IntelliDrive autonomous robot powered by large language model

Imran Ulla Khan[1,2], D. R. Kumar Raja[3]
[1]School of Computer Science, REVA University, Bangalore, India
[2]Department of CSE, Sri Krishna Institute of Technology, Bangalore, India
[3]School of Computer Science, REVA University, Bangalore, India

Article Info

Article history:

Received Mar 15, 2025
Revised Jun 14, 2025
Accepted Jul 10, 2025

Keywords:

Artificial intelligence
Autonomous robot
Deep learning
Large language models
Robotics

ABSTRACT

The rapid advancements in artificial intelligence (AI) and robotics have paved the way for innovative autonomous systems capable of performing complex tasks. This project integrates robotics with Large Language Models (LLMs) to develop an intelligent, versatile and user-friendly robotic system. The robot is designed to interpret structured commands, make real-time decisions, and navigate autonomously in dynamic environments, addressing key challenges faced by traditional autonomous systems. Central to the system is a Raspberry Pi 4, which serves as the main processing unit, integrating components such as a webcam for visual data capture, an L298N motor driver for motor control, and a Bluetooth speaker for real-time feedback. The LLM API enables the robot to process natural language commands, providing context-aware task execution and adaptability to changing scenarios. Testing has demonstrated the system’s ability to perform autonomous navigation, detect obstacles, and execute tasks effectively. This research offers a foundation for various industries, including logistics, healthcare, education, and hazardous environment operations. By incorporating LLMs the robot overcomes limitations of traditional rule-based systems, enhancing dynamic decision-making and user interaction. With its modular design and scalability, it bridges the gap between human-like intelligence and mechanical precision, setting the stage for future advancements in AI-driven robotics.

Corresponding Author:

Imran Ulla Khan
School of Computer Science, REVA University
Bangalore, India
Email: imran161984@gmail.com

1. INTRODUCTION

The field of robotics has witnessed significant advancements over the years, transitioning from simple mechanical machines to highly sophisticated systems powered by artificial intelligence (AI). Autonomous robotics, in particular, has emerged as a crucial area of innovation, addressing challenges in automation, precision, and efficiency across various industries [1], [2]. These robots are designed to perform tasks independently, reducing human intervention and enhancing productivity. The integration of AI, specifically large language models (LLMs), has further expanded the scope of robotics by enabling natural language processing, adaptive decision-making, and improved human-robot interaction [3], [4].

Traditional robotics often relies on pre-defined algorithms and sensor-based systems, which limit their adaptability and effectiveness in dynamic environments. These systems are typically rigid, requiring extensive programming for each new task or environment. As a result, they struggle to handle real-world scenarios that demand flexibility and quick decision- making. Moreover, the lack of real-time adaptability in these systems creates barriers for wider adoption in industries that require diverse and variable operations. In this research we have addresses these limitations by integrating an LLM into the robot's control system [5]–[7]. The primary challenge is to design a robot capable of interpreting and processing natural language commands accurately [8], [9], making intelligent decisions in real-time based on environmental inputs [10], and executing tasks autonomously with minimal human intervention [11].

By bridging the gap between traditional robotics and artificial intelligence, this system aims to unlock the potential of autonomous machines in solving complex real-world problems. The development of such systems could lead to smarter, more versatile robots with widespread applications [12], [13]. Autonomous robotics stands at the crossroads of multidisciplinary fields, including engineering, computer science, and cognitive systems. The overarching goal is to create machines capable of responding to complex environments, adapting to new tasks, and collaborating with humans seamlessly. With the rise of LLMs, the intersection between natural language understanding and mechanical actuation has brought forth new dimensions of interaction [14], [15]. This study explores the integration of LLM-based control with computer vision and motor actuation to achieve robust real-time adaptability and human-like interaction.

The proposed project leverages these advancements to create a system that combines hardware precision with AI-driven intelligence. This project aims to overcome the limitations of traditional sensor-based robots by incorporating real-time decision-making capabilities powered by LLMs. The key finding of this study is that integrating large language models (LLMs) with autonomous robotics significantly enhances the robot's ability to interpret natural language commands and make real-time decisions. This approach improves adaptability, user interaction, and autonomous navigation compared to traditional rule-based systems.

The literature survey as shown in Table 1 explores advancements in robotics, including LLM-based decision-making, machine vision for navigation, and modular system architectures. The discussion highlights gaps such as the lack of hardware- software integration, limited scalability, and inadequate adaptability in dynamic environments. Research insights include studies on GPT models for natural language processing, YOLO-based object detection, and frameworks for autonomous decision- making, forming the foundation for the proposed system.

Table 1. Comparative analysis

Title	Command interpretation accuracy (%)	Real-time decision latency (ms)	Navigation efficiency (%)	Error rate in obstacle detection (%)
IntelliDrive Autonomous Robot	89	170	88	5
ChatGPT-Controlled Robot [16]	89	160	80	8
Reinforcement Learning-based Robot [17]	90	160	85	7
Deep Learning Autonomous Driving Robot [18]	88	180	82	6

This paper introduce an innovative approach that merges large language models (LLMs) with behavior trees (BTs). This method dynamically adapts robotic tasks to environmental changes by leveraging ChatGPT for real- time reasoning and a semantic mapping framework for task execution. The use of LLMs enhances the ability of BTs to handle unforeseen events, improving adaptability and robustness in task management [19], [20]. Bharathi *et al.* [21] explored the transformative role of machine vision in robotic systems. The study highlights key technologies such as convolutional neural networks (CNNs) for real- time object detection and simultaneous localization and mapping (SLAM) for autonomous navigation. Applications include defect detection, quality assurance, and navigation in manufacturing and healthcare, emphasizing the precision and adaptability offered by machine vision.

This research investigate the integration of LLMs and generative AI (Gen AI) in humanoid robots. This hybrid model significantly enhances natural language processing and emotional intelligence, enabling intuitive interactions with humans. The research focuses on ethical AI deployment to assist vulnerable populations such as the elderly and disabled [22]. This framework automates complex robotic development tasks, including code generation and parameter tuning. The study demonstrates the framework's efficacy in simplifying robotics development for non-experts, validated through experiments on quadruped robots [23].

2. PROPOSED METHOD

Existing autonomous robotic systems are often limited by their reliance on predefined algorithms, sensor data, and rule-based mechanisms. These systems lack flexibility and adaptability in dynamic environments, as they depend heavily on pre-programmed instructions and specific sensor configurations.

The proposed system addresses the limitations of existing systems by integrating hardware precision with advanced AI capabilities. The system leverages large language models (LLMs) to enable natural language command interpretation and adaptive decision-making. Below are the key features of the proposed system [24].

a. The robot integrates LLM to understand and execute complex natural language commands.
b. It processes environmental inputs to generate motor control commands dynamically, enabling real-time decision-making.
c. Cost-effective hardware, including a Raspberry Pi, BO motors, and a basic webcam, ensures affordability without compromising functionality.
d. Scalable, modular architecture allows for the integration of additional features like advanced sensors or navigation algorithms.
e. A user-friendly interface enhances usability, making it accessible for non-technical users

By combining advanced AI with efficient hardware, the proposed system aims to deliver a flexible, reliable, and scalable solution for autonomous robotics. Figure 1 Illustrates the architecture of an autonomous robot powered by natural language processing and real-time adaptability. The system is structured into three primary components: input, processing, and output. The Input Components consist of a webcam for capturing real-time visual data and a user interface for receiving natural language commands. The webcam encoder converts visual data into a Base64 format, which is transmitted to the processing unit. The Processing Unit comprises a Raspberry Pi that acts as the central controller [25], [26]. It processes visual inputs and user commands with the aid of an LLM API, which interprets and generates appropriate responses. These responses are then translated into actionable control signals. The Output Components include an L298N motor driver, which converts the control signals into precise instructions for BO motors, enabling movement.

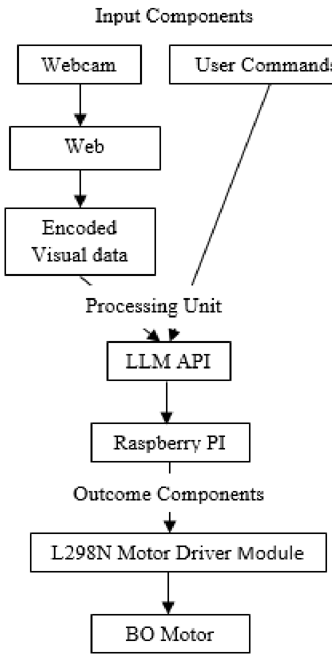


Figure 1. System architecture

3. METHOD

The development of the IntelliDrive LLM-powered autonomous rover involves integrating natural language processing, real-time decision-making, and modular robotics to create an intelligent, scalable system. The methodology starts with selecting cost-effective hardware such as a Raspberry Pi 4, a webcam, BO motors, and an L298N motor driver. The system architecture is designed around three primary subsystems.

3.1. Input for capturing visual and command data

The input subsystem is responsible for capturing environmental and user data necessary for the robot's operation. It includes components like a webcam for visual input and a user command interface for natural language instructions. The visual input from the webcam is represented as a 2D image matrix $I(x, y)$, where each pixel $I(x, y)$ contains RGB or grayscale intensity values.

3.2. Processing for decision-making using an LLM API

The processing subsystem is the computational core of the architecture, performing decision-making based on the input data. It includes the Raspberry Pi 4 and LLM API to interpret natural language commands, This API excel in processing natural language and converting it into structured actions, such as "Move forward 5 meters" into motor commands and make real-time decisions for navigation and task execution.

The robot's linear velocity (v) and angular velocity (ω) are calculated based on the rotational speeds of its wheels. Linear velocity represents the forward or backward movement speed, while angular velocity measures the rate of rotation around the robot's center. By controlling the individual speeds of the left and right wheels, the robot can achieve precise linear and angular motion, allowing it to move straight, turn in place, or follow curved trajectories as in (1) and (2).

$$v = \frac{R}{2}(\omega_L + \omega_R) \quad (1)$$

$$v = \frac{R}{2}(\omega_L - \omega_R) \quad (2)$$

Where R is the wheel radius, L is the distance between the wheels and ω_L, ω_R are the angular velocities of the left and right wheels.

The robot relies on a webcam and algorithms like YOLO for object detection, which identifies obstacles, avoid them and navigates paths in real-time. Once YOLOv5 detects obstacles, vector-based algorithms compute a safe trajectory by balancing attraction toward the goal and repulsion from obstacles [27], [28] as shown in (3).

$$V_{safe} = V_{goal} + V_{repulsion} \quad (3)$$

V_{safe} A vector pointing toward the target/destination, $V_{repulsion}$ A vector pointing away from detected obstacles to avoid collisions.

Precise movements are achieved using PID controllers as in (4), which dynamically adjust motor speed and direction to maintain stability and follow trajectories. The PID controller adjusts motor signals for accurate movement.

$$S = K_p.e + K_i \int edt + K_d \frac{de}{dt} \quad (4)$$

where e is the error between the desired and actual position or speed.

A* (A-Star) algorithm is employed to calculate the shortest path as in (5) from the start to the goal while avoiding obstacles. This algorithm ensure that the robot navigates complex environments safely and effectively.

$$f(n) = g(n) + h(n) \quad (5)$$

where $g(n)$ is the cost to reach the node, and $h(n)$ is the heuristic estimate of the cost to the goal.

3.3. Output for executing motor commands

The output subsystem executes the actions decided by the processing unit. It includes the L298N motor driver and BO motors for movement, as well as any feedback mechanisms like a Bluetooth speaker for user interaction. The motor's rotational speed and direction are directly controlled by pulse width modulation (PWM) signals using (6). The preprocessing actions are executed with precise motor control and provides feedback for continuous refinement.

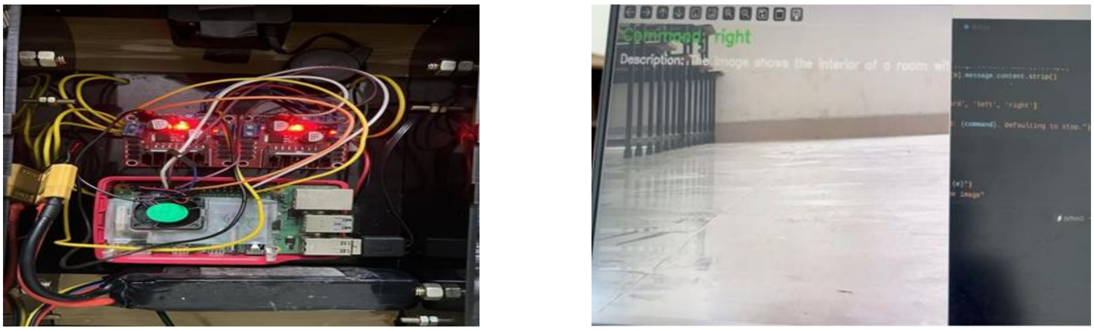
$$\text{PWM } Duty \text{ Cycle} = \frac{Desigred\ Speed}{Max\ Speed}\ x\ 100 \quad (6)$$

3.4. Implementation

The implementation phase involves translating the system design into a working model through coding, hardware integration, and software configuration. The proposed system is implemented using a modular approach, ensuring that individual components are developed and tested independently before integration.

3.4.1. Hardware Implementation

The hardware implementation of the IntelliDrive Autonomous Rover involved carefully assembling and integrating its physical components to ensure seamless interaction as shown in Figure 2(a). A sturdy four-wheeled chassis was used to house the components, with BO motors securely mounted to provide mobility. The L298N motor driver was connected to the Raspberry Pi GPIO pins and BO motors, with PWM signals configured for precise speed and direction control. As shown in Figure 2(b) a webcam was mounted at an optimal angle to capture the environment, enabling effective image processing for navigation. A computer screen showing a live camera feed with code for controlling a robot, displaying a command interface. The system was powered by a rechargeable lithium-ion battery pack, which supplied energy to the Raspberry Pi, motor driver, and motors.



(a) (b)

Figure 2. Hardware implementation (a) robot internal circuit and (b) camera capturing environment

3.4.2. Software implementation

The software implementation of the IntelliDrive autonomous rover centered on programming the system to process inputs, interact with the LLM API, and control the robot's movements. The Raspberry Pi was configured by installing Raspberry Pi OS along with essential libraries such as OpenCV for image processing, RPi.GPIO for hardware control, and the LLM API client for natural language processing. The LLM API was integrated to interpret user commands, with secure authentication tokens and endpoints established for seamless communication. OpenCV was utilized to process video feeds from the webcam, enabling real-time navigation and obstacle detection as shown in Figure 3(a) with YOLOv5 by capturing and analyzing frames to generate navigation inputs. Python scripts were developed to translate structured responses from the LLM API into motor driver commands, implementing functions for forward, backward, left, and right movements. This integrated software system ensured the robot's ability to understand commands, interpret environmental data, and execute precise movements effectively. As shown in Figure 3(b) there is another angle of the robot chassis, focusing on the wheels and motor connections avoiding the obstacle.

(a) (b)

Figure 3. Software implementation (a) detecting an object and (b) avoiding the obstacle

3.4.3. Offline fallback mechanisms

The robot faces processing delays during LLM-based command execution, which impacted real-time responsiveness. It relies on stable internet connectivity for cloud-based LLM APIs, making it vulnerable to network disruptions. System inaccurately interpreted commands, leading to erroneous actions and compromising operational safety. Network latency also disrupted real-time navigation. To address network latency and connectivity challenges, we have implemented several offline fallback mechanisms to ensure consistent performance. Command caching allows frequently used or critical commands (such as "move forward," "turn left," and "stop") to be stored locally, enabling instant responses even during connectivity issues. A hybrid processing approach leverages lightweight pre-trained models for essential commands while reserving cloud processing for more complex tasks, reducing latency without sacrificing advanced capabilities. We have also incorporated preemptive command execution by predicting the next likely command during network instability and preparing it locally to minimize response time.

4. RESULTS AND DISCUSSION

Testing is a crucial phase in the development process that ensures the system performs as expected and meets the defined requirements. Our "Autonomous Robot with LLM-Based Control" underwent rigorous testing to validate its functionality, reliability, and performance. Here we have discussed the various types of testing conducted, including unit testing, integration testing, functional testing and power consumption analysis.

4.1. Unit test

Unit testing as mentioned in Table 2 was conducted to verify the functionality of individual components in isolation and ensure their reliability. Motor control testing confirmed the BO motors responded accurately to PWM signals from the Raspberry Pi, with speed and direction control functioning as expected. The webcam was tested to ensure it captured real-time video feeds without latency, with frame resolution and clarity suitable for image processing tasks. The LLM API was evaluated by sending sample natural language commands and verifying the responses, including testing edge cases with ambiguous or incomplete instructions. All components passed their respective unit tests with minimal errors, and issues such as frame delays were identified and promptly resolved, ensuring robust system performance.

Table 2. Unit testing

Test Case Attribute	TC_U01_01	TC_U01_02
Description	Motor Direction Testing	Motor Direction Testing
Input	PWM signals to motors	PWM signals to motors
Expected Output	Motors rotate in the correct direction.	Motors rotate in the correct direction.
Actual Result (Initial)	Motors rotated in the wrong direction.	Motors rotated correctly.
Test Case	Fail	Pass
Troubleshooting	Rewired motor connections to match polarity.	None required.
Actual Result (Final)	Pass	Pass

4.2. Integration test

Integration testing as shown in Table 3 was performed to ensure smooth interaction between the hardware and software components of the system. The synchronization of motor drivers with Raspberry Pi GPIO pins was verified, ensuring motor control commands were executed correctly based on LLM-generated responses. The interaction between the LLM API and the Raspberry Pi was tested for accurate command interpretation and task execution. OpenCV was integrated with the webcam to process video feeds and identify obstacles, with real-time data transfer between the camera and processing scripts successfully validated. All integrated systems functioned seamlessly after minor adjustments, and latency issues during command processing were optimized for improved performance.

Table 3. Integration testing

Test Case Attribute	TC_U01_01	TC_U01_02
Description	Command processing integration	Command processing Integration
Input	Motor control commands	Motor control commands
Expected Output	Motors respond to commands generated by LLM.	Motors respond to commands generated by LLM.
Actual Result (Initial)	Motors did not respond as expected.	Motors responded correctly.
Test Case	Fail	Pass
Troubleshooting	Debugged GPIO control and LLM parsing.	None required.
Actual Result (Final)	Pass	Pass

4.3. Functional test

Functional testing as shown in Table 4 was conducted to validate that the system met its requirements and performed expected tasks in real-world scenarios. Navigation commands such as “move forward,” “turn left,” and “stop” were tested to confirm the robot's ability to follow user instructions accurately. Obstacle detection was evaluated by placing obstacles in the robot’s path and verifying its capability to avoid them, including adapting to dynamic changes in the environment. The system’s ability to interpret and execute complex commands, such as “turn right and move forward 3 steps,” was assessed, along with fallback mechanisms for ambiguous or invalid instructions. Results showed the system effectively navigated, interpreted commands, and avoided obstacles, while edge cases with incomplete commands were gracefully handled by providing appropriate error messages.

Table 4. Functional testing

Test Case Attribute	TC_U01_01	TC_U01_02
Description	Navigation Commands Execution	Navigation Commands Execution
Input	Commands like “move left”	Commands like “move left”
Expected Output	Robot follows instructions accurately.	Robot follows instructions accurately.
Actual Result (Initial)	Robot misinterpreted the commands.	Robot followed commands perfectly.
Test Case	Fail	Pass
Troubleshooting	Improved command parsing logic in LLM.	None required.
Actual Result (Final)	Pass	Pass

4.4. Power consumption analysis

We conducted a power consumption analysis by evaluating the energy usage of each component, including the Raspberry Pi, motors, camera, and LLM API requests shown in Table 5. We calculated the power draw during continuous and intermittent operation to estimate battery life accurately. This analysis helped us to identify energy-intensive components and optimize the system for prolonged autonomous operation.

Table 5. Power consumption analysis

Component	Power Consumption (W)	Usage Type
Raspberry Pi 4	6	Continuous
BO Motors (x4)	1.5 each	During movement
Webcam	2	Continuous
L298N Motor Driver	1.5	During movement
LLM API Requests	1	Intermittent

We have incorporated power-saving mechanisms to enhance the energy efficiency of the autonomous robot. Low-power modes are configured to reduce the CPU clock speed during low computational demand and to deactivate the camera when not in use. Duty cycling is employed to alternate between active and sleep states for components that are not continuously required, such as activating the webcam only when an obstacle is suspected.

5. CONCLUSION

The proposed system showcases the seamless integration of advanced AI and robotics to address real-world challenges. Leveraging large language models (LLMs) for command interpretation and decision-making, the system overcomes the limitations of traditional autonomous robots by enabling adaptability and dynamic responses. Its modular architecture, comprising components like the Raspberry Pi, webcam, and motor drivers, supports reliable navigation, obstacle detection, and user-friendly interaction. Rigorous testing validated the system’s robustness, making it suitable for applications in logistics, healthcare, education, and hazardous operations. Looking ahead, the robot’s scalability allows for further enhancements.

ACKNOWLEDGMENTS

We are grateful to REVA University for providing the necessary resources and facilities for this study.

FUNDING INFORMATION

We, the authors, declare that no funding was received for this work.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Imran Ulla Khan	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	
D. R. Kumar Raja					✓		✓	✓		✓		✓	✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] Z. Orpek, B. Tural, and Z. Destan, "The language model revolution: LLM and SLM analysis," in *8th International Artificial Intelligence and Data Processing Symposium, IDAP 2024*, Sep. 2024, vol. 3, pp. 1–4. doi: 10.1109/IDAP64064.2024.10710677.

[2] H. Lin, J. Wang, Y. Zhan, and Y. Zhang, "Tour guide robot based on large language model and user behavior analysis," in *2024 5th International Conference on Artificial Intelligence and Computer Engineering, ICAICE 2024*, Nov. 2024, pp. 794–798. doi: 10.1109/ICAICE63571.2024.10864058.

[3] C. Y. Kim, C. P. Lee, and B. Mutlu, "Understanding large-language model (LLM)-powered human-robot interaction," in *ACM/IEEE International Conference on Human-Robot Interaction*, Mar. 2024, pp. 371–380. doi: 10.1145/3610977.3634966.

[4] S. H. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor, "ChatGPT for robotics: Design principles and model abilities," *IEEE Access*, vol. 12, pp. 55682–55696, 2024, doi: 10.1109/ACCESS.2024.3387941.

[5] D. M. Biji and Y. W. Kim, "Evaluating the performance of large language models in classifying numerical data," in *International Conference on ICT Convergence*, Oct. 2024, pp. 840–844. doi: 10.1109/ICTC62082.2024.10827495.

[6] L. Grassi, C. T. Recchiuto, and A. Sgorbissa, "Enhancing LLM-based human-robot interaction with nuances for diversity awareness," in *IEEE International Workshop on Robot and Human Communication, RO-MAN*, Aug. 2024, pp. 2287–2294. doi: 10.1109/RO-MAN60168.2024.10731381.

[7] S. Shibi and S. Zaidi, "STEM approach to enhance robot-human interaction through AI large language models and reinforcement learning," in *2024 IEEE Integrated STEM Education Conference, ISEC 2024*, Mar. 2024, pp. 1–2. doi: 10.1109/ISEC61299.2024.10665163.

[8] A. Bucker, L. Figueredo, S. Haddadinl, A. Kapoor, S. Ma, and R. Bonatti, "Reshaping robot trajectories using natural language commands: A study of multi-modal data alignment using transformers," in *IEEE International Conference on Intelligent Robots and Systems*, Oct. 2022, pp. 978–984. doi: 10.1109/IROS47612.2022.9981810.

[9] S. Ortuno-Chanelo, L. Contreras, J. Savage, and H. Okada, "Keep it simple: Understanding natural language commands for general-purpose service robots," in *2024 IEEE/SICE International Symposium on System Integration, SII 2024*, Jan. 2024, pp. 1320–1325. doi: 10.1109/SII58957.2024.10417341.

[10] Y. Xia, Q. Li, R. Huang, and X. Zhao, "Design of intelligent medical service robot based on Raspberry Pi and STM32," in *IEEE Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, Jun. 2022, pp. 1577–1581. doi: 10.1109/ITAIC54216.2022.9836515.

[11] S. Joy, R. L. Paulraj, M. Punith, M. Shalini, S. Goudar, and S. Ruthesh, "A Raspberry Pi-based smart security patrol robot," in *Proceedings - 7th International Conference on Computing Methodologies and Communication, ICCMC 2023*, Feb. 2023, pp. 1140–1145. doi: 10.1109/ICCMC56507.2023.10083908.

[12] I. Hameem Shanavas, P. B. Reddy, and M. C. Doddegowda, "A personal assistant robot using raspberry Pi," in *Proceedings - 2018 International Conference on Design Innovations for 3Cs Compute Communicate Control, ICDI3C 2018*, Apr. 2018, pp. 133–136. doi: 10.1109/ICDI3C.2018.00038.

[13] K. S. Chinmay and G. K. Nisha, "Review of autonomous mobile robots," in *International Conference on E-Mobility, Power Control and Smart Systems: Futuristic Technologies for Sustainable Solutions, ICEMPS 2024*, Apr. 2024, pp. 1–5. doi: 10.1109/ICEMPS60684.2024.10559359.

[14] J. Wu *et al.*, "TidyBot: Personalized robot assistance with large language models," in *IEEE International Conference on Intelligent Robots and Systems*, Oct. 2023, pp. 3546–3553. doi: 10.1109/IROS55552.2023.10341577.

[15] Z. Zhao, S. Lou, R. Tan, and C. Lv, "An AR-assisted human-robot interaction system for improving LLM-based robot control," in *Proceedings of the IEEE International Conference on Cybernetics and Intelligent Systems, CIS*, Aug. 2024, no. 2024, pp. 144–149. doi: 10.1109/CIS-RAM61939.2024.10673005.

[16] S. H. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor, "ChatGPT for robotics: Design principles and model abilitiess," *IEEE Access*, vol. 12, pp. 55682–55696, 2024, doi: 10.1109/ACCESS.2024.3387941.

[17] A. Slavova and V. Hristov, "Application of reinforcement learning in autonomous mobile robots," in *National Conference with International Participation, TELECOM*, Nov. 2024, pp. 1–4. doi: 10.1109/TELECOM63374.2024.10812227.
[18] S. Paniego, N. Paliwal, and J. M. Canas, "Model optimization in deep learning based robot control for autonomous driving," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 715–722, Jan. 2024, doi: 10.1109/LRA.2023.3336244.
[19] H. Liu *et al.*, "Enhancing the LLM-based robot manipulation through human-robot collaboration," *IEEE Robotics and Automation Letters*, vol. 9, no. 8, pp. 6904–6911, Aug. 2024, doi: 10.1109/LRA.2024.3415931.
[20] H. Zhou, Y. Lin, L. Yan, J. Zhu, and H. Min, "LLM-BT: Performing robotic adaptive tasks based on large language models and behavior trees," in *Proceedings - IEEE International Conference on Robotics and Automation*, May 2024, pp. 16655–16661. doi: 10.1109/ICRA57147.2024.10610183.
[21] G. P. Bharathi, P. Kavitha, L. B. Abhang, G. Rathinasabapathi, V. Sharma, and T. Sathish, "Machine vision applications in robotics and automation," in *Proceedings of 9th International Conference on Science, Technology, Engineering and Mathematics: The Role of Emerging Technologies in Digital Transformation, ICONSTEM 2024*, Apr. 2024, pp. 1–5. doi: 10.1109/ICONSTEM60960.2024.10568755.
[22] G. S. Murugesan and S. A. Viswanathan, "Enhancing human-machine interaction: A study on deployment of LLM and Gen AI hybrid models in responsible humanoids for human assistance," in *2024 15th International Conference on Computing Communication and Networking Technologies, ICCCNT 2024*, Jun. 2024, pp. 1–6. doi: 10.1109/ICCCNT61001.2024.10724819.
[23] Z. Luan *et al.*, "Automatic robotic development through collaborative framework by large language models," in *Proceedings - 2023 China Automation Congress, CAC 2023*, Nov. 2023, pp. 7736–7741. doi: 10.1109/CAC59555.2023.10450432.
[24] G. Siwach and C. Li, "Enhancing human cobot interaction using natural language processing," in *2023 IEEE 4th International Multidisciplinary Conference on Engineering Technology, IMCET 2023*, Dec. 2023, pp. 21–26. doi: 10.1109/IMCET59736.2023.10368263.
[25] R. Farkh, G. Oudinet, F. Sananes, S. Alzahrani, S. Alhuwaimel, and T. De Leruyelle, "Intelligent visual tracking robots using cascade PID controllers," in *SCC 2023 - IEEE 3rd International Conference on Signal, Control and Communication*, Dec. 2023, pp. 1–6. doi: 10.1109/SCC59637.2023.10527483.
[26] L. M. Patarroyo Godoy, S. J. Salinas, and J. A. Chaparro, "Machine learning techniques used in the design of an automatic terrain identifier for a rover robot," in *IEEE Andescon, ANDESCON 2024 - Proceedings*, Sep. 2024, pp. 1–6. doi: 10.1109/ANDESCON61840.2024.10755543.
[27] A. Anish, R. Sharan, A. Hema Malini, and T. Archana, "Enhancing surveillance systems with YOLO algorithm for real-time object detection and tracking," in *2nd International Conference on Automation, Computing and Renewable Systems, ICACRS 2023 - Proceedings*, Dec. 2023, pp. 1254–1257. doi: 10.1109/ICACRS58579.2023.10404710.
[28] S. Wang, Y. Duan, H. Ding, Y. P. Tan, K. H. Yap, and J. Yuan, "Learning transferable human-object interaction detector with natural language supervision," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2022, vol. 2022-June, pp. 929–938. doi: 10.1109/CVPR52688.2022.00101.

BIOGRAPHIES OF AUTHORS

Imran Ulla Khan is a research scholar at Reva University and an assistant professor in the Department of Computer Science and Engineering at Sri Krishna Institute of Technology, Bangalore, India. He is a tech enthusiast with a keen interest in image processing and deep learning. His research focuses on sketch-based image retrieval and generative adversarial networks (GANs) for enhancing sketch-to-realistic image transformation. Imran has presented and published his work in various conferences and journals. He is passionate about advancing his expertise in deep learning architectures to address challenges in image synthesis and related domains. He can be contacted at imran161984@gmail.com.

Dr. D. R. Kumar Raja is currently working as professor in the School of Computer Science and Engineering at Reva University, Bengaluru, Karnataka, India. He received his Bachelor of Technology (B. Tech) from JNTUA College of Engineering and Master of Technology (M. Tech) from National Institute of Technology Karnataka (NITK), Surathkal, Karnataka, India. He received a Doctor of Philosophy (Ph.D.) from St. Peters University, Chennai, India for An effective context-driven recommender system for e-commerce applications. He did post-doctoral research at Universiti Teknikal Malaysia, Melaka, Malaysia from September 2023 to September 2024. He received funding amounting to 18,000 USD for carrying out research projects: one for 16,000 USD from Reva University for the project Humanoid Robot, completed in 2022, and another for 2,000 USD from Sree Vidyanikethan Educational Trust for the project Automation of aerators for aquaculture using IoT, completed in 2019. His research areas include the Internet of Things, data mining, machine learning, and artificial intelligence. He can be contacted at kumarrajadr@gmail.com.