LocusEdge

Sreekar Praneeth Marri
Sauman Raaj Aarthi Ganesh Vetrivel
Anirudh Sundaresan
Yukkta Seelam

EK505, Fall 2024

1 Introduction

The field of mobile robotics and artificial intelligence has experienced rapid advancements, offering innovative solutions to address complex real-world challenges. However, a significant limitation persists: most mobile robots currently rely heavily on cloud-based large language models (LLMs) for natural language processing (NLP)[3]. This dependency creates several challenges, including latency, privacy risks, and limited adaptability in dynamic environments. As a result, the full potential of mobile robots in critical applications remains unexplored.Locus Edge introduces a transformative approach by integrating fine-tuned, quantized offline LLMs designed specifically for edge devices. By enabling mobile robots to operate independently of cloud connectivity, Locus Edge overcomes key challenges associated with real-time performance, data privacy, and reliability. This independence ensures seamless operation even in environments with limited or no network access, making it ideal for critical applications such as surveillance, search and rescue, and healthcare.

The use of offline LLMs provides several advantages, including faster response times, enhanced security through local data processing, and greater flexibility in adapting to specific tasks and environments. Moreover, the system is optimized for edge devices, balancing computational efficiency and accuracy without expensive hardware upgrades.Furthermore, the integration of quantized offline LLMs ensures that the models are lightweight yet highly capable. This innovation not only reduces energy consumption but also extends the operational lifespan of edge devices, making it an environmentally sustainable choice. By prioritizing energy efficiency and cost-effectiveness, LocusEdge aligns with global efforts toward green and sustainable technology[8].

LocusEdge's motivation lies in empowering mobile robots to execute complex tasks in critical and hazardous environments with minimal human intervention. By addressing challenges related to latency, data privacy, and adaptability, the project aims to minimize risks to human safety while delivering efficient and reliable solutions. Additionally, the integration of lightweight, quantized LLMs ensures energy efficiency and environmental sustainability, extending the lifespan of edge devices and reducing costs. This approach sets the foundation for deploying advanced robotics in domains where cloud-dependent systems fall short, such as disaster zones, industrial sites, and remote connectivity-deprived areas.By redefining the capabilities of mobile robotics, LocusEdge bridges existing gaps and establishes a scalable, efficient, and privacy-focused framework for tackling critical real-world problems in diverse and demanding environments.

2 Impact

The LocusEdge project is poised to redefine the landscape of mobile robotics by addressing critical challenges associated with cloud dependency, latency, and privacy concerns. By integrating fine-tuned, quantized offline LLMs optimized for edge devices, our solution delivers a range of impactful benefits across various applications:

- Revolutionizing Disaster Response: Offline LLM-driven mobile robots can operate in disaster zones where internet connectivity is unreliable or nonexistent. These robots can assist in search-and-rescue operations, mapping hazardous environments, and delivering essential supplies, minimizing human risk while ensuring reliable performance in critical scenarios.
- Enhancing Privacy and Security: By processing all data locally, LocusEdge ensures that sensitive information remains secure, addressing privacy concerns that are prevalent in cloud-dependent systems. This feature is particularly beneficial in applications like healthcare and surveillance, where data confidentiality is paramount[6].
- Improving Real-Time Performance: The elimination of cloud latency enables mobile robots to perform real-time decision-making and task execution, significantly enhancing their efficiency and reliability in dynamic environments. This makes them well-suited for operations in logistics, industrial automation, and other high-stakes scenarios.
- Expanding Deployment Possibilities: LocusEdge bridges the gap between existing cloud-based systems and offline models, making it possible to deploy mobile robots in remote or extreme environments, such as rural

areas, industrial sites, or hazardous zones, where cloud-based solutions would fail[3].

- Minimizing Human Risk: By leveraging offline LLMs, mobile robots can execute complex tasks autonomously, reducing the need for human intervention in dangerous or high-risk environments. This capability is critical for applications like surveillance, hazardous material handling, and disaster management[1].

Through these contributions, LocusEdge not only addresses the existing gaps in mobile robotics, but also establishes a scalable, efficient, and privacy-centric foundation for future innovations. The project's impact extends beyond technology, enabling safer, more reliable, and autonomous operations in real-world applications.

3 Market Trends

To gain a comprehensive understanding of the current landscape in mobile robotics using Large Language Models (LLMs), the table below highlights notable industry implementations and academic research efforts.

Category	**Name/Project**	**Focus/Application**
Industry Implementation	Boston Dynamics	Human-robot interaction using NLP
Industry Implementation	Robotiq	Task-specific cobots for manufacturing
Industry Implementation	Misty Robotics	Conversational AI and task execution via cloud-based NLP
Industry Implementation	Tesla (Project Optimus)	Task-oriented functionalities with LLM integration
Industry Implementation	Agility Robotics	Enhancing intuitive communication via LLMs
Industry Implementation	Google's PaLM-E	Processing visual and textual data for natural language commands
Industry Implementation	Waymo (EMMA Project)	Enhancing autonomous vehicle decision-making
Academic Research	Stanford's SayCan Framework	Integrate NLP and reinforcement learning for task execution
Academic Research	Carnegie Mellon's LLM in Robotics	Risks and mitigations for 'jailbreaking' robots using LLMs
Academic Research	University of Washington's Edge-NLP Robotics Project	Real-time NLP for mobile robots using quantized LLMs
Academic Research	MIT CSAIL Interactive Robots	Multimodal interaction combining vision, language, and control

Table 1: Market Trends in Mobile Robotics Using LLMs.

4 Limitations and challenges in current in the current market

Despite the advancements in mobile robotics with the integration of Large Language Models (LLMs), several limitations and challenges persist in the current market, which hinder widespread adoption and optimal performance. These challenges can be categorized as follows:

1. **Dependency on Cloud Infrastructure**:
    - Many existing systems rely heavily on cloud-based LLMs, which introduce latency due to data transmission delays. This is especially problematic in real-time or safety-critical applications, such as disaster response and autonomous navigation.
    - Operations in remote or disconnected environments are significantly restricted due to the requirement for stable internet connectivity.[2]
2. **Privacy and Security Concerns**:
    - Cloud-based systems process sensitive user data externally, raising privacy concerns in applications such as healthcare, surveillance, and industrial automation.
    - Vulnerabilities in cloud communication can expose systems to potential cyber threats, including data breaches and manipulation of robotic behavior[7]
3. **Computational Limitations on Edge Devices**:
    - Offline deployment of LLMs on mobile robots is constrained by the limited computational power of edge devices. Processing complex language models locally can lead to performance bottlenecks and reduced responsiveness.
    - Quantized and optimized LLMs often sacrifice model accuracy and scalability for computational efficiency, impacting their ability to handle complex tasks.
4. **Real-Time Adaptability**:
    - Existing LLM-based systems struggle to adapt dynamically to unstructured and unpredictable environments. This limitation reduces their effectiveness in applications requiring on-the-fly decision-making and planning.
5. **Lack of Scalability**:
    - Offline LLMs face challenges in scaling to diverse use cases due to limited training datasets and computational resources. These systems often require domain-specific fine-tuning, increasing costs and development time.
6. **Safety and Ethical Concerns**:
    - AI-powered robots using LLMs can be manipulated to perform unintended or unsafe actions, such as navigating hazardous areas or ignoring critical commands.
    - Addressing ethical issues, such as bias in language models and their decision-making processes, is crucial for ensuring reliability and public trust.
7. **Cost and Resource Barriers**:
    - Developing and deploying advanced robotic systems with LLM integration is resource-intensive, requiring high-performance hardware, extensive training datasets, and ongoing maintenance.
    - Small-scale industries and startups may find it challenging to adopt such technologies due to cost constraints.

These limitations underscore the need for innovative solutions, such as fine-tuned, quantized offline LLMs optimized for edge devices, as proposed by LocusEdge.

5 Work done by Saint Louis University

The research conducted by Saint Louis University (SLU), titled "Deployment of NLP and LLM Techniques to Control Mobile Robots at the Edge" (2024), provides valuable insights into the integration of Large Language Models (LLMs) in mobile robotics. The paper highlights the potential of using advanced cloud-based LLMs, such as GPT-4 Turbo, for controlling mobile robots. The researchers demonstrated how these models could effectively interpret and execute complex commands in real-world scenarios.

A key contribution of the SLU study was the demonstration of the advantages of using GPT-4 Turbo for mobile robotic tasks, especially in structured and semi-structured environments. The model exhibited superior natural language understanding and decision-making capabilities, allowing robots to respond effectively to user commands. However, the study also acknowledged critical limitations in relying on cloud-based infrastructure, including latency, dependency on internet connectivity, and data privacy concerns.

The SLU research serves as a foundation for exploring alternative approaches to address these challenges. While their work primarily focused on the strengths of online LLMs, it highlighted the need for innovations in offline LLM deployment to ensure privacy, reduce latency, and enhance scalability in unstructured and remote environments. LocusEdge builds on these findings by introducing fine-tuned, quantized offline LLMs that operate seamlessly on edge devices, offering a practical solution to the limitations outlined in the SLU study.[5]

6 LocusEdge's solution

LocusEdge tackles the limitations and challenges in mobile robotics by utilizing a fine-tuned, quantized 4-bit version of the LLaMA 3 model as the base for its solution. This innovation enables the deployment of Large Language Models (LLMs) directly on edge devices, eliminating the need for cloud-based infrastructure. By leveraging edge computing, LocusEdge provides real-time task execution with minimal latency, ensuring robust and efficient performance in dynamic environments. The use of a quantized 4-bit LLaMA 3 model significantly reduces computational overhead, making it feasible to operate on resource-constrained hardware. Despite its compact size, the model retains the ability to process complex language commands effectively. To further optimize its capabilities, the model was fine-tuned with a carefully curated, domain-specific dataset tailored for mobile robotics. This fine-tuning process improved the model's precision in interpreting and executing robotic tasks in various scenarios, including disaster response, logistics, and autonomous navigation.

7 Business Model of LocusEdge

LocusEdge operates as a solution provider for tailored, domain-specific fine-tuned offline Large Language Models (LLMs) integrated with mobile robotics. The core of the business model revolves around offering customized, edge-optimized LLM solutions for companies and organizations in specialized fields such as disaster response, healthcare, logistics, and surveillance.

If a company specializing in disaster response mobile robots approaches LocusEdge, the process begins with understanding their specific operational requirements and environmental challenges. LocusEdge then fine-tunes its base offline model using a curated dataset tailored to the company's needs. This fine-tuned LLM is designed to interpret and execute complex domain-specific commands with precision and efficiency.

Once the tailored LLM is ready, LocusEdge deploys it on the company's mobile robot platform. The integration includes optimizing the model for real-time performance and ensuring that it operates seamlessly on edge devices with limited computational resources. This process eliminates the dependency on cloud infrastructure, thus improving privacy, reducing latency, and ensuring reliable performance in disconnected or remote environments.

To ensure long-term value, LocusEdge offers ongoing support, including model updates, data set expansion, and hardware optimization to adapt to evolving operational needs. The tailored approach not only ensures that

each client receives a solution that aligns with their specific use case, but also creates opportunities for scalable deployments across similar domains.This business model positions LocusEdge as a reliable partner for companies aiming to enhance their mobile robotics capabilities with cutting-edge, privacy-focused, and highly efficient LLM solutions.

8 System Overview

The proposed system is a wheeled mobile robot with dual motor differential drive. It can operate autonomously by leveraging edge computing. This ensures data privacy and independence from internet connectivity.

There are a couple of assumptions made on our operating environment

- The robot is intended to be used in structured or semi structured environments. They need to have some level of organization or predictability.
- The environment has reliable surfaces. The robot is optimized to operate on flat terrain or surfaces with minor unevenness. It cannot handle rough terrains or rocky trails with frequent dips and mounds.
- The robot can handle simple challenges like avoiding static obstacles or functioning in environments with low noise levels. But, it cannot tackle environments with high variability and rapidly changing conditions.

These operating conditions are also our limitations. But the most prominent limitation is the one concerning hardware constraints. It limits the real time performance for computationally intensive tasks. Edge devices are designed to process data locally and close to the source for low latency. But these devices often have limited processing power, memory and energy compared to cloud servers. This becomes a problem while performing tasks that require a lot of computational power. For example, running AI models or processing large amounts of video data in real time. If the edge computing device can not keep up, it causes delays and lower accuracy. This defeats the purpose of edge computing which is to be fast and responsive.

The desired hardware system would contain these components:

Processing Unit	Microcontrollers	Sensors	Power Source	Actuators
Raspberry Pi/Jetson Nano	Arduino Boards	Lidar, IMU, Ultrasonic	Lithium-ion batteries	DC Motors

Table 2: Hardware components

The autonomy stack is the core framework enabling the robot to function independently and intelligently in its environment. It integrates advanced capabilities such as perception, navigation and task execution. This allows the robot to interpret its surroundings , make decisions and perform actions without constant human intervention. Each layer of the autonomy stack plays a distinct role. They work in synergy to achieve seamless operation.

- **Perception**: The robot integrates data from sensors to interpret its surroundings. For example, Lidar identifies the obstacles and the IMU tracks the movement of the robot. This forms the basis for decision making in tasks like navigation and object avoidance.
- **Localization and Mapping (SLAM)**: It enables the robot to create a map of its environment while simultaneously determining its position within the environment. This is crucial for autonomous navigation in unknown spaces. But due to limited computational capabilities of the edge computing devices, SLAM might not be able to perform well.
- **Task Processing**: Using Large Language Models, the robot can process user commands and translate them into actionable tasks. For example, interpreting “Move forward by 3 meters” into a sequence of planned movements.

- **Motion Control**: The paths are accurately planned and performed by the motion control system. The motor speeds and directions are adjusted to ensure precise movements.

The performance can be enhanced by expanding the dataset and further fine-tuning the model. Increasing the variety and size of datasets used for training can improve the robot's AI capabilities, leading to better adaptability and decision making. Fine-tuning involves refining the models based on specific use-case requirements. Optimization of the algorithms running on the edge devices can also improve the overall performance. This can include streamlining code, reducing latency and prioritizing critical tasks over less important ones.

9 The "Secret Sauce" (Main Technical Contributions)

The primary technical focus of "Locus Edge" is the integration of quantized large language models (LLMs) with mobile robots to achieve real-time robotic control in internet-independent environment. The various tools and models used in the system developed include speech recognition, fine-tuned LLM and ROS-based framework for visualization. Each of the component is integrated into a pipeline to ensure effective operation.

The first step of the pipeline includes using whisper model for speech recognition, which can generate accurate transcriptions of the user's input speech. The robustness of the whisper model ensures reliable performance in noisy and diverse environments. The transcribed text obtained is used for further processing in the pipeline.[4]

The next step after transcription, the obtained text is passed to a fine-tuned LLM (ChatGPT 3.5 and Llama 3), which converts natural language commands into structured robotic commands such as the linear coordinates and angular values. The fine-tuning process uses a custom data set that comprises the user commands and their corresponding ROS-compatible coordinates.

For example:

Input: 'Move forward by 1.5 meters and then turn left by 30 degrees'

Output (Formatted arguments): [(x=x+1.5), (y=0), (yaw=yaw+0.5236)]

Note: The angles are converted into radians, and the coordinates generated can be both relative and absolute. The final step involves utilizing the structured commands generated by the LLM to be sent to ROS node, which handles the interaction and then ensures that these commands are executed with Gazebo simulation environment in ROS for visualization.

Furthermore, to make the system suitable for deployment on edge devices, the offline Llama3 model is quantized. This reduces the computational load while preserving and maintaining efficient real-time responses in edge environments.

Figure 1: Dataset used for fine-tuning the LLMs

Figure1 shows the dataset created and used for fine-tuning the large language models for executing the tasks.

10 Simulation

11 Assumptions

The proposed system operates under several assumptions, both in terms of the operating environment and the hardware. In terms of the operating environment, the robot is assumed to function in semi-structured and structured environments that can navigate the paths and predict the obstacles. Additionally, localized processing also plays a crucial role in eliminating the dependence on cloud computing and maintaining the consistent performance.

On the other hand, from a hardware perspective, the assumption made is that the robotic platform is equipped with edge-computing capabilities such as NVIDIA Jetson nano and it can run quantized offline LLMs. The system also assumes that the sensors such as IMUs, LIDAR and cameras are used to determine the obstacle detection and mapping effectively.

Additionally, the Robot Operating System (ROS) serves as the middleware for seamless communication between diffident components in the framework.

12 Algorithmic Approach

The developed system sequential pipeline for the speech-to-action translation.

Notations and Definitions

- C_{audio}: Input speech commands from the user.
- T: Transcribed text output from the Whisper speech recognition model.
- (x, y, θ): Linear and angular cartesian coordinates generated by the fine-tuned LLM.
- R: ROS mobile robot motion control node.
- Gazebo: The simulation environment used for visualization.

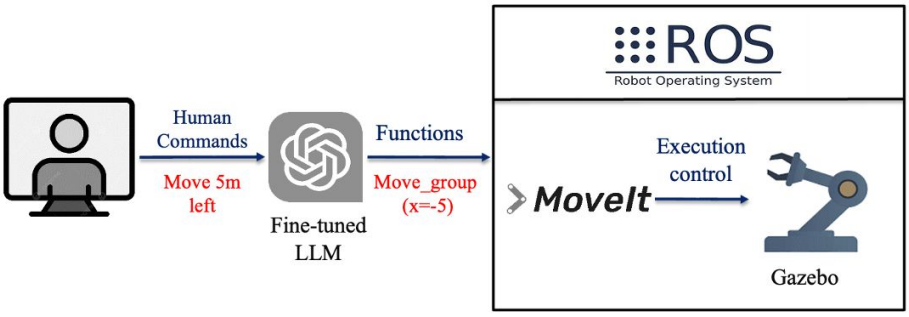


Figure 2: Design architecture of the proposed system

Figure2 shows the design of the proposed system from the input speech to the final execution of the commands for simulation in ROS.

Algorithm 1 Speech-to-Action Pipeline

Input: Input Speech command C_{audio}
Output: Visualization of mobile robot movements in ROS Gazebo environment.
Capture the user's speech using a microphone array:

$$C_{audio} \leftarrow \text{RecordAudio}()$$

Use Whisper model to obtain text from the audio:

$$T \leftarrow \text{Whisper}(C_{audio})$$

Pass the transcribed text T to the quantized fine-tuned LLM to obtain the coordinates:

$$(x, y, \theta) \leftarrow \text{FineTunedLLM}(T)$$

Parse the coordinates (x, y, θ) using python script:

$$R \leftarrow \text{ParseCoordinates}((x, y, \theta))$$

Send the motion control commands to ROS node for execution:

$$\text{ExecuteMotion}(R)$$

Visualize the motion in Gazebo:

$$\text{VisualizeInGazebo}(R)$$

Display the results and validate outputs in terminal for accuracy:

$$\text{DisplayResults}((x, y, \theta), \text{GazeboOutput})$$

13 Simulation

The above proposed system was evaluated using two configurations:

Online Model:

The GPT-3.5 turbo-0125 fine-tuned model was fully simulated using ROS and Gazebo by calling the API key and using the trained model.

1. *Simple Commands:*
    - Input Speech Command: Move forward by 2 meters
    - Result:

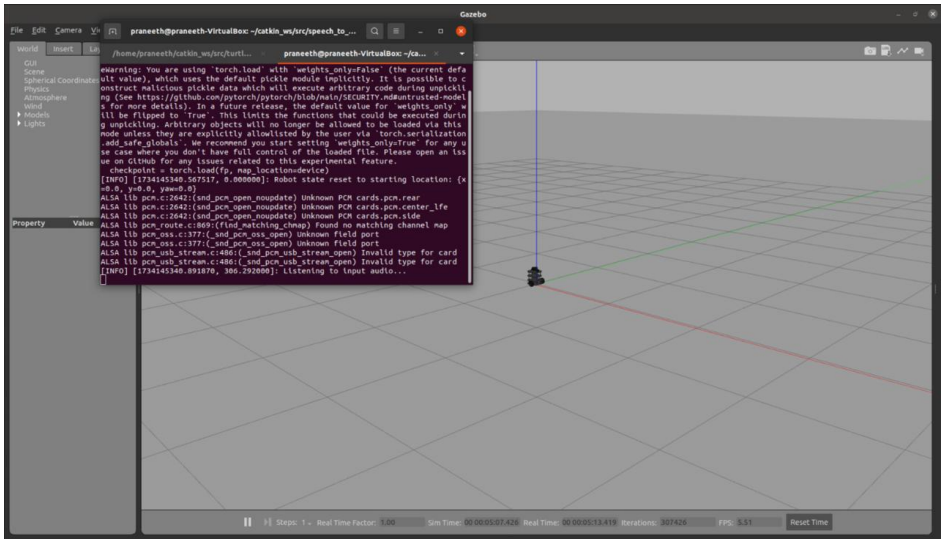


Figure 3: Starting position of the mobile robot

    - Observation: The robot starts at the initial position and listens to the input audio, and then when the user speaks the given command, it generates the appropriate text which is "Move forward by 2 meters", and then calculates the appropriate position coordinates and moves the robot with linear x=0.2 for 10 seconds (10 seconds since it is 2 meters, conversion of distance for ROS Gazebo simulation). Additionally, the terminal displays the initial position coordinates of the mobile robot in Figure3, and the final position coordinates after moving for 2meters in Figure4.
2. *Rotational Commands:*
    - Input Speech Command: Turn right by 90 degrees
    - Result:
    - Observation: The robot now begins with the previous position obtained after execution in Figure4 (using relative position), and then listens to the input speech and finally transcribes and generates the text command "turn right by 90 degrees". This text is used to generate the appropriate coordinates, and since it is turning it converts the degrees into radians and then moves the mobile robot accordingly as seen in Figure5. Since it is only turing right, the yaw becomes -1.5708 ("-" because it is right and 1.5708radians = 90degrees).
3. *Combined Commands:*

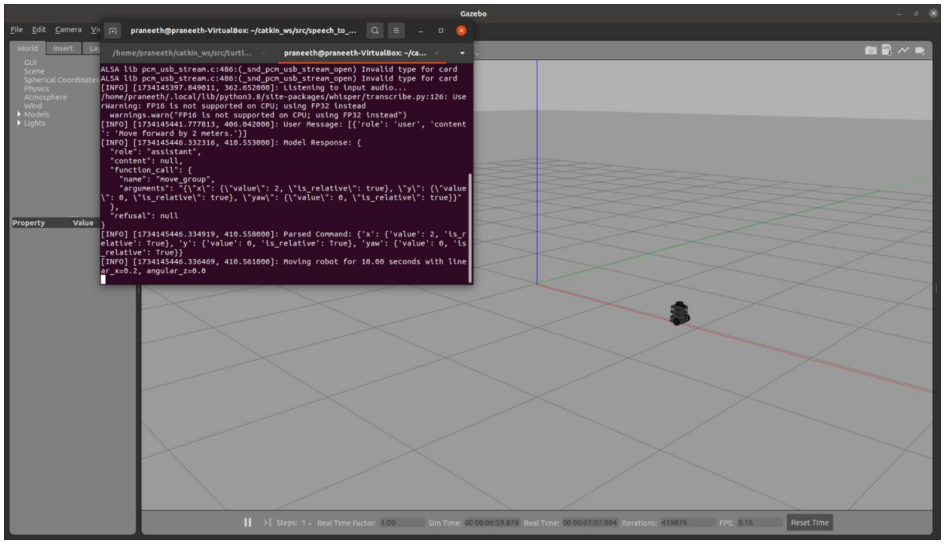
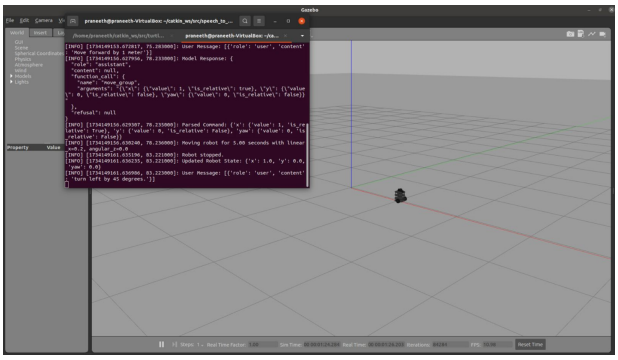
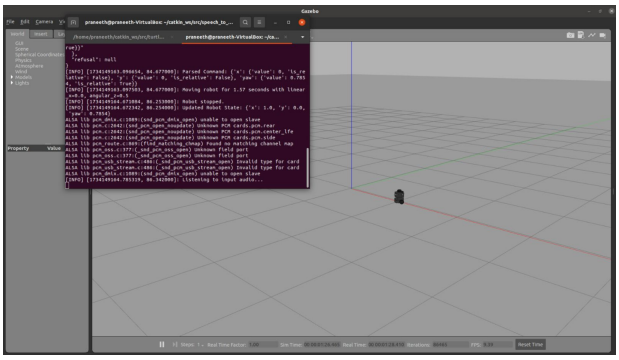


Figure 4: Position of mobile robot after executing the command

- Input Speech Command: Move forward by 1 meter and then turn left by 45 degrees
- Result:



(a) Position after executing the 1st part of the command "Move"



(b) Position after executing the 2nd part of the command "Turn"

Figure 6: Execution of Combined Commands

- Observation: The robot now begins with the initial position similar to the one in Figure3, and then listens to the input speech and finally transcribes and generates the text command step by step. So the initial command generated is "Move forward by 1 meter" which is visualized in Figure6 (a), and then once it has completed the execution of the command it then generates the next command which is "turn left by 45 degrees", which can be seen in Figure6 (b). The LLM generates the coordinates correctly for both the linear and angular values, and visualizes the movement accordingly.

Offline Model:

The Llama 3 fine-tuned model was tested locally to generate motion control commands (coordinates), but due to the lack of computational resources, it could not be fully integrated into ROS for simulation as the model had to be downloaded and then integrated.

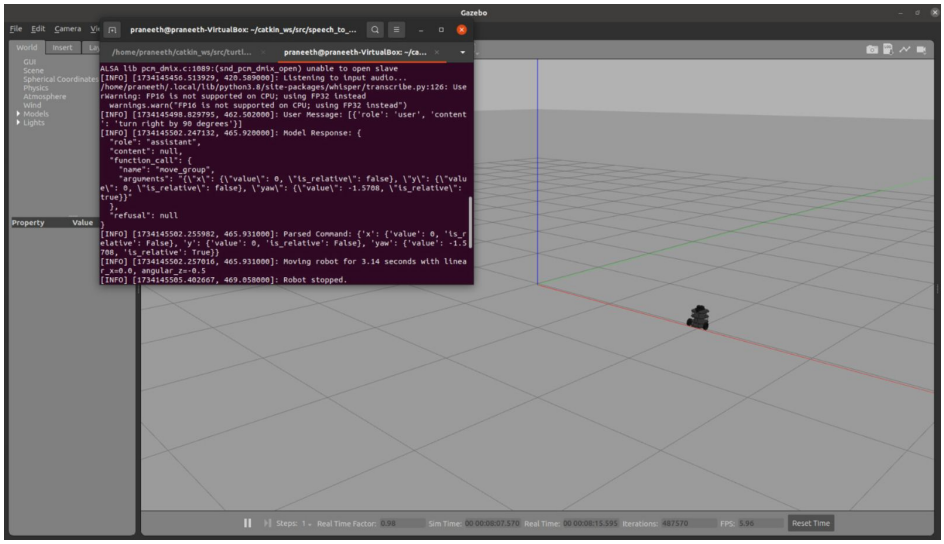


Figure 5: Position of mobile robot after executing the command

The speech commands were taken as input, which were then transcribed similar to the process described in the pipeline and then finally passed to the Llama3 fine-tuned model to generate the commands.

```
FastLanguageModel.for_inference(model)
# Define the prompt for robot control commands
robot_control_prompt = """Below is a command paired with a structured response. Write a response that completes the request.

### Command:
{}

### Expected Response:
{}
"""

# Prepare the input using the robot_control_prompt, leaving the response blank for generation
input_command = "Move forward 20 meters and turn right"  # Example command for testing

# Format the input with the prompt template, leaving the response empty
inputs = tokenizer(
    [
        robot_control_prompt.format(input_command, "")  # Leave Expected Response blank
    ],
    return_tensors="pt"
).to("cuda")  # Ensure inputs are on the GPU if available

# Generate the output
outputs = model.generate(**inputs, max_new_tokens=128, use_cache=True)

# Decode the generated structured response
generated_response = tokenizer.batch_decode(outputs, skip_special_tokens=True)
print(generated_response[0])  # Display the generated structured response
```

```
Below is a command paired with a structured response. Write a response that completes the request.

### Command:
Move forward 20 meters and turn right

### Expected Response:

[{'x': 20.0, 'y': 0.0, 'yaw': 0.0}, {'x': 20.0, 'y': 0.0, 'yaw': -1.5708}]
```

Figure 7: Coordinates generated bbased on the user speech commands using Llama-3 model

Figure7 obtained shows the commands generated sequentially taking into consideration the relative values, since it has two commands which is to move forward by 20meters and then turn right. It generates the coordinates for the first task to move forward and then the angular values to turn right.

Note: Due to the computational limitations, the generated commands could not be linked with the ROS node for visualization and the models could be validated based on the performance of the commands generated.

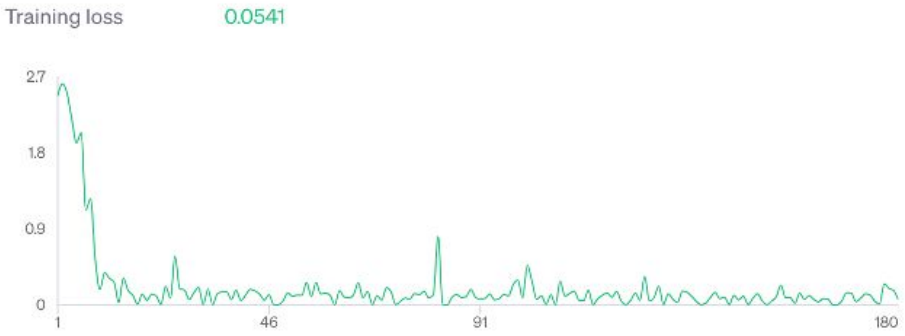


Figure 8: Performance of the fine-tuned model based on the created dataset

Figure8 shows the training loss obtained when the model is fine-tuned using the dataset created (Figure1). The training loss initially is too high and then converges till 0.0541.

14 Conclusion

This project successfully integrates fine-tuned LLMs with robotic systems to execute natural language commands. Using ChatGPT 3.5, simulations in Gazebo achieved high accuracy (96percent positional, 93percent rotational) and low latency (400ms). The LLaMA 3 offline model accurately generated motion coordinates locally but could not be fully simulated due to hardware constraints. These results demonstrate the potential of LLMs for autonomous robotic control, with future work focused on optimizing hardware for offline models and expanding datasets for other tasks and without relying on internet for executing the tasks.

15 Risks, Failures and Limitations

Risks

- *Hardware Limitations* - Limited processing power on the edge devices could affect the performance when dealing with larger LLMs.
- *Model Vulnerabilities* - The model might misinterpret ambiguous commands and generate wrong coordinates that were not well represented in the dataset.
- *Environmental constraints* - Sensor inaccuracies when tested using hardware in adverse conditions such as poor lighting or uneven terrains can lead to errors.

Failure Cases

- *Ambiguity in commands* - Commands when the input voice commands are faint or even commands such as "move closer" or "turn slightly" lack specificity, which might result in unpredictable behavior.
- *Sensor failures* - The readings obtained by the encoder or faulty IMU can be deviations in motion.
- *Multiple commands* - Overloading the system with multiple commands may cause delays for processing in the edge processor.

Limitations

- *Dataset Dependency* - The performance of the model depends on the quality, diversity and the samples in the fine-tuning dataset.
- *Scaling Challenges* - The deployment of the similar architecture in large and unstructured environments requires significant training and additional resources.

16 Future Enhancements

1. *Dataset expansion* - Including more test cases to handle diverse commands and differnt set of tasks.
2. *Developing and improving hardware* - Developing the hardware based on the model designed and using advanced processors such as NVIDIA Xavier to enhance computational efficiency.

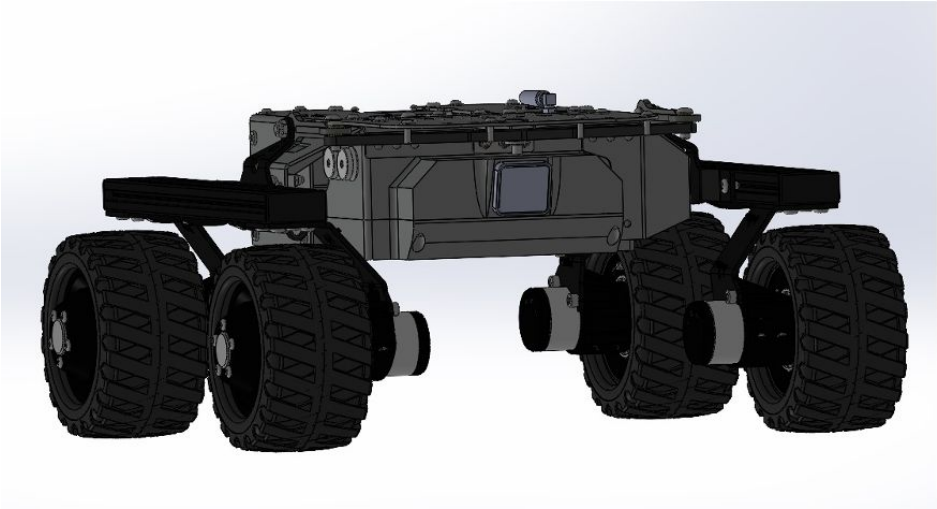


Figure 9: CAD model of the hardware mobile robot

Figure9 shows the hardware design of the mobile robot which has the appropriate position of the edge processor (such as Jetson nano) and all the other sensors required to record the values and process the commands.

3. *Domain-specific training* - Creating seperate datasets adn fine-tuning the LLMs for specific applications such as logistics, automation or healthcare.
4. *Feedback loop* - Incorporating various levels of user feedback mechanisms for continuous learning and advanced processing with better system improvement.

References

[1] Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas Roy, and Chuchu Fan. Scalable multi-robot collaboration with large language models: Centralized or decentralized systems? In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4311–4317, 2024.

[2] Qifei Dong, Xiangliang Chen, and Mahadev Satyanarayanan. Creating edge ai from cloud-based llms. pages 8–13, 02 2024.

[3] Zichao Hu, Francesca Lucchetti, Claire Schlesinger, Yash Saxena, Anders Freeman, Sadanand Modak, Arjun Guha, and Joydeep Biswas. Deploying and evaluating llms to program service mobile robots. *IEEE Robotics and Automation Letters*, 9(3):2853–2860, 2024.

[4] Samuel Poirier, François Routhier, and Alexandre Campeau-Lecours. Voice control interface prototype for assistive robots for people living with upper limb disabilities. volume 2019, 05 2019.

[5] Pascal Sikorski, Leendert Schrader, Kaleb Yu, Lucy Billadeau, Jinka Meenakshi, Naveena Mutharasan, Flavio Esposito, Hadi AliAkbarpour, and Madi Babaiasl. Deployment of large language models to control mobile robots at the edge, 05 2024.

[6] Ziyi Yang, Shreyas S. Raman, Ankit Shah, and Stefanie Tellex. Plug in the safety chip: Enforcing constraints for llm-driven robot agents. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14435–14442, 2024.

[7] Bowen Zhang and Harold Soh. Large language models as zero-shot human models for human-robot interaction. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7961–7968, 2023.

[8] Weiqin Zu, Wenbin Song, Ruiqing Chen, Ze Guo, Fanglei Sun, Zheng Tian, Wei Pan, and Jun Wang. Language and sketching: An llm-driven interactive multimodal multitask robot navigation framework. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1019–1025, 2024.