Parallel control model for navigation tasks on service robots

Holman Montiel¹, Fernando Martínez², and Fredy Martínez³
Universidad Distrital Francisco José de Caldas, Facultad Tecnológica, Cra. 7 No.40B-53, Bogotá D.C., Colombia
E-mail: ¹hmontiela@udistrital.edu.co, ²fmartinezs@udistrital.edu.co, ³fhmartinezs@udistrital.edu.co

Abstract. Autonomous mobility remains an open research problem in robotics. This is a complex problem that has its characteristics according to the type of task and environment intended for the robot’s activity. Service robotics has in this sense problems that have not been solved satisfactorily. These robots must interact with human beings in environments designed for human beings, which implies that one of the basic sensors for structuring motion control and navigation schemes are those that replicate the human optical sense. In their normal activity, robots are expected to interpret visual information in the environment while following a certain motion policy that allows them to move from one point to another in the environment, consistent with their tasks. A good optical sensing system can be structured around digital cameras, with which it can apply visual identification routines of both the trajectory and its environment. This research proposes a parallel control scheme (with two loops) for the definition of movements of a service robot from images. On the one hand, there is a control loop based on a visual memory strategy using a convolutional neural network. This system contemplates a deep learning model that is trained from images of the environment containing characteristic elements of the navigation environment (various types of obstacles and different cases of free trajectories with and without navigation path). To this first loop is connected in parallel a second loop in charge of defining the specific distances to the obstacles using a stereo vision system. The objective of this parallel loop is to quickly identify the obstacle points in front of the robot from the images using a bacterial interaction model. These two loops form an information feedback motion control framework that quickly analyzes the environment and defines motion strategies from digital images, achieving real-time control driven by visual information. Among the advantages of our scheme are the low processing and memory costs in the robot, and the no need to modify the environment to facilitate the navigation of the robot. The performance of the system is validated by simulation and laboratory experiments.

1. Introduction
Events such as social isolation due to the SARS-CoV-2 virus, coupled with the tendency of society’s long working hours to leave children, the elderly, and people with medical needs at home alone for long periods, bring to the table the growing need for automated systems to provide specialized care, training, and monitoring services in the home. Service robots are one solution to this problem, and the great advances made to date suggest that they will have a strong impact on our society. However, these robots still have many unresolved problems, such as those related to their autonomous navigation in unfamiliar environments, and which have been built for humans [1].
The human being can navigate comfortably in this type of environment relying mainly on the sense of sight. This is the main sensor for navigating in both familiar (home, workplace, etc.) and unfamiliar (shopping mall, bank office, etc.) environments. The signals picked up by the eye are then processed at different levels, in some cases to reason a response, in others to reflexively avoid danger [2]. Accordingly, it is natural to think that a service robot, which will navigate in the same environments, should have similar capabilities to enable it to navigate efficiently in these environments [3]. This principle should provide it with a system with functional performance similar to that of a human, at least in terms of autonomous movement.

Previous experiences constitute a large database that human beings use to recognize new events and elements, comparing them with those already known, and thus defining their interaction with them, it is in this way that previous experiences define their way of navigating in an environment [4]. This can be seen in simple everyday examples, if a person is in a hallway, he can walk down the hallway until he reaches, for example, a door and enters the room, and perform this task while simultaneously dodging obstacles such as pets, boxes or other obstacles in the way. It is also easy for him to identify his destination, and therefore recognize when he reaches it, which is natural when he first enters the bathroom of a hotel or an unfamiliar house. The task is performed by identifying different visual features such as specific shapes, colors, or even combinations of these. Given the similarity of the problem, a robot would also succeed in performing these tasks if it uses a similar control scheme, this scheme is known as visual memory (VM) [5].

The way to implement this structure is similar to the biological model, it is to consolidate over two stages. During the first stage, a training process is carried out using images with characteristic elements of the environment [6]. Some of the images correspond to obstacles from different perspectives, while others correspond to free paths through which the robot can navigate. All these images correspond to labels related to the expected response of the robot. In this way, the learning process corresponds to the creation of an autonomous visual classifier. This process is carried out off-line; the second stage corresponds to an on-line system of propagation and classification of the images by the robot. This stage is much faster and seeks to provide additional support to the first stage.

The functionality of such a parallel control scheme is its high functional performance and reduced computational cost. This scheme is very efficient in structured and static environments, and even with a certain level of dynamics. It has the great advantage of not requiring a complete mapping of the environment, reducing the problem to reactive behavior through Euclidean point reconstruction, which considerably reduces processing requirements, while also reducing response times [7]. Therefore, it can be implemented in parallel on the robot with other interaction functions without a reduction in performance.

This scheme is very consistent with the biological model. Autonomous navigation strategies based on image feedback turn out to have a high performance in real applications. The images contain all the information needed to navigate safely, even in dynamic environments, and do not require prior knowledge of the environment [8]. According to the parameters to be extracted from the images, the processing tends to be fast, allowing real-time operation with very small intervals between processed images (short response time) [9]. These features are ideal for a robot that has to respond simultaneously to several criteria.

One problem, however, of these schemes is in the discontinuity between images generated by the movements of the robot. It is complex to identify the same parameters between two images after the robot’s movement. One solution is the use of specific references in the environment [10], another option is to generate autonomous responses from each image without the need to link the information between different images. For the latter, it is necessary to extract more information from the environment from each image, but the advantage is that the environment is not modified to guarantee the robot’s navigation [11].
In this paper we propose an autonomous navigation scheme for service robots in structured indoor environments along predefined paths in the environment, using images through the VM strategy. The learning of the geometric constraints of the environment is done on a convolutional neural network. To avoid the problem of the discontinuity between images caused by the robot movement, we propose a second control loop in parallel with the VM loop supported by stereoscopic vision and geometric location of relevant points of the obstacles in front of the robot. It is well known that a stereoscopic optical system (with two cameras side by side) outperforms the single-camera [12] system due to its depth perception capability, a key element of our parallel control loop [10, 13–15].

This second control loop will provide specific information related to the distance and position of the objects in the environment, again taking as a reference the human optical system, that is, implementing a visual sensor system formed by two simultaneous images of the same obstacle from the perspective of the robot, but displaced a small distance to define information related to the depth [15]. With the assistance of the system geometry, it is possible to determine precisely the distance of the robot to the obstacle (or a point on the obstacle) from the obstacle distances in the two images, taking into account the focal distance of the cameras [16].

The basic idea is to represent the spatial geometry (x, y, and z) of the environment and the 3D projections of objects in this space (z-dependent distance functions), from the two 2D perspectives (stereoscopic vision) that simulate the human optical system. In most of the schemes reported in the literature, this process is computationally expensive [17] and needs additionally a very good calibration of the cameras, which in real-time applications on an embedded system becomes almost impossible to achieve [18]. Another important element in the strategy is the quality of the lighting in the 2D images, as this has a strong impact on the 3D model [19]. The algorithm is a back-projection from the 2D images (considering the angular distance and the distance between cameras) to obtain the 3D localization of a point in the space [15, 20]. Many applications, such as ours, do not require the complete three-dimensional reconstruction of the environment; it is enough to identify some points of the obstacles, which considerably reduces the computational cost [21]. Again concerning the biological model, we have similar behavior in the human brain when it processes information from the eyes, since it only processes part of the optical information (in which the focus and attention are located). The information collected is then contrasted with the VM loop to define specific parameters of the environment to allow safe navigation [22, 23].

In terms of strategies for estimating distances to obstacles in stereoscopic vision systems, there are two trends [14, 24]. In the first one, there is an active strategy, since the sensors send signals to the obstacles (using a laser or visible light), and their reflection is detected and analyzed [15]. The great advantage of this strategy is the precision and versatility to establish distances, much higher than the human one. However, the system as such is expensive, slow, and complex to implement real applications. Also, a laser system can establish distances to an obstacle point, but sweeps are required to establish the general topology of the obstacle [25]. The second trend is supported by passive methods, which estimate the location of the obstacle by processing the information recorded in two stereoscopic images [21, 26]. Traditionally these images are processed digitally and allow a certain degree of independence concerning the type of cameras required. The classic problems of occlusion and overlap of elements in the images are still solved by parallel strategies [27].

Our research is oriented to the development of service robots, in particular in applications related to the care and monitoring of people (children and elderly). In this sense, the performance of the robot must be characterized by safe interaction with humans, robust in the development of tasks, and high performance at low cost [19, 25]. In this work the last two criteria are of special interest, the robot must perform the navigation along the path identifying obstacles and free paths as it moves, and the processing of the algorithms must be performed on the
robot in real-time. In this sense, the strategy we propose uses a double loop control scheme to guarantee robustness, and the algorithms used allow real-time execution on embedded hardware. Other advantages of our strategy are that there is no need for complete planning and its high performance in any environment [28].

2. Problem statement
Considering the robot environment as one characterized by indoor spaces spatially limited by walls, obstacles (static and dynamic) and doors that can give access (or not, depending on its configuration) to other regions of the environment, our problem is focused on the definition of motion for an autonomous robot in that environment from the analysis in real-time of environmental images captured from the perspective of the robot, with characteristics of low resource consumption and high reliability in the development of the task.

The robot must move along a path drawn in the environment. Geometrically we define as \( \mathcal{W} \) the workspace of the robot, which is an open subset of \( \mathbb{R}^3 \) and is much larger than the robot, so the robot can move without problems in the free space \( E \in \mathcal{W} \). This free space corresponds to the free interior of \( \mathcal{W} \), which is defined as \( \mathcal{W} - O \), where \( O \) corresponds to the set of all obstacles in \( \mathcal{W} \) that represent volumes not accessible to the robot. These obstacles are finite in size and quantity and are visually identifiable by the robot. The walls of the environment that limit and shape \( \mathcal{W} \) correspond to the boundaries of \( \mathcal{W} \) and are denoted as \( \partial \mathcal{W} \).

To identify the path, obstacles, and \( \partial \mathcal{W} \), the robot has two digital cameras that form a stereoscopic vision system. This system is located in \( r(t) \in \mathbb{R}^3 \) and has \( R(t) \in SO(3) \) orientation, where \( SO(3) \) denotes the special orthogonal group of dimension three with respect to a global frame of reference for every instant \( t \geq 0 \).

The estimate of the location of these elements in the environment is made concerning a relative reference frame, whose origin is defined in relation to the digital cameras installed in the robot (i.e. concerning the robot’s field of vision, Figure 1). These cameras are labeled as Left Camera (\( LC \)) and Right Camera (\( RC \)). According to the robot dimensions (the robot used as platform is the ARMOS TurtleBot developed by the research group), the central position of \( LC \) and \( RC \) to the relative reference frame are \((-0.14, 0, 0)\) and \((0.14, 0, 0)\). The horizontal distance between the centres of the two cameras is \( b = 0.14 + 0.14 = 0.28 \) m.

![Figure 1. Reference frame relative to the robot used to estimate the location of points of the obstacles. The figure corresponds to the elevation view and contains the geometrical relations of the experiments.](image-url)
The navigation of the robot requires the robot to recognize the path to follow, and at the same time to recognize the obstacles to avoid along the way. To achieve the robot’s movement a set of movement policies is defined, which determine the behavior of the two robot motors from the visual identification of the path and the obstacles in the environment. The use of two control loops for visual identification is proposed. A first loop capable of identifying characteristics of the environment in front of the robot from the training of known conditions (images with the path, free spaces for navigation and obstructed spaces with different obstacles), and a second loop that determines concrete points of the obstacles (estimated distance to the obstacle) through geometric analysis of the stereoscopic images.

Each of the obstacles in $O$ is indexed by $i \in \mathcal{H} = \{1, 2, 3, ..., n\}$. These are identified by the first control loop, but their exact position is determined by the second control loop. For the classification of images by training, we propose the use of a convolutional neural network, specifically a Dense Convolutional Network (DenseNet). This network is characterized by a dense connection, fewer parameters, and high accuracy compared to other convolutional networks such as ResNet (Residual Neural Network). The design of its structure is based on the concept that the accuracy of the network increases if it contains shorter connections between the layers near the input and the layers near the output, consequently the network concatenates each output of the previous layers forward (Figure 2).

![Figure 2. Densely Connected Convolutional Network (DenseNet) architecture.](image)

The position of the obstacle $O_i$ concerning the global reference frame is given by (equation 1):

$$x_i(t) = R(t)p_i(t) + r(t)$$

(1)

Where $p_i(t)$ is the position of the obstacle to the robot. For an obstacle in the three-dimensional system, this information corresponds to the set of points that belong to the obstacle. The determination of all the points of the obstacle carries a high computational cost, and its calculation is not necessary for our algorithm. Instead, we propose to identify a small group of points of the obstacle that allows us to estimate with precision the distance of the obstacle to
the robot, and therefore to define the movement policy more adapted to follow the established path of navigation.

To search and accelerate the convergence to these points we use a search algorithm based on a simplified model of bacterial interaction developed by the research group. In our search algorithm, we defined a bacterial population (m bacteria in total, without considering reproduction and death processes), which move in the state space conformed by the three-dimensional space $W$ according to local interaction rules. For the search, the bacteria are initially located in $W$, and the movement update is done according to the information provided by the two parallel images of the cameras. If a bacterium is identified on an obstacle, then we can estimate the direct distance of the robot to the obstacle, and consequently, plan the movement of the robot in the environment (Figure 3).

![Figure 3. Densely Connected Convolutional Network (DenseNet) architecture.](image)

The obstacles correspond to locations in $W$ of high interest for bacteria (equivalent to areas with food concentration). Thus, when a bacterium finds an obstacle, it continues to explore the neighboring region to identify other bacteria in it. A region is more attractive to other bacteria if there are several bacteria in it, this process is reinforced by the activation of Quorum Sensing (QS), which means that if the amount of bacteria in a region of $W$ exceeds a quorum threshold $T$, then the performance of the region is increased. In our previous research, we have found that QS reduces convergence time by at least 30%.

Population density is evaluated using a norm to establish the distance between bacteria. For example, a bacterium is defined as $V = (p)$, where $p$ is a point in three-dimensional space, and the distance between bacteria $V_i$ and $V_j$ is defined as $d(V_i, V_j)$. As norm, the Euclidean, Lebesgue or Sobolev distance can be used.

The algorithm to solve the three-dimensional location of the bacteria from the two-dimensional images evaluates the similarity of the neighboring pixels to the bacteria in the two camera images according to the following equation (equation 2):

$$F = \frac{\left| \nabla (M_L) \cdot \nabla (M_R) \right|}{\sum_{color} \sum_{(i,j) \in N} \left[ L(x_{L+i}, y_{L+j}) - R(x_{R+i}, y_{R+j}) \right]^2} + f(QS)$$  

(2)

Where $(x_L, y_L)$ and $(x_R, y_R)$ correspond to the coordinates of the left and right projections.
of the bacterium (Figure 3), $L$ corresponds to the grayscale value of the pixel addressed in both the left and right image, $N$ is the neighborhood around the evaluated bacterium, and $D(M_R)$ is Sobel gradient norm over the left and right projections, which is intended to penalize uniform regions.

3. Methods

Our autonomous navigation system proposal is designed for applications on small service robots in indoor environments with sensorial limitations (no access to GPS and limited WiFi communication) and processing limitations due to the need for real-time operation and the development of multiple tasks simultaneously (in addition to the desire to reduce its total cost). These features have forced us to develop minimalist, high-performance solutions. In particular, the problem addressed corresponds to the development of a navigation scheme that involves large displacements of the robot autonomously in known and structured environments.

As a solution, we propose a motion control scheme based on images by recognizing and tracking landmarks in the environment located in such a way that together they specify a navigation path. As tracking patterns, we have experimented with different types of landmarks on walls and floor of the environment, but in this paper, we document the highest performance scheme, a colored line drawn on the floor of the environment. This pattern is identified by the control unit, as well as the constraints of the environment involving walls, doors, and obstacles. The control scheme is based on feedback of information generated from the geometric characteristics of the landmarks and the geometric restrictions of the environment, in both cases resulting from a stereoscopic vision system (epipolar geometry) consisting of two digital cameras.

The proposed scheme consists of two control loops that take as input parameters the same information, the images of the two cameras of the stereoscopic system (Figure 4). However, neither of the two loops requires decomposition of explicit pose parameters, a feature that is generally used in similar investigations, and which considerably increases the processing demands on the control unit.

The first loop corresponds to a VM system trained with images captured from the actual navigation environment in our laboratory. The images were separated into four categories:

(i) Front of the robot without obstacles and with navigation landmark.
(ii) Front of the robot without obstacles and without navigation landmark.
(iii) Front of the robot with obstacles and navigation landmark.
(iv) Front of the robot with obstacles and without navigation landmark.

Each category was identified with the label corresponding to the number and was made up of 200 images (800 images in total, of which 80% were used for training, i.e. 640 images). The VM system model was built with a DenseNet type convolutional neural network. We chose the DenseNet network because it is more efficient in terms of parameters compared to other topologies such as ResNet, maintaining equivalent classification performance.

The motion control unit analyses the motion projection derived from the two control loops and determines the direction and distance to be covered. It does this according to control policies in correspondence with the results of the two control loops. For example, if the two loops define that there are no obstacles in front of the robot, that the navigation landmark is in front of it, and the distance to obstacles is large, then the advance distance is adjusted and the robot is ordered to advance. If on the other hand there are obstacles or it does not detect the landmark, it must rotate on its axis and look for more information. Before advancing the robot must identify the landmarks near its position and locate an obstacle-free path.

The motion speed is constant in the robot, and its activation on the motors is adapted to the identified path and distances. The images are taken simultaneously and under controlled conditions from the robot, i.e. the robot is kept still and without vibrations (this is verified from the control unit and guaranteed by using an accelerometer). In this way, the quality of the images is guaranteed and discontinuities are avoided.

The second control loop calculates the depth of points on obstacles using stereoscopic vision. In short, stereoscopic vision produces a geometry called epipolar geometry, characterized by two two-dimensional projections corresponding to a single three-dimensional environment. Recently we have developed a work related to the 3D reconstruction of obstacles in front of a robot, for the fast and precise calculation of distances from the robot to the obstacle through stereographic optical sensors. This system allows us to predict with high reliability the structure of an obstacle from two parallel images that provide depth information. We build on previous work that provides algorithms for the reconstruction of stereoscopic data from 2D digital camera images and put those same algorithms into a scalable, low-cost computational framework. Our approach is more efficient than most, because it only identifies a limited set of points of the obstacles, offering a partial reconstruction of the 3D vision in real-time.

4. Results

The VM-based control loop was implemented using a convolutional neural network. This solution was chosen due to its high capacity to differentiate characteristics of the environment from images and its easy propagation on the robot. The training was done off-line using real images from the laboratory.

In particular, we used a DenseNet network of 121 layers deep, these layers were organized in a $5 + (6 + 12 + 24 + 16) \times 2 = 121$ structure, where 5 is (conv, pooling) + 3 transition layers + classification layer. Multiplication by 2 is done because each dense block has two layers ($1 \times 1$ conv and $3 \times 3$ conv). The number of nodes in the input layer is determined by the shape of the images. We scaled all input images to $255 \times 255$ pixels and processed them in RGB, so the shape of the input layer was $255 \times 255 \times 3$. The number of nodes in the final output layer is the number of possible class labels, in this case, the output layer had four nodes. With this structure, the neural network required 6,957,956 trainable parameters and 83,648 non-trainable parameters.

As loss function, we use categorical cross-entropy, as optimizer we use stochastic gradient descent, and as metrics to evaluate the performance of the neural model we calculate the
accuracy, recall, f1-score, and support (Figure 5 and table 1). Also to evaluate the model performance, we generate the confusion matrix and ROC curve of the trained network (Figure 6, 7, and 8). The training was carried out during 10 epochs, and it was verified that there was no over-adjustment.

![Figure 5. Model behavior based on training and validation data.](image)

Table 1. Model metrics.

|   | Precision | Recall | F1-score | Support |
|---|-----------|--------|----------|---------|
| 0 | 0.98      | 0.92   | 0.95     | 48      |
| 1 | 0.92      | 0.98   | 0.95     | 45      |
| 2 | 1.00      | 1.00   | 1.00     | 32      |
| 3 | 1.00      | 1.00   | 1.00     | 35      |
| Macro avg. | 0.97 | 0.97 | 0.97 | 160 |
| Weighted avg. | 0.97 | 0.97 | 0.97 | 160 |

The second control loop, based on bacterial interaction, processes the images in parallel and uses two images captured simultaneously by the two cameras (stereoscopic system), which are 0.5 m from the ground. These images are scaled in this case to 800 × 600 pixels. The projection of the bacteria $i$ on the images is determined by equations 3 and 4.

Left image
\[
\begin{align*}
x_p &= 400 + \frac{(x_i + 0.14)800}{2z_i \tan(35^\circ)} \\
y_p &= 300 - \frac{(y_i)600}{2z_i \tan(30^\circ)}
\end{align*}
\]

Right image
\[
\begin{align*}
x_p &= 400 + \frac{(x_i - 0.14)800}{2z_i \tan(35^\circ)} \\
y_p &= 300 - \frac{(y_i)600}{2z_i \tan(30^\circ)}
\end{align*}
\]

The bacterial population (with a size of 50 agents) is initialized with random and known position distributed in the three-dimensional space able to be detected by the cameras of the robot, this is denoted by the dotted line in Figure 1. The coordinates of the bacterium $i$ is given by the vector $(x_i, y_i, z_i)$. Replacing these values in equations 3 and 4 gives the projections
in the two-dimensional images \((x_p, y_p)\). The performance of the region neighboring bacterium \(i\) is evaluated with equation 2. If the bacterium is on the surface of an obstacle, then the neighboring pixels will be similar in both images, and the function assigns a higher value to the position of the bacterium. According to the performance value, the bacterium \(i\) navigate the state space, and after some cycles, it groups with the other bacteria on the surface of the obstacle.

The navigation strategy was evaluated on our ARMOS TurtleBot robotic platform. The VM loop has 97% accuracy in correctly identifying the environment according to the four labeled cases. On the other hand, the epipolar analysis loop with a bacterial population of 50 agents can reconstruct in great part the most outstanding elements of the obstacles according to the conditions categorized by the first loop (Figure 9). It was observed in the experiments that factors such as lighting conditions affect the performance of the strategy, however, it is an effect that also occurs in humans.

The code used to control the robot’s movements, as well as the acquisition and processing of the images was written in Python [29]. The entire system ran on the robot’s control unit, which was implemented on a DragonBoard 410c Development Board from Arrow Electronics, which has a Qualcomm APQ8016e 64-bit processor.

5. Conclusions
Given the need for an autonomous reactive system for our service robot, we propose a parallel control scheme based on stereoscopic vision able to operate in real-time on our platform with
low resource consumption. The task to be developed by the robot involves mainly autonomous navigation of long distances supported by the identification of landmarks and obstacles in the environment. Also, a low computational cost and robustness as a guarantee in the development of the task are sought. To solve the problem we implemented a control scheme based on information conformed by two parallel control loops. Our strategy is based on a loop that propagates a convolutional network and another loop that implements our bio-inspired algorithm for three-dimensional obstacle reconstruction. The two loops operate separately, but take as their information the same images captured by two cameras on the front of the robot. The neural network identifies environmental and landmark conditions, and the stereoscopic system estimates the distance to obstacles in front of the robot. This information is processed with a set of motion policies that determine the final movement of the robot. The resulting motion control scheme have several advantages over other methods that directly control the entire nonlinear system. The two control loops demonstrate accuracy ranges above 90%, allowing the robot to closely follow landmarks and avoid obstacles in 98% of cases. The results shown correspond to laboratory tests developed on our ARMON TurtleBot robot. These tests were structured in an environment equivalent to that expected in domestic tasks, and the results fully validated the proposed strategy.

Acknowledgments
The authors wish to acknowledge and thank the Technological Faculty of the Universidad Distrital Francisco José de Caldas for the support and funding during the course of this research work.

References
[1] Decker M, Fischer M and Ott I 2017 Robotics and Autonomous Systems 87 348–354 ISSN 0921-8890
[2] Winter S, Tomko M, Vasardani M, Richter K, Khoshelham K and Kalantari M 2019 ACM Computing Surveys 52 1–24 ISSN 0360-0300
[3] Castañeda J and Salguero Y 2017 Tekhnē 14 73–86 ISSN 1692-8407
[4] Macik M 2018 Acta Polytechnica Hungarica 15 149–167 ISSN 1785-8860
[5] Courbon J, Mezouar Y and Martinet P 2009 IEEE Transactions on Intelligent Transportation Systems 10 392–402
[6] Tsunekawa K, Leiva F and Ruiz J 2018 IEEE Robotics and Automation Letters 3 3247–3254
[7] Bista S, Robuffo P and Chaumette F 2016 IEEE Robotics and Automation Letters 1 423–430
[8] Lee K, Gibson J and Theodorou E 2020 IEEE Robotics and Automation Letters 5 1207–1214
[9] Montiel H, Martínez F and Martínez F 2019 TELKOMNIKA (Telecommunication Computing Electronics and Control) 17 1468 ISSN 1693-6930

[10] Ferreira M, Costa P, Rocha L and Moreira A 2016 The International Journal of Advanced Manufacturing Technology 85 57–69 ISSN 0268-3768

[11] Rodrigues R, Basiri M, Aguiar A and Miraílo P 2018 IEEE Robotics and Automation Letters 3 2079–2086

[12] Hongshan Y, Jiang Z, Yaonan W, Wenyan J, Mingui S and Yadong T 2014 Sensors 14 10753–10782 ISSN 1424-8220

[13] Freundlich C, Zhang Y, Zhu A, Mordohai P and Zavlanos M 2017 The International Journal of Robotics Research 36 1268–1285 ISSN 0278-3649

[14] Dawood Y, Ruhana K and Kamioka E 2017 6th International Conference on Computing and Informatics (ICOCI 2017) pp 235–242

[15] Mohamed A, Chenguang Y and Cangelosi A 2016 4th IFAC International Conference on Intelligent Control and Automation Sciences pp 161–168

[16] Mahammed M, Melhum A and Kochery F 2013 International Journal of Science and Applied Information Technology 2 5–8 ISSN 2278-3083

[17] Martínez F, Jacinto E and Martínez F 2020 International Journal of Electrical and Computer Engineering 10 2164–2172 ISSN 2088-8708

[18] Mehta S 2008 Computers and Electronics in Agriculture 63 28–37 ISSN 0168-1699

[19] Martinez F, Jacinto E and Martinez F 2020 International Journal of Electrical and Computer Engineering 10 2164–2172 ISSN 2088-8708

[20] Mehrabi M, Peek E, Wensual B and Lutteroth C 2013 Proceedings of the Fourteenth Australasian User Interface Conference (AUIC 13) vol 139 pp 91–100

[21] Lins R, Givigi S and Gardel P 2015 IEEE Transactions on Instrumentation and Measurement 64 2950–2958 ISSN 0018-9456

[22] Moreno A and Páez C 2017 Tekhnie 14 61–72 ISSN 1692-8407