

Fig. 10 Auxiliary fragment between a NS and an EW flow points to build the "Quarter-circle" type of path fragments

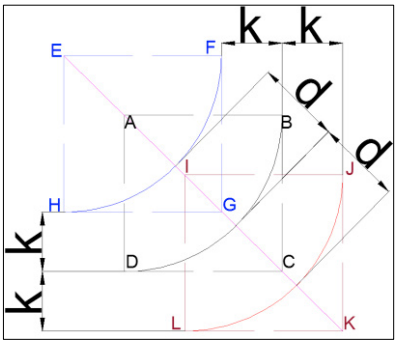

Fig. 11 Main path fragment's curved segment construction from auxiliary fragment's curved segment.

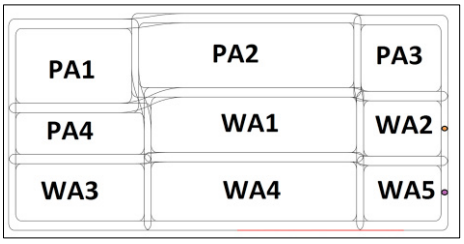

Fig. 12 All path fragments

Once the main program has started, the system is constantly analyzing if any station is in need of material delivery. This decision is based on the remaining time for a station to use the material that it has available, the amount of time that a robot would require to reach that station, a reserve of material that should always remain in the station as backup.

When the program decides to send a robot to a certain station, it follows the next steps:

- Search the associated [Flow point-direction] to that specific station (this should be configured while defining the stations). For example, stations 1 to 7 are linked to [WA1S- east].
- Then search in the sets of "going" multipaths all the multipaths with the established [Flow point- direction] as destination.
- Find the "cost" for each multipath as the sum of the amount of time it would require added to the total distance, both multiplied by an appropriate constant to make those values comparable, then sort the results.
- Select a fixed number of the best multipaths according to the cost-sorting (top three multipaths is used on the simulation).
- With the chosen multipaths a "predictive simulation" would find the fastest. During the predictive simulation, all the robots that has already been deployed continue their assigned routes, and a "simulated robot" is assigned to each of the selected cost efficient multipaths. The predictive simulation tries to foresee the performance of each multipath, allowing the system to find the fastest one.
- The "predictive simulation" is finished when one of the "simulated robots" arrives to its destination.

The route of the first simulated robot that arrived to the destination is fastest route. Therefore, it is assigned to a real robot so that it could start the material delivery.

When a robot reaches the destination flow point, it must be assigned a different route to return to the warehouse. The required process is the same, but using the current [Flow point-direction] as the target, and searching it in the "returning" sets of multipaths instead of the "going" ones.

4. SIMULATION RESULTS

In this section the results obtained will be analyzed. The proposed system has been tested with a discrete system using Matlab. The systems contains, among other data and functions, the next elements:

- The simulation area's layout and dimensions, the robot's dimensions, the robots' maximum values of: linear speed, angular speed, linear force and torque.
- A simple dynamic model of differential robots.
- The offline path creation system.
- Static obstacle avoidance with c-space border tracking.
- Collision avoidance between robots by speed reduction of one of the robots upon a possible crash detection.
- The required functions to calculate, store and display the currents status of the robots and the stations for every sampling time.
- The route selector based on predictive simulations.

This results will be supported with the graphical output of the system, in which the black bordered rectangles are considered obstacles. The small black figures with a different colored rectangle on top of each one are the robots. Each robot's assigned route is represented with a chain of small rectangles of the same color.

The assigned routes have a vertical and horizontal offset (depending on the robot's number), so that their representations do not overlap with the other routes in case that two or more robots are assigned with the same or a similar route. As a result some of the routes seem to be close or even inside of certain layout cells.

The "simulated robots" are considered as the same one, therefore all of them have the same color, in order to avoid interaction between them. Therefore, an arbitrary labelling for the simulated robots and some arrows that hint the trajectory of the multipaths had been added to make it easier to understand the graph.

Under normal conditions the system will assign the shortest route as the fastest route since the speed of the robots is practically constant. However, under certain specific scenarios the shortest route is not necessarily going to be selected. This kind of interactions will be harder to find with an algorithm in charge of obtaining the closest route.

The scenarios presented in Fig. 13 and Fig. 14 represent an instant during the "predictive simulation" in which the simulated robots (R, G and B) had been assigned different multipaths to reach a certain flow point.

In Fig. 13 the three best possible paths to reach [WA1S-east] are being compared. This situation is being developed without any other deployed robots and without any other obstacles besides the layout cells, therefore the robot R with the smallest route is more likely to arrive first.

But with more robots deployed, even without additional obstacles, it is possible that the "predictive simulation" would yield different results from the cost based route selection, because of the interactions between the "simulated robots" and the real robots. In Fig. 14 it is possible to see that the shortest route (the one being followed by robot R) is being delayed due to a collision avoidance between the "simulated robot" R and the first deployed robot (red). But since the other possible routes are larger than the first one the simulated robot R will arrive first.

Another possible scenario is with additional obstacles like in Fig. 15, where the small rectangle labeled as "Obs" is a static obstacle. In this case the shortest multipath assigned to R is blocked by the obstacle Obs, which results in G arriving faster than R, but the route assigned to R is still better than B's route.

5. CONCLUSION

Despite the amount of efficient algorithms formerly developed and tested, the methodologies to solve path-planning problems

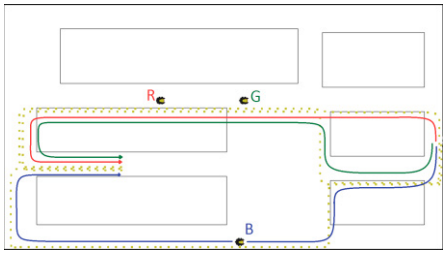

Fig. 13 Predictive simulation to reach [WA1S-east] while any other robots have been deployed

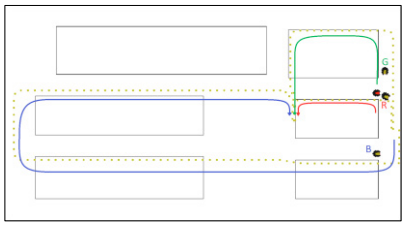

Fig. 14 Predictive simulation for robot #2 to reach [WA2W-south]

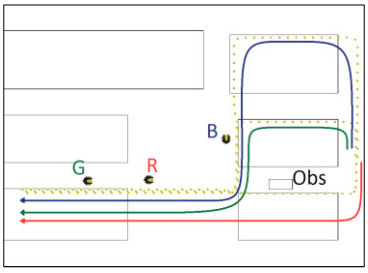

Fig. 15 Predictive simulation for the first robot to reach [WA4N-west] with an obstacle near WA5N

keep growing. This work provides a simple and effective approach with the possibility to keep growing and improving.

Traditional algorithms normally try to obtain the shortest route. Using a route selector with predictive simulation is able to obtain a fastest route under unexpected conditions.

In this paper the proposed system has been designed in such a way that it could be easily modified to fit any industrial environment, with any model of mobile robot. However, there are certain considerations that may need to be modified to be implemented on a specific industrial environment, such as the space between two adjacent layout cells and the dimensions of the robots.

From the results of the designed system we have been able to prove that a system conformed by a simple offline path-planning, an appropriate obstacles evasion and an accurate predictive simulation is an optimal way to distribute production material. The creation of all main routes beforehand reducing the processing load, which could in turn be used to run the predictive simulations, ensuring that the fastest possible route is assigned to the delivery robot.

The system was designed to find and assign the fastest route. However, it is only required to modify the termination condition of the predictive simulation and the analyzed variable(s) to change the objective of the route assigner.

REFERENCES

Balch , T., & Arkin, R. C. (1998). Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation, 14*(6), 926-939.

Han, Z., Wang, D., Liu, F., & Zhao, Z. (2017). Multi-AGV path planning iwth double-path constraints by using an improved genetic algorithm. *PLoS ONE, 12*(7). doi:https://doi.org/10.1371/journal.pone.0181747

Jia, F., Ren, C., & Chen, Y. (2017). A system control strategy of a conflict-free multi-AGV routing based on improved A∗ algorithm. *Mechatronics and Machine Vision in Practice (M2VIP)*.

Kang, H., Lee, B., & Kim, K. (2008). Path Planning ALgorithm Using the Particle Swarm Optimization and the Improved Dijktra Algorithm. *Computational Intelligence and Industrial Application*, 1002-1004.

Lozano-Pérez, T., & Wesley, M. A. (1979). An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM, 22*(10), 560-570.

Mathew, N., Smith, S. L., & Waslander, S. L. (2015). Optimal Path Planning in Cooperative Heterogeneous Multi-robot Delivery Systems. En H. L. Akin, N. M. Amato, & V. Isler, *Algorithmic Foundations of Robotics XI* (págs. 407-423). Switzerland: Springer International Publishing. Obtenido de https://doi.org/10.1007/978-3-319-16595-0_24

Raja, P., & Pugazhenthi, S. (2012). Optimal path planning of mobile robots: A review. *International Journal of Physical Sciences, 7*(9), 1314-1320. doi:10.5897/IJPS11.1745

Sai-nan, L. (2008). Optimization problem for AGV in automated warehouse system. *2008 IEEE International Conference on Service Operations and Logistics, and Informatics.* Beijing.

Švestka, P., & Overmars, M. H. (1998). Coordinated path planning for multiple robots. *Robotics and Autonomous Systems, 23*(3), 125-152.

Törenlí, A. (2009). Assembly line design and optimization. Göteborg: Chalmers University of Technology.

Wang, C., Wang, L., & Qin Jian, e. a. (2015). Path Planning of Automatid Guided Vehicles Based on Improved A-Star Algorithm. *International Conference on Information and Automation*, 2071-2076.

Wang, J. (1995). Operating primitives supporting traffic regulation and control of mobile robots under distributed robotic systems. *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, *2*, págs. 1613-1618. Nagoya.