2026 IEEE International Conference on Artificial Intelligence and eXtended and Virtual Reality (AIxVR) | 979-8-3315-4967-1/26/$31.00 ©2026 IEEE | DOI: 10.1109/AIXVR67263.2026.00015

Augmented Reality Visual Retrieval for Object Detection and Corpus-Guided Content Generation

Alireza Taheritajar*, Jieqiong Zhao†, Jason Orlosky‡
School of Computer and Cyber Sciences, Augusta University, Augusta, GA, USA
Email: *ataheritajar@augusta.edu, †jzhao@augusta.edu, ‡jorlosky@augusta.edu

***Abstract*—Object detection models have recently become lightweight enough for deployment on resource-constrained devices such as smartphones and Augmented Reality (AR) headsets. Techniques such as few-shot, zero-shot, and incremental learning have moved toward lower training times and faster inference for detection models, but most frameworks still require substantial training or retraining to dynamically add new object classes in real-time for practical use with AR. In this paper, we present a novel AR-integrated object detection system that leverages Visual Retrieval Augmented Generation (ARVRag for short), eliminating much of the lengthy and expensive training processes associated with traditional object detection models. By vectorizing a smaller set of training images and conducting a local similarity search, we achieve enough accuracy for practical use with AR in significantly less time than state-of-the-art models like YOLO. Moreover, we have built this process into support systems for maintenance and industrial operations. End-users can select an object using a 2D image capture interface that automatically retrieves semantically similar items from a local corpus (database), and the resulting detection provides metadata-enhanced object-specific responses from a large language model (LLM). This work demonstrates a practical and scalable pathway for LLM-supported object detection in AR applications, especially for manufacturing and repair.**

***Index Terms*—Mixed reality, augmented reality, object detection, visual retrieval augmented generation.**

I. Introduction

The integration of artificial intelligence (AI) and extended reality (XR) has created new opportunities to build intelligent systems that can interact with users in immersive environments. A key challenge in this area is enabling these systems to understand and describe objects in the 3D world in real-time. Object detection plays a central role in this task. Although many recent methods have improved the efficiency and accuracy of object detection (e.g., zero-shot [1], few-shot [2], and incremental learning [3]), they still face significant limitations. Adapting these techniques for novel objects during runtime presents significant challenges, especially when dealing with lightweight hardware. This limits their use in real-world augmented reality (AR) applications, where users require quick and flexible interaction with various objects.

At the same time, large language models (LLMs) and multimodal systems have demonstrated significant progress in tasks such as image captioning [4] and question answering [5]. However, they also have several limitations when applied to real-world 3D environments. Most of these models rely on cloud services, which can cause delays and raise privacy concerns when users have to send image data to external servers. Additionally, because they are typically pre-trained on everyday objects, these methods often struggle to accurately detect uncommon or domain-specific objects. This makes it challenging to apply these models to practical tasks in areas such as manufacturing, maintenance, or education, where accurate and private object understanding are necessary.

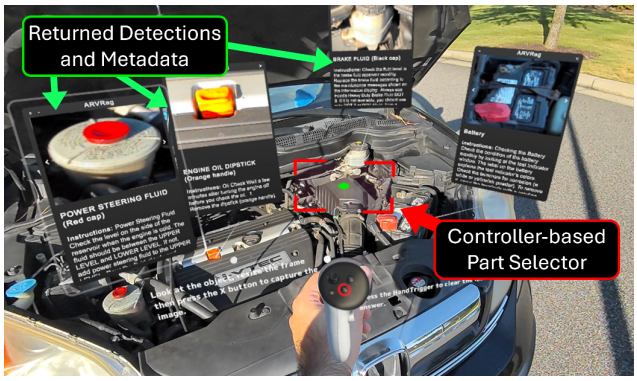

Fig. 1. Image demonstrating the functionality of our ARVRag framework, showing the resizable planar window used to select an object for detection (red window), and resulting in-situ detection results with corresponding metadata from the vector database (green window). An end-user can trigger both the object detection and metadata retrieval with a single controller interaction.

To address these issues, we introduce a novel system that combines object detection, visual retrieval, and language generation into a single, real-time pipeline designed for AR environments. Our system allows users to select an object in a 3D scene, retrieve the most visually similar match from a local database, and use its descriptive metadata as contextual input for an LLM. This enables the generation of object- and context-aware instructions without requiring the transmission of raw image data to an external cloud.

The main contributions of this paper are:

- An end-to-end Visual Retrieval Augmented Generation (VRAG) framework that enables object recognition and significantly reduces the overhead associated with training or re-training traditional models,
- an integrated 3D object selection interface that leverages head-mounted camera movements for object targeting and enables adjustable image cropping through controllers or hand gestures,
- and demonstration in an AI-assisted maintenance scenario, showing how users can interact with parts to

2771-7453/26/$31.00 ©2026 IEEE
DOI 10.1109/AIxVR67263.2026.00015

display real-time, context-aware AR guidance.

In short, this work aims to improve the scalability and practical usage of object recognition for AR applications that support manufacturing, repair, and training. By combining local object recognition with semantic reasoning, our system enables a more convenient interaction paradigm between users and AI-assisted systems in real-world 3D settings.

II. Related Work

We review four research areas that inform our approach: object detection paradigms, multimodal large language models, object interaction in AR, and visual retrieval with contextual generation. These areas highlight the strengths and limitations of current methods and motivate the design of our system.

A. Object Detection and Learning Paradigms

Traditional OD has been a central problem in computer vision, with models such as Faster R-CNN [6], YOLO [7], and SSD [8] forming the backbone of real-time detection systems. These approaches, however, rely heavily on large-scale labeled datasets and require full retraining to recognize new object classes. To address these limitations, alternative learning paradigms have been explored, including zero-shot [1], few-shot [2], and incremental learning [3]. Despite their promise, these methods often suffer from catastrophic forgetting, performance degradation on previously learned categories, and computational overhead, making them unsuitable for real-time AR environments where users interact dynamically with diverse objects.

Open-Vocabulary Object Detection: This type of detection extends detection beyond fixed label sets by leveraging vision-language models. ViLD [9] addresses this problem through knowledge distillation, transferring representations from a pretrained vision-language classification model to a two-stage detector. By aligning region embeddings with image and text embeddings, ViLD achieves strong generalization to novel categories, outperforming supervised baselines on LVIS [10] and transferring well to benchmark datasets such as PASCAL VOC, COCO, and Objects365. Similarly, Zareian et al. [11] propose combining limited bounding box annotations with plentiful image-caption pairs to scale detection to unseen object classes. Their method outperforms previous zero-shot approaches while maintaining high accuracy on seen classes, striking a practical balance between supervised and weakly supervised learning for scalability.

B. Multimodal Large Language Models

Multimodal large language models (MLLMs), such as CLIP [12], Flamingo [13], and GPT-4V [14], integrate vision and language to support tasks like captioning, visual question answering (VQA), and cross-modal retrieval. Despite their powerful capabilities, these models are trained on general-purpose datasets, struggle with fine-grained or domain-specific object recognition, and often rely on cloud infrastructure, limiting their suitability for real-time XR. To address these issues, approaches such as VCoder [15] incorporate additional modalities (e.g., segmentation masks and depth maps) and introduce enriched datasets (e.g., COST) to improve perception; however, challenges remain in basic tasks such as counting or detecting small, overlapping objects.

Object Detection with MLLMs: Extending MLLMs to detection, ContextDET [16] introduces a generate-then-detect pipeline that leverages LLM tokens to dynamically adapt to open-vocabulary tasks, showing promising results on their generated Contextual Object Detection (CODE) benchmark. Domain-specific efforts include Florence-2 + SAM2 for waste detection [17], which achieves fine-grained segmentation but struggles with broader categorization, such as plastic and metal. In 3D, a survey [18] emphasizes the challenges of aligning text with point clouds and voxel grids, while Chen et al. [19] address spatial reasoning gaps by generating large-scale 3D VQA datasets, enabling improvements in distance estimation, size comparison, and downstream applications in AR and robotics. Together, these works highlight both the potential and current limitations of MLLMs for object detection and spatially grounded XR cases.

C. Object Interaction in Augmented Reality

Object-centric interaction in AR has been widely studied, with prior work exploring methods for selecting, labeling, and tracking 3D objects using bounding boxes, gaze tracking, or hand gestures [20], [21]. Existing systems that incorporate object recognition into head-mounted displays often rely on fixed sets of labels, reducing their flexibility in handling unseen object classes. To address these limitations, Taheritajar et al. [22] propose an active incremental learning system that supports the efficient addition of new classes in an AR language learning application. Our work builds on this type of approach by completely replacing the object detection model and retraining pipeline with vectorization and direct retrieval, which works at a fraction of the time and computation cost.

Recent advances in semantic scene understanding have introduced more profound AI–AR integration, yet many systems still depend on pre-trained detectors and do not leverage LLMs to generate explanations or responses that humans easily understand. Patricio et al. [23] integrate AI with AR to enhance satellite assembly, integration, and testing (AIT) using Microsoft HoloLens 2. To improve data efficiency, they employ synthetic data and introduce SAMAL (Segment Anything Model for Automatic Labelling), which accelerates annotation while maintaining high accuracy. This work demonstrates the potential of AI–AR systems to improve industrial workflows.

D. Visual Retrieval and Contextual Generation

Visual retrieval techniques aim to match queries–typically images or regions–with similar items in a database using learned embeddings [24], [25], and are fundamental to large-scale image search and recognition. More recently, Retrieval-Augmented Generation (RAG) approaches combine visual or textual retrieval with LLMs, enriching prompts with external knowledge to improve reasoning and generation [26]. While

RAG has proven effective in natural language question answering, its application to real-time AR remains limited.

Several studies highlight the advantages of retrieval-enhanced reasoning in multimodal systems. Jing et al. [27] show that coupling LLMs with vector databases mitigates hallucination, outdated knowledge, and memory constraints by enabling vector-based storage and retrieval. Similarly, Huang et al. [28] and Sharifymoghaddam et al. [29] introduce Uni-RAG, a model-agnostic framework that integrates interleaved image–text retrieval at inference time. UniRAG improves captioning and generative tasks across both commercial and open-source models (e.g., GPT-4o [30], Gemini-Pro [31], LLaVA [32]), but its applicability to spatially grounded AR tasks remains underexplored.

Beyond RAG, Zhu et al. [33] propose a scalable multimodal knowledge base that integrates visual, textual, and structured data via a Markov Random Field, enabling flexible query handling without retraining. Yasunaga et al. [34] further emphasize that plug-and-play, in-context retrieval is underutilized in vision–language systems, highlighting the need for adaptable systems in dynamic AR environments. Efforts toward fine-grained reasoning include Xue et al. [35], who enhance multimodal RAG with structured scene graphs, improving spatial and relational reasoning on tasks such as object localization, quantification, and VQA. In parallel, similarity-based alignment methods such as ViSGA [36] demonstrate the value of feature-level retrieval in object detection by aligning instance-level features through adversarial group adaptation.

Overall, across the reviewed areas, prior AR object detection and retrieval approaches generally rely on large-scale supervised training and static model updates (e.g., YOLO-based pipelines). While these methods achieve strong benchmark results, they often suffer from long retraining times, poor adaptability to new classes, and sensitivity to environmental conditions such as low light. Our proposed VRAG-based framework addresses these gaps by combining CLIP-based retrieval with lightweight database insertion, enabling rapid integration of new categories, metadata augmentation, and robustness across varied capture sources. In contrast to prior literature, this shifts the emphasis from heavy model training toward flexible retrieval and multimodal integration, which we argue is more suitable for real-world AR.

III. FRAMEWORK AND METHODOLOGY

We present the technical architecture and key components of our proposed system, which integrates visual retrieval and a generative language model to support object understanding in AR. The system is designed for real-time interaction on the Meta Quest 3, but it is scalable to most any other AR headsets that can register spatial information and have a forward-facing camera. The Quest in our implementation is connected to a local PC that functions as a server and performs inference and retrieval operations that are streamed between server and headset, as shown in Figure 2. The goal is to assist users, particularly in industrial assembly scenarios, by enabling them

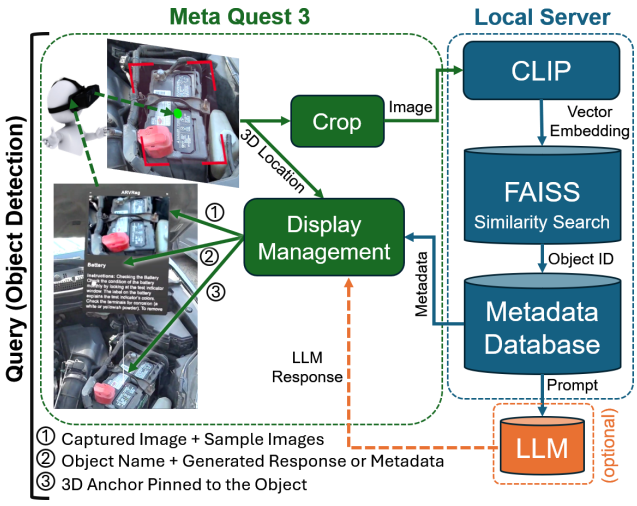

Fig. 2. Object detection pipeline: the user selects an object via a crop function, which is then sent to the VRAG server. The server generates vector embeddings, retrieves the best match and metadata, and uses an LLM to produce a response. The headset then receives the output with 1) sample images, 2) adds text annotations, and 3) overlays a 3D leader line.

to retrieve object metadata and usage instructions through an effective 3D interface.

1) System Overview: The system consists of two main components and one optional component: (1) a user-side interface running on Meta Quest 3 and (2) a backend server hosted on a local PC, and optionally (3) a local or cloud-based LLM server. As shown in Figure 2, users begin by selecting and cropping a region of interest within their visual field using a custom UI implemented in the AR environment. The cropped image is then sent to the server via a local network using FastAPI endpoints. This architecture ensures low latency and avoids cloud-dependent image processing, improving privacy and performance.

Upon receiving the image, the server encodes it into a vector embedding using the Contrastive Language–Image Pretraining (CLIP) [12] model. In general, CLIP jointly learned visual and textual representations by training on large-scale image–text pairs. However, in this system, only the image encoder of CLIP is used to generate a dense embedding of the cropped image. This embedding is then compared to a pre-indexed database using FAISS [37], an efficient library for similarity search.

FAISS performs an Approximate Nearest Neighbor (ANN) search to find the most visually similar image in the database. In addition, we applied a threshold of 0.4 to the calculated *Distance* parameter, a value empirically determined to improve the accuracy of object detection. We also dedicated another database to record the metadata and sample images related to each object by its object ID. So each query result contains not only an image embedding but also metadata, sample images (optionally), and descriptive instructions provided by domain experts (optionally).

Furthermore, once the best match is identified, its metadata is retrieved and integrated into a predefined prompt format.

This prompt is then passed to an LLM, which may be hosted locally or accessed through an online API, depending on deployment preferences. Note that the LLM response is optional based on the needs of the application, and the system can use just the detected object name or metadata as an annotation. Finally, the generated response is returned to the Meta Quest 3 headset, where it is rendered in 3D space at the location of the originally selected object. A flexible 3D UI displays the textual answer, along with sample images associated with the retrieved object. This provides contextual assistance to users in understanding the object's function or assembly process.

A. Spatial Object Selection and Pixel Cropping in AR

To enable intuitive object selection in an AR environment, we designed a *virtual camera frame* that is dynamically positioned based on the user's view (similar to Gaze). This frame leverages the Depth API of Meta Quest 3 to determine a raycast from the user's left camera (an approximation of head direction) into the scene. A ray is cast onto the environment mesh, and the frame is positioned at the raycast hit point on the surface geometry, typically aligning with the object the user is looking at. Once the frame is placed, the user can adjust its size to fit the desired object. This adjustment can be performed using either hand gestures or joystick inputs from Meta Quest controllers. The resizing operation preserves the frame's alignment with the surface, allowing the user to accurately encapsulate the object regardless of its distance or orientation in the 3D environment.

After framing the object (placing a view-aligned virtual camera frame around the object), the user triggers the capture process. At this point, the system captures the front camera image along with the 3D world coordinates of the four corners of the frame. These corner positions are then projected into the image's UV coordinate space using the headset's camera intrinsics and transformation matrices. This process enables the system to precisely crop the selected region from the captured image.

The resulting cropped image is then transmitted to the server for further processing, including vector embedding, visual retrieval, and LLM-based answer generation as described in previous subsections. This object selection mechanism enables low-effort interaction, allowing users to query object information simply by looking at and framing the object, without relying on manual object recognition or segmentation pipelines.

We also experimented with different methods for manually positioning the virtual camera frame in the scene without depending on the environment mesh provided by the Depth API. However, we encountered errors in the position calculations for the rendered frame, which caused some level of inaccuracy when converting the corner locations to UV coordinates. Consequently, we decided to use the raycast hit from the mesh instead to ensure the accuracy of the object's 3D position and to keep it centered in the field of view at the time of image capture.

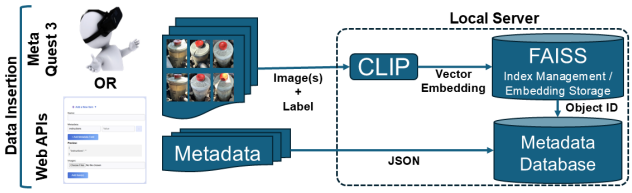

Fig. 3. Data insertion process for storing images and associated metadata in the FAISS database. The process can be performed either through Meta Quest 3 or via the webpage interface.

B. Visual Retrieval Augmented Generation

Our system uses VRAG as an image classification approach, as a system explicitly tailored for AR environments. Rather than relying solely on raw visual input, our system performs visual retrieval of object metadata, which is then incorporated into large language model prompts. This design enables accurate, context-aware, and privacy-preserving responses to be generated directly on-device, supporting real-time interaction with spatially selected objects.

1) CLIP for Visual Embedding: Contrastive Language–Image Pretraining (CLIP) [12] is a vision-language model trained on 400 million image–text pairs using a contrastive learning objective. It learns to project both images and text into a shared embedding space where semantically similar pairs are mapped close together. In our system, only the image encoder component of CLIP is utilized to extract visual embeddings from user-captured or admin-uploaded images.

We adopt the ViT-B/32 variant of CLIP, which utilizes a Vision Transformer backbone and provides a suitable trade-off between accuracy and computational efficiency. The model can execute on a GPU if one is available; otherwise, it will fall back to using the CPU. To generate an embedding, the input image is first converted to RGB and passed through CLIP's standard preprocessing pipeline. This includes resizing, center cropping, normalization, and tensor formatting. The processed image is then forwarded through the encoder, and the resulting feature vector, $\mathbf{v}_{\text{norm}}$, is L2-normalized to ensure consistent scaling and cosine similarity compatibility:

$$\mathbf{v}_{\text{norm}} = \frac{\mathbf{v}}{\|\mathbf{v}\|_2}$$

The final output is a fixed-length vector (512-dimensional in ViT-B/32) returned as a NumPy array and used directly for similarity search via FAISS. This embedding step is performed without gradients, ensuring low-latency execution during inference.

2) FAISS for Similarity Search: Facebook AI Similarity Search (FAISS) [37], [38] is a high-performance library for efficient nearest neighbor search in high-dimensional vector spaces. It is widely used in large-scale similarity search tasks due to its speed, scalability, and support for various indexing structures such as Inverted File (IVF), Product Quantization (PQ), and Hierarchical Navigable Small World (HNSW). In our system, we adopt the `IndexFlatL2` index, which is

a brute-force exact nearest neighbor search based on L2 (Euclidean) distance. Despite its simplicity, it provides fast and reliable performance for small to medium-sized embedding collections, and is particularly suitable for real-time inference with moderate hardware resources.

We developed a custom wrapper class around FAISS that handles both vector indexing and metadata management. Each object embedding, produced by the CLIP image encoder, is cast to a `float32` NumPy array and reshaped to the required `(1, d)` format before being added to the index. Each embedding is assigned a unique identifier (e.g., object or part ID), which is stored separately to enable mapping between FAISS's internal indices and the scene-level metadata repository.

This implementation enables efficient retrieval without retraining or reloading the full database, making it well-suited for real-time AR applications. The modular design also allows for future upgrades, such as switching to approximate indices (e.g., IVF+PQ or HNSW) as the database size scales.

3) Data Entry and Object Indexing Pipeline: As shown in Figure 3, to populate the visual database, we developed a data entry pipeline with a web-based frontend and RESTful APIs. Through this interface, administrators can upload object images and attach corresponding metadata, including instructions, category labels, and sample images. The CLIP image encoder processes each image to extract its vector representation, which is stored in the FAISS index alongside its metadata. To improve robustness against viewpoint variation, administrators can upload multiple images of the same object taken from different angles. These entries are linked to the same metadata, enabling the system to match user-captured images regardless of perspective. For example, an object like a wheel may be indexed using several images from different views but tied to a single semantic concept in the database. This design enhances retrieval accuracy and supports diverse visual inputs during runtime.

For any RAG system, it is essential to maintain a clean and well-structured database. To achieve this, we utilized the 3D annotation method proposed by Taheritajar et al. [22] to capture images of all test objects while removing background clutter and collecting multiple views of each object. Additionally, we carefully proofread the text extracted from commercial manuals (e.g., robots, cars) before linking it to the corresponding object images.

4) Prompt Construction and LLM Inference: Once a matching object is retrieved, the system constructs a prompt by embedding its metadata into a predefined template. This includes a description, assembly instructions, and any other relevant context. The prompt is sent to an LLM for natural language generation. The LLM may be a local model (e.g., LLaMA or Mistral) or an online API-based model such as GPT-4, depending on resource availability and latency requirements. The output from the LLM is then visualized within the AR environment using a dynamic UI panel. The UI is anchored to the spatial coordinates of the selected object, enhancing immersion and contextual understanding for the user.

5) Flexible 3D Display Panel and Spatial Anchoring: To present the generated information within the Augmented reality environment, we developed a flexible 3D display panel system. This panel serves as the primary user interface for visualizing the language model's response, associated metadata, and relevant example images corresponding to the selected object. It is instantiated in the scene after the completion of the retrieval and generation pipeline.

The display panel is spatially anchored via a visual connection—a thin 3D arrow—that links it to a small semi-transparent sphere positioned at the geometric center of the selected object. This anchoring provides an intuitive spatial association between the UI content and its corresponding physical or virtual object. The anchor point is derived from the world-space center of the selection frame, while the arrow extends between the panel and this anchor sphere.

To ensure readability from any viewing angle, the panel always faces the user through a billboard effect that aligns the panel's normal with the camera's forward vector. Users can interactively reposition the panel. This enables manual layout management, helping users avoid visual occlusions between multiple UI elements or objects in the environment.

The panel is designed to be both informative and interactive. It displays:

- The original cropped image selected by the user,
- A set of sample images retrieved from the server that depict the same object from various viewpoints,
- Scrollable textual content generated by the LLM, including instructions or object descriptions.

To further enhance usability, the display panel is resizable. Users can adjust its scale dynamically using controller inputs or hand gestures, allowing for flexible reading experiences and accommodation of longer or shorter text. These design elements collectively support an immersive, ergonomic, and context-aware interaction flow within the AR application.

IV. SYSTEM EVALUATION

We evaluated our proposed VRAG system on the Meta Quest 3 headset connected to a local CPU-based server. The experiments focused on three aspects: (1) retrieval accuracy, (2) system latency and runtime performance, and (3) usability of the AR interface for spatial object selection and 3D information display.

A. Retrieval Accuracy and Comparison with Classification Baselines

As shown in Figure 4, the CLIP-based embedding and FAISS similarity search achieved reliable matching across a curated database of 23 car components. Each object was represented by an average of 2–5 images from different viewpoints (a total of 91 images). Using a similarity distance threshold of 0.4, the system correctly identified objects in 95.7% of test queries. False positives primarily occurred in visually similar parts such as screws and bolts, where shape overlap was high. Incorporating multiple viewpoint images reduced misclassification, confirming the robustness of our

TABLE I
COMPARISON OF YOLO11N-CLS (TRAINED FOR 100 EPOCHS) AND VRAG ACROSS THREE DATASETS. YOLO11N REQUIRES FULL RETRAINING FOR NEW CATEGORIES, WHILE VRAG ONLY REQUIRES INSERTING EMBEDDINGS INTO FAISS.

Dataset	#Train	#Test	#Classes	YOLO11n Top-1	VRAG Top-1	YOLO11n Training Time	VRAG Insertion Time
Oxford 102 Flowers	6,149	1,020	102	98.8%	90.7%	65 min	45 s
CIFAR-10	41,129	10,000	10	95.7%	91.0%	7 hrs	58 s
Car Parts (ours)	68	23	23	100.0%	95.7%	4 min	11 s

Fig. 4. Example of the AR interface for car component detection. The system identifies multiple engine components under the hood (e.g., washer fluid, power steering fluid, engine oil dipstick, brake fluid, engine oil fill cap, and battery) and overlays contextual retrieval cards containing both the component label and step-by-step maintenance instructions, as well as sample images retrieved from the database. The controller-defined planar window allows the user to select an object for identification in real-time.

data entry and indexing pipeline. Importantly, the database images were captured with a standard smartphone camera and manually cropped before insertion into the FAISS index. This demonstrates that the proposed approach is not restricted to the Meta Quest 3 capture pipeline and can flexibly integrate images collected from external sources, supporting its adaptability to diverse input modalities.

To further assess the general accuracy of the proposed retriever as an object classification system, we evaluated it on two widely used benchmarks: *CIFAR-10* [39] and the *Oxford 102 Flowers* dataset [40]. On *CIFAR-10*, which contains 10 object categories with 10,000 training and 41,129 test images, VRAG achieved a top-1 accuracy of 91.0%. On the more fine-grained *Oxford 102 Flowers* dataset, comprising 102 flower categories with 1,020 training and 6,149 test samples, the system reached 90.7% accuracy despite significant intra-class variation and inter-class similarity. Finally, on our custom *Car Parts* dataset, which included 23 categories with 23 training and 68 test images, the retriever obtained 95.7% accuracy, confirming its ability to generalize even in very small-data regimes.

We also compared the retrieval-based pipeline with a conventional supervised classification baseline, YOLO11n-cls, trained for 100 epochs on each dataset using an early stopping strategy (patience = 20) on our *NVIDIA 3080* GPU. Table I summarizes the results. On Oxford 102 Flowers, YOLO11n-cls achieved 98.8% accuracy, outperforming VRAG's 90.7%, but required over an hour of training time. On CIFAR-10, YOLO11n-cls reached 95.7%, again higher than VRAG but at the cost of approximately seven hours of training. For our Car Parts dataset, YOLO11n-cls achieved perfect classification (100%), while VRAG followed closely with 95.7%, requiring only seconds to insert embeddings. These comparisons illustrate the trade-off between peak accuracy and efficiency: while YOLO11n-cls consistently produces slightly higher accuracy, it demands extensive retraining whenever new categories are added. By contrast, VRAG supports near-instant category insertion, making it especially well-suited for rapid deployment and dynamic AR scenarios.

Overall, these results demonstrate that the CLIP+FAISS retrieval pipeline is effective not only for the targeted AR use case but also in general-purpose object recognition tasks. While supervised classifiers like YOLO11n-cls offer superior peak accuracy, VRAG provides improved efficiency, requiring no retraining and enabling instant insertion of new object categories. This makes it particularly valuable in dynamic environments where new object classes appear constantly.

B. Metadata Integration for Contextual Enrichment

Beyond visual retrieval, our pipeline supports the integration of textual metadata associated with each object. In the car parts dataset, we attached descriptive information sourced from the company's customer manuals to each component. This metadata can be directly displayed in AR overlays or combined into prompts for LLMs, enabling enriched responses such as maintenance instructions, compatibility checks, or contextual explanations. Notably, if we rely solely on data retrieval without using an LLM for generative responses, the accuracy of the retrieved information is 100%, reflecting the reliability of our metadata integration. Such integration illustrates the potential of retrieval-augmented AR systems to bridge visual recognition with interactive, knowledge-driven assistance.

C. Latency and Runtime Performance

We measured the end-to-end latency of the system, starting from spatial object selection to the final rendering of information in the AR interface. Table II summarizes the breakdown across processing stages.

The largest contributor was the display rendering stage (600 ms), which accounted for over 75% of the total delay. The server-side embedding and retrieval (including transmission) required only 100 ms, while local preprocessing steps (projection, cropping, and texture conversion) together added 95 ms. Despite the rendering overhead, the system achieved a total latency of 795 ms, which remains below the 1-second threshold generally considered acceptable for real-time AR interaction.

TABLE II
LATENCY BREAKDOWN OF THE RETRIEVAL PIPELINE IN MILLISECONDS (MS).

Processing Stage	Latency (ms)
3D Points to 2D Pixel Projection	40
Cropping	15
Texture to JPG Conversion	40
Transmission + Server Processing	100
Display Rendering in AR	600
Total End-to-End Latency	**795**

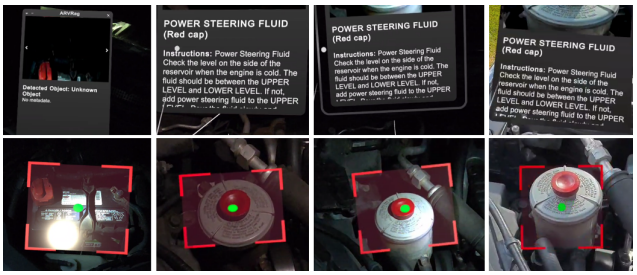

Fig. 5. Captures under various low-light conditions, including (from left to right): a failure case using lighting from only a smartphone, a success case from the same smartphone, a success case from a flashlight, and a success case in daylight. Though poor lighting and noise present challenges to detection accuracy, initial results are still promising.

This demonstrates that the proposed architecture can support responsive, in-situ assistance during industrial assembly tasks. In comparison, the authors of [22] reported that the average overall end-to-end latency for their Yolo-based object detection method on the Meta Quest 3 device was approximately 200 ms. However, their method only displays a simple label on top of each detected object and does not account for the rendering latency that is relevant to our approach. Importantly, if the retrieved metadata or cropped image is forwarded to an LLM, the response time depends on whether a local or cloud model is used. A lightweight local model typically adds 300-500 ms, whereas a cloud-based LLM can add higher variability depending on network conditions and model size.

V. DISCUSSION AND FUTURE WORK

Practical testing of our automotive maintenance assistant (see supplementary video) revealed several insights about the system's applicability and limitations. The framework effectively integrates object detection, information retrieval, and interactive AI guidance, allowing real-time assistance in tasks that require recognizing and understanding automotive components. This demonstrates clear utility in educational contexts, where students can learn about components interactively, and in industrial assembly or maintenance, like preventive maintenance checks and services (PMCS), where technicians can receive contextual guidance without referring to manuals.

Even with greater than 90 percent accuracy for datasets with over 100 classes present, the vector database may still not scale for exceptionally large databases, for example, when many small or similar parts are present. To alleviate this problem, we are exploring a cascading model that first recognizes higher-level objects (for example a type of car) and then switches to a more fine-grained model (for example a Toyota Camry engine) to improve accuracy.

Using LLMs allows the system to generate dynamic, context-aware responses by combining user queries with retrieved information. This facilitates flexible prompt engineering and provides a foundation for a responsive AI assistant capable of handling diverse user queries. However, the system exhibits notable sensitivity to environmental factors, particularly lighting. Low-light trials using a handheld flashlight with the Meta Quest 3 headset demonstrated a significant drop in detection accuracy. Figure 5 shows a misclassification and other low-confidence matches observed during practical testing. Poor confidence was largely due to uneven illumination, specular highlights, and sensor noise, highlighting the need for robust preprocessing and lighting adaptation strategies.

Another observation is that, while the system is highly accurate when retrieving component metadata from the database, LLM responses can vary depending on prompt formulation and context concatenation. Maintaining coherent multi-turn interactions remains a challenge, particularly as conversations grow in length or complexity. Additionally, while our current evaluation focused on individual components, real-world scenarios often involve overlapping or partially occluded objects, suggesting the need for more comprehensive testing in cluttered environments. In the future, we will focus on both robustness and functionality improvements. Specific directions include:

- Low-light and challenging environment adaptation: Employing advanced preprocessing techniques or adaptive illumination to improve detection in varied lighting.
- Enhanced LLM-driven interaction: Integrating multimodal LLMs that combine visual and textual inputs, systematically evaluating their responses, and implementing structured conversation memory to maintain context across multi-turn dialogues and contexts.
- Extended application scenarios: Exploring the system's use in other industrial, educational, or maintenance domains and testing under more realistic, cluttered environments with multiple overlapping objects.

By addressing these challenges, the system can develop into a general platform for interactive AR assistance.

A. Conclusion

In summary, this work demonstrates the potential of AR-based maintenance assistants that combine object detection, database retrieval, and interactive AI responses. The framework provides accurate and context-aware guidance under typical conditions and shows strong promise for both educational and industrial applications, with a practical implementation for vehicle inspection and maintenance. According to the proposed server-client architecture, other mixed reality headsets or glasses with see-through displays that can create environmental meshes may also be utilized. Future directions include handling low-lighting, expansion into avionics, and

improving detection, multimodal reasoning, and conversational continuity. With continued refinement, including enhanced environmental adaptability and LLM-driven interaction, this system can serve as a versatile, real-time AR assistant for a wide range of technical tasks, integrating human expertise with AI-assisted collaboration in-situ.

ACKNOWLEDGMENTS

This work was funded in part by the National Science Foundation, grant number 2223035.

REFERENCES

[1] A. Bansal, K. Sikka, G. Sharma, R. Chellappa, and A. Divakaran, "Zero-shot object detection," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 384–400, 2018.

[2] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell, "Few-shot object detection via feature reweighting," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 8420–8429, 2019.

[3] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

[4] T. Ghandi, H. Pourreza, and H. Mahyar, "Deep learning approaches on image captioning: A review," *ACM Computing Surveys*, vol. 56, no. 3, pp. 1–39, 2023.

[5] Y. Srivastava, V. Murali, S. R. Dubey, and S. Mukherjee, "Visual question answering using deep learning: A survey and performance analysis," in *International Conference on Computer Vision and Image Processing*, pp. 75–86, Springer, 2020.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pp. 21–37, Springer, 2016.

[9] X. Gu, T.-Y. Lin, W. Kuo, and Y. Cui, "Open-vocabulary object detection via vision and language knowledge distillation," *arXiv preprint arXiv:2104.13921*, 2021.

[10] A. Gupta, P. Dollar, and R. Girshick, "Lvis: A dataset for large vocabulary instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5356–5364, 2019.

[11] A. Zareian, K. D. Rosa, D. H. Hu, and S.-F. Chang, "Open-vocabulary object detection using captions," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14393–14402, 2021.

[12] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, pp. 8748–8763, PmLR, 2021.

[13] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, *et al.*, "Flamingo: a visual language model for few-shot learning," *Advances in neural information processing systems*, vol. 35, pp. 23716–23736, 2022.

[14] Z. Yang, L. Li, K. Lin, J. Wang, C.-C. Lin, Z. Liu, and L. Wang, "The dawn of lmms: Preliminary explorations with gpt-4v (ision)," *arXiv preprint arXiv:2309.17421*, 2023.

[15] J. Jain, J. Yang, and H. Shi, "Vcoder: Versatile vision encoders for multimodal large language models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27992–28002, 2024.

[16] Y. Zang, W. Li, J. Han, K. Zhou, and C. C. Loy, "Contextual object detection with multimodal large language models," *International Journal of Computer Vision*, vol. 133, no. 2, pp. 825–843, 2025.

[17] D. F. C. Silva, A. R. S. Vitória, and A. R. Galvão Filho, "Deep neural garbage recognition: An augmented reality study case," in *2025 IEEE International Conference on Artificial Intelligence and eXtended and Virtual Reality (AIxVR)*, pp. 425–428, IEEE, 2025.

[18] R. Sapkota, K. I. Roumeliotis, R. H. Cheppally, M. F. Calero, and M. Karkee, "A review of 3d object detection with vision-language models," *arXiv preprint arXiv:2504.18738*, 2025.

[19] B. Chen, Z. Xu, S. Kirmani, B. Ichter, D. Sadigh, L. Guibas, and F. Xia, "Spatialvlm: Endowing vision-language models with spatial reasoning capabilities," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14455–14465, 2024.

[20] M. D. Dogan, E. J. Gonzalez, K. Ahuja, R. Du, A. Colaço, J. Lee, M. Gonzalez-Franco, and D. Kim, "Augmented object intelligence with xr-objects," in *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–15, 2024.

[21] J. H. Israel, O. Belaifa, A. Gispen, and R. Stark, "An object-centric interaction framework for tangible interfaces in virtual environments," in *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*, pp. 325–332, 2010.

[22] A. Taheritajar, J. Benson, A. Gibson, B. Wilburn, J. Zhao, and J. Orlosky, "Scalable object detection in mixed reality using incremental re-training and one-shot 3D annotation," in *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*, ISMAR 2025, (Los Alamitos), pp. 1–11, IEEE, 2025.

[23] Á. Patrício, J. Valente, A. Dehban, I. Cadilha, D. Reis, and R. Ventura, "Ai-powered augmented reality for satellite assembly, integration and test," in *2025 IEEE International Conference on Artificial Intelligence and eXtended and Virtual Reality (AIxVR)*, pp. 1–9, IEEE, 2025.

[24] A. Gordo, J. Almazan, J. Revaud, and D. Larlus, "End-to-end learning of deep visual representations for image retrieval," *International Journal of Computer Vision*, vol. 124, no. 2, pp. 237–254, 2017.

[25] J. Revaud, J. Almazán, R. S. Rezende, and C. R. d. Souza, "Learning with average precision: Training image retrieval with a listwise loss," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5107–5116, 2019.

[26] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.

[27] Z. Jing, Y. Su, and Y. Han, "When large language models meet vector databases: A survey," in *2025 Conference on Artificial Intelligence x Multimedia (AIxMM)*, pp. 7–13, IEEE, 2025.

[28] W. Huang, H. Liu, M. Guo, and N. Z. Gong, "Visual hallucinations of multi-modal large language models," *arXiv preprint arXiv:2402.14683*, 2024.

[29] S. Sharifymoghaddam, S. Upadhyay, W. Chen, and J. Lin, "Unirag: Universal retrieval augmentation for multi-modal large language models," *arXiv preprint arXiv:2405.10311*, 2024.

[30] 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.

[31] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, *et al.*, "Gemini: a family of highly capable multimodal models," *arXiv preprint arXiv:2312.11805*, 2023.

[32] B. Li, Y. Zhang, D. Guo, R. Zhang, F. Li, H. Zhang, K. Zhang, P. Zhang, Y. Li, Z. Liu, *et al.*, "Llava-onevision: Easy visual task transfer," *arXiv preprint arXiv:2408.03326*, 2024.

[33] Y. Zhu, C. Zhang, C. Ré, and L. Fei-Fei, "Building a large-scale multimodal knowledge base system for answering visual queries," *arXiv preprint arXiv:1507.05670*, 2015.

[34] M. Yasunaga, A. Aghajanyan, W. Shi, R. James, J. Leskovec, P. Liang, M. Lewis, L. Zettlemoyer, and W.-t. Yih, "Retrieval-augmented multimodal language modeling," *arXiv preprint arXiv:2211.12561*, 2022.

[35] J. Xue, Q. Deng, F. Yu, Y. Wang, J. Wang, and Y. Li, "Enhanced multimodal rag-llm for accurate visual question answering," *arXiv preprint arXiv:2412.20927*, 2024.

[36] F. Rezaeianaran, R. Shetty, R. Aljundi, D. O. Reino, S. Zhang, and B. Schiele, "Seeking similarities over differences: Similarity-based domain alignment for adaptive object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9204–9213, 2021.

[37] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, 2019.
[38] 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.
[39] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images." Technical report, University of Toronto, 2009.
[40] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729, IEEE, 2008.