

A Multimodal RAG-based Maintenance Chatbot for Robotic Arm Manuals

Yen-Hua Lu and Ching-Hung Lee, *Senior Member, IEEE*

Abstract—With the rapid advancement of large language models (LLMs), intelligent chatbots are increasingly being adopted for maintenance documentation, fault diagnosis, and personnel training. This study introduces a multimodal Retrieval-Augmented Generation (RAG) chatbot designed to provide accurate and natural-language support for robotic arm maintenance tasks. The system separates textual and visual content from maintenance manuals and processes them through two complementary pipelines. Caption RAG employs a vision-language model (VLM) to generate contextual captions for images, improving the retrieval of relevant documents. VLM RAG then integrates retrieved text and associated images, using GPT-4o to deliver more precise and context-aware answers. To address industrial data privacy concerns, the system supports local deployment using open-source LLaMA and Taiwan's TAIDE LLM models. The evaluation dataset was curated and validated by senior experts from an industrial robotic arm manufacturer, ensuring strong domain alignment. Experimental results show high accuracy—96% with GPT-4o, 92% with LLaMA 8B, and 74.67% with TAIDE 8B. Incorporating visual context via VLM RAG further improved performance to 96.67%, highlighting the benefit of multimodal integration. In summary, the proposed chatbot enhances maintenance efficiency and fault resolution while preserving data privacy, making it a practical solution for real-world industrial deployment.

Index Terms—Multimodal, RAG, Large Language Model, Vision-Language Model, Chatbot, Maintenance Manuals, Robotic Arm

I. INTRODUCTION

WITH the rapid advancement of large language models (LLMs) [1], companies have increasingly adopted LLMs to manage complex internal documents [2], such as technical manuals and operational procedures. To adapt LLMs for enterprise-specific knowledge, two main approaches have emerged: **fine-tuning** [3], [4] and **retrieval-augmented generation (RAG)** [5], [6]. Fine-tuning customizing the model parameters using domain-specific data, allowing the model to internalize domain knowledge. In contrast, RAG integrates external knowledge bases into the inference process, allowing LLMs to dynamically access and retrieve relevant documents without modifying the model parameters.

In current research, the development of chatbots using RAG systems is more commonly seen in domains such as law [7], [8]

and healthcare [9], while its application to robotic arm maintenance manuals remains relatively limited.

A unique challenge in robotic arm maintenance manuals is that the same query may produce different solutions depending on the product model. Fine-tuning in such cases may risk confusing the LLM, leading to ambiguous or incorrect responses. In comparison, RAG provides strong scalability and flexibility, as new or model-specific documentation can be directly integrated into the retrieval database without retraining. This property is particularly well-suited for managing robotic arm maintenance manuals, where manuals for different product models can be stored separately. In this way, the system avoids the confusion that may arise from fine-tuning an LLM on multiple models, while also reducing the likelihood of hallucinations [10]. Therefore, this study adopts the RAG approach to ensure accurate, scalable, and maintainable access to enterprise knowledge.

RAG typically operates in two stages: a data pre-processing stage and an inference stage. This architecture enables knowledge extension without fine-tuning and supports excellent scalability. The detailed process is illustrated in Fig. 1. During the data pre-processing stage, the textual content of maintenance manuals is split into multiple independent chunks. Each chunk is transformed into a high-dimensional vector using an embedding model to capture its semantic features. These vectors are stored in a vector database for subsequent retrieval.

For inference stage, when a user submits a query, the system encodes the query into a vector and computes similarity scores with all stored passage vectors. The Top- K most relevant passages are then selected, converted back into text, and provided to the LLM along with the query to generate a response. This architecture enables knowledge extension without the need for model fine-tuning and offers excellent scalability.

However, traditional RAG frameworks are limited to textual data and lack the ability to retrieve and interpret visual content. In maintenance manuals, visual information, such as assembly diagrams, component layouts, and procedural illustrations, often plays a critical role in helping engineers. Without image processing capabilities, RAG-based systems may produce incomplete or inaccurate answers, limiting their utility in multimodal documentation.

Previous studies have explored various approaches to enhance RAG systems for handling visual content. For example, Yu *et al.* proposed ViRAG [11], a method that converts full-page document images into embeddings for retrieval and passes them into a vision-language model (VLM) to generate responses. Although this simplifies the document parsing pipeline, it heavily depends on the OCR and layout understanding capabilities of VLM, which may lead to hallucinations and misinterpretations, especially when the

Yen-Hua Lu is with the Institute of Electrical and Control Engineering, National Yang Ming Chiao Tung University, 1001 University Road, Hsinchu, 30010, Taiwan.

Ching-Hung Lee is with the Institute of Electrical and Control Engineering, National Yang Ming Chiao Tung University, 1001 University Road, Hsinchu, 30010, Taiwan, also with the Department of Electrical Engineering, Chung Yuan Christian University, Taiwan (e-mail: chl@nycu.edu.tw).

Manuscript received 11 September 2025; accepted 11 October 2025; Date of publication 12 November 2025.

image quality is low or the document structure is complex.

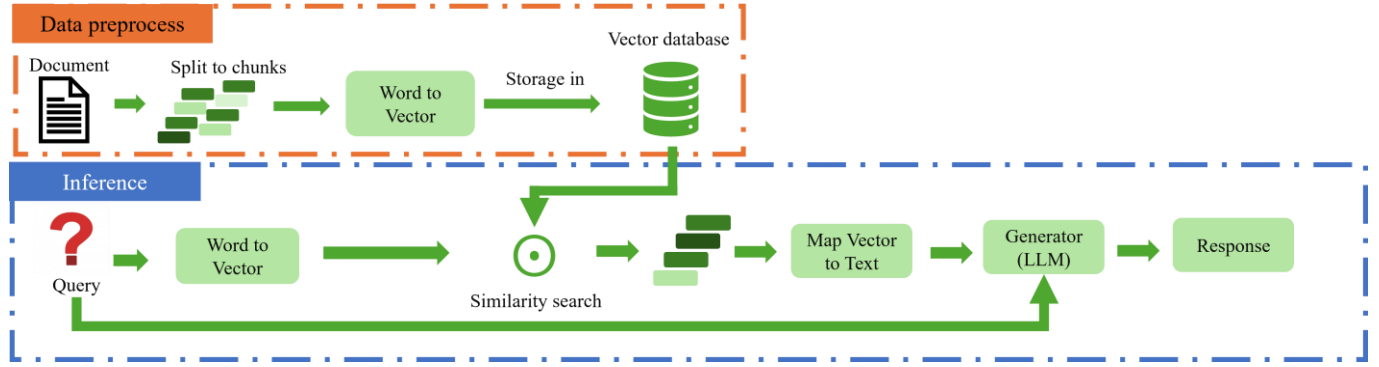


Fig 1. Architecture of RAG [5]

Wang *et al.* introduced ViDoRAG [12], a hybrid multimodal retrieval framework that constructs separate embedding spaces for text and images, and dynamically determines the number of top k retrieved segments using a Gaussian Mixture Model (GMM). Additionally, they employ a three-agent architecture—Seeker, Inspector, and Answer Agent—to simulate human reasoning and perform iterative refinement. Nonetheless, the framework still relies on OCR-dependent VLMs, which remain susceptible to conversion errors and inconsistencies in semantic interpretation.

Zhang *et al.* proposed OHRBench [13], a benchmark that reveals how semantic and formatting noise introduced by OCR substantially deteriorates both retrieval accuracy and answer generation. Their findings show that even state-of-the-art VLMs struggle to overcome hallucination problems, confirming that OCR-based interpretation continues to be a key bottleneck in multimodal RAG systems.

To address these limitations, we propose two multimodal RAG frameworks: **Caption RAG** and **VLM RAG**. Both approaches are based on document-level text-image separation. In Caption RAG, the image context and the image itself are input into a VLM to generate a descriptive caption, which is then inserted back into the document for retrieval. In VLM RAG, the retrieved text segments that reference images are used to locate the corresponding images from a repository, and both text and images are fed into a VLM for answer generation. Our experimental results demonstrate that both approaches significantly improve visual-semantic comprehension in RAG systems.

Moreover, we deploy the Caption RAG framework in a local environment using the LLaMA [14] and TAIDE [15] models. The results show that our system maintains strong accuracy even without relying on cloud-based computational resources.

The major contributions of our study are as follows.

- We design two approaches — **Caption RAG** and **VLM RAG** — to enable effective multimodal RAG retrieval in mixed text-image documents. These methods are specifically validated in robotic arm maintenance manuals, which contain extensive image information. Empirical evaluation demonstrates their superior accuracy and robustness in this real-world scenario.

- The evaluation data set that we used was provided by a senior engineer from an industrial robotic arm manufacturer. The data set was carefully designed to reflect real-world production line issues and corresponding solutions, ensuring both domain expertise and practical applicability.

- We successfully implemented **Caption RAG** in a local setup, showing strong system performance and inference efficiency without relying on a cloud-based model. This confirms the practical applicability and flexibility of the framework's deployment under limited computational resources.

- **Caption RAG** achieves a balance between accuracy and cost by converting images into textual descriptions, making it a good fit for resource-constrained environments. In contrast, **VLM RAG** directly leverages multimodal inputs and delivers the highest accuracy.

II. THE PROPOSED MULTIMODAL RAG-BASED CHATBOT SYSTEM

To effectively solve the conversion errors and hallucination issues caused by OCR processes in VLM, this study proposes two RAG-based retrieval methods for handling mixed text-image documents. Caption RAG and VLM RAG. The key innovation of these methods is based on the application of text-image separation, which processes text and images independently. This approach significantly reduces conversion errors and hallucinated content that may arise during OCR conversion. The complete architecture is shown in Fig. 2.

A. Text-image separation

This paper applies the pymupdf [16] tool to implement the text-image separation technique. The tool extracts all copyable text blocks from the document and independently extracts all embedded images, as shown in Fig. 3.

During the image separation process, each image is assigned a unique identifier (e.g., image_1, image_2, etc.), and a corresponding image tag is inserted into the original text content to preserve the positional relationship between images and text within the document.

Our approach leverages text-image separation to effectively overcome conversion errors and hallucinations typically induced by VLM during the OCR process.

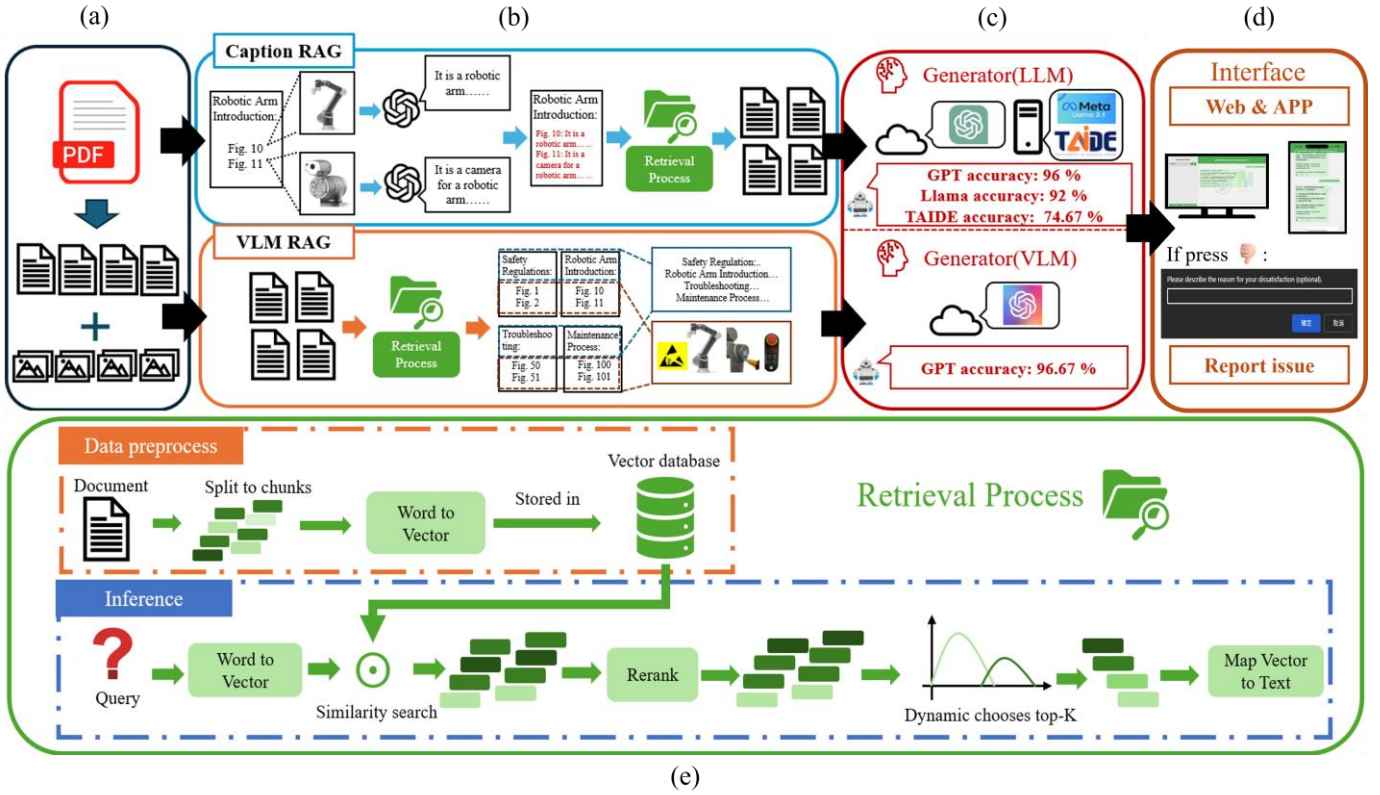


Fig 2. Architecture of our system. (a) Text-image separation; (b) The proposed methods: Caption RAG and VLM RAG; (c) Generator: Usage of different LLMs, VLMs, and experimental results; (d) Interface with feedback system (Web and APP versions); (e) Retrieval process.

In subsequent processing stages, this image tag information enables quick localization of the corresponding images, serving as the foundation for Caption RAG generation or VLM RAG multimodal processing.

B. The Proposed Methods

1) *Caption RAG*: In this method, based on the results of text-image separation, we extract image tag information (e.g., image_1) and retrieve 100 contexts of text both before and after the tag as the prompt. This prompt, along with the separated image (e.g., image_1), is input into the VLM (GPT-4o) [17], which is then asked to generate a description of the image, as shown in Fig. 4.

Since the prompt includes the contextual text surrounding the image, GPT-4o can understand how the image is described within the document. The generated caption is then inserted back into the original position of the image tag. Using this approach, all images in the maintenance manuals are progressively converted into the corresponding textual descriptions, ultimately constructing a fully text-based version of the document. Following the above procedure, we proceed with RAG-based retrieval to identify the correct answer to the given query.

2) *VLM RAG*: This method further extends the processing results of the text-image separation step. The RAG algorithm is applied directly to the textual portion of the document, including image tags. During the retrieval process, the system identifies text fragments containing image tags and uses these

references to locate the corresponding images from a preprocessed image repository.

Once the relevant text and associated images are retrieved, they are jointly entered into a VLM -GPT-4o is adopted in this study. This model can simultaneously process both text and several images within a single context, integrating semantic text with visual information from the images. As a result, it generates contextually enriched and semantically comprehensive responses.

This approach significantly improves the model's ability to interpret complex technical documents, making it particularly suitable for applications such as maintenance manuals, where visual elements, such as assembly diagrams, component schematics, or procedural illustrations, often carry critical additional information. These visuals effectively compensate for the limitations in textual descriptions. The complete system architecture of VLM RAG is illustrated in Fig. 5.

C. Generator

In terms of generator selection strategy, we differentiate and optimize the input format to LLMs based on the distinct data processing characteristics of the Caption RAG and VLM RAG methods.

In the Caption RAG method, we first perform pre-processing on the images. The system uses VLMs to generate semantic descriptions of each image, effectively converting the images into the corresponding textual content.

TABLE I
MODELS USED IN THE DIFFERENT RAG METHODS

Method	Image Description	Generator
Caption RAG	GPT-4o (cloud)	GPT-4o (Cloud) TAIDE 8B (Local) LLaMA3.1 8B (Local)
VLM RAG	Not required	GPT-4o (Cloud)

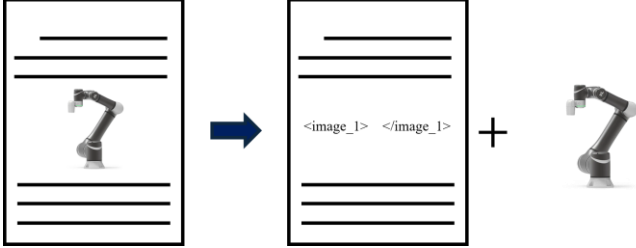


Fig 3. Text-image separation

In this method, we use the GPT-4o VLM model to generate semantic descriptions of the images. As a result, all the images in the document are transformed into text descriptions. Therefore, the generator in the Caption RAG pipeline receives only textual input, without any image content. We select three models with strong generative capabilities as generators: GPT-4o, LLaMA 8B, and TAIDE 8B. Among them, GPT-4o utilizes OpenAI's cloud-based inference resources, while TAIDE 8B and LLaMA3.1 8B can be executed locally, offering flexibility in deployment and cost advantages.

In contrast, the VLM RAG method does not pre-process the images in advance. During the RAG retrieval process, the system retrieves not only relevant text fragments but also their corresponding image tags. These retrieved text-image pairs are then input jointly into the generator. Consequently, the generator for the VLM RAG method must support image input and possess multimodal semantic understanding capabilities. To meet this requirement, we adopt GPT-4o as the generator for the VLM RAG approach, leveraging its ability to process both images and text simultaneously for integrated generation and semantic fusion.

However, current publicly available VLMs such as LLaMA Vision [18], Qwen-VL [19] do not support batch input of multiple images. Therefore, in this project, GPT-4o is exclusively used as the cloud-based VLM for system validation.

Table I summarizes the methods utilized each stage of the proposed approaches.

D.Retrieval Process

The retrieval process proposed in this study is based on the traditional RAG framework. We employ LangChain's [20]

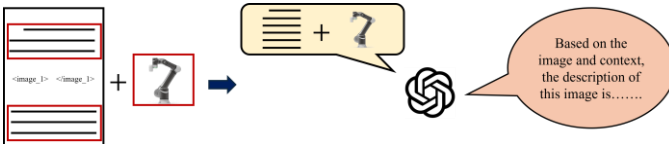


Fig 4. Caption RAG

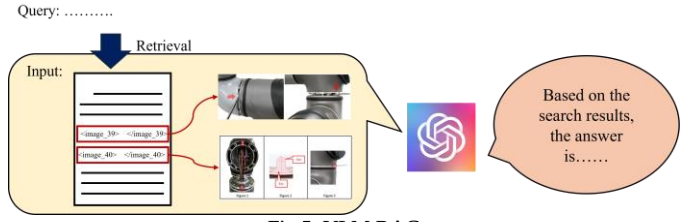


Fig 5. VLM RAG

RecursiveCharacterTextSplitter function to segment the document, where the text is divided into chunks based on a predefined chunk size and overlap. Each chunk is then converted into a vector representation using OpenAI's embedding model [21] and stored in a FAISS [22] vector database.

When a user query is input, the query is encoded into a vector using the same embedding model. The cosine similarity search is then performed in the FAISS database to retrieve the top-K most relevant chunks. Next, a re-ranking model (ms-marco-MiniLM-L-6-v2 [23]) reorders these candidates so that the most semantically relevant passages are prioritized. Re-ranking is based on a cross-encoder architecture, where the query and document are concatenated and jointly fed into the model. Through the attention mechanism, the model captures semantic interactions between the two, ultimately outputting a relevance score. This approach enables a more fine-grained assessment of the matching degree between candidate passages and the query, thereby improving ranking accuracy [24]. Finally, inspired by the ViDoRAG [12] framework, we further incorporate the dynamic paragraph selection method to enhance the quality and semantic relevance of the information retrieved by the retriever.

In traditional RAG systems, the retriever typically selects the top K most similar passages (top-K chunks). However, this method lacks flexibility and may mix in irrelevant information. To solve this issue, we adopt the GMM mechanism proposed by ViDoRAG to dynamically determine the number of passages to be included in each retrieval.

The GMM takes the similarity scores S_i produced by the retriever and assumes that S_i follows a mixture of two Gaussian distributions—one for high similarity and one for low similarity. The overall mixture distribution is shown in (1). Here, $\mathcal{N}(s_i|\mu_k, \sigma_k^2)$ denotes the k-th Gaussian component ($k = 0$ for low similarity, $k = 1$ for high similarity); π_k is the mixture weight of component k , and μ_k, σ_k are the mean and standard deviation of that component.

$$P(s_i) = \pi_0 \cdot \mathcal{N}(s_i|\mu_0, \sigma_0^2) + \pi_1 \cdot \mathcal{N}(s_i|\mu_1, \sigma_1^2) \quad (1)$$

Next, we introduce a latent variable $z_i \in \{0, 1\}$ that indicates whether passage i belongs to the low-similarity component ($z_i = 0$) or the high-similarity component ($z_i = 1$). The Expectation-Maximization (EM) algorithm is then used to compute the posterior probability that each passage S_i belongs to the high-similarity component, as shown in (2). Based on a

Algorithm 1 Caption RAG Process

```

1: Constants: Embedding Model  $M_{emb}$ , Generative LLM  $M_{llm}$ ,
   VLM Model  $M_{vlm}$ 
2: Input: Document:  $\mathcal{D}$ , User Query  $q$ 
3: Output: Final Response  $r$ 
4: Data Preprocess
5:  $(T, I) \leftarrow \text{SeparateDocument}(\mathcal{D})$   $\triangleright$  Split document into text  $T$  and
   images  $I$ 
6: for each image  $i \in I$  do
7:    $c_i \leftarrow M_{vlm}(i)$   $\triangleright$  Generate caption  $c_i$  for image  $i$ 
8:    $T \leftarrow \text{InsertAtOrigPos}(T, c_i, i)$ 
9: end for
10:  $\mathcal{D}_{text} \leftarrow T$   $\triangleright$  Obtain augmented document with text and caption
11: Data Ingestion
12:  $\mathcal{C} \leftarrow \text{ChunkText}(\mathcal{D}_{text})$   $\triangleright$  Split document into chunks
13:  $E_{chunks} \leftarrow M_{emb}(\mathcal{C})$   $\triangleright$  Embed each chunk
14:  $DB_{vector} \leftarrow \text{Store}(E_{chunks})$ 
15: Query Processing
16: While true do  $\triangleright$  Continuous query input from user
17:    $q \leftarrow \text{UserInput}()$ 
18:    $E_{query} \leftarrow M_{emb}(q)$ 
19:    $\mathcal{C}^* \leftarrow \text{VectorSearch}(E_{query}, DB_{vector})$ 
20:    $\mathcal{C}_{rank} \leftarrow \text{Rerank}(q, \mathcal{C}^*)$ 
21:    $\mathcal{C}_{gmm} \leftarrow \text{GMM}(\mathcal{C}_{rank})$ 
22:    $r \leftarrow M_{llm}(q, \mathcal{C}_{gmm})$ 
23:   Display( $r$ )
24: end While

```

predefined posterior probability threshold (0.5 in this study), all passages classified as belonging to the high-similarity distribution (i.e., those satisfying $P(z_i = 1 | s_i) > 0.5$) are retained to enable dynamic paragraph selection.

By integrating the original RAG retrieval strategy with the GMM-based dynamic top-K selection mechanism, we can reduce the inclusion of irrelevant passages in the LLM input,

thereby mitigating the noise that may interfere with the model's ability to generate accurate answers.

$$P(z_i = 1 | s_i) = \frac{\pi_1 \cdot \mathcal{N}(s_i | \mu_1, \sigma_1^2)}{\pi_0 \cdot \mathcal{N}(s_i | \mu_0, \sigma_0^2) + \pi_1 \cdot \mathcal{N}(s_i | \mu_1, \sigma_1^2)} \quad (2)$$

E. Methodological Framework

In this section, we present the overall methodological framework that integrates the components described in the previous subsections. Specifically, two algorithmic variants are summarized to illustrate the complete pipeline of our approach:

the *Caption RAG Process* (Algorithm 1) and the *VLM RAG Process* (Algorithm2).

The Caption RAG pipeline uses the vision language model (M_{vlm}) to generate textual descriptions for images, which are

Algorithm 2 VLM RAG Process

```

1: Constants: Embedding Model  $M_{emb}$ , Generative VLM  $M_{vlm}$ 
2: Input: Document:  $\mathcal{D}$ , User Query  $q$ 
3: Output: Final Response  $r$ 
4: Data Preprocess
5:  $(T, I) \leftarrow \text{SeparateDocument}(\mathcal{D})$   $\triangleright$  Split document into text  $T$  and
   images  $I$ 
6: Data Ingestion
7:  $\mathcal{C} \leftarrow \text{ChunkText}(\mathcal{D}_{text})$   $\triangleright$  Split document into chunks
8:  $E_{chunks} \leftarrow M_{emb}(\mathcal{C})$   $\triangleright$  Embed each chunk
9:  $DB_{vector} \leftarrow \text{Store}(E_{chunks})$ 
10: Query Processing
11: While true do  $\triangleright$  Continuous query input from user
12:    $q \leftarrow \text{UserInput}()$ 
13:    $E_{query} \leftarrow M_{emb}(q)$ 
14:    $\mathcal{C}^* \leftarrow \text{VectorSearch}(E_{query}, DB_{vector})$ 
15:    $\mathcal{C}_{rank} \leftarrow \text{Rerank}(q, \mathcal{C}^*)$ 
16:    $\mathcal{C}_{gmm} \leftarrow \text{GMM}(\mathcal{C}_{rank})$ 
17:    $\text{Tags} \leftarrow \text{FindImageTags}(\mathcal{C}_{gmm})$   $\triangleright$  Check retrieved chunks for
   figure tags
18:    $I_{retrieved} \leftarrow \emptyset$ 
19:   for each tag  $t \in \text{Tags}$  do
20:      $i \leftarrow \text{GetImageByTag}(I, t)$ 
21:      $I_{retrieved} \leftarrow I_{retrieved} \cup \{i\}$   $\triangleright$  Add the retrieved image  $i$  into
   the set  $I_{retrieved}$ 
22:   end for
23:    $r \leftarrow M_{vlm}(q, \mathcal{C}_{gmm}, I_{retrieved})$   $\triangleright$  Answer with both and
   images
24:   Display( $r$ )
25: end While

```

then inserted into the document to form an augmented text-only representation. This enables the downstream retrieval and generation to be carried out purely on textual embeddings, with the final response produced by the generative language model (M_{llm}).

In contrast, the VLM RAG pipeline preserves the original multimodal nature of the document. Text chunks are directly embedded and stored in the vector database, and during query processing, the retrieved results are examined for figure references. If such figure tags are detected, the corresponding images are extracted from the image set I . The generative vision-language model (M_{vlm}) then produces the final response by conditioning on both the retrieved text chunks and the associated images.

These two algorithms show how our framework accommodates both caption-based augmentation and direct multimodal reasoning.

III. EXPERIMENTAL RESULTS AND ANALYSIS

A. Document and Evaluation Dataset

The document used in our study is a 94-page PDF

maintenance manual for a robotic arm, which contains numerous images. After applying our text–image separation process, a total of 171 images were extracted.

The evaluation dataset used in this study was provided by a senior engineer from a robotic arm manufacturer. It was carefully curated based on maintenance manuals, containing 30 representative real-world issues and their corresponding standard answers, all of which were meticulously designed to ensure reliability and domain relevance.

B. Evaluation Metrics

This study adopts two evaluation metrics: **BERT Score** [25] and **LLM Score** [26] to assess the semantic similarity and correctness of the answers generated by the system.

The BERT score metric is used primarily to quantify the semantic similarity between two sentences. In this study, we employed the BERT Score Recall (R_{BERT}) to measure the semantic consistency between the system-generated answer and the reference answer, serving as a key metric of generation quality.

As shown (3), x_i and $\hat{x}_j$ are the vectors transformed by the BERT model[27]. These vectors are normalized, and we apply greedy matching to maximize the matching similarity score.

$$R_{BERT} = \frac{1}{|X|} \sum_{x_i \in X} \max_{\hat{x}_j \in \hat{X}} x_i^\top \hat{x}_j \quad (3)$$

However, BERT Score is sensitive to sentence length, structure, and word choice. If there are significant differences in length or expression between the generated and reference answers, even semantically similar responses may result in low BERT Scores, thereby affecting evaluation accuracy.

To improve consistency and objectivity in the evaluation process, we used a **short answer mechanism** as an intermediate step for semantic alignment prior to BERT Score calculation. After generating a full answer, the system is instructed to produce a concise answer using the LLM. This design addresses the issue of score distortion caused by length discrepancies between generated answers and reference answers. Since our RAG system retrieves a large amount of relevant information during the retrieval stage, the LLM tends to include additional explanations to help users better understand and resolve the issue. Although such supplemental content is useful, it may lead to significant differences in length and structure compared to the reference answer, potentially lowering the BERT Score despite high semantic similarity.

Therefore, we evaluate BERT Score based only on the LLM-generated concise answer and the corresponding reference answer. This effectively eliminates the interference from verbose expressions and non-essential information, ensuring that BERT Score reflects the true semantic similarity between the system's response and the ground truth.

As for the **LLM Score**, we use an LLM to evaluate the accuracy of the generated answers. The study by Arjun Panickssery, Samuel R. Bowman, and Shi Feng [28] demonstrates that a language model, with self-recognition capability, may tend to give higher scores to the responses it

generated itself. To avoid potential bias caused by the GPT-4o recognizing its own generated response, we use an additional model-Gemini 2.0 Flash-Lite [29] as the evaluation tool. The prompt includes the question, the reference answer, and the generated answer. Gemini is instructed to judge whether the generated answer is correct, using the reference answer as the primary reference.

Formally, for each item we construct a triple

$$u_i = (q_i, a_i^{ref}, a_i^{gen}), \quad (4)$$

where q_i is the question, a_i^{ref} is the reference answer, and a_i^{gen} is our RAG system generated answer.

The LLM judge $M_{llm}(\cdot)$ returns a binary correctness label

$$s_i^{(t)} = M_{llm} \in \{0, 1\}, \quad t = 1, \dots, T, \quad (5)$$

where $s_i^{(t)}=1$ if the generated answer is judged correct at trial t , and 0 otherwise.

The per-trial accuracy is define as

$$Acc^{(t)} = \frac{1}{N} \sum_{i=1}^N s_i^{(t)}. \quad (6)$$

To reduce potential randomness and instability, we adopt a multi-evaluation strategy by repeating the judging process T times (with $T = 5$ in our experiments). The final LLM Score is then computed as the mean accuracy across trials:

$$LLM\text{-Score} = \frac{1}{T} \sum_{t=1}^T Acc^{(t)}. \quad (7)$$

This approach provides a quantitative measure of response correctness and allows for statistical analysis of the system's overall accuracy.

C. Impact of Prompt Language on Model Performance

Our study conducts a comparative analysis of different LLM and their respective performance under varying prompt languages. Using the Caption RAG approach, we evaluate the performance of TAIDE 8B, LLaMA 8B, and GPT-4o by testing each model with both Chinese and English prompts. The evaluation metric adopted is the LLM Score.

The experimental results, as shown in the Table II, TAIDE is trained by the National Science and Technology Council (NSTC) on a large-scale Traditional Chinese dataset, exhibits strong proficiency and alignment with Traditional Chinese prompts. In contrast, LLaMA 8B shows significantly stronger performance when prompted in English compared to Chinese, revealing a marked sensitivity to prompt language. We speculate that this is due to the relatively small model size of

LLaMA (8 billion parameters) and its lack of training on multilingual datasets. On the other hand, while GPT-4o also performs better with English prompts, its sensitivity to language differences is noticeably lower than that of LLaMA.

To maximize answer accuracy in subsequent experiments, we adopt specific prompt language: TAIDE will use Traditional Chinese prompts, while LLaMA and GPT-4o will be prompted in English.

TABLE II

COMPARISON OF LLM PERFORMANCE UNDER DIFFERENT LANGUAGES

	TAIDE 8B	LLaMA 8B	GPT 4o
Chinese prompt (%)	74.67	64.67	80
English prompt (%)	70	92	96

D. Result

Table III presents the experimental results of our proposed methods compared to the baseline Naive RAG architecture. Three system variants are evaluated: (1) Naive RAG, the original RAG system which only supports pure text retrieval without image processing capability; (2) Caption RAG, and (3) VLM RAG, both of which are proposed in this study. All three systems use GPT-4o as the generator to ensure consistency.

As shown in the results, when evaluating answer similarity using BERT Score, Caption RAG (4o) achieves the highest similarity score of 76.4%, followed by VLM RAG (4o) at 76%. In terms of answer correctness evaluated via LLM Score, VLM RAG (4o) achieves the best performance with an accuracy of 96.67%, while Caption RAG (4o) reaches 96%.

These results indicate that the proposed image-enhanced retrieval strategies can effectively improve response correctness. Compared to Naive RAG, both Caption RAG and VLM RAG show significant improvements in both accuracy and semantic similarity, demonstrating the practical effectiveness of our methods.

It is also worth noting that BERT Score tends to be affected by response length. Since VLM RAG is capable of leveraging both visual and textual input simultaneously, its generated responses are often more detailed. This may lead to lower BERT Scores despite high accuracy, resulting in a divergence between BERT and LLM Score metrics.

 TABLE III
 Compare with Naive RAG and our work

	Naive RAG (4o)	Caption RAG (4o)	VLM RAG (4o)
BERT Score (%)	75.27	76.4	76
LLM Score (%)	86	96	96.67

 TABLE IV
 Compare with the local model and the cloud model

	Caption RAG (TAIDE 8B)	Caption RAG (LLaMA 8B)	Caption RAG (4o)
Chinese prompt (%)	74.67	64.67	80
English prompt (%)	70	92	96

In addition, we conducted experiments analyzing the local deployment of the Caption RAG system using two local models: TAIDE 8B and LLaMA 8B. The results are presented in the

Table IV. The results reveal that LLaMA significantly outperforms TAIDE in terms of answer accuracy, achieving an LLM Score of 92% compared to TAIDE's 74.67%. This demonstrates that our proposed system can be effectively deployed on local models with relatively lower computational resources, showing promising potential for real-world applications.

Fig. 6 is the inference time comparison in this study. We evaluated the local models (TAIDE, LLaMA) on a GeForce RTX 4090 GPU. The results reveal that the LLaMA model demonstrates superior inference efficiency compared to the TAIDE model, despite both models having a similar number of parameters.

Furthermore, the Caption RAG method demonstrates significantly shorter inference time, since it only relies on textual information and can directly operate on pre-extracted captions without the need for complex multimodal alignment. In contrast, the VLM RAG method exhibits longer inference time. This is because it needs to check whether the retrieved textual content contains image tags and then process the identified images through the VLM. Specifically, this involves passing the image data through a vision encoder for alignment, followed by cross-modal integration, which collectively leads to longer inference time.

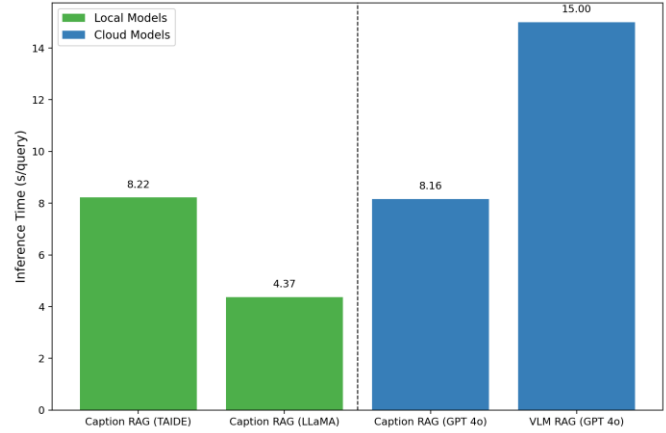


Fig 6. Inference time comparison

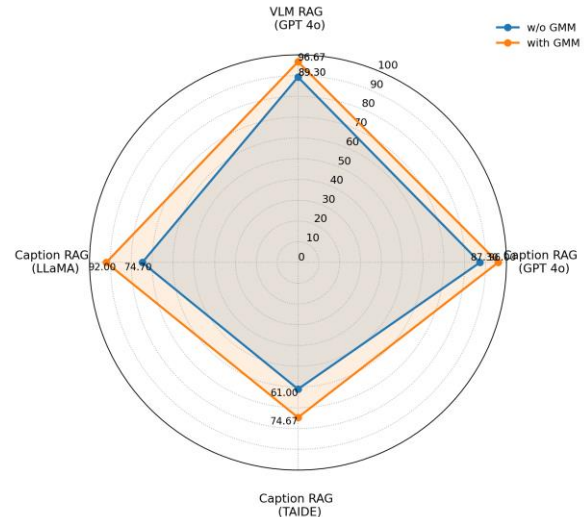


Fig 7. Comparison of different RAG methods under two settings: without GMM and with GMM. We also carried out experiments to assess the impact of

integrating the GMM module into our system, and the results are presented in Fig. 7. The blue lines indicate the accuracy of the LLM score without the GMM module (w/o GMM), while the orange lines represent the accuracy of the LLM score with the GMM module (with GMM). The results demonstrate that the integration of the GMM module consistently improves the overall accuracy. This improvement arises because the GMM effectively filters out irrelevant noise, thereby reducing the number of unrelated chunks that are passed to the LLM and potentially interfere with its responses. The effect of the GMM module is particularly evident in local models. For example, the difference between the Caption RAG (TAIDE) and the Caption RAG (LLaMA) results with and without the GMM module is more significant. This is because local models have a limited number of parameters, unlike large-scale cloud-based models such as GPT-4o, which possess a significantly larger parameter capacity. As a result, when too many noisy chunks are input into local models, their responses are more likely to become inaccurate.

E. Cost Analysis

In our proposed methods, since we used the cloud-based model GPT-4o, the use of its API leads to corresponding costs. Therefore, we conducted a cost analysis. At the time of our calculation, the pricing for GPT-4o was **\$2.5 per one million input tokens** and **\$10 per one million output tokens**. The results of this analysis are summarized in Table V.

In the Caption RAG method, images are first converted into textual descriptions through the image description process, which results in an additional cost. However, this step must be performed only once. In contrast, the VLM RAG method directly feeds the retrieved images into the VLM, and thus does not incur such pre-processing costs.

Regarding query processing, the Caption RAG method requires only textual input, leading to relatively lower API usage costs. On the other hand, VLM RAG directly processes both text and images within the VLM, resulting in significantly higher costs. Overall, the API usage cost of VLM RAG is approximately 6.8 times greater than that of Caption RAG.

TABLE V
Cost comparison of different RAG methods (USD)

Method	Image Description (Only one time)	Per query
Caption RAG (TAIDE)		\$0
Caption RAG (LLaMA)	\$2.36	\$0
Caption RAG (GPT 4o)		\$0.0073/query
VLM RAG	\$0	\$0.05/query

F. Case Study

In Table VI, this case study presents the input question: **How to start Snake Dance project?**

For comparison, we employed the latest web-based version of ChatGPT 5 [30]. In this test, we uploaded the same document to the ChatGPT web interface and asked the same question. Since the relevant instructions in the document are presented in image form, our system was able to correctly answer the question because it includes an image-processing module that extracts and explains visual content.

At the same time, our system's response also includes **Citation Pages**, which serve as direct references to the source document. This feature allows users who may still have doubts

about the system's generated explanation to trace back and quickly locate the exact page where the answer originates. Moreover, it helps prevent the LLM from introducing information unrelated to the document during the generation process.

In contrast, the ChatGPT 5 web version returned the response "The document does not provide direct instructions on how to start the Snake Dance project."

This outcome highlights a key advantage of our method: by incorporating specialized image understanding, our system can accurately respond even when the necessary information appears as images within the original document, whereas the ChatGPT 5 web version's document reading function fails to do this case.

G. Cross-Platform Interface

We designed a dedicated user interface that integrates seamlessly with the proposed system. Two versions of the interface were developed: **a web-based version** and **a mobile application**. In Fig. 8 (a), the web version is intended to be deployed alongside production machines or used in employee training programs. To enhance accessibility, we further implemented a mobile version, as illustrated in Fig. 8 (b), enabling users to interact with the system more conveniently on their personal devices.

In addition, a feedback system was incorporated into the interface. Specifically, a feedback button allows users to evaluate the chatbot's responses. If a user indicates dissatisfaction, a problem-reporting dialog is triggered, through which detailed feedback can be submitted. This feedback system provides us with clearer insights into user experiences and helps guide the continuous improvement of the system.

IV. CONCLUSION

To improve RAG systems for multimodal technical documents, this study introduces two methods: **Caption RAG** and **VLM RAG**. By separating text and images and applying vision-language model (VLM) pipelines, both approaches enhance the system's ability to understand and respond to documents containing visual data. Caption RAG converts images into contextual descriptions, while VLM RAG processes retrieved text and images together using GPT-4o for more accurate responses. Experimental results, validated on datasets curated by senior experts from an industrial robotic arm manufacturer, show that VLM RAG (GPT-4o) achieves the highest accuracy at 96.67%, while Caption RAG (GPT-4o) scores a BERT Score of 76.4%, reflecting strong semantic consistency. Both outperform the baseline Naive RAG in accuracy and coherence. Caption RAG was also tested in a local environment using the LLaMA model, achieving 92% accuracy with an average inference time of 4.37 seconds, demonstrating suitability for resource-limited settings. In summary, the proposed multimodal RAG system enhances image-text understanding, delivers high accuracy and efficiency, and supports flexible deployment—making it well-suited for real-world applications such as intelligent maintenance and enterprise knowledge management.

TABLE VI
COMPARISON BETWEEN OUR SYSTEM AND CHATGPT 5 WEB VERSION

Q: How to start Snake Dance project	Answer: To start the "Snake Dance" project for testing robotic arms, follow these detailed steps. 1. Configuration of Nodes: Create two nodes with specific joint angles for six joints (J1 through J6). The angles for Node 1 and Node 2 are mirrored as follows: J1: 260, -260 J2: 90, -90 J3: -150, 150 J4: 90, -90 J5: 170, -170 J6: 110, -110 2. Execution of the Test: After configuring the nodes with the specified joint angle, run these nodes in a loop at 60% speed for 30 minutes. 3. Assessment: During the test, monitor the robotic arm for any abnormalities. If no abnormalities are detected, the robot is considered ready for use. This procedure ensures that the robotic arm is functioning correctly and is ready for operational tasks. The "snake dance" test is a critical step in verifying the robot's readiness and performance. Citation Pages: Techman confidential 78 (Page 79, paragraph describing the snake dance procedure). [Confidence: 9]
ChatGPT 5 (OpenAI, Web version)	The document does not provide direct instructions on how to start the Snake Dance project. However, it can be inferred that this is a post-calibration verification procedure. You must first complete all calibration processes and save the calibration data, then execute the "Snake Dance" test in Maintenance Mode to verify the motion accuracy of the system. If detailed operation steps are required, it is recommended to refer to Techman's software operation manual or internal maintenance tool (Maintenance Mode) documentation for further guidance.

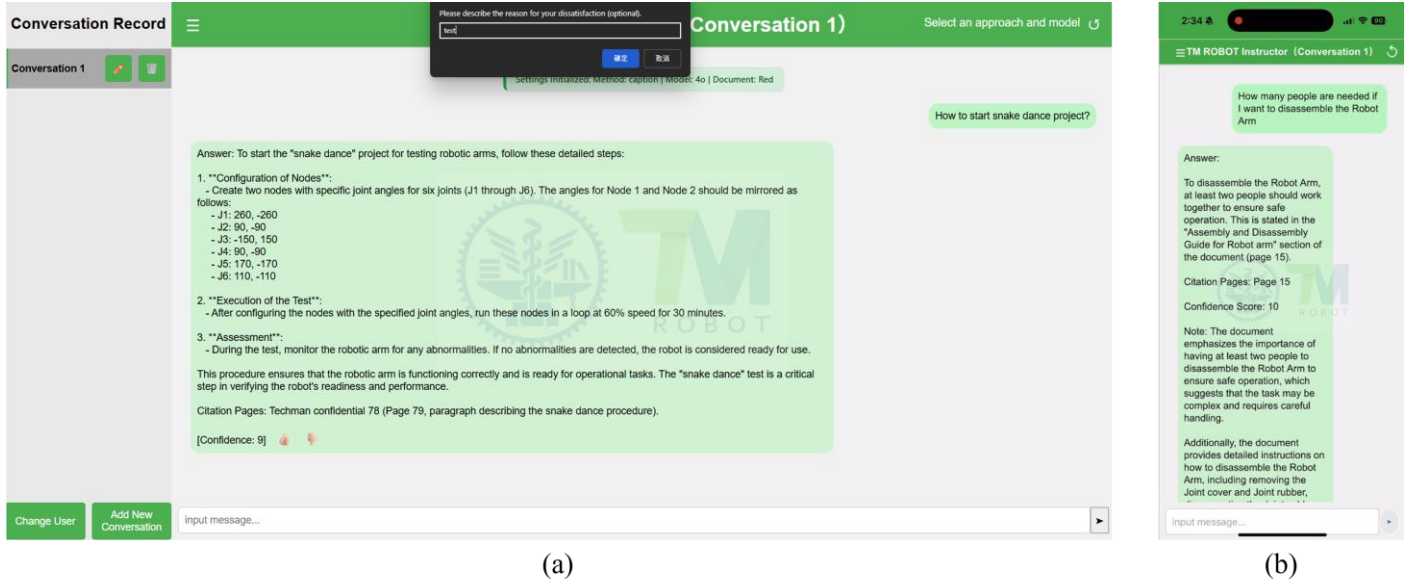


Fig 8. User interface of the proposed system: (a) Web version with problem-reporting dialog, (b) Mobile application.

ACKNOWLEDGMENT

This work was supported in part by the National Science and Technology Council of Taiwan under contracts NSTC 113-2224-E-A49-001, 114-2224-E-A49-004, and 114-2218-E-A49-022, and supported by the Center for Intelligent Team Robotics and Human-Robot Collaboration under the "Top Research Centers in Taiwan Key Fields Program" of the Ministry of Education (MOE), Taiwan. The authors sincerely appreciate the provision of technical support and the product parameters by the staff of TECHMAN ROBOT INC, Taiwan.

REFERENCES

[1] M. A. K. Raiaan, M. S. H. Mukta, K. Fatema, N. M. Fahad, S. Sakib, M. M. J. Mim, J. Ahmad, M. E. Ali, and S. Azam, "A review on large

language models: Architectures, applications, taxonomies, open issues and challenges," IEEE Access, vol. 12, pp. 26839–26874, 2024.

[2] Y. Fukui, Y. Kawata, K. Kobashi, Y. Nagatani, and H. Iguchi, "Evaluation of a retrieval-augmented generation system using a japanese institutional nuclear medicine manual and large language model-automated scoring," Radiological Physics and Technology, vol. 18, no. 3, pp. 861876, 2025.

[3] 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.

[4] R. Rosati, F. Antonini, N. Muralikrishna, F. Tonetto, and A. Mancini, "Improving industrial question answering chatbots with domain-specific llms fine-tuning," in 2024 20th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA), 2024, pp. 1–7.

[5] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kuttler, M. Lewis, W.-t. Yih, T. Rocktaschel et al., "Retrieval augmented

- generation for knowledge-intensive nlp tasks,” *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.
- [6] S. Vidielli, M. Ramachandran, and A. Dharunbalaji, “Efficiency driven custom chatbot development: Unleashing langchain, rag, and performance-optimized llm fusion,” *Computers, Materials & Continua*, vol. 80, no. 2, 2024.
- [7] M. Hindi, L. Mohammed, O. Maaz, and A. Alwarafy, “Enhancing the precision and interpretability of retrieval-augmented generation (rag) in legal technology: A survey,” *IEEE Access*, vol. 13, pp. 46171–46189, 2025.
- [8] R. S. M. Wahidur, S. Kim, H. Choi, D. S. Bhatti, and H.-N. Lee, “Legal query rag,” *IEEE Access*, vol. 13, pp. 36978–36994, 2025.
- [9] Y. H. Ke, L. Jin, K. Elangovan, H. R. Abdullah, N. Liu, A. T. H. Sia, C. R. Soh, J. Y. M. Tung, J. C. L. Ong, C.-F. Kuo et al., “Retrieval augmented generation for 10 large language models and its generalizability in assessing medical fitness,” *npj Digital Medicine*, vol. 8, no. 1, p. 187, 2025.
- [10] J. Li, Y. Yuan, and Z. Zhang, “Enhancing llm factual accuracy with rag to counter hallucinations: A case study on domain-specific queries in private knowledge-bases,” *arXiv preprint arXiv:2403.10446*, 2024.
- [11] S. Yu, C. Tang, B. Xu, J. Cui, J. Ran, Y. Yan, Z. Liu, S. Wang, X. Han, Z. Liu et al., “Visrag: Vision-based retrieval-augmented generation on multi-modality documents,” *arXiv preprint arXiv:2410.10594*, 2024.
- [12] Q. Wang, R. Ding, Z. Chen, W. Wu, S. Wang, P. Xie, and F. Zhao, “Vidorag: Visual document retrieval-augmented generation via dynamic iterative reasoning agents,” *arXiv preprint arXiv:2502.18017*, 2025.
- [13] J. Zhang, Q. Zhang, B. Wang, L. Ouyang, Z. Wen, Y. Li, K.-H. Chow, C. He, and W. Zhang, “Ocr hinders rag: Evaluating the cascading impact of ocr on retrieval-augmented generation,” *arXiv preprint arXiv:2412.02592*, 2024.
- [14] M. AI, “LLaMA 3.1 8B Instruct on hugging face,” <https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct>, 2024.
- [15] T. L. L. M. D. under the National Science and T. C. (NSTC), “Llama-3.1 taide-lx-8b-chat on hugging face,” <https://huggingface.co/taide/Llama-3.1-TAIDE-LX-8B-Chat>, 2024.
- [16] Jsvine, “pdfplumber: Plumb a PDF for detailed information about each char, rectangle, line, et cetera and easily extract text and tables,” <https://github.com/jsvine/pdfplumber>, 2016, accessed: 2025-05-22.
- [17] OpenAI, “Gpt-4o: Introducing our new flagship model,” 2024, <https://openai.com/index/hello-gpt-4o/>.
- [18] Meta AI, “Llama 3.2 90b vision: Multimodal model released on hugging face,” <https://huggingface.co/meta-llama/Llama-3.2-90B-Vision>, 2024, accessed: 2025-05-22.
- [19] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge et al., “Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution,” *arXiv preprint arXiv:2409.12191*, 2024.
- [20] langchain-ai langchain contributors, “LangChain: Build context-aware reasoning applications,” 2025.
- [21] OpenAI, “text-embedding-3-large: Next-generation large embedding model,” Official announcement and documentation, 2024.
- [22] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou, “The faiss library,” *arXiv preprint arXiv:2401.08281*, 2024.
- [23] Hugging Face, “Cross-encoder: ms-marco-minilm-l-6-v2,” <https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2>, 2020.
- [24] R. Nogueira and K. Cho, “Passage re-ranking with bert,” *arXiv preprint arXiv:1901.04085*, 2019.
- [25] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, “Bertscore: Evaluating text generation with bert,” *arXiv preprint arXiv:1904.09675*, 2019.
- [26] Y. Wang, A. G. Hernandez, R. Kyslyi, and N. Kersting, “Evaluating quality of answers for retrieval-augmented generation: A strong llm is all you need,” *arXiv preprint arXiv:2406.18064*, 2024.
- [27] 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.
- [28] A. Panickssery, S. R. Bowman, and S. Feng, “LLM evaluators recognize and favor their own generations,” in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37. Curran Associates, Inc., 2024, pp. 68772–68802. [Online]. Available: <https://proceedings.neurips.cc/paper/2024/file/7f1f0218e45f514c79c0679633e47bc-Paper-Conference.pdf>
- [29] G. DeepMind, “Gemini 2.0: A family of multimodal ai models,” 2025, <https://deepmind.google/technologies/gemini/>.
- [30] OpenAI, “Chatgpt 5,” <https://chat.openai.com>, 2025.



Yen-Hua Lu was born in Changhua, Taiwan, in 2002. He received the B.S. degree from the Department of Automatic Control Engineering, Feng Chia University, Taichung, Taiwan, in 2024. He is currently pursuing the M.S degree with the Intelligent Control and Applications Laboratory, Institute of Electrical and Control Engineering, National Yang Ming Chiao Tung University, Hsinchu, Taiwan.



Ching-Hung Lee (Senior Member, IEEE) was born in Taiwan in 1969. He received his B.S. and M.S. degrees from the Department of Control Engineering, National Chiao Tung University, Hsinchu, Taiwan, in 1992 and 1994, respectively, and his Ph.D. degree from the Department of Electrical and Control Engineering, National Chiao Tung University, in 2000. He is currently the Ding-Hua Hu Chair Professor and Distinguished Professor of the Institute of Electrical and Control Engineering, National Yang Ming Chiao Tung University, Taiwan. He received

the Researcher Excellence Award and the Wu Ta-Yu Medal (Young Researcher Award) from the National Science and Technology Council, Taiwan, in 2023 and 2008, respectively. He was also awarded the Fellow, Youth, and Excellent Automatic Control Engineering Awards from the Chinese Automatic Control Society, Taiwan, in 2019, 2009, and 2016, respectively. He is currently an Associate Editor of *IEEE Sensors Journal* and *International Journal of Fuzzy Systems*. His research interests include artificial intelligence, smart manufacturing, signal processing, deep learning, generative AI, robotics control, motion control, and optimization.