Scientific Paper / Artículo Científico

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR
INGENIUS
Revista de Ciencia y Tecnología
https://doi.org/10.17163/ings.n32.2024.08
pISSN: 1390-650X / eISSN: 1390-860X

Neural network-based robot localization using visual features

Localización de robots basada en red neural utilizando características visuales

Felipe Trujillo-Romero[1]

Received: 16-11-2023, Received after review: 29-05-2024, Accepted: 12-06-2024, Published: 01-07-2024

Abstract

This paper outlines the development of a module capable of constructing a map-building algorithm using inertial odometry and visual features. It incorporates an object recognition module that leverages local features and unsupervised artificial neural networks to identify non-dynamic elements in a room and assign them positions. The map is modeled using a neural network, where each neuron corresponds to an absolute position in the room. Once the map is constructed, capturing just a couple of images of the environment is sufficient to estimate the robot's location. The experiments were conducted using both simulation and a real robot. The Webots environment with the virtual humanoid robot NAO was used for the simulations. Concurrently, results were obtained using a real NAO robot in a setting with various objects. The results demonstrate notable precision in localization within the two-dimensional maps, achieving an accuracy of ± (0.06, 0.1) m in simulations contrasted with the natural environment, where the best value achieved was ± (0.25, 0.16) m.

Keywords: Visual Features, Bidimensional Maps, Inertial Odometry, Humanoid Robot NAO, A-KAZE descriptor, Growing Cell Structure

Resumen

Este artículo presenta el desarrollo de un módulo que puede desarrollar un algoritmo de construcción de mapas mediante odometría inercial y características visuales. Utiliza un módulo de reconocimiento de objetos basado en características locales y redes neuronales artificiales no supervisadas para conocer elementos no dinámicos en una habitación y asignarles una posición. El mapa está representado por una red neuronal donde cada neurona corresponde a una posición absoluta en la habitación. Una vez construido el mapa, basta con capturar un par de imágenes del entorno para estimar la ubicación del robot. Los experimentos se realizaron mediante simulación y utilizando un robot real. Se utilizó el entorno Webots con el robot humanoide virtual NAO para realizar las simulaciones. Al mismo tiempo, se obtuvieron resultados utilizando un robot NAO real en un escenario con diversos objetos. Los resultados muestran una buena precisión en la localización dentro de los mapas bidimensionales de ±(0,06, 0,1)m en simulación en contraste con el entorno natural; el mejor valor obtenido fue ±(0,25, 0,16)m.

Palabras clave: características visuales, mapas bidimensionales, odometría inercial, robot humanoide NAO, descriptor A-KAZE, estructura de crecimiento celular

[1,*]Departamento de Ingeniería Electrónica, DICIS, Universidad de Guanajuato, México.
Corresponding author ✉: fdj.trujillo@ugto.mx.

Suggested citation: Trujillo-Romero, F. "Neural network-based robot localization using visual features," *Ingenius, Revista de Ciencia y Tecnología*, N.° 32, pp. 77-89, 2024, DOI: https://doi.org/10.17163/ings.n32.2024.08.

1. Introduction

According to the International Federation of Robotics (IFR) [1], a service robot is a robotic system that operates wholly or partially autonomously to perform valuable services for the well-being of humans and equipment, excluding manufacturing operations. Service robots are specifically designed for human environments, such as homes, hospitals, and restaurants, requiring them to make complex decisions. These include identifying, detecting, recognizing, and manipulating various objects within their surroundings.

For a service robot to operate autonomously, it must be equipped with a control system that enables interaction with its environment to make the right decisions and achieve specific objectives. A critical component of this control system for service robots involves learning about the environment in which they will operate. Initially, the robot must familiarize itself with the location and the non-dynamic elements it will interact with. For instance, in some competitions, participants are given a period to acquaint themselves with the scenario interactions and perform the necessary calibrations to complete the tasks.

The ability to see enhances its interaction with people and the environment. For instance, in [2], vision sensors are used for localization and mapping. In [3], a stereovision system is developed to detect targets from the generated depth map. Additionally, Scona et al. [4] used a stereovision sensor to explore challenges such as motion blur, lack of visual features, illumination changes, and fast motion.

In environmental localization by mobile robots, in [5] a vision system was implemented to develop algorithms for simultaneous localization and mapping (SLAM). In [6], an application was developed for topological mapping and navigation using visual SLAM. Ovalle-Magallanes et al. [7] utilized visual information to create an appearance-based localization system for a humanoid robot. Lasguignes et al. [8] implemented an ICP-based localization system using visual information on the TALOS humanoid robot. Conversely, Wozniak et al. [9] proposed an algorithm for visual place recognition using images acquired by a humanoid robot, with a neural network as the recognizer. In [10], an augmented landmark vision-based ellipsoidal SLAM was developed on an NAO humanoid robot for indoor scenarios. Additionally, a method for efficient SLAM using a forward-viewing monocular vision sensor was implemented in [11].

In addition to RGB cameras, other sensors are employed, such as in [12], where an IMU sensor was utilized to locate a humanoid robot in the environment. In [13], a combination of 2D LiDAR and odometry was implemented to enable a robot to navigate and find itself. Wen et al. [14] presented an EKF-SLAM using camera and laser sensors for indoor localization and mapping. In [15], a vision-based SLAM allows a mobile robot to navigate unknown environments. In [16], a SLAM system is proposed to estimate the robot's poses and build a 3D environment map. Furthermore, features-based tracking from a stereo vision sensor was combined to obtain a hybrid SLAM [17].

Meanwhile, Cheng, Sun, and Meng [18] utilized feature points to develop a method that integrates optical flow with ORB-SLAM to differentiate between dynamic and static elements. In [19], Ganesan et al. proposed a method to reduce the search space for the RRT* algorithm in path-planning tasks. Feature matching for map-building algorithms was explored using the distance from a point cloud obtained from a range finder sensor [20]. An environment map is built using a sensor fusion of odometry, 2D laser, and RGB-D [21]. A proposal where the environment is represented by 3D polygons that enable a robot to localize is presented in [22]. In contrast, a topological navigation system based on symbolic representation was proposed in [23] for a humanoid robot.

All the works mentioned above employ techniques to enhance localization, mapping, or object searching within a human environment, carried out by a mobile robot. For this reason, the humanoid robot NAO [24] is utilized as a platform for implementing localization and mapping in this study.

The remainder of this paper is structured as follows: Section 2 outlines the various methods and materials used in this study. Subsequently, Section 2.5 presents the implementation of the proposed system. The results obtained with both platforms are detailed in Section 3. Finally, Section 4 discusses the conclusions and directions for future research.

2. Materials and Methods

2.1. NAO robot

The NAO robot, as depicted in Figure 1 (a) is the pivotal robotic platform chosen to implement the developed system. NAO, a medium-height autonomous and programmable robot [24], is widely recognized as one of the market's most sophisticated and comprehensive robots. Over the years, five versions have been introduced, each incorporating specific enhancements, while the fundamental concept remains unaltered.

Figure 1 (b) presents a schematic of the robot, indicating its dimensions, including height, width, and arm length. The NAO robot is equipped with the embedded software NAOqi, which operates on the robot to provide autonomy. NAOqi is integrated into the robot's operating system, OpenNAO, an embedded GNU/Linux distribution based on Gentoo. This system includes numerous libraries and programs essential for NAOqi. A notable feature is the capability to run

copies of NAOqi on a computer, facilitating the use of virtual robots.

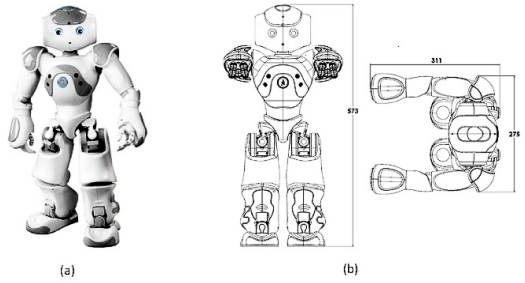

Figure 1. NAO (a) Robot, and (b) dimensions in mm [25]

2.2. A-KAZE descriptor

The A-KAZE [26] method, depicted in Figure 2, is divided into three main tasks: (1) construction of a nonlinear scale space, (2) feature detection, and (3) feature description. The construction of the nonlinear scale space involves processing an input image using the Fast Explicit Diffusion (FED) [25] numerical method, applied with a pyramidal approach.

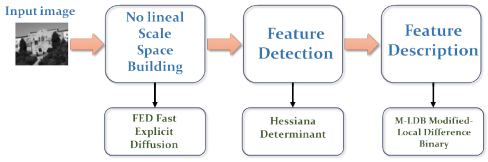

Figure 2. Overview of the A-KAZE algorithm

Initially, the scale space is discretized into a series of O octaves and S sublevels, identified by discrete indices (o and s, respectively). Subsequently, they are mapped to their corresponding scale, σ, using Equation (1).

$$\sigma(o,s) = 2^{\frac{o+s}{s}} \tag{1}$$

The input image is convolved with a Gaussian standard deviation σ 0 to reduce noise and potential artifacts, considering both the input image and a contrast factor λ, which is automatically calculated by the algorithm. Subsequently, 2D features of interest that exhibit a normalized scale determinant of the Hessian response are detected across the nonlinear scale space for each filtered image. Normalization is performed using a factor that accounts for the scale of each image in the nonlinear scale space, as illustrated in Equation (2).

$$L^i_{Hessiana} = \sigma^2_{i,norm}(L^i_{xx}L^i_{yy} - L^i_{xy}L^i_{yx}) \tag{2}$$

The Scharr Concatenated Filter [27] calculates second-order derivatives to approximate rotation invariance. Initially, the maximum response of the detector at a specific spatial location is obtained to estimate the 2D position of the key point. This is achieved by fitting a quadratic function to the maximum response of the Hessian determinant within a 3x3 neighborhood.

Finally, the principal orientation of the key point is calculated using the Modified-Local Difference Binary (M-LDB) descriptor [28]. This method utilizes information about gradients and intensity from the nonlinear scale space to generate a descriptor vector of length 64.

In the case of the descriptor used, its primary advantage is its superior performance in obtaining visual information when implementing the mapping system, owing to its invariance to scale and rotation changes. Additionally, it operates faster than other descriptors, and the algorithm's author provides the code. Among the disadvantages, it is necessary to mention that precise tuning of the threshold used to identify characteristic points is required, along with the adjustment of the number of levels and sublevels within the nonlinear scale space.

2.3. Growing Cell Structure

Growing Cell Structures (GCS) [29] are available in supervised and unsupervised variants. The variant of interest in this context is the unsupervised model, which offers the significant advantage of automatically determining an appropriate network structure and size. This capability is facilitated through controlled growth, which includes the periodic removal of units. This model builds upon Kohonen's [30] work on self-organizing maps. The pseudocode for GCS is presented in Algorithm 1 (see figure 3).

Algorithm 1: Growing Cell Structure [30].

Data: ϵ_b Best matching, ϵ_n neighboring and λ steps
Start: k-dimensional simplex $V = R^b$
while ($\neq$ *desired network size*) **do**
  for *adaptationsteps* $= 0 \rightarrow \lambda$ **do**
    Choose an input signal ξ according to $P(\xi)$
    Locate the best matching unit $s = \phi_w(\xi)$.
    Increase matching:
    $\Delta w_s = \epsilon_b(\xi - w_s)$
    $\Delta w_c = \epsilon_n(\xi - w_c)(\forall c \in N_s)$
    Increment the signal counter of s:
    $\Delta\tau_s = 1$
    Decrease all signal counters by a fraction α in the network A:
    $\Delta\tau_c = -\alpha\tau_c(\forall c \in A)$
Determine $q : h_q \geqslant h_c(\forall c \in A)$
Look q largest distance neighbor $f : \| w_f - w_q \| \geqslant \| w_c - w_q \| (\forall c \in N_q)$
Insert cell r between q ans f.
Initialize $r : w_r = 0 : 5(w_q + w_f)$
Redistribute counter:

$$\Delta\tau_c = \frac{|F_c^{new}| - |F_c^{old}|}{|F_c^{old}|}\tau_c$$

Initialize new cell:

$$\tau_r = -\sum_{c \in N_r} \Delta\tau_c$$

After insertion, check $\hat{p}_i < \eta$
Cells remove:

$$\hat{p} = \tilde{p} - \sum_{c \in A} \tilde{f}_c$$

Figure 3. Growing cell structure [30]

The Growing Cell Structure (GCS) algorithm offers several key advantages. It can autonomously adjust the number of neurons, adding or removing them as required. It operates as an unsupervised network, enabling it to form associations of input vectors independently of external input. Its simplicity of implementation is also a notable feature. However, a noted disadvantage is that the network may fail if the vectors to be associated are very close to each other.

2.4. WEBOTS robotic simulator

Webots [31] is an open-source, multi-platform desktop application for robot simulation. For this reason, Webots will be utilized to simulate the system and facilitate its respective validation.

This software simulator enables testing applications and algorithms for the NAO robot within a virtual environment. Figure 4 illustrates the software environment, a virtual world that simulates NAO movements while adhering to physical laws. This environment offers a secure setting for testing behaviors before they are implemented on a real robot.

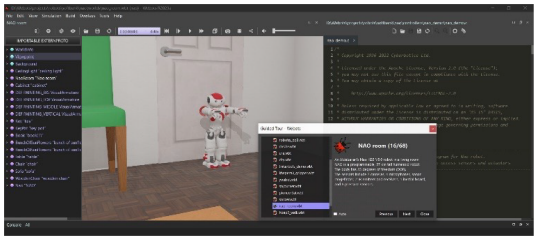

Figure 4. Webots environment [32]

2.5. Bidimensional map construction

As mentioned in the Introduction section, autonomy is achieved through a system of activity planning and control, designed to ensure the fulfillment of its objectives. One key feature of these systems is spatial navigation, which enables the calculation of a robot's pose (position and orientation) based on incremental, inertial, and visual measurements. This section presents the concepts of odometry, and visual features employed in the spatial navigation module. These tools enable the robot to construct a two-dimensional map and localize itself while navigating.

2.5.1. Odometry

Odometry facilitates estimating a mobile robot's relative position within an environment during navigation, starting from its initial location. Additionally, it records and tracks the robot's movement within a space to construct a two-dimensional map. The NAO robot has functions that address several challenges, including odometry. Algorithm 2 (see figure 5 displays the pseudocode where functions from Aldebaran's inertial odometry [24] are utilized.

Algorithm 2: Pseudo code to store inertial odometry using Aldebaran functions [25].

```
1 //Store the initial position
2 AL::Math::Pose2D worldToRobotInit= Pose2D(getRobotPosition())
3 //Wait until it finishes scrolling
4 //Storing the final position
5 AL::Math::Pose2D worldToRobotAfter= Pose2D(getRobotPosition())
6 Pose2D robotMove = pose2DInverse(worldToRobotInit)+worldToRobotAfter //Movement
7 theta = modulo2PI(robotMove.theta)//Angle
```

Figure 5. Pseudo code to store inertial odometry using aldebaran functions [26]

In this implementation, the two-dimensional position of the robot is initialized with explicit values obtained from the initialized pose values, which are retrieved from the articulations' magnetic rotary encoders (MRE). Each time the robot is activated, it records an absolute position within the scenario world.

In constructing the two-dimensional map, the robot first saves its initial position. Then, the robot is instructed to follow a predetermined closed-circuit path within the room, advancing a specified distance while walking. As it moves, the robot's two-dimensional position is recorded periodically. Subsequently, the displacement and angle traversed by the robot are calculated. The two-dimensional position between consecutive points is then computed to accurately reflect the robot's movement.

Thus, the general implementation of odometry can be established as follows:

1. Capture the robot's position relative to the world before walking.
2. Detect when the robot starts walking.
3. Simultaneously begin collecting odometric data.
4. Process and accumulate odometric data.
5. Detect the completion of the robot's walk. If the walk is not completed, repeat steps 3 and 4.
6. Calculate the distance traveled by the robot.
7. Store the robot's distance and position data to construct the two-dimensional map.

Figure 6 presents the general flow diagram for generating the bidimensional map. In this diagram, the algorithm initiates with the capture of an image. Subsequently, visual features are extracted from this image; these features serve as inputs for the learning system, namely the neural network. Following this, the robot learns and records the spatial localization corresponding to its position. If the designated path is completed, the algorithm concludes. If not, the robot moves to the next position and the algorithm continues until the end of the path is reached.

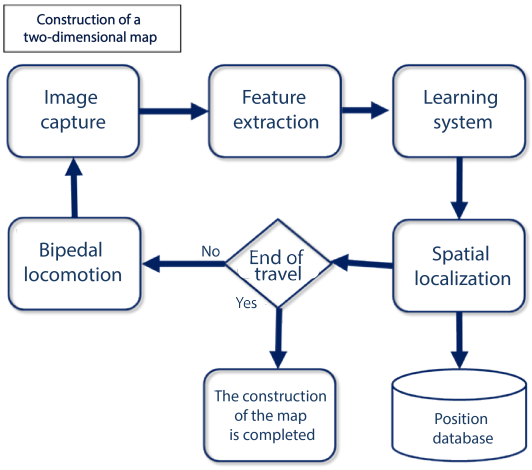

Figure 6. Dataflow diagram for the map reconstruction

2.6. Visual features

Visual elements are identified by analyzing and cataloging existing details in the environment, considering the robot's position when the image is captured. It is generally considered that the most significant visual elements for representing the environment are those located on or near the walls. A two-dimensional map of the room traversed by the robot can be constructed using these visual elements and the estimated location derived from odometry. For instance, figure 7 illustrates a virtual room with various everyday objects typically found in a home. These objects usually remain stationary. Therefore, the robot must navigate through this room following a closed, preferably quadrangular, circuit, capture images, and record the estimated position from where each image was taken.

Figure 7. Virtual room in Webots

Furthermore, the robot should focus on capturing images of the nearest wall to its path. Figure 8 illustrates three captures made by the robot at different points. During navigation, it will take a screenshot at each step based on the number of images specified by the user for the room. For instance, if 20 images are required in a room where each wall is 4 meters long, one image will be taken every 20 cm. In addition to the number of captures and the dimensions of the room's walls, the tour frequency can also be determined. The more circuits completed, the more detailed the construction of the room map will be, and the easier it will be to localize the robot.

Once the circuits are completed, the robot uses the stored information to construct the two-dimensional map. The captured images will contain objects from which specific details must be extracted. The object recognition module [32] processes the images to obtain descriptors, which are then learned and linked to the robot's pose during capture. This information is integrated into a two-dimensional representation, forming the map of the room. Before initiating any room tour, the robot must identify the nearest wall to determine where it will focus its image captures by simply turning its head toward the visible wall. This wall detection is achieved by visually estimating distances. Before navigation, the robot should be positioned parallel to the selected wall and placed in a corner of the room. It then captures an image, which is subsequently analyzed by dividing it into two parts.

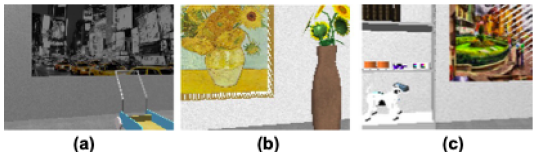

Figure 8. Images acquired by the NAO robot in a simulated room

For instance, Figure 9 displays three different captures of a room taken from various robot positions. In image 9(a), the nearest wall is to the left, while in images 9(b) and 9(c), it is on the right. For each image, interest points on each side are identified using the A-KAZE algorithm [26]. The image with the most salient points indicates the location of the nearest wall, assuming the room is free of obstacles.

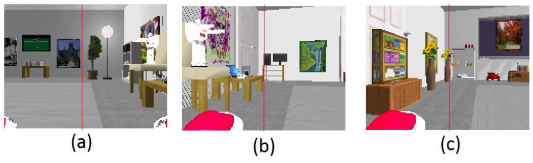

Figure 9. Images taken by the NAO robot before

Figure 10 displays the evaluation results for each image, with salient points indicated by small colored circles. In image 10(a), the right side contains the most salient points, numbering 108 compared to 36 on the left; in 10(b), the left side predominates with

128 points versus 50 on the right; and in 10(c), the left side again leads with 119 points compared to 53 on the right. Based on these observations, the robot then turns towards the side with more salient points to continue its exploration of the room.

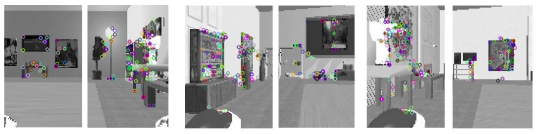

Figure 10. Wall detection using descriptors

2.7. Algorithm for the map construction

The construction of the two-dimensional map proceeds as follows: Initially, the robot performs a closed-loop circuit around a square room, capturing and recording images along with their respective poses. Knowledge of the wall dimensions, the step size during movement, and the number of iterations is essential. The robot enhances its understanding of the room with each additional circuit completed. At the commencement of Algorithm 3 (see figure 11), the robot executes an initial capture to detect the nearest wall and determine the angle for its subsequent turn.

Before commencing its movement, the robot records its current pose through odometry as the global reference point for the room. Subsequently, the distance traveled is logged, indicating both the length of the robot's journey and the total distance it needs to navigate within the room. This measurement is continually monitored by a work cycle, which persists until the traveled distance equals the combined length of the room's four walls.

Algorithm 3: Navigation Module. *Execution of a closed lap.*

Data: ***d*** dimensions of the room, ***p*** step size of the robot when walking, ***n*** number of paths
Result: ***data*** images and poses

```
picture =TakePicture()
AngleYaw=DetectNearestWall(picture)
TurnHead(AngleYaw)
O_w=CurrentPose()
TotalDistanceWalked=0
for j = 1 a n do
    while TotalDistanceWalked ≠ d × 4 do
        DistanceWalked=0
        while DistanceWalked ≠ d do
            Walk(p)
            RP=CurrentPose()
            picture =TakePicture()
            data = (P, I)
            DistanceWalked=DistanceWalked+p
        TurnBody(-AngleYaw)
        TotalDistanceWalked=TotalDistanceWalked+DistanceWalked
```

Figure 11. Navigation module. Execution of a closed lap

After completing the circuit and storing the room database, Algorithm 4 (see figure 12), is initiated to learn from a new database that includes capture and pose information. All interest points are extracted, histograms are constructed, and a neural network is trained using the Growing Cell Structures (GCS) method [32].

Algorithm 4: Navigation Module. *Construction of two-dimensional map*

Data: ***I*** images, ***P*** poses.
Result: ***classes(N,P)*** classes of the two-dimensional map.

```
for i ← 1 to I do
    keypoints =AKAZE(I(i))
    H(i)=BuildHistos(keypoint)
ANN_trained=GCS(H)
classes(N)=ANN_trained
for i ← 1 to N do
    NewPose(i) = (1/n) Σ_{j=1}^{n} P(classes(i))
```

Figure 12. Navigation module. Construction of two-dimensional map

Algorithm 5 (see figure 13), is used to evaluate the map. This module processes the images, extracting salient points and constructing histograms. These histograms are then used to assess the trained neural network. This process helps identify the corresponding neurons. Once the classes are determined, the associated poses are retrieved. The two-dimensional positions on the map are then calculated and returned.

Algorithm 5: Navigation Module. *Using the two-dimensional map*

Data: ***I*** images
Result: ***class(I)*** object classes, ***Pose*** pose

```
keypoints =AKAZE(I)
H(I)=BuildHistos(keypoint)
classes(I)=ANN_trained
Pose=(1/n) Σ_{j=1}^{n} NewPose(I)(classes(i))
```

Figure 13. Navigation module. Using the two-dimensional map

The algorithm operates within certain constraints, including knowing the room's dimensions to calculate the total distance the robot will traverse around it. Additionally, the environment must be free of obstacles, as this work does not incorporate obstacle-avoidance strategies.

Finally, if any elements within the room have been moved, the robot must reconstruct its navigation map to reflect these changes.

3. Results and discussion

The experiments described in this section are divided into two parts: (1) constructing a two-dimensional map and (2) localization within the map. These experiments have been conducted using both virtual and real NAO robots.

3.1. Virtual environment

3.1.1. Map construction

The simulated room depicted in Figure 7, measuring 6×6 meters, was created using Webots. This room was

furnished with various objects, such as chairs, tables, and portraits. A virtual NAO robot was employed to construct the two-dimensional map of the room. The robot initiated its journey from the lower left corner of the room, navigating in a quadrangular closed circuit. The robot turned its head towards the walls throughout its journey to capture images. The walls were numbered from 1 to 4 in a counterclockwise direction to illustrate the results of constructing the two-dimensional map.

The robot completed two counterclockwise circuits around the room, capturing images and recording their spatial relationships. The number of images taken per wall is detailed in Table 1. The term 'lap' refers to the number of circuits the robot completes. 'Images' denotes the total number of images saved during each lap. 'Wall1', 'Wall2', 'Wall3', and 'Wall4' indicate the number of images stored for each wall. In total, 164 images and their associated poses, which were used to construct the two-dimensional map, were recorded.

Table 1. Main parameters of the experiments: Construction of the two-dimensional map

Lap	Images	Wall 1	Wall 2	Wall 3	Wall 4
1	89	20	21	31	17
2	75	18	20	24	13

The parameters corresponding to object recognition are detailed in Table 2. As noted, three iterations of two-dimensional map construction were conducted using 100, 200, and 300 neurons, respectively. The objective was to assess the module's effectiveness in constructing a map accurately reflecting the robot's observations within the room.

Table 2. Parameters of the object recognition module for constructing the two-dimensional map

Experiment	Training	Neurons	Epochs	Time (sec)
1	164	100	100	4,063
2	164	200	200	14,287
3	164	300	300	33,347

After training, two-dimensional maps containing 72, 111, and 132 poses were generated. The points marked on each map in Figure 14 represent a pose associated with a neuron. It is evident that as the number of neurons increases, the distribution of poses becomes more refined. It is important to note that the poses were homogenized across the coordinates and were kept constant during the tour to ensure a precise distribution is displayed.

The distribution in the map constructed with 100 neurons is suboptimal, as it includes some poses within areas where the robot has not traveled, along with clustering of poses in certain sections. The distribution is significantly improved in the map constructed with 200 neurons, although some pose stacking is still evident. The map with 300 neurons exhibits the best distribution, covering more areas comprehensively. While a few erroneous poses are still present, they are minimal.

Thanks to the two-dimensional map, the robot can identify the locations of walls, allowing it to avoid them while executing its tasks.

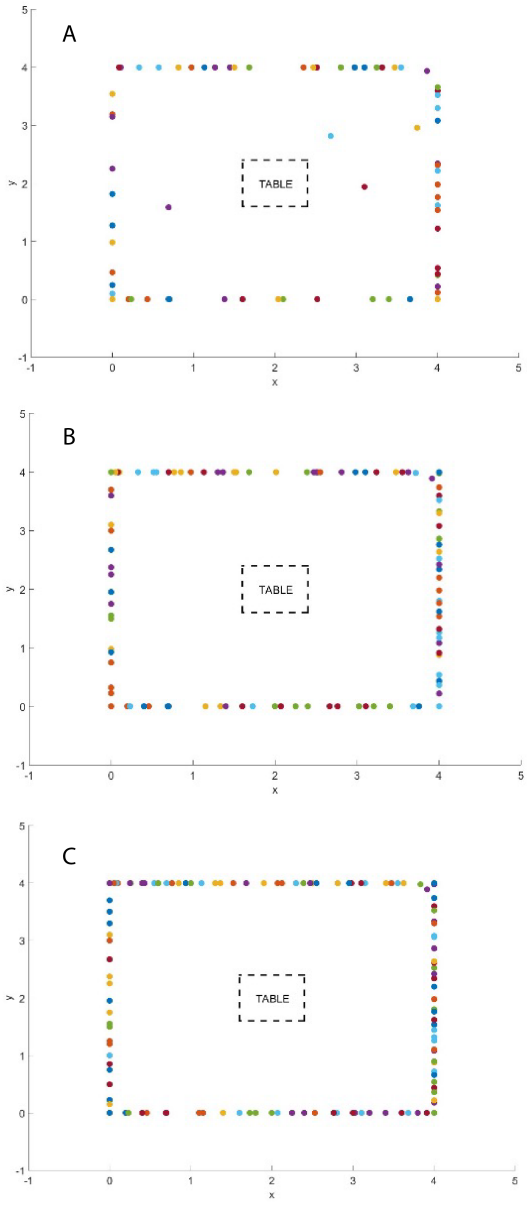

Figure 14. Distribution of neurons by poses in the room of the experiments: (a) 1, (b) 2, and (c) 3

3.1.2. Map location

The purpose of the two-dimensional map is to enable the robot to return to the global position 0 on the map once it has completed its tasks. With the constructed map, the robot can determine its location

within the room using one or two images of the nearest walls. Four experiments were conducted to validate this functionality.

Table 3 lists the parameters, which include the experiment number, the two-dimensional map constructed in the previous section (1, 2, and 3), and the actual position to be calculated (x, y) in meters.

The two-dimensional map construction is evaluated as follows: the virtual robot captures two images from different perspectives at each of the four positions nearest to the walls under assessment. Examples of these captures by the robot are illustrated in Figure 15. At each position, two images of the closest walls are captured.

The first two upper images correspond to the position (0,0), while the subsequent two correspond to the position (4,4) within the room. The results are presented in Table 3, which details five evaluations with two images for each experiment. The module records the individual poses captured in each column for the four experiments, featuring a pair of images per evaluation.

Table 3. Results of the poses evaluations and locations on the map

N°	Map	(x,y) m	1	2	3	4	5
1	1	(3.5,0.5)	(3.7,1.7)	(3.6,1.6)	(3.7,1.7)	(3.8,1.8)	(3.3,1.7)
2	1	(0.0)	(0.1,0.2)	(0.2.0.2)	(0.3.0.1)	(0.3,0.0)	(0.3.0.2)
3	2	(0.5,3.5)	(0.0,2.0)	(0.0,2.0)	(0.2,2.2)	(0.0,2.0)	(0.2,2.2)
4	3	(4,4)	(4.0,3.8)	(3.9,3.9)	(4,4)	(3.9,3.9)	(3.9,3.9)

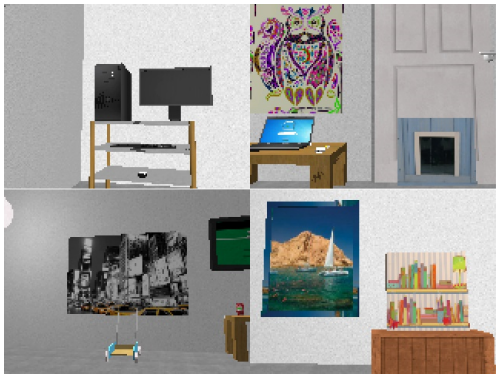

Figure 15. Examples of captures made by the virtual NAO robot

3.2. Real Scenario

3.2.1. Map construction

The two-dimensional map was constructed in a 4×3 meter room, within which the robot developed a 3×3 meter map. The room contains various elements, including posters with diverse information. Figure 16 displays the room's four walls, illustrating the elements used for learning. Additionally, a 30 cm platform is positioned at the center of the room, as shown in Figure 17. This platform holds 20 objects distributed along the edges, enhancing visibility for the robot and ensuring the objects remain within the work area of the handlers for easy retrieval.

This evaluation completed three circuits to construct a more accurate map. The robot initiated its route from the room's global coordinate (0,0), positioned in the right corner of wall number one.

During the tours, the robot turns its head toward the wall to capture images while advancing and maintaining its relative position (see Figure 18).

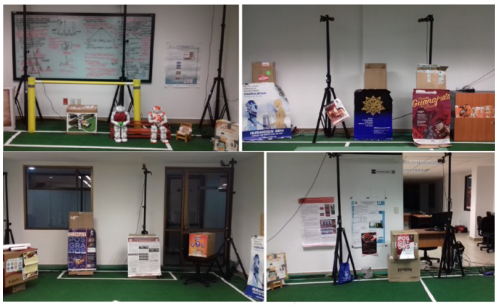

Figure 16. The walls of the real scenario for building the map

Figure 17. Platform with objects placed in the center of the room

Figure 18. The robot makes its path by heading towards the wall to capture images

Table 4 details the images captured during each circuit along the walls. This table specifies the number

of circuits completed, the total images taken, and the images captured corresponding to each wall. A total of sixty-eight images and poses used to construct the two-dimensional map were recorded.

Like the virtual scenario, the captured images are input to the object recognition module, responsible for feature learning and generating the room map.

Table 4. Main parameters of the experiments: Construction of the two-dimensional map

Lap	Images	Wall 1	Wall 2	Wall 3	Wall 4
1	25	7	7	5	6
2	26	5	8	5	8
3	17	5	5	4	3

Table 5 presents the parameters used to generate the room map, including the number of experiments conducted, the images used for training, the number of neurons, and the epochs involved.

Table 5. Parameters for constructing the map

Experiment	Training	Neurons	Times
1	68	400	100
2	68	500	200

After training, the two-dimensional maps contained 27 and 36 poses, respectively. Figure 19 displays these maps, where blue points indicate the poses at which the robot captured images. Figure 19(a) illustrates the ideal map, showing the target poses for image capture during the experiments. The map constructed for experiment 1 corresponds to Figure 19(b), while the map for the second experiment is depicted in Figure 19(c). These maps reflect the distribution of neurons associated with each pose. It is noted that increasing the number of neurons from 400 to 500 slightly improves the distribution. However, the size of the constructed map was reduced from 3 × 3 meters to 1.5 × 1.5 meters.

From the analysis above, it can be inferred that the reduction in map size resulted from numerous false positives and the inter-association of poses, which led to their consolidation.

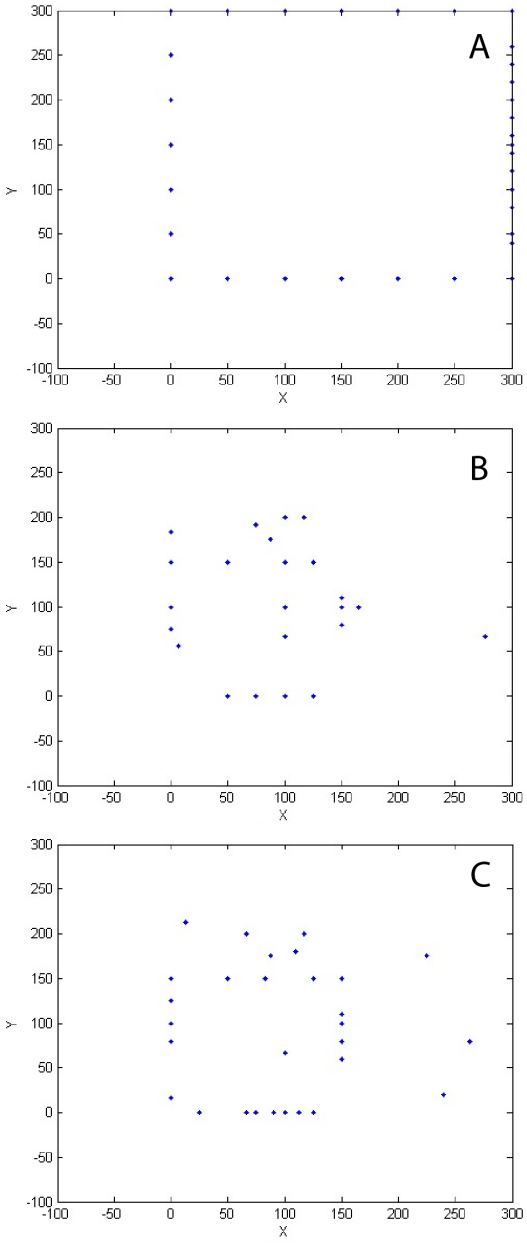

Figure 19. Distribution of neurons by poses in the room

3.2.2. Map localization

The task of localization on the two-dimensional map serves several purposes. One key objective is for the robot to return to the starting point to deliver an object as requested by the user. Additionally, the robot uses the map to locate room walls, which helps avoid them during search tasks for items.

Therefore, with the constructed map, the robot can determine its location within the room using one or two images of the nearest walls. This capability was assessed through ten experiments that were conducted.

Table 6 lists the experiment numbers and the corresponding positions to be calculated (x, y) in meters.

Table 6. Main parameters of experiments for the map location

N°	1	2	3	4	5	6	7	8	9	10
(x,y)m	(0,0)	(0,1)	(0,2)	(0,3)	(1,0)	(2,0)	(3,0)	(3,3)	(1.5,2)	(2,1.5)

The accuracy of the robot's location on the two-dimensional map is determined using two images captured from the closest potential positions of the wall adjacent to that point.

The results are presented in Table 7, where the 10 locations are associated with two poses derived from evaluating the two images taken from each position. Due to inaccuracies in constructing the two-dimensional map, the obtained poses do not closely match the actual positions.

The highest precision was achieved with pose number (1), showing an accuracy of ± (0.25, 0.16) close to the expected pose. The next best precision was obtained with pose number (9), showing an accuracy of ± (0.34, 0.62). The least accurate poses were (2) and (8), with precisions of ± (1.50, 0.50) and ± (1.75, 2.00), respectively. Although the constructed map was inaccurate, the evaluation yields favorable results given the trained map. Figure 20 provides a graphical representation of the 10 locations determined using the module on the previously trained two-dimensional map. It is observed that most poses are very close to the trained positions, except for poses 2, 7, and 8, which were significantly misaligned.

Table 7. Results of the evaluations of poses for each experiment

N°	1	2	3	4	5	6	7	8	9	10
1	(0,0.16)	(1.50,1.50)	(0.75,0)	(1.00,0)	(0,0.80)	(0,1.0)	(0.25,0)	(1.25,0)	(1.06,1.75)	(1.50,0.80)
2	(0.25,0)	(1.50,0.60)	(1.25,0)	(0,0.80)	(0.25,0)	(0.25,0)	(0.66,0)	(0,1.00)	(1.25,1.00)	(1.25,1.00)

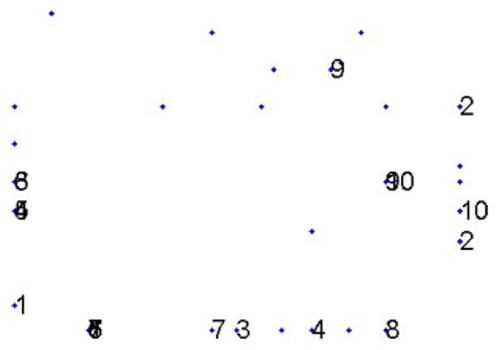

Figure 20. Plot with the 10 locations determined on the trained map

4. Conclusions

This study presents the development of an algorithm for constructing two-dimensional maps using inertial odometry and visual elements. The two-dimensional map is created utilizing an object recognition module based on local features and unsupervised artificial neural networks. This module is employed to learn the room's layout and associate a pose with each neuron in the network, which is trained to represent the two-dimensional map.

Experiments were conducted using (1) a virtual NAO robot and (2) a real NAO robot within an authentic scenario. The results are promising, as it was possible to construct a two-dimensional map of the room and accurately locate the mobile robot with a precision of up to ± (0.06, 0.1) in simulation and ±(0.25, 0.16) in the natural environment. These results can be further improved by enhancing the quality of the images.

1. The approach to generating maps from visual information has several limitations, including the following: The NAO robot's cameras are not optimal for capturing high-quality images, leading to errors in both the learning and recognition phases.
2. The environment must be structured to include sufficient visual references on the room's walls to improve the robot's localization accuracy.
3. This implementation does not account for dynamic elements; therefore, the scene only contains the robot, the table, and the surrounding objects.
4. The robot's path must be straight, necessitating a clear path free of objects to allow proper positioning relative to the wall and its visual markers.
5. The approach relies heavily on visual information, so the absence of such information would cause confusion and significantly hinder the robot's ability to navigate around the room.

4.1. Future work

1. **Enhanced Sensor Fusion:** Future work will focus on improving data integration from inertial

odometry and visual elements. This approach aims to reduce reliance on solely visual features, thereby enhancing the robustness and accuracy of the system.

2. **Evaluation of Neural Network Architectures:** Various neural network architectures will be evaluated to determine the most suitable for the construction map task. The architecture that demonstrates the best performance will be selected for further development and implementation.

3. **Testing Advanced Feature Point Detectors:** To enhance system performance, state-of-the-art feature point detectors will be tested. These detectors are expected to offer significant improvements in the detection and processing of feature points, contributing to overall system efficiency.

References

[1] IFR. (2024) Homepage. International Federation of Robotics. International Federation of Robotics. [Online]. Available: https://ifr.org/

[2] Y. Omori, T. Furukawa, T. Ishikawa, and M. Inaba, "Humanoid vision design for object detection, localization and mapping in indoor environments," in *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2018, pp. 1–6. [Online]. Available: https://doi.org/10.1109/SSRR.2018.8468604

[3] X. Cui, M. Wang, B. Fan, and J. Yi, "Target detection based on binocular stereo vision," in *2017 International Conference on Computer Technology, Electronics and Communication (ICCTEC)*, 2017, pp. 1093–1097. [Online]. Available: https://doi.org/10.1109/ICCTEC.2017.00239

[4] R. Scona, S. Nobili, Y. R. Petillot, and M. Fallon, "Direct visual SLAM fusing proprioception for a humanoid robot," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 1419–1426. [Online]. Available: https://doi.org/10.1109/IROS.2017.8205943

[5] L. K. Garzón Obregón, L. A. Forero Rincón, and O. M. Duque Suárez, "Diseño e implementación de un sistema de visión artificial usando una técnica de mapeo y localización simultánea (SLAM) sobre una plataforma robótica móvil," *Mundo FESC*, vol. 8, no. 16, pp. 8–17, 2018. [Online]. Available: https://is.gd/pqjvTy

[6] F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart, "Topomap: Topological mapping and navigation based on visual SLAM maps," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3818–3825. [Online]. Available: https://doi.org/10.1109/ICRA.2018.8460641

[7] E. Ovalle-Magallanes, N. G. Aldana-Murillo, J. G. Avina-Cervantes, J. Ruiz-Pinales, J. Cepeda-Negrete, and S. Ledesma, "Transfer learning for humanoid robot appearance-based localization in a visual map," *IEEE Access*, vol. 9, pp. 6868–6877, 2021. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.3048936

[8] T. Lasguignes, I. Maroger, M. Fallon, M. Ramezani, L. Marchionni, O. Stasse, N. Mansard, and B. Watier, "ICP Localization and Walking Experiments on a TALOS humanoid robot," in *2021 20th International Conference on Advanced Robotics (ICAR)*, 2021, pp. 800–805. [Online]. Available: https://doi.org/10.1109/ICAR53236.2021.9659474

[9] P. Wozniak, H. Afrisal, R. G. Esparza, and B. Kwolek, "Scene recognition for indoor localization of mobile robots using deep CNN," in *Computer Vision and Graphics*, L. J. Chmielewski, R. Kozera, A. Orłowski, K. Wojciechowski, A. M. Bruckstein, and N. Petkov, Eds. Cham: Springer International Publishing, 2018, pp. 137–147. [Online]. Available: https://doi.org/10.1007/978-3-030-00692-1_13

[10] E. S. Lahemer and A. Rad, "An adaptive augmented vision-based ellipsoidal slam for indoor environments," *Sensors*, vol. 19, no. 12, 2019. [Online]. Available: https://doi.org/10.3390/s19122795

[11] T.-J. Lee, C.-H. Kim, and D.-I. D. Cho, "A monocular vision sensor-based efficient slam method for indoor service robots," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 1, pp. 318–328, 2019. [Online]. Available: https://doi.org/10.1109/TIE.2018.2826471

[12] M. Fourmy, D. Atchuthan, N. Mansard, J. Solà, and T. Flayols, "Absolute humanoid localization and mapping based on IMU Lie group and fiducial markers," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 237–243. [Online]. Available: https://doi.org/10.1109/Humanoids43949.2019.9035005

[13] S. J. Dignadice, J. R. Red, A. J. Bautista, A. Perol, A. Ollanda, and R. Santos, "Application of simultaneous localization and mapping in the development of an autonomous robot," in *2022 8th International Conference on Control, Automation and Robotics (ICCAR)*,

2022, pp. 77–80. [Online]. Available: https://doi.org/10.1109/ICCAR55106.2022.9782658

[14] S. Wen, M. Sheng, C. Ma, Z. Li, H. K. Lam, Y. Zhao, and J. Ma, "Camera recognition and laser detection based on EKF-SLAM in the autonomous navigation of humanoid robot," *Journal of Intelligent & Robotic Systems*, vol. 92, no. 2, pp. 265–277, Oct 2018. [Online]. Available: https://doi.org/10.1007/s10846-017-0712-5

[15] X. Deng, Z. Zhang, A. Sintov, J. Huang, and T. Bretl, "Feature-constrained active visual SLAM for mobile robot navigation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7233–7238. [Online]. Available: https://doi.org/10.1109/ICRA.2018.8460721

[16] A. Li, J. Wang, M. Xu, and Z. Chen, "DP-SLAM: A visual SLAM with moving probability towards dynamic environments," *Information Sciences*, vol. 556, pp. 128–142, 2021. [Online]. Available: https://doi.org/10.1016/j.ins.2020.12.019

[17] N. Krombach, D. Droeschel, S. Houben, and S. Behnke, "Feature-based visual odometry prior for real-time semi-dense stereo SLAM," *Robotics and Autonomous Systems*, vol. 109, pp. 38–58, 2018. [Online]. Available: https://doi.org/10.1016/j.robot.2018.08.002

[18] Y. S. Jiyu Cheng and M. Q.-H. Meng, "Improving monocular visual slam in dynamic environments: an optical-flow-based approach," *Advanced Robotics*, vol. 33, no. 12, pp. 576–589, 2019. [Online]. Available: https://doi.org/10.1080/01691864.2019.1610060

[19] S. Ganesan and S. K. Natarajan, "A novel directional sampling-based path planning algorithm for ambient intelligence navigation scheme in autonomous mobile robots," *Journal of Ambient Intelligence and Smart Environments*, vol. 15, pp. 269–284, 2023, 3. [Online]. Available: https://doi.org/10.3233/AIS-220292

[20] K. Zhang, H. Gui, Z. Luo, and D. Li, "Matching for navigation map building for automated guided robot based on laser navigation without a reflector," *Industrial Robot: the international journal of robotics research and application*, vol. 46, no. 1, pp. 17–30, Jan 2019. [Online]. Available: https://doi.org/10.1108/IR-05-2018-0096

[21] C. Wang, J. Wang, C. Li, D. Ho, J. Cheng, T. Yan, L. Meng, and M. Q.-H. Meng, "Safe and robust mobile robot navigation in uneven indoor environments," *Sensors*, vol. 19, no. 13, 2019. [Online]. Available: https://doi.org/10.3390/s19132993

[22] A. Roychoudhury, M. Missura, and M. Bennewitz, "3D polygonal mapping for humanoid robot navigation," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, 2022, pp. 171–177. [Online]. Available: https://doi.org/10.1109/Humanoids53995.2022.10000101

[23] F. Martín, J. Ginés, D. Vargas, F. J. Rodríguez-Lera, and V. Matellán, "Planning topological navigation for complex indoor environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9. [Online]. Available: https://doi.org/10.1109/IROS.2018.8594038

[24] Aldebaran. NAO Documentation. Aldebaran NAO Documentation. Aldebaran NAO Documentation. [Online]. Available: https://is.gd/eSNPWH

[25] MIA. (2023) Mathematical image analysis group. MIA Group. [Online]. Available: https://is.gd/69mEso

[26] P. Fernández Alcantarilla, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," in *British Machine Vision Conference (BMVC)*, 09 2013. [Online]. Available: http://dx.doi.org/10.5244/C.27.13

[27] H. Scharr, *Optimale Operatoren in der Digitalen Bildverarbeitung.* University of Heidelberg, Germany, 2000. [Online]. Available: https://doi.org/10.11588/heidok.00000962

[28] X. Yang and K. Cheng, "LDB: an ultra-fast feature for scalable augmented reality on mobile devices," *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 49–57, 2012. [Online]. Available: https://doi.org/10.1109/ISMAR.2012.6402537

[29] B. Fritzke, "Growing cell structures—a self-organizing network for unsupervised and supervised learning," *Neural Networks*, vol. 7, no. 9, pp. 1441–1460, 1994. [Online]. Available: https://doi.org/10.1016/0893-6080(94)90091-4

[30] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990. [Online]. Available: https://doi.org/10.1109/5.58325

[31] Cyberbotics. (2023) Simulating your robots with webots. Cyberbotics - Robotics simulation services. Cyberbotics - Robotics simulation services. [Online]. Available: https://is.gd/Q31yau

[32] K. L. Flores-Rodríguez, F. Trujillo-Romero, and W. Suleiman, "Object recognition modular

system implementation in a service robotics context," in *2017 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, 2017, pp. 1–6. [Online]. Available: https://doi.org/10.1109/CONIELECOMP.2017.7891833