

Large Multimodal Model-Based Task-Oriented Communications for Autonomous Robot Agents

Jiseob Kim, Donghoon Kim, Jihoon Moon, and Byonghyo Shim

INMC, Department of Electrical and Computer Engineering, Seoul National University, Seoul, Korea

Email: {jskim, dhkim, jhmoon}@islab.snu.ac.kr, bshim@snu.ac.kr

Abstract—Large multimodal models (LMMs) enable task-level reasoning for physical AI by jointly processing language and sensing data, but their use in multi-agent robots is challenged by the prohibitive on-device inference cost and the congestion incurred by raw sensing data uplink. This paper proposes task-oriented wireless communications for autonomous robot agents (TCAA), an LMM-aided framework that decouples sensing from reasoning by transmitting compact task-relevant features. Each robot agent encodes camera observations using a vision transformer (ViT) and transmits product quantization (PQ) indices, while the central unit (CU) dequantizes the indices and performs LMM-based cross-modal reasoning to generate action commands. We further adopt reasoning-augmented fine-tuning to improve the robustness of action generation. Simulations in a Unity-based warehouse environment confirm that TCAA reduces uplink payload substantially and improves task success ratio compared with a JPEG-based uplink baseline under the same rate constraint.

I. INTRODUCTION

The dramatic success of Large Language Models (LLMs) has fundamentally changed human's expectations for machine capabilities, particularly in solving complex reasoning and task-based problems. A promising extension of this paradigm is Large Multimodal Models (LMMs), which are key ingredients for physical AI applications, such as robotics, smart manufacturing, and autonomous vehicles [1], [2]. By processing language and multimodal sensing information concurrently, LMMs can capture the semantic meaning and geometric structure of the surrounding environment, enabling more informed decision for downstream tasks [3], [4]. For example, it has been reported that LMM-based robotic system can adaptively control the robots by exploiting the camera images [5], [6].

While LMM-based frameworks excel in perception-reasoning quality, their deployment in practical multi-agent robot systems is difficult due to the high computational cost and power consumption of LMM. In fact, since each robot agent has limited memory, battery, and relatively small computational capabilities, deploying a full-scale LMM on each and every robot agent is prohibitive. A viable solution to circumvent this issue is to offload the computational load to the edge or cloud server [7], [8]. In doing so, the server equipped

with the graphics processing unit (GPU) can run large-scale LMM having several billion parameters, while the robot agents handles lightweight subtasks. In such centralized system, large uplink bandwidth is required to transmit the raw sensing data (e.g., RGB image, LiDAR point cloud, and depth maps) from an agent to the server [9], [10]. Often, due to wireless link congestion, queuing delay might increase rapidly, ending up with service failure.

An aim of this paper is to propose an LMM-aided wireless communication framework to support autonomous tasks of multi-agent robot systems. Key challenge of the task-oriented wireless communications for autonomous robot agents (TCAA) is how to alleviate computational burden and reduce communication overhead while accomplishing the task goal under a stringent delay constraint. Our core idea is to decouple sensing process from the LMM inference by transmitting *light* task-relevant semantics instead of *heavy* raw sensing data. TCAA addresses this issue by deploying LMMs at the edge/cloud server (we henceforth call this place the central unit (CU)), which generates the command using task-related features delivered from the robot agent. This feature-level design significantly reduces the computational burden of robot agents and also uplink rate, thereby increasing the mobility and battery time of robot agents.

In a nutshell, as depicted in Fig. 1, overall command generation process of TCAA is divided into two steps: 1) product quantization (PQ)-based sensing data encoding at agent and 2) LMM-based cross-modal inference at CU. Specifically, on the agent-side, each robot agent converts acquired vision sensing information using Vision Transformer (ViT)-based vision encoder, producing a sequence of high-dimensional, say 2k, visual feature tokens [11]. To further compress the obtained features, we use PQ, a vector quantization technique partitioning the high-dimensional source vector into multiple subvectors and then mapping each subvector to a codeword in the codebook [12]. By partitioning the feature vector into small-sized sub-feature vectors and then applying subspace-wise codebooks, PQ converts high-dimensional features into compact indices and thus reduces the payload substantially. In our implementation, our proposed compression reduces the visual-feature payload from 29.86 Mbps for unquantized feature transmission to 0.37 Mbps (reduction of more than 98.7%) with negligible degradation in task success probability.

On the CU-side, the CU dequantizes the received PQ indices to obtain an approximate feature vector. The concatenated feature vectors, along with the language task prompt

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) under the Graduate School of Artificial Intelligence Semiconductor(IITP-2025-RS-2023-00256081) grant funded by the Korea government(MSIT), and by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (2022M3C1A3099336).

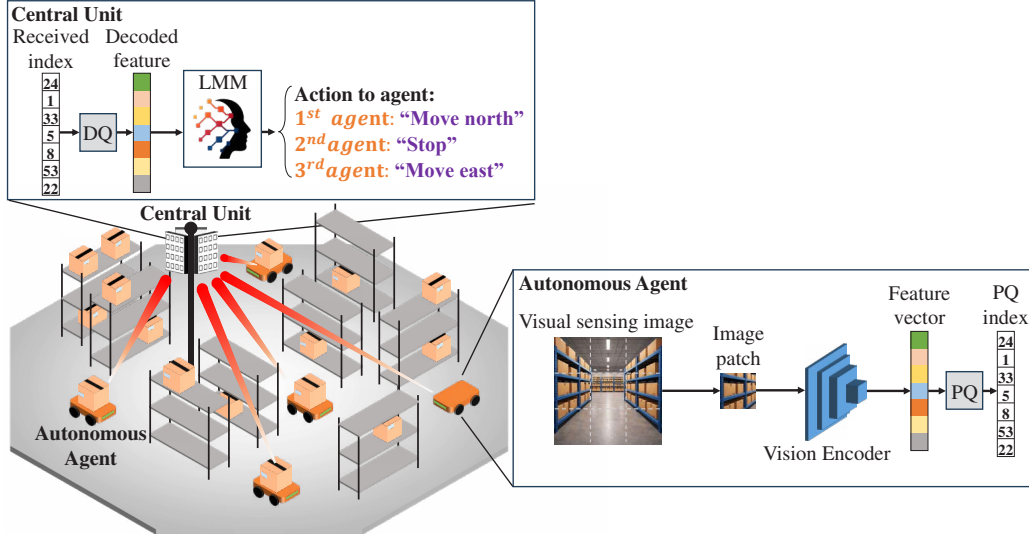


Fig. 1: Overview of the proposed TCAA. Each robot agent encodes its observation into feature vectors and uploads compact PQ indices. The CU dequantizes the indices and uses an LMM to generate action commands.

(e.g., "Where should the robot move next to complete its pick-and-place task?"), are used as inputs to LMM. Using these, LMM generates task-level action commands (e.g., "The robot should move north.") and then delivers them to the robot agent. In order to perform the designated task under the quantization and channel noise, the pretrained LMM needs to be further refined to the target task and environment via the task-specific fine-tuning. This process requires large-scale training data with sufficient diversity in both tasks and environments, which is obviously too expensive. To avoid the hassle, we collect a synthetic smart warehouse dataset using a Unity-based simulator [13] and then generate chain-of-thought (CoT)-structured text labels for each control step. Specifically, we use a structured text label consisting of an input prompt, intermediate reasoning sentences, and a robotic action sentence. Salient feature of our proposed TCAA is to generate reasoning sentences along with the robotic action sentence to better organize action-relevant information and generate more accurate context-consistent action. During the fine-tuning, the input prompt is provided to the LMM, and the model is trained to autoregressively generate the reasoning and action sentences sequentially, which helps the model organize decision-relevant information and generate outputs that are more consistent with the current context and surrounding environments.

The main contributions of this paper are summarized as follows:

- We propose a partitioned LMM-aided autonomous communication framework called TCAA. In TCAA, each robot agent performs PQ-based sensing data compression and a CU performs the LMM-based multimodal reasoning for the action generation. By offloading the LMM inference to CU, TCAA can support energy-efficient agent-side operation.

- We propose a sensing data compression technique based on the ViT encoder and PQ-based compression. By transmitting PQ-encoded indices, TCAA reduces the uplink payload by 98% relative to the full transmission approach, while improving the task success rate by 20% over the JPEG-based transmission.
- From the Unity-based smart-warehouse navigation task, we demonstrate that TCAA improves the task success ratio by 20% compared to the JPEG-based uplink baseline under the same uplink rate.
- In order to perform the CoT-style inference, we design a reasoning-based fine-tuning strategy that trains the LMM to generate a reasoning text before the action sentence generation. The fine-tuned model achieves around 10% higher success rate than the action-only fine-tuning.

II. TASK-ORIENTED WIRELESS COMMUNICATIONS FOR AUTONOMOUS ROBOT AGENTS

In this section, we present TCAA framework and the corresponding LMM training strategy. We first discuss the encoding strategy at the robot agent and decoding and LMM inference at CU. We then discuss the reasoning-augmented CoT supervision and the corresponding fine-tuning procedure for reliable action generation.

A. PQ-Based Agent Encoding

Consider K robot agents indexed by $k \in \{1, \dots, K\}$ over a discrete timeline $T = \{0, 1, 2, \dots\}$. At time t , robot agent k acquires the sensing image $\mathbf{I}_{k,t}$ and a robot agent state $\mathbf{s}_{k,t} = [x_{k,t}, y_{k,t}, h_{k,t}, \psi_{k,t}]$, where $(x_{k,t}, y_{k,t}, h_{k,t})$ and $\psi_{k,t}$ denote the position and heading angle. A ViT encoder E_θ partitions $\mathbf{I}_{k,t}$ into P patches and maps each patch to a feature vector $\mathbf{z}_{k,t,p} \in \mathbb{R}^D$. The resulting token features is denoted by $\{\mathbf{z}_{k,t,p}\}_{p=1}^P$.

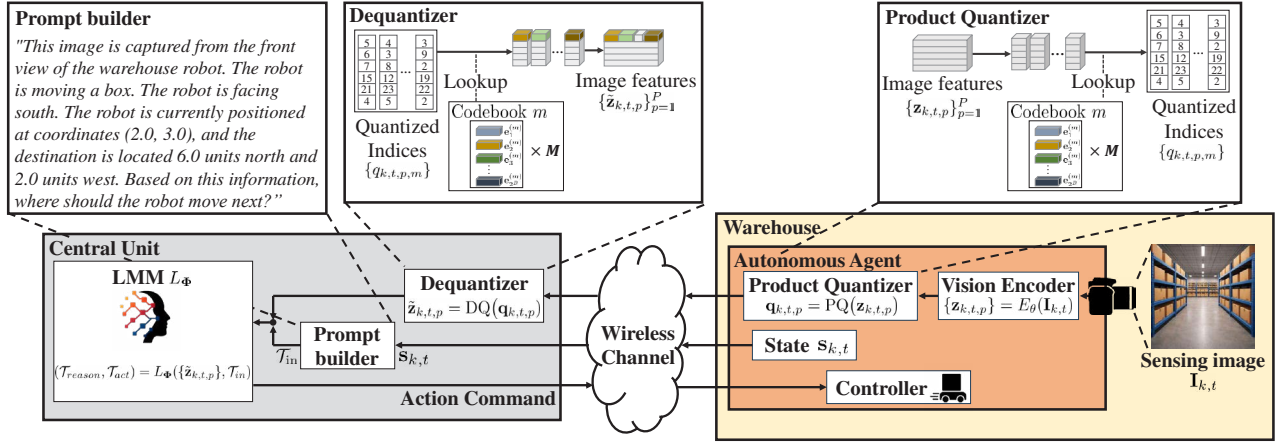


Fig. 2: Block diagram of the TCAA. Each robot agent transmits the quantized feature indices to the CU. The CU dequantizes the received indices and generates an LMM-based action command.

To perform the bandwidth-efficient sensing data transmission, we apply PQ to each feature vector. First, each feature vector $\mathbf{z}_{k,t,p}$ is split into M subvectors of dimension $d = D/M$. For each subvector index $m \in \{1, \dots, M\}$, we have a set of representative codewords $\{\mathbf{e}_j^{(m)} \in \mathbb{R}^d\}_{j=1}^{2^B}$, where each codeword index can be represented by B bits. The m -th subvector $\mathbf{z}_{k,t,p}^{(m)}$ is mapped to the nearest codeword in Euclidean distance. The chosen codeword index $q_{k,t,p,m}$ is given by $q_{k,t,p,m} = \arg \min_j \|\mathbf{z}_{k,t,p}^{(m)} - \mathbf{e}_j^{(m)}\|_2^2$. Each robot agent transmits the resulting PQ indices $\{q_{k,t,p,m}\}$ together with the robot agent state $\mathbf{s}_{k,t}$ to the CU. At the CU, we dequantize the received indices by mapping each subvector index to the corresponding codeword, and then assemble the codewords to approximate the feature vector $\tilde{\mathbf{z}}_{k,t,p}$. Group of approximated feature vectors $\{\tilde{\mathbf{z}}_{k,t,p}\}_{p=1}^P$ is then used as an input to the LMM.

B. Reasoning-Augmented Supervision

Action-only supervision offers limited interpretability and often produces inconsistent outputs, since the labels contain the final action exclusively (e.g., "Move west") and do not specify action-relevant environment information. To provide more informative supervision, we train the LMM to generate explicit reasoning sentences before the action sentence. To this end, we design a special task label that specifies the context and rationale as well as action. Our proposed task label consists of three segments: 1) an *input prompt* $\mathcal{T}_{in}$ that summarizes the current observation, robot agent state, and goal for the next action decision, 2) *reasoning sentences* $\mathcal{T}_{reason}$ proving an interpretable rationale summarizing why the chosen command is appropriate for the given environment and context, and 3) an *action sentence* $\mathcal{T}_{act}$ that designates the final action command. At the CU, we fine-tune the LMM to autoregressively generate output sentences $\mathcal{T}_{out} = (\mathcal{T}_{reason}, \mathcal{T}_{act})$ when $\{\tilde{\mathbf{z}}_{k,t,p}\}_{p=1}^P$ and $\mathcal{T}_{in}$ are used as inputs. At inference time, the CU generates $\mathcal{T}_{out}$ and converts $\mathcal{T}_{act}$ (e.g., "The robot should stop") into a succinct command (e.g., "Stop") and then transmits it to the robot agent.

Examples of an input prompt, reasoning sentences, and an action sentence are shown below.

- Input prompt $\mathcal{T}_{in}$
"This image is captured from the front view of the warehouse robot. The robot is moving a box. The robot is facing south. The robot is currently positioned at coordinates (2.0, 3.0), and the destination is located 6.0 units north and 2.0 units west. Based on this information, where should the robot move next?"
- Reasoning sentences $\mathcal{T}_{reason}$
"Since the robot is currently positioned at (2.0, 3.0), and the destination is located 6.0 units north and 2.0 units west, for the given view image,"^a
- Action sentence $\mathcal{T}_{act}$
"The robot should move west."

^aIn our label format, $\mathcal{T}_{reason}$ and $\mathcal{T}_{act}$ are combined into output sentences. Accordingly, $\mathcal{T}_{reason}$ is written as a subordinate prefix that leads into the action sentence.

C. TCAA Training Strategy

Using the task label defined in Section II-B, we train two components, viz., PQ codebooks and LMM, of the TCAA. First, we train PQ codebooks to support the bandwidth-efficient uplink transmission. Once we finish this step, the learned PQ codebooks are frozen and then we perform the fine-tuning of LMM to autoregressively generate the output text $\mathcal{T}_{out} = (\mathcal{T}_{reason}, \mathcal{T}_{act})$ from the visual features and input prompt $\mathcal{T}_{in}$.

1) *PQ Codebook Training*: To support index-based uplink transmission, we train PQ codebooks using ViT token features $\mathbf{z}_{k,t,p}$ extracted from the training images. We keep the ViT encoder E_θ fixed and train the codewords $\{\mathbf{e}_j^{(m)}\}_{j=1}^{2^B}$ for each subvector $m \in \{1, \dots, M\}$. Separate sub-codebooks better match the distinct statistics across subvectors. We optimize codewords by minimizing the PQ reconstruction loss:

$$\mathcal{L}_{PQ} = \sum_{(k,t) \in \mathcal{D}_u} \sum_{p=1}^P \sum_{m=1}^M \|\mathbf{z}_{k,t,p}^{(m)} - \mathbf{e}_{q_{k,t,p,m}}^{(m)}\|_2^2, \quad (1)$$

where $\mathbf{z}_{k,t,p}^{(m)}$ is the m -th subvector of $\mathbf{z}_{k,t,p}$ and $\mathcal{D}_{tr}$ is the training set for the codebook learning. In practice, we train each sub-codebook using a mini-batch k-means procedure that alternates between nearest-codeword assignment and codeword updates. Each mini-batch consists of feature vectors extracted from a training image, and codewords are updated via a running mean of the assigned samples.

2) *Reasoning-Augmented LMM Fine-Tuning*: For each training sample indexed by (k, t) , we dequantize the received PQ indices to obtain the sequence of approximate visual features $\{\tilde{\mathbf{z}}_{k,t,p}\}_{p=1}^P$. By concatenating these and token-embedding sequence $\mathbf{w}_{k,t}$ of the input prompt $\mathcal{T}_{in}$, we generate the input sequence $\mathbf{x}_{k,t} = [\tilde{\mathbf{z}}_{k,t,1}; \tilde{\mathbf{z}}_{k,t,2}; \dots; \tilde{\mathbf{z}}_{k,t,P}; \mathbf{w}_{k,t}]$ for the fine-tuning LMM. We fine-tune the LMM to output $\mathcal{T}_{reason}$ before $\mathcal{T}_{act}$, inducing explicit intermediate reasoning. To autoregressively generate the target output text $\mathcal{T}_{out} = (\mathcal{T}_{reason}, \mathcal{T}_{act})$, we employ the low-rank adaptation (LoRA) [14]. Let Φ be the pretrained LMM parameters and $\Delta\Phi$ be the LoRA parameters, then we optimize $\Delta\Phi$ by minimizing the token-level cross-entropy loss between the predicted next-token distribution $\mathbf{p}_{\Phi+\Delta\Phi}(\cdot | \mathbf{x}_{k,t}, u_1, \dots, u_{i-1})$ and the ground-truth next token u_i :

$$\mathcal{L}_{CE} = - \sum_{i=1}^{|\mathcal{T}_{out}|} \mathbf{v}_i^T \log \mathbf{p}_{\Phi+\Delta\Phi}(\cdot | \mathbf{x}_{k,t}, u_1, \dots, u_{i-1}) \quad (2)$$

$$= - \sum_{i=1}^{|\mathcal{T}_{out}|} \log p_{\Phi+\Delta\Phi}(u_i | \mathbf{x}_{k,t}, u_1, \dots, u_{i-1}), \quad (3)$$

where $\mathbf{v}_i$ is the one-hot vector corresponding to token u_i .¹

III. SIMULATION RESULTS

A. Evaluation Strategy and Setup

To validate the effectiveness of the TCAA framework, we conduct robot navigation experiments in a Unity-based virtual simulator. We design the warehouse with 18 shelf racks arranged in a 3×6 array, forming narrow aisles and cross-aisles that induce occlusions and constrained turning spaces. In each episode, we randomly sample a feasible start-goal pair from the navigable area of the warehouse. We also randomly add dynamic obstacles including four moving pedestrians and two moving robots, each following randomized waypoint trajectories. An episode is terminated when an agent reaches the goal or the episode length exceeds T_{max} control steps (T_{max} is set to 600 in our simulation).

The simulation follows the client-server architecture described in Fig. 3. At time step t , a robot agent in the Unity client captures a 1920×1080 front-view sensing image and the robot agent state. After the compression, this information is transmitted to the Python server acting as the CU. As an LMM model at CU, we employ TinyLLaVA-3.1B, a large multimodal model that takes image-text pair as inputs and generates text outputs.

In TCAA, each 1920×1080 sensing image is mapped to 729 patches, each of which is mapped into a 2560-dimensional

¹For example, if u_i corresponds to the token indexed by j in the vocabulary set, $\mathbf{v}_i$ is the one-hot vector with a single 1 at position j .

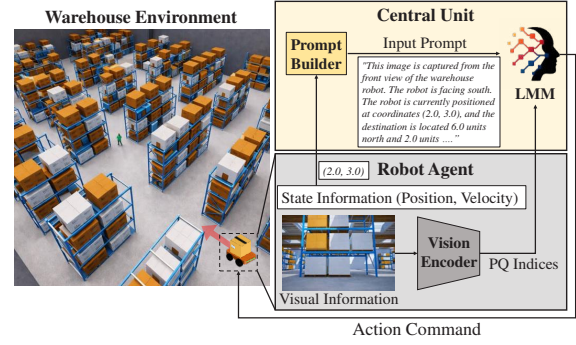


Fig. 3: Simulation environments and setup for evaluating the TCAA. Robot agents transmit quantized PQ indices and state information to a CU and the LMM at CU generates and transmits the next action command to the robot agent.

feature vector. Each feature vector is split into M subvectors and then each subvector is quantized into 8-bit PQ indices. In our experiments, we test three subvector dimensions with $M \in \{64, 80, 128\}$. For comparison, we test an uncompressed feature-transmission scheme and a JPEG-based compression scheme. The uncompressed transmission follows the same CU-side LMM control pipeline as TCAA but the raw ViT feature vectors are transmitted without quantization. In the JPEG-based scheme, each sensing image is compressed by the JPEG quality factor $Q \in \{60, 80\}$.

B. Uplink Load and Task Success Rate

We evaluate the navigation task success rate under various SNR conditions from 0 dB to 30 dB. Fig. 4(a) summarizes the required uplink rate under the same sensing configuration and uplink update rate. Overall, the required rate of TCAA is significantly lower than uncompressed feature transmission and JPEG-based sensing information transmission, achieving 98% and 71% reduction, respectively. In Fig. 4(b), we plot the task success rate as a function of SNR when the uplink rate is fixed. To ensure this condition, we adjust the uplink update rate of the transmission scheme such that all schemes operate at the same uplink rate. For the steps where uplink information is unavailable, the CU reuses the most recent sensing information in the LMM inference. Under this setting, TCAA achieves about 70% success rate at 0 dB, whereas JPEG stays around 50% and the uncompressed feature-uplink scheme drops to 8%. At high SNR, TCAA consistently outperforms competing schemes by 16~25%.

C. Effect of Reasoning-Augmented Supervision

As introduced in Section II-B, we perform the fine-tuning with a structured task label. To quantify the effect of the proposed task label, we perform an ablation study with three strategies: zero-shot, action-only fine-tuning, and TCAA. As a zero-shot baseline, we use the pretrained TinyLLaVA-3.1B model without any task-specific fine-tuning. The second baseline is an action-only fine-tuned model, in which the LMM is trained to generate $\mathcal{T}_{act}$ directly from $\mathcal{T}_{in}$, without including explicit reasoning.

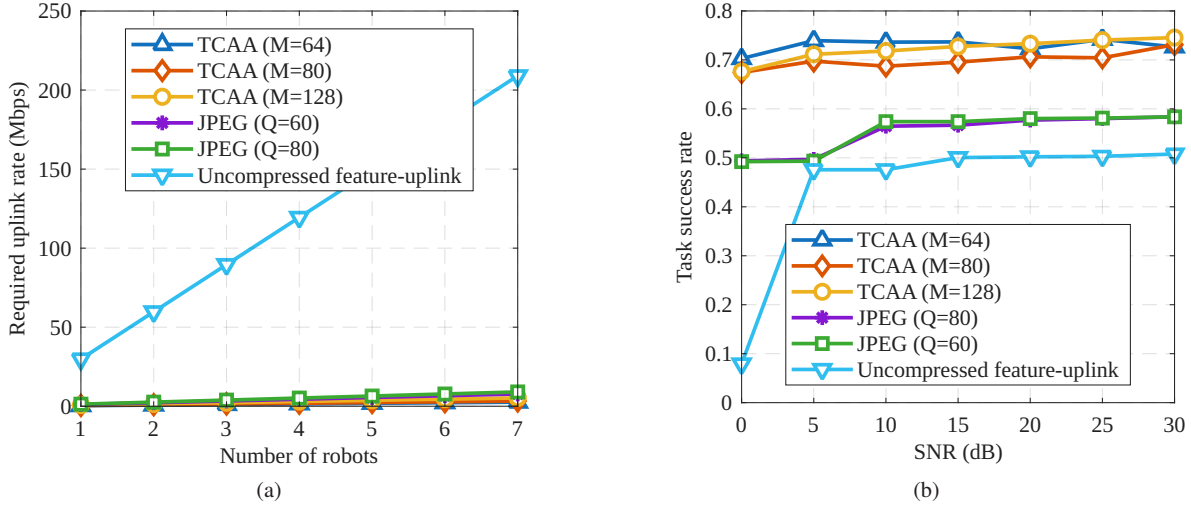


Fig. 4: Required uplink rate versus number of robots and task success rate versus SNR.

TABLE I: Task success rate in a noiseless uplink setting.

Framework	Task success rate
Uncompressed feature-uplink (zero-shot)	27.56%
TCAA (M=64, zero-shot)	30.14%
TCAA (M=80, zero-shot)	30.30%
TCAA (M=128, zero-shot)	29.11%
Uncompressed feature-uplink (w/o reasoning)	73.95%
TCAA (M=64, w/o reasoning)	63.29%
TCAA (M=80, w/o reasoning)	65.54%
TCAA (M=128, w/o reasoning)	67.99%
Uncompressed feature-uplink	77.04%
TCAA (M=64)	73.32%
TCAA (M=80)	73.11%
TCAA (M=128)	74.92%

As shown in Table I, zero-shot baseline achieves only around 30% success rate, confirming that task-specific fine-tuning is needed for reliable control in our navigation scenarios. In fact, reasoning-augmented supervision improves TCAA performance from 67.99% to 74.92% (for $M = 128$), which clearly demonstrates the effectiveness of the proposed approach.

IV. CONCLUSION

In this paper, we proposed TCAA, an LMM-aided wireless communication framework to support autonomous operations of multi-agent robot systems under a stringent delay constraint. The key idea of TCAA is to decouple sensing from the LMM inference by transmitting the quantized indices of task-relevant sensing information to alleviate both computational burden and communication overhead. To this end, we designed the PQ-based sensing compression technique and the reasoning-augmented LMM fine-tuning strategy using CoT-style structured labels. Simulation results in a Unity-based warehouse environment demonstrate the effectiveness of TCAA as a practical solution for LMM-based multi-agent robot control. Future work includes validating TCAA on real robot systems and extending the framework to additional sensing modalities, including depth and LiDAR.

REFERENCES

- [1] H. Lee, B. Lee, H. Yang, J. Kim, S. Kim, W. Shin, B. Shim, and H. V. Poor, "Towards 6G hyper-connectivity: Vision, challenges, and key enabling technologies," *J. Commun. Netw.*, vol. 25, no. 3, pp. 344–354, 2023.
- [2] Y. Hu, F. Wang, D. Ye, M. Wu, J. Kang, and R. Yu, "LLM-based misbehavior detection architecture for enhanced traffic safety in connected autonomous vehicles," *IEEE Trans. Veh. Technol.*, 2025.
- [3] I. Lee, W. Park, J. Jang, M. Noh, K. Shim, and B. Shim, "Towards comprehensive scene understanding: Integrating first and third-person views for lvmis," *Adv. Neural Inf. Process. Syst.*, vol. 38, pp. 18576–18628, 2026.
- [4] S. Kim, S. Saha, S. Jeong, B. Shim, and M. Z. Win, "Large multimodal model-based environment-aware beam management," *IEEE J. Sel. Areas Commun.*, 2026.
- [5] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al., "RT-2: Vision-language-action models transfer web knowledge to robotic control," in *Proc. Conf. Robot Learn. (CoRL)*, 2023, pp. 2165–2183.
- [6] H. J. Yang, H. Lee, K. Shim, J. Kwak, H. Kim, D. Kim, K. A. Ngo, S. Ryu, J. Choi, Y. Kim, et al., "Advancing Multi-Robot Networks via MLLM-Driven Sensing, Communication, and Computation: A Comprehensive Survey," *IEEE Commun. Surveys Tuts.*, 2026.
- [7] W. Zhan, C. Luo, G. Min, C. Wang, Q. Zhu, and H. Duan, "Mobility-aware multi-user offloading optimization for mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3341–3356, 2020.
- [8] Y. Su, W. Fan, L. Gao, L. Qiao, Y. Liu, and F. Wu, "Joint DNN partition and resource allocation optimization for energy-constrained hierarchical edge-cloud systems," *IEEE Trans. Veh. Technol.*, vol. 72, no. 3, pp. 3930–3944, 2022.
- [9] S. Kim, J. Moon, J. Kim, Y. Ahn, D. Kim, S. Kim, K. Shim, and B. Shim, "Role of sensing and computer vision in 6G wireless communications," *IEEE Wireless Commun.*, 2024.
- [10] H. J. Yang, H. Kim, H. Noh, S. Kim, and B. Shim, "Large Multimodal Model-Empowered Task-Oriented Autonomous Communications: Design Methodology and Implementation Challenges," *IEEE Veh. Technol. Mag.*, 2025.
- [11] J. Kim, K. Shim, J. Kim, and B. Shim, "Vision transformer-based feature extraction for generalized zero-shot learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2023, pp. 1–5.
- [12] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 117–128, 2010.
- [13] A. Juliani, V.-P. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Mattar, et al., "Unity: A General Platform for Intelligent Agents," 2018, arXiv:1809.02627.
- [14] E. J. Hu et al., "LoRA: Low-Rank Adaptation of Large Language Models," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022.