International Journal on Advanced Science Engineering Information Technology

Vol.12 (2022) No. 1
ISSN: 2088-5334

Path Planning for Robotic Training in Virtual Environments using Deep Learning

Javier O. Pinzón-Arenas [a,*], Robinson Jiménez-Moreno [a], Astrid Rubiano [a]

[a] *Department of Mechatronics Engineering, Militar Nueva Granada University, Bogotá D.C, 110111, Colombia*
Corresponding author: [*]u3900231@unimilitar.edu.co

***Abstract*—Reducing costs in the acquisition of industrial robots and their benefit in continuous production workdays have increased the number of investigations that expand the robot's action capabilities. This work proposes a trajectory planning system for a UR3 robot that is very usual in academic, research, and industrial applications. The system is presented based on a convolutional network training for regression tasks focused on learning the desired trajectory. A virtual environment has been developed to simulate different trajectories based on the interaction between UR3 robot and object detection and location through the convolutional network employed. This work exposes the network's training and the results of the transport of the object, where the robot can position itself on the desired tool (scissors and screwdriver), which is recognized by training a Faster network R-CNN and the re-localization of the tool in a conveyor band. For the trained trajectory's, a ResNet-50 model is proposed, and the overall performance achieved was 92.63%, with a mean square error of 24.7 mm in the trained trajectory's repetition. Also, the boxplot of each ax in the trajectory is exposed since they show in a more detailed way the deviation of each of the points in the whole validation set. The average collection time, from when the system takes the workspace capture to its initial positioning after leaving the tool on the belt, was 51.3 seconds, enough for real-time applications.**

***Keywords*— CNN regression; faster R-CNN; path planning; virtual environment.**

Manuscript received 27 Jul. 2020; revised 17 Mar. 2021; accepted 7 Jun. 2021. Date of publication 28 Feb. 2022.

I. Introduction

In recent decades, the use of robots has increased in different fields due to the high versatility they allow and the reduction of their costs, both at the industrial level [1] and the social level [2]. Likewise, the number of investigations carried out in robotics is also increasing, allowing the tasks they can perform to be made more efficient [3] [4]. Among the tasks that a robot can carry out, one of the essential aspects that must be determined is how its displacement will be carried out [5], known as path planning.

Robot path planning requires that it extract the necessary information from the environment, for example, to move free of collisions for mobile robots as presented in Campbell *et al.* [6], and based on points in space for fixed robots, as presented in Hou *et al.* [7]. In general, it is required to know the kinematics of the robot to determine its displacement [8], where the path to follow can be established even using optimization techniques that allow improved movement [9] [10]. A traditional trajectory planning technique is the filling algorithm [11] [12] that has been improved with hybrid models like the one presented in Xizhi *et al.* [13]. Likewise, as discussed in Han and Yu [14] and Kumar *et al.* [15], heuristic models have been implemented as trajectory planning algorithms.

However, artificial intelligence techniques are being used with remarkable success in the planning of trajectories. In Jiménez-Moreno and Brito [16], fuzzy logic algorithms are used to determine the path of the robot's movement, and in Moteir *et al.* [17] and Wang *et al.* [18], the authors employ deep learning techniques based on convolutional neural networks (CNN) [19]. For locational tasks, regressions CNN has shown great robustness [20].

Deep Learning techniques also allow the development of an entire robotic assistance system as set out in Jiménez-Moreno *et al.* [21], where the system recognizes from the user's voice what tool he wants, identifies it, and takes it, handing it over to his hand, the algorithms developed are based on CNN. For the development of object identification, Faster R-CNN networks are one of the most versatile algorithms for their ability to recognize and locate an object [22]. Faster R-CNN has many applications in object recognition; in Li *et al.* [23], they are used for facial recognition of expressions, in Wan and Goudos [24] for fruit detection in robotic vision systems.

Given the great robustness of Deep Learning algorithms, a robotic path planning system based on the location of the desired object is exposed in this work. A Faster R-CNN network is used to identify the object over which the robot must be positioned for its grip, and to establish the trajectory, a CNN architecture for the regression task, is used. A virtual environment is used to simulate the robot trajectory planning task since virtual media has become a support tool for robotics investigations and simulation of tasks in safe environments prior to the implementation of robot operation [25]-[27].

The document is divided into four sections. The first section corresponds to the introduction, where state-of-the-art was exposed within the framework of the proposed developments. Section two presents the methodology used for tool detection and trajectory estimation. The third section exposes the analysis and results of the algorithms exposed in the robotic displacement task, and finally, section four exposes the conclusions reached**.**

II. Materials and Methods

The implemented tool collection system is composed of two stages. The first stage consists of detecting the tools available in the workspace. Once identified, it is verified which is closer to the manipulator, moving on to the second stage. An estimation of the path to be followed by the manipulator is made, from the point where the tool is located to the conveyor belt where it will be left. For each of the stages, two artificial intelligence techniques based on Deep Learning were implemented: a Faster R-CNN [28] and a CNN by regression, respectively. This system is used within a virtual environment previously built, where a UR3 robot and two types of tools (scissors and screwdriver) are used. The development of the system is described.

A. Tool Detection

For the implementation of the tool detection subsystem, a database is created, which consists of 1400 images of size 640x480 pixels in RGB format, taken from the top of the work environment. Each image was manually labeled, locating the tools by means of a bounding box, as shown in Fig. 1. In total, 1100 of the images are used to perform the network training and 300 for the validation of this.

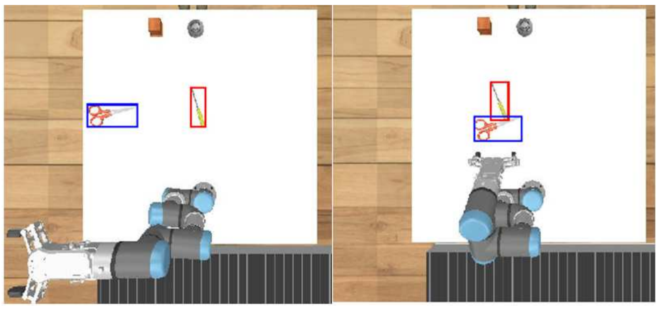


Fig. 1 Samples of the database for tool detection

The model selected for detection was the ResNet-50 [29], which was modified to match the Faster R-CNN architecture, as described in Shaoqing *et al.* [28]. As it is a pre-trained model, transfer learning is done by fine-tuning its weights.

For this, the training parameters shown in Table 1 are established for each learning stage since the model used was trained with a learning rate of 0.1. Also, a large number of training epochs are not required, thanks to the patterns that the model has already learned.

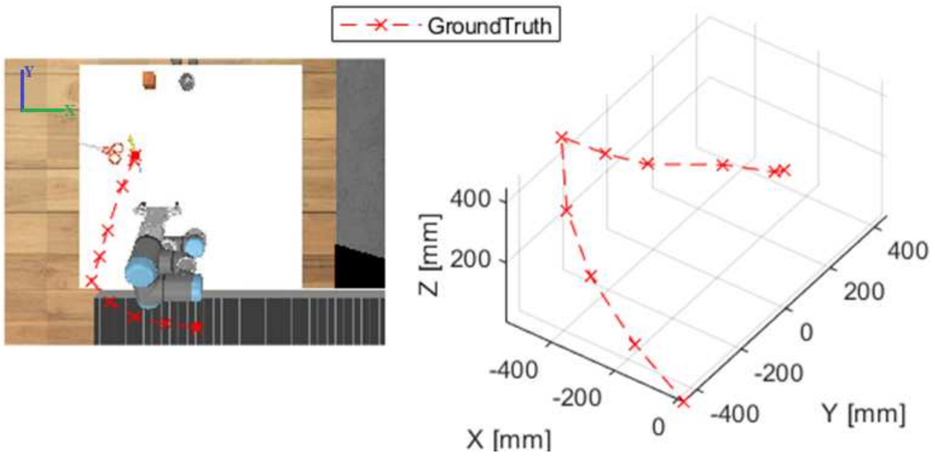
TABLE I
Training Parameters of the Faster R-CNN.

	Learning Rate	**Epochs**
Stage 1	$1x10^{-3}$	10
Stage 2	$1x10^{-3}$	10
Stage 3	$5x10^{-4}$	8
Stage 4	$5x10^{-4}$	8

B. Trajectory Estimation

It was decided to employ a CNN whose output is done using regression for the implementation of the second stage of the system. It will not classify the image but will produce a continuous output that depends on what is seen in the working environment. The output in this situation refers to the path that the UR3's final effector must take from the tool's location to the conveyor where it will exit it. Based on this, the database is built, with 400 of the photos used in the detection stage (300 images for training and 100 for validation). The network developed in the first stage is used to tell the network which tool to focus on as a starting point. The distance between each object and the robot base is estimated based on the center of the detection boxes, taking into account the real dimensions of the workspace. Once the closest object has been found, a marker is placed to the image obtained at the center of the detection box (the tool's position in the image) and, in turn, at the end of the path on the conveyor belt.

With this, the manual labeling of each path in all the images is done. For this labeling, three factors are taken into account: first, the initial reference point (0.0.0), in millimeters, is located at the base of the robot. Second, depending on the location of the tool, the robot must take the most appropriate path; for example, if the nearest object is on the right (or with a negative value on the X-axis), the robot must go from the object to the conveyor belt on the right side, similarly if the object is on the left of the robot (positive value on the X-axis), it must follow a path on this same side. Third, the estimated path is made up of 10 points in space, so that in total, the network must estimate 30 values (three values for each point representing its position on the X-, Y- and Z-axes). Fig. 2 shows the two possible paths for the manipulator movement, depending on the location of the nearest tool.



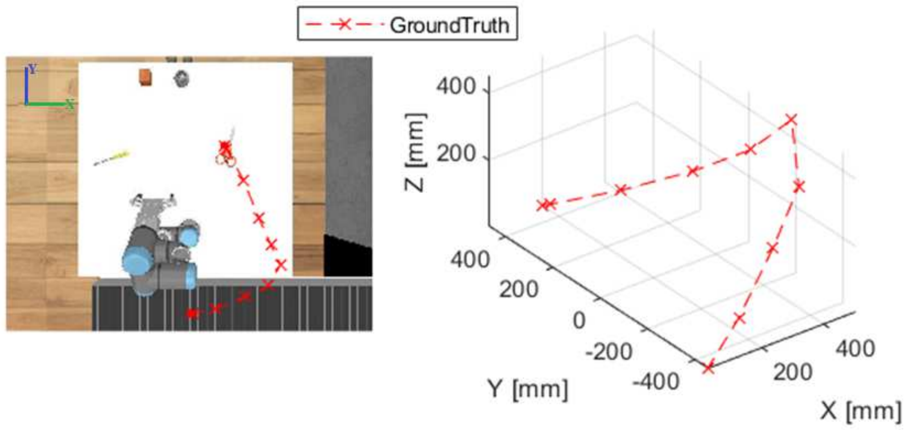


Fig. 2 Samples of paths used for network training.

C. *Training*

For the neural network to be used, the ResNet-50 model is proposed. Since based on tests made to three different models for trajectory estimation in a 2D plane [30], this obtained the best performance. For its training, the following training parameters are set, initial learning rate of 10-4, with a reduction factor of 0.5 every 40 periods, 250 training epochs, and a mini-batch size of 10 images per iteration. With this, the network is trained, obtaining the results shown in Fig. 3.

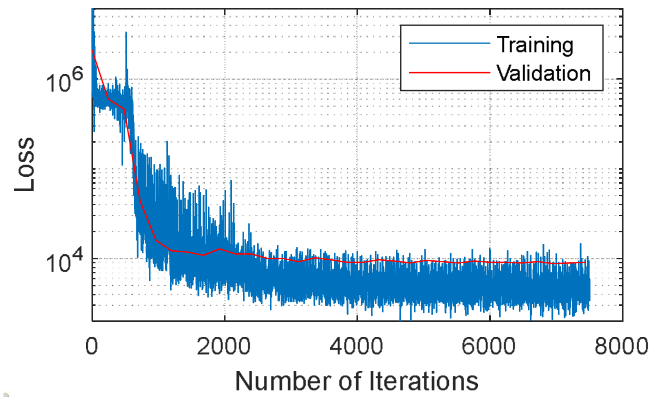


Fig. 3 CNN regression's behavior during training (Loss).

As it is possible to observe, because the model used specializes mainly in classification when used for regression, it causes significant losses in its first iterations since the root mean square error (RMSE) of the estimation of each of the points of the trajectory exceeds 1000 mm. However, the network improves its learning around iteration 600, stabilizing over iteration 3000, reaching an RMSE of less than 150 mm (Fig. 4).

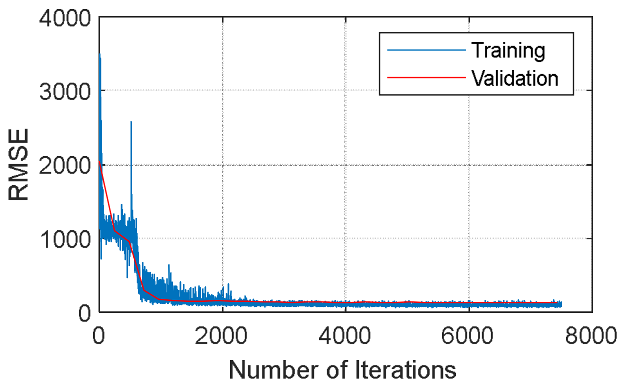


Fig. 4 CNN regression's behavior during training (RMSE)

III. Results and Discussions

To verify the performance of the detection network in a more detailed way, the precision vs. recall plot is made, which shows the robustness of the network by locating the bounding boxes on each of the objects, using the validation set. Fig. 5 shows this plot, where it is possible to see that the network was able to detect the two tools, with an average accuracy above 97%. This is reflected in the example in Fig. 6a, whose detection boxes were remarkably close to the established ground-truth. On the other hand, the accuracy drop in the screwdriver was due to situations where the scissors were occluding it entirely, causing the network to find only the scissors. However, when the screwdriver is placed over the scissors, as in Fig. 6b, the network was able to detect it.

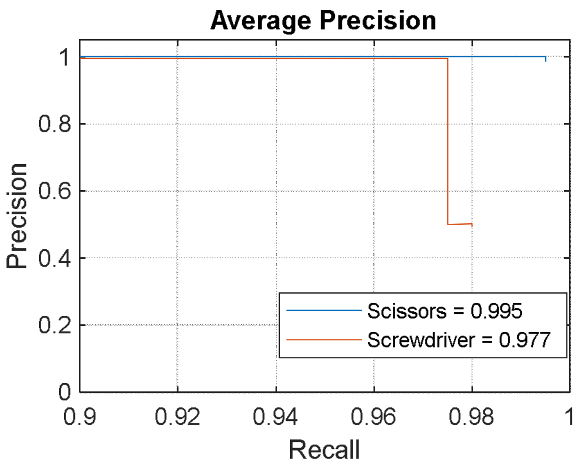


Fig. 5 Precision vs recall plot of the network for tool detection

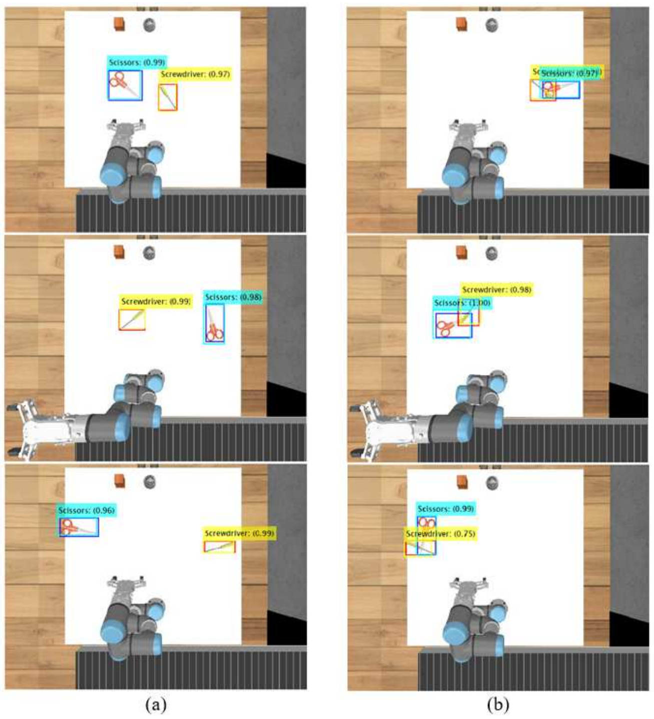


Fig. 6 Examples of detection on the validation set

For the validation of the CNN regression in charge of estimating the paths to be followed, the accuracy of each of the points estimated in each axis is verified, i.e., how much deviation they had with respect to the proposed trajectory. For this, it is proposed to use a threshold, within which, if the point exceeds its distance above it, then it will be given as an erroneous estimate. Likewise, the RMSE is used to verify the total deviation on each axis and the average of the entire trajectory.

Because not all points necessarily need to be strictly subject to the proposed path, different thresholds are chosen for each section of the path. For example, the grip and the final position points of the tool must have a small deviation, as if a point is estimated far from these, then the grip would not be made, and the tool would not be correctly located on the belt. On the other hand, the intermediate points of the path can have a higher degree of deviation without affecting it, however, they must be within the robot's range, to avoid singularities. Bearing this in mind, it is proposed to establish three threshold distances in each path section, as can be seen in Table 2, although for the Z-axis, the threshold used is constant, of 20 mm. With this, it is obtained that the two points with the greatest deviation were point 1 (located on the tool), 2 and 8 on the X-axis, with average accuracies of less than 70%. In contrast to the X-axis, in the Y- and Z-axis, the accuracy results with respect to the thresholds were above 90%, with the exception of point 4 on the Y-axis.

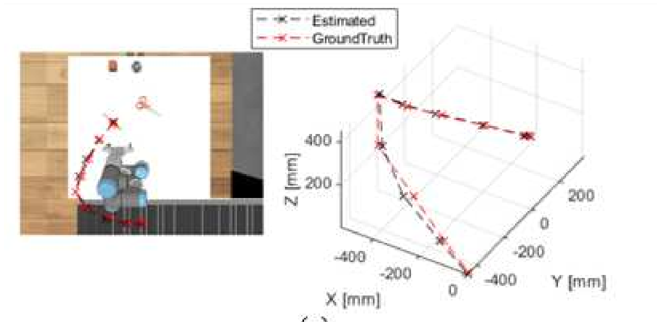
TABLE II
ACCURACY OF EACH POINT, AVERAGE RMSE AND OVERALL PERFORMANCE OF THE NEURAL NETWORK

Point	X-axis	Y-axis	Z-axis	Threshold X-Y [mm]
1	61%	98%	100%	20
2	58%	94%	100%	
3	90%	91%	100%	50
4	85%	76%	100%	
5	94%	97%	100%	70
6	93%	100%	100%	
7	89%	100%	100%	
8	68%	99%	100%	50
9	86%	100%	100%	
10	100%	100%	100%	20
Average accuracy	82.4%	95.5%	100%	
RMSE [mm]	35.77	23.31	2.66	
Total Network RMSE [mm]		24.7		
Network Accuracy		92.63%		

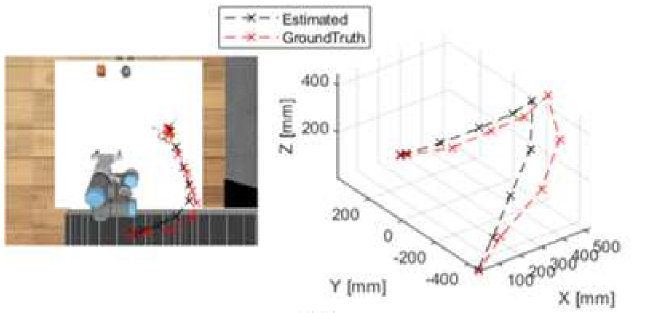
In general, the network obtained an overall accuracy of 92.63% in estimating the points on the trajectory, even having a relatively small RMSE of 24.7 mm, which is very close to the 20 mm threshold set for the points requiring the highest accuracy. The axis with the greatest deviation was the X, with 35.77 mm.

To understand these results more clearly, some examples of estimated paths compared to the proposals are shown in Fig. 7. Fig. 7a represents a very precise trajectory, which follows very closely the trajectory that was proposed. Another correct path, although it does not fully follow the one proposed, is presented in Fig. 7b, where in the first stage of the estimation the network locates the initial and final point accurately, while in points 6 to 9, a shorter path is generated, but within the limits of the established threshold and robot's workspace.

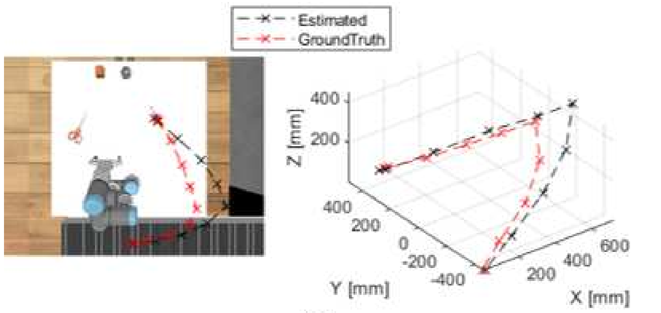
On the other hand, there is the path estimated in Fig. 7c, in which, although the initial and final points were correct, the intermediate points are very far from the X-axis, reaching distances of more than 200 mm at point 6 with respect to the proposed path, even taking them to areas out of the robot's workspace. Fig. 7d presents another case of erroneous estimation, mainly because of the initial points, since these were located far from the grip point that would cause the robot not to collect the tool. These last two examples show possible reasons why on the X-axis, the RMSE was higher than on the other axes.



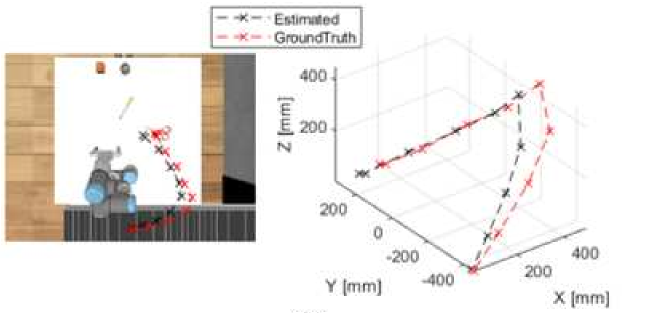
(a)



(b)



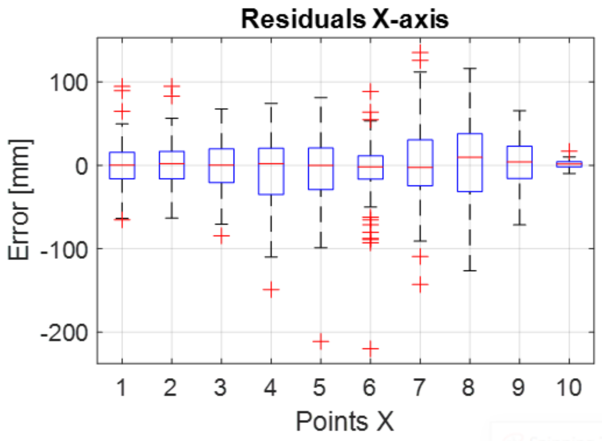
(c)



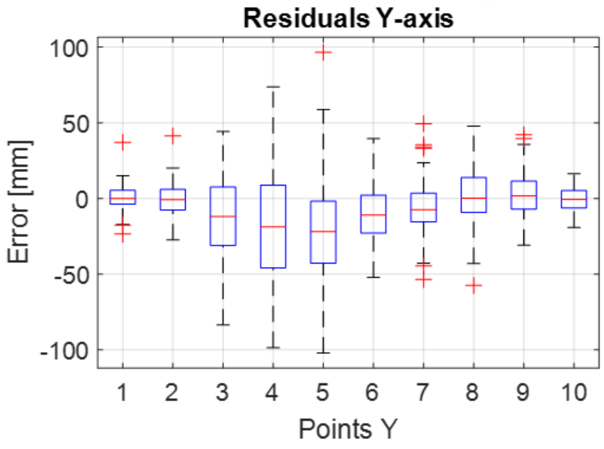
(d)

Fig. 7 Trajectory estimation generated by the CNN regression

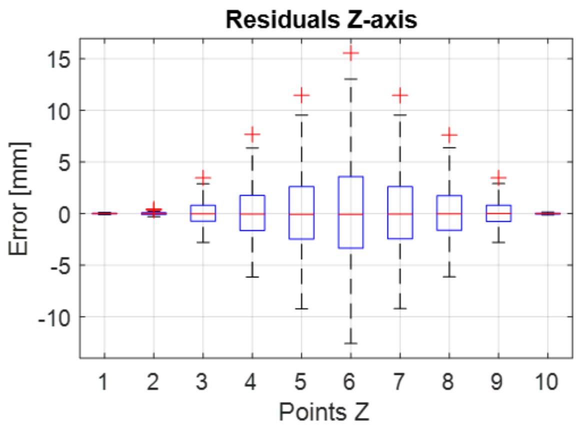
In the same way, the boxplot of each of the axes is presented in Fig. 8 since they show the deviation of each of the points in the whole validation set in a more detailed way. It should be noted that the median and even the ranges obtained at all points remained within the thresholds established for each section of the path, with the median of the furthest error at point 5 on the Y-axis having a value of 21 mm. Although it was the least accurate axis on the X-axis, the medians remained close to 0; however, it is on this axis where there are more outliers, reaching maximum values of up to 219 mm of error in point 6. Also, in most cases on this axis, the minimum and maximum values (quartiles 1 and 4, respectively) remained above the threshold of each point. On the other hand, on the Y-axis, these values only exceeded the threshold in points 3, 4, and 5. As for the Z-axis, it was the one that obtained the smallest deviation in the estimation of its values, with atypical errors of less than 20 mm.



(a)



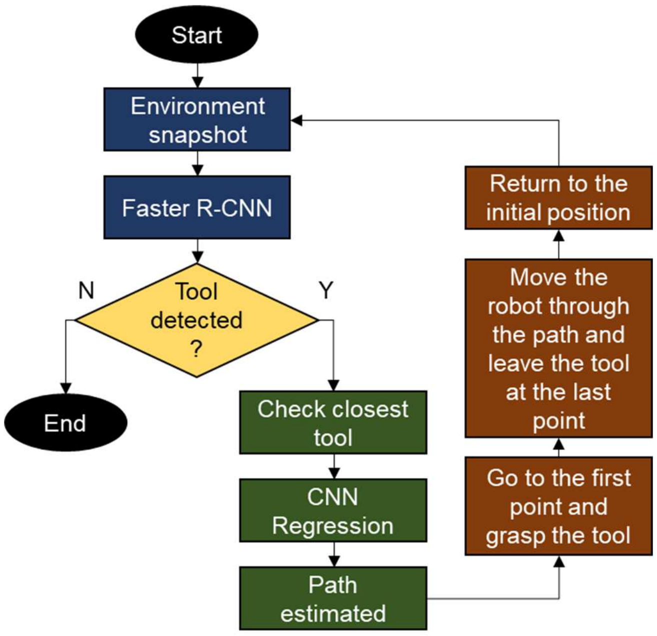
(b)



(c)

Fig. 8 Boxplot for the points on (a) the X-axis, (b) the Y-axis, and (c) the Z-axis

Once the two neural networks to be used have been validated, their coupling is carried out within the virtual environment so that the robot collects the two tools within the workspace. The whole system follows the logic of the flowchart shown in Fig. 9, where first, a capture of the environment is taken. This image is entered into the Faster R-CNN, which performs tool detection. If there are tools in the environment, the system will calculate the distances of the tools from the robot base to verify which one is closer. The closest one is tagged, and the image captured with the tag is entered into CNN regression. This network estimates the trajectory the robot should follow. The first step that is indicated to the robot is to go to the first point of the path, which is where the tool is located, so that the tool grip is performed. Once the end-effector gripper is closed, the robot follows the estimated path until it reaches the last point, which is the location of the conveyor belt. In this part, the robot releases the tool and then returns to its initial position. Finally, the work environment is captured again. If there are no more tools available, the program ends.



Start
Environment snapshot
Faster R-CNN
Tool detected ?
N
Y
End
Check closest tool
CNN Regression
Path estimated
Go to the first point and grasp the tool
Move the robot through the path and leave the tool at the last point
Return to the initial position

Fig. 9 System flowchart

An example of this can be seen in Fig. 10, where the robot collects both the screwdriver (left) and the scissors (right), in three frames sequences for each. The image shows the trajectory made in the environment during the tests, drawn by a green line. In the screwdriver collection test, the start of the trajectory can be observed, where the robot is placed in its initial position, descends to where the tool was, and then moves it to the belt. In the second case, part of the trajectory of the previous grip is shown, evidencing that the robot actually passed through the belt to leave the tool, then to its initial position, to finally pick up the scissors in order to leave them on the conveyor. The estimated trajectory that the robot is performing to collect the scissors can be seen in Fig. 11.

In total, 20 repetitions were performed, randomly placing both tools in position and in angle. The results obtained from the operation of the total system are shown in Table 3. The tool that more times were able to collect the robot was the screwdriver, making a mistake only twice, in which the estimated grip point was located very close to the edge of it, making the robot did not make a good grip and let the tool fall down.

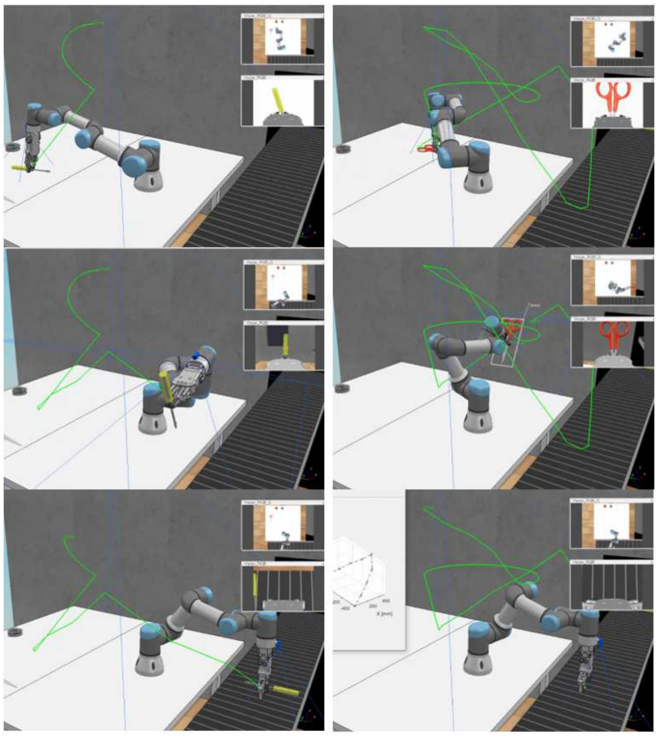


Fig. 10 Tool collection tests and the trace of the trajectory made by the robot

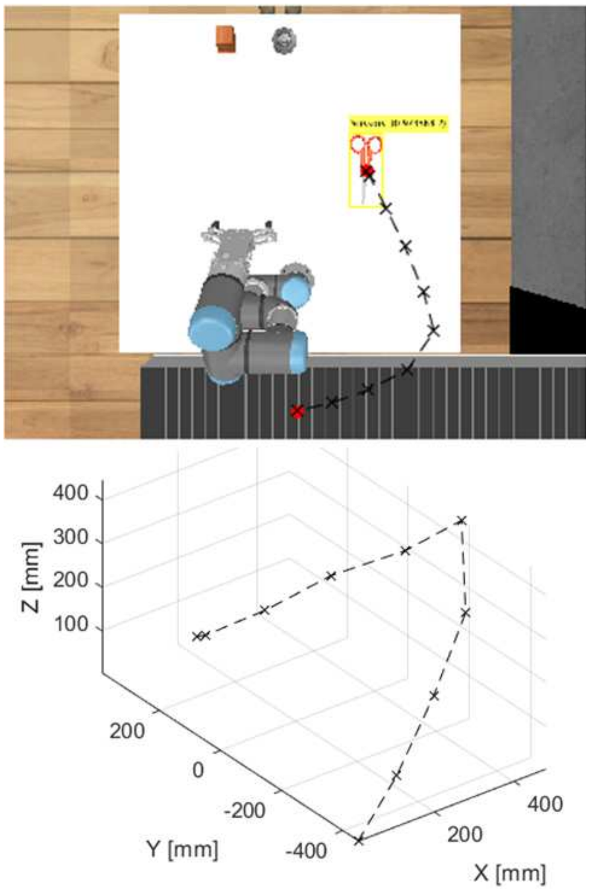


Fig. 11 Estimated trajectory during the test for collecting the scissors

On the other hand, the successful attempts of collecting in the scissors were only 35% (7 correct), due to the difficulty of the grip of these. Since these are thin and most of the time, their grip point was estimated in the area of the blades, making the robot to release it at the beginning. It also failed due to the label generated based on the bounding box, since this was outside the scissors and. Therefore, the network located the initial point away from the tool. The average collection time, from when the system takes the workspace capture to its initial positioning after leaving the tool on the belt, was 51.3 seconds.

TABLE III
RESULTS OF THE TESTS PERFORMED

	Screwdriver collection	**Scissors Collection**
Correct/Repetitions	18/20	7/20
Success rate	90%	35%
Average time	51.3 s	

IV. CONCLUSION

In the present work, a tool collection system was implemented, using Deep Learning techniques for the detection of these tools and subsequent trajectory to follow for the collection and location of them. One of the techniques used was a Faster R-CNN, which obtained a performance in the detection of the two required tools higher than 97% of average accuracy, even managing to detect the tools when they overlapped one over the other. The other technique used was a CNN regression, which was responsible for estimating the path that the robot should follow in order to collect the tools and place them on a conveyor belt. This last network achieved an accuracy of 92.63% in the correct estimation of the 10 established points, even making trajectories obeying the rule of their location. If the tool was located on the right side of the robot, the trajectory had to be made on that side, also for the left side.

When coupling the algorithms for the global system, operational and success tests were performed to collect the tools, showing that the system has a high success rate to collect the screwdriver. However, for thin or difficult-to-handle tools, such as scissors, the system tends to fail. This is because the grip point depends only on the starting point of the path, with no estimation of the tool position angle or a better tool grip point. For this reason, it is expected to implement an addition to the system as future work, which is the use of a Deep Learning technique for the estimation of angle and grip point of the tool to be manipulated.

ACKNOWLEDGMENT

The authors are grateful to the Universidad Militar Nueva Granada, which, through its Vice chancellor for research, finances the present project with code IMP-ING-2935 (being in force 2019-2020) and titled " Prototipo robótico flexible para asistencia alimentaria" from which the present work is derived.

REFERENCES

[1] F. Xiaoqing, B. Qun, X. Hongjun, f. Xiaolan, "Diffusion of industrial robotics and inclusive growth: Labour market evidence from cross country data", *Journal of Business Research*, 2020, ISSN 0148-2963, DOI :10.1016/j.jbusres.2020.05.051.

[2] M. Cho, M. Jang and Y. Cho, "Evaluation of Social Robot Intelligence in Terms of Social Interactive Motion," *17th International Conference on Ubiquitous Robots* (UR), Kyoto, Japan, pp. 608-611, 2020, DOI: 10.1109/UR49135.2020.9144920.

[3] P. Galambos, "Cloud, Fog, and Mist Computing: Advanced Robot Applications," in *IEEE Systems, Man, and Cybernetics Magazine*, vol. 6, no. 1, pp. 41-45, Jan. 2020, DOI: 10.1109/MSMC.2018.2881233.

[4] F. Sherwani, M. M. Asad and B. S. K. K. Ibrahim, "Collaborative Robots and Industrial Revolution 4.0 (IR 4.0)," *International Conference on Emerging Trends in Smart Technologies (ICETST)*, Karachi, Pakistan, pp. 1-5, 2020, DOI: 10.1109/ICETST49965.2020.9080724.

[5] T. Cvitanic, V. Nguyen, S. N. Melkote, "Pose optimization in robotic machining using static and dynamic stiffness models", *Robotics and Computer-Integrated Manufacturing*, Volume 66, 2020, 101992, ISSN 0736-5845, DOI: 10.1016/j.rcim.2020.101992.

[6] S. Campbell, N. O'Mahony, A. Carvalho, L. Krpalkova, D. Riordan and J. Walsh, "Path Planning Techniques for Mobile Robots A Review," *6th International Conference on Mechatronics and Robotics Engineering (ICMRE)*, Barcelona, Spain, 2020, pp. 12-16, DOI: 10.1109/ICMRE49073.2020.9065187.

[7] Y. Hou, Y. Liu and F. Wang, "Research on Intelligent Path Planning Based on Transit Point," *Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC)*, Dalian, China, 2020, pp. 419-422, DOI: 10.1109/IPEC49694.2020.9115175.

[8] Y. Zhang, C. Wang, L. Hu and G. Qiu, "Inverse kinematics problem of industrial robot based on PSO-RBFNN," *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC),* Chongqing, China, 2020, pp. 346-350, DOI: 10.1109/ITNEC48623.2020.9085179.

[9] S. Ivanov, L. Ivanova and Z. Meleshkova, "Calculation and Optimization of Industrial Robots Motion," *26th Conference of Open Innovations Association (FRUCT)*, Yaroslavl, Russia, 2020, pp. 115-123, DOI: 10.23919/FRUCT48808.2020.9087376

[10] B. Yang, W. Li, J. Wang, J. Yang, T. Wang and X. Liu, "A Novel Path Planning Algorithm for Warehouse Robots Based on a Two-Dimensional Grid Model," in *IEEE Access*, vol. 8, pp. 80347-80357, 2020, DOI: 10.1109/ACCESS.2020.2991076.

[11] J. Herrera, D. Espitia, R. Jimenez-Moreno, R. Hernández, "Flood Fill Algorithm Dividing Matrices for Robotic Path Planning". *International Journal of Applied Engineering Research* ISSN: 0973-4562,v.13 fasc.11 p.8862 - 8870 ,2018.

[12] J. Herrera, C. Pachon-Suescun, R. Jimenez-Moreno, "Scara Robot Path Planning Through flood fill Algorithm". *International Journal of Applied Engineering Research* ISSN: 0973-4562, v.13 fasc.19 p.14273 - 14281 ,2018.

[13] H. Xizhi, J. Zhihui and X. Congcong, "Vehicle Path Planning Fusion Algorithm Based on Road Network," *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Chongqing, China, 2020, pp. 98-102, DOI: 10.1109/ITNEC48623.2020.9084895.

[14] S. D. Han and J. Yu, "DDM: Fast Near-Optimal Multi-Robot Path Planning Using Diversified-Path and Optimal Sub-Problem Solution Database Heuristics," in *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1350-1357, April 2020, DOI: 10.1109/LRA.2020.2967326.

[15] R. Kumar, L. Singh and R. Tiwari, "Comparison of Two Meta – Heuristic Algorithms for Path Planning in Robotics," *2020 International Conference on Contemporary Computing and Applications (IC3A)*, Lucknow, India, 2020, pp. 159-162, DOI: 10.1109/IC3A48958.2020.233289.

[16] R. Jiménez-Moreno and L. Brito, "Planeación de trayectorias para un móvil robótico en un ambiente 3D," *2014 IEEE Biennial Congress of Argentina (ARGENCON)*, Bariloche, 2014, pp. 125-129, DOI: 10.1109/ARGENCON.2014.6868483.

[17] I. G. M. I. Moteir, K. Ismail, F. M. Zawawi and M. M. M. Azhar, "Urban Intelligent Navigator for Drone Using Convolutional Neural Network (CNN*)," 2019 International Conference on Smart Applications, Communications and Networking (SmartNets),* Sharm El Sheik, Egypt, 2019, pp. 1-4, DOI: 10.1109/SmartNets48225.2019.9069781.

[18] J. Wang, W. Chi, C. Li, C. Wang and M. Q. -H. Meng, "Neural RRT*: Learning-Based Optimal Path Planning," in *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 1748-1758, Oct. 2020, DOI: 10.1109/TASE.2020.2976560.

[19] I. Rafegas, M. Vanrell, L. A. Alexandre, G. Arias, “Understanding trained CNNs by indexing neuron selectivity”, *Pattern Recognition Letters*, 2019, ISSN 0167-8655. DOI: 10.1016/j.patrec.2019.10.013.

[20] K. Elawaad, M. Ezzeldin and M. Torki, "DeepCReg: Improving Cellular-based Outdoor Localization using CNN-based Regressors," *2020 IEEE Wireless. Communications and Networking Conference (WCNC)*, Seoul, Korea (South), 2020, pp. 1-6, DOI: 10.1109/WCNC45663.2020.9120714.

[21] R. Jiménez-Moreno, J. Pinzón-Arenas, C. Pachón-Suescún, “Assistant robot through deep learning”, *International Journal of Electrical and Computer Engineering (IJECE)*, Vol 10, No 1: February 2020, p. 1053-1062.

[22] Z. Zhong, L. Sun, Q. Huo, “Improved localization accuracy by LocNet for Faster R-CNN based text detection in natural scene images”, *Pattern Recognition*, Volume 96, 2019, 106986, ISSN 0031-3203, DOI: 10.1016/j.patcog.2019.106986.

[23] J. Li, D. Zhang, J. Zhang, J. Zhang, T. Li, Y. Xia, Q. Yan, L. Xun, “Facial Expression Recognition with Faster R-CNN”, Procedia Computer Science, Volume 107, 2017, Pages 135-140, ISSN 1877-0509, DOI: 10.1016/j.procs.2017.03.069.

[24] S. Wan, S. Goudos, “Faster R-CNN for multi-class fruit detection using a robotic vision system”, *Computer Networks*, Volume 168, 2020, 107036, ISSN 1389-1286, DOI: 10.1016/j.comnet.2019.107036.

[25] W. Xie, X. Fang and S. Wu, "2.5D Navigation Graph and Improved A-Star Algorithm for Path Planning in Ship inside Virtual Environment," *2020 Prognostics and Health Management Conference (PHM-Besançon)*, Besancon, France, 2020, pp. 295-299, DOI: 10.1109/PHM-Besancon49106.2020.00057.

[26] J. Qi, H. Yang and H. Sun, "MOD-RRT*: A Sampling-based algorithm for robot path planning in dynamic environment," *in IEEE Transactions on Industrial Electronics*, ISSN: 1557-9948 June 2020. DOI: 10.1109/TIE.2020.2998740.

[27] G. Bolano, A. Roennau, R. Dillmann and A. Groz, "Virtual Reality for Offline Programming of Robotic Applications with Online Teaching Methods*," 2020 17th International Conference on Ubiquitous Robots (UR)*, Kyoto, Japan, 2020, pp. 625-630, DOI: 10.1109/UR49135.2020.9144806.

[28] R. Shaoqing, et al. “Faster R-CNN: Towards real-time object detection with region proposal networks”. *Advances in neural information processing systems*. p. 91-99. 2015.

[29] H. Kaiming, et al. “Deep residual learning for image recognition”. *Proceedings of the IEEE conference on computer vision and pattern recognition*. p. 770-778. 2016.

[30] J. Pinzón-Arenas, R. Jiménez-Moreno and A. Rubiano. “Comparative approach of CNN regression architectures for robotic manipulator 2D trajectory estimation”. In *Multimedia Conference 2019*.