ISSN 2007-9737

Automated Recognition and Spatial Localization of Laboratory Instrumentation in a Materials Synthesis Setting Using a Collaborative Robot

Alberto Esteban Reyes Peralta[1], Rigoberto Cerino Jiménez[1], Francisco José López Cortés[1], Rabi Soto Camacho[1], David Pinto[1,*], M. Judith Percino[2]

[1] Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación, Mexico

[2] Instituto de Ciencias, Puebla, Unidad de Polímeros y Electrónica Orgánica, Mexico

alberto.reyesp@alumno.buap.mx, cerino_rigoberto@hotmail.com, franciscojlc21@gmail.com, rabi87_soto22@hotmail.com, david.pinto@correo.buap.mx, judith.percino@correo.buap.mx

Abstract. This work presents a methodology for estimating the spatial position of instruments in a materials synthesis laboratory using a 2D camera. This approach is considered a cost-effective alternative to depth sensors for automating tasks in such laboratories. In our case, the methodology is integrated into an ABB collaborative robot, YuMi IRB 14000 model. Instrument detection is performed using a YOLO-NAS network retrained with a specialized dataset. Subsequently, the 2D position of the detected objects is estimated based on camera calibration and the use of projective geometry. Finally, the integration of this system with the YuMi robot controller in Python is described, enabling autonomous interaction with the instruments. Error and precision metrics for distance and location estimation are presented, demonstrating that this solution is viable for low-cost robotic manipulation applications.

Keywords. Spatial localization, laboratory instruments, materials synthesis, collaborative robots, computer vision, 2D camera.

1 Introduction

The automation of processes in materials synthesis laboratories represents a significant advancement for scientific research, enabling greater efficiency and reproducibility. These types of processes require precise, efficient, and low-cost solutions for the identification and manipulation of objects. Collaborative robots (cobots), such as the ABB YuMi, have emerged as key tools in this context, offering an ideal solution due to their ability to safely interact with humans in shared tasks [1]. However, environmental perception remains a major limiting factor. In particular, the accurate estimation of the distance to objects within the robot's field of view is essential for tasks such as manipulation, navigation, and collision avoidance.

Traditionally, spatial localization of objects is achieved using depth sensors such as RGB-D cameras or LiDAR systems. However, these come with high costs, and 2D cameras remain an attractive alternative due to their low cost and simplicity.

This article proposes a computer vision-based solution using only a 2D camera, which employs convolutional neural networks (CNNs) to recognize common objects in materials synthesis laboratories. A practical approach is proposed for distance estimation using 2D images, highlighting its relevance in the context of cobots, with the ultimate goal of enabling these types of robots to autonomously conduct experiments.

This system has been integrated into an ABB YuMi collaborative robot as part of a

project focused on the automation of synthesis and spectroscopic characterization of organic compounds.

2 State of the Art

Distance estimation and object detection have evolved significantly with the development of new technologies, ranging from traditional geometric methods to advanced approaches based on artificial intelligence and deep learning. Depth information has traditionally been obtained through sensors such as LiDAR, ultrasonic, or infrared devices. However, with the widespread adoption of RGB cameras and the efficient processing of images, computer vision has taken on a key role in these tasks, especially in systems where cost reduction is a priority.

The literature documents various methodologies for object detection and localization. These range from multi-level segmentation techniques to automatic correction algorithms for embedded systems and process control. For example, Quintana et al. [2] present concepts and methodologies for developing a computer vision system for counting moving objects based on pattern recognition and image processing techniques. In the agricultural domain, García-Santillán et al. [3] developed a system for automatically distinguishing between crops and weeds in potato fields, while Herrera et al. [4] proposed a system for the automatic classification of coffee fruits during the production stage using color features and supervised learning algorithms.

Computer vision applications also extend to medical and environmental areas. Medina et al. [5] describe the development of a digital photopedometer to support gait diagnosis using computer vision techniques, and Revollo et al. [6] propose a coastal monitoring system using digital images to study shoreline dynamics through the development of custom vision software. Similarly, Bohorquez [7] developed an embedded computer vision system for detecting and tracking objects from an unmanned aerial vehicle (UAV), demonstrating the feasibility of integrating deep learning into mobile platforms with limited computational resources.

In recent years, a growing integration of computer vision with collaborative robotics has been observed. Otero et al. [8] highlight the impact of artificial intelligence in collaborative industrial environments, emphasizing how collaborative robots (cobots) can benefit from visual capabilities to recognize objects and people and make autonomous decisions. Huayanay and Meneses [9], meanwhile, present the design and predictive control of a collaborative robot for agricultural and mining tasks, combining vision sensors with machine learning techniques.

The evolution of convolutional neural networks (CNNs) has revolutionized object detection in 2D images. Models like Faster R-CNN, SSD, and especially YOLO (You Only Look Once) allow for real-time object identification and localization with high accuracy. The YOLO-NAS version, developed through Neural Architecture Search, optimizes the balance between performance and speed, which is crucial for embedded systems and collaborative environments.

Beyond detection, some works have addressed the direct estimation of depth maps from monocular images using neural networks. Although these approaches require large amounts of accurately labeled data, their practical application is limited in systems where computational resources and real-time performance are critical. In this context, combining lightweight models like YOLO with simplified projective geometry emerges as an effective solution for specific tasks such as locating beakers or flasks in a laboratory.

In the field of collaborative robotics, Ventura-Cruz et al. [10] explain how the integration of artificial intelligence and mechatronics has transformed manufacturing processes, especially in the automotive industry. Through production line automation, predictive maintenance, and the use of smart sensors, efficiency, customization, and operational safety have been improved. This transformation is driven by the adoption of collaborative robots capable of adapting to dynamic environments and performing complex tasks with precision.

The ABB YuMi IRB 14000 collaborative robot has established itself as a versatile platform for assembly, sorting, and manipulation tasks in industrial settings. Thanks to its dual-arm

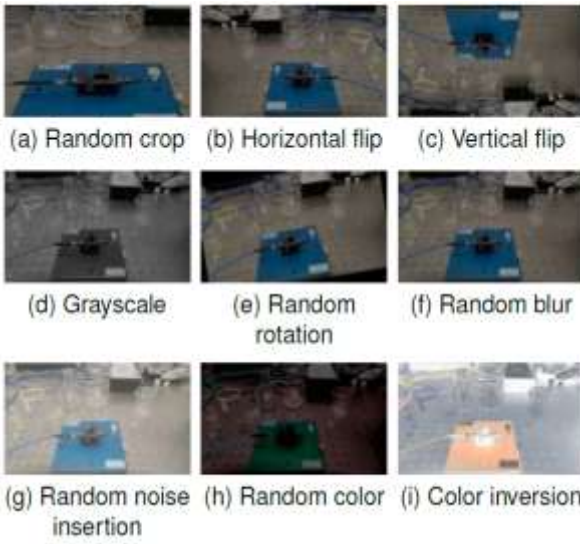

Fig. 1. Images generated using geometric and color space transformations from the original image

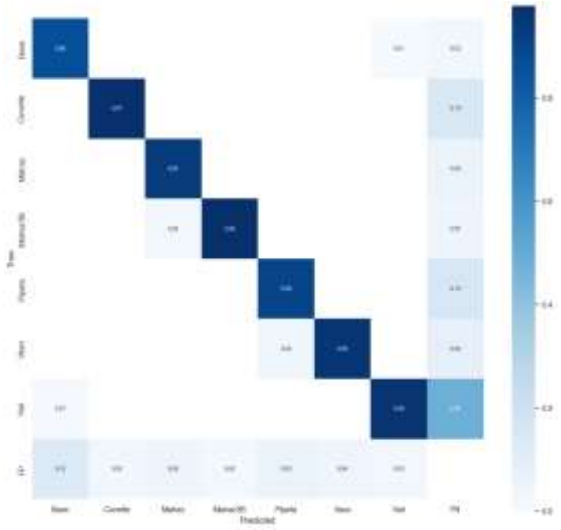

Fig. 2. Confusion matrix obtained from the evaluation of the model trained with the custom dataset

architecture and collision detection capabilities, it is commonly used in processes where direct interaction with humans is essential. Although specific studies on YuMi are scarce in the reviewed literature, research by González and Vega [11] demonstrates the effectiveness of integrating computer vision into cobots for performing manipulation tasks, while other studies, such as that of García and López [12], analyze from a theoretical perspective how vision systems enhance human-robot interaction in collaborative environments.

In applied research, artificial vision has been implemented in combination with optimized inference engines such as TensorRT and ONNX Runtime, enabling high-performance real-time object detection and localization. These capabilities are integrated with control platforms like RoboDK and ROS (Robot Operating System), facilitating synchronization between visual detection, pose estimation, and robotic movement.

For example, Holmes and Toscano [13] designed an automated palletizing system using three collaborative arms, optical sensors, QR tags, and visual processing in Python with OpenCV. This system was controlled through a modular architecture including vision, decision logic, and robotic execution, representing a strong example of how collaborative robots can benefit from advanced visual capabilities to perform high-precision industrial tasks.

This trend toward the fusion of computer vision and artificial intelligence in collaborative robotic environments represents progress toward the development of smarter, more autonomous, and safer systems.

The ability to estimate the three-dimensional position of an object from a single 2D image, relying on geometric models and CNN-based detection, offers practical solutions for manipulation tasks where precision is important but not critical such as identifying and repositioning items within an automated laboratory.

Thus, the literature reveals a growing interest in integrated systems that combine computer vision, deep learning, and collaborative robotics. As reviewed across several articles, the interaction between computer vision and collaborative robotics is booming.

Robots such as the ABB YuMi have been used in assembly, classification, and manipulation tasks in shared environments with humans. However, most industrial solutions are based on 3D sensors or stereo vision systems.

The proposal to use a 2D camera and neural networks trained specifically for the laboratory environment represents a novel and efficient contribution.

Fig. 3. Chessboard used for camera calibration

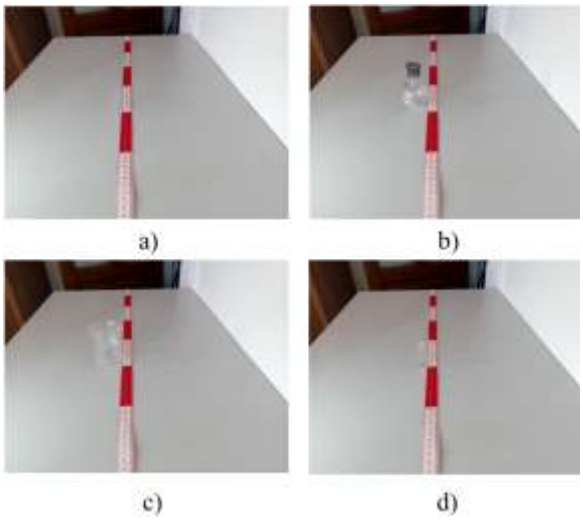

Fig. 4. Reference distance, a) Cuvette, b) Flask, c) Beaker glass, d) Vial

3 Methodology

The methodology is divided into four main stages: object detection, camera calibration, distance estimation, and integration with the YuMi robot.

3.1 Object Detection with YOLO-NAS

CNNs have proven to be a powerful tool for object recognition in images [14, 15]. To achieve optimal performance, it is essential to train the network with a dataset that accurately represents the object classes to be detected. A 2D camera mounted on the body of the YuMi robot was used to capture images of the workspace. In this work, a process for creating a custom dataset was implemented.

Image Collection: A total of 809 color images of the selected laboratory instruments were captured using a 2D camera with a resolution of 1080 × 1920 pixels.

Image Labeling: The objects in the images were manually labeled using the open-source software Label Studio [16]. This process is essential for providing the CNN with accurate information about the location and class of each laboratory instrument.

Data Augmentation: Data augmentation techniques were employed to increase the variability of the dataset and improve the model's generalization. Geometric transformations and color space transformations were used. These techniques were implemented in Python using the Albumentations library [17]. The operations performed included: random cropping, horizontal and vertical flipping, random rotation, grayscale conversion, random blurring, random noise insertion, random color adjustment, and color inversion. This implementation resulted in a total of 6,777 images [18]. Figure 1 shows the transformations applied.

The YOLO-NAS architecture [19] was implemented and retrained using a custom dataset of laboratory images (beakers, flasks, vials, etc.). The training was conducted in PyTorch using the SuperGradients library, with data augmentation and dataset partitioning into training, validation, and testing sets. A mean average precision (mAP@0.5) of 91.5% was achieved. The confusion matrix shown in Fig. 2 displays values above 0.75 along the main diagonal and low rates of false positives and false negatives.

3.2 Camera Calibration

A 2D USB camera was mounted in a fixed position on the body of the YuMi robot. Intrinsic calibration was performed using a chessboard pattern and the OpenCV library. This calibration yielded the camera matrix and distortion coefficients required to map pixel coordinates to real-world coordinates in millimeters. Specifically, a standard calibration pattern was used, a 10x7 chessboard (see Figure 3). Multiple images of the pattern were captured from different perspectives.

Table 1. Focal Distance Calculation

Objet	Reference distance	Width	Width in píxels	Focal Distance
Cuvette	30 cm	1.5 cm	23 px	460.0
Flask	30 cm	5 cm	64 px	402.0
Beaker glass	30 cm	5.8 cm	87 px	450.0
Vial	30 cm	1.5 cm	26 px	52.0

Subsequently, an algorithm was implemented in Python using the OpenCV library to detect the corners of the pattern in each captured image. The known 3D points of the pattern and the 2D points detected in the images serve as inputs to the `cv.calibrateCamera` function, which calculates the intrinsic and extrinsic parameters of the camera. The intrinsic parameters describe the internal characteristics of the camera, such as focal length and optical center, while the extrinsic parameters relate the camera's coordinate system to the real-world coordinate system.

Once the calibration parameters are obtained, they are stored in a file for use in later tasks, such as distance estimation. The results show a significant improvement in image quality, demonstrating the removal of typical lens-induced distortions.

After obtaining these parameters, their effectiveness is evaluated on images from the training set. To do this, the `cv.undistort` function is applied to correct distortions using the calibrated parameters.

3.3 Distance and Position Estimation

Spatial localization of objects is essential in applications such as robotics, augmented reality, and autonomous navigation. This process aims to determine the position and distance of objects within a three dimensional environment. There are three main approaches:

1. **Depth sensors** such as LiDAR, which emit laser pulses to generate accurate 3D maps of the environment. They offer high precision but are expensive [20].
2. **Stereo 3D cameras**, which simulate human binocular vision to estimate depth by calculating the parallax between two images [9].
3. **Computer vision with geometric models**, which use principles of perspective and visual features (such as lines and corners) to infer depth from 2D images.

In addition, deep learning-based methods, particularly convolutional neural networks (CNNs), stand out for their ability to estimate depth directly from images. Although these approaches require large volumes of data and significant computational power, they offer high accuracy in complex and challenging environments.

Once the object has been identified, the next task is to determine the distance to that object. To do this, the method based on triangle similarity is employed. Specifically, for each detected object, the width of the bounding box in pixels is calculated, and using the distance formula derived from triangle similarity, the depth of the object is estimated:

$$F = \frac{(P \times D)}{W} \tag{1}$$

where **F** is the focal distance (calculated), **D** is the reference distance to the object, **W** is the real width of the object (in mm), and **P** is the width of the object in pixels. Using this distance, along with the coordinates of the center of the bounding box, the object's position is projected into the robot's workspace.

A Python module was developed that, using an object detection model, captured images at a reference distance of 30 cm for four objects found in the materials synthesis laboratory (see Figure 4). Additionally, the pixel width of each object was obtained, and thus focal length was calculated. This value was stored in a configuration file for later use. The results obtained are presented in Table 1.

Once the focal distances have been recorded, it becomes possible to estimate the distance to an object using the following formula:

$$D' = \frac{(W \times F)}{P} \tag{2}$$

ISSN 2007-9737

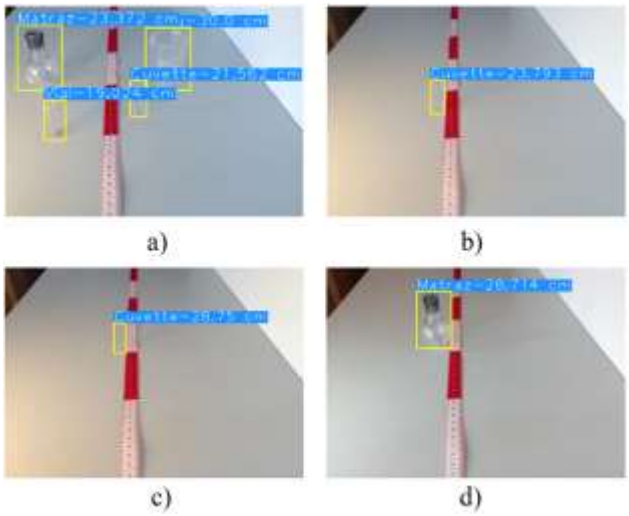

Fig. 5. Distance estimation. a) Cuvette, Flask, Beaker glass and Vial, b) Cuvette, c) Cuvette, d) Flask

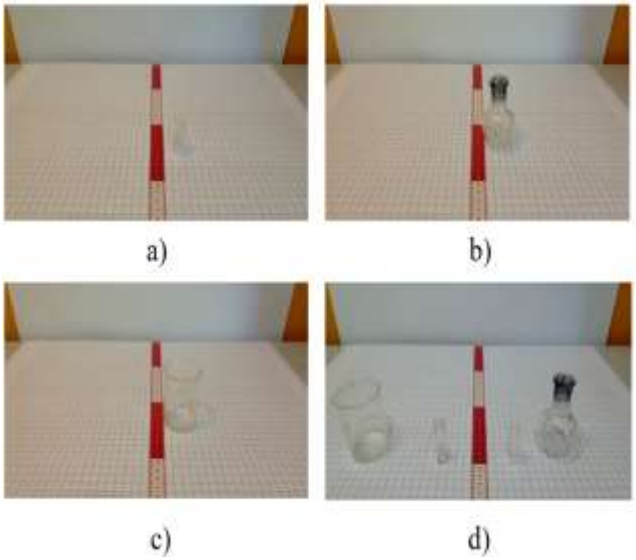

Fig. 6. Reference distance in a workspace identical to that of the collaborative robot. a) Cuvette, b) Flask, c) Beaker glass, b) Cuvette, c) Beaker glass, Vial, Cuvette and Flask

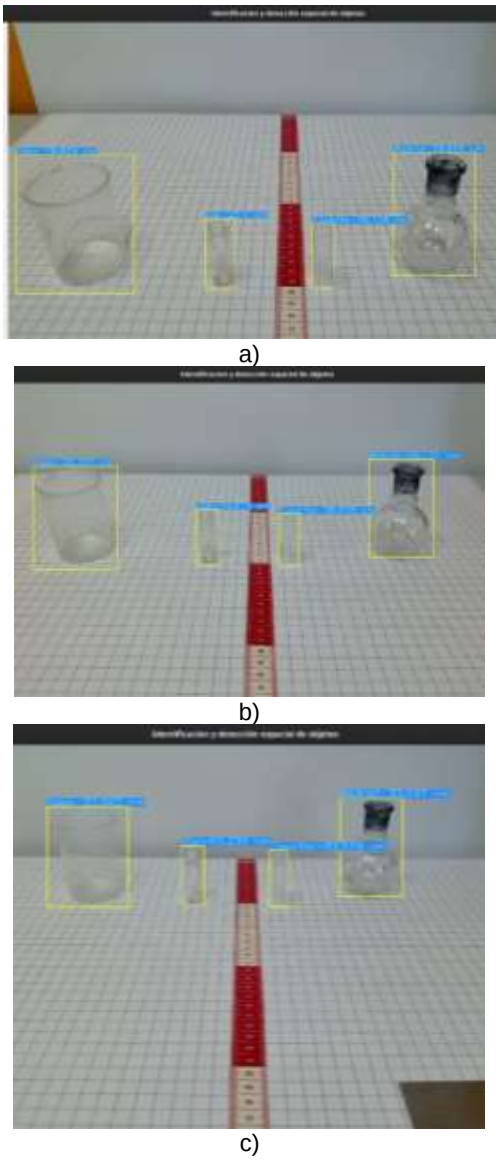

Fig. 7. Distance estimation at interval distances. a) 20 cm, b) 30 cm y c) 40 cm

Based on this, the object detection module integrates a procedure to automatically calculate the distance to objects, provided that the previously recorded focal distance for the detected object is available. Some examples of distance estimations are illustrated in figure 5.

After obtaining the initial distance estimations, a new simulation was conducted in an environment that replicates the dimensions of the collaborative robot's workspace. This procedure allows for evaluating whether the distance estimates are accurate enough to ensure optimal robot performance in its operational environment (see Fig. 6).

A new calculation of the focal distance was performed, with the results presented in Table 2.

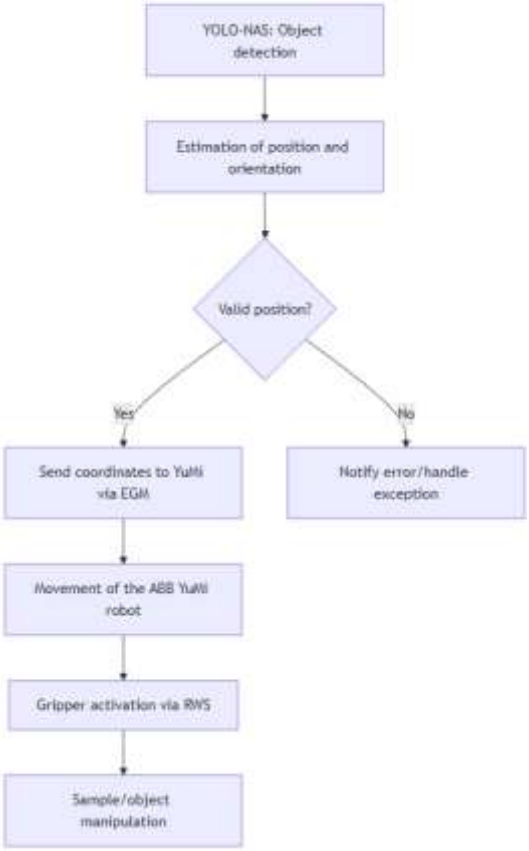

Fig. 8. Diagram illustrating the workflow of the proposed integration.

Subsequently, distance estimations are carried out at 10 cm intervals (20, 30, and 40 cm). The results shown in figure 7, reveal opportunities to refine the model and achieve more accurate measurements.

3.4 Integration with the ABB YuMi Robot

Communication with the robot is carried out via RobotWare and Python, using the ABB module to send estimated coordinates and execute positioning routines. The routines were developed in RAPID and are triggered by real-time reading of the values sent from the vision system.

The integration is established between a vision system based on deep neural networks and an ABB YuMi IRB 14000 collaborative robot, using the Externally Guided Motion (EGM) and Robot Web Services (RWS) protocols to enable autonomous, flexible, and safe manipulation of materials and samples in a laboratory environment.

The integration begins with the creation of a virtual YuMi system in RobotStudio. The EGM and RWS modules are enabled to allow external control and web-based communication, respectively. The detailed configuration was carried out following the following steps:

1. Creation of the virtual YuMi system with RobotWare modules: EGM, RWS, and PC Interface.
2. Definition and configuration of digital signals for gripper control.
3. Development of a RAPID program that initializes the EGM channel and provides procedures to open/close the gripper via digital signals.
4. Connectivity testing in simulation to validate communication and safety before transferring the configuration to the physical robot.

Communication between the vision system and the robot is carried out as follows:

- **EGM:** Used to send real-time coordinates and orientations from Python to the robot. EGM enables Cartesian control of the robot's axes, adapting movements to the positions calculated after object detection.
- **RWS:** Used for controlling the gripper and executing RAPID procedures. Through HTTP (RESTful) requests, the Python server can open or close the gripper and query the system status.

The typical workflow is as follows (see Fig. 8):

1. The YOLO-NAS network detects and locates objects of interest.
2. The Python server calculates the target position and orientation.
3. The motion information is sent to the robot via EGM.
4. Once the position is reached, the gripper is activated through RWS.

The system validation was carried out in stages:

Table 2. New Focal Distance Calculation

Objet	Reference distance	Width	Width in píxels	Focal Distance
Cuvette	30 cm	1.5 cm	23 px	460.0
Flask	30 cm	5 cm	64 px	402.0
Beaker glass	30 cm	5.8 cm	87 px	450.0
Vial	30 cm	1.5 cm	26 px	52.0

Table 3. Distance estimation error by object type

Object	MAE (mm)
Beaker glass	3.2
Flask	4.1
Cuvette	2.8
Vial	3.5

Table 4. Error calculation in distance estimation for the cuvette

Real Distance	Estimated Distance	Difference	Errror (%)
20 cm	20.294 cm	0.294 cm	1.47 %
25 cm	23.793 cm	-1.207 cm	4.82 %
30 cm	28.750 cm	-1.250 cm	4.16 %
35 cm	28.750 cm	-6.500 cm	18.57 %
40 cm	32.857 cm	-7.143 cm	17.85 %

Fig. 9. ABB YuMi IRB 14000 cobot in the process of grasping a beaker glass

- **EGM Test:** Initially, simple position commands were sent via Python scripts to verify the robot's response in simulation.
- **RWS Test:** The ability to open and close the gripper, as well as to execute RAPID procedures from the Python environment via HTTP requests, was verified.
- **Full Integration:** Finally, the artificial vision system was integrated, enabling autonomous manipulation of objects detected in real time.

4 Obtained Results

The system was evaluated in a laboratory environment simulating manipulation tasks. Four types of objects were used, and for each one, its position relative to the robot's frame was estimated. Table 3 shows the mean absolute error (MAE) in the estimated distance.

Distance estimation tests were carried out using objects such as a beaker, a cuvette, a flask, and a vial. These objects were placed at positions located 20, 25, 30, 35, and 40 cm away. To evaluate the accuracy of the estimations, the distances calculated by the system were compared with the actual distances using equation 3 to determine the percentage of error. Table 4 shows the distance estimation results for the cuvette.

$$e_r(\%) = \frac{D_{est} - D_{real}}{D_{real}} \times 100 \quad (3)$$

The results presented in the tables reveal a clear trend: the accuracy of distance estimations decreases as the distance between the camera and the object increases. This drop in precision is attributed to the loss of object detail at greater distances. Notably, the maximum recorded error percentage was 17.85%, while the minimum was 1.47%, specifically for the cuvette. Interestingly, the second estimation did not show a significant improvement; in some cases, the error percentage even increased, suggesting the need for further adjustments in the method or parameters used to improve the precision of distance estimations.

The system enabled accurate identification and approach of the robot to the objects, with an average deviation of less than 5 mm, which is sufficient for grasping tasks in structured

environments. A success rate of 94% was achieved in the execution of grasping tasks by the robot, using only the coordinates estimated from the 2D camera.

5 Discussion

The implementation demonstrates that it is possible to obtain useful spatial coordinates for robotic manipulation using only a 2D camera, provided that precise calibration is performed and the real dimensions of the objects are known. The detection system based on YOLO-NAS proved robust against lighting variations and moderate angular positions. This approach offers advantages in terms of cost, integration simplicity, and inference time, although it has limitations when 3D coordinates or highly dynamic environments are required.

The integration with the YuMi collaborative robot validates the feasibility of this technique in real-world robotic manipulation tasks. The integration using EGM and RWS proved to be robust and flexible. Precise control of the YuMi robot was achieved in both simulation and physical operation, with minimal latency and safe responses even under variable load conditions. The proposed architecture allows expansion to more complex tasks and it could integrate additional sensors due to its modular communication design.

It was observed that EGM provides smooth, real-time motion control, ideal for delicate manipulation tasks. On the other hand, RWS facilitated the management of digital signals and interaction with RAPID procedures, simplifying control logic and integration with external software.

6 Conclusion and Future Work

A 2D vision-based position estimation system was developed for integration with an ABB YuMi IRB 14000 robot. Object detection using YOLO-NAS and position estimation based on camera geometry enable accurate localization without the need for depth sensors. This methodology has applications in laboratory automation, light manufacturing, and educational robotics. The seamless integration with the ABB YuMi robot via Python opens the door to autonomous experimentation systems in scientific environments.

This work presents a promising approach for automating processes in materials synthesis laboratories through the use of cobots. The combination of CNNs specifically trained for laboratory environments with distance estimation techniques represents a significant step toward enabling autonomous experimentation.

The methodology based on EGM and RWS for integrating intelligent systems with ABB YuMi robots proves to be an effective alternative for advanced automation in research laboratories. This integration enables flexible and safe routines, allowing researchers to focus on higher value-added tasks.

Future improvements may include the incorporation of ROS-Industrial for greater interoperability and the use of reinforcement learning techniques to optimize robotic manipulation.

Future work will focus on:

- Optimizing distance estimation techniques to improve the system's accuracy and robustness.
- Developing trajectory planning algorithms that consider the physical limitations of the cobot and the characteristics of the environment.
- Validating the system in real laboratory scenarios, demonstrating its ability to autonomously perform complex experiments.

Acknowledgments

This work has been funded by Project CF-2023-G-828 "Automatización de procesos de síntesis y caracterización espectroscópica de compuestos orgánicos, mediante el uso de un robot colaborativo", SECIHTI.

References

1. **M. Guertler, L. Tomidei, N. Sick (2023).** When is a Robot a Cobot? Moving Beyond

Manufacturing and Arm-Based Cobot Manipulators. Proceedings of the Design Society, Vol. 3, pp. 3889–3898. doi: 10.1017/pds.2023.390.

2. **Quintana, J., Martínez, D.I., Ruiz, G. (2018).** Sistema de visión artificial para el conteo de objetos en movimiento utilizando reconocimiento de patrones. Revista Científica y Tecnológica de la UJAT, Vol. 7, No. 1, pp. 63–73.
3. **García-Santillán, M., Macías, A. (2021).** Desarrollo de un sistema de visión artificial para la discriminación automática entre cultivos y malezas en sembradíos de papa. Revista Mexicana de Ciencias Agrícolas, Vol. 12, No. 4, pp. 753–765.
4. **Herrera, D., Morales, R. (2021).** Clasificación automática de frutos de café utilizando características de color y aprendizaje supervisado. Revista Politécnica, Vol. 17, No. 34, pp. 55–63.
5. **Medina, A., Otero, A. (2014).** Diseño y validación de un fotopodómetro digital para diagnóstico clínico de la pisada. Ingeniería y Ciencia, Vol. 10, No. 20, pp. 139–154.
6. **Revollo, J., Gutiérrez, A. (2019).** Sistema de monitoreo costero basado en imágenes digitales para estudiar la dinámica del litoral. Ingenius, Vol. 22, pp. 53–62.
7. **Bohorquez, J. (2019).** Sistema embebido de visión artificial para la detección y seguimiento de objetos en un vehículo aéreo no tripulado (UAV). Tesis de Maestría, Universidad Nacional de Colombia.
8. **Otero, A., Ayala, J. (2022).** Impacto de la inteligencia artificial en entornos colaborativos industriales: de la percepción visual a la toma de decisiones. Ingeniería Industrial, Vol. 45, No. 3, pp. 201–210.
9. **Huayanay, W., Meneses, F. (2022).** Diseño y control predictivo de un robot colaborativo agrícola-minero con sensores de visión e inteligencia artificial. Revista Iberoamericana de Automática e Informática Industrial, Vol. 19, No. 2, pp. 151–160.
10. **Ventura-Cruz, I., Pérez-Romero, M.E., Jiménez-Islas, D. (2024).** La integración de la Inteligencia Artificial y la Mecatrónica en la fabricación de vehículos. REIA, Vol. 8, No. 9, pp. 87–99.
11. **González, C., Vega, L. (2022).** Aplicación de visión por computador en un brazo robótico colaborativo para tareas de manipulación. Memorias del Congreso Internacional de Robótica Avanzada, pp. 102–109.
12. **García, J., López, M. (2021).** Sistemas de visión para interacción humano-robot en entornos colaborativos. Revista Iberoamericana de Tecnologías Colaborativas, Vol. 6, No. 2, pp. 77–85.
13. **Holmes, C., Toscano, G. (2023).** Sistema automatizado de paletizado con robots colaborativos y visión artificial usando OpenCV y RoboDK. Actas del Congreso Internacional de Automatización Industrial, Vol. 4, No. 1, pp. 31–42.
14. **Shi, Y., Li, X., Chen, M. (2023).** SC-YOLO: A Object Detection Model for Small Traffic Signs, in IEEE Access, Vol. 11, pp. 11500–11510, doi: 10.1109/ACCESS.2023.3241234**.**
15. **Zhao, H., Tang, Z., Li, Z. (2024).** Real-Time Object Detection and Robotic Manipulation for Agriculture Using a YOLO-Based Learning Approach, arxiv, pp. 1–7. doi: 10.48550/ arXiv.2401.15785.
16. **Tkachenko, M., Malyuk, M., Holmanyuk, A. (2022).** Label Studio: Data labeling software. Open source software available from https://github.com/heartexlabs/label-studio.
17. **Buslaev, A., Iglovikov, V.I., Khvedchenya, E. (2020).** Albumentations: Fast and Flexible Image Augmentations. Information, Vol. 11, No. 2, pp. 1–20. doi: 10.3390/info11020125.
18. **Reyes-Peralta, A.E., Pinto-Avendaño, D.E., López-Cortés, F.J. (2024)**. Development of a Customized Image Set for Object Detection in a Materials Synthesis Lab Using Convolutional Neural Networks. Computación y Sistemas. Vol. 28, No. 4, pp. 2363–2368. doi: 10.13053/cys-28-4-5359.
19. **Ultralytics (2024).** YOLO-NAS: Next-generation YOLO for industrial applications. https://github.com/ultralytics/yolonas.
20. **Frank, E. (2024).** Lidar sensors Technology and applications author.

21. **Bolas, M.T., Woods, A.J., Merritt, J.O. (2002)**. Stereoscopic Displays and Virtual Reality Systems IX, Electronic Imaging, Vol. 4660.
22. **ABB Robotics (2023).** Externally Guided Motion. Application Manual. ABB Library. https://library.e.abb.com/public/86e4c5eebc3f4b41a6b2c5d3c2dfe7e9/3HAC050942-001_RevK.pdf.
23. **ABB Robotics (2024).** Robot Web Services. Application Manual. ABB Library. https://library.e.abb.com/public/9e6c5d3a7dbe4b8db5a2d8b5c6bc0e5a/3HAC078336-001.pdf.
24. **Kramberger, A., Skubic, B., Podržaj, P. (2017)**. Externally Guided Motion for Online Control of Industrial Robots. Industrial Robot: An International Journal, Vol. 44, No. 4, pp. 457–466. doi: 10.1108/IR-05-2017-0081.

Article received on 09/06/2025; accepted on 11/10/2025.
**Corresponding author is David Pinto.*