VOL. 15, NO. 2, JANUARY 2020 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

OBJECT SORTING IN AN EXTENDED WORK AREA USING COLLABORATIVE ROBOTICS AND DAG-CNN

Javier O. Pinzón-Arenas and Robinson Jiménez-Moreno
Faculty of Engineering, Nueva Granada Military University, Bogotá D.C., Colombia
E-Mail: u3900231@unimilitar.edu.co

ABSTRACT

This paper presents the development of an algorithm for grouping and ordering of up to 10 different objects, where segmentation techniques are used for background elimination for the detection process, and a CNN (convolutional neuronal network) type DAG (Directed Acyclic Graph) for the classification of the elements. The algorithm detects and classifies the objects found on a table and controls an anthropomorphic manipulative robot to take each one and deliver it to a second robot, which takes the object and moves it to the grouping zone, where it orders each received element, one next to another, at a constant distance between them. A CNG type DAG with an accuracy of 100% was used, whose input is a rectangular image of 70x35 pixels that contains the object to be classified, and outputs the classification of each element, its location on the table, its dimensions (width and height), and its orientation in degrees, managing to interact and order up to 10 elements, both in simulation and in a physical environment.

Keywords: object recognition; DAG-CNN; image segmentation; sorting algorithm; collaborative robotics.

1. INTRODUCTION

The widespread use of robotics, both in industrial and in everyday environments, has led to the development of multiple automation algorithms that allow the robot to perform tasks of interaction with the environment that surrounds it, and from there make decisions that allow it to move without collisions and completing specific tasks, as explained in [1].

In some applications, it is required to carry out processes of ordering in shelves or boxes according to the requirements of order of the user, as worked in [2], where collaborative filters are based on crowdsourced and mined data is used to establish the order of the elements, where an solution easy to update was achieved, without complex models and with the ability to work in two ordering scenarios, or other cases such as the one developed in [3], where the sorting logic of the robot is adapted to the user's requirements through the learning of preference patterns of order, in order to order objects inside boxes or containers, according to the user's requirements.

On the other hand, in applications like the one shown in [4], a multi-robot system was developed where two manipulators were used to automate the rice harvesting process in Japan, in which each robot was coupled with a modem 760MHz wireless to communicate the position, posture and stage of work in which each one is, and thus avoid collisions.

To achieve tasks such as those previously described, a vision system is required that provides each manipulator with information about the environment that surrounds it and about the elements that must be manipulated to complete the assigned tasks. Some of the most recent methods implemented in robotic vision systems are deep neural networks [5,6], some of which are presented below.

The convolutional neural networks (CNN) use a series of convolution filters for the extraction of patterns in images that allow generating a classification [7], where a database consisting of a group of images of each category is used, for the supervised training of said filters. These networks have been implemented in face detection applications in the real environment [8], and in the classification of elements, animals and letters of large databases such as NORB, CIFAR10 and MNIST, with error percentages of 2.53%, 19.51 % and 0.35%, respectively [9].

Additionally, a neural network structured as DAG (Directed Acyclic Graph) has been developed that consists of multiple CNN branches that increase the number of features extracted per image, and improves the classification process, as presented in [10], thanks to its ability to obtain information of multiple sizes. The DAG-CNN has been used in applications as diverse as the classification of animals and objects, developed in [11], or the diagnosis of inflammatory bowel disease (IBD) through a histological examination of tissue imaging, where a structure is required multilayer that allows the detection and extraction of characteristics of different sizes and shapes in the images of muscular and disordered regions, as explained in [12], which is of vital importance for the diagnosis of the disease.

The development of an algorithm of sorting of objects in an extensive work area is presented, where two robotic manipulators are used for the grip and the organization of the elements present in the work area.

The development gives as a result an alternative method to the examples shown previously, for the interaction of two manipulators through a collaborative sorting process, where information on the position and orientation of each robot in the work space is used, to ensure a successful exchange of elements, and of a DAG-CNN for the recognition of the elements in the workspace and organization of them in the grouping zone, considering variables such as their dimensions and the physical restrictions of the manipulators.

This application allows the manipulation of objects in an extensive workspace, where a single robot

does not reach to cover the entire area and requires the support of another.

The present article is divided into 4 main sections: the first one is the introduction, where a brief approach to the collaborative robotics and the algorithms of detection and classification of objects by deep and structured neural networks type DAG is made. The second section describes the operation of the sorting algorithm, from the first recognition of the work area, to the location of the last tool to be sorted within the grouping area, detailing the processes of detection, classification and interaction between the manipulators. Subsequently, in the third section the results obtained are presented and an analysis is carried out on them, to finally raise a series of conclusions, in the fourth section, regarding the work developed and the contingencies presented during the application of the algorithm in a physical environment.

2. METHODS AND MATERIALS

There are 6 subsections in which the operation of the extended sorting algorithm is explained. The first section gives a basic description of the logic and the general operation of the program. In the second section, the area and working conditions under which the functionality tests were performed are presented. In the third section, the initialization of general variables of the program is presented. In the fourth section, it is described the process of detection and classification of tools by means of background segmentation and DAG-CNN, respectively. The fifth section shows the ordering of the information obtained in the previous section and the obtaining of the sorting end points, in order to have all the data ready for the next part of the program, shown in the sixth section, where the logic of the grouping and sorting algorithm is explained.

A. Basic Operation of the Algorithm

The algorithm was designed to sort up to 10 different objects, by using two robot manipulators programmed to work collaboratively. The objective of the program is to move the existing tools in the whole work area to a "grouping zone", in which they are arranged in an orderly manner as shown in Figure-1, where the 6 detected objects were placed, in two rows, an upper line and a lower line. The initial, or disordered, position of each of the objects is shown in Figure-2.

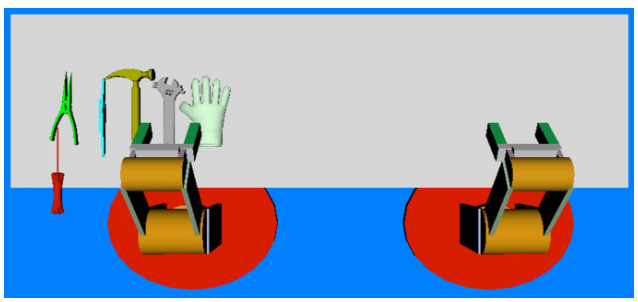

Figure-1. Tools ordered in the grouping zone.

The program is responsible for performing a process of detection and classification of tools by means of segmentation and DAG-CNN, respectively, to know the work space and simulate it in a virtual environment, as shown in Figure-2, where it is emulated each of the objects with their respective positions and orientations.

Once the workspace is known, it is proceeded to evaluate which of the detected tools are within reach of the manipulators, in order to eliminate from the program those that cannot be physically reached by either of the two robots.

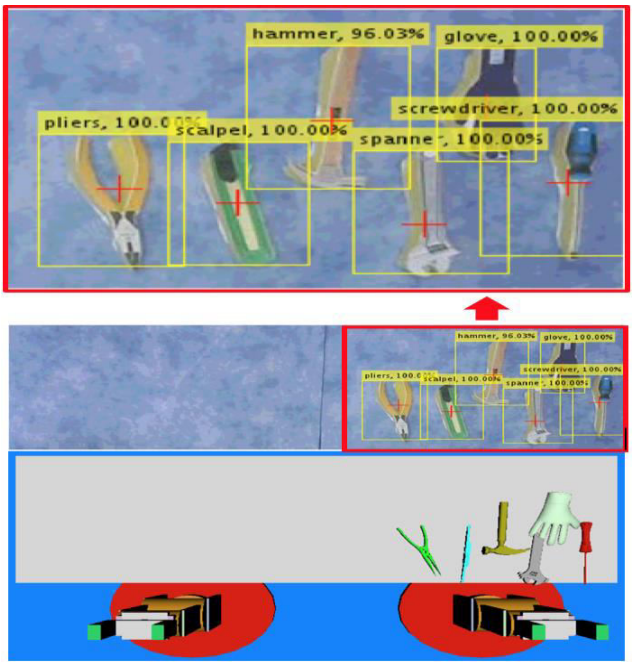

Figure-2. Simulation of the work environment.

The program is responsible for performing a process of detection and classification of tools by means of segmentation and DAG-CNN, respectively, to know the work space and simulate it in a virtual environment, as shown in Figure-2, where it is emulated each of the objects with their respective positions and orientations.

Once the workspace is known, it is proceeded to evaluate which of the detected tools are within reach of the manipulators, in order to eliminate from the program those that cannot be physically reached by either of the two robots.

Subsequently, the tools that can be grasped are separated into two groups, where in the first group are stored the objects that the robot on the right (R1) can take, and in the other those that can be reached by the robot on the left (R2), to establish which of them will make the grip of each object.

During the separation of groups, priority is given to R1, i.e. if the same tool can be taken by either of the two robots, then it will belong to the group of objects taken by R1, in order to increase the probabilities of interaction between the robots and give exclusivity to R2 to focus on the sorting.

Figure-3 shows an example of the elimination of one of the tools that cannot be reached physically by any of the manipulators, a situation in which a message is generated that indicates which tool will not appear in the program due to its location.

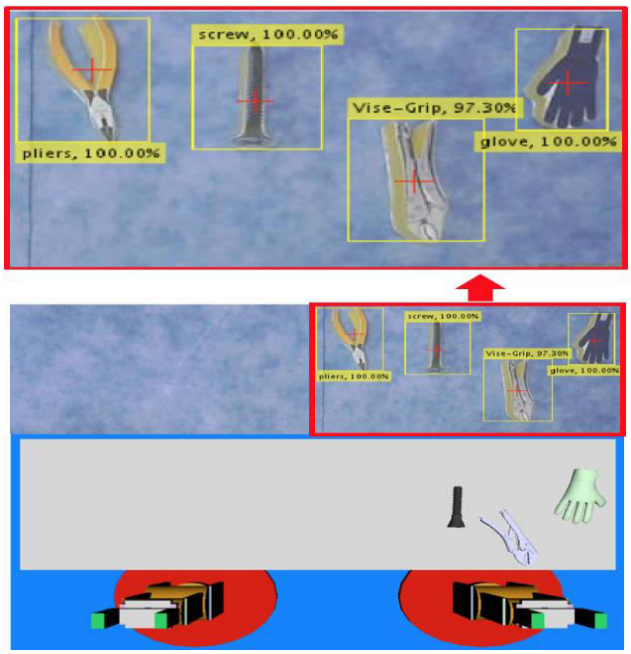

Figure-3. Elimination of physically inaccessible elements ("pliers").

For the example of Figure-3, pliers is too far away from both R1 and R2, so it is not located in the virtual environment and a message is generated that says: "The" pliers "tool is out of range for both manipulators". The program continues sorting the tools that robots can reach.

Depending on the general location of all the elements, a sequence of grip and ordering is established, where the objects that R2 can take are ordered first, and then proceeds to grab the objects that reaches R1 (from left to right for both cases) and take them to a "meeting point" with R2, where it approaches to take the tool that holds R1 and make a gripper to gripper exchange.

Figure-4 shows, in summary, a flow diagram with the sequence and the general operation of the algorithm, from the initialization process of variables to the final ordering of all the elements.

The first step is to initialize variables such as the DAG-CNN to be used, the dimensions of the robots, the dimensions of the work area, the virtual environment, among others.

In the second step, the process of detection and classification of all the objects that are on the table is performed, using a process of segmentation by elimination of background and a DAG-CNN for the classification of up to 10 elements. In this step, the quantity of tools found, the position of each one of them, its orientations and the maximum width of each one will be obtained.

In the third step, all the dimensions are converted to meters and the local coordinates of each tool are calculated with respect to the zero position of each robot. Additionally, all the information obtained in the previous step is grouped and ordered, eliminating the elements that cannot be reached, and the separation of groups is done according to which robot reaches which tool. Subsequently, all the sorting end points are calculated, in which each of the tools will be located in the grouping zone, always ensuring that each position is accessible to R2.

In the fourth step, the extended tools sorting algorithm is performed, which is divided into a series of functions that are executed according to the position of each element to be sorted. Its process consists of using the information ordered in the previous step and the separation of groups established according to the distance of the objects with each robot. In the process, the grip and ordering that R2 can perform without the collaboration of R1 is executed at first, and when finished, R1 is activated to take the missing elements and pass them to R2. Each of the objects is taken in order, from left to right, for any of the cases described, and is ordered at a certain distance between them, and parallel to each other.

The cycle is repeated until all the objects on the table, accessible to the robots, are arranged in the grouping area.

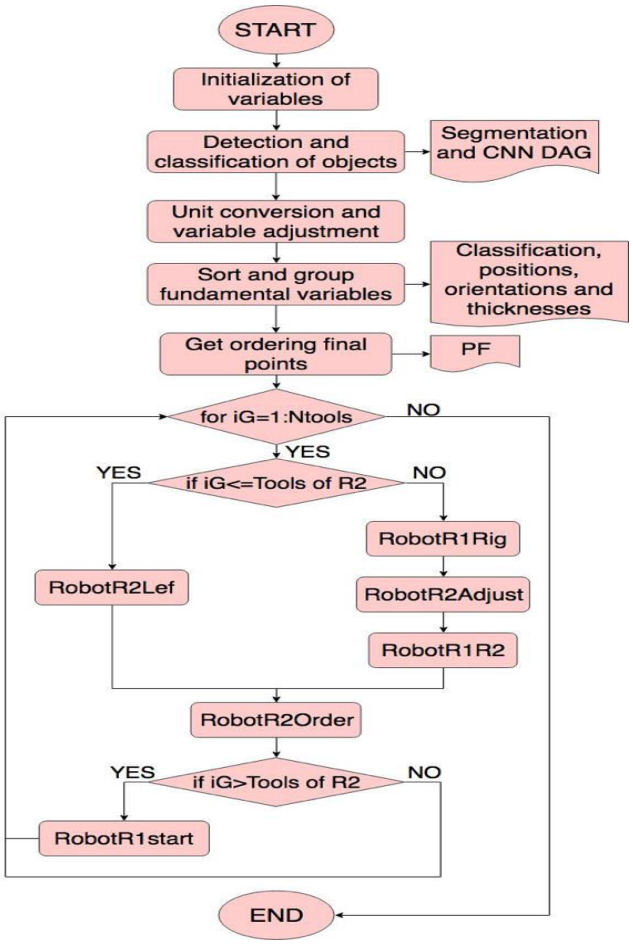

Figure-4. Algorithm flowchart.

B. Workspace

The workspace was organized as shown in Figure-5, where an example of the environment image that captures the global camera and the 10 elements to be ordered are shown, and in the lower part of the table is the location of the two manipulators, both oriented in front of the work area (the Arduino programming card [13] being the front part) in a way that they can rotate up to 90 degrees towards each side.

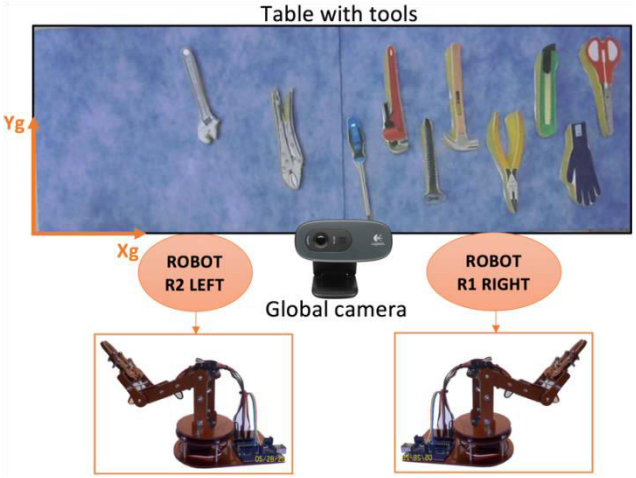

Figure-5. Workspace.

The color of the table was alternated between blue and white, in order to prevent the algorithm from memorizing the background, and the objects were made in foam to reduce the influence of their weight on the dynamics of the manipulators, and facilitate the grip on the allowance of a slight deformation when the gripper closes on them.

C. Step 1: Initialization of Variables

In the program the DAG-CNN network is loaded for the classification of tools, whose architecture is shown in Section 2.4., and variables are initialized, such as the initial position of the robots, the length of each of its links, the dimensions of the image captured by the camera (in pixels), the dimensions of the workspace (in meters), the fixed distance between the ordered tools (in meters), the obtaining of the average color of the work environment (for the subsequent segmentation), and the environment simulated in VRML. Table-1 shows the initialization of the mentioned variables.

Table-1. Initialization of variables.

Variable	Value
Image dimensions	640x240 pixels
Physical dimensions	57x17 cm
Distance between tools	cm

D. Step 2: Detection and Classification

Once each of the variables has been initialized, the VRML is loaded as shown in Figure-6, and a continuous video capture of the physical working environment is opened, in which all the objects that are desired to be sorted are located. After each of the objects has been randomly located, the "1" key is pressed to capture the current image and start the detection and classification process.

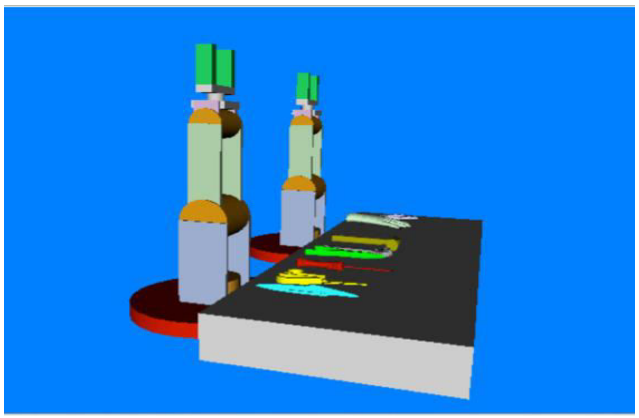

Figure-6. Start of VRML.

For the detection process, a subtraction was made between the average color obtained from the background, and that of the image captured by the camera, in such a way that the presence of each new element represents a change in the tonality of specific areas of the image and makes the difference between these areas higher than a set threshold (40 for the red and green components, 30 for the blue), according to the maximum level of variation of the background against small changes of light (obtained experimentally), thus allowing the detection of objects.

On each area where the difference surpass the threshold, it is established as an element detection and is demarcated with white pixels, as shown in Figure-7, then a detection box is generated on the groups of white pixels higher than 500 pixels (to eliminate noise), and only that portion of the whole image is extracted, to enter it to the DAG-CNN for its classification.

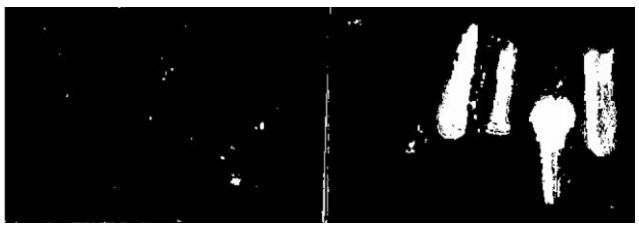

Figure-7. Segmentation process.

A DAG-CNN was trained, with a structure like the one shown in Figure-8, where the architecture of each of its branches is presented in Table-2. The input image was set at 70x35 pixels, since a box of these dimensions allows to cover any of the tools captured by the camera, and to support an inclination of around 15° for any of

them, and its value was established experimentally, locating the tools under the camera and varying the height and width of the box until obtaining the final box.

Where CONV refers to a convolution layer, BATCH to a batch normalization layer, RELU to a rectified linear units Layer, MAXPOOL to a Max Pooling, FC to a Fully Connected layer and DROP to a Dropout layer [7].

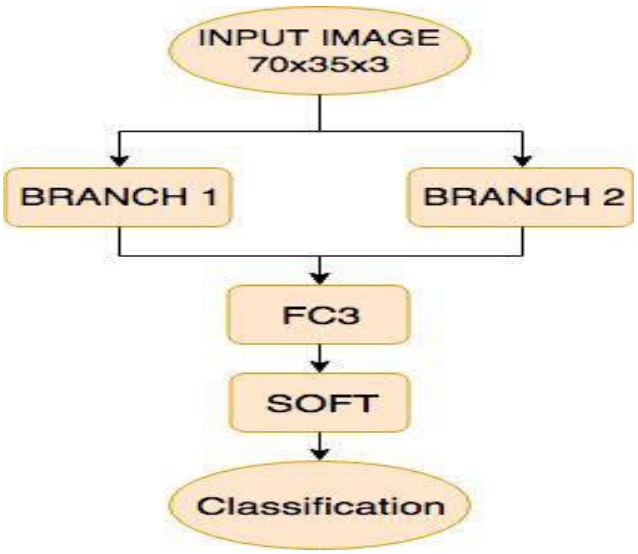

Figure-8. DAG-CNN trained.

Table-2. Architecture of the branches of the DAG-CNN.

LAYERS BRANCH 1	Filter Size HxW	Stride	Number of Filters
CONV+BATCH+RELU	3x3	1	16
CONV+BATCH+RELU	3x3	1	128
CONV+BATCH+RELU	3x3	1	160
MAXPOOL	2x3	2	--
CONV+BATCH+RELU	3x3	1	192
CONV+BATCH+RELU	3x3	1	256
MAXPOOL	2x2	2	--
CONV+BATCH+RELU	3x3	1	512
FC1+RELU+DROP	--	--	--
FC2+RELU+DROP	--	--	--
LAYERS BRANCH 2			
CONV+BATCH+RELU	4x3	1	16
CONV+BATCH+RELU	4x3	1	64
MAXPOOL	2x3	2	--
CONV+BATCH+RELU	4x3	1	128
CONV+BATCH+RELU	4x3	1	256
MAXPOOL	2x3	2	--
CONV+BATCH+RELU	4x3	1	512
FC1+RELU+DROP	--	--	--
FC2+RELU+DROP	--	--	--

The network was trained for 30 epochs with a total of 18, 000 training images (1800 per category) and 4000 test images (400 per category), reaching 100% accuracy as shown in the confusion matrix of Figure-9.a, where the numbers from 1 to 10 correspond to each of the classification categories presented in Figure-9.b.

In Figure-9.b, an example of the database used for training and testing the network is shown, where it is possible to see the use of two different backgrounds, different light intensities, addition of speckle noise and the presence of random shadows, as well as variations in the position and orientation of the elements.

The database was obtained from a total of 550 images per category, which were augmented using the Data Augmentation program developed in [14] that allowed the generation of variations in lighting and the random generation of noise.

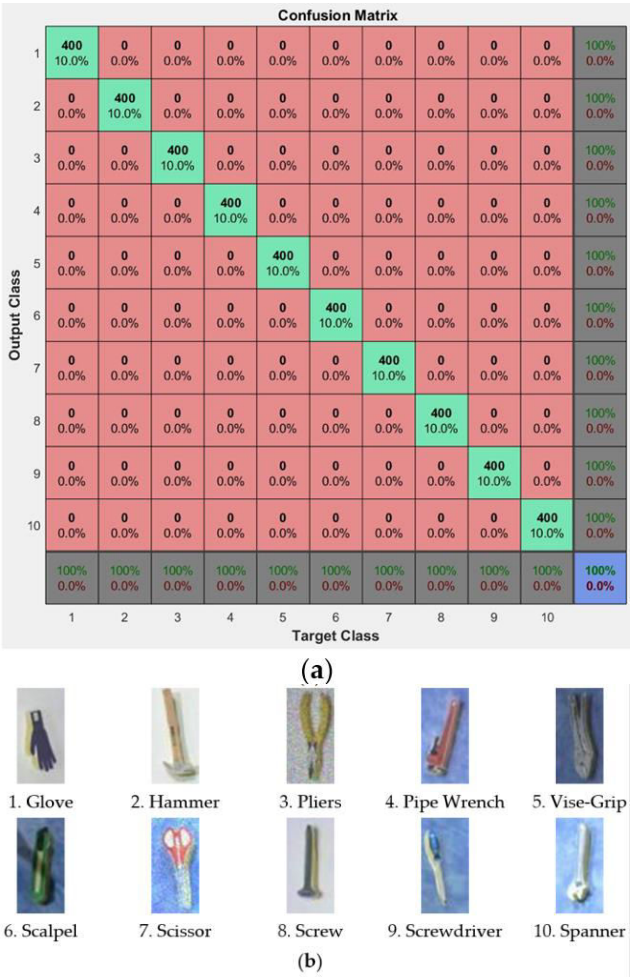

Figure-9. (a) Confusion matrix for the classification of 10 different objects. (b) Classification categories for 10 objects.

On the other hand, for the application it was necessary to adjust said detection box to the proportions of the input image of the DAG-CNN, in such a way that the height/width relation was always maintained, in order to avoid the deformation of the objects by resizing each image to 70x35 pixels. To do this, the proportion of each detection frame generated was calculated, and according to whether it was bigger or smaller than the input image of the network (70/35), pixels were added to one of the two dimensions until obtaining the desired proportion. In case the height/width ratio was higher than expected, the width of the frame was increased, otherwise the length was increased.

In equation (1) the calculation of the width required by the table is shown to achieve the expected ratio ( ), according to the current length of the table, and in (2) the number of pixels that must be added to each side of the box to achieve the expected width ( ), while in equations (3) and (4) the same process is shown but for height adjustment.

$$Adjust1 = \frac{Height}{ratio} \quad (1)$$

$$pxWidth = \frac{Adjust1 - width}{2} \quad (2)$$

$$Adjust2 = width \cdot ratio \quad (3)$$

$$pxHeight = \frac{Adjust2 - Height}{2} \quad (4)$$

Where is the relation between the height and the width of the input image of the network, is the length required by the frame to achieve the ratio, and is the number of pixels that must be added to the height to meet the necessary ratio.

Once the dimensions of the boxes have been adjusted, they are resized to 70x35 pixels, and they are entered into the DAG-CNN for their classification.

The entire process described previously in step 2, outputs the classification of each of the tools found (Classif), its location (Pg), and the dimensions of the detection boxes that cover them (without the adjustment of proportion, Cj is the width). Additionally, the first activation of branch 1 of the network (output image of the first RELU of branch 1) is used to segment the tool from the background and thereby obtain its orientation (Or), using an ellipse as it was done in [15]. This branch was selected due to the quality of its activations, which can be seen in Figure-11 of the results and analysis section.

Finally, this information is delivered to Step 3 of the extended sorting algorithm, to continue with the program.

E. Step 3: Conversion of Units, Grouping and Sorting End Points

In step 3, all the information collected is ordered, the accessible points for the manipulators are determined, those that are not are eliminated, and a matrix containing all the sorting end points is generated.

In the first place, the conversion of pixels to meters is done both for the position of each tool (Pg) and for its thickness (Cj), using the relationships shown in equations (5) and (6), where the first converts all the dimensions, with respect to the Yg-axis (see Figure-5), from pixels to meters, and the second one, those corresponding to the Xg-axis, being and the equivalents, in meters, of the height and width of the image of the workspace, and, its dimensions of height and width in pixels, and , the values to be converted, and the results of the conversion, in meters.

$$disYg = AjjY - \frac{inY \cdot AjjY}{DimY} \quad (5)$$

$$disXg = \frac{inX \cdot AjjX}{DimY} \quad (6)$$

For the case of (5), subtract the total distance of the image by the result of the conversion, in order to invert the values and leave the center of coordinates in the lower part of the image, as shown in Figure-5, given that takes the zero point from the top.

Subsequently, the matrices Pg, Cj, Or and Classif are sorted according to the location of each tool in the workspace, organizing them in ascending order with respect to their coordinates in Xg. Then, the global coordinates (Pg) are converted to the local coordinates for each robot, and the distance between each local point is calculated with its respective center of coordinates, using (7), in order to determine which tools are out of reach, and thus eliminate them from the local matrices, but not necessarily from the simulation.

$$dis = \sqrt{P_x^2 + P_y^2 + P_z^2} \quad (7)$$

Where, and are the coordinates of each point, and the distance calculated.

After eliminating the tools that are out of reach for each manipulator, the resulting matrices are compared in order to find the inaccessible points for both manipulators, and thereby eliminate them completely from the program and the simulation, as shown in Figure-3. Then, the matrices that contain the local results of one robot are compared with those of the other, to look for the tools that are repeated and thus eliminate the copies and leave only one, either because both manipulators can reach the same object (the repeated tool is eliminated in the matrices of R2), or because some error in the classification generated the repeated detection of a tool (in this case the first tool detected is kept, eliminating the others from the matrices).

From these results, it is determined how many tools are going to be ordered, how many of them are grasped only with R2 and how many with R1, and the final matrices that contain all local coordinates, accessible and ordered, are assembled for both manipulators (PVT), the orientations of the tools (OT), their dimensions (CT), the global coordinates (PT) and the classification of each of them (LT). The matrices maintain an ascending order with respect to Xg.

The total number of tools to be ordered is defined as and is calculated with (8), where is the number of tools gripped by R2 and the holds by R1.

$$Ntools = izq + der \quad (8)$$

Finally, CT matrix is used to calculate the sorting end points, making the distance between each tool constant and equal to d, as shown in Figure-10, where is the width of the tool already ordered and the width of the next tool to be sorted.

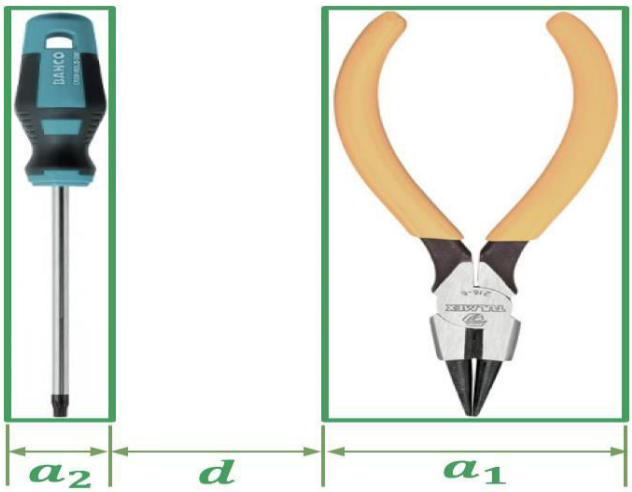

Figure-10. Constant distance between tools.

With the Algorithm 1, (global) sorting points are calculated, using a for cycle from i = 1 to the total number of tools ( ), where, for i = 1, becomes, as shown below.

Algorithm 1: End Sorting Points Calculation	
1.	for $i = 1{:}h2$
2.	if $i == 1$
3.	$EP(1,3) = \frac{CT(1)}{2}$
4.	else
5.	$EP(1,3) = \mathrm{EP(i-1,3)} + \mathrm{d} + \frac{\mathrm{CT(i)}}{2} + \frac{\mathrm{CT(i-1)}}{2}$
6.	end
7.	end

The variable is a matrix of n x 3, where n is the number of tools to be sorted, and columns represent the Y, Z, X coordinates, respectively. The distance X between each point is calculated with Algorithm 1, the height Z is constant and equal to 0.056 m (height of the table that holds the tools), and the distance Y depends on the number of tools to be ordered and its dimensions, which determine if one or two sorting lines are used.

The upper and lower lines were established, each at a distance of 0.17m and 0.08m from R2, respectively. For each line, a maximum of 5 tools was determined, in order to ensure that the horizontal space of the table was sufficient for all the objects, and a conditional was added to allow accessible points only for R2, so that if the 5 tools do not fit into a single line, an immediate change is made to the next one. The line that is filled first is the upper one. All the coordinates of are taken locally with respect to R2, since it is the only manipulator that will order the elements.

F. Step 4: Extended tool sorting algorithm

In step 4, the complete process of ordering elements was programmed, making the movement of the manipulators both in simulation and in a physical environment. In this step, everything previously done is gathered to execute each of the grouping and sorting steps, performing an interaction between the manipulators R1 and R2.

Based on the flow diagram of Figure-4, the first step is to make an initial reading of the number of elements that have been sorted, in order to stop the process when completing. Then, an evaluation is made in which the manipulator will perform the initial movement, depending on and variables.

If there are elements that can be taken only by R2 ( ) the sub-algorithm RobotR2Lef is applied, otherwise or in case all the objects that R2 can take have already been ordered ( ), it is proceeded to execute the sub-algorithms RobotR1Rig, RobotR2Adjust, and RobotR1R2. Each of the aforementioned sub-algorithms is responsible for taking the objects to be sorted according to an established order. Finally, the sub-algorithm RobotR2Order is executed, which is responsible for performing the sorting process.

The last conditional of the diagram of Figure-4 is used only when robot R1 has been moved and allows executing the sub-algorithm RobotR1start, responsible for returning R1 to its initial position and thus restarting the grip process.

Each of the sub-algorithms used are described below.

RobotR2Lef

In this sub-algorithm, R2 is moved from its current position to a position located on the tool to be grasped, then the gripper is moved downwards in a straight line to the desired object, the gripper is closed on it, and it is raised again in a straight line.

RobotR1Rig

In this sub-algorithm R1 is moved from its current position to a position located on the tool to be grasped, then the robot is lowered in a straight line to the desired object, the clamp is closed on it, it is raised again in a straight line, and it moves to the neutral point (PN), where the exchange of objects between the robots takes place.

The neutral point is found in the coordinates, with the positive X axis to the left of R1, positive Y in the direction of Yg, and positive Z, upwards.

RobotR2Adjust

This sub-algorithm is responsible for bringing R2 to the neutral point. To do this, the end effector is moved from its initial position to a point below the neutral point, with coordinates, and finally, it goes to, where it waits for the instruction to perform the exchange of the tool.

RobotR1R2

This sub-algorithm is responsible for performing the exchange of objects between robots. First, the gripper of R2 is closed on the tool, then the gripper of R1 is opened, and finally, the end effector of R2 is lowered to PN2.

RobotR2Order

This sub-algorithm is responsible for moving R2 from PN2 to a point above the respective sorting position to the current iteration ( ), then lowering the gripper to leave the tool in its new position, the empty gripper is raised in a straight line, and finally, R2 is moved to a central local point, (see Figure-1), to restart the process.

RobotR1start

This sub-algorithm is responsible for moving R1 from its current position (PN) to the central local point, finishing an iteration of the cycle with the locations shown in Figure-1. From that position the process is restarted.

The cycle ends when all the tools in the table are arranged in the grouping area, when the program ends.

3. RESULTS AND DISCUSSIONS

In view of the high accuracy percentage of the DAG-CNN, we proceeded to observe the activations of each one of the trained categories, using the objects of Figure-9.b as test elements, in order to evaluate the process of feature extraction facing different backgrounds, changes in lighting, noise and shadows.

In Figure-11, the activations of the two branches of the DAG-CNN are shown for each category, where the intensity of the white color represents the relevance of the feature extracted by the network. The activations for each branch were ordered from left to right, with the image on the left being the output of the corresponding RELU1.

The activations of Figure-11 allow observing the ability of the filters to extract relevant information from each image, such as the contours or the complete shape of the object. In most cases, branch 1 extracts more information than branch 2, including some details such as the Spanner marks on the handle and part of the head, or the handle of the screwdriver, the hammer, and the finger rings of the scissors, extracted with the first CONV+BATCH+RELU, where it is possible to visualize a large amount of details.

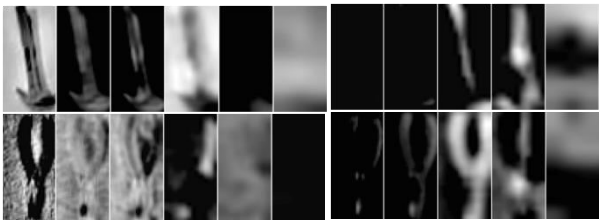

Figure-11. Activations of branches 1 (left) and 2 (right) for Hammer (upper) and Pliers (lower).

On the other hand, for the use of the algorithm in a physical environment it was necessary to adjust the

process of exchange of elements, initially programmed, according to a series of situations not considered during the simulation, such as the inclination of the tool in the gripper according to the point of grip, in such a way that it is possible to avoid false grips or collisions between the gripper or with the element.

Initially, the arrival of R2 to the neutral point was from a height higher than PN and the gripper fell to take the object, however, when performing the physical tests there were difficulties due to the inclination of the tool gripped by R1 and the way it fell R2. This inclination, as well as the angle at which the robots were (shown in Figure-12), caused the element to collide with the R2 gripper while it was accommodating, causing the tool not to be between the fingers of the gripper and, therefore, it was not caught.

Figure-12 shows an example of an upper grip where it can be seen the inclination of the element and the angle in which the robots are. The only way to achieve a successful exchange by this method was that the object was thin in its lower area, as in the case of the glove in Figure-12, otherwise it would generate a collision with the gripper of R2.

Figure-12. R1 and R2 at the neutral point. Upper grip.

Due to the presented situation, the exchange process was changed, and the one presented in previous section was established, where R2 arrives at the neutral point from a lower position. This method avoids collisions with the tool, because the inclination of the same moves away from the gripper, as shown in Figure-13.a, where the handle is above the robots, leaving the tips of pliers free.

In Figure-13, the whole process of exchange and transfer of the tool from the neutral point to the grouping zone is shown.

Figure-14 shows other examples of exchange of objects with elements of other categories. The second difficulty presented during the physical assembly was to sort the tools in the grouping zone, because the tilt of the tool in the gripper of R2, acquired during the exchange process, does not allow any of the flat faces of the tool have direct contact with the table.

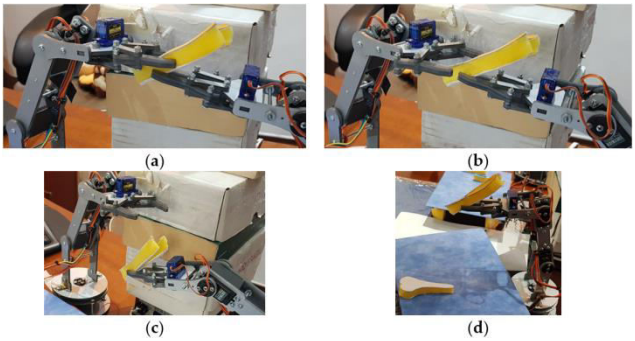

Figure-13. Successful exchange of the Pliers. (a) Gripper of R2 closes. (b) Gripper of R1 opens. (c) R2 moves away from PN. (d) R2 over the grouping area.

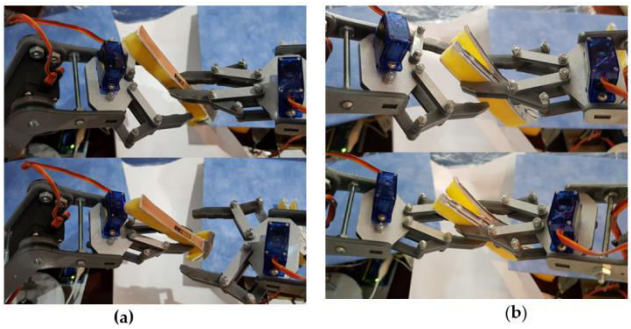

Figure-14. Exchange of tools for (a) Hammer y (b) Vise-Grip.

Additionally, it is not possible to determine the degree of inclination of the tool when taken by R2, nor to ensure that it is always the same for any object, since it depends on variables such as the point where R1 grabs it (in the center, at the top or bottom), the point where R2 grips it during the exchange, and any additional changes due to the robot's own movement.

Due to all these factors, it was not possible to ensure an accurate ordering for each of the objects using 3 DoF (Degrees of Freedom) robots. It is proposed to use robots with two additional degrees in the end effector, one that allows to rotate with respect to Zg to ensure that they are parallel to each other, and another that allows to eliminate the difference of inclination between the flat face of the tool and the table.

However, with the simulation it is possible to verify that the ordering end points are fulfilled, as previously shown in Figure-1 and that with the use of a gripper that allows adjusting the orientation of the tool, it is possible to perform the expected ordering.

Figure-15 shows an example of the extended ordering process, in simulation, from the taking of the tool to sort until the return to the central point of each robot with all the elements grouped together.

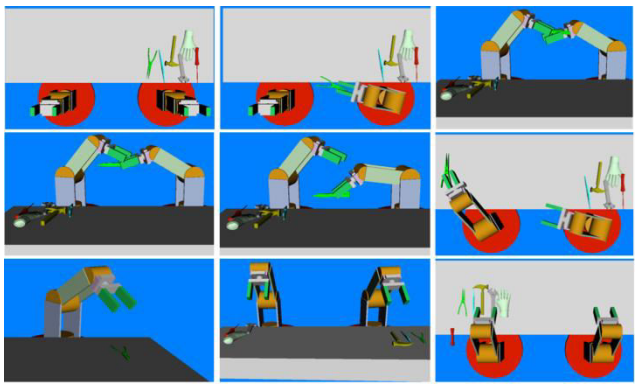

Figure-15. Simulation of the extended sorting process, ordered from left to right, and top to bottom.

In comparison to the physical process, the inclination of the tools was kept constant with respect to the table, allowing ensuring the ordering, and the necessary rotation was made to each element to leave them parallel to each other.

A series of tests of detection and classification of objects was made, with the purpose of observing the behavior of the program facing different working conditions, with variations both of the number of tools on the table, as of its position, and from this, to determine its capacity of recognition in a real environment, where the lighting conditions have been changed with respect to those used during the taking of the database without augmentation.

The situations that generate some type of failure in the detection process are those in which the upper and lower sections of two tools are almost in contact, generating the illusion of being part of a single element, and the second case, it is generated when the screwdriver is completely frontal to the camera, causing the color of its handle to be confused with that of the table. Both situations led to detection reaching 93% accuracy.

On the other hand, in all cases where the tool was detected without any of the aforementioned failures, the DAG-CNN correctly classified it, but when more than one tool per box were found, the classification stopped being accurate, causing an average of 98% accuracy.

4. CONCLUSIONS

The process of exchange of elements between the grippers of two robots requires a high degree of precision at the point of change, and a prior knowledge of all possible positions that the tool can acquire when grasped by the first manipulator, since these will define the grip position that the second robot must acquire to avoid collisions with the element and with the gripper of R1, as well as ensuring that it is reached to hold the element before the other manipulator releases it.

On the other hand, the number of degrees of freedom of the gripper of the physical manipulator restricts the capacity of the manipulator, both to do a better grip in the process of exchange of elements (with the inclination of the object to grasp), and to achieve a physical order closer to the results of the simulation, since it is not possible to ensure a contact area, between the face of the tool and the table large enough to prevent the object from falling, without changing the end point previously established.

Knowing the position of all the tools in the workspace and their distance from the manipulators allowed the application to be executed within an accessible working range, avoiding failures due to grip points too far for any of the robots, and ensuring that each tool was grasped by a single manipulator, in such a way that there was no possibility of collision between them due to a double grip on the same object.

The sequence of grip and order established for the manipulators allows them to work together without invading the space of the other, outside the common area for the exchange process, thanks to the fact that the first ordered tools are the closest to the grouping zone and manipulated mainly by R2.

It was possible to observe how the DAG-CNN achieves a high percentage of accuracy in the classification of 10 objects that differ both in size and shape, reaching to the point of extracting details as small as the spanner handle marks, and others as large as the complete form of the glove, in addition to supporting variations in the input image such as noise, shadows or changes in lighting.

The detection and classification process developed for the program, generated results over 90% accuracy, ensuring an adequate functioning in most working conditions, which allows both the grip points and the dimensions of the tools as their classifications do not affect the execution of the algorithm.

ACKNOWLEDGMENTS

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 (2017-2018) and titled "Prototype of robot assistance for surgery", from which the present work is derived. Likewise, acknowledgement is given to the participation of the engineer Paula Useche who, through the service provision order number 0721/2018, was hired for the development of the exposed algorithms, with the financing of the project.

REFERENCES

[1] Kragic D. 2017. Acting, interacting, collaborative robots. In Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction, Vienna, Austria, 06-09 March, ACM; pp. 293–293. https://doi.org/10.1145/2909824.3020260.

[2] Abdo N.; Stachniss C.; Spinello L.; Burgard W. 2015. Robot, organize my shelves! Tidying up objects by predicting user preferences. In 2015 IEEE international conference on robotics and automation (ICRA), Seattle, WA, USA, 26-30 May, IEEE; pp.

1557-1564. https://doi.org/10.1109/ICRA.2015.7139396.

[3] Abdo N.; Stachniss C.; Spinello L.; Burgard W. 2015. Collaborative filtering for predicting user preferences for organizing objects. arXiv preprint arXiv:1512.06362.

[4] Iida M.; Harada S.; Sasaki R.; Zhang Y.; Asada R.; Suguri M.; Masuda R. 2017. Multi-combine robot system for rice harvesting operation. In 2017 ASABE Annual International Meeting. American Society of Agricultural and Biological Engineers, p. 1. https://doi.org/10.13031/aim.201700321.

[5] Melis M.; Demontis A.; Biggio B.; Brown G.; Fumera G.; Roli F. 2017. Is deep learning safe for robot vision? Adversarial examples against the icub humanoid. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22-29 Oct. IEEE; pp. 751-759. https://doi.org/10.1109/ICCVW.2017.94.

[6] Vernon D. 1991. Machine vision - Automated visual inspection and robot vision. In NASA STI/Recon Technical Report A, Publisher: Prentice Hall, Englewood Cliffs, NJ. Vol. 92.

[7] Krizhevsky A.; Sutskever I.; Hinton G.E. 2012. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems. pp. 1097-1105.

[8] Li H.; Lin Z.; Shen X.; Brandt J.; Hua G. 2015. A convolutional neural network cascade for face detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, Boston, MA, USA, 7-12 June, IEEE; pp. 5325-5334. https://doi.org/10.1109/CVPR.2015.7299170.

[9] Ciresan D.C.; Meier U.; Masci J.; Gambardella L.M.; Schmidhuber J. 2011. Flexible, high performance convolutional neural networks for image classification. In Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, 16-22 July. pp. 1237-1242.

[10] Yang S.; Ramanan D. 2015. Multi-scale recognition with DAG-CNNs. In Proceedings of the IEEE international conference on computer vision, Santiago, Chile, 7-13 Dec. IEEE; pp. 1215-1223. https://doi.org/10.1109/ICCV.2015.144.

[11] Long J.; Shelhamer E.; Darrell T. 2016. Fully convolutional networks for semantic segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 39, 640-651. https://doi.org/10.1109/TPAMI.2016.2572683.

[12] Wang J.; MacKenzie J.D.; Ramachandran R.; Chen D.Z. 2016. A deep learning approach for semantic segmentation in histology tissue images. In International Conference on Medical Image Computing and Computer-Assisted Intervention, 02 October, Springer, Cham; pp. 176-184. https://doi.org/10.1007/978-3-319-46723-8_21

[13] Badamasi Y.A. 2014. The working principle of an Arduino. In 2014 11th International Conference on Electronics, Computer and Computation (ICECCO), Abuja, Nigeria, 29 Sept. -1 Oct. IEEE; p. 1-4. https://doi.org/10.1109/ICECCO.2014.6997578.

[14] Murillo P.C.U.; Arenas J.O.P.; Moreno R.J. 2017. Implementation of a data augmentation algorithm validated by means of the accuracy of a convolutional neural network. J. Eng. Appl. Sci., 12, 5323-5331. http://dx.doi.org/10.3923/jeasci.2017.5323.5331.

[15] Moreno R.J.; Murillo P.C.U.; Beleño R.D.H. 2018. Algorithm for Tool Grasp Detection, Int. Rev. Mech. Eng. 12, 1-8. https://doi.org/10.15866/ireme.v12i1.12513.