International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 12 (2018) pp.10412-10418

Use of CNNs for follower mobile agents with safe distance

Paula C. Useche Murillo, Robinson Jiménez Moreno, Javier O. Pinzón Arenas.

Department of Mechatronics Engineering, Militar Nueva Granada University, Bogotá, Colombia

Abstract

In the following paper, it is presented the development of a leader-follower application between two mobile agents, with security distance control between them, where Convolutional Neural Networks (CNN) were used to detect proximity between both objects, the free-motion agent and the follower, and geometric analysis to determine the locations of mobile agents in space. The application was executed in a virtual environment with the objective of establishing the basic associations of travel for collaborative agents, and in which the existence of a camera in the upper part of the work area was emulated, generating as output the wheel speed of the follower for its subsequent implementation in a real environment. A CNN was implemented with a 97.5% accuracy for the classification of the distances "Far" and "Close" between the mobiles, and a margin of safety distance between 140 and 125 pixels away, in a work area of 480x480 pixels, achieving a follow-up action to a leading robot, without collisions.

Keywords: Mobile Robotics, Security Distance, Convolutional Neural Networks, Geometric Analysis, Mobile Objects Tracking.

INTRODUCTION

The development of mobile robotics has expanded, since early 2000, to multi-robotic systems where numerous vehicles must perform tasks autonomously and / or jointly, such as transportation of objects, multi-robotic cooperation architectures, location, mapping and exploration, among others [1]. The interest in this topic has focused on fields such as cooperation, learning [2] and movements with avoidance of obstacles [3], where it can be found applications such as the development of an autonomous wheelchair that avoids obstacles and maintains the safety of the user [4].

On the other hand, the movement of mobile agents in uncontrolled environments has generated the need to develop trajectory planning algorithms, such as the one presented in [5], where it is sought to set trajectories that meet safety, length and smoothness objectives in the movement. Another example is presented in [6] which uses genetic algorithms to plan the trajectories of a mobile agent.

The planning of trajectories and the evolution in cooperative and interactive robotics with the dynamic environment that surrounds it, has led to the development of applications where trajectories are set for multi-robotic agents that track mobile elements. For instance, there can be found the work presented in [7], in which pedestrians are monitored in real time, maintaining a safe distance through PID controllers and on-off controllers. In the same way, the application presented in [8] follows a route with evasion of obstacles maintaining a preset safety distance.

As can be seen, the interaction of a mobile with its environment requires a distance control that ensures its displacement without collisions, especially if there are moving objects around it. The works presented in [7] and [8] propose two different ways of maintaining a margin of safety for a mobile agent with respect to its environment, however, they do not evaluate the possibility of finding more than two agents that, apart from having to maintain a safe distance, must carry out some task altogether or move in a parallel way in a certain environment. Therefore, in this work, it is proposed to follow a moving vehicle by means of a robotic agent, where a margin of safety distance is assured at all times, between both mobiles, using CNNs for the detection of proximity between the elements, where the developed algorithm allows extending its application to a multi-agent environment, through the identification of parameters.

As presented in [9], CNN is a type of neural network that uses convolution kernels to extract relevant information from images and generate a classification of patterns or objects from the training of a database that contains images of the categories to be classified. Its architecture varies according to the need of features extraction, where the programmer selects and organizes the layers of the network, and defines the parameters of each of them according to the information required to extract from the categories.

In [10] an application example of a CNN is presented where it is used for the detection of geometric coincidence between two images, while in [11] it is used for the classification of tissue samples of head and neck excised for the detection of cancer in said areas, using hyperspectral images.

Based on the above, the following paper presents the development of a follower application with safety distance, where the movement of two mobiles is simulated, the purpose of which is to make one follow the other while maintaining a certain distance. This algorithm offers the possibility of conditioning the virtual work environment to a real one, where an aerial image of the workspace is received as input and the speed of the wheels for a differential mobile is generated as output.

Next, it is presented the five main sections, in which the article was divided, where the first "Methods and Materials" exposes the working conditions of the simulation. The second section, "CNN for distance detection", presents the obtaining of a CNN for the recognition of distance between two objects. The third, "Follow the moving object ", explains the logic used in the development of the algorithm. The fourth, "Results and Analysis", presents the results obtained, and the fifth "Conclusions" give the conclusions reached and possible changes to achieve better results.

METHODS AND MATERIALS

The virtual environment consists of a 2D environment where two mobiles are simulated, one that moves freely through the work area, represented by a green circle, and another that follows it at a certain distance, marked with a blue triangle, as shown in Figure 1. To more closely resemble the tests to a real environment, soil was added to the simulation, with the possibility of changing it before each execution, and the positions of the mobiles were initialized randomly.

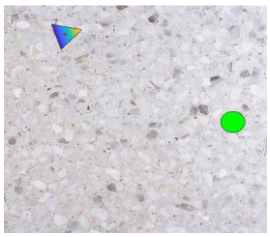
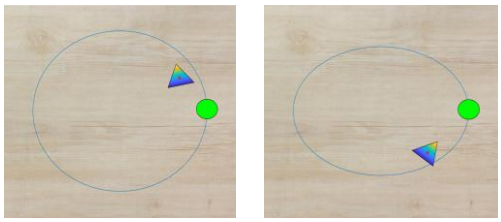


Figure 1. Workspace

For the simulation, the free mobile (or leader) was programmed with two types of trajectories, one elliptical and one circular, both performed at constant speed, and the other agent was conditioned to follow the leader, moving both forward and backward direction, with the triangle's clearest vertex being the front of the vehicle. Figure 2 shows the circular and elliptical path that the leader performs, drawn with a continuous line.



a. b.

Figure 2. a) Circular and b) elliptical trajectory of the leader

The center of mass of each mobile agent was located in the center of each vehicle, with the wheels on both sides of it, as shown in Figure 3, where *r* represents the radius of the wheels, *L* the distance between them and *Ang* the angle of rotation of the mobile with respect to the horizontal.

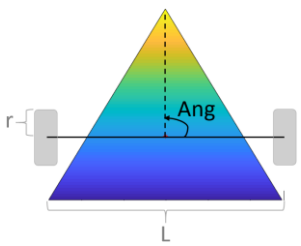


Figure 3. Follower Mobile Agent

A radius of 2cm was defined for the wheels, a distance *L* of 55.4cm for the follower, and for the leader, the trajectory was divided into 250 points and its location was iterated between each of them, simulating a smooth displacement on the trajectory.

The angular velocity of the wheels for the follower oscillates between 8 rad/s and 28 rad/s according to the distance it has with respect to the leader, and a time differential of 0.1s was used to calculate the next position of the vehicle based on the speed of each of its wheels. For which the equation shown in (1) was used where w_l is the speed of the left wheel, w_r the speed of the right wheel, and θ the angle of rotation of the agent. In [12] the process for obtaining the direct kinematics of a differential mobile is shown.

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = 0.1 * \begin{bmatrix} -\frac{r}{2}\sin(Ang) & -\frac{r}{2}\sin(Ang) \\ \frac{r}{2}\cos(Ang) & \frac{r}{2}\cos(Ang) \\ -\frac{r}{L} & \frac{r}{L} \end{bmatrix} * \begin{bmatrix} w_l \\ w_r \end{bmatrix} \quad (1)$$

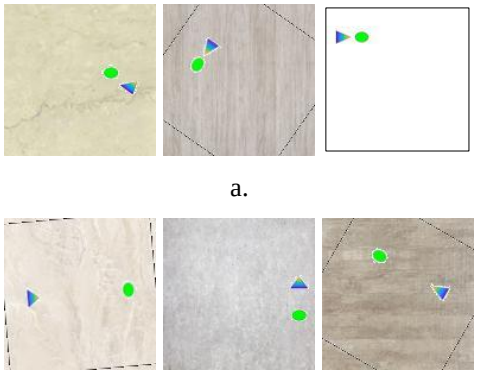
The time differential of 0.1s allows emulating a smooth and continuous movement for the follower, without sudden jumps of position between one moment of time and the next.

The dimensions of the workspace are 480x480 pixels, where the leader has a radius of 22 pixels and the follower has a diagonal of 32 pixels between the center of mass and each of its points. The radius of the trajectory for the free mobile is 180 pixels with center at 240x240 pixels where, to generate the ellipse, the Y coordinates are multiplied by a factor of 0.7, as shown in (2), where *n* is each of the points of the trajectory and ranges from 1 to 250.

$$\begin{aligned} x &= 180 * \cos(n * 1.44°) + 240 \\ y &= 0.7 * 180 * \sin(n * 1.44°) + 240 \end{aligned} \quad (2)$$

CNN FOR DISTANCE DETECTION

A CNN was trained to detect the "Far" and "Close" states among mobile agents, where 1030 images were used per category, all at 128x128 pixels in color, with different types of ground. The database was obtained from an original database with 373 images for the "Close" category and 608 images for the "Far" category, each of them without a back ground, where the Data Augmentation techniques developed in [13] were used to generate the training database, which has the original images, rotations of them and background changes, as can be seen in Figure 4.



a.

b.

Figure 4. Training database for a) Close and b) Far

For the CNN, the architecture shown in Table 1 was used, where small filters (from 7x7 to 3x3 pixels) and three normalizations layers were set to avoid a high degree of shifting of the activations of the mobiles agents between the output image of one layer and the next, in order to avoid confusion within very short distances of the "Far" category with "Close" type distances. There were used the abbreviations CONV for the convolution layers, RELU for the Rectified Linear Units, MAXPOOL for the Max Pooling, NORM for the normalizations, and W and H for the width and height of each filter, respectively.

Table 1. CNN Architecture

LAYERS	Filter Size WxH	Stride
CONV+RELU	7x7	1
CONV+RELU	6x6	1
MAXPOOL	**3x3**	**2**
CONV+RELU	5x5	1
CONV+RELU+NORM	4x4	1
MAXPOOL	**3x3**	**2**
CONV+RELU+NORM	3x3	1
CONV+RELU+NORM	3x3	1
MAXPOOL	**3x3**	**2**
Fully Connected	2	--

The architecture used allowed obtaining activations such as those shown in Figure 5, where there is a complete extraction of the mobile agents with respect to their surroundings and a constant distance between them with the passage of the layers, with the white pixels being the most important characteristics extracted by each convolution of the image.

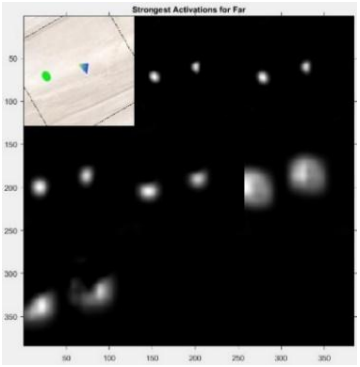


Figure 5. Activations for "Far"

There were used 400 epochs for the training, achieving a 97.5% accuracy. 100 test images per category were used, where only 5 images from the "Close" category were misclassified as "Far", as shown in the confusion matrix of Figure 6, in which the columns represent the actual categories, the rows the network classifications and the lower right corner the total accuracy. The category "Far" is number 1, and "Close" is number 2.

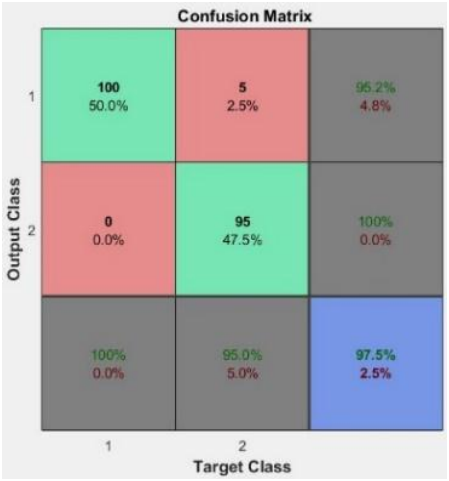


Figure 6. Confusion matrix

FOLLOW THE MOBILE OBJECT

To perform the application of following a mobile object, the sequence shown in Figure 7 was established, where initialization of variables is carried out, which includes the random position of the follower, the selection of the ground in the simulation, the creation of the trajectory of the leader, the first obtaining of the state "Far" or "Close" according to the CNN classification, and the definition of other intermediate variables. Then, in the following stages, the whole process of detecting the mobiles is done, the direction and the speed of turn for the follower is set, the movement is generated, and the whole scenario is re-evaluated with CNN before resuming step 2.

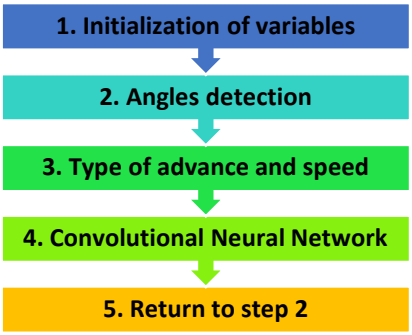


Figure 7. Algorithm flowchart

A. Angles Detection

In the second stage, it is considered that there is no prior knowledge of the positions of the agents in the workspace, but that a picture of the environment has been received as input, so it is proceeded to extract the mobile agents from the image using the activation of the first convolution layer, and obtain other characteristics such as the angle between the leader and the follower, and the current direction of the follower.

Figure 8 shows the extraction of vehicles, where the network filters the green and blue colors of each mobile, leaving the tip of the triangle as if it were part of the background and clearly marking the geometric shape of each element.

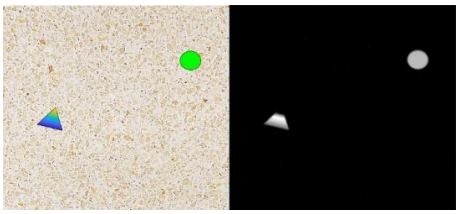
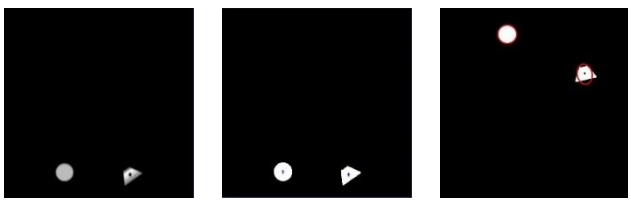


Figure 8. Extraction of vehicles

From the obtained activation, the image is binarized, the centroids of each object are obtained, and the inclination of the triangle is extracted, as shown in Figure 9, generating an ellipse on both objects.



a. b. c.

Figure 9. Example of a) activations, b) centroids and c) inclination

With the coordinates of the centroids, the angle between both vehicles (θ_{ct}) is obtained using (3), where the coordinates with subscript "c" correspond to the centroid of the circle and the others to the one of the triangle, and with the inclination of the follower an advance direction between 0º and 180º is estimated, where it is not possible to know which side of the ellipse the tip of the triangle is located.

$$\theta_{ct} = \tan^{-1}\left(\frac{Y_c - Y_t}{X_c - X_t}\right) \quad (3)$$

The spatial coordinates were taken with respect to the zero point of the workspace located in the lower left corner of the image.

On the other hand, to know which centroid is the circle and which of the triangle, it is gotten the difference between the major and minor diagonal of the ellipse that covers the vehicles, and the one with the biggest difference is the one with the centroid corresponding to the triangle.

B. Type of advance and speed

Once the angle detection has been made, the difference between θ_{ct} and the inclination of the triangle (*Ang*) is calculated, as shown in (4), and depending on the result, the type of advance is established, whether turning on the axis in a clockwise direction or, on the contrary, moving forwards or backwards. If the difference (*difA*) is between -7° and 7°, it is considered that the follower can move forward or backward, but if it is above 7° the mobile is rotated counterclockwise and if it is below -7°, the turn is made in favor of the clock hands.

$$difA = \theta_{ct} - Ang \quad (4)$$

Then, based on the classification made by the network (*label*) and its percentage of recognition (*score*), a forward speed is defined, which consists of a scalar variable *Run* that is multiplied by the standard mobile speeds to generate an increase in speed depending on the distance. The amount of increase and standard speeds of the mobile were determined experimentally, causing the vehicle to move through the environment using different types of speeds.

Table 2 shows the standard wheel advance values and the speed variation set from the labels (Far and Close) and scores obtained by CNN, where $Giro$ represents the standard speed of the follower during turns, and Av the standard speed during forward or reverse.

Table 2. Speed variation

Close	***Far***			***Standard speed***	
--	$score \geq 95\%$	$score < 95\%$ $score \geq 70\%$	$score < 70\%$	w_l Rad/s	w_r Rad/s
$Run = 4$	$Run = 7$	$Run = 6$	$Run = 5$	$Giro = \pm 2$ $Av = \pm 4$	$Giro = \mp 2$ $Av = \pm 4$

Because the algorithm does not allow knowing if the follower is looking towards the leader or not, even though the difference of their angles is close to zero, a conditional was established that generates an advance in the mobile and, after 3 iterations, it evaluates if the distance between both objects increased or decreased, generating a change of direction in case of an increase, by means of the denial of the variable "Run". The 3 iterations prevent the agent from changing direction with any slight change of distance and, at the same time, avoid to move too far from the leader before correcting its forward direction.

Finally, the direct kinematics of the mobile is evaluated, replacing in (1) the current angle of rotation of the follower (*Ang*), the speed of the wheels previously obtained, the radius of their wheels, and the distance *L* between them.

C. Convolutional Neural Network

After generating the advance of the follower, an image of the work environment is captured and entered into CNN to re-evaluate the distance between the vehicles and repeat the process from step 2 of Figure 7, using the new classification.

In Figure 10, it is possible to observe the variation of the percentage of classification as the follower approaches the leader, advancing in a straight line, without the leader moving. There it is possible to see how the percentage of "Far" decreases as the follower approaches the "Close" state, and then how a rapid increase is generated when the state changes. The change of percentage in said section generates, according to Table 2, a smooth reduction of speed, preventing the follower from abruptly approaching the leader when they are close. Additionally, the speed of the follower is reduced in case of not having a lot of certainty of the classification, as it is observed in the peaks of between 55% and 65% of recognition.

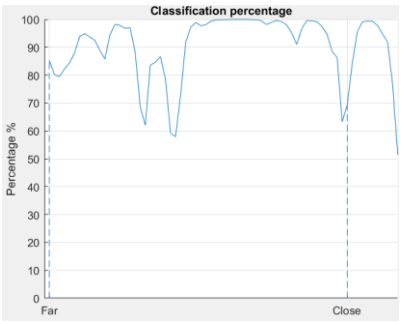
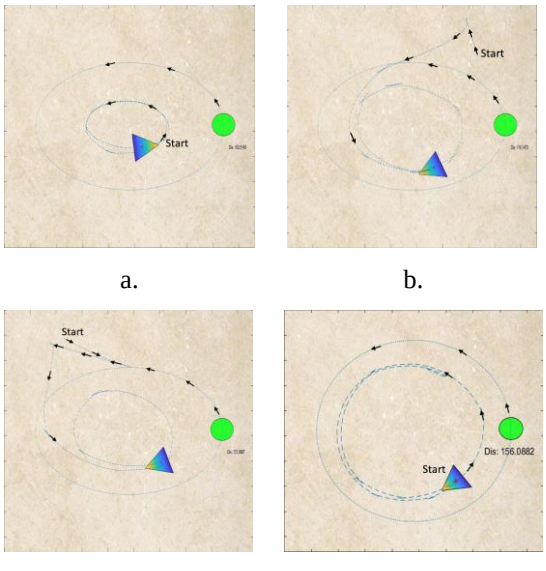


Figure 10. Classification percentage

RESULTS AND ANALYSIS

Figure 11 shows 4 different trajectories performed by the follower, where in the case of Figure 11a the initial position of the follower is within the ellipse made by the leader, so the vehicle only adjusts its orientation towards the leader, without moving from its place, and begins to move behind it, making an elliptical path within the greater ellipse.

In the case of Figure 11b, the follower began its journey at a point outside the ellipse and, additionally, the leader began moving in the direction of the follower, so the first reaction of the follower was to move away from the leader to avoid a collision, and begin to follow it only after it moved away to continue with the ellipse, where the rest of the follower's trajectory resembled that of Figure 11a.



a. b.

c. d.

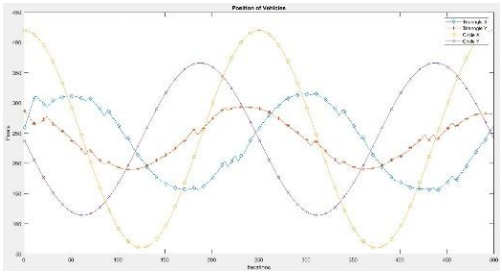
Figure 11. Trajectories of the follower

In the case of Figure 11c, the initial position of the follower was very close to the upper left corner, so its first reaction was to approach the leader to reduce the distance between them and then, when the leader got too close, the follower changed direction and moved away to avoid a collision and wait for the leader to continue its trajectory before returning to follow it, repeating the elliptical trajectory of the first case.

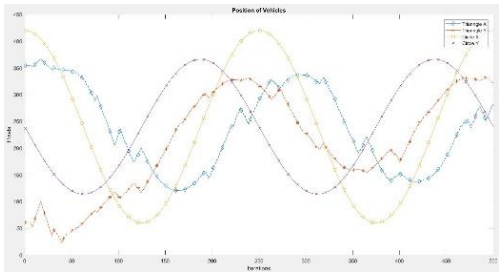
The case of Figure 11d is very similar to that of Figure 11a, only that the trajectory of the leader was circular.

Each of the positions X, Y, for the four cases shown, were plotted in the images of Figure 12, where the horizontal axis shows the number of iterations and the vertical the position in pixels of the center of mass of each agent throughout the trajectory, which allows observing how the follower resembles its behavior to that of the leader, always maintaining a constant safety distance.

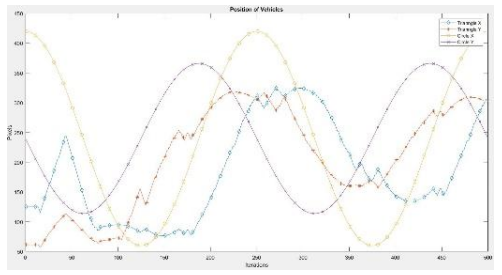
The circles and Xs correspond to the X-Y positions of the circle, while the diamonds and crosses correspond to the X-Y positions of the triangle, respectively.



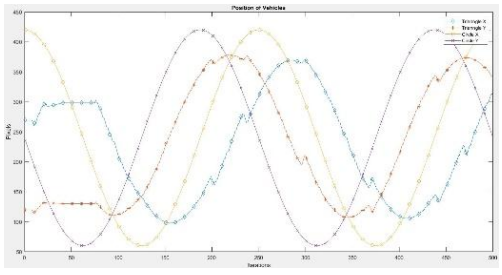
a.



b.



c.



d.

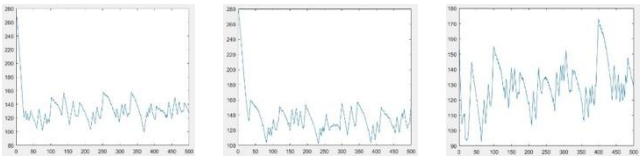
Figure 12. Position tracking with distance safety margin

In Figures 12a and 12d it can be seen how both the X and Y positions of the follower follow the behavior of the leader's trajectory, behaving as sinusoidal waves with a smaller amplitude than the trajectory they try to follow, and with a slight lag.

On the other hand, for the cases of Figure 12b and 12c, the mobile follower tries to approach the positions of the leader during the first 100 iterations and, after being close enough, begins to follow it, maintaining the safety distance.

Despite the fact that in all the presented cases an adequate following of the leader was achieved, the trajectories of Figures 12b and 12c showed a greater oscillation in their curves. This is due to the difficulty of the mobile to avoid a collision and accommodate to follow, while the other cases were softer, thanks to the fact that there was no stage of collision precaution but only following.

In the graphs of Figure 13 it is possible to check how the margin of safety distance between the 125 and 140 pixels between the centers of mass of both mobiles is maintained, with some oscillations due to distance changes that are generated between the vehicles during the instants in which the follower rotates to equalize its angle with that of the leader, or when it detects that they are too close or too far away and tries to make a quick correction. For 3 of the tests performed on the algorithm, the distance between the mobiles was recorded on all three occasions, obtaining the results of Figure 13, with average distances of 133, 136 and 129 pixels.



Mean=132.8 Mean=136.36 Mean=128.94

Figure 13. Safety distance

When the distance between the vehicles abruptly increases or decreases, it means that the follower stopped to correct its angle of inclination with respect to the leader while it continued advancing, causing an increase in distance between both due to the rotation of the follower on its center of mass, without displacement. On the other hand, small changes in distance between one iteration and another are due to short rotations of the follower on its position, or because it is at a limit distance between "Far" and "Close" that causes the conditions of agent advancement to change suddenly.

CONCLUSIONS

The speed changes dependent on the recognition percentage of the "Far" and "Close" categories allowed the follower to maintain a safe distance from an object in movement without colliding in situations of mutual approach. However, the lack of knowledge of the follower's sense of advance made it difficult for them to move around in the workspace, especially

at very short distances from the leader, since advances in the wrong direction during displacement correction generate the possibility of collision before being able to establish the appropriate advance.

On the other hand, the rotations of the follower on its axis give space to the leader to move too far away from it, which causes an increase in distance between both elements without the follower entering the advance condition to correct said inconvenience. This difficulty could be solved with an increase in the speed of rotation or generate rotations during the advance instead of rotating on the center of mass, in such a way that the follower has the ability to adjust its direction as it moves towards the leader and thus ensure all the time the margin of distance.

The developed simulations allow observing an adequate behavior of the algorithm for the fulfillment of the established objective, however, before applying the program in a real application it is necessary to set the speed of the follower wheels, properly locate the camera that captures the entire environment, and manage conditions of controlled light and markers in the form of a circle and triangle of the colors worked for the vehicles, in order to obtain images similar to those used for the CNN training presented in the article. The use of these markers allows the algorithm to be used in applications with different types of differential mobiles without having to re-train another network.

The training of a CNN for the distance control between two mobiles achieved its objective for the developed application, however, the safety distance cannot be established exactly because it depends on the quality of the network and not a mathematical calculation, so it is necessary to establish a database robust enough to ensure a certain distance that meets the requirements of a specific application.

ACKNOWLEDGEMENT

The authors are grateful to the Nueva Granada Military University, which, through its Vice chancellor for research, finances the present project with code IMP-ING-2290 (being in force 2017-2018) and titled "Prototype of robot assistance for surgery", from which the present work is derived.

REFERENCES

[1] PARKER, Lynne E., et al. Current State of the Art in Distributed Autnomous Mobile Robotics. In DARS. 2000. p. 3-14.

[2] CAO, Y. Uny; FUKUNAGA, Alex S.; KAHNG, Andrew. Cooperative mobile robotics: Antecedents and directions. Autonomous robots, 1997, vol. 4, no 1, p. 7-27.

[3] HAN, Woong-Gie; BAEK, Seung-Min; KUC, Tae-Yong. Genetic algorithm based path planning and dynamic obstacle avoidance of mobile robots. In Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation. 1997 IEEE International Conference on. IEEE, 1997. p. 2747-2751. doi: 10.1109/ICSMC.1997.635354

[4] LANKENAU, Axel; ROFER, Thomas. A versatile and safe mobility assistant. IEEE Robotics & Automation Magazine, 2001, vol. 8, no 1, p. 29-37. doi: 10.1109/100.924355

[5] HIDALGO-PANIAGUA, Alejandro, et al. Solving the multi-objective path planning problem in mobile robotics with a firefly-based approach. Soft Computing, 2017, vol. 21, no 4, p. 949-964.

[6] HU, Yanrong; YANG, Simon X. A knowledge based genetic algorithm for path planning of a mobile robot. In Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on. IEEE, 2004. p. 4350-4355. doi: 10.1109/ROBOT.2004.1302402

[7] LUNA, Marco A., et al. Mobile robot with vision based navigation and pedestrian detection. 2017.

[8] EGERSTEDT, Magnus; HU, Xiaoming. A hybrid control approach to action coordination for mobile robots. Automatica, 2002, vol. 38, no 1, p. 125-130.

[9] KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. Imagenet classification with deep convolutional neural networks. En Advances in neural information processing systems. 2012. p. 1097-1105.

[10] ROCCO, Ignacio; ARANDJELOVIĆ, Relja; SIVIC, Josef. Convolutional neural network architecture for geometric matching. arXiv preprint arXiv:1703.05593, 2017.

[11] HALICEK, Martin, et al. Deep convolutional neural networks for classifying head and neck cancer using hyperspectral imaging. Journal of Biomedical Optics, 2017, vol. 22, no 6, p. 060503.

[12] DUDEK, Gregory; JENKIN, Michael. Computational principles of mobile robotics. Cambridge university press, 2010.

[13] MURILLO, Paula C. Useche; ARENAS, Javier O. Pinzón; MORENO, Robinson Jiménez. Implementation of a Data Augmentation Algorithm Validated by Means of the Accuracy of a Convolutional Neural Network. Journal of Engineering and Applied Sciences, 2017, vol. 12, no. 20, p. 5323-5331.