# RDMM: Fine-Tuned LLM Models for On-Device Robotic Decision Making with Enhanced Contextual Awareness in Specific Domains

Shady Nasrat* Minseong Jo, Myungsu Kim, Seonil Lee, Jiho Lee, Yeoncheol Jang, and Seung-joon Yi

***Abstract*— Large language models (LLMs) represent a significant advancement in integrating physical robots with AI-driven systems. We showcase the capabilities of our framework within the context of the real-world household competition. This research introduces a framework that utilizes RDMM (Robotics Decision-Making Models), which possess the capacity for decision-making within domain-specific contexts, as well as an awareness of their personal knowledge and capabilities. The framework leverages information to enhance the autonomous decision-making of the system. In contrast to other approaches, our focus is on real-time, on-device solutions, successfully operating on hardware with as little as 8GB of memory. Our framework incorporates visual perception models equipping robots with understanding of their environment. Additionally, the framework has integrated real-time speech recognition capabilities, thus enhancing the human-robot interaction experience. Experimental results demonstrate that the RDMM framework can plan with an 93% accuracy. Furthermore, we introduce a new dataset consisting of 27k planning instances, as well as 1.3k text-image annotated samples derived from the competition. The framework, benchmarks, datasets, and models developed in this work are publicly available on our GitHub repository at https://github.com/shadynasrat/RDMM.**

## I. Introduction

In the rapidly advancing field of robotics and artificial intelligence, the imperative to augment the decision-making capabilities of autonomous systems has been a paramount concern. These models can enhance decision-making, interaction, and planning through their linguistic and contextual understanding abilities. Nevertheless, the direct deployment of large language models in domain-specific robotic tasks faces significant challenges. These key challenges include first, insufficient ability to integrate and leverage personal contextual knowledge about the agent itself, such as its background, capabilities, and specific skills. Second, deployment in real-time on-device settings necessitates efficient inference mechanisms, which can be limited by the computational complexity of large language models.

Recently, there are many methods for solving the grounding problems of LLMs in robotics. PaLM-E [1] generates control sentences according to multi-modal data. RT-X [2] directly infer instructions based on languages and images. ChatGPT for Robotics [3] needs the declaration of APIs for reasoning the actions of tasks. SayCan [4] selects most suitable actions according to environmental information. VoxPoser [5] converts the observation space into a 3D value maps for generating trajectories. While existing methods can achieve domain-specific planning and handle some partial disturbances, a key limitation is their inability to incorporate the agent's own knowledge, such as personal background information, capabilities, and skills. This personal contextual knowledge is crucial for well-reasoned question-answering to support effective planning processes.

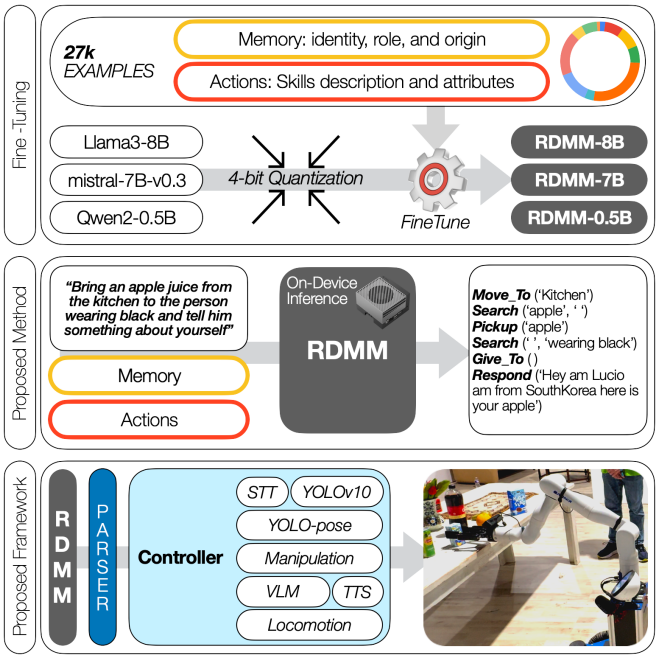


Fig. 1. **RDMM Overview**: The process begins by fine-tuning quantized LLM models on our specialized dataset to create RDMM models. The illustration showcases an example of RDMM's On-Device inference, followed by the proposed framework parsing the RDMM-generated plans for execution. These plans are carried out using a controller that interacts with various models and enabling both robotic manipulation and locomotion.

For instance, a domestic robot assistant could be given a simple task such as delivering an apple to the individual wearing a black t-shirt, and then engaging in a conversation about its recent achievements or favorite color. Existing methods would face difficulties in executing this request, as the employed large language models lack access to the robot's personal knowledge. In contrast, our RDMM framework enables the agent to retrieve and utilize its own information, including its identity, role, and origin, to formulate an appropriate and informative response. This could involve statements like *'I am Lucio, a household robot assistant. How may I assist you?'* or *'Hello, I am Lucio, and I*

*This project was funded by Police-Lab 2.0 Program(www.kipot.or.kr) funded by the Ministry of Science and ICT(MSIT, Korea) & Korean National Police Agency(KNPA, Korea) (No. 082021D48000000) and Korea Institute for Advancement of Technology(KIAT) grant funded by the Korea Government(MOTIE)(P0008473, HRD Program for Industrial Innovation)

Authors are with Faculty of Electrical Engineering, Pusan National University, Busan, South Korea seungjoon.yi@pusan.ac.kr

*originate from South Korea'*. Furthermore, a straightforward task that would challenge other methods is *'What can you do?'*, which necessitates the robot's understanding of its own capabilities. Our RDMM framework would provide an informative response highlighting its abilities, such as: *'I can help you with tasks such as moving to a location, searching for objects or people, picking up objects, placing them on a surface, and answering questions.'*.

This paper focuses on developing RDMM models by fine-tuning large language models to acquire self-aware domain-specific planning capabilities. First, the study constructed a comprehensive dataset centered on the tasks and rules of the RoboCup@Home competition. Building upon this foundation, the dataset was further expanded to incorporate the agents' personal knowledge and information regarding their own capabilities and skills. This approach empowers the large language models to not only plan effectively for the given tasks, but also engage in meaningful interactions by providing insightful responses to inquiries about their personal details and abilities, such as their identity, role and background.

This paper makes the following key contributions:

- A local framework that leverages RDMM models to enhance the autonomous decision-making capabilities of robots, integrating knowledge of their skills and personal information.
- Comparative analysis of our method against base language models, GPT-4o-mini and GPT-4o and other LLM-based approaches, showcasing the advantages of RDMM models in terms of planning accuracy, On-Device compatibility and inference speed.
- Real-world evaluation of our system at the RoboCup@Home competition, demonstrating its ability to handle complex robotic tasks within a household environment.
- Open-source framework, benchmarks, RDMM models, a specific-domain planning dataset of 27k text pairs, and a dataset of annotated 1.3k images to facilitate further research and development in this area.

## II. Related Work

Large language models represent a significant advancement in integrating physical robots with AI systems. This approach aims to address the limitations of large language models, which often lack the necessary contextual grounding for effective decision-making in real-world environments. By conditioning language models with pre-trained behaviors, LLM-based systems enable robots to engage in more natural interactions, understand task-specific constraints, and generate executable plans tailored to their capabilities.

The field of LLM-based robotics has witnessed the development of several notable approaches that demonstrate the potential of integrating large language models with robotic systems [1]–[15]. For instance, LM-Nav [11] proposes a goal-conditioned policy that utilizes large, unannotated datasets, combining pre-trained models for navigation, image-language association, and language modeling.

TABLE I
LLM-Based Methods for complex robotics tasks comparison

| **Methods** | **Inputs** Text + | **Output** | **Model Info.** (on-device) |
|---|---|---|---|
| LLM-BT [6] | Images | Variable BTs | (x) ChatGPT |
| SayCan [4] | Images | Actions | (x) PaLM |
| VoxPoser [5] | Images | Trajectories | (x) GPT-4 |
| PaLM-E [1] | Multi-modal | Description | (x) PaLM(540B) |
| Huang et al. [7] | – | Actions | (x) GPT-3(175B) |
| Raman et al. [8] | – | Actions | (x) GPT-3 family |
| Text2Motion [9] | Scene desc. | Actions | (x) GPT-3.5 family |
| ProgPrompt [10] | – | Code | (x) GPT-3 |
| LM-Nav [11] | Image | | (x) GPT3 |
| TidyBot [12] | | | (x) GPT3 |
| RT-X2 [2] | | | (x) RT2X-55B |
| LLM+P [13] | Scene desc. | Description | (x) GPT-4 |
| ViLaIn [14] | Image | Description | (x) ChatGPT4 |
| Code as Policies [15] | Images | Code | (x) GPT-3 |
| ChatGPT for Robotics [3] | APIs | Actions | (x) ChatGPT4 |
| RDMM(Ours) | Actions Memory | Actions | (✓) RDMM-8B<br>(✓) RDMM-7B<br>(✓) RDMM-0.5B |

This enables robots to navigate complex environments based on natural language instructions without the need for expensive supervision or fine-tuning, showcasing the practical applications of pre-trained models. Similarly, TidyBot [12] focuses on personalizing robotic assistance for household tasks by learning user preferences through language-based planning and perception, leveraging the few-shot summarization capabilities of LLMs to quickly adapt to new scenarios. Furthermore, LLaRP [16] adapts large language models for reinforcement learning in robotics tasks, utilizing a frozen LLM to take text instructions and visual observations, and outputting actions directly in the environment. This system demonstrates robustness in diverse rearrangement tasks, highlighting the potential of LLMs in reinforcement learning for robotics. Additionally, the Code as Policies [15] approach leverages LLMs trained on code-completion to generate robot policy code from natural language commands, enabling the synthesis of policy code that processes perception outputs and parameterized control primitives, showcasing the expressive power of LLMs in translating high-level instructions into executable robot behaviors. Despite advancements, robots still need to improve natural interactions by better leveraging their knowledge and capabilities. Efficient inference requires local operation for speed and affordability. As shown in Table I, most previous methods depend on large models with server-based inference, increasing costs. Our approach eliminates the need for cloud services by running smaller models to run directly on the robot, resulting in reduced latency, improved autonomy, improved privacy and security, and greater reliability for practical applications.

## III. Method

### A. Dataset Creation

To create a comprehensive dataset for household robots, we drew inspiration from the RoboCup@Home competition tasks, ensuring it covers a wide range of essential skills needed for domestic activities. The dataset was designed into three categories: action-oriented tasks and self-awareness-oriented tasks, each essential for enhancing the robot's operational efficiency and decision-making capabilities in real-world environments. The action-oriented section trains the robot to handle tasks like manipulation, navigation, searching, describing, and counting objects, ensuring it can generate effective strategies for these specific robotic tasks. In contrast, the self-awareness-oriented section equips the robot with a deeper understanding of its identity, capabilities, and purpose, enabling it to engage in more human-like interactions, such as guiding, following and meeting individuals. The final category involves tasks that require a combination of action and memory, where the robot must integrate both types of knowledge to execute complex plans, such as delivering an item and engage in a conversation where it require recalling a relevant detail from its memory.

The dataset comprises 27,514 manually annotated examples, each consisting of textual input-output pairs specifically focused on household tasks. Dataset are structured into 42 scenario-based segments, with each scenario categorized under distinct task types, shown in Fig.2. The dataset encompasses 21 distinct skills, each outlined with detailed attributes in Table II. To enhance the robot's decision-making and operational efficiency, system messages provide action descriptions, usage information, and access to the robot's personal memory, allowing it to recall its knowledge in efficiently. This dataset not only serves as a benchmark for evaluating our models but also plays a crucial role in training the robot for real-world applications. By incorporating both action-based and memory-based tasks, the dataset helps the robot develop a deeper understanding of its role, fostering more rational, context-aware decision-making.

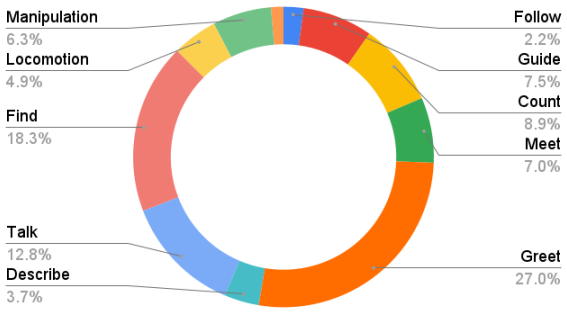


Fig. 2. **Dataset Distribution by Task:** An overview of the dataset allocation, illustrating the ratio of data dedicated to each specific task. Ensuring balanced and comprehensive training for task-specific model performance.

TABLE II
Summary of Dataset Actions

| Actions | Description |
|---|---|
| Respond(request) | Respond to user |
| Move_To(location) | Move to a location |
| Pour_In(object) | Pour object into a container |
| Search_Object(name$^o$, desc.$^*$) | Search for an object |
| Search_Person(name$^o$, desc.$^*$) | Search for a person |
| Pickup() | Pickup an object |
| Place_On(placement) | Place picked up object on placement |
| Place_Next(object) | Place picked up object next to object |
| Give_To() | Give an object to user |
| Open(object) | Open a door |
| Close(object) | Close a door |
| Vision_Ask(Question$^*$) | Ask VLM and return in Answer() |
| Answer() | Retrieve answer |
| Follow() | Follow a person |
| New_Request() | Take a new request |
| Count_Person(desc.$^*$) | Count people and return in Answer() |
| Count_Object(name$^o$, desc.$^*$) | Count object and return in Answer() |
| Ask_Name() | Ask name and return in Answer() |
| What_Time() | Retrieve time |
| What_Day() | Retrieve date |
| What_Tomorrow() | Retrieve tomorrow date |

$^*$: Arguments is processed using VLM, $^o$: Arguments is processed using YOLO

### B. Quantization and Fine-Tuning Details

Llama3-8B [17], Mistral-7B-v0.3 [18], and Qwen2-0.5B [19] was selected as base models for fine-tuning due to their optimal balance of size and performance for Jetson Edge devices. To enhance inference efficiency, GPTQ [20] method is applied for quantization, which compresses the model to 4-bit precision while preserving performance. We also utilize QLoRA [21], freezing the pre-quantized model and train only a new subset of parameters act as an adapter. Training conducted with a learning rate of 2.5e-5 and capped at 1000 steps, while targeting specific layers such as *q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj* and *down_proj*. QLoRA combines the 4-bit NormalFloat quantization, Double Quantization, and Low-Rank Adapters (LoRA) [22] to achieve efficient 4-bit quantization. For a single linear layer in the quantized base model with a single LoRA adapter, QLoRA is defined as:

$$\begin{aligned}\mathbf{Y}^{BF16} = \mathbf{X}^{BF16} * doubleDeq(c_1^{FP32}, c_2^{k-bit}, \mathbf{W}^{NF4}) \\ +\mathbf{X}^{BF16}\mathbf{L}_1^{BF16}\mathbf{L}_2^{BF16}\end{aligned} \quad (1)$$

where $doubleDeq$ is the double de-quantization process:

$$\begin{aligned}&doubleDeq(c_1^{FP32}, c_2^{k-bit}, \mathbf{W}^{k-bit}) \\ &= dequant(dequant(c_1^{FP32}, c_2^{k-bit}), \mathbf{W}^{4-bit}) \\ &= \mathbf{W}^{FB16}\end{aligned} \quad (2)$$

QLoRA uses NF4 for the weights ($\mathbf{W}$) and FP8 for the quantization constants ($c_2$). The block-size is set to 64 for $\mathbf{W}$ for higher precision and 256 for $c_2$ to conserve memory. During the backward pass, only the gradients with respect to the LoRA adapter weights ($\frac{\delta E}{\delta \mathbf{L}_i}$) are computed, not for the 4-bit weights ($\frac{\delta E}{\delta \mathbf{W}}$). However, computing ($\frac{\delta E}{\delta \mathbf{L}_i}$) involves calculating $\frac{\delta \mathbf{X}}{\delta \mathbf{W}}$, which requires dequantizing the storage

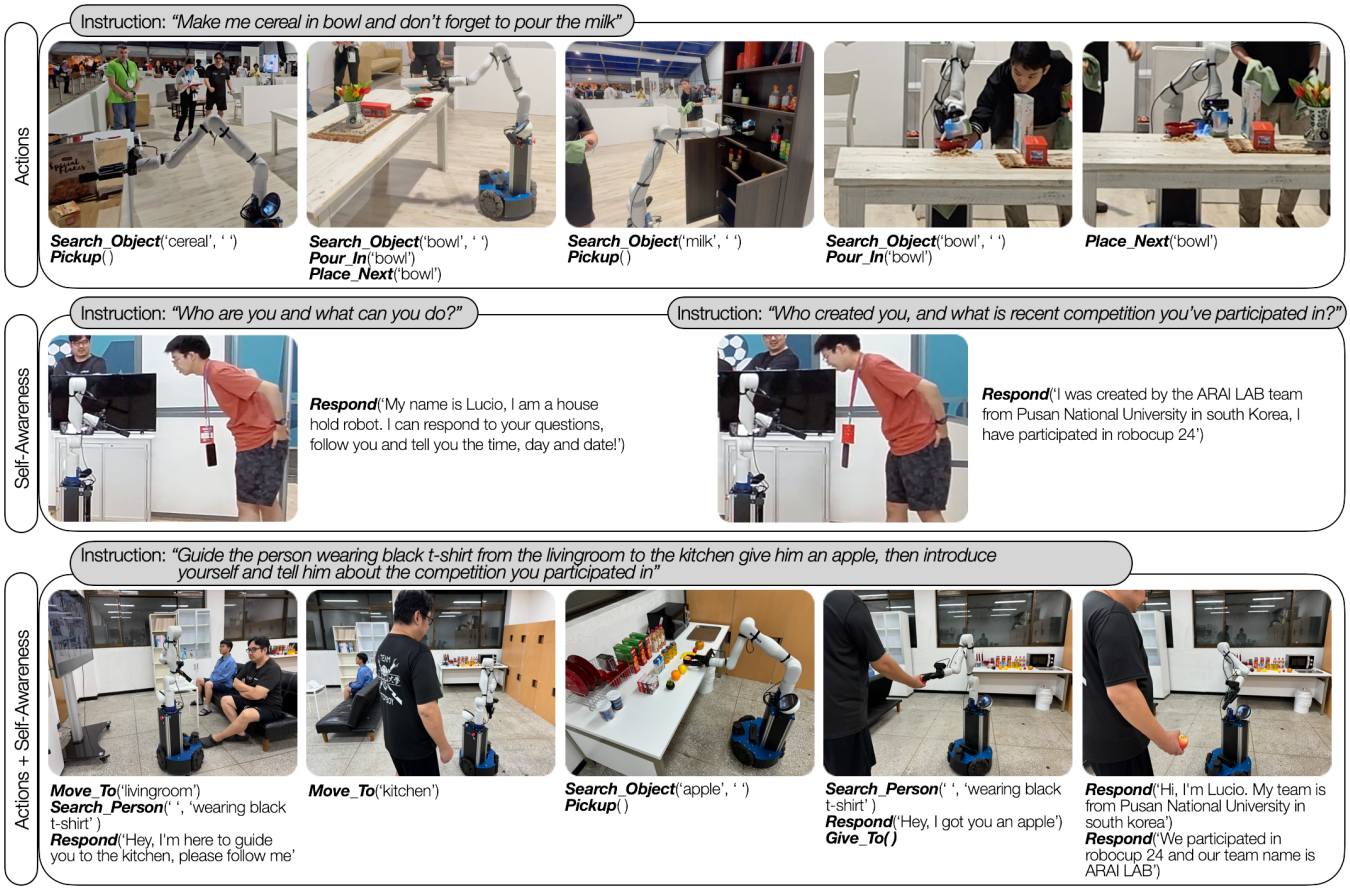


Fig. 3. **Household Robot Planning with RDMM:** This illustration shows three examples of Lucio, a home service robot, using local RDMM model inference to plan and execute tasks. These include planning actions to make cereal, answering self-awareness questions about Lucio's personal memory, and combining actions with self-awareness by retrieving an apple for a person and engaging in conversation about itself.

$\mathbf{W}^{NF_4}$ to the computation data type $\mathbf{W}^{BF16}$. In summary, QLoRA uses 4-bit NormalFloat as the storage data type and 16-bit BrainFloat as the computation data type. The storage data type is dequantized to the computation data type for the forward and backward passes, but gradients are only computed for the LoRA parameters in 16-bit precision. Training time were 24 minutes for RDMM-8B, 11 minutes for RDMM-7B, and 5 minutes for RDMM-0.5B on a single NVIDIA RTX 4090 GPU.

### C. Framework Overview

*1) Parser & Controller:* The parser component of our framework is responsible for translating the RDMM-generated plans into actionable commands that the robot can execute. The controller then interprets these commands and interacts with various models, such as VLMs, YOLO, STT and TTS models, to perform specific tasks.

*2) Vision Language Model:* Visual perception models are crucial for enabling robots to navigate and interact with their surroundings effectively. We employ a 4-bit quantized internlm-xcomposer2-vl-7b [23] Vision-Language Model (VLM) to interpret contextual cues and extract detailed visual information. This model provides accurate descriptions of people, objects, and scenes, making it a reliable source of visual intelligence. For example, the VLM can accurately identify if a person is wearing shoes or holding a cup. In Fig. 3, within the actions + self-awareness example, the generated plan includes the action *Search_Person(' ', 'wearing black t-shirt')*, where the second argument is processed by the VLM to interpret the person's description.

*3) YOLO Model:* For our real-time object detection algorithms supporting robotic manipulation tasks, the first priority is accurately identifying objects in the environment. To achieve this, we trained a YOLOv10L model on an annotated dataset containing 1.3k images sourced from the RoboCup@Home competition. In Fig. 3, within the actions example, the generated plan includes the action *Search_Object('cereal', ' ')*, where the first argument is processed by YOLO to detect object location. Additionally, for human detection and pose estimation, we utilize the YOLOv8-pose model.

*4) Automatic Speech Recognition:* We use Whisper for speech recognition, transcribing audio into text and providing feedback to indicate the robot is listening. For natural responses, we use Seliro-TTS for human-like text-to-speech.

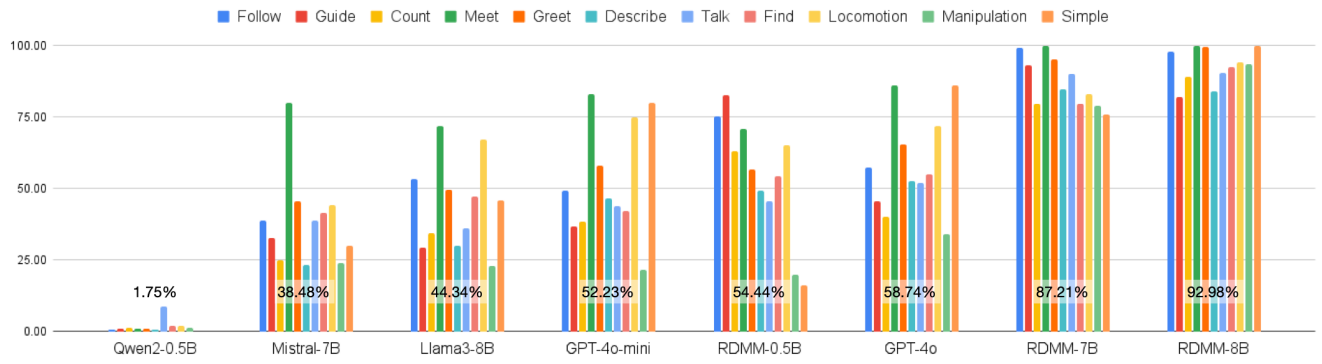


Fig. 4. **Benchmark Accuracy Across Tasks:** This graph presents the evaluation results for RDMM-8B, RDMM-7B, and RDMM-0.5B models, compared with 20-shot conditioned baseline models Llama3-8B, Mistral-7B, and Qwen2-0.5B, alongside GPT-4o and GPT-4o-mini. It highlights their accuracy across various tasks, offering insights into each model's performance in different task scenarios.

## IV. Experiments

We evaluated the accuracy, on-device compatibility and inference speed of our RDMM models, comparing them to baseline models, GPT-4o-mini and GPT-4o. Additionally, we tested our model's real-world performance during the RoboCup@Home competition.

### A. Models Planning Accuracy

The accuracy comparison graph in Fig.4 compares the accuracy of several models across various tasks. It highlights the strong performance of the RDMM models (RDMM-8B, RDMM-7B, and RDMM-0.5B), with a particular focus on their improvements over base models and GPT-4o-mini and GPT-4o. both baseline and GPT models were conditioned with 20-shots examples from the dataset to ensure a fair evaluation across each task. The RDMM-8B model achieves the highest accuracy, with an average of 92.98%, showcasing a significant improvement from its base model's 44.34%. This indicates a substantial leap in capabilities, particularly in tasks like "Follow," "Meet," and "Simple." Similarly, the RDMM-7B model reaches an impressive 87.21% accuracy, surpassing both its base model's performance (38.48%) and other comparative models, such as GPT-4o. The RDMM-0.5B model, while smaller in scale, still demonstrates a marked improvement over its base model, increasing accuracy from 1.75% to 54.44%. Although it slightly trails behind GPT-4o, which achieved 58.74%, it still outperforms GPT-4o-mini at 52.23%, indicating the model's competitive edge despite its smaller size.

### B. On-Device Inference Compatibility

The compatibility of RDMM models for on-device inference was evaluated across various Jetson hardware platforms, including the Orin AGX 64GB, Xavier AGX 32GB, Xavier AGX 16GB, Orin NX 16GB, and Xavier NX 8GB, all of which employ ARM architecture with integrated RAM and VRAM.

*1) RDMM On-Device Compatibility:* The RDMM models—RDMM-8B, RDMM-7B, and RDMM-0.5B—were tested to ensure local inference on these devices. RDMM-8B, requiring 1.1GB RAM and 8.5GB VRAM, and RDMM-7B, requiring 1GB RAM and 6.8GB VRAM, successfully operated on most platforms. However, the Xavier NX 8GB, with limited memory, could only support the RDMM-0.5B model, which demands 0.34GB RAM and 1.9GB VRAM. The larger RDMM models exceeded the available memory on the Xavier NX 8GB, highlighting the importance of aligning model size with hardware constraints for effective on-device inference.

*2) Framework On-Device Compatibility:* We also evaluated the full system framework, including VLM, Whisper, Serlio-TTS, YOLOv8-pose, and YOLOv10, alongside the RDMM model. The results, illustrated in Fig.5, shows the memory usage ratios of each model on a local device. The entire system required 30GB of memory, making the 32GB Xavier AGX the smallest device capable of running it.

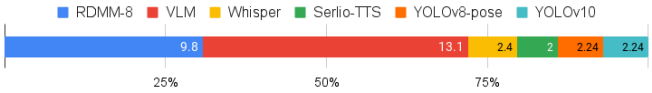


Fig. 5. **Framework VRAM consumption:** A graphical representation depicting the VRAM usage of each model within the framework.

### C. Models Inference Speed Comparison

The performance evaluation graph presented in Fig.6 demonstrates the inference speed comparison of RDMM models against other models on various Jetson devices, highlights a slight trade-off between speed and enhanced capabilities. While RDMM models are marginally slower than their base models—such as Llama3-8B, Mistral-7B, and Qwen2-0.5B this slowdown is primarily due to the Progressive Fine-Tuning with Layer-wise Re-calibration approach, which integrates a QLoRA compact neural network adapter. For instance, on the ORIN AGX 64GB, the RDMM-8B model achieved 6.12 tokens per second (T/s), compared to Llama3-8B's 10.86 T/s and Mistral-7B's 11.87 T/s. Similarly,

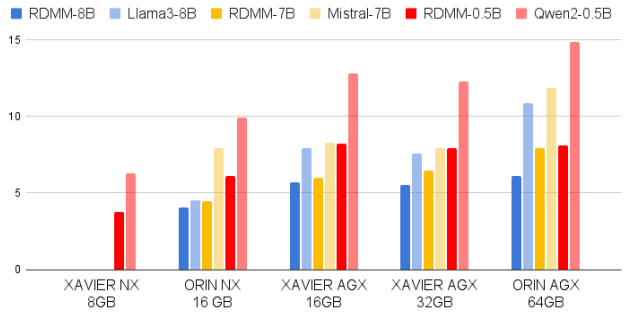


Fig. 6. **On-Device inference speed comparison:** A detailed analysis comparing the inference speeds of RDMM and baseline models across various Jetson devices. This comparison highlights the efficiency and performance of each model when deployed directly on hardware.

on the XAVIER AGX 32GB, the RDMM-8B model achieved 5.54 T/s, compared to Llama3-8B's 7.56 T/s and Mistral-7B's 7.95 T/s. On smaller on-device platforms like the XAVIER AGX 16GB and ORIN NX 16GB, RDMM models still showed competitive results. For instance, on the ORIN NX 16GB, RDMM-0.5B delivered 6.12 T/s compared to Qwen2-0.5B's 9.90 T/s. Even on the entry-level XAVIER NX 8GB, where only RDMM-0.5B could run, it managed 3.75 T/s, showcasing the model's inference on limited hardware.

### *D. Real World Evaluation*

The real-world evaluation of the RDMM models took place during the RoboCup@Home Competition, using Lucio, a custom-built home service robot platform. In this environment, the RDMM models were responsible for handling various household and service-oriented tasks that required not only decision-making but also a level of self-awareness. These tasks involved navigating through complex environments, following people while carrying luggage, and guiding individuals to specific locations. Lucio's ability to understand its role was essential in tasks such as acting as a receptionist or handing items to people, where it needed to interact naturally and engage in small talk, as shown in Fig.3. An example of this is guiding a person while engaging in small talk about a specific topic, highlighting how self-awareness improves interaction and enhances service quality in real-world situations.

## V. Conclusion

This research presents the development and deployment of RDMM models, addressing key challenges that LLMs face when applied to domain-specific tasks. By integrating personal contextual knowledge into the decision-making process, RDMM models offer enhanced capabilities for self-aware planning, interaction, and task execution. Unlike existing methods, which struggle to incorporate an agent's personal background and specific skills Our approach demonstrates the viability of running powerful language models locally on edge devices without compromising accuracy at a promising inference speed, even on devices with as little as 8GB of memory. This achievement not only enhances the autonomy of robots in practical applications but also reduces reliance on external cloud-based systems, making it an affordable solution. The comprehensive dataset we constructed, including task-specific scenarios and self-awareness-oriented examples, lays the groundwork for future advancements in self-aware robotic planning and interaction.

## References

[1] D. D. et al., "Palm-e: An embodied multimodal language model," 2023. [Online]. Available: https://arxiv.org/abs/2303.03378
[2] E. C. et. al, "Open x-embodiment: Robotic learning datasets and rt-x models," 2024.
[3] S. V. et al., "Chatgpt for robotics: Design principles and model abilities," 2023. [Online]. Available: https://arxiv.org/abs/2306.17582
[4] M. A. et al., "Do as i can, not as i say: Grounding language in robotic affordances," 2022. [Online]. Available: https://arxiv.org/abs/2204.01691
[5] W. H. et al., "Voxposer: Composable 3d value maps for robotic manipulation with language models," 2023. [Online]. Available: https://arxiv.org/abs/2307.05973
[6] H. Z. et al., "Llm-bt: Performing robotic adaptive tasks based on large language models and behavior trees," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 33. IEEE, May 2024, p. 16655–16661. [Online]. Available: http://dx.doi.org/10.1109/ICRA57147.2024.10610183
[7] W. H. et al., "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," 2022. [Online]. Available: https://arxiv.org/abs/2201.07207
[8] S. S. R. et al., "Planning with large language models via corrective re-prompting," January 2022. [Online]. Available: http://www.cs.utexas.edu/users/ai-labpub-view.php?PubID=127989
[9] K. L. et al., "Text2motion: from natural language instructions to feasible plans," *Autonomous Robots*, vol. 47, no. 8, p. 1345–1365, Nov. 2023. [Online]. Available: http://dx.doi.org/10.1007/s10514-023-10131-7
[10] I. S. et al., "Progprompt: Generating situated robot task plans using large language models," 2022. [Online]. Available: https://arxiv.org/abs/2209.11302
[11] D. Shah, B. Osinski, B. Ichter, and S. Levine, "LM-nav: Robotic navigation with large pre-trained models of language, vision, and action," in *6th Annual Conference on Robot Learning*, 2022. [Online]. Available: https://openreview.net/forum?id=UW5A3SweAH
[12] J. Wu, R. Antonova, A. Kan, M. Lepert, A. Zeng, S. Song, J. Bohg, S. Rusinkiewicz, and T. Funkhouser, "Tidybot: Personalized robot assistance with large language models," *Autonomous Robots*, 2023.
[13] B. L. et al., "Llm+p: Empowering large language models with optimal planning proficiency," 2023. [Online]. Available: https://arxiv.org/abs/2304.11477
[14] K. S. et al., "Vision-language interpreter for robot task planning," 2024. [Online]. Available: https://arxiv.org/abs/2311.00967
[15] J. L. et al., "Code as policies: Language model programs for embodied control," 2023. [Online]. Available: https://arxiv.org/abs/2209.07753
[16] A. Szot, M. Schwarzer, B. Mazoure, H. Agrawal, W. Talbott, K. Metcalf, N. Mackraz, D. Hjelm, and A. Toshev, "Large language models as generalizable policies for embodied tasks," *preprint*, 2023.
[17] AI@Meta, "Llama 3 model card," 2024. [Online]. Available: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md
[18] M. ai, "mistral-7b-instruct-0.3v," 2024. [Online]. Available: https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3
[19] A. Y. et al., "Qwen2 technical report," 2024. [Online]. Available: https://arxiv.org/abs/2407.10671
[20] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "Gptq: Accurate post-training quantization for generative pre-trained transformers," 2023. [Online]. Available: https://arxiv.org/abs/2210.17323
[21] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms," 2023.
[22] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=nZeVKeeFYf9
[23] X. D. et al., "Internlm-xcomposer2: Mastering free-form text-image composition and comprehension in vision-language large model," 2024. [Online]. Available: https://arxiv.org/abs/2401.16420