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

Vol 28 (2)

ISSN
0123-3033
e- 2027-8284

Artículo Original

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

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

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

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

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 Coppelia-Sim, 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.

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.

¿Cómo citar?

Bejarano JM, Jiménez R, Espitia AA. Implementación de técnicas avanzadas de reconocimiento de voz y visión computacional en un sistema robótico. Ingeniería y Competitividad, 2026, 28(2) e-20215437

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

Recibido: 20/11/25
Revisado: 19/01/26
Aceptado: 15/04/26
Online: 15/05/26

Correspondencia

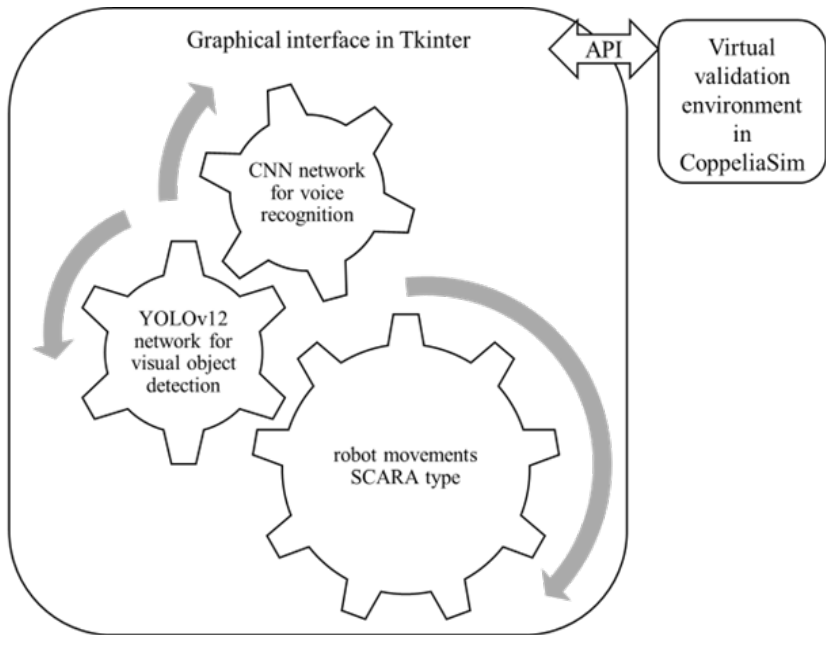
anny.espitia@unimilitar.edu.co

¿Por qué se realizó el estudio?
La contribución a la literatura radica en la integración y validación de un sistema de automatización inteligente en un entorno virtual. Específicamente, proporciona una metodología que integra: una arquitectura CNN propia para el reconocimiento de voz, una red YOLOv12 para tareas de visión artificial y control proporcional-derivativo para un robot SCARA en un sistema operado mediante una interfaz gráfica de usuario.

¿Cuáles fueron los hallazgos clave?
Una precisión promedio del 98,9 % en el reconocimiento de voz de los cuatro comandos definidos: iniciar, detener, agarrar y entregar. Una eficiencia superior al 93 % en la identificación y localización de objetos seleccionados. Un rendimiento del sistema de procesamiento de 20 a 25 fotogramas por segundo. Convergencia del modelo de red CNN con un tiempo de inferencia de 0,003 segundos por muestra, prácticamente instantáneo.

¿Qué aportan estos hallazgos?
Una mejor interacción humano-robot gracias a los comandos verbales que facilitan la comunicación con el sistema. Mayor capacidad para identificar y manipular objetos según su ubicación visual, lo que permite al robot operar de forma autónoma en entornos dinámicos. Sienta las bases para el desarrollo de futuras aplicaciones en entornos robóticos industriales y de asistencia complejos y dinámicos. Permite definir futuras líneas de investigación en gestión de procesos dinámicos y optimización de hardware, dadas las limitaciones identificadas.

Graphical Abstract



doi: 10.25100/iyc.v28i2.15437

Introducción

Los sistemas robóticos controlados por voz, basados en algoritmos de inteligencia artificial como redes neuronales convolucionales o redes LSTM (1), se están implementando en diversas aplicaciones de diferentes campos de conocimiento. Algunos ejemplos en el campo de la robótica incluyen sistemas de automatización para el control de agarre mediante la colaboración humano-robot (2), reconocimiento de gestos y acciones guiados por voz (3), y control por voz del movimiento de vehículos robóticos (4).

Además, los sistemas robóticos controlados por voz se están expandiendo a áreas como la sanidad, mediante robots de asistencia quirúrgica (5), y la robótica industrial (6). En la industria, se han logrado numerosos avances gracias a la integración robótica (7-9) y, actualmente, a la interacción mediante comandos de voz (10).

Sin embargo, la robótica autónoma también requiere el uso de sistemas de visión artificial, que actualmente se basan en algoritmos de inteligencia artificial. Uno de los principales algoritmos implementados es la red Solo Tú. Look Once (YOLO) ofrece un buen equilibrio entre velocidad y precisión (11). Algunos ejemplos del uso de YOLO incluyen la detección de tomates (12) o manzanas (13) en sistemas de agricultura inteligente (14, 15), detección de vehículos en conducción inteligente (16), monitorización en tiempo real (17), análisis de documentos (18) y detección de defectos (19), entre otros (20-22). Un aspecto importante de este algoritmo de identificación y localización de objetos es la mejora continua de su rendimiento, lo que ha dado lugar a múltiples versiones (23, 24) que demuestran mejoras en precisión, generación de cajas delimitadoras y arquitectura (25).

Estos avances permiten la integración de técnicas robóticas controladas por voz y su activación automática mediante visión artificial. En este contexto, este artículo presenta el diseño de un sistema de selección y colocación utilizando un entorno robótico controlado por voz y visión artificial para la selección y entrega del producto. El sistema emplea una interfaz de usuario que interactúa con el entorno virtual, contribuyendo al estado de la tecnología en la integración y validación de un sistema de automatización inteligente dentro de un entorno virtual. Este sistema es fácilmente gestionable gracias a su interfaz gráfica de usuario. El reconocimiento de voz se logra mediante una arquitectura CNN propietaria, el algoritmo de visión artificial se centra en la implementación de la versión 12 de la red YOLO (el modelo más reciente en el momento de la validación del sistema), y el control robot SCARA es derivado proporcional.

Materiales y métodos

Para implementar técnicas avanzadas de reconocimiento de voz y visión artificial en un sistema robótico diseñado para la selección y entrega de productos, se definió una metodología trifásica, distribuida en reconocimiento de voz, reconocimiento y localización de objetos en imágenes mediante visión artificial, y su integración en un entorno de simulación para la propulsión robótica, como se describe a continuación.

doi: 10.25100/iyc.v28i2.15437

Primero, se desarrolló el sistema de reconocimiento de voz. Se construyó una base de datos basada en grabaciones de los cuatro comandos definidos para la manipulación robótica: inicio, detención, agarre y entrega. Esta base de datos se amplió utilizando técnicas de aumento de datos que emulan variaciones de voz. Las características espectrales se extrajeron de los archivos de audio y se usaron como entrada bidimensional para entrenar una red neuronal convolucional con TensorFlow-Keras usando un script en Python. Una vez entrenada, la red se validaba y probaba dividiendo la base de datos final en tres subconjuntos: 75% para entrenamiento, 12,5% para validación y 12,5% para pruebas.

En segundo lugar, se implementa un sistema de visión artificial para la clasificación y localización de los cinco objetos seleccionados durante el desarrollo: pizza, tenedor, manzana, tijeras y control remoto. Esta clasificación y localización se realiza utilizando un modelo preentrenado de la arquitectura YOLOv12 en Ultralytics, que utiliza bases de datos de imágenes públicas, como se explica más adelante.

En la tercera fase, se realiza la integración para lograr la consistencia funcional y una respuesta oportuna a los comandos de voz. Para ello, la red entrenada YOLOv12 está integrada con un sensor de visión virtual instrumentado e integrado en el robot SCARA dentro del entorno virtual CoppeliaSim. Los controles robóticos se definen e implementan a través de una interfaz para sincronizar la detección visual de objetos con los movimientos del robot SCARA. Los módulos de reconocimiento de voz, detección visual y control de robots SCARA están integrados a través de la API remota de CoppeliaSim, estableciendo comunicación concurrente en Python usando hilos para mantener la ejecución simultánea y evitar bloqueos.

Finalmente, el sistema integrado se valida mediante pruebas experimentales en un entorno controlado simulado para evaluar su rendimiento en reconocimiento de comandos de voz, identificación y seguimiento de objetos, y manipulación de objetos por parte del robot, permitiendo calcular métricas de precisión, latencia y estabilidad, así como la identificación de limitaciones y futuras líneas de investigación.

El desarrollo se implementó en un ordenador portátil con especificaciones de hardware basadas en un procesador AMD Ryzen 7 5825U con gráficos Radeon G, 40 GB de RAM y el sistema operativo Linux Mint 22.1 x86_64 Cinnamon 6.4.8, basado en Ubuntu 24.04 LTS. El software utilizado para el desarrollo fue PyCharm 2025.2.0.1 (Edición Comunitaria), y el entorno para la simulación fue CoppeliaSim Edu, versión 4.10.0 (rev 0) de 64 bits.

Fase 1. Sistema de reconocimiento de voz

Se desarrolló un sistema de reconocimiento de voz basado en redes neuronales convolucionales (CNN) para permitir la interacción entre el usuario y el brazo robótico mediante comandos de voz, facilitando la comunicación al reconocer que la conversación es la forma de lenguaje más natural para los usuarios. La red fue entrenada para identificar los comandos de voz para acciones robóticas

doi: 10.25100/iyc.v28i2.15437

como "iniciar", "parar", "agarrar" y "entregar", y posteriormente transformarlos en acciones que el robot pudiera ejecutar.

Para el entrenamiento de CNN, se construyen espectrogramas (véanse algunos ejemplos en la Figura 1). Inicialmente, un script Python establece la secuencia de instrucciones que permite grabar instrucciones de audio usando auriculares Logitech G Pro con un micrófono local que opera en el rango de frecuencias de 100 Hz a 10 kHz. El script genera automáticamente una carpeta con el nombre de cada instrucción/comando y activa la grabación de audio durante 1,5 segundos al pulsar el teclado. Cada archivo de audio, en .wav formato, se almacena en su carpeta correspondiente.

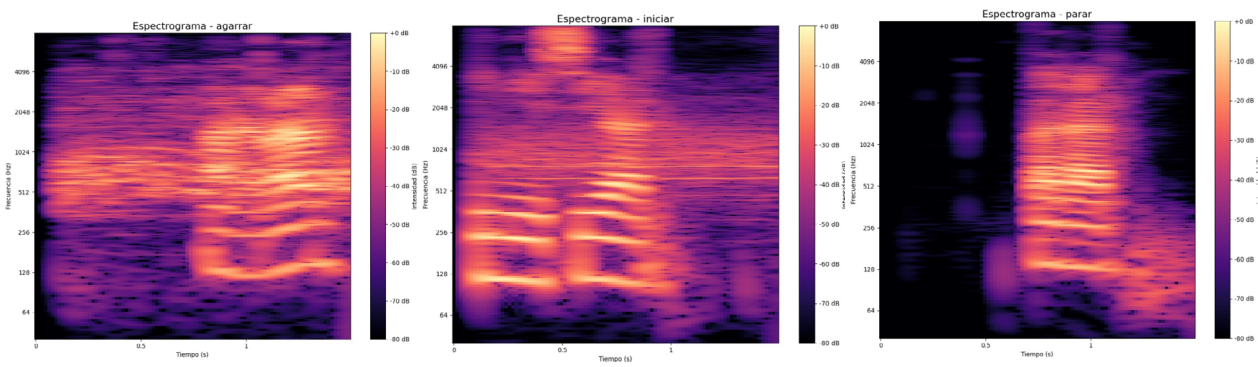


Figura 1. Ejemplos de espectrogramas de voz utilizados

Para aumentar el tamaño de la base de datos y su generalizabilidad, se incluyeron grabaciones de voces artificiales generadas con tecnología de síntesis de voz basada en IA. Además, como técnica de aumento de datos, se diseñó un algoritmo para generar nuevas grabaciones que mantuvieran la integridad semántica de la instrucción verbal cambiando parámetros como velocidad, tono y ruido de fondo. Esto emula variaciones naturales en la voz humana asociadas a diferentes entonaciones y ritmos, y también equilibra el número de muestras disponibles por instrucción. Cada archivo de audio en la base de datos ampliada se convierte en un espectrograma y se normaliza para identificar sus características espectrales. Esta representación bidimensional de energía y frecuencia (espectrograma MFCC) permite el entrenamiento de una red neuronal convolucional (CNN), ya que preserva las correlaciones espaciales de las señales acústicas. La base de datos final contiene 1.001 archivos de audio de cada tipo, para un total de 8.008, que se dividieron en tres subgrupos: 75% para entrenamiento, 12,5% para validación y 12,5% para pruebas.

La arquitectura CNN está estructurada de la siguiente manera:

tres capas convolucionales con 16, 32 y 64 filtros respectivamente, todas con función de activación de ReLU, capas de normalización por lotes y dropout (0.2–0.3) para estabilizar el entrenamiento y mitigar el sobreajuste, operaciones de maxpooling y una capa globalpooling 2D para la reducción de características espaciales, y una capa intermedia densa de 64 neuronas y una capa de salida con activación softmax, correspondiente al número total de clases de comando.

El modelo se compiló usando un optimizador Adam con una tasa de aprendizaje inicial de 0,001. La función de pérdida se definió usando entropía cruzada categórica dispersa, adecuada para la clasificación multiclase. El modelo fue entrenado durante 60 épocas con un tamaño de lote de 32.

doi: 10.25100/iyc.v28i2.15437

Se implementaron funciones de control, como ModelCheckpoint, para guardar automáticamente el modelo con la máxima precisión de validación; EarlyStopping, para detener el entrenamiento en caso de estancamiento; y ReduceLROnPlateau, para disminuir la tasa de aprendizaje en caso de sobreajuste.

Fase 2. Sistema de visión artificial

Para dotar al robot de capacidades de percepción visual, se implementó un sistema de detección de objetos utilizando aprendizaje por transferencia basado en el modelo YOLOv12 dentro del entorno Ultralytics. Este sistema permite la identificación y localización en tiempo real de objetos en el entorno virtual de simulación y proporciona la información necesaria para los movimientos del robot SCARA, en relación con la ubicación espacial del objeto en la escena.

La red YOLOv12 utilizada está preentrenada con el conjunto de datos COCO (Common Objects in Context), que tiene 80 tipos de objetos en escenas cotidianas e incluye pesos optimizados para detección multiclase, por lo que no se requiere entrenamiento adicional, lo que acelera la implementación.

Del conjunto de datos COCO, se eligieron cinco tipos de objetos con diferentes características para la interacción robótica. Los objetos elegidos son:

Pizza (3319 imágenes)

Fork (3710 imágenes)

Apple (1662 imágenes)

Control remoto (3221 imágenes)

Tijeras (975 imágenes)

Cada imagen capturada por el sensor CoppeliaSim. Los datos del VisionSensor se procesan usando la biblioteca OpenCV, se redimensionan a 640×640 píxeles y se envían al modelo YOLOv12 para su inferencia. Durante el postprocesado, se aplicaron umbrales de confianza de 0,35 y supresión no máxima (NMS) para excluir detecciones redundantes y asegurar la estabilidad de los resultados presentados en la interfaz. La red opera en su modo de inferencia por defecto, con los hiperparámetros predeterminados en el marco Ultralytics, que corresponden a un tamaño de lote de 16, un optimizador automático y aumento de datos con rotaciones, variaciones de brillo, escalado y invertido horizontal.

Fase 3. Integración funcional de sistemas en Python y CoppeliaSim

El sistema de reconocimiento de voz basado en CNN interpreta instrucciones verbales desde una interfaz de usuario en Python, que a su vez envía señales de control robótico a CoppeliaSim. El sistema de detección y seguimiento visual YOLOv12 en tiempo real procesa imágenes capturadas por el sensor integrado en el robot SCARA y guía los movimientos de sus tres articulaciones principales y el efector final según la posición del objeto. Estos sistemas están integrados a través de la API remota ZMQ de CoppeliaSim, que permite la comunicación directa entre el entorno de simulación y las secuencias de instrucciones programadas en Python. Esta integración se sincroniza

doi: 10.25100/iyc.v28i2.15437

utilizando procesos paralelos (hilos) para asegurar una comunicación fluida y estable, evitando retrasos en la ejecución.

Cada instrucción verbal activa una de las siguientes funciones específicas:

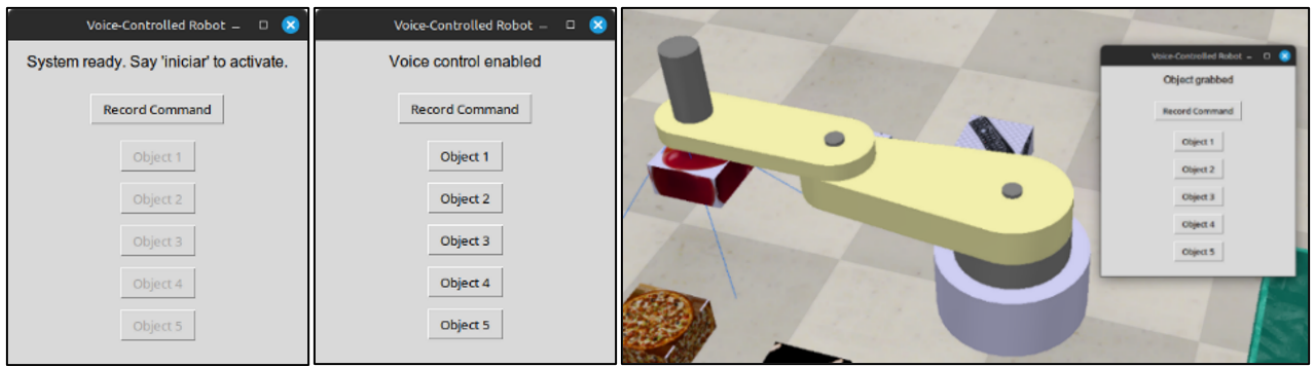
"Iniciar": activa los controles y otros comandos de voz

"Parar": desactiva los controles e impide cualquier acción

"Agarar": ejecuta la rutina de descenso, activando el sistema de succión y levantando el efector final

"Entregar": mueve el brazo a la posición de entrega, desactiva el agarre y devuelve al robot a su posición inicial

La Figura 2 muestra la interfaz gráfica diseñada, que incluye capturar la voz del usuario (Figura 2a), mostrar la instrucción verbal detectada (Figura 2b) y activar los botones para la selección de objetos (Figura 2c). Se utilizó un diseño modular para el sistema, permitiendo mantener el reconocimiento de voz en un hilo separado, asegurando así la ejecución simultánea tanto de la simulación como del reconocimiento de instrucciones verbales.



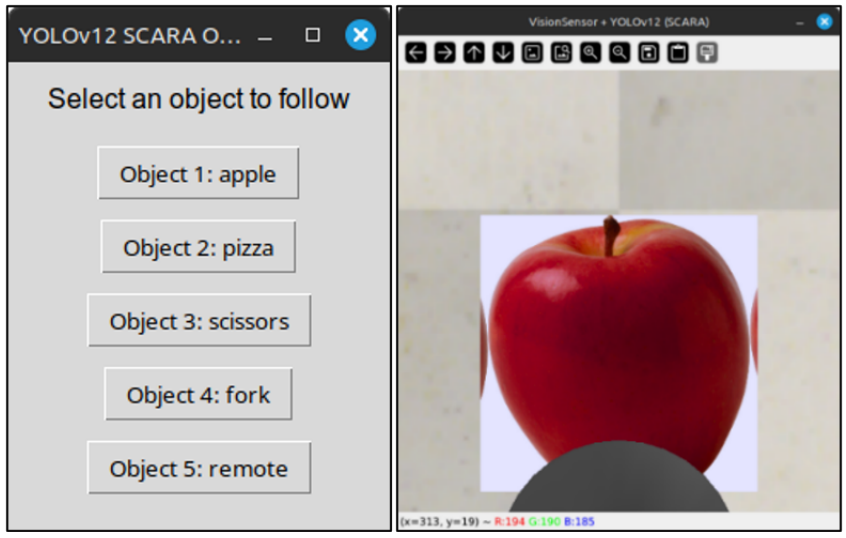
a) Inicio de la demolición b) Control activado c) Captura de objetos

Figura 2. Demostración de integración de sistemas de reconocimiento de voz (CNN **)**

El sistema de visión artificial permite al robot rastrear activamente un objeto seleccionado por el usuario en el campo de visión del sensor de captura virtual. El modelo YOLOv12 produce las coordenadas de la caja delimitadora donde se encuentra el objeto, las etiquetas que lo identifican y los niveles de confianza para cada detección. Basándose en las coordenadas del objeto obtenidas de las cajas delimitadoras, se desarrolló un controlador de derivada proporcional (PD) para centralizar el punto de agarre del objeto. Su salida permite ajustar los ángulos de movimiento del robot SCARA según la entrada en función del error horizontal del área de detección del objeto. El controlador se implementa mediante autoajuste computacional, logrando un kp=0,0001 y un kd=0,00004. Este método permite que el objeto permanezca centrado en la imagen y que la distancia aparente se ajuste usando la relación de área de la caja delimitadora. Cuando el sistema no detecta el objeto, se activa un modo de búsqueda que mueve los dos ejes principales alternativamente, generando un barrido de ±160° para localizar el objetivo.

doi: 10.25100/iyc.v28i2.15437

La aplicación cuenta con una interfaz de usuario (Figura 3) que muestra cinco botones, cada uno asociado a una clase de objeto diferente dentro del conjunto COCO establecido (manzana, pizza, tijeras, tenedor o mando a distancia) (Figura 3a). Seleccionar una de estas clases inicia automáticamente el proceso de detección y seguimiento en tiempo real, como se muestra en la Figura 3b.



a) Inicio b) Vista del objeto

Figura 3. Demostración de la integración de sistemas de visión y seguimiento (**YOLOv12)**

Una vez que todos los componentes han sido probados individualmente, se crea un script final que unifica los tres subsistemas (reconocimiento de voz usando una CNN personalizada, detección y seguimiento visual YOLOv12, y control robot SCARA). La aplicación se basa en una arquitectura de hilos concurrentes, con un hilo principal para gestionar la interfaz gráfica de usuario (GUI), un hilo dedicado para el bucle de visión responsable de la captura, inferencia y seguimiento, y hilos adicionales para el procesamiento de audio y la ejecución de acciones específicas (por ejemplo, agarrar o manejar). Esta independencia permite que la interfaz funcione mientras la simulación está en ejecución, manteniendo la secuencia de visualización activa.

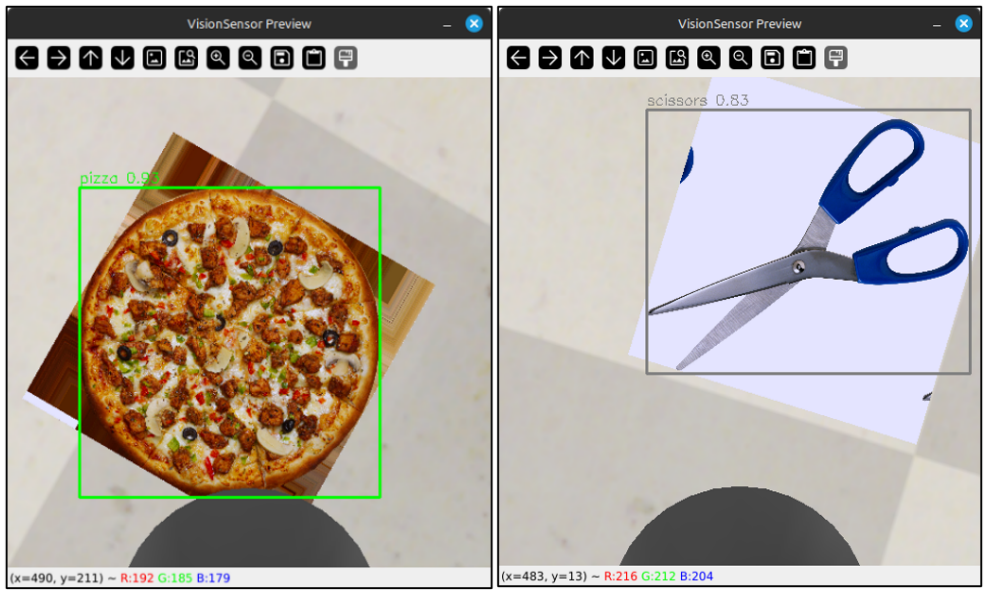
Como estrategia principal para mantener la coherencia entre el sistema de visión y el movimiento del robot, se implementan tres capas (indicador de visión, bloqueo de acción y envoltorios seguros) para evitar que rutinas que involucran movimiento interfieran con la lectura del sensor de visión o el procesamiento de la red YOLO v12. Primero, se utiliza un indicador de visión (vision_ enabled), que se activa mientras no se ejecutan rutinas críticas (agarrar, entregar o volver a la posición inicial). Esto mantiene la red YOLO en estado de espera mientras la visión se desactiva, evitando lecturas incompletas o conflictos paralelos.

El bloqueo de acción garantiza que cada movimiento crítico se realice en un entorno seguro, trabajando exclusivamente en la ejecución y evitando interferencias con el estado del robot. Los envoltorios seguros para lecturas y escrituras de API adquieren bloqueo global, reduciendo fallos y excepciones derivados de llamadas paralelas. La función para mover las articulaciones del robot

doi: 10.25100/iyc.v28i2.15437

se ejecuta en pequeños pasos con velocidad y número limitado de iteraciones, por lo que los movimientos no son bruscos y se evita el bloqueo debido a objetivos ambiciosos, respetando los límites angulares de cada articulación.

Para dibujar cuadros delimitadores alrededor de objetos de interés, un solo hilo se ejecuta en un marco local, impidiendo el acceso concurrente. Además, se aplica filtrado según las clases relevantes y se utiliza un umbral de confianza de 0,35 para reducir los falsos positivos. El recuadro se muestra en verde cuando coincide con el objeto seleccionado (Figura 4a) y en gris cuando no coincide (Figura 4b). Una vez detectado el objetivo con la confianza requerida, se activa el controlador de PD y se interrumpe el procesamiento de nuevas detecciones en ese fotograma, evitando interferencias con la lógica de seguimiento. El dibujo se realiza exclusivamente dentro de la ventana OpenCV, sin modificar la representación interna de CoppeliaSim, reduciendo así la probabilidad de artefactos gráficos.



a) Objeto seleccionado b) Objeto no seleccionado

Figura 4. Cajas delimitadoras

Se implementa un método de sincronización con lógica de voz y una interfaz gráfica usando variables de control que activan o desactivan botones según el estado del robot (Figura 5).

doi: 10.25100/iyc.v28i2.15437

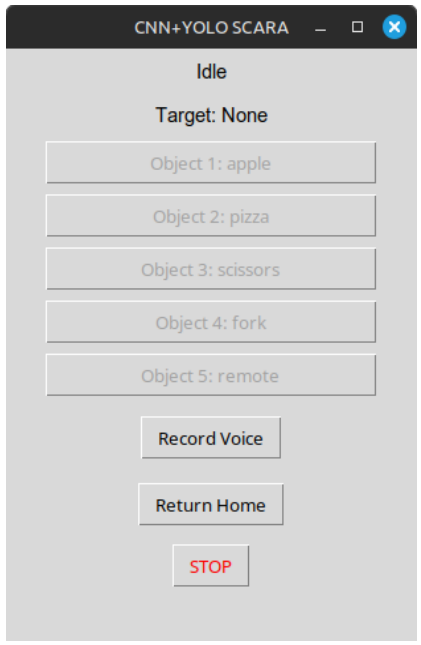
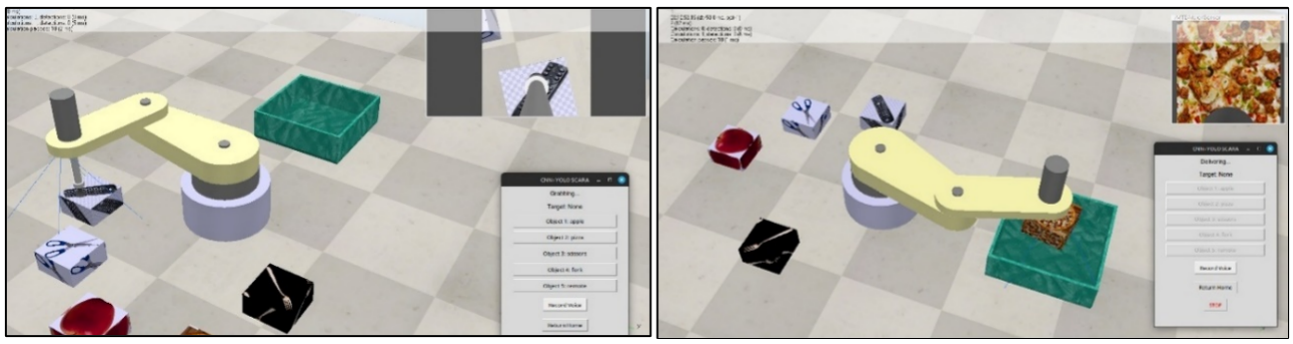


Figura 5. Interfaz gráfica de usuario

El procesamiento de comandos de voz se realiza en un hilo separado que, tras predecir el comando con la CNN, actualiza las variables de control y ejecuta las rutinas correspondientes (Figura 6), que pueden ser para agarrar (Figura 6a) en el punto de ubicación de cada objeto o para la entrega (Figura 6b) en el punto central de la caja de destino en verde.



a) rutina de agarre b) rutina de entrega

Figura 6. Rutinas

Resultados

Rendimiento del sistema de reconocimiento de voz

El modelo de clasificación de comandos de voz, entrenado usando la red neuronal convolucional, mostró una convergencia estable y rápida durante el entrenamiento. La época óptima se alcanzó en la iteración 36, con una precisión máxima de validación de 0,9940 y una pérdida mínima de 0,0180. Todo el proceso de entrenamiento duró 274,83 s. En pruebas realizadas con el conjunto de evaluación (1001 muestras), el modelo alcanzó una precisión global del 98,9% y una puntuación ponderada F1 de 0,9890, superando la precisión media del 97,64% del modelo de memoria a corto plazo bidireccional profundo con unidad recorrente controlada (BiLSTM - GRU profundo) (26). El tiempo medio de inferencia era de 0,0003 s por muestra, lo que permitía respuestas prácticamente instantáneas durante la ejecución del sistema embebido.

doi: 10.25100/iyc.v28i2.15437

Los resultados del sistema de reconocimiento de voz utilizando los datos del conjunto de pruebas permitieron la construcción de la matriz de confusión mostrada en la Figura 7, que destaca una clasificación casi perfecta para las cuatro clases de instrucciones verbales. Hubo pocas confusiones entre los comandos "agarrar" y "parar" (1,6%), "parar" y "entregar" (1,2%), y "agarrar" y "entregar" (0,8%) y "entregar" y "parar" (0,8%), lo que puede deberse a similitudes fonéticas entre las palabras, especialmente en sus terminaciones, y a problemas de sincronización temprana en la grabación durante la entrega del audio.

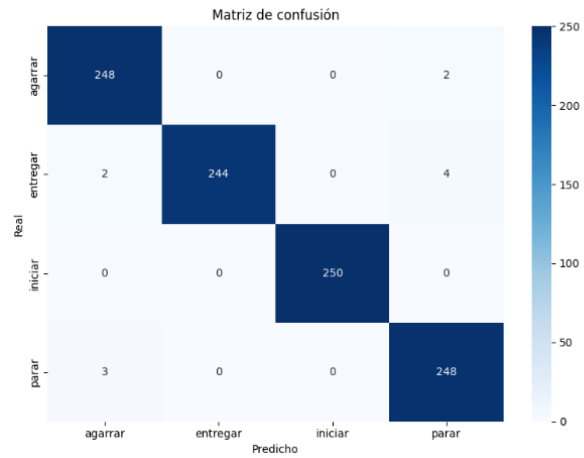


Figura 7. Matriz de confusión de la red CNN utilizada para el reconocimiento de voz

Los resultados demuestran la alta fiabilidad de la CNN diseñada para el reconocimiento de voz, específicamente para los cuatro comandos hablados definidos, en entornos de bajo ruido. Está claro que habrá un sesgo hacia cualquier palabra fuera del contexto no entrenado que desencadene aleatoriamente el movimiento del robot. Por lo tanto, los comandos para operar el robot deben estar claramente definidos o, como trabajo futuro, debe incluirse una clase adicional "no identificada".

Rendimiento del sistema de visión artificial

Durante las pruebas, el sistema de detección visual basado en YOLOv12 mostró una excelente estabilidad. En tiempo real, identificaba con precisión todos los objetos en el entorno virtual simulado con CoppeliaSim, con una tasa de procesamiento que oscilaba entre 20 y 25 fotogramas por segundo (FPS). La tasa de refresco del VisionSensor permaneció sin cambios durante la integración con el entorno virtual y no se vio afectada por el funcionamiento de la red YOLOv12, incluyendo el uso simultáneo de control de roscas y la suspensión del sensor mientras el robot ejecutaba rutinas críticas.

El control de seguimiento implementado mediante un controlador PD (derivada proporcional) mantuvo con éxito el objeto seleccionado dentro del área central del campo de visión con errores inferiores al 7%, que corresponden a un máximo de ±25 px. Esto se lograba ajustando en tiempo

doi: 10.25100/iyc.v28i2.15437

real la posición angular de los eslabones del robot. Este error afecta al punto de agarre del objeto, que en casos extremos acaba cerca de los bordes. Sin embargo, el control PD también redujo la velocidad de búsqueda del centro del objeto, lo que llevó a un ajuste manual hasta encontrar un valor que no restringiera el movimiento y fuera eficaz para evitar oscilaciones y mantener el agarre al alejarse del centro del objeto (error máximo).

Rendimiento del sistema integrado en un entorno simulado

El entorno de simulación bajo CoppeliaSim cuenta con un motor de física integrado que proporciona una representación más detallada del comportamiento de objetos del mundo real. Este entorno de simulación (Figura 8) consta de los siguientes elementos:

Modelo robótico: Se utilizó el robot MTB, un robot SCARA simplificado pero funcional, que tiene dos articulaciones giratorias, una articulación prismática y un efector final con un sistema de succión integrado (suctionPad). Durante la simulación, la base del robot no se movía, las articulaciones giratorias estaban limitadas a ángulos específicos y la almohadilla de succión se controlaba mediante Python para evitar conflictos con las articulaciones del brazo

Sensor de visión: El robot MTB no dispone de un sistema de visión, por lo que un sensor de visión (visionSensor) se integró en el segundo eslabón del brazo, que se configuró para detectar objetos usando la red YOLO v12, permitiendo su seguimiento

Objetos de escena: El entorno virtual contenía cinco cuboides situados delante del robot y un contenedor de almacenamiento. Cada cuboide tiene una textura y color diferentes, simulando objetos con distintas características visuales.

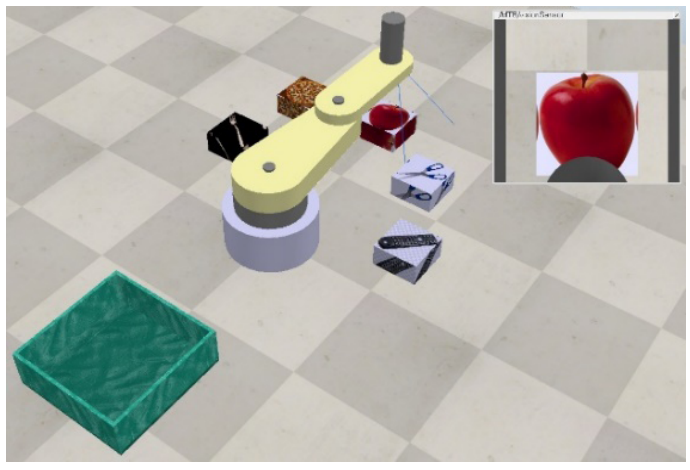


Figura 8. Entorno de simulación virtual en CoppeliaSim

Se realizaron diez pruebas de agarre de objetos en el sistema integrado, revelando errores en el comando de voz o en la propia captura del objeto, tal como se tabula en la Tabla 1 bajo la proporción éxito/fallo. Los principales problemas de agarre se produjeron con los objetos extremos, “tenedor” y “control remoto”, debido al limitado movimiento angular del robot dentro del área de trabajo. Los errores con los objetos centrales se debían principalmente a que el objeto caía durante el proceso de entrega cuando el dispositivo de agarre se alejaba del centro del objeto. El comando de voz mostraba errores en la detección y confusión de clases, siendo el problema más crítico la composición de la etiqueta de “control remoto”.

doi: 10.25100/iyc.v28i2.15437

Tabla 1. Índice de éxito/fracaso de las pruebas.

Objetivo	Comando de voz	Agarre
Pizza	10/0	9/1
Fork	9/1	8/2
Manzana	9/1	10/0
Tijeras	8/2	10/0
Control remoto	7/3	9/1

Discusión

Durante la operación, todo el sistema mostró un rendimiento consistente y una coordinación completa entre sus componentes. Gracias a estrategias de bloqueo de acción y banderas de control, se evitaron los conflictos entre el sistema de reconocimiento de voz, el sistema de reconocimiento visual y el control de movimiento del brazo robótico. Suspender temporalmente el sistema de reconocimiento visual durante la ejecución de rutinas críticas de captación y entrega impide la lectura paralela del VisionSensor, eliminando así problemas de sincronización y sobrecarga de la API de CoppeliaSim.

La evaluación del rendimiento del sistema integrado en el entorno virtual simulado identificó tres limitaciones. Primero, durante la ejecución de la red YOLOv12 en la CPU, se observó latencia en la tasa de inferencia cuando se presentaban múltiples objetos o se encontraban fondos complejos, lo que resultaba en una reducción de la tasa de refresco a aproximadamente 15 FPS. En segundo lugar, se observó una ejecución lenta en los movimientos del brazo robótico debido al control que limita la velocidad angular máxima, buscando movimientos suaves. Finalmente, durante pruebas prolongadas, se observó un alto consumo de memoria cuando la interfaz gráfica y los hilos activos para el reconocimiento visual y de voz estaban activados simultáneamente. Esto podría superarse mediante la gestión dinámica de procesos, por lo que se propone como tema para futuras investigaciones.

Sin embargo, el sistema completo logró mantener tanto la coherencia funcional entre el sistema de reconocimiento de voz, el sistema de reconocimiento visual y el control robótico, como la manipulación estable y autónoma del brazo robótico SCARA.

Conclusiones

Basándose en los resultados obtenidos durante la validación en un entorno simulado, se puede concluir que la implementación de técnicas avanzadas de reconocimiento de voz y visión por ordenador en un sistema robótico no solo es factible, sino también exitosa. La integración lograda representa un avance significativo que mejora la interacción humano-robot al responder a órdenes verbales y fomentar la autonomía robótica en entornos dinámicos al identificar y manipular objetos según su ubicación mediante visión artificial.

doi: 10.25100/iyc.v28i2.15437

Investigaciones futuras proponen optimizar el consumo de memoria y el uso de recursos limitados para permitir aplicaciones del sistema completo en robótica industrial y asistencial, que suelen ser entornos dinámicos y complejos. Además, se establecerá un sistema de entrenamiento vocal que permitirá al usuario pronunciar palabras sin afectar al conjunto de datos entrenado en el disco robótico, como se mencionó anteriormente.

Agradecimientos

Producto derivado del proyecto de investigación "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 financiado por la Vicerrectoría de Investigación de la Universidad Militar Nueva Granada, año 2025. Los autores, en calidad de profesores asociados, agradecen a la Universidad Militar Nueva Granada la financiación del proyecto y el tiempo dedicado a su desarrollo.

Declaración de contribución de autoría de CreditT

Conceptualización - Ideas: Robinson Jiménez Moreno. Curación de datos: José Miguel Bejarano Restrepo. Análisis formal: José Miguel Bejarano Restrepo, Robinson Jiménez Moreno. Investigación: José Miguel Bejarano Restrepo, Robinson Jiménez Moreno. Metodología: Robinson Jiménez Moreno. Dirección de Proyecto: Anny Astrid Espitia Cubillos. Recursos: Anny Astrid Espitia Cubillos. Software: José Miguel Bejarano Restrepo, Robinson Jiménez Moreno. Supervisión: Anny Astrid Espitia Cubillos. Validación: José Miguel Bejarano Restrepo, Robinson Jiménez Moreno. Redacción - borrador original - Elaboración: Robinson Jiménez Moreno. Redacción - revisión y edición -Elaboración: Anny Astrid Espitia Cubillos.

Financiación: no declara. Conflicto de intereses: no declara. Aspecto ético: no declara.

Referencias

1. Bakouri M. Desarrollo de algoritmo de control por voz para sillas de ruedas robóticas utilizando modelos MIN y LSTM. Ordenadores, Materiales y Continua, 2022; 73(2):2441-2456,

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

2. Behera A. K., Mohanraj R. y Prem A. Agarre Inalámbrico Autónomo Inteligente para Operación Colaborativa Humano-Robot. 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. Reconocimiento de gestos con las manos y robot controlado por voz. Materiales de hoy: Actas, 2020; 33(7):4121-4123,

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

4. Saradi V. P. y Kailasapathi P. Control de movimiento basado en voz de un vehículo robótico mediante comunicación por luz visible. Informática e Ingeniería Eléctrica, 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. Cirugía de navegación con voz: Asistencia robótica controlada para el enfoque endoscópico de la hipófisis. Otorrinolaringología - Cirugía de Cabeza y Cuello, 2005; 133(2-Suplemento):P 153,

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

6. Feng Y., He J., Luo J., Fang Z., Zhang C. y Yang G. Un método de autocalibración basado en la visión para robots industriales que utiliza restricciones de postura variable. Robótica y Fabricación Integrada por Ordenador, 2026; 98:103142,

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

7. Lenţoiu I., Răileanu S., Borangiu T., Constantinescu M. y Morariu O. Predicción instantánea de potencia para robots industriales utilizando métodos de aprendizaje automático basados en árboles. Aplicaciones de ingeniería de la inteligencia artificial, 2025; 162(Parte 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. Hacia la manipulación de herramientas robóticas de disparo cero en contexto industrial: un marco modular de VLM mejorado por la representación multimodal de la affordance. Robótica y Fabricación Integrada por Ordenador, 2026; 98:103161,

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

9. Rogowski, A. Control remoto por voz web de células robotizadas. Robótica y Fabricación Integrada por Ordenador, 2013; 29(4):77-89,

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

10. Yu Z., Zhang P. y Shi J. Transformación de la robótica industrial con modelos de lenguaje natural: progreso reciente y perspectivas de futuro. Robótica y Fabricación Integrada por Ordenador, 2026; 97:103113,

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

11. Ali M. L. y Zhang Z. El marco YOLO: Una revisión exhaustiva de la evolución, aplicaciones y referencias en detección de objetos. Informática, 2024; 13(12):336,

https://doi.org/10.3390/computers13120336

12. Deng X., Huang T., Wang W. y Feng W. SE-YOLO: Un marco mejorado con sobel para detección ligera de tomates en tiempo real y de alta precisión con capacidad de despliegue de borde. Informática y Electrónica en la Agricultura, 2025; 239(Parte 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: Múltiples YOLO heterogéneos para la detección de plagas en huertos de manzanos. Procesamiento de la Información en Agricultura; 2025,

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

14. Gong Y., Zhang G., Wang C. y Xiao D. CD-ViT-YOLO: Un modelo híbrido ligero ViT-YOLO para el reconocimiento del comportamiento de patos en jaula bajo condiciones de iluminación variables. Tecnología Agrícola Inteligente, 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: Una red de aprendizaje profundo para la identificación precisa de tallos de fruta de tomate en entornos no estructurados. Tecnología Agrícola Inteligente, 2025; 12:101465,

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

doi: 10.25100/iyc.v28i2.15437

16. Zhou W., Wang J., Meng X., Wang J., Song Y. y Liu Z. MP-YOLO: Poda adaptativa de capas basada en fusión de características multidimensionales YOLO para algoritmo de detección de objetos de vehículos densos. 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. Vigilancia y aplicación inteligente en tiempo real para la detección de lanzamientos de polvo y reconocimiento de identidad usando YOLO 12 y 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: Convolución rectangular adaptativa multinivel para análisis de layout de documentos. Reconocimiento de patrones, 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. Detección de defectos en paneles solares usando los algoritmos YOLO v10 y v11. Electrónica, 2025; 14(2):344,

https://doi.org/10.3390/electronics14020344

20. Chen M., Tian J., Cao X., Fu Z. y Zhang D. DM-YOLO para la detección automática de defectos de los MLCC. Óptica y Tecnología Láser, 2025; 192 (Parte E):113977,

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

21. Kumar A., Dhanalakshmi R., Rajesh R. y Sendhil R. Un Tiny YOLO infundido con características espaciales y pérdida de peso ajustadas para la detección de sombras. Procesamiento de señales: Comunicación de imágenes, 2026; 140:117408,

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

22. Lv L., Li J. y Zhao Y. DMP-YOLO: Percepción densa multiescala para escenas complejas algoritmo YOLO Prunus humilis detección de objetivos pequeños. Tecnología Agrícola Inteligente, 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: Un marco mejorado YOLOv7-tiny para la identificación de la gravedad de la enfermedad de avena con integración de ReXNet y cabeza desacoplada. Tecnología Agrícola Inteligente, 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: Un modelo ligero de detección de enfermedades de la hoja de tomate basado en YOLO11. Tecnología Agrícola Inteligente, 2025; 12:101349,

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

25. Murat A. A. y Kiran M. S. Una revisión exhaustiva de las versiones de YOLO para detección de objetos. Engineering Science and Technology, una revista internacional, 2025; 70:102161,

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

26. Mehra, S., Ranga, V. y Agarwal, R. Un enfoque de aprendizaje profundo para la clasificación de enunciados disartrícos con BiLSTM-GRU, filtrado de señales de voz y espectrogramas log mel. The Journal of Supercomputing, 2024; 80:14520-14547,

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