UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Tecnura

http://revistas.udistrital.edu.co/ojs/index.php/Tecnura/issue/view/818

DOI: http://dx.doi.org/10.14483/udistrital.jour.tecnura.2017.1.a08

ESTUDIO DE CASO

Global navigation approach for assistant robot

Enfoque de navegación global para un robot asistente

Fernando Martínez Santa[1], Santiago Orjuela Rivera[2], Mario Arbulú Saavedra[3]

Fecha de recepción: 12 de marzo de 2016 **Fecha de aceptación:** 23 de noviembre de 2016

Cómo citar: Martínez S., F.; Orjuela R., S. y Arbulú S., M. (2017). Global navigation approach for assistant robot. *Revista Tecnura, 21*(51), 105-117. doi: 10.14483/udistrital.jour.tecnura.2017.1.a08

Abstract

Context: This work shows a novel navigation approach based on images for an assistant hybrid robot composed by a humanoid and an omnidirectional platform.

Method: This approach introduces a complex space analysis, using Zeros and Poles attraction-repulsion principle. In order to perform the algorithm, an integrated system is developed; this system includes: an external camera to take a global navigation surface view, the assistant robot, and communication devices. Navigation is supported by some digital image processing algorithms and performed using the root location technique.

Results: An integrated system of global navigation with external sensors was successfully implemented for the proposed hybrid robot.

Conclusions: Some simulation and experimental tests will be discussed in order to validate this proposal and the whole system. Additionally, some suggestions for future research are proposed.

Keywords: Navigation, Path planning, Mobile robot, Root Locus, Omnidirectional platform.

Resumen

Contexto: En este trabajo se muestra un enfoque de navegación novedoso basado en imágenes para un robot asistente híbrido compuesto por un humanoide y una plataforma omnidireccional.

Método: Este enfoque presenta un análisis del espacio complejo, usando el principio de atracción y repulsión de polos y ceros. Para desarrollar el algoritmo se desarrolla un sistema integrado, el cual incluye: una cámara externa (para tomar la vista de la superficie global de navegación), el robot asistente, y los algunos dispositivos de comunicación. La navegación está soportada por algoritmos de procesamiento digital de imágenes y llevada a cabo usando la técnica de localización de raíces.

Resultados: Se obtuvo un sistema integrado de navegación global con sensórica externa para el robot híbrido propuesto.

Conclusiones: Algunas simulaciones y pruebas experimentales se discuten con el fin de validar esta propuesta y el sistema entero. También se dan sugerencias para trabajos futuros.

Palabras clave: navegación, planeación de rutas, robot móvil, localización de raíces, plataforma omnidireccional.

1 Electronic Control and Instrumentation Engineer, magister in Electronic Engineer and computers. Professor of District University Francisco José de Caldas. Bogotá, Colombia. Contact: fmartinezs@udistrital.edu.co

2 Electronic Engineer student, Corporación Unificada Nacional de Educación Superior (CUN). Bogotá, Colombia. Contact: santiago.orjuela@cun.edu.co

3 Mechanical Electrical Engineer, doctor in Electrical Electronic Engineering and Automatics, post doctorate in Robotics. Professor of Corporación Unificada Nacional de Educación Superior (CUN). Bogotá, Colombia. Contact: mario_arbulu@cun.edu.co

INTRODUCTION

Many path-planning algorithms had been proposed in order to increase the robot motion autonomy, at any environment. For example, granular control algorithm to solve a labyrinth right path, in which the robot is maintained at the center of corridor during its motion. This approach makes IR sensor linear with very accurate precision (Ahsan & Hasan, 2015), but the application is constrained to a well linearized method. Another approach is (Balestrino & Landi, 2002), which details a multiple vehicle path-planning coordination using root locus approach, but it is constrained to structured work space with known obstacles, to navigate at flat surface without collisions. Other proposals can be discussed, such as neural networks approach which can find paths developing networks (Andrakhanov, 2013) and contribute to solve paths on uncertainty environments, but at a high computation cost. Classical methods such as potential field are another choice at plane surface (Arslan & Koditschek, 2016; Bermudez, Castellar, Montiel, & Ceballos, 2004) and 3D Cartesian space (Chen, Luo, Mei, Yu & Su, 2016; Mac, Copot, Hernandez & De Keyser, 2016; Melingui, Merzouki, Mbede & Chettibi, 2014; Orozco-Rosas, Montiel & Sepulveda, 2015; Vechet, Chen & Krejsa, 2014), with low cost computation time and feasible to real-time applications. The idea of this research is to try innovative path-planning techniques for real-time applications. In some cases, it is necessary to build an environment map because it is not possible to have external information of global navigation surface (i.e. by external camera). Thus when the robot moves, the environment map is built, and then real-time path-planning is developed during the robot navigation at any environment (Fujimori, Murakoshi, & Ogawa, 2004). However, in our case there is no need to build a map, because it is possible to use an external camera. On the other hand, the case of manipulator robot must take into account self-collisions between its links while its end effector is going to the goal (Gaschler, Nogina, Petrick & Knoll, 2014), so 3D path planning algorithms are proposed too.

Our approach takes into account a mobile robot (so self-collision avoidance is not a problem); it also proposes evolutionary algorithms, well known as genetic algorithms, in order to learn the environment performance and got the right path without collisions (Lin, Chen, Liu, & Yu, 2013). However, the trouble with those methods is strong computation and high cost while the system learns. Another approach, often used when the obstacles on the environment are in motion too, improves the path estimation because in real world the mobile robots are in motion in environments with people, other robots, and so on; thus optimization algorithms got the right path (Mohajer, Kiani, Samiei & Sharifi, 2013). This work proposes the root locus approach, which simplifies the path-planning computation and finds the free-obstacle-collision path faster than classical potential fields, neural networks, and evolutionary algorithms. Additionally, our approach uses the single poles and zero attraction and repulsion principles on each obstacle to find the better path. In this case a previous approach contributes the basis to this work (Arbulú Saavedra, Martínez Santa & Montiel Ariza, 2015), when the right path could be found in unstructured work space.

The proposed approach can reach similar or even improved execution time than that of other algorithms, such as Voronoi diagrams (Pehlivanoglu, 2012). Likewise, it works similar to artificial potential field, but it is much better over round environments or with round-shaped obstacles. In general, this approach would be successfully used in radial navigation environments due that the planned paths draw curved trajectories and semi circles, especially semicircular ones. In summary this work contributes to create a solution that:

Reach better execution times than Voronoi diagrams.

- Works similar to the artificial potential field.
- Is useful in environments with radial distribution.
- Can generate soft curved paths.

So, this work is divided as follows: section 2, theoretical background; section 3, proposed approach; section 4, simulations and experimental results; section 5, conclusions; and section 6, future work.

THEORICAL BACKGROUND

HSV image color threshold

The *HSV* image representation is an alternative to *RGB* regular representation of color images. It has three matrices: Hue *H*, Saturation *S* and Value *V*. Hue matrix represents the color itself; Saturation and Value represent the purity and brightness of color, respectively. This images color representation has the advantage to simplify the making of the color threshold by computing one matrix instead of three matrices. The color threshold produces a binary image *B*, comparing the Hue matrix *H* with a low threshold and a high one, as equation (1) shows.

$$B = \left(H < t_h\right) \wedge \left(H > t_l\right) \tag{1}$$

Then image *B* contains pixel of the original image that belong to the color range defined by t_l and t_h. In this paper these thresholds are defined depending on the object to apply threshold segmentation.

Image convolution blurring

The blurring method used in this paper applies a two-dimensional matrix convolution operation, using an average disk-shaped convolution matrix. This means that the summation of point values of the disk is one, which produces an average result after convolution finishes. It also means that the blurred images are soft due that the shape used has not hard angles.

Root locus method

So far, poles and zeros location have been one of the most used method to design controllers for dynamic systems. It is based on the relationship between system root location in the complex plane and its temporal behavior (Arbulú Saavedra *et al.*, 2015). Root locus method creates parametric curves starting from original system poles, and those curves represent the new position of the poles when a feedback constant *K* is changed until tending to infinite. Each path connects a pole with a zero of the same system; then the system has less zeros than poles, the missing zeros tending to infinite, and the correspondent generated paths too.

METHODOLOGY

This global navigation method is implemented for an assistant robot composed by NAO™ and FESTO Robotino™ robotic platforms for redundant manipulation and locomotion (Muñoz Martinez, Gil-Gómez & Garcia Cerezo, n.d.; Zambrano Pérez, 2015), respectively, as it is shown in Figure 1. Its specifications are detailed in Table 1. That navigation is based on external sensing and control; it uses a panoramic IP camera, a computer, and the assistant robot. The complete diagram block of the system is shown in Figure 2. The panoramic IP camera is placed in the middle of the scene, two and a half meters from the floor. The process has four main steps: image acquisition and pre-processing, object detection, path-planning algorithm, and locomotion performance, each one is described in detail below.

Table 1. Assistant robot specifications

	Unit	Specifications
Redundant manipulator (NAO™)	Height	58 cm
	Weight	5.4 Kg
	Autonomy	90 min
	DOF	25
	CPU	Intel Atom @1.6 GHz
	Built-in OS	NAOgi 2.0 (linux-based)
	Compatible OS	Windows, MAC OS, Linux
	Programming languages	C++, Python, Java, MATLAB, Urbi, C, .Net
	Sensors	3 HD cameras, 4 microphones, sonar rangefinder, 2 infrared emitters and receivers, inertial unit, 9 tactile sensors, 8 pressure sensors
	Connectivity	WiFi, Ethernet
Wheeled locomotion (Robotino™)	Height	29 cm
	Weight	5 Kg
	Diameter	45 com
	CPU	Intel Core i5 4.4 GHz, 8 GB RAM
	Compatible OS	Windows
	Programming languages	C/C++, Java, .Net, LabVIEW, MATLAB, Simulink, ROS, Microsoft Rob, Developer St.
	Sensors	9 distance infrared sensors, inductive sensor, 2 optical sensors, Full HD camera
	Connectivity	2 Ethernet, 6 USB, 1 VGA, 2 PCI

Source: own work.

Figure 1. Assistant robot (NAO plus FESTO Robotino)

Source: own work.

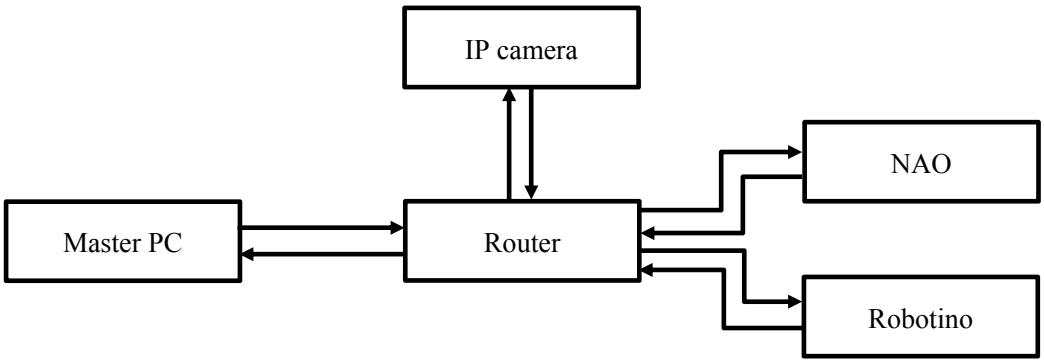

Figure 2. Complete diagram block of system

Source: own work.

Image acquisition and pre-processing

The first step is image acquisition, which uses a fisheye panoramic IP camera. The camera is able to do the geometric transformation for viewing a square scene from the fisheye lens; then it obtains an image of 1600x1200 pixels at 30Hz (figure 3). After that, a region of interest *(ROI)* is selected in order to limit only the navigable area, as shown in figure 4. The *ROI* is a 1300x470-pixel image centered in the original image.

Figure 3. Image acquisition of navigation scene

Source: own work.

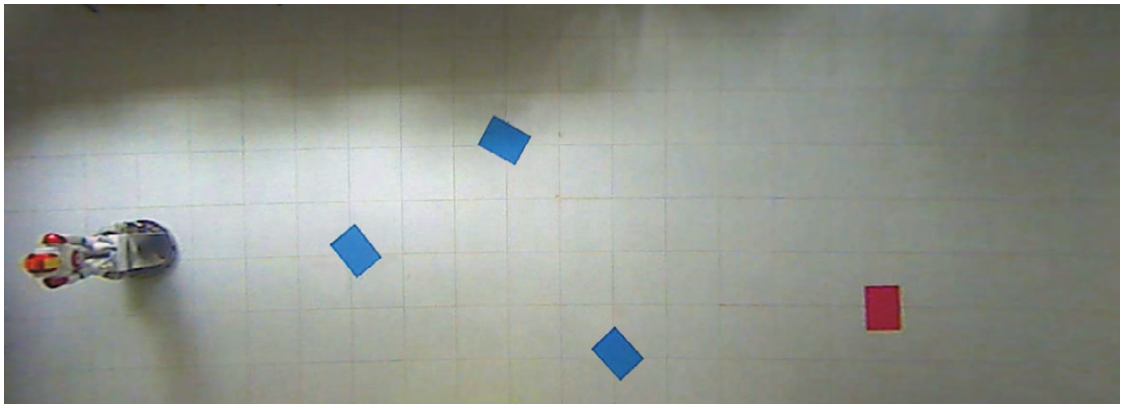

Figure 4. Region of interest *(ROI)* selection

Source: own work.

The acquired image is turned into *HSV* representation from *RGB*, in order to take advantage of the colors organization in only one matrix (Hue), which facilitates the color threshold. After that, a reduction of the brightness pattern effect is done changing saturation *S* and value *V* matrices by blurring these matrices in order to obtain matrices S_b and V_b. Then the proposed correction is a weighted matrix subtraction, as shown in the following equations (2) and (3).

$$S_r = S - S_b * k_s \quad (2)$$

$$V_r = V - V_b * k_v \quad (3)$$

Object detection

This step is used to locate all the objects in the scene, including obstacles, robot and target. Then the system uses color marks: blue for obstacles and red for target. Likewise, for the robot there are two marks, orange and yellow, not only for locating it, but detecting its direction, as shown in Figure 4. After that, a color threshold for all the marking colors is done over hue matrix *H* from the image in *HSV* representation, which allows to obtain four binary images: one for the obstacles, one for the target and two for the robot. The binary image of obstacles is shown in Figure 5. On the other hand,

Figure 5. Binary image of obstacle segmentation

Source: own work.

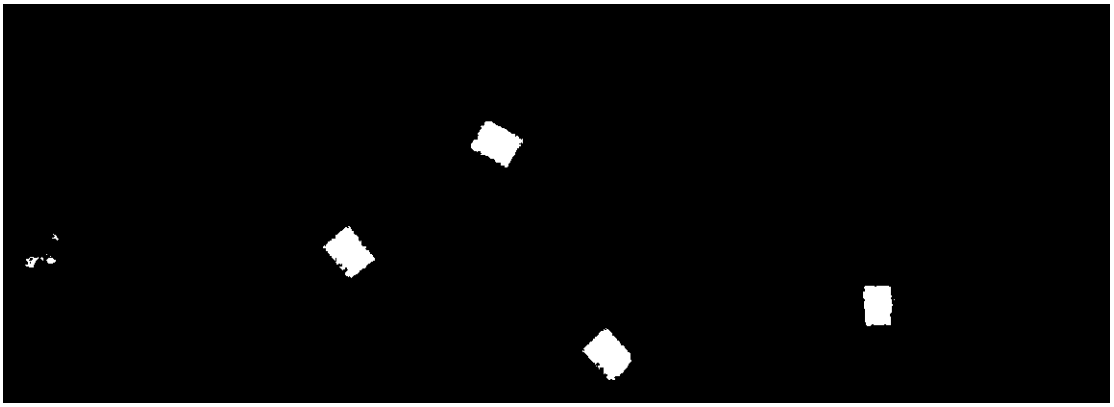

Figure 6. Binary image of all the objects in the scene

Source: own work.

Figure 6 shows a binary image with all objects in the scene, obstacles target point, and robot. Then, all locations in the space are found through the objects centroids in binary images, likewise the initial direction of robot is computed as the vector created between centroids of the orange and yellow marks.

Path-planning

Using root locus method, it is possible to obtain navigable paths, by setting a zero and a pole as starting and ending points on the scene (Arbulú Saavedra *et al.*, 2015). For doing that, the first step is matching the original image coordinates with a complex plane; in this case it was used the upper-right region of the complex plane, where the real and the imaginary coordinates are positive. Then, all the possible points to be placed correspond to a complex coordinate. Generally in a root locus plot, when the system has a pole and a zero, both complex, it generates arc-shaped path between them, tending to the upper side. In order to avoid this arc getting out of the scene, a pole and a zero are placed on the upper-right and the upper-left corners. Each corner root has to be placed as the same kind (pole or zero) of nearest root (starting point or target). After that, a pole is placed on the robot location and a zero on the target one (the example image is shown in Figure 7, upper part), which correspond to equation (4). Each placed root has its conjugated pair in the lower-right region of the plane, but they are not shown.

$$H_1(s) = \frac{(s+1-4i)(s+1+4i)}{(s+14-6i)(s+14+6i)} \qquad (4)$$

In figure 7 there are two different solutions given by the algorithm. The first one places the scene in the upper-right part of complex plane, and the second one in the lower-right part. They generate concave and convex paths respectively. Both paths are large curves instead of straight lines due that root locus parametric functions tend to generate semicircular paths when starting and ending points are horizontal or diagonal. The only way to obtain a straight line is when those points are vertical. The proposed algorithm traces a straight line instead of using root locus technique, when there are not obstacles in the middle of starting and ending points.

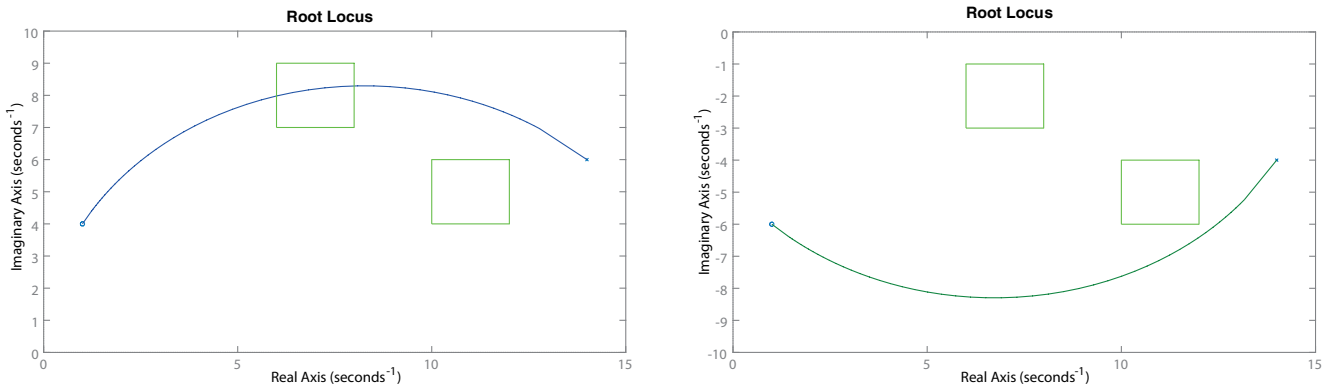

Figure 7. Regular path shape between two points using root locus

Source: own work.

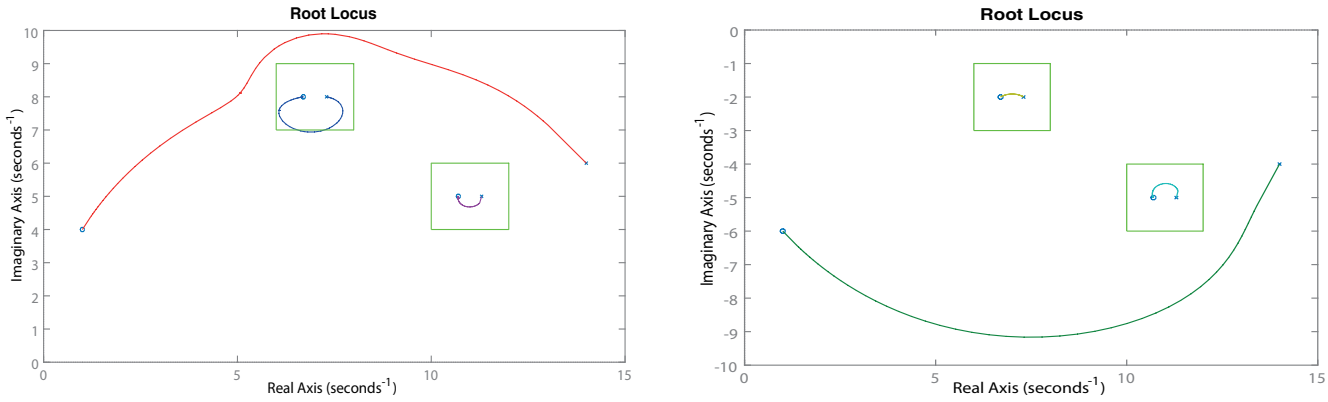

Figure 8. Path shape obtained using root locus, after adding a pole and a zero per obstacle

Source: own work.

Both examples of Figure 7 used the same obstacles; the first one shows a collision with one of them and the second one passes too close to the other, so it is necessary to improve those paths. Poles and zeros behavior looks like positive and negative electric charges, equal roots repel themselves and different roots attract themselves. So, if the starting point has a pole, the nearest obstacle has to have a pole too in order to repel the robot to that one. At the same time a contrary root has to be placed not to change the original path between initial and target points. This additional root has to be placed close to the first one, so the distance between the new pole and zero is proportional to the robot radius using the factor f, in order to prevent the robot to pass through spaces narrower than it. Therefore, it is necessary a pair of pole and zero for each obstacle in the scene, which is represented as one transfer function per each obstacle, such as Figure 8 and equations (5), (6) and (7).

$$H_2(s) = \frac{(s+10.7-5i)(s+10.7+5i)}{(s+11.3-5i)(s+11.3+5i)} \quad (5)$$

$$H_3(s) = \frac{(s+6.7-8i)(s+6.7+8i)}{(s+7.3-8i)(s+7.3+8i)} \quad (6)$$

$$H_4(s) = H_1(s) * H_2(s) * H_3(s) \quad (7)$$

As shown in Figure 8 the obtained path is not necessary the shortest one, because the original path without obstacles (Figure 7) is a large curve instead of a straight line. But both paths effectively avoid obstacles from the start to the end.

The obtained paths can be turned into more straight-shape ones mapping the original scene into the complex plane, changing the scale of *x* axis and continue with the same scale of *y* axis, that produces a path composed by a section of a bigger circle than the original and obtaining a flatter curve, as shown in Figure 9.

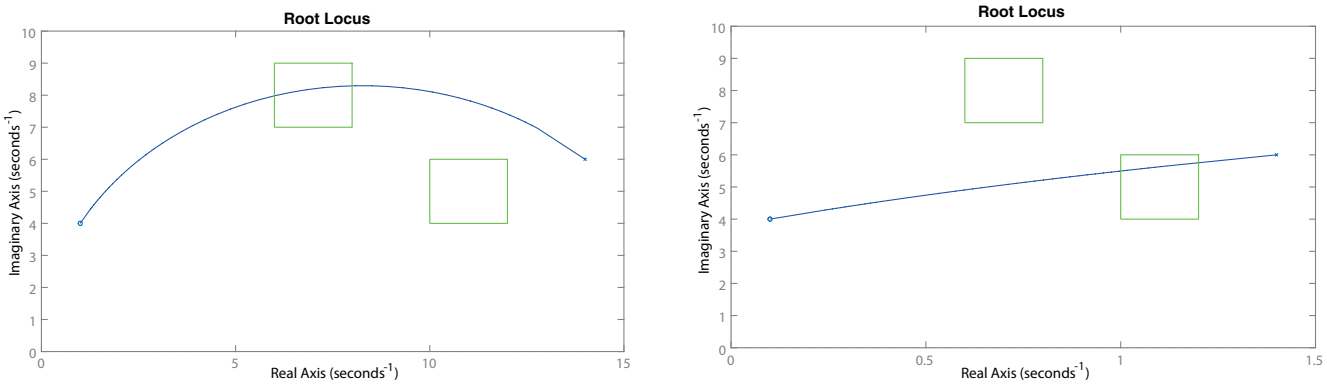

Figure 9. The path is gotten straight, scaling only the *x*-axis

Source: own work.

In addition, the choice of the path between concave and convex can be done detecting the distribution of obstacles inside the scene by means of image processing, using the center of all obstacles in the scene and determining if most of them are located in the upper or lower side of scene.

Locomotion performance

Once the path is obtained, the software decimated it in order to reduce the information managed by the robot, and sends it to the robot for navigating the scene. Around 16 points per path are sent via WiFi. In order to keep the algorithm as simple as possible, Robotino platform was used as a differential robot, in spite of being an omnidirectional (Mao, Wiedmann & Chen, 2011; Mbede, Melingui, Zobo, Merzouki & Bouamama, 2012; Melingui, Chettibi, Merzouki & Mbede, 2013). Then, the robot has to move in straight lines, stop and turn over its axis to perform the movement on the path; the robot repeats this process for each point of the given path.

RESULTS

Using the proposed approach, it was obtained the navigation path as shown in Figure 10, matching the original *(x;y)* image coordinates with the complex plane. For this specific application, *x*-axis of input image is larger than *y*-axis, and then the first generated path is a big semicircle that gets out of the scene. Therefore, it is necessary to add other two roots: one pole and one zero in the upper-left and upper-right scene corners, *p* and *z* respectively, taking into account that these should match with the same kind of root with the nearest main point (start or target); as shown in equations (8) and (9).

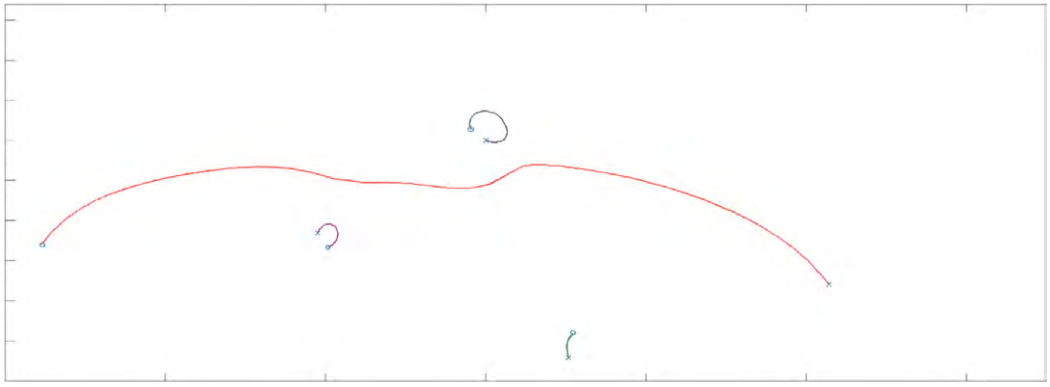

Figure 10. Navigation path obtained by a root locus plot

Source: own work.

$$H_5(s) = \frac{(s + z_r + z_i * i)(s + z_r - z_i * i)}{(s + p_r + p_i * i)(s + p_r - p_i * i)} \quad (8)$$

$$H_6(s) = H_4(s) * H_5(s) \quad (9)$$

After that, the generated path is drawn on the scene image in order to assure that it will not generate collisions with obstacles or borders of the scene, as shown in Figure 11. Then, the decimation of path vector produces a minimum amount of navigation points, which are sent to the robot, as shown in Figure 12.

The difference between the generated path and the one performed by the robot is shown in Figure 13, which shows an accumulative error over the resultant path. Additionally, the average distance d_a and relative error between paths e are computed as follows in equation (10) and (11), where d_m is the maximum distance in the original image. The average distance between points of both paths gave as a result 0.56 (in distance units) and the relative error was equal to 4.4%.

$$d_a = \sum_{j=1}^{n} \frac{\sqrt{(x_2 j - x_1 j)^2 + (y_2 j - y_1 j)^2}}{n} \quad (10)$$

$$e = \frac{d_a}{d_m} \quad (11)$$

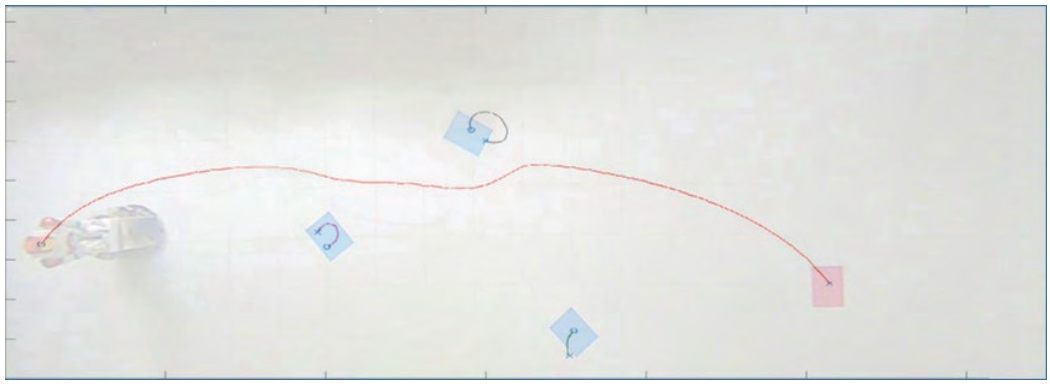

Figure 11. Original scene plus navigation path

Source: own work.

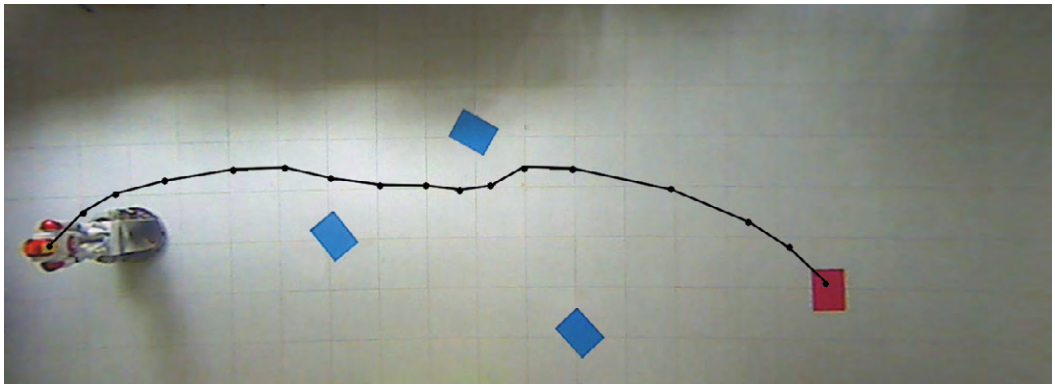

Figure 12. Navigation path as point vector representation

Source: own work.

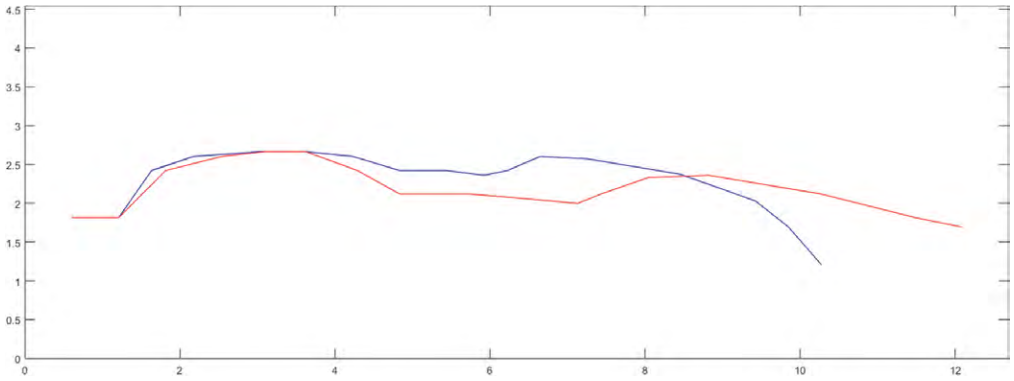

Figure 13. Difference between generated path and the performed by the robot (the red line represents the actual path while the blue line represents the generated path)

Source: own work.

CONCLUSIONS

The proposed approach of root locus for mobile robot navigation can solve the free obstacle path on flat surface successfully, so the assistant robot can navigate on real static environment safely. The generated paths are semicircles in most of the cases, which are not always the shortest way from the starting point to the target, but at the same time those paths are softer than the generated by other path planning methods. Additionally, this approach works similar to artificial potential fields but generates semicircular paths; it can execute as fast as Voronoi diagrams or faster. Obtained paths can be optimized in time, mapping the scene into a complex plane, scaled only in the *x*-axis, to obtain flatter paths than those shown here. Finally, the system integration (IP camera, mobile robot, communication protocols, and navigation algorithm) was successfully experimentally validated.

FUTURE WORK

It is possible to take advantage of the round shape of paths created by root locus algorithm by applying it directly to original images generated by the fish-eye IP camera. It means that there is no need to apply any geometric transformation at

the beginning of the process, but just a coordinate transformation instead, at the end. This method would briefly reduce the computing time, working on separate points instead of the complete image when the geometric transformation is performed.

Taking into account the error presented due to the robot position referring to the camera, it is important to do a point-of-view x-coordinate correction in order to reduce the accumulated error presented in the whole generated path.

It is necessary to change the robot-to-server communication approach in order to avoid the position error presented, due to communication missing. The way to solve this problem would be generating our own communication protocol and load a specific firmware on Robotino platform that performs each movement command, even when communication fails in the middle of that movement.

In order to achieve a smother movement of assistant robot, it is important to use Robotino platform with the complete control of its three motors as omnidirectional platform. Once it is done, a great improvement would be reached sending not only isolated points of path to the robot, but sending the equation that generates the path in order to generate the path internally. This proposal will required certain compute power, due that the robot processor will need to solve in real time a higher order equation; that depends on both the number of presented obstacles and the correspondent speed profile sent to the motors. Combining self-generation path and omnidirectional movement, the performance of prototype would improve considerably.

Additionally, some features of algorithm as *x*-axis scaling, position and separation of obstacle roots, additional poles and zeros placement, etc., can be chosen automatically using an optimization algorithm like genetic algorithms. This fact could improve the generated paths, but gets the execution time of general approach higher.

ACKNOWLEDGMENT

We would like to give acknowledgment to the Electronic Program, technical staff, and Research Direction from Corporación Unificada Nacional de Educación Superior, for their support and funding this project, under the "Robot asistencial para hogares y oficinas" research project.

REFERENCES

Ahsan, F. & Hasan, K.M.U. (2015). Seeker: Autonomous maze-navigating and ball-potting robot. In 2015 *International Conference on Open Source Systems & Technologies (ICOSST)* (pp. 52–57). IEEE. http://doi.org/10.1109/ICOSST.2015.7396402

Andrakhanov, A. A. (2013). Navigation of Autonomous Mobile Robot in Homogeneous and Heterogeneous Environments on Basis of GMDH Neural Networks. In *Proceedings of 4th International Conference on Inductive Modelling (ICIM-2013)* (pp. 133–138).

Arbulú Saavedra, M.R., Martínez Santa, F. & Montiel Ariza, H. (2015, November 4). Metodología para el uso de la técnica de localización de raíces en la planeación de rutas para robots móviles. *Revista Tecnura*. http://doi.org/10.14483/udistrital.jour.tecnura.2015.4.a04

Arslan, O. & Koditschek, D. E. (2016). Exact robot navigation using power diagrams. In *Robotics and Automation, 2016 IEEE International Conference on (accepted)*.

Balestrino, A. & Landi, A. (2002). High level path coordination for multiple vehicles and reciprocal root locus. *IFAC Proceedings Volumes*, *35*(1), 415–420.

Bermudez, G., Castellar, L.A.R., Montiel, H. & Ceballos, M. (2004). Aplicación del método de campos de potencial artificial para un robot móvil autónomo. *Revista Tecnura*, *7*(14), 86–96.

Chen, Y., Luo, G., Mei, Y., Yu, J. & Su, X. (2016). UAV path planning using artificial potential field method updated by optimal control theory. *International Journal of Systems Science*, *47*(6), 1407–1420.

Fujimori, A., Murakoshi, T. & Ogawa, Y. (2004). Navigation and local path planning of mobile robots with real-time map-building. *Integrated Computer-Aided Engineering*, *11*(3), 281–288. Retrieved from https://search.ebscohost.com/login.aspx?direct=true%7B&%7Ddb=aph%7B&%7DAN=13857304%7B&%7Dlang=es%7B&%7Dsite=ehost-live

Gaschler, A., Nogina, S., Petrick, R.P.A. & Knoll, A. (2014). Planning perception and action for cognitive mobile manipulators. In *IS&T/SPIE Electronic Imaging* (p. 90250D—90250D).

Lin, C.-J., Chen, Y.-L., Liu, C.-H. & Yu, S. K. (2013). Path planning of a Mobile Robot Using Real-coded Genetic Algorithm Based Simultaneous Exploration. In *2nd International Conference on Advances in Computer Science and Engineering (CSE 2013)*.

Mac, T.T., Copot, C., Hernandez, A. & De Keyser, R. (2016). Improved potential field method for unknown obstacle avoidance using UAV in indoor environment. In *2016 IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMI)* (pp. 345–350).

Mao, Y.F., Wiedmann, H. & Chen, M. (2011). Autonomous Mobile Robots and Development of Vision Based Automotive Assistance Systems. *Applied Mechanics and Materials*, *121–126*, 2416–2420. http://doi.org/10.4028/www.scientific.net/AMM.121-126.2416

Mbede, J.B., Melingui, A., Zobo, B.E., Merzouki, R. & Bouamama, B. O. (2012). zSlices based type-2 fuzzy motion control for autonomous robotino mobile robot. In *Proceedings of 2012 IEEE/ASME 8th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications* (pp. 63–68). IEEE. http://doi.org/10.1109/MESA.2012.6275538

Melingui, A., Chettibi, T., Merzouki, R. & Mbede, J. B. (2013). Adaptive navigation of an omni-drive autonomous mobile robot in unstructured dynamic environments. In *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)* (pp. 1924–1929). IEEE. http://doi.org/10.1109/ROBIO.2013.6739750

Melingui, A., Merzouki, R., Mbede, J. B. & Chettibi, T. (2014). A novel approach to integrate artificial potential field and fuzzy logic into a common framework for robots autonomous navigation. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, *228*(10), 787–801.

Mohajer, B., Kiani, K., Samiei, E. & Sharifi, M. (2013). A New Online Random Particles Optimization Algorithm for Mobile Robot Path Planning in Dynamic Environments. *Mathematical Problems in Engineering*, *2013*, 1–9. http://doi.org/10.1155/2013/491346

Muñoz Martinez, V.F., Gil-Gómez, G. & Garcia Cerezo, A. (n.d.). *Modealdo cinematico y dinamico de un robot móvil omni-direccional,* (April 2016), 9.

Orozco-Rosas, U., Montiel, O. & Sepulveda, R. (2015). Pseudo-bacterial potential field based path planner for autonomous mobile robot navigation. *International Journal of Advanced Robotic Systems*, *12*.

Pehlivanoglu, Y.V. (2012). A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV. *Aerospace Science and Technology*, *16*(1), 47–55. http://doi.org/10.1016/j.ast.2011.02.006

Vechet, S., Chen, K.S. & Krejsa, J. (2014). Hybrid navigation method for dynamic indoor environment based on mixed potential fields. In *Mechatronics 2013* (pp. 575–582). Springer.

Zambrano Pérez, V.D. (2015). *Implementación de algoritmos de determinación de rutas para el robotino de festo*. Escuela Politécnica Naciona EPN | Ecuador. Retrieved from http://bibdigital.epn.edu.ec/handle/15000/11042