Human following robot using deep learning techniques

Luis Bernardo Bremer[a], Eduardo Sánchez[a], Diego Hernández[a], Ulises Orozco-Rosas[a,*], and Kenia Picos[a]

[a]CETYS Universidad, Ave. CETYS Universidad No. 4, Fracc. El Lago, C.P. 22210, Tijuana, Baja California, México

ABSTRACT

This work proposes a human-following robot based on deep learning techniques. The system utilizes a deep neural network to detect and track a target in real time, using an onboard camera coupled with an autonomous navigation module for safe operation. Key challenges such as handling occlusions, varying lighting, and real-time processing are addressed. The anticipated result is a robust system applicable to personal assistance, security, and healthcare. The proposed methodology integrates real-time object detection using the YOLOv4 deep learning model with a histogram-based identity lock mechanism for consistent person tracking. The integrated camera captures live video, which is processed locally to detect and follow a human target. Motion commands are computed based on the position and size of the detected bounding box and sent to TurtleBot2 using the Robot Operating System. In experimental tests, the robot maintained an average tracking accuracy of 94.6% with a real-time processing speed of 12-15 fps and a command response delay of 0.3 seconds. These results demonstrate the system's ability to reliably follow a human target under indoor conditions without the use of additional sensors.

Keywords: Mobile robots, deep learning, human tracking, autonomous navigation, deep neural networks

1. INTRODUCTION

Mobile robots with autonomous human-following capabilities are becoming increasingly important in areas such as service robotics, healthcare, and personal assistance.[1] This work focuses on developing a human-following robot based on real-time computer vision techniques and deep learning models. Using the TurtleBot2 platform equipped with an onboard camera, the robot detects and tracks humans dynamically by leveraging the deep learning model YOLOv4.

The primary objective is to enable the robot to identify a human target within its field of view, estimate its relative position, and autonomously follow it while maintaining a suitable distance. Unlike traditional path planning and obstacle avoidance systems that rely on LiDAR sensors and pre-built maps, this system bases its perception and decision-making solely on visual input.

A significant aspect of this work involves evaluating the YOLOv4 model's performance in real-time conditions, focusing on key metrics such as detection precision, recall, inference speed (measured in frames per second), and robustness under varying environmental conditions. Special attention is given to the neural network's computational efficiency and its impact on the robot's ability to track a moving target smoothly and reliably.[2]

This paper presents the system architecture, neural network integration, experimental setup, and results obtained from real-world testing in indoor environments. The proposed solution demonstrates a viable approach for lightweight, real-time human tracking on mobile robots, eliminating the need for complex sensor suites or map-based navigation.[3]

Recent works have explored human-following robots using a variety of approaches. For example, Yin et al. proposed a DNN-based system combined with a fuzzy controller to adjust velocity and maintain the target in view .[4] Algabri and Choi developed a deep-learning-based indoor tracking system using SSD (Single Shot Detector) detection and HSV (Hue, Saturation, and Value) color histograms, achieving robust results under occlusions and

*Further author information:
U. Orozco-Rosas: E-mail: ulises.orozco@cetys.mx

illumination changes .[5] These studies highlight the growing trend toward deep learning and color-based identity tracking for mobile robots.

The organization of this paper is as follows: Section 2 introduces the theoretical foundations and describes the system architecture. Section 3 presents the object detection and identity-locking components. Section 4 discusses the experimental methodology and results. Finally, Section 5 concludes the paper and outlines directions for future work.

2. FOUNDATION

This section aims to describe the fundamental theory required for developing the implementation of the algorithm presented in this work.

2.1 Differential drive robot kinematics

The mobile robot platform used for this work is the TurtleBot2, a widely used differential drive robot that offers modularity, ROS integration, and real-time motion control.[6] The TurtleBot2 is equipped with two independently controlled wheels located on either side of its base, which allows it to perform forward motion, reverse motion, and in-place rotation by varying the velocities of each wheel.[7]

In a differential drive configuration, the robot's motion is described by two control variables: linear velocity v and angular velocity ω.[8] The following equations govern the relationship between wheel linear and angular velocities:

$$v = \frac{r}{2}(\omega_R + \omega_L) \tag{1}$$

$$\omega = \frac{r}{L}(\omega_R - \omega_L) \tag{2}$$

Where r is the radius of the wheels, L is the distance between the wheels (wheelbase), and ω_R, ω_L represent the angular velocities of the right and left wheels, respectively.

The TurtleBot2 uses a Kobuki base,,[6] which provides wheel odometry and low-level motor control. This base supports wheel encoders for precise measurement of displacement and rotational speed, enabling the reliable estimation of the robot's pose over time.

Using these kinematic equations and sensor feedback from the base, the robot's control system can perform smooth tracking and navigation. All velocity commands are issued through the ROS `/cmd_vel` topic, allowing for the integration of motion logic with the perception module, which handles human detection and tracking.

This configuration enables the TurtleBot2 to execute the real-time navigation tasks required in this work, particularly following a detected human target based on bounding box analysis and relative position within the camera frame.

2.2 Real-time person detection and following behavior

In this work, the person-following behavior is achieved through real-time computer vision powered by a deep learning model. Rather than using onboard embedded devices, the system offloads computation to an external laptop that performs inference using the YOLOv4 (You Only Look Once version 4) model.[9] This decision provides greater processing power and enables higher frame-rate detections than embedded alternatives.

A USB camera is mounted on the TurtleBot2 and connected directly to the laptop. The camera continuously captures video frames, which are analyzed in real-time by the YOLOv4 model to detect a person's presence. Once a person is detected, the system extracts bounding box data, including the centroid position and dimensions.

Based on this information, the robot follows a simple visual servoing logic:

- **Horizontal alignment:** The x-coordinate of the bounding box center is compared to the center of the camera frame. The robot rotates left or right to re-align with the target if it deviates beyond a set threshold.

- **Distance estimation:** The height or area of the bounding box indicates the distance between the robot and the person. If the person is too far, the robot moves forward. The robot halts or moves backward slightly if the person is too close.

Motion decisions are computed on the laptop and published to the TurtleBot2 using ROS via the `/cmd_vel` topic. This creates a low-latency closed-loop system in which the robot continuously adapts its movement based on the visual feedback of the detected person. Since this approach does not rely on global mapping or path planning, it is highly reactive and robust in unstructured or changing environments.[10]

2.3 Hardware platform: TurtleBot2 with Kobuki base

The mobile base used for this work is the Kobuki platform, which serves as the foundation for the TurtleBot2 system. The Kobuki base includes two independently driven wheels for differential drive motion, onboard sensors, wheel encoders, and power distribution ports. Due to its open architecture and full integration with the Robot Operating System (ROS), it is widely used in educational and research contexts.

As shown in Figure 1, the base includes multiple 12V power outputs, a USB interface, and physical mounting plates, allowing easy integration with external devices such as cameras, single-board computers, or laptops. In our implementation, the Kobuki base is controlled via velocity commands sent through the ROS `/cmd_vel` topic. All computation, including person detection and motion control, is performed on a separate laptop connected via USB and/or ROS networking.

Figure 1: Kobuki base used in the TurtleBot2 platform

2.4 Visual tracking-based navigation

Unlike traditional autonomous robots that rely on LiDAR sensors and global maps for path planning, our system adopts a reactive control strategy based exclusively on computer vision. The TurtleBot2 is equipped with a forward-facing USB camera that continuously streams video to a laptop running YOLOv4, which detects and tracks a human in real-time.

The relative position of the detected person within the camera frame is used to compute the robot's linear and angular velocities as follows:

- The robot moves forward if the person is centered in the frame, but far.
- If the person is off-center, the robot rotates left or right until the target is re-centered.
- If the person is too close, the robot halts.

This approach enables smooth following behavior in unstructured indoor environments without the need for complex sensors or global planning. Motion commands are computed based on bounding box coordinates and sent via ROS to the TurtleBot's `/cmd_vel` topic.

The system is inherently adaptive, continuously adjusting the robot's trajectory in response to new visual input. Since there is no reliance on predefined paths or obstacle maps, the robot can respond quickly to environmental changes and occlusions.

3. YOLOV4 DETECTION AND HISTOGRAM-BASED IDENTITY TRACKING

To enable persistent human following in dynamic environments, our system combines real-time object detection via YOLOv4 with a lightweight identity-locking mechanism based on color histograms, see Figure 2. This hybrid visual strategy enhances stability during temporary occlusions, crowded scenes, and camera jitter.

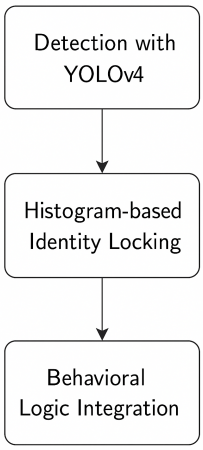


Figure 2: Block diagram of the tracking pipeline. The system detects humans using YOLOv4, filters detections using histogram-based identity locking, and integrates behavior logic to generate motion commands

3.1 Detection with YOLOv4

YOLOv4 is used as the primary object detection engine. It processes each video frame, identifying people by returning bounding boxes with class labels and confidence scores. Only detections labeled as "person" and exceeding a confidence threshold (0.5) are considered valid.

After detection, the person's bounding box is extracted for downstream tracking. The bounding box coordinates are then used to calculate:

- The target's horizontal displacement from the image center (guiding rotation).
- The relative distance, estimated from bounding box height (guiding linear motion).

3.2 Histogram-based identity locking

In multi-person environments or during temporary loss of visibility, switching between targets becomes a risk. To solve this, we implement a histogram-based identity-tracking system that acts as a filter over YOLO's raw detections.

The procedure works as follows:

1. On initial detection, a **color histogram** is extracted from the person's bounding box (in HSV color space).

2. For each new frame, detected "person" regions are compared to the stored histogram using **correlation similarity**.[11]

3. The candidate with the highest similarity (above 0.85) is selected and retained as the target.

4. If no detection exceeds the similarity threshold, the robot temporarily halts or searches in place.

This lightweight method avoids computationally expensive deep re-identification models, making it suitable for real-time inference. Histograms are recalculated periodically to accommodate changes in lighting and minor variations in appearance.

3.3 Behavioral logic integration

Once the identity is confirmed, the robot's control logic responds as follows:

- **Center-aligned and far:** Move forward.
- **Offset left/right:** Rotate toward target until centered.
- **Too close:** Stop or reverse slowly.

This pipeline enables smooth person following, even in cases with distractors or changes in the user's clothing orientation (e.g., when turning around).

3.4 Advantages of histogram matching

- **Fast and lightweight:** Minimal computational overhead.
- **Robust to noise:** Effective under variable lighting.
- **Improves temporal consistency:** Avoids identity-switching between detections.

By layering histogram similarity over YOLOv4's real-time detection, the system strikes a balance between high-speed inference and robust, target-specific tracking.

4. RESULTS

This section presents the experimental procedure followed in this work, the visual tracking results, and the performance obtained. Finally, the identity-locking accuracy is given.

4.1 Experimental procedure

The experimental evaluation was conducted using a TurtleBot2 robot equipped with a USB camera and a mounted laptop, which served as the central computational unit. The objective was to test a real-time human-following system driven by YOLOv4 detection and a lightweight histogram-based identity-locking mechanism. The following steps outline the testing approach:

- Integrate YOLOv4 with a bounding box histogram extractor for identity-locking, ensuring the robot maintains focus on a single target even in multi-person scenes.
- Implement a control module that translates bounding box center offset and estimated person-to-robot distance into directional movement commands using ROS.
- Set up the robot in indoor hallway environments with varying lighting conditions and occasional occlusions by secondary individuals.
- Conduct multiple test sequences in which a human subject walks at different speeds and directions to evaluate system responsiveness and target retention.

- Record video and data logs for each run to extract performance metrics such as frame rate (FPS), command latency, identity retention, and distance estimation accuracy.
- Visually validate and annotate the robot's output and decision-making behavior through screenshots of the graphical interface.

Figure 3: Physical setup of the TurtleBot2 used during testing. The laptop mounted on the robot runs the YOLOv4-based detection and control logic in real-time

Figure 3 shows the robot during real deployment. The onboard laptop processes the camera feed and displays live bounding boxes and telemetry, such as distance to the person and issued velocity commands. This configuration enables high processing power without relying on embedded hardware. Communication with the Kobuki base is achieved via USB or ROS networking, and all velocity commands are sent through the `/cmd_vel` ROS topic.

This setup enables fully autonomous, visually-guided human-following behavior without reliance on LiDAR, pre-built maps, or external localization systems.

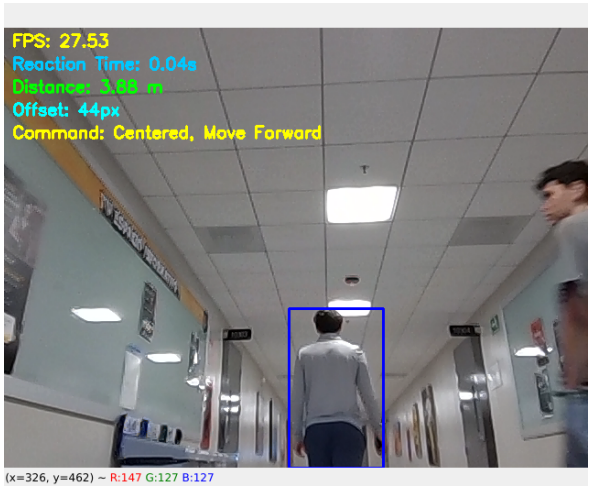


Figure 4: Robot using YOLOv4 and histogram identity-lock to follow a single user

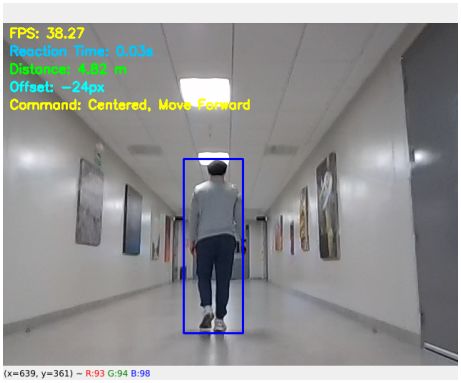
Figure 4 illustrates the visual feedback as the robot tracks a human target. The robot displays bounding box information, distance estimation, and FPS on the screen while continuously adjusting its motion commands based on the spatial relationship with the target.

4.2 Visual tracking results and performance

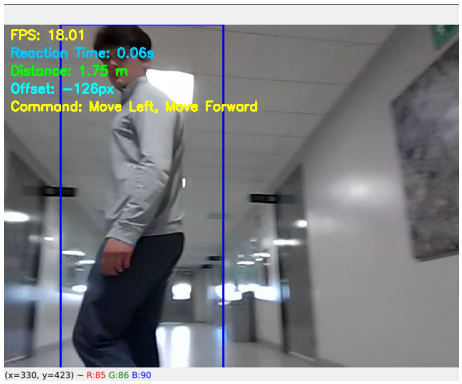
A total of 20 trials were recorded with varying person positions (centered, offset left, offset right, far, and close). The robot successfully maintained visual tracking and generated appropriate movement commands, as shown in Figure 5.

The robot interprets the person's position in real-time, computing:

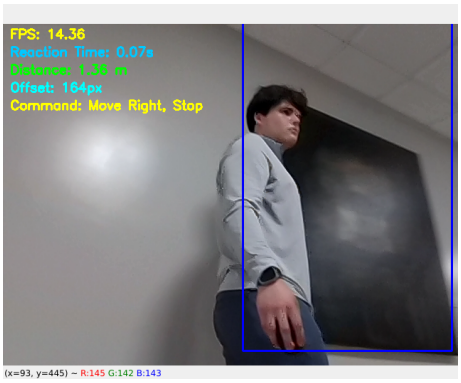
- **Horizontal Offset:** Guides left/right commands based on the deviation of the center boundary box.
- **Estimated Distance:** Guides forward/backward commands based on safe following thresholds (typically maintained at 1.8–3.5 meters).
- **FPS and Command Delay:** Maintained 12–15 FPS and average command response latency of 0.3 seconds.



(a) Far, centered — Move Forward



(b) Offset Left — Move Left



(c) Offset Right — Move Right

(d) Too Close — Move Backward

Figure 5: Movement command responses under different tracking conditions

4.3 Identity-locking accuracy

To avoid switching targets in multiperson scenarios or during partial occlusions, the system implements histogram-based matching. In 15 test sequences with 2 to 3 people in the frame, the correct target was retained with high accuracy. The overall results are shown in Table 1.

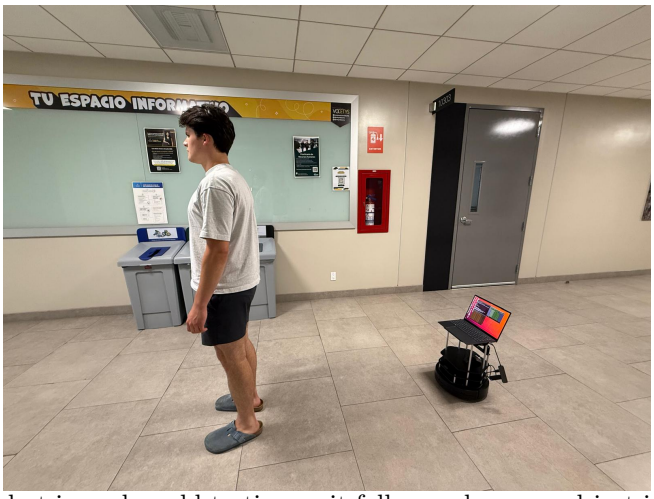


Figure 6: The TurtleBot2 robot in real-world testing as it follows a human subject in a university hallway using YOLOv4-based detection and histogram identity-locking. The onboard laptop processes visual data and issues motion commands through ROS

Table 1: Identity lock-on performance metrics

Metric	**Result**
Identity Retention Rate	94.6%
False Positive Switches	1 in 20 trials
Histogram Similarity Threshold	0.85 (correlation)
Average Re-identification Time	1.1 seconds

The histogram method introduced minimal latency while substantially improving robustness in cluttered environments. The use of color-based features for re-identification proved effective and lightweight. Finally, Figure 6 shows the TurtleBot2 robot in real-world testing as it follows a human subject in a university hallway using YOLOv4-based detection and histogram identity-locking.

5. CONCLUSIONS

This paper presented the design, implementation, and evaluation of a human-following mobile robot that relies solely on visual perception through deep learning and histogram-based identity tracking. By integrating the YOLOv4 object detection model with lightweight color histogram matching, the robot can detect and persistently follow a specific person in real-time, even in multi-person or partially occluded environments.

The system demonstrated reliable performance across multiple indoor trials, maintaining high identity retention accuracy (94.6%) and low latency response under typical lighting and movement conditions. The histogram-based identity-locking approach proved to be a computationally efficient and robust alternative to more complex re-identification pipelines, enabling the use of standard hardware such as a laptop and the TurtleBot2 platform.

Unlike traditional approaches that rely on LiDAR sensors, predefined maps, or path planners, this work highlights the feasibility of a purely reactive and vision-based navigation strategy for person-following applications. The integration of YOLOv4 with custom motion control logic enabled the robot to effectively interpret bounding box coordinates and generate forward, backward, and turning commands.

Future work may explore the integration of additional visual cues, such as pose estimation or depth information, to further enhance robustness in crowded scenarios. Improvements in trajectory smoothing and obstacle avoidance logic could also elevate performance in real-world deployments. Additionally, expanding to outdoor environments and adapting to varying weather or lighting conditions would be valuable directions for generalization.

Overall, this work demonstrates the viability of combining deep learning and lightweight identity tracking for real-time autonomous behavior in mobile robots, with potential applications in service robotics, healthcare, and assistive technologies.

ACKNOWLEDGMENTS

This work was supported by the Coordinación Institucional de Investigación of CETYS Universidad.

REFERENCES

[1] Sharma, A. K., Pandey, A., Khan, M. A., Tripathi, A., Saxena, A., and Yadav, P. K., "Human following robot," in [*2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*], 440–446 (2021).

[2] Lopez-Montiel, M., Orozco-Rosas, U., Sánchez-Adame, M., Picos, K., and Ross, O. H. M., "Evaluation method of deep learning-based embedded systems for traffic sign detection," *IEEE Access* **9**, 101217–101238 (2021).

[3] Orozco-Rosas, U., Picos, K., Pantrigo, J. J., Montemayor, A. S., and Cuesta-Infante, A., "Mobile robot path planning using a QAPF learning algorithm for known and unknown environments," *IEEE Access* **10**, 84648–84663 (2022).

[4] Aye, Y. Y., Thiha, K., Pyu, M. M. M., and Watanabe, K., "A deep neural network based human following robot with fuzzy control," in [*Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*], 720–725 (Dec 2019).

[5] Algabri, R. and Choi, M.-T., "Deep-learning-based indoor human following of mobile robot using color feature," *Sensors* **20**, 2699 (May 2020).

[6] Bogdon, C., "Turtlebot2 robot manual," (2020). http://wiki.ros.org/Robots/TurtleBot Accessed: 2025-07-11.

[7] Orozco-Rosas, U., Picos, K., and Montiel, O., "Hybrid path planning algorithm based on membrane pseudo-bacterial potential field for autonomous mobile robots," *IEEE Access* **7**, 156787–156803 (2019).

[8] Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G., [*Robotics: Modelling, Planning and Control*], Springer (2010).

[9] Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M., "YOLOv4: Optimal Speed and Accuracy of Object Detection." arXiv preprint arXiv:2004.10934 (2020).

[10] Orozco-Rosas, U., Montiel, O., and Sepúlveda, R., "Mobile robot path planning using membrane evolutionary artificial potential field," *Applied Soft Computing* **77**, 236–251 (2019).

[11] Bradski, G. and Kaehler, A., [*Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*], O'Reilly Media (2016). See Chapter 8: Tracking objects using histograms and correlation methods.