2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

AssemMate: Graph-Based LLM for Robotic Assembly Assistance

Qi Zheng†, Chaoran Zhang†, Zijian Liang†, EnTe Lin, Shubo Cui, Qinghongbing Xie, Zhaobo Xu, Long Zeng*

***Abstract*— Large Language Model (LLM)-based robotic assembly assistance has gained significant research attention. It requires the injection of domain-specific knowledge to guide the assembly process through natural language interaction with humans. Despite some progress, existing methods represent knowledge in the form of natural language text. Due to the long context and redundant content, they struggle to meet the robots' requirements for real-time and precise reasoning. In order to bridge this gap, we present a novel graph-based LLM, denoted as AssemMate, which consists of two stages: graph-based question answering and vision-enhanced grasp execution. The first stage enables natural language question answering on a knowledge graph, supporting human-robot interaction and assembly task planning for specific products. The second stage then utilizes the planning generated before as a target, senses stacked scenes, and executes grasping to assist with assembly. Specifically, a self-supervised Graph Convolutional Network (GCN) encodes knowledge graph entities and relations into a latent space and aligns them with LLM's representation, enabling the LLM to understand graph information. In addition, a vision-enhanced strategy is employed to address stacked scenes in grasping. Through training and evaluation, AssemMate outperforms existing methods, achieving 6.4% higher accuracy, 3 times faster inference, and 28 times shorter context length, while demonstrating strong generalization ability on random graphs. And our approach further demonstrates superiority through robotic grasping experiments in both simulated and real-world settings. More details can be found on the project page `https://github.com/cristina304/AssemMate.git`.**

I. INTRODUCTION

Recently, the advancements in LLMs have shown promising results in robotic assembly assistance [1]–[3]. LLM-based robotic assembly assistance, as shown in Fig. 1, requires the injection of domain-specific knowledge into LLM, supporting real-time interaction with humans to guide accurate assembly. It also involves executing physical operations, such as grasping and installing, to improve assembly efficiency. The primary challenge lies in how to effectively and accurately inject the indispensable domain-specific knowledge into LLM. Existing methods attempt to do so by using natural language text as external knowledge. However, the long context length and redundant content of such text hinder the ability to meet the real-time and precise reasoning requirements in robotics [4].

Compared to text, the graph is a concise and precise data structure, as shown in Fig. 1(a). There has been widespread research on transferring the text comprehension abilities of LLMs to graph structures [5]–[8]. Among them, the two most prominent approaches are representing graphs as text sequences [9]–[11] or using Graph Neural Networks (GNNs) to encode graphs before feeding them into LLMs [12]–[14]. These studies are limited to classical graph learning tasks, such as link prediction and node classification. In contrast, robotic assembly assistance requires knowledge graph question answering (KGQA) for human-robot interaction, guiding the correct assembly process and planning downstream operations. Moreover, the specific types of relationships in graphs are crucial, yet their representations in existing graph-based LLMs are limited to the simple existence or non-existence of edges. These limitations hinder the application of graph in robotic assembly assistance.

In response to these limitations, we propose AssemMate, a novel graph-based LLM that leverages GCN to vectorize graphs and encode relations for knowledge injection into LLM. Specifically, as shown in Fig. 1(b), all entities and relations in the knowledge graph are encoded via a Graph Encoder to capture both structural and semantic information, and the resulting embeddings are subsequently aligned with the LLM's representation. By injecting assembly knowledge, LLM enables human-robot interaction and assembly task planning in the form of efficient and accurate natural language KGQA. Furthermore, vision enhancement is applied to sense stacked scenes, and the robot executes the grasp to assist assembly. In addition, considering the fact that domain-specific graphs lack corresponding QA data to support network training, we propose a method for automatically constructing diverse QA datasets.

In the interactive QA phase, we use accuracy to measure single-hop QA performance, while for multi-hop QA, we use normalized LCS (nLCS) to assess sequence consistency and Weighted Jaccard Similarity (wJaccard) to evaluate content accuracy. In the grasping phase, we use the optimal planning rate (OPR) to measure the grasping planning ability in stacked scenes. Experimental results demonstrate that AssemMate achieves an accuracy of 82.1% on single-hop QA, with an average inference time of only 0.48s. For multi-hop QA, it attains 66.7% nLCS and 65.3% wJaccard. The comparative experiments reveal that AssemMate surpasses existing methods with 6.4% higher accuracy, 3 times faster inference, and 28 times shorter context length on single-hop QA, as well as achieving over 20% higher nLCS and wJaccard on multi-hop QA. When tested on random graphs, the QA performance remains stable, validating the generalization capability of our method. For grasping tasks

Video available at `https://youtu.be/eX5uBRrdV6s`.

† Equal contribution. (e-mail: zheng-q25@mails.tsinghua.edu.cn)
* corresponding author.
Qi Zheng, Chaoran Zhang, Zijian Liang, EnTe Lin, Shubo Cui, Qinghongbing Xie, Zhaobo Xu, and Long Zeng are with Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China.

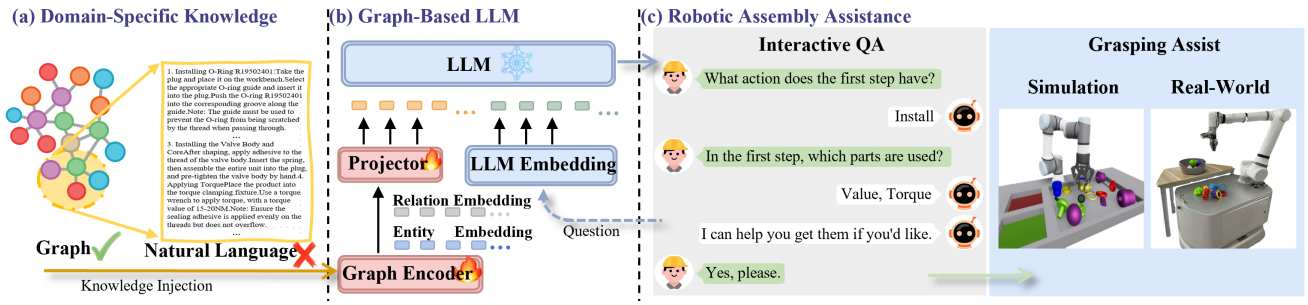

Fig. 1. Our method, AssemMate, efficiently injects concise graph-structured data into LLM as external knowledge to enable interactive QA during the assembly process. It further supports grasping in stacked scenes, serving as an intelligent mate for robotic assembly assistance.

in both simulation and real-world scenarios, the OPR reaches 71.2% and 64.3%, respectively.

In summary, the main contributions of our work are:

- To the best of our knowledge, we are the first to inject graph-structured data as domain-specific knowledge into LLM for robotic assembly assistance.
- We propose AssemMate, a pioneering approach that realizes natural language KGQA to support human-robot interaction and assembly task planning, while also enabling the modeling of specific relationships for accurate assembly assistance.
- Extensive experiments show that our approach outperforms existing methods with 6.4% higher accuracy, three times faster inference, and 28 times shorter context length.

II. Related Work

The most closely related topics are reviewed in this section, the other robotic assembly assistant techniques, such as knowledge driven assembly model [15], assembly control language [16], can be found in the given reference paper.

A. Large Language Models for Graphs

Exploring whether LLMs can transfer their text comprehension abilities to graphs has garnered significant attention [8]. Currently, two main approaches exist: The first method involves converting graph information into natural language prompts, allowing LLMs to perform tasks without additional training, as in GraphText [17] and Graph-LLM [18], or introducing a projection layer as in LLaGA [10] to further leverage the text information. The second method [9], [13] uses a GNN encoder to tokenize the graph before processing it with the LLM. However, research still primarily focuses on traditional graph learning tasks, including node classification, link prediction, and graph classification, without fully leveraging the natural language capabilities of LLMs to unify these tasks into graph-based question answering and solve them with a unified paradigm. Apart from this, these methods lack explicit modeling of different relations, making them unsuitable for the practical needs of the robotics domain.

In contrast, our AssemMate requires only basic graph triples, a fundamental and interpretable graph description form. By leveraging a self-supervised GCN to encode both entities and relations, we obtain the embedding representation of the graph, which is then integrated with the LLM through a simple and flexible projector.

B. Language-Guided Grasping

Different to deep learning based grasping pose estimation [19]–[21]. Large-scale models pretrained on vast amounts of internet data have demonstrated zero-shot generalization ability to unseen scenarios [22], [23]. Research on using LLMs, Vision-Language Models (VLMs), and Vision-Language-Action (VLA) models for grasping tasks is gaining significant attention. Some methods [24]–[26] leverage the language processing capabilities of LLMs to enhance robots' understanding of human instructions. ThinkGrasp [27] and VLG [28] directly utilize the vision-language alignment capabilities of Multi-modal Large Language Models (MLLMs) to enable efficient grasping. VLA [29], [30], as an end-to-end approach for generating grasp poses, is also a current research hotspot, but it is greatly limited by dataset issues and the sim-to-real gap. Despite significant progress, these methods could further exploit the power of pre-trained models and place more emphasis on real-world stacked scenarios.

On the contrary, our AssemMate determines the target object based on interactive QA, and adopts vision augmentation to compensate for the limitations of MLLMs in spatial understanding and industrial parts perception. This enables it to handle stacked situations, achieving reliable object grasping for robotic assembly assistance.

III. Problem Statement

We define the tasks that AssemMate can perform as follows: Given a human language query Q and an assembly knowledge graph G, it can perform both single-hop and multi-hop question answering. The answers A contain both assembly knowledge K and the sequence of assembly parts S, enabling human-robot interaction and assembly task planning. Furthermore, based on the part sequence S, AssemMate can execute grasping operations $\hat{A}$ in stacked scenarios Env to assist with human assembly. The overall process is defined as follows:

$$\begin{aligned} &\text{AssemMate}(G, Q) \rightarrow A \\ &\text{If required AssemMate}(A, Env) \rightarrow \hat{A} \end{aligned} \tag{1}$$

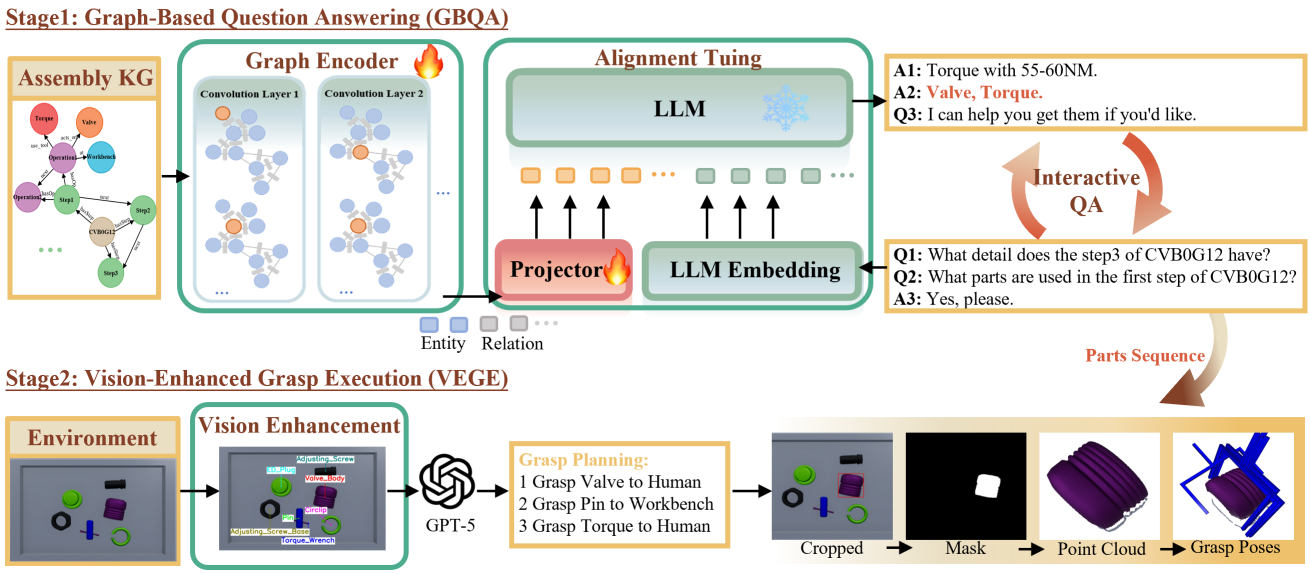

Fig. 2. Framework of AssemMate, a self-supervised GCN to encode knowledge graph entities and relations into a latent space, aligning them with LLM's representation. Based on the KGQA, VEGE leverages MLLMs with vision enhancement to sense stacked scenes and generate grasping plans, followed by segmentation and grasp poses generation.

In this process, the assembly knowledge graph $G = \langle \mathcal{N}, \mathcal{E} \rangle$, where each entity $n_i \in \mathcal{N}$ represents *a part*, *tool*, or *workspace*. Each edge $\epsilon_i \in \mathcal{E}$ connects two entities, representing a relation type $r_i \in \mathcal{R}$, such as *acts_on*, *acts_to*, *uses_tool*, etc. The input knowledge graph can be created manually or automatically [31].

Additionally, the two types of question answering are defined as follows.

- **Single-hop Question Answering**: Understanding the question and reaching the answer entity from a given entity through only one edge in the graph.
- **Multi-hop Question Answering**: Understanding the question and reaching the answer entities from a given entity by traversing multiple entities or edges in the graph.

IV. Method

AssemMate can be divided into two stages, including Graph-Based Question Answering (GBQA) and Vision-Enhanced Grasp Execution (VEGE). The implementation details are depicted in Fig. 2. AssemMate, an intelligent mate designed for industrial assembly scenarios, can seamlessly understand the injected knowledge graph and interact with humans through QA to guide correct assembly. Moreover, assembly task planning can be realized within this interaction process. And AssemMate is also capable of sensing stacked scenes and executing grasping to complete the planning, achieving efficient and accurate robotic assembly assistance.

A. Graph-Based Question Answering

Graph-Based Question Answering can be divided into two parts: Graph Encoder and Alignment Tuning. The first part encodes the information of the knowledge graph through a self-supervised learning GCN, mapping it to a latent space representation. $E_G = [e_1, e_2, \ldots, e_m] \in \mathbb{R}^{m \times d'}$, where m is the total number of entities $\mathcal{N}$ and relations $\mathcal{R}$ in the graph $\mathcal{G}$. The second part achieves alignment between E_G and the LLM's pre-training representation by training two linear projection layers, allowing the LLM to understand the graph information.

Graph Encoder. Considering the complexity of relationships in industrial assembly, we use GCN on multi-relational graphs as an encoder. We adopt the graph encoding method from CompGCN [32]. After comparative experiments, we chose Corr [33] as the entity-relation composition operation, which is defined as:

$$r = \phi(h, t) = h \odot t, \tag{2}$$

where $\phi : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ is a composition operator, $h \in \mathbb{R}^d$, $r \in \mathbb{R}^d$, and $t \in \mathbb{R}^d$ are the embedding vectors of head entity, relation, and tail entity, respectively. The symbol $\odot$ represents element-wise multiplication.

Additionally, we use DistMult [34] as the score function:

$$\text{score}(h, r, t) = \sum_{i=1}^{d} h_i \cdot r_i \cdot t_i, \tag{3}$$

Alignment Tuning. By using the GCN, the structural information of the graph is stored in a latent space, where each entity or relation has a corresponding $\mathbb{R}^d$ embedding. To validate the method's ability to learn graph structural information, we adopt an unordered knowledge graph representation method to characterize the entire graph.

Specifically, for all entities and relations in the dataset, we maintain a dictionary in random order. For a graph with a total of m entities and relations, its vectorized representation is the ordered sequence of these $\mathbb{R}^d$ embeddings as they

appear in the dictionary, yielding:

$$E_{\text{structural}} = [k_1, k_2, \ldots, k_m] \in \mathbb{R}^{m \times d}, \tag{4}$$

where $k_i \in \mathbb{R}^d$ is the structural information vector of the i-th entity or relation in this graph.

Given that structural and semantic information are both important for zero-shot transfer [6], we further encode the textual information represented by entities or relations using BERT [35], yielding:

$$E_{\text{semantic}} = [t_1, t_2, \ldots, t_m] \in \mathbb{R}^{m \times d}, \tag{5}$$

where $t_i \in \mathbb{R}^d$ is the semantic information vector of the i-th entity or relation in this graph.

Then, considering that LLM alone cannot fully capture the structural and semantic information of knowledge graphs, and that discrepancies exist between the two. We propose using two separate two-layer linear projectors for Alignment Tuning:

$$E_{\text{structural}} = \text{softmax}(\text{MLP}_{\text{struct}}(E_{\text{structural}})), \tag{6}$$

$$E_{\text{semantic}} = \text{softmax}(\text{MLP}_{\text{sem}}(E_{\text{semantic}})), \tag{7}$$

$$E_G = \text{concat}(E_{\text{structural}}, E_{\text{semantic}}) \in \mathbb{R}^{m \times d'}, \tag{8}$$

where d' is the embedding dimension of the LLM. Finally, we combine E_G with the user input instruction and train the projectors on a frozen-weight LLM to enable question answering about graph knowledge.

Loss Function. Consistent with language autoregressive models, we use the standard cross-entropy loss function L_{ce}. Additionally, we incorporate a top-k sparsity penalty term L_{penalty} along with a restriction mechanism:

$$L_{\text{penalty}} = \frac{\lambda_{\text{penalty}}}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} \left(\text{logits}_{\text{top-k}}[i, j] \right)^2, \tag{9}$$

$$L = L_{\text{ce}} + \min\left(L_{\text{penalty}},\ L_{\text{ce}} \cdot \text{max\_ratio}\right), \tag{10}$$

where λ_{penalty} controls the strength of the penalty term, N is the sequence length, K is the number of top-k values, $\text{logits}_{\text{top-k}}$ represents the top-k largest logits, and max_ratio limits the maximum contribution of the penalty term to the overall loss function.

B. Vision-Enhanced Grasp Execution

In industrial assembly, grasping in cluttered scenes is a critical issue that must be addressed. To tackle this, we apply vision enhancement, which leverages the powerful reasoning capabilities of MLLMs to analyze the information in the current scene and generate optimal grasping plans. After that, VEGE detects the position of the target objects in the scene and generates the optimal grasp pose.

Vision Enhancement. Aiming to improve the spatial perception and cross-modal alignment abilities of MLLMs, we introduce a text- and arrow-based visual prompt enhancement method, which annotates object names with text labels and uses arrows to connect the text labels to the object's center, as shown in Fig. 3. This improves the cross-modal alignment capability of the model. To reduce occlusion of the original image, text labels are placed in background areas that are weakly related to the task, and the scanning circle strategy is employed to select the most suitable position for placing the text labels.

TABLE I
EXAMPLE QUESTIONS FOR SINGLE-HOP QA

Category	Example Questions
Action	What action is performed in step1 of the CV01S? What action is done by step1 of the CV01S ? What action does step1 of the CV01S carry out?
Tools	What tool is used in step1 of the CV01S ? Which tool is utilized by step1 of the CV01S ? Can you name the tools used by step1 of the CV01S ?
Workspace	To what position does step1 of the CV01S act? Where does step1 of the CV01S act to? To which location does step1 of the CV01S act?
Detail	What detail does step1 of the CV01S provide? What detail is given for step1 of the CV01S? What detail is associated with step1 of the CV01S ?

V. EXPERIMENTS

A. Dataset

We propose an automated method for constructing diverse QA datasets. Specifically, for a given knowledge graph, we decompose it entirely into $<$ *head entity*, *relation*, *tail entity* $>$ triples. Then, we use chain-of-thought (CoT) reasoning [41] and few-shot [42] strategies to guide LLM in generating rich and diverse single-hop and multi-hop QA pairs based on these triples, involving *actions*, *tools*, *workspaces*, *subassemblies*, *assemblies*, *object attributes*, and *assembly details* during the assembly process.

Examples of single-hop QA are shown in Table I, while examples of multi-hop QA are as follows:

- *List the parts and tools used in step1 of the CV01S, in the order they are used.*
- *In step1 of the CV01S, what are the parts and tools involved sequentially?*
- *Identify the sequential parts and tools used in step1 of the CV01S.*

where *CV01S* represents the type of the pressure-reducing valve. The answers for multi-hop QA consist of sequences of objects involved in the assembly in order, which serve as plans for downstream grasping operations.

Assembly knowledge graphs are obtained from multiple information sources, such as assembly process documents and 3D models of assemblies. We used the assembly graphs of 12 pressure-reducing valves as the training set, applying a strategy that randomly splits the entire graph into subgraphs, thereby expanding the training dataset to 148 graphs to improve generalization. Using our QA construction strategy, we generated a dataset of 22,000 examples for training. The testing set consists of the assembly graphs of 6 products,

TABLE II

COMPARISON EXPERIMENT AGAINST THE USE OF NATURAL LANGUAGE TEXT AS DOMAIN-SPECIFIC KNOWLEDGE INJECTION.

Method	Single-hop QA			Multi-hop QA	
	In-context Length ↓	Infer Time (s) ↓	Acc. (%) ↑	nLCS (%) ↑	wJaccard (%) ↑
Text Triple [36]	3510	1.50	75.7	45.7	43.4
Text Triple + Fine-tuning [37]	3510	1.51	78.4	47.0	45.2
Document [38]	764	0.85	50.0	46.3	43.4
Document + Fine-tuning [39], [40]	764	0.83	75.8	53.6	53.8
Ours (Graph + Projector)	**125**	**0.48**	**82.1**	**66.7**	**65.3**

TABLE III

ABLATION STUDY ON ASSEMMATE MODULE. W/O, WITHOUT

Method	Single-hop QA		Multi-hop QA	
	Infer Time (s) ↓	Acc. (%) ↑	nLCS (%) ↑	wJaccard (%) ↑
w/o GCN	0.48	78.9	56.0	53.6
w/o relation	0.64	18.4	1.8	1.3
w/o entity	0.65	21.1	—	—
Ours	**0.48**	**82.1**	**66.7**	**65.3**

where the entire product graphs are used as input to align with real-world application scenarios.

In the Alignment Tuning stage, both the question and the corresponding graph knowledge are simultaneously injected into the LLM as input. To better distinguish the information, we adopt the following instruction format:
<kg_start_token>{graph information} <kg_end_token>
 # Task: Based on the assembly knowledge graph information above, answer the question
 # Question {question}
Here, *graph information* refers to the graph embedding E_G. The other parts of the instruction are encoded by the LLM's embedding layer and concatenated with E_G.

B. Training details

For the Graph Encoder, we employed CompGCN [32], with the hyperparameters set as follows: score_func = distmult, opn = corr, init_dim = 128, embed_dim = 768, and gcn_layer = 2. In the Alignment Tuning stage, we used LLaMA3-8b-4bit [23] as the base model. The implementation is carried out using the PyTorch framework with the AdamW optimizer, an initial learning rate of 1×10^{-4}, and cosine decay scheduling. The projector layer consists of two 2-layer Feed-Forward Networks (FFNs). The model is trained for a maximum of 100 epochs with a batch size of 1, across three NVIDIA RTX 3090 GPUs. The loss function parameters are set as $\lambda_{\text{penalty}} = 1 \times 10^{-5}$, top_k $= 10$, and max_ratio $= 0.1$.

C. Evaluation

Metrics. For single-hop QA, evaluation is based on answer accuracy (Acc). For multi-hop QA, we use normalized LCS (nLCS) to measure the consistency of sequence order and Weighted Jaccard Similarity (wJaccard) to assess content consistency. For the grasping experiments in stacked scenarios, we adopt two metrics: average step and optimal planning rate (OPR), defined as

$$\text{OPR} = \frac{\text{Number of Optimal Plans}}{\text{Number of Actual Operations}} \tag{11}$$

Comparative Experiments. As previously mentioned, existing methods often represent domain-specific knowledge in the form of natural language text. To ensure the credibility of our comparative experiments, we compare our approach with existing methods that perform QA tasks based on external knowledge. These methods can be categorized into four types:

- **Document:** Using plain textual documents directly as prompts for LLMs [38].
- **Document + Fine-tuning:** Fine-tuning LLMs on the basis of these textual documents [39], [40].
- **Text Triple:** Converting graph-structured data into natural language text as input to LLMs [36].
- **Text Triple + Fine-tuning:** Fine-tuning LLMs on graph-derived textual representations [37].

In these experiments, we consistently adopt LoRA [43] as the method for fine-tuning.

As shown in Table II, compared to existing methods, our approach demonstrates clear advantages in both efficiency and accuracy. This can be intuitively attributed to the fact that, by encoding the graph knowledge via GCN into embeddings, the required context length is reduced by 28 times, and inference is accelerated by 3 times with only 0.48s, which greatly reduces computational resource requirements, facilitates edge deployment, and better suits the needs of real-world robotic assembly assistance. Moreover, by training only the projector layer, our method achieves an accuracy of 82.1% on single-hop QA, 66.7% nLCS, and 65.3% wJaccard on multi-hop QA. It surpasses existing methods with 6.4% higher accuracy on single-hop QA, and 21% higher nLCS and 21.9% higher wJaccard on multi-hop QA.

In addition, in the comparative experiments, we observed that assembly process documents contain a large amount of distracting information, such as symbols and referential expressions, which results in the lowest quality of injected knowledge when the LLM is not fine-tuned. In contrast, although triples, like graphs, are structured data and conducive to querying, they suffer from significant entity redundancy. Specifically, the same entity may be connected by multiple edges, causing it to be repeatedly represented in the triples, which increases the context length and consequently prolongs the inference time.

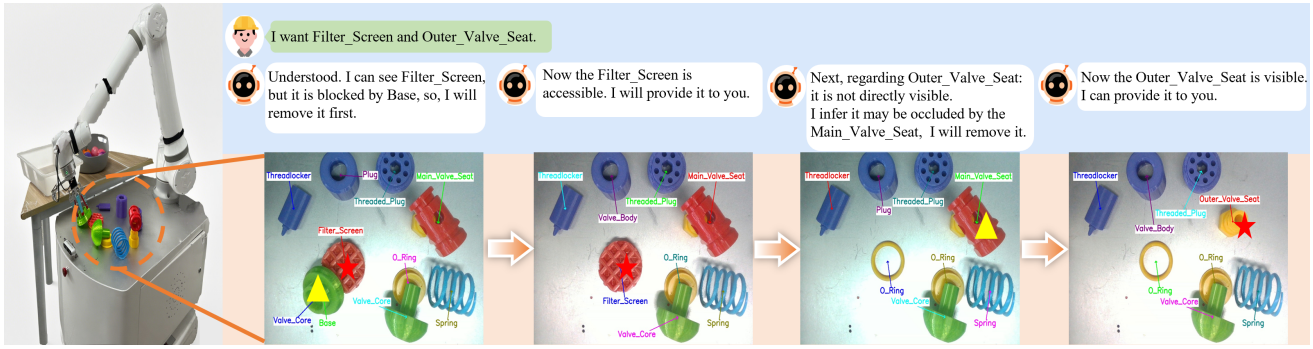

Fig. 3. In real-world experiments, AssemMate dynamically determines the optimal object to grasp based on the current scene, where the yellow triangle indicates the current considered object and the red star marks the target object.

TABLE IV
TESTING GENERALIZATION ABILITY ON RANDOM GRAPHS.

Dataset	Single-hop QA	Multi-hop QA	
	Acc. (%) ↑	nLCS (%) ↑	wJaccard (%) ↑
Testing Graph	82.1	66.7	65.3
Random Graph	75.9	56.9	55.6

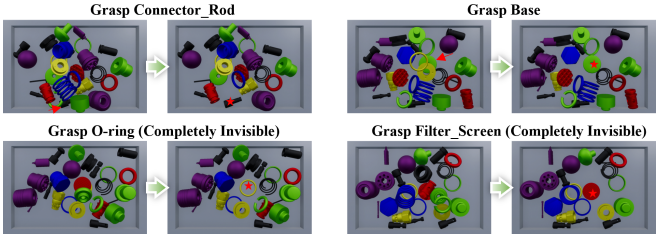

Fig. 4. The heavily stacked scenarios within the simulation environment, the target object is marked with a red star.

Ablation Study. As mentioned earlier, our graph knowledge E_G consists of two components: $E_{\text{structural}}$ and E_{semantic}. The ablation of the GCN module essentially removes the $E_{\text{structural}}$ information. As shown in Table III, this leads to a performance drop in both single-hop and multi-hop QA tasks, particularly in multi-hop QA, where the decrease is nearly 10%, compared to around 3% in single-hop QA. This aligns with the fact that multi-hop reasoning relies more heavily on structured information than single-hop reasoning, highlighting the necessity of the GCN module. In addition, as indicated in Table III, removing the relation embedding results in catastrophic performance degradation, underscoring the critical importance of encoding specific relations in robotic applications.

Generalization to Random Graphs. To validate the generalization of our knowledge injection method, we construct random graphs at both the step and operation levels for testing. And the results in Table IV demonstrate that our method performs excellently on these tests, with only a minor decrease of 8.5% in single-hop QA accuracy, and 10.8% and 14.3% in nLCS and wJaccard for multi-hop QA, respectively, compared to the testing graph. This indicates that our approach conducts reasoning over the injected graph embedding rather than relying solely on the memorization capability of LLM, suggesting promising potential for extending our method to broader domains.

D. Simulation and Real-World Experiment

System Setup. The VEGE pipeline consists of a vision-enhanced reasoning module, an object perception module, and a grasp synthesis module. Vision-enhanced reasoning is performed using the GPT-5 API [44]. The model takes the task description together with the enhanced-scene image generated by the vision enhancement method described in Section IV-B as input, and outputs an optimal grasping plan. Given the selected target from the plan, object-level perception is carried out using YOLOv11 [45] for 2D object detection, followed by SAM2 [46] for precise instance segmentation. The resulting segmentation mask, together with depth information, is provided as input to GraspNet-1Billion [47] to generate candidate grasp poses on the target object. The grasp with the highest predicted success score is selected and executed by the robot. To improve robustness under cluttered conditions, we collected 2,000 stacked-scene images in a simulation environment to fine-tune the YOLOv11 detector for 2D localization.

Our simulation environment is built on the Unity engine, using a UR3 robotic arm equipped with a Robotiq 2f-85 two-finger gripper. For real-world experiments, we employ an ELITE ROBOTS EC612 Collaborative Robot with a WHEELTEC four-finger flexible electric gripper and an Intel RealSense D435i depth camera.

We respectively evaluate grasping performance under the answers of single-hop and multi-hop QA as upstream planning. For each QA, 10 planning cases are tested, executed 10 times in both simulation and real-world setups. The stacked scenes consist of 22 randomly stacked objects, which are kept consistent between simulation and real-world experiments.

In the real-world grasping experiments, we 3D-printed 22 components and tools based on the 3D model of the pressure-reducing valve to serve as graspable objects.

Results and Analyze. The real-world grasping experiment is shown in Fig. 3. AssemMate dynamically determines the

TABLE V

EVALUATION OF GRASPING PERFORMANCE IN STACKED SCENARIOS.

	Single-hop		Multi-hop	
	Simulation	Real	Simulation	Real
OPR (%) ↑	71.2	64.3	64.0	52.4
Average Step ↓	3.1	3.9	6.2	7.3

optimal object to grasp based on the current stacked scene, enabling it to sense scenarios where objects are stacked or even fully occluded. This shows that AssemMate can remove obstructing objects and grasp the user-desired one in a stacked industrial environment, assisting in assembly. The results of the grasping experiments are presented in Table V. Under upstream planning with single-hop QA, the optimal planning rate for grasping in simulation reaches 71.2%, while for multi-hop QA planning of object sequences, it reaches 64.0%. This demonstrates the effectiveness and efficiency of our approach in sensing stacked scenes and executing grasping. In real-world experiments, due to object collisions during the grasping process and lighting effects during detection, the performance is lower than in simulation by 6.9% for single-hop QA and 11.6% for multi-hop QA, showing good sim-to-real transfer capability. Complex stacked scenarios in simulation are illustrated in Fig.4, where AssemMate efficiently and accurately removes occluding objects surrounding the target.

VI. CONCLUSIONS

In this paper, we introduce a reliable mate, AssemMate, which, for the first time in the field of robotic assembly assistance, leverages graph—the concise and precise data structure—as the means to represent assembly knowledge and efficiently inject it into LLM. This enables LLM to acquire knowledge absent during pretraining and respond to human queries to support interactive QA for correct assembly guidance. Moreover, we employ vision enhancement techniques to fully exploit the capabilities of MLLMs, allowing execution of grasping in stacked scenarios, thereby improving assembly efficiency. Experimental results demonstrate that our method outperforms existing approaches on the same tasks in terms of accuracy, efficiency, and resource consumption, and can be easily extended to private graphs and specific tasks across various domains.

VII. ACKNOWLEDGEMENT

This work is supported by National Natural Science Foundation of China (Grant No. 92467204).

REFERENCES

[1] M. Jiang, Q. Wang, H. Ai, Z. Dong, Y. Hu, A. Liang, Y. Wang, R. Li, Q. Liu, M. Chen *et al.*, "Prompt2Act: Mapping Prompts into Sequence of robotic Actions with Large Foundation Models," *Information Fusion*, p. 103923, 2025.

[2] J. Lim, S. Patel, A. Evans, J. Pimley, Y. Li, and I. Kovalenko, "Enhancing human-robot collaborative assembly in manufacturing systems using large language models," in *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2024, pp. 2581–2587.

[3] Y. Hua, K. Li, R. Wang, Y. Li, G. Wang, and Y. Yan, "Integration of dynamic knowledge and LLM for adaptive human-robot collaborative assembly solution generation," *Advanced Engineering Informatics*, vol. 68, p. 103613, 2025.

[4] Z. Ni, X. Deng, C. Tai, X. Zhu, Q. Xie, W. Huang, X. Wu, and L. Zeng, "Grid: Scene-graph-based instruction-driven robotic task planning," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 13 765–13 772.

[5] J. Huang, X. Zhang, Q. Mei, and J. Ma, "Can LLMs Effectively Leverage Graph Structural Information: When and Why," in *NeurIPS 2023 Workshop: New Frontiers in Graph Learning*, 2023. [Online]. Available: https://openreview.net/forum?id=jyfiPivRBH

[6] Y. Li, P. Wang, X. Zhu, A. Chen, H. Jiang, D. Cai, V. W. Chan, and J. Li, "Glbench: A comprehensive benchmark for graph with large language models," *Advances in Neural Information Processing Systems*, vol. 37, pp. 42 349–42 368, 2024.

[7] X. Ren, J. Tang, D. Yin, N. Chawla, and C. Huang, "A survey of large language models for graphs," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 6616–6626.

[8] X. Dai, H. Qu, Y. Shen, B. Zhang, Q. Wen, W. Fan, D. Li, J. Tang, and C. Shan, "How Do Large Language Models Understand Graph Patterns? A Benchmark for Graph Pattern Comprehension," in *The Thirteenth International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=CkKEuLmRnr

[9] J. Tang, Y. Yang, W. Wei, L. Shi, L. Su, S. Cheng, D. Yin, and C. Huang, "Graphgpt: Graph instruction tuning for large language models," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024, pp. 491–500.

[10] R. Chen, T. Zhao, A. K. Jaiswal, N. Shah, and Z. Wang, "LLaGA: Large language and graph assistant," in *Proceedings of the 41st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, Eds., vol. 235. PMLR, 21–27 Jul 2024, pp. 7809–7823. [Online]. Available: https://proceedings.mlr.press/v235/chen24bh.html

[11] R. Ye, C. Zhang, R. Wang, S. Xu, and Y. Zhang, "Language is all a graph needs," *EACL*, 2024.

[12] J. Zhao, M. Qu, C. Li, H. Yan, Q. Liu, R. Li, X. Xie, and J. Tang, "Learning on large-scale text-attributed graphs via variational inference," *ICLR*, 2023.

[13] D. Wang, Y. Zuo, F. Li, and J. Wu, "Llms as zero-shot graph learners: Alignment of gnn representations with llm token embeddings," *Advances in Neural Information Processing Systems*, vol. 37, pp. 5950–5973, 2024.

[14] B. Jin, W. Zhang, Y. Zhang, Y. Meng, X. Zhang, Q. Zhu, and J. Han, "Patton: Language model pretraining on text-rich networks," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.

[15] Z. Xu, C. Zhang, S. Hu, Z. Han, P. Feng, and L. Zeng, "Reconfigurable flexible assembly model and implementation for cross-category products," *Journal of Manufacturing Systems*, vol. 77, pp. 154–169, 2024.

[16] L. Xiao, L. Zeng, Z. Xu, and X. Liu, "Assembly language design and development for reconfigurable flexible assembly line," *Robotics and Computer-Integrated Manufacturing*, vol. 80, p. 102467, 2023.

[17] J. Zhao, L. Zhuo, Y. Shen, M. Qu, K. Liu, M. M. Bronstein, Z. Zhu, and J. Tang, "Graphtext: Graph reasoning in text space," in *Adaptive Foundation Models: Evolving AI for Personalized and Efficient Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=6cRM0OT03F

[18] Z. Chai, T. Zhang, L. Wu, K. Han, X. Hu, X. Huang, and Y. Yang, "GraphLLM: Boosting graph reasoning ability of large language model," 2024. [Online]. Available: https://openreview.net/forum?id=PIl69UIAWL

[19] L. Zeng, W. J. Lv, X. Y. Zhang, and Y. J. Liu, "ParametricNet: 6dof pose estimation network for parametric shapes in stacked scenarios," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 772–778.

[20] Y. H. Xie, W. J. Lv, X. Y. Zhang, Y. H. Chen, and L. Zeng, "ParametricNet++: A 6dof pose estimation network with sparse keypoint recovery for parametric shapes in stacked scenarios," in *2024*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2024, pp. 7181–7188.

[21] D.-T. Huang, E.-T. Lin, L. Chen, L.-F. Liu, and L. Zeng, "SD-Net: Symmetric-aware keypoint prediction and domain adaptation for 6d pose estimation in bin-picking scenarios," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 2747–2754.

[22] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[23] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[24] S. Jin, J. Xu, Y. Lei, and L. Zhang, "Reasoning grasping via multimodal large language model," *Conference on Robot Learning*, 2024.

[25] R. Mirjalili, M. Krawez, S. Silenzi, Y. Blei, and W. Burgard, "Lan-grasp: An effective approach to semantic object grasping using large language models," in *First workshop on vision-Language Models for navigation and manipulation at ICRA 2024*, 2024.

[26] C. Tang, D. Huang, W. Ge, W. Liu, and H. Zhang, "Graspgpt: Leveraging semantic knowledge from a large language model for task-oriented grasping," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7551–7558, 2023.

[27] Y. Qian, X. Zhu, O. Biza, S. Jiang, L. Zhao, H. Huang, Y. Qi, and R. Platt, "ThinkGrasp: A Vision-Language System for Strategic Part Grasping in Clutter," 2024.

[28] K. Xu, S. Zhao, Z. Zhou, Z. Li, H. Pi, Y. Zhu, Y. Wang, and R. Xiong, "A joint modeling of vision-language-action for target-oriented grasping in clutter," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 11 597–11 604.

[29] M. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn, "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[30] 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 *Conference on Robot Learning*. PMLR, 2023, pp. 2165–2183.

[31] Z. Xu, C. Zhang, S. Hou, Z. Han, L. Zeng, and P. Feng, "Assembly task planning framework based on knowledge graph," *Journal of Intelligent Manufacturing*, pp. 1–24, 2025.

[32] S. Vashishth, S. Sanyal, V. Nitin, and P. Talukdar, "Composition-based multi-relational graph convolutional networks," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=BylA_C4tPr

[33] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.

[34] B. Yang, W. tau Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *International Conference on Learning Representations*, 2014. [Online]. Available: https://api.semanticscholar.org/CorpusID:2768038

[35] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.

[36] J. Kim, Y. Kwon, Y. Jo, and E. Choi, "KG-GPT: A general framework for reasoning on knowledge graphs using large language models," in *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. [Online]. Available: https://openreview.net/forum?id=QAZ2QV8SqN

[37] J. Feng, Y. Chen, and M. Zhang, "GRIP: In-Parameter Graph Reasoning through Fine-Tuning Large Language Models," in *Second Workshop on Test-Time Adaptation: Putting Updates to the Test! at ICML 2025*.

[38] Y. Hua, K. Li, R. Wang, Y. Li, G. Wang, and Y. Yan, "Integration of dynamic knowledge and llm for adaptive human-robot collaborative assembly solution generation," *Advanced Engineering Informatics*, vol. 68, p. 103613, 2025.

[39] D. Su, Y. Xu, G. I. Winata, P. Xu, H. Kim, Z. Liu, and P. Fung, "Generalizing question answering system with pre-trained language model fine-tuning," in *Proceedings of the 2nd workshop on machine reading for question answering*, 2019, pp. 203–211.

[40] M. Basem, I. Oshallah, B. Hikal, A. Hamdi, and A. Mohamed, "Optimized quran passage retrieval using an expanded qa dataset and fine-tuned language models," in *The International Conference of Advanced Computing and Informatics*. Springer, 2024, pp. 244–254.

[41] S. Xuan, Q. Guo, M. Yang, and S. Zhang, "Pink: Unveiling the power of referential comprehension for multi-modal llms," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 13 838–13 848.

[42] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[43] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models." *ICLR*, vol. 1, no. 2, p. 3, 2022.

[44] OpenAI, "Introducing gpt-5," https://openai.com/index/introducing-gpt-5/, 2025, accessed: 2025-09-05.

[45] R. Khanam and M. Hussain, "Yolov11: An overview of the key architectural enhancements," *arXiv preprint arXiv:2410.17725*, 2024.

[46] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer, "Sam 2: Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024. [Online]. Available: https://arxiv.org/abs/2408.00714

[47] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "Graspnet-1billion: A large-scale benchmark for general object grasping," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 444–11 453.