Ingeniería y Competitividad
Vol 28 (2) 2026 doi: 10.25100/iyc.v28i2.15437

Vol 28 (2)

ISSN
0123-3033
e- 2027-8284

Original Article

Implementation of advanced voice recognition and computer vision techniques in a robotic system

Implementación de técnicas avanzadas de reconocimiento de voz y visión computacional en un sistema robótico

Jose Miguel Bejarano Restrepo[1] Robinson Jiménez Moreno[1] Anny Astrid Espitia Cubillos[1]

[1] Universidad Militar Nueva Granada, Bogotá, Colombia.

Abstract

Objetive: The present article exposes both the design and validation of a system for a SCARA-type robot with voice recognition using convolutional neural networks (CNNs), visual detection using the YOLOv12 algorithm, and movement control in a simulated environment developed in CoppeliaSim using parallel processing in Python for simultaneous management.
Materials and methods: The speech recognition algorithm (CNN) was trained on a database that was augmented using data augmentation techniques. YOLOv12 was selected as the visual detection algorithm, which is pre-trained on the COCO (Common Objects in Context) dataset. Proportional-derivative control is used for robotic movements.
Results: The voice recognition algorithm achieved an average accuracy of 98.9% on the four defined commands: start, stop, grab, and hand over. The YOLOv12 network correctly identified and located the five selected objects in the working environment with errors of less than 7% corresponding to a maximum of ± 25 px in real time.
Conclusions: The results in the virtual environment demonstrated not only stable but also coordinated performance; However, limitations were identified in CPU processing and memory consumption during prolonged operations. It is concluded that integrating voice recognition, visual detection, and control into the robotic system enhances its autonomy. Therefore, it is proposed that work be done to optimize the hardware and resource utilization for its future application in industrial and assistive robotics environments, which are typically not only dynamic but also complex.

Keywords: Voice recognition, visual detection, robotic system, convolutional neural networks, deep learning, robotic control, simulation.

Resumen

Objetivo: El presente artículo expone tanto el diseño como la validación de un sistema para un robot tipo SCARA el cual es operado por reconocimiento de voz mediante redes neuronales convolucionales (CNN), detección visual de objetos mediante la red YOLOv12, y control de movimientos para interacción en un entorno simulado desarrollado en CoppeliaSim, para lo cual se empleó procesamiento paralelo en Python para su gestión simultánea.
Metodología: El algoritmo de reconocimiento de voz (CNN) se entrenó con una base de datos que fue ampliada mediante técnicas de aumento de datos. Se seleccionó YOLOv12 como algoritmo para la detección visual, el cual está preentrenado con el conjunto de datos COCO (Common Objects in Context). Para asegurar el posicionamiento del brazo se utiliza un control proporcional-derivativo en los movimientos robóticos.
Resultados: El algoritmo de reconocimiento de voz alcanzó una precisión media del 98,9 % en los cuatro comandos definidos: iniciar, detener, agarrar y entregar, en un entorno controlado. La red YOLOv12 identificó y localizó correctamente los cinco objetos seleccionados en el entorno de trabajo con errores inferiores al 7% correspondiente a máximo ±25 px en tiempo real.
Conclusiones: Los resultados en el entorno virtual evidenciaron un desempeño no solo estable sino también coordinado, sin embargo, se identifican limitaciones en el procesamiento en CPU y en el consumo de memoria cuando las operaciones son prolongadas. Se concluye que la integración de reconocimiento de voz, detección visual y control en el sistema robótico favorece la autonomía del sistema, por lo que se propone trabajar en la optimización del hardware y uso de recursos para su aplicación futura en entornos industriales y de robótica asistencial que suelen ser no solo dinámicos sino también complejos.

Palabras clave: Reconocimiento de voz, detección visual, sistema robótico, redes neuronales convolucionales, aprendizaje profundo, control robótico, simulación.

How to cite?

Bejarano JM, Jiménez R, Espitia AA. Implementation of advanced voice recognition and computer vision techniques in a robotic system Ingeniería y Competitividad, 2026, 28(2)e-20215437

https://doi.org/10.25100/iyc.v28i2.15437

Received: 20/11/25
Reviewed: 19/01/26
Accepted: 15/04/26
Online: 15/05/26

Correspondence

anny.espitia@unimilitar.edu.co

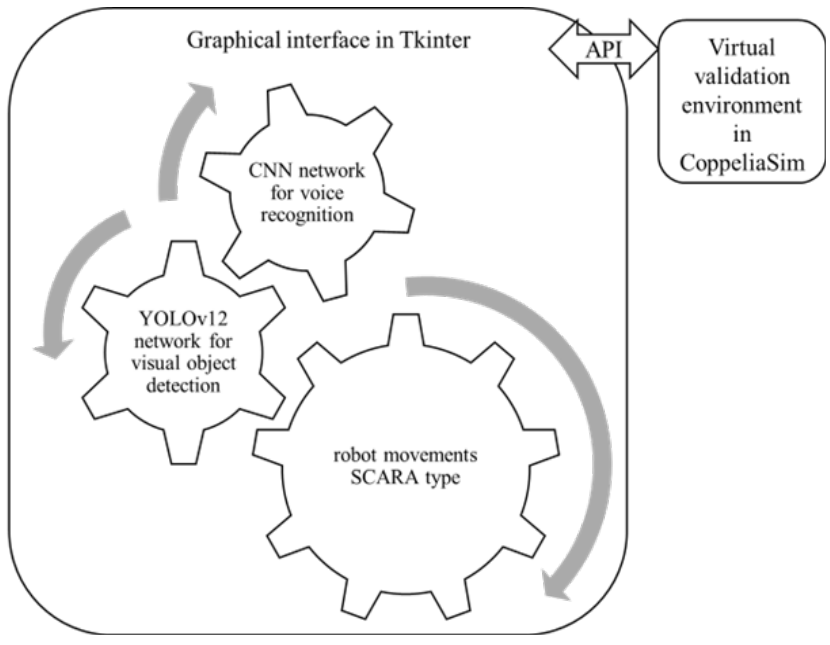
Spanish version

Why was the study conducted?
The contribution to the literature lies in the integration and validation of an intelligent automation system in a virtual environment. Specifically, it provides a methodology that integrates: a proprietary CNN architecture for speech recognition, a YOLOv12 network for machine vision tasks, and proportional-derivative control for a SCARA robot in a system operated via a graphical user interface.

What were the key findings?
An average accuracy of 98.9% in speech recognition of the four defined commands: start, stop, grab, and deliver.
An efficiency exceeding 93% in the identification and location of selected objects.
A system processing throughput of 20 to 25 frames per second.
Convergence of the CNN network model with an inference time of 0.003 seconds per sample, almost instantaneous.

What do these findings contribute?
Improved human-robot interaction thanks to verbal commands that facilitate communication with the system.
Enhanced ability to identify and manipulate objects based on their visual location, allowing the robot to operate autonomously in dynamic environments.
Provides a foundation for developing future applications in complex and dynamic industrial and assistive robotics environments.
Enables the definition of future research in dynamic process management and hardware optimization, given the identified limitations.

Graphical Abstract



doi: 10.25100/iyc.v28i2.15437

Introduction

Voice-controlled robotic systems, based on artificial intelligence algorithms such as convolutional neural networks or LSTM networks (1), are being implemented in various applications across different fields of knowledge. Some examples in the field of robotics include automation systems for gripping control through human-robot collaboration (2), voice-guided gesture and action recognition (3), and voice control of robotic vehicle movement (4).

Furthermore, voice-controlled robotic systems are expanding into areas such as healthcare, through surgical assistance robots (5), and industrial robotics (6). In industry, numerous advances have been achieved thanks to robotic integration (7-9) and, currently, to interaction via voice commands (10).

However, autonomous robotics also requires the use of machine vision systems, which are currently based on artificial intelligence algorithms. One of the main algorithms implemented is the You Only network. Look Once (YOLO) offers a good balance between speed and accuracy (11). Some examples of YOLO's use include detecting tomatoes (12) or apples (13) in smart farming systems (14, 15), detecting vehicles in intelligent driving (16), real-time monitoring (17), document analysis (18), and defect detection (19), among others (20-22). An important aspect of this object identification and location algorithm is the continuous improvement of its performance, which has led to multiple versions (23, 24) that demonstrate improvements in accuracy, bounding box generation, and architecture (25).

These advancements enable the integration of voice-controlled robotic techniques and their automatic activation via machine vision. In this context, this paper presents the design of a pick-and-place system using a voice-controlled robotic environment and machine vision for product selection and delivery. The system employs a user interface that interacts with the virtual environment, contributing to the state of the art in the integration and validation of an intelligent automation system within a virtual environment. This system is easily managed thanks to its graphical user interface. Voice recognition is achieved through a proprietary CNN architecture, the machine vision algorithm focuses on the implementation of version 12 of the YOLO network (the most recent model at the time of system validation), and the SCARA robot control is proportional-derivative.

Materials and methods

To implement advanced voice recognition and machine vision techniques in a robotic system designed for product selection and delivery, a three-phase methodology was defined, distributed in voice recognition, recognition and localization of objects in images using machine vision, and its integration into a simulation environment for robotic drive, as described below.

First, the voice recognition system was developed. A database was built based on recordings of the four commands defined for robotic manipulation: start, stop, grasp, and deliver. This

doi: 10.25100/iyc.v28i2.15437

database was augmented using data augmentation techniques that emulate voice variations. Spectral features were then extracted from the audio files and used as two-dimensional input to train a convolutional neural network with TensorFlow-Keras using a Python script. Once trained, the network was validated and tested by dividing the final database into three subsets: 75% for training, 12.5% for validation, and 12.5% for testing.

Secondly, a machine vision system is implemented for the classification and localization of the five objects selected during development: pizza, fork, apple, scissors, and remote control. This classification and localization is performed using a pre-trained model of the YOLOv12 architecture in Ultralytics , which utilizes publicly available image databases, explained later.

In the third phase, integration is carried out to achieve functional consistency and timely response to voice commands. To this end, the trained YOLOv12 network is integrated with a virtual vision sensor instrumented and integrated into the SCARA robot within the CoppeliaSim virtual environment. Robotic controls are defined and implemented through an interface to synchronize visual object detection with the SCARA robot's movements. The voice recognition, visual detection, and SCARA robot control modules are integrated via the CoppeliaSim remote API, establishing concurrent communication in Python using threads to maintain simultaneous execution and prevent deadlocks.

Finally, the integrated system is validated through experimental tests in a simulated controlled environment to evaluate its performance in voice command recognition, object identification and tracking, and object manipulation by the robot, allowing the calculation of accuracy, latency, and stability metrics, and identifying limitations and future lines of research.

The development was implemented on a laptop computer with hardware specifications based on an AMD Ryzen 7 5825U processor with Radeon G graphics, 40 GB of RAM, and the Linux Mint 22.1 x86_64 Cinnamon 6.4.8 operating system, based on Ubuntu 24.04 LTS. The software used for development was PyCharm 2025.2.0.1 (Community Edition), and the environment for the simulation was CoppeliaSim Edu, version 4.10.0 (rev 0) 64-bit.

Phase 1. Voice recognition system

A voice recognition system based on convolutional neural networks (CNNs) was developed to enable interaction between the user and the robotic arm through voice commands, facilitating communication by recognizing that conversation is the most natural form of language for users. The network was trained to identify the voice commands for robotic action such as "start," "stop," "grab," and "deliver," and subsequently transform them into actions that the robot could execute.

For CNN training, spectrograms are constructed (see some examples in Figure 1). Initially, a Python script establishes the sequence of instructions that allows the recording of audio instructions using Logitech G Pro headphones with a local microphone operating in the frequency range of 100 Hz to 10 kHz. The script automatically generates a folder with the name of each instruction/command and activates audio recording for 1.5 seconds when the keyboard is pressed. Each audio file, in .wav format, is stored in its corresponding folder.

doi: 10.25100/iyc.v28i2.15437

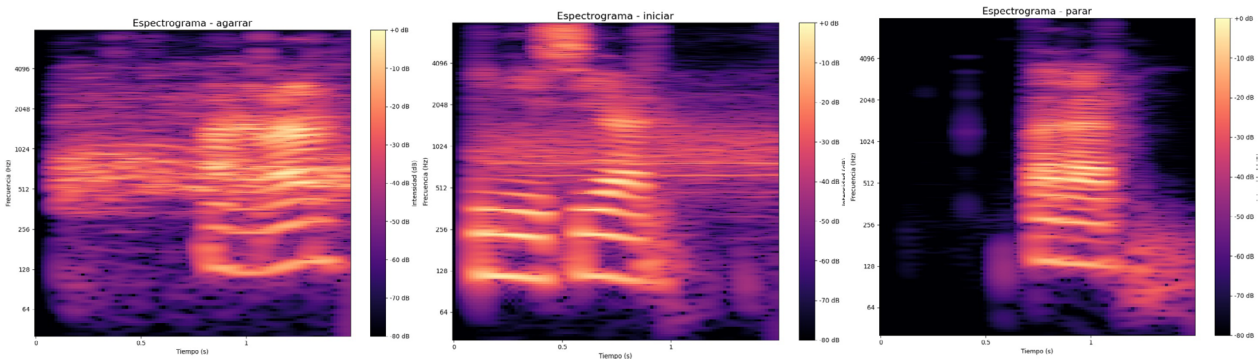


Figure 1. Examples of voice spectrograms used

To increase the size of the database and its generalizability, recordings of artificial voices generated with AI-based speech synthesis technology were included. Additionally, as a data augmentation technique, an algorithm was designed to generate new recordings that maintain the semantic integrity of the verbal instruction by changing parameters such as speed, pitch, and background noise. This emulates natural variations in the human voice associated with different intonations and rhythms, and also balances the number of samples available per instruction. Each audio file in the expanded database is converted to a spectrogram and normalized to identify its spectral characteristics. This two-dimensional representation of energy and frequency (MFCC spectrogram) allows for the training of a convolutional neural network (CNN), as it preserves the spatial correlations of the acoustic signals. The final database contains 1,001 audio files of each type, for a total of 8,008, which were divided into three subsets: 75% for training, 12.5% for validation, and 12.5% for testing.

The CNN architecture is structured as follows:

three convolutional layers with 16, 32 and 64 filters respectively, all with ReLU activation function,

batch normalization layers and dropout (0.2–0.3) to stabilize training and mitigate overfitting,

maxpooling operations and a 2D globalpooling layer for spatial feature reduction, and

a dense intermediate layer of 64 neurons and an output layer with softmax activation, corresponding to the total number of command classes.

The model was compiled using an Adam optimizer with an initial learning rate of 0.001. The loss function was defined using Sparse Categorical Cross-Entropy, suitable for multiclass classification. The model was trained for 60 epochs with a batch size of 32. Control functions were implemented, such as ModelCheckpoint, to automatically save the model with the highest validation accuracy; EarlyStopping, to stop training in case of stagnation; and ReduceLROnPlateau, to decrease the learning rate in case of overfitting.

doi: 10.25100/iyc.v28i2.15437

Phase 2. Machine vision system

To equip the robot with visual perception capabilities, an object detection system using transfer learning based on the YOLOv12 model was implemented within the Ultralytics environment. This system enables the real-time identification and localization of objects in the virtual simulation environment and provides the necessary information for the movements of the SCARA robot, in relation to the object's spatial location in the scene.

The YOLOv12 network used is pre-trained with the COCO (Common Objects in Context) dataset, which has 80 types of objects in everyday scenes, and includes weights optimized for multi-class detection, so no further training is required, which speeds up implementation.

From the COCO dataset, five types of objects with different characteristics were chosen for robotic interaction. The chosen objects are:

Pizza (3319 images)

Fork (3710 images)

Apple (1662 images)

Remote control (3221 images)

Scissors (975 images)

Each image captured by the CoppeliaSim sensor The VisionSensor data is processed using the OpenCV library, resized to 640×640 pixels, and sent to the YOLOv12 model for inference. During post-processing, confidence thresholds of 0.35 and non-maximum suppression (NMS) were applied to exclude redundant detections and ensure the stability of the results presented in the interface. The network operates in its default inference mode, with the default hyperparameters in the Ultralytics framework, corresponding to a batch size of 16, an automatic optimizer, and data augmentation with rotations, brightness variations, scaling, and horizontal flipping.

Phase 3. Functional integration of systems in Python and CoppeliaSim

The CNN-based voice recognition system interprets verbal instructions from a Python user interface, which in turn sends robotic control signals to CoppeliaSim . The real-time YOLOv12 visual detection and tracking system processes images captured by the sensor integrated into the SCARA robot and guides the movements of its three main joints and end effector according to the object's position. These systems are integrated via CoppeliaSim 's ZMQ remote API, which enables direct communication between the simulation environment and the Python-programmed instruction sequences. This integration is synchronized using parallel processes (threads) to ensure smooth and stable communication, preventing execution delays.

Each verbal instruction activates one of the following specific functions:

"Iniciar": activates the controls and other voice commands

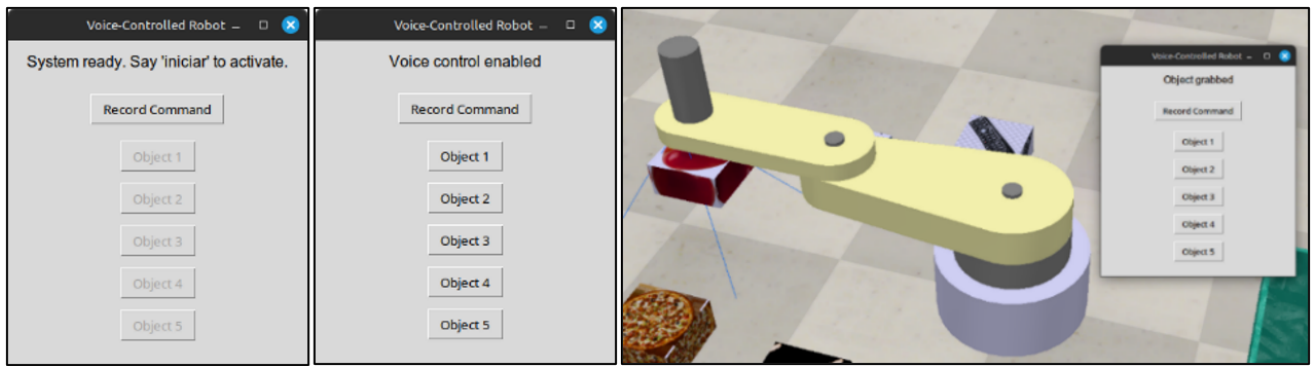
"Parar": disables the controls and prevents any action

"Agarar": executes the descent routine, activating the suction system and raising the end effector

doi: 10.25100/iyc.v28i2.15437

"Entregar": moves the arm to delivery position, deactivates the grip, and returns the robot to its initial position

Figure 2 shows the designed graphical interface, which includes capturing the user's voice (Figure 2a), displaying the detected verbal instruction (Figure 2b), and activating the buttons for object selection (Figure 2c). A modular design was used for the system, allowing voice recognition to be kept on a separate thread, thus ensuring the simultaneous execution of both the simulation and the recognition of verbal instructions.



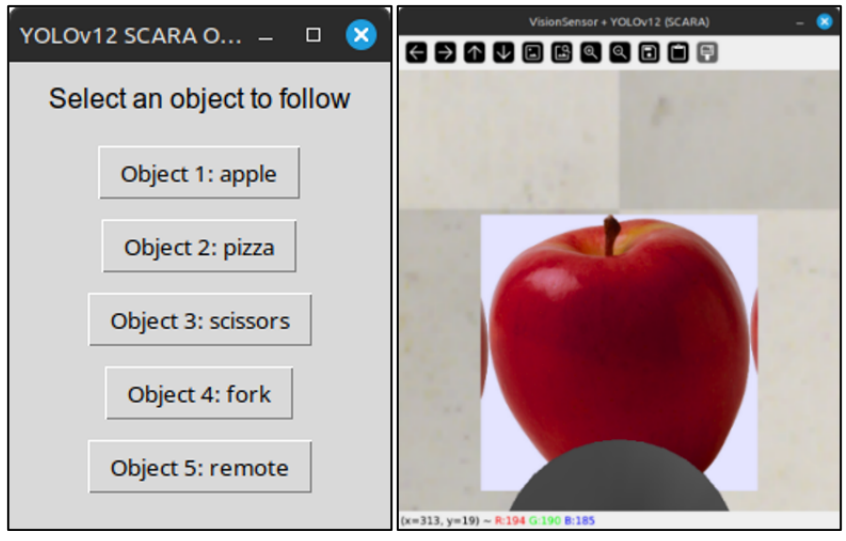
a) Demo start b) Control activated c) Object capture

Figure 2. Demo of speech recognition system integration (CNN **)**

The machine vision system allows the robot to actively track a user-selected object in the field of view of the virtual capture sensor. The YOLOv12 model produces the bounding box coordinates where the object is located, the labels that identify it, and the confidence levels for each detection. Based on the object coordinates obtained from the bounding boxes, a proportional-derivative (PD) controller was developed to centralize the object's gripping point. Its output allows the SCARA robot's movement angles to be adjusted according to the input based on the horizontal error of the object's detection area. The controller is implemented using computational self-tuning, achieving a kp=0.0001 and a kd=0.00004. This method allows the object to remain centered in the image and the apparent distance to be adjusted using the bounding box area ratio. When the system does not detect the object, a search mode is activated that moves the two main axes alternately, generating a ±160° sweep to locate the target.

The application features a user interface (Figure 3) that displays five buttons, each associated with a different object class in the established COCO set (apple, pizza, scissors, fork, or remote control) (Figure 3a). Selecting one of these classes automatically initiates the real-time detection and tracking process, as shown in Figure 3b .

doi: 10.25100/iyc.v28i2.15437



a) Start b) View of the object

Figure 3. Demonstration of vision and tracking system integration (**YOLOv12)**

Once all components have been individually tested, a final script is created that unifies the three subsystems (speech recognition using a custom CNN, YOLOv12 visual detection and tracking, and SCARA robot control). The application is based on a concurrent thread architecture, with a main thread for managing the graphical user interface (GUI), a dedicated thread for the vision loop responsible for capture, inference, and tracking, and additional threads for audio processing and executing specific actions (e.g., grasping or handing). This independence allows the interface to function while the simulation is running, keeping the view stream active.

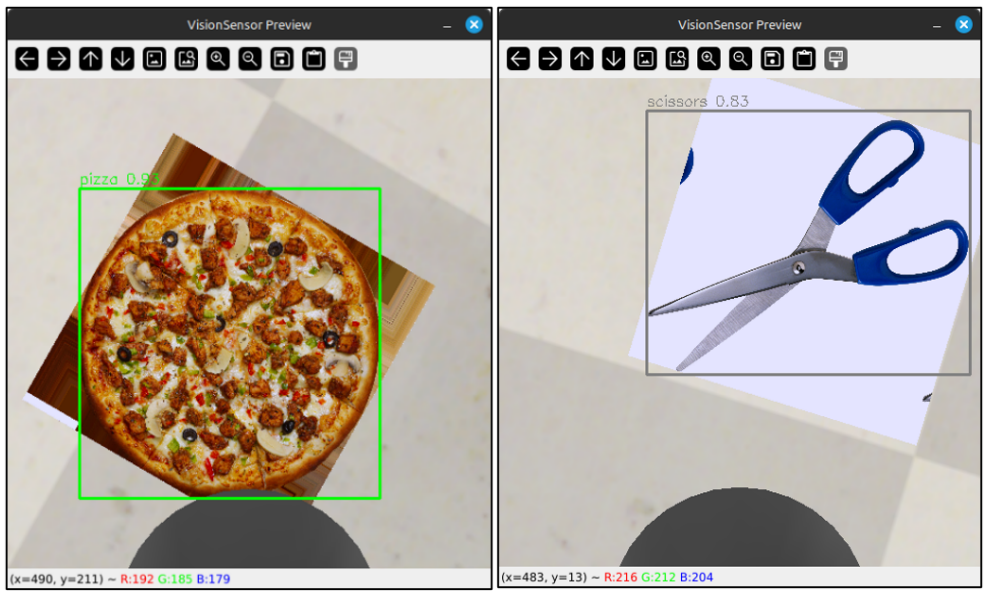
As a primary strategy to maintain consistency between the vision system and the robot's movement, three layers (vision indicator, action blocking, and safe wrappers) are implemented to prevent routines involving movement from interfering with the vision sensor reading or the processing of the YOLO v12 network. First, a vision indicator (vision_enabled) is used, which is activated while critical routines (grasping, delivery, or returning to the initial position) are not being executed. This keeps the YOLO network in a waiting state while vision is deactivated, preventing incomplete readings or parallel conflicts.

Action locking ensures that each critical movement is performed in a safe environment, working exclusively on the execution and preventing interference with the robot's state. Safe wrappers for API reads and writes acquire global locking, reducing failures and exceptions resulting from parallel calls. The function for moving the robot's joints executes in small steps with limited speed and number of iterations, so movements are not abrupt and blocking is avoided due to ambitious goals, respecting the angular limits of each joint.

To draw bounding boxes around objects of interest, a single thread runs on a local frame, preventing concurrent access. Additionally, filtering is applied based on relevant classes, and a confidence threshold of 0.35 is used to reduce false positives. The box is displayed in green when it matches the selected object (Figure 4a) and in gray when it does not (Figure 4b). Once the

doi: 10.25100/iyc.v28i2.15437

target is detected with the required confidence, the PD handler is activated, and the processing of further detections in that frame is interrupted, preventing interference with the tracking logic. The drawing is performed exclusively within the OpenCV window, without modifying CoppeliaSim 's internal rendering , thus reducing the likelihood of graphical artifacts.



a) Selected object b) Unselected object

Figure 4. Bounding boxes

A synchronization method is implemented with voice logic and a graphical interface using control variables that enable or disable buttons depending on the robot's state (Figure 5).

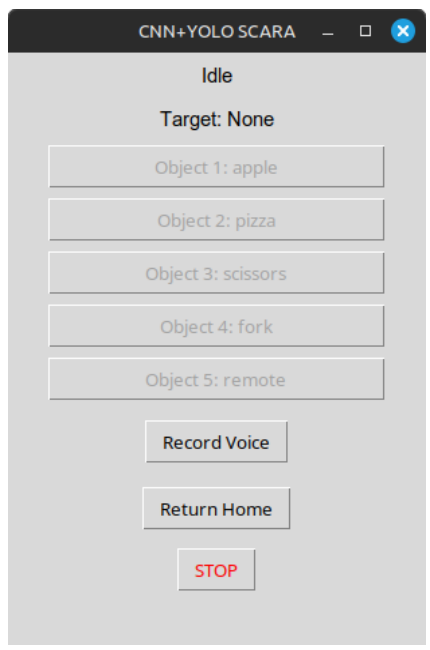
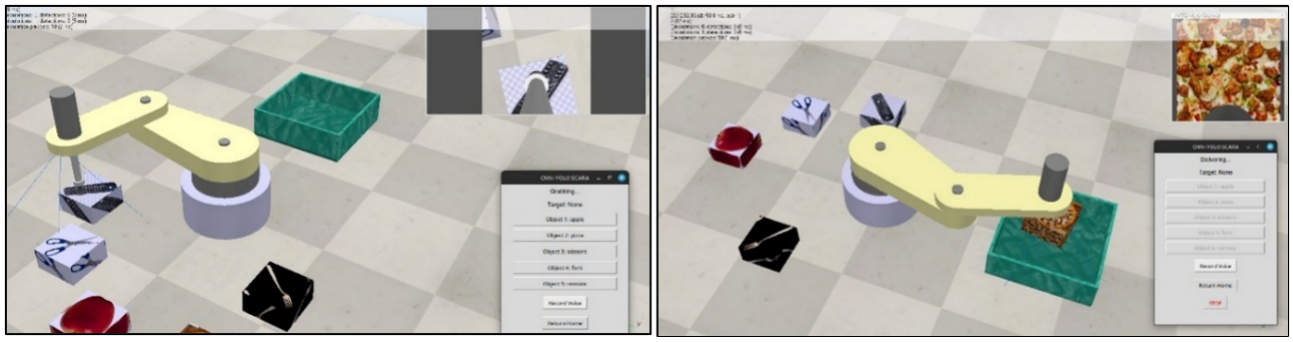


Figure 5. Graphical user interface

doi: 10.25100/iyc.v28i2.15437

Voice command processing is performed on a separate thread which, after predicting the command with the CNN, updates the control variables and executes the corresponding routines (Figure 6), which can be for gripping (Figure 6a) at the location point of each object or for delivery (Figure 6b) at the central point of the destination box in green.



a) grip routine b) delivery routine

Figure 6. Routines

Results

Performance of the voice recognition system

The voice command classification model, trained using the convolutional neural network, showed stable and rapid convergence during training. The optimal epoch was reached at iteration 36, with a maximum validation accuracy of 0.9940 and a minimum loss of 0.0180. The entire training process took 274.83 s. In tests performed with the evaluation set (1001 samples), the model achieved an overall accuracy of 98.9% and a weighted F1 score of 0.9890, surpassing the average accuracy of 97.64% of the deep bidirectional short-term memory model with controlled recurrent unit (BiLSTM -deep GRU) (26). The average inference time was 0.0003 s per sample, enabling virtually instantaneous responses during embedded system execution.

The results from the speech recognition system using the test set data allowed for the construction of the confusion matrix shown in Figure 7, which highlights an almost perfect classification for the four classes of verbal instructions. There were few confusions between the commands “agarrar” and “parar” (1.6%), “parar” and “entregar” (1.2%), and “agarrar” and “entregar” (0.8%) and “entregar” and “parar” (0.8%), which may stem from phonetic similarities between the words, especially in their endings, and from early recording synchronization issues during the audio delivery.

doi: 10.25100/iyc.v28i2.15437

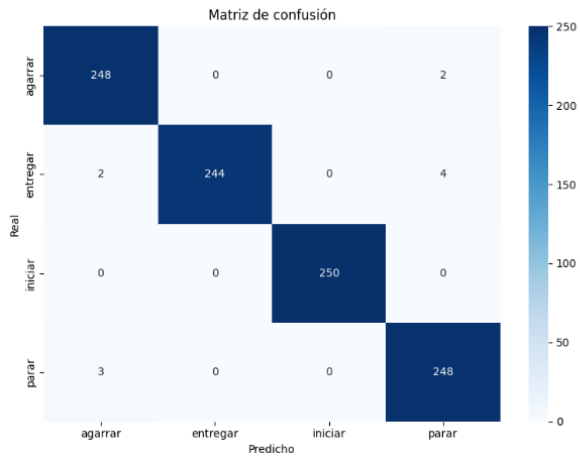


Figure 7. Confusion matrix of the CNN network used for speech recognition

The results demonstrate the high reliability of the CNN designed for speech recognition, specifically for the four defined spoken commands, in low-noise environments. Clearly, there will be a bias towards any word outside the untrained context that randomly triggers the robot's movement. Therefore, the commands for operating the robot must be clearly defined, or, as future work, an additional "unidentified" class should be included.

Performance of the machine vision system

During testing, the YOLOv12-based visual detection system exhibited excellent stability. In real time, it accurately identified all objects in the virtual environment simulated with CoppeliaSim, with a processing rate ranging from 20 to 25 frames per second (FPS). The VisionSensor 's refresh rate remained unchanged during integration with the virtual environment and was unaffected by the YOLOv12 network's operation, including the use of concurrent thread control and the sensor's suspension while the robot executed critical routines.

The tracking control implemented using a PD (Proportional-Derivative) controller successfully kept the selected object within the central area of the field of view with errors of less than 7%, corresponding to a maximum of ±25 px . This was achieved by adjusting the angular position of the robot's links in real time. This error affects the object's gripping point, which in extreme cases ends up near the edges. However, the PD control also reduced the speed of the search for the object's center, leading to manual adjustment until a value was found that did not restrict movement and was effective in preventing oscillations and maintaining grip when moving away from the object's center (maximum error).

Performance of the integrated system in a simulated environment

The simulation environment under CoppeliaSim features an integrated physics engine that provides a more detailed representation of the behavior of real-world objects. This simulation environment (Figure 8) consists of the following elements:

Robotic model: The MTB robot was used, a simplified but functional SCARA robot, which has two rotating joints, one prismatic joint, and an end effector with an integrated suction system

doi: 10.25100/iyc.v28i2.15437

(suctionPad). During the simulation, the robot's base did not move, the rotating joints were limited to specific angles, and the suctionPad was controlled via Python to avoid conflicts with the arm joints

Vision sensor: The MTB robot does not have a vision system, so a vision sensor (visionSensor) was integrated into the second link of the arm, which was configured to detect objects using the YOLO v12 network, allowing them to be tracked

Scene objects: The virtual environment contained five cuboids positioned in front of the robot and a storage container. Each cuboid has a different texture and color, simulating objects with different visual characteristics.

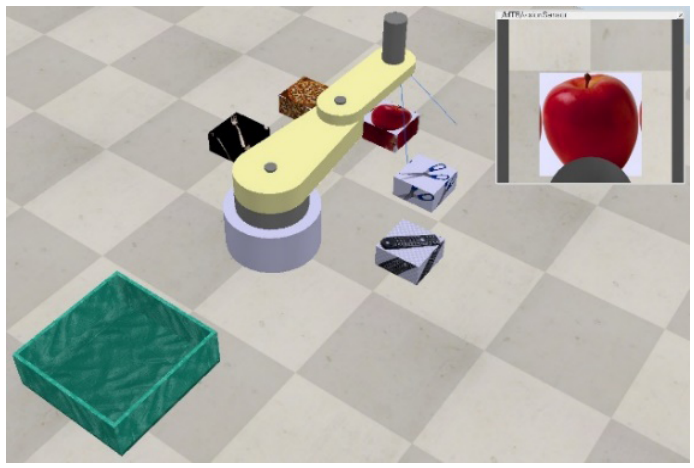


Figure 8. Virtual simulation environment in CoppeliaSim

Ten object grasping tests were performed on the integrated system, revealing errors in the voice command or in the object grasp itself, as tabulated in Table 1 under the success/failure ratio. The main grasping problems occurred with the extreme objects, "fork" and "remote control," due to the robot's limited angular movement within the work area. Errors with the central objects were primarily caused by the object falling during the delivery process when the gripping device moved away from the object's center. The voice command exhibited errors in detection and class confusion, with the most critical issue occurring in the composition of the "remote control" label.

Table 1. Success/failure ratio of the tests.

Object	Voice command	Grip
Pizza	10/0	9/1
Fork	9/1	8/2
Apple	9/1	10/0
Scissors	8/2	10/0
Remote control	7/3 9/1	

doi: 10.25100/iyc.v28i2.15437

Discussion

During operation, the entire system exhibited consistent performance and complete coordination among its components. Thanks to action blocking strategies and control flags, conflicts between the voice recognition system, the visual recognition system, and the robotic arm's motion control were avoided. Temporarily suspending the visual recognition system during the execution of critical grasping and delivery routines prevents parallel reading of the VisionSensor , thus eliminating synchronization issues and overloading of the CoppeliaSim API .

The evaluation of the integrated system's performance in the simulated virtual environment identified three limitations. First, during the execution of the YOLOv12 network on the CPU, latency was observed in the inference rate when multiple objects were presented or complex backgrounds were encountered, resulting in a reduction of the refresh rate to approximately 15 FPS. Second, slow execution was observed in the robotic arm's movements due to the control that limits the maximum angular velocity, aiming for smooth movements. Finally, during extended tests, high memory consumption was observed when the graphical interface and the active threads for visual and speech recognition were simultaneously enabled. This could be overcome through dynamic process management, and is therefore proposed as a topic for future research.

However, the complete system managed to maintain both functional coherence between the voice recognition system, the visual recognition system and the robotic control, as well as stable and autonomous manipulation of the SCARA robotic arm.

Conclusions

Based on the results obtained during validation in a simulated environment, it can be concluded that the implementation of advanced speech recognition and computer vision techniques in a robotic system is not only feasible but also successful. The integration achieved represents a significant advancement that enhances human-robot interaction by responding to verbal commands and fostering robotic autonomy in dynamic environments by identifying and manipulating objects based on their location using machine vision.

Future research proposes optimizing memory consumption and the use of limited resources to enable applications of the complete system in industrial and assistive robotics, which are typically dynamic and complex environments. Additionally, a voice training system will be established that allows the user to pronounce words without affecting the dataset trained on the robotic drive, as previously mentioned.

Acknowledgments

Product derived from the research project "Fortalecimiento de los procesos de recepción de pedidos y control de inventario de materias primas con el apoyo de la Industria 4.0" INV-ING-4150 financed by the Vice-Rectorate of Research of the Universidad Militar Nueva Granada,

year 2025. The authors, in their capacity as associate professors, thank the Universidad Militar Nueva Granada for funding the project and the time allocated for its development.

CrediT authorship contribution statement

Conceptualization - Ideas: Robinson Jiménez Moreno. Data curation: José Miguel Bejarano Restrepo. Formal analysis: José Miguel Bejarano Restrepo, Robinson Jiménez Moreno. Investigation: José Miguel Bejarano Restrepo, Robinson Jiménez Moreno. Methodology: Robinson Jiménez Moreno. Project Management: Anny Astrid Espitia Cubillos. Resources: Anny Astrid Espitia Cubillos. Software: José Miguel Bejarano Restrepo, Robinson Jiménez Moreno. Supervision: Anny Astrid Espitia Cubillos. Validation: José Miguel Bejarano Restrepo, Robinson Jiménez Moreno. Writing - original draft - Preparation: Robinson Jiménez Moreno. Writing - revision and editing -Preparation: Anny Astrid Espitia Cubillos.

Financing: does not declare.

Conflict of interest: does not declare. Ethical aspect: does not declare.

References

1. Bakouri M. Development of Voice Control Algorithm for Robotic Wheelchair Using MIN and LSTM Models. Computers, Materials and Continua, 2022; 73(2):2441-2456,

https://doi.org/10.32604/cmc.2022.025106

2. Behera A. K., Mohanraj R. y Prem A. Smart Autonomous Wireless Gripper for Human-Robot Collaborative Operation. Procedia CIRP, 2025; 136:880-885,

https://doi.org/10.1016/j.procir.2025.08.150

3. Meghana M., Kumari C. U., Priya J. S., Mrinal P., Sai K. A. V., Reddy S. P., Vikranth K., Kumar T. S. y Panigrahy A. K. Hand gesture recognition and voice controlled robot. Materials Today: Proceedings, 2020; 33(7):4121-4123,

https://doi.org/10.1016/j.matpr.2020.06.553

4. Saradi V. P. y Kailasapathi P. Voice-based motion control of a robotic vehicle through visible light communication. Computers & Electrical Engineering, 2019; 76:154-167,

https://doi.org/10.1016/j.compeleceng.2019.03.011

5. Chakradeo V. K., Malhotra K., Lee M. R. y Nathan C. A. O. Navigational Surgery with Voice: Controlled Robotic Assist for Endoscopic Approach to the Pituitary. Otolaryngology - Head and Neck Surgery, 2005; 133(2-Supplement):P153,

https://doi.org/10.1016/j.otohns.2005.05.344

6. Feng Y., He J., Luo J., Fang Z., Zhang C. y Yang G. A vision-based self-calibration method for industrial robots using variable pose constraints. Robotics and Computer-Integrated Manufacturing, 2026; 98:103142,

https://doi.org/10.1016/j.rcim.2025.103142

7. Lenţoiu I., Răileanu S., Borangiu T., Constantinescu M. y Morariu O. Instantaneous power prediction for industrial robots using tree-based machine learning methods. Engineering Applications of Artificial Intelligence, 2025; 162(Part A):112339,

https://doi.org/10.1016/j.engappai.2025.112339

8. Zhou Q., Gu Y., Li J., Feng B., Li B. y Bi Y. Towards zero-shot robot tool manipulation in industrial context: A modular VLM framework enhanced by multimodal affordance representation. Robotics and Computer-Integrated Manufacturing, 2026; 98:103161,

https://doi.org/10.1016/j.rcim.2025.103161

9. Rogowski, A. Web-based remote voice control of robotized cells. Robotics and Computer-Integrated Manufacturing, 2013; 29(4):77-89,

https://doi.org/10.1016/j.rcim.2012.11.002

10. Yu Z., Zhang P. y Shi J. Transformation of industrial robotics with natural language models: Recent progress and future prospects. Robotics and Computer-Integrated Manufacturing, 2026; 97:103113,

https://doi.org/10.1016/j.rcim.2025.103113

11. Ali M. L. y Zhang Z. The YOLO Framework: A Comprehensive Review of Evolution, Applications, and Benchmarks in Object Detection. Computers, 2024; 13(12):336,

https://doi.org/10.3390/computers13120336

12. Deng X., Huang T., Wang W. y Feng W. SE-YOLO: A sobel-enhanced framework for high-accuracy, lightweight real-time tomato detection with edge deployment capability. Computers and Electronics in Agriculture, 2025; 239(Part B):110973,

https://doi.org/10.1016/j.compag.2025.110973

13. Ma B., Sun L., Mu J., Ren Z., Kang G., Liu R., Liu S., Hu X., Zhang H. y Wang J. MH-YOLO: Multiple heterogeneous YOLO for apple orchard pest detection. Information Processing in Agriculture; 2025,

https://doi.org/10.1016/j.inpa.2025.08.001

14. Gong Y., Zhang G., Wang C. y Xiao D. CD-ViT-YOLO: A lightweight Hybrid ViT-YOLO model for caged duck behaviour recognition under varying lighting conditions. Smart Agricultural Technology, 2025; 12:101414,

https://doi.org/10.1016/j.atech.2025.101414

15. Weng W., Lai Z., Cui Z., Chen Z., Chen H., Lin T., Wang J., Zheng S. y Chen G. GCD-YOLO: A deep learning network for accurate tomato fruit stalks identification in unstructured environments. Smart Agricultural Technology, 2025; 12:101465,

https://doi.org/10.1016/j.atech.2025.101465

16. Zhou W., Wang J., Meng X., Wang J., Song Y. y Liu Z. MP-YOLO: multidimensional feature fusion based layer adaptive pruning YOLO for dense vehicle object detection algorithm. Journal of Visual Communication and Image Representation, 2025; 112:104560,

https://doi.org/10.1016/j.jvcir.2025.104560

17. Tiruvikraman V., Selvakumar D. y Dijayakumar P. Real-Time Smart Surveillance and Enforcement for Dust throw Detection and Identity Recognition Using YOLO 12 and SA - FaceXNet. SIViP, 2025; 19:983,

https://doi.org/10.1007/s11760-025-04582-x

18. Ma W., Cao M., Ma J., Dong Z., Yang C. y Li Z. Mamba-YOLO: Multi-level adaptive rectangular convolution for Document Layout Analysis. Pattern Recognition, 2026; 170:112031,

https://doi.org/10.1016/j.patcog.2025.112031

19. Ghahremani A., Adams S. D., Norton M., Khoo S. Y. y Kouzani A. Z. Detecting Defects in Solar Panels Using the YOLO v10 and v11 Algorithms. Electronics, 2025; 14(2):344,

https://doi.org/10.3390/electronics14020344

20. Chen M., Tian J., Cao X., Fu Z. y Zhang D. DM-YOLO for MLCCs' automatic defect detection. Optics & Laser Technology, 2025; 192(Part E):113977,

https://doi.org/10.1016/j.optlastec.2025.113977

21. Kumar A., Dhanalakshmi R., Rajesh R. y Sendhil R. A spatial features and weight adjusted loss infused Tiny YOLO for shadow detection. Signal Processing: Image Communication, 2026; 140:117408,

https://doi.org/10.1016/j.image.2025.117408

22. Lv L., Li J. y Zhao Y. DMP-YOLO: Dense multi-scale perception for complex scenes YOLO algorithm Prunus humilis small target detection. Smart Agricultural Technology, 2025; 12:101461

https://doi.org/10.1016/j.atech.2025.101461

23. Zhang Y., Xu Y., Xu T., Wang C., Li C. y Wang H. RSD-YOLO: An improved YOLOv7-tiny framework for oat disease severity identification with integration of ReXNet and decoupled head. Smart Agricultural Technology, 2025; 12:101433

https://doi.org/10.1016/j.atech.2025.101433

24. Yao J., Li Y., Xia Z., Nie P., Li X. y Li Z. WTAD-YOLO: A lightweight tomato leaf disease detection model based on YOLO11. Smart Agricultural Technology, 2025; 12:101349,

https://doi.org/10.1016/j.atech.2025.101349

25. Murat A. A. y Kiran M. S. A comprehensive review on YOLO versions for object detection. Engineering Science and Technology, an International Journal, 2025; 70:102161,

https://doi.org/10.1016/j.jestch.2025.102161

26. Mehra, S., Ranga, V. y Agarwal, R. A deep learning approach to dysarthric utterance classification with BiLSTM-GRU, speech cue filtering, and log mel spectrograms. The Journal of Supercomputing, 2024; 80:14520-14547,

https://doi.org/10.1007/s11227-024-06015-x

doi: 10.25100/iyc.v28i2.15437