

# Episodic Memory Verbalization using Hierarchical Representations of Life-Long Robot Experience

Leonard Bärmann<sup>1</sup>, Chad DeChant<sup>2</sup>, Joana Plewnia<sup>1</sup>, Fabian Peller-Konrad<sup>1</sup>, Daniel Bauer<sup>2</sup>, Tamim Asfour<sup>1</sup> and Alex Waibel<sup>1</sup>

**Abstract**—Verbalization of robot experience, i.e., summarization of and question answering about a robot’s past, is a crucial ability for improving human-robot interaction. Previous works applied rule-based systems or fine-tuned deep models to verbalize short (several-minute-long) streams of episodic data, limiting generalization and transferability. In our work, we apply large pretrained models to tackle this task with zero or few examples, and specifically focus on verbalizing life-long experiences. For this, we derive a tree-like data structure from episodic memory (EM), with lower levels representing raw perception and proprioception data, and higher levels abstracting events to natural language concepts. Given such a hierarchical representation built from the experience stream, we apply a large language model as an agent to interactively search the EM given a user’s query, dynamically expanding (initially collapsed) tree nodes to find the relevant information. The approach keeps computational costs low even when scaling to months of robot experience data. We evaluate our method on simulated household robot data, human egocentric videos, and real-world robot recordings, demonstrating its flexibility and scalability.

Code, data and demo videos at [hierarchical-env.github.io](https://hierarchical-env.github.io).

## I. INTRODUCTION

Verbalizing their own experiences is an important ability robots should have to improve natural and intuitive human-robot interaction [1], [2], [3], [4], [5]. It involves summarization of and question answering (QA) about a robot’s past actions, observations and interactions, such as the dialog shown on the right of Fig. 1. Building a representation of an agent’s Episodic Memory (EM) [6] is crucial to enable such verbalizations, as a system must efficiently store the information from the continuous stream of experience, organize it, and retrieve relevant past events from its EM in response to a user’s query. This is particularly challenging as the time horizon of the EM grows.

Existing work on Episodic Memory Verbalization (EMV) either relies on rule-based verbalization of log files [2], [3], or fine-tuning deep models on hand-crafted or auto-generated datasets [1], [7] to perform QA and summarization tasks given the recorded experiences. Both approaches are limited,

This work has been supported by the Baden-Württemberg Ministry of Science, Research and the Arts (MWK) as part of the state’s “digital@bw” digitization strategy in the context of the Real-World Lab “Robotics AI”, the Carl Zeiss Foundation through the JuBot project and the German Federal Ministry of Research, Technology and Space (BMFTR) under the Robotics Institute Germany (RIG).

<sup>1</sup>Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Germany E-mails: {baermann, asfour}@kit.edu

<sup>2</sup>Computer Science Department, Columbia University, NY, United States E-mail: chad.dechant@columbia.edu

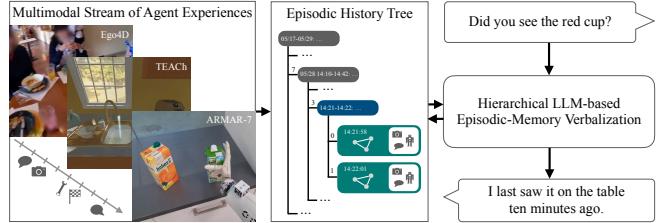


Fig. 1. Our system answers queries about life-long experience of an agent (human or robotic) by exploring a tree representation of episodic memory.

as they require designing vast numbers of rules or collecting large amounts of experience data.

To avoid training a system, which typically entails collecting large amounts of multimodal experience data, previous works [8], [9] use language-based representations of the past which can be obtained from pretrained multimodal models. Given such a language-based representation of an agent’s history of episodic events, one practical way to perform QA is to pass the question and the history to a large language model (LLM) and prompt it to produce an answer. While this works nicely for short histories [8], [10], in this paper, we focus on how to scale such approaches for verbalization of *life-long* experience streams. Although recent LLMs offer increasingly long context windows (i.e., the maximum number of tokens they can process), up to 2M tokens [11], previous studies [12], [13] have shown that these models have difficulty in using all information contained in such long contexts. Furthermore, the computation of transformer models scales quadratically with context length – reducing the number of tokens is thus time- and cost-effective.

Therefore, to scale EMV to life-long experience streams while maintaining a low token budget, we propose to derive a tree-like representation from EM and use an LLM agent for QA to interactively search the tree to find relevant information. Our system, H-EMV (Hierarchical Episodic Memory Verbalization, Fig. 1), processes the continuous stream of experiences and inserts it into a hierarchical representation of the robot’s history of episodic events. Different levels of this hierarchy represent different abstraction levels, with the lowest level being raw observations and proprioception and higher levels being represented as natural language concepts. An LLM is prompted for segmentation and summarization in order to recursively create higher-level abstractions. To process queries to the EM, we repurpose the interactive prompting scheme described in our previous work [14]. An

LLM is provided with the user’s query and functions to access the history tree, and the LLM itself decides which functions to use in order to fulfill the query, i.e., to answer the question or provide a summary. Since the history tree will grow large over time, we apply an interactive semantic search, inspired by work on robot navigation [15], [16]. Specifically, we contract the tree, i.e., the LLM initially only sees the top-level node, and then interactively explores the graph to retrieve the information relevant to the query.

To evaluate our system, we use simulated household episodes from TEACh [17], real-world egocentric human recordings from Ego4D [18], and real-world robot episodes from ARMAR-7, the newest member of the ARMAR humanoid robot family developed at KIT [19]. Our experiments show that H-EMV efficiently scales to extremely long histories of multiple simulated months or real-world human egocentric videos of over six hours, outperforming several baselines and ablations. Real-world robot demonstrations showcase the wide applicability of our system. We provide our code, evaluation data, and demonstration videos at [hierarchical-env.github.io](https://hierarchical-env.github.io).

## II. RELATED WORK

**Episodic Memory for Robots:** The concept of EM stems from human cognition [6] and is useful for various technologies including smart wearables [18], [20], smart rooms [21], and especially robotics. For instance, robotic EM can be represented using latent vectors created by deep neural models [1], [22], [23], or by explicitly storing relevant information in a memory system [24], [25], [26]. Another approach is to represent the history of episodic events as text produced by pretrained models [8], [9]. In this paper, we also represent the history tree in form of text, following REFLECT [9] for the broad structure of the hierarchy’s lower levels. However, we extend this by adding hierarchical summarization. Furthermore, our multimodal episodic history tree can be dynamically explored by an LLM to gather information from all levels, including the raw observations.

**Robot Experience Verbalization:** The first work to introduce the term of “verbalizing” robot experiences was [2]. With a rule-based system, they converted a navigation route taken by a mobile service robot to natural language. [3] adapted this framework to verbalization of manipulation activities performed by a humanoid household robot. Similarly, [27] use templates to convert their robot’s observations and actions to natural language. More recent works phrase EMV in a more interactive setting, defined as summarization and QA on robot experiences [4]. Both [1], [7] propose end-to-end trained networks receiving multimodal experiences and a question to produce an answer, using training data from simulated household tasks. While [7] work on visual data only, [1] additionally use symbolic and subsymbolic information from the robot’s task execution and perception components. In contrast to these systems, H-EMV uses pretrained foundation models and does not require additional training data, thus increasing its versatility and easing deployment to the real world. Most similar to our work,

ReMEmbR [28] propose to use a robot’s experience for QA and navigation. Using an LLM as an agent to decide what data to retrieve from a vector store of memories, they can answer queries based on time, place, and content. The “search” function available to H-EMV’s LLM agent works similar to this but acts on a specific tree node instead of the overall history. Further, ReMEmbR’s memory is not hierarchical, thus limiting both temporal and content-based retrieval to short moments in time, struggling to handle queries that require larger context (such as “summarize your day”). Focusing on failure identification and recovery, [29] produce robot narrations from multimodal robotic data streams. This can be seen as orthogonal to our work as these narrations could enrich the lower levels of our hierarchy. Similar to our setting, QA from streaming data [30], [31] tackles the problem of answering questions based on a long stream of data where the question is not known in advance and the raw data cannot be stored. However, we apply this to robotics, and approach it with an interpretable, modular system instead of end-to-end trained memory models. Furthermore, some works in natural language processing propose hierarchical variants of retrieval-augmented generation to tackle multi-document QA tasks [32], [33], [34], which is similar to our work in that a graph-like structure is accessed, but differs as we use an LLM as an agent and apply it on inherently temporal, multimodal robotics data.

**Video Understanding:** Video Understanding, especially Video Question Answering (VideoQA), is related to EMV as it also involves QA on a data stream, which, however, is only a video instead of a multimodal robotic experience stream. VideoQA is an active research area [35] where current major challenges include long-form videos beyond clips of a few seconds as well as egocentric video understanding. Ego4D [18] is a large collection of unconstrained egocentric videos showing daily activities of human camera wearers. Ego4D GoalStep [36] and HCap [10] provide hierarchical annotations for subsets of Ego4D, facilitating reasoning on different abstraction levels. Recent long-form egocentric VideoQA benchmarks include QAEgo4D [20] and EgoSchema [37].

Recent methods for VideoQA can be grouped into (i) end-to-end approaches [38], [39], [40], [10], [41] that typically connect pretrained frozen visual encoders with LLMs by some trained adapter, and (ii) training-free “socratic” [8] approaches [42], [43], [44], [45], [46], [47], [48] that invoke various off-the-shelf models to convert the video into text to be processed by a few-/zero-shot prompted LLM. For instance, [46], [48] use video captioning to produce a transcript of the video and then apply an LLM for QA based on this transcript. VideoTree [43] adaptively selects the frames to caption using a top-down query-relevance-based tree expansion instead of uniform sampling. Both [49], [50] build structured representations from egocentric video using various models, e.g., for object and hand tracking, as well as activity and location labeling. [47] generate executable Python code from a question, invoking different APIs to query visual and language foundation models. MoReVQA [45] decomposes this into multiple stages, making the LLM’s

job easier at each stage by focusing on either event parsing, grounding, or reasoning, instead of all at once. In contrast to these predefined prompting schemes, both [42], [44] use an LLM as an agent to analyze the video content in an interactive loop. While [44] iteratively ask the LLM whether to gather more detailed information (by captioning more intermediate frames) or produce the final answer, [42] provide the LLM with API functions invoking tools to search in a database of tracked objects or a memory of frame captions.

Our method similarly treats the LLM as an agent, thus not relying on any predefined information flow. However, we use the full flexibility of code [51] instead of single API calls like in [42], [44]. Compared to VideoTree [43], our history tree is constructed independently of the user’s query, since future questions cannot be known in advance in realistic settings, and storing lifelong “raw” video experiences is prohibitive [20]. In contrast to all of the above works, we consider real-world dates and times an integral part of the process. While the recent work TimeChat [52] is also time-sensitive, they refer to video timestamps (“second 42 to 57”) instead of real-world date-times (“yesterday in the afternoon”). Furthermore, and most crucially, we deal with long sequences of multimodal *robotic* experiences, with the longest experiment having over six hours of video or nearly two months on a simulated timeline.

### III. METHOD

Our goal is to enable an artificial agent to verbalize and answer questions about its past. Given the continuous, multimodal stream of experiences of a robot agent, we build up a hierarchical and interpretable representation of EM (Sec. III-A). When a user later asks a question, an LLM interactively explores the history tree to gather relevant information, detailed in Sec. III-B.

#### A. Episodic Memory Construction

From a stream of multimodal robot experiences, we derive a hierarchical representation of the robot’s EM, a *history tree*, as shown in Fig. 2, with the lower levels broadly following [9]. Specifically, the tree’s levels are:

**L0 – Raw Experiences:** Leaf nodes collect the raw information available at a specific timestep during the robot’s task execution. This includes all modalities that can be perceived by the robot: RGB and depth camera images and recorded audio, as well as information deduced from this data, i.e., recognized objects, their positions, and a text transcription of the audio, if there is user speech. Furthermore, we include everything the agent knows about its state: robot proprioception (joint configuration, mobile platform position), symbolic information about the current action and goal, and text to be spoken by the robot’s text-to-speech component.

**L1 – Scene Graphs:** The first level of non-leaf nodes in the history tree has a one-to-one mapping to the L0 leafs. On this level, we derive a scene graph from the given observations, consisting of the detected objects as nodes and their spatial relations (e.g., on top, inside) as edges. The

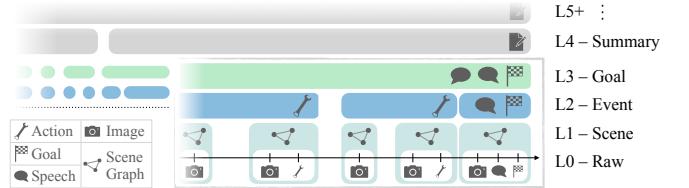


Fig. 2. From the continuous, multimodal stream of robotic experiences, we construct a *history tree*, a hierarchical representation of the EM.

exact method for constructing the scene graph varies in our experiments. For the pure vision-based approach, objects are detected using pretrained models and heuristics are applied to infer semantically meaningful relations [9], whereas in our real-robot experiments, we use the existing components in our robot’s software framework ArmarX [53] that already provide semantic scene information.

**L2 – Events:** Next, we group and summarize the nodes from the previous level based on changes in the scene graph, the currently executed action or goal, as well as when there is a new speech recognition. We also create a template-based natural-language summary, including the latest scene graph, the current action, and recognized speech command. In our real-robot experiments, we use an LLM to filter and summarize the raw action parameters, which would be excessively detailed otherwise.

**L3 – Goals:** Based on the current goal from the L0 node, we group event nodes and again create a rule-based natural-language summary containing the current goal and the verbalization of the latest event. Note that we allow goal nodes to have children of mixed types: either events or other goal nodes. This allows representing subgoals of complex tasks and is used in our real-robot experiments.

**L4+ – Higher-Level Summaries:** Summaries are generated dynamically by recursively asking an LLM to summarize the previous level’s nodes. Specifically, given the set of nodes  $S_\ell$  at level  $\ell \geq 3$ , we list them in order and prompt an LLM to identify consecutive ranges of items that belong together considering their times and content, and provide a summary for each range. The output is parsed to group the child nodes and create the summary nodes  $S_{\ell+1}$  for the next level. We apply this strategy recursively, until  $|S_\ell| = 1$  or there is no further reduction, i.e.,  $|S_{\ell+1}| = |S_\ell|$ . In the latter, the LLM is explicitly prompted to provide a single concise summary of all items to force obtaining one root node. To make this procedure robust to LLM errors, e.g., invalid output syntax, missing an item or including one in multiple ranges, we re-prompt on failure, listing detailed error messages and asking for improvement. In case the LLM fails to provide a valid grouping after three trials, we fall back to the single summary prompting.

#### B. Episodic Memory Access

Given a user’s query and the history tree built from all experiences so far, we use an LLM as an agent [54] to explore the tree, search relevant information, and eventually answer the question. For this, we define an API to interact

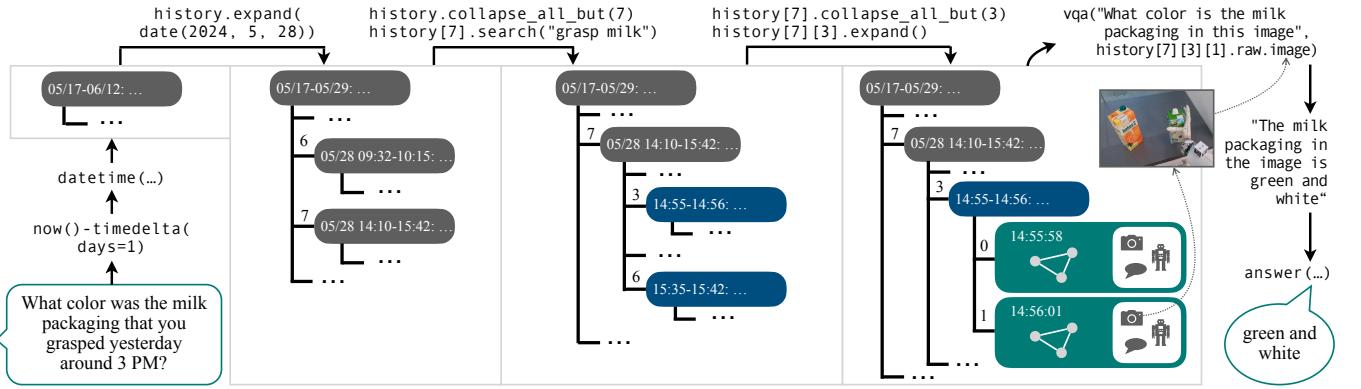


Fig. 3. To answer a user’s question, H-EMV prompts an LLM to interactively explore the history tree containing the agent’s experiences. The LLM can further invoke tools (index search, VLM) or perform other calculations to gather relevant information, eventually invoking the `answer` function. This figure shows an example from our real-world evaluation on the humanoid robot ARMAR-7, modified for illustrative purposes.

with the history tree. We initially define each node of the tree to be in a collapsed state, i.e., its textual representation will only contain the node’s time range and natural-language summary, but not list the child nodes. The LLM can then interactively expand and collapse nodes, according to what seems relevant given the user’s query. Furthermore, we provide different tools to the LLM, e.g., to invoke a Vision-Language-Model (VLM) to perform visual QA on the images associated with leaf nodes. Moreover, there is a function to perform tree search based on semantic similarity, selectively expanding the children of the searched node in the tree that match the search query.

Fig. 3 illustrates typical steps the LLM performs to answer a user’s question. Given the initially collapsed tree, the LLM first expands the root node’s children based on the requested date. It then selectively explores the respective child nodes that seem relevant to the question using the search function. Note that the LLM is prompted to collapse irrelevant nodes again in order to limit the number of tokens used and speed up further requests. In the given example, when reaching a leaf node, the answer to the question is not evident from any of the natural-language summaries on each level so the LLM decides to invoke a VLM to gather more information. Finally, it invokes the `answer` function to answer the user’s question.

Our implementation of the LLM agent uses a prompting style inspired by the simulated Python console of [14]. The LLM can issue any command – including compound statements such as loops – using the provided API. After the execution of the respective code, the LLM can “see” the output of its command(s), or any execution error. This process is repeated, and the prompt to the LLM always contains the (growing) execution history. Our API returns detailed error messages and usage hints in case the LLM uses it in a wrong way, and we also detect repetitive generations, prompting the LLM to try to solve the problem differently. In case an answer is not obtained after a set amount of maximum iterations, we use a special prompt to force an immediate answer given the current state of the history tree. Zero-shot experiments prompt the LLM with only a

static prefix to explain the task and the available API, while few-shot prompting adds top- $k$  examples selected based on semantic similarity to the current user query. The final part of the prompt is always a string representation of the history tree’s current state.

#### IV. EVALUATION

##### A. Simulated Household Episodes

Following our previous work [7], [1], we use simulated household episodes and automatically annotate them with QA pairs based on the ground-truth (GT) simulation state. Specifically, we use the TEACH dataset [17], featuring episodes of two real humans, one commander, and one follower, interacting with the AI2THOR environment [55]. We adapt this data by rephrasing the commander to be a human user, and the follower to be a robot interacting with the environment (and the user). Thus, each TEACH episode describes a robot experience, comprising egocentric images, robot states and actions, and dialog with the user.

**Data:** Episodes in TEACH are on average  $6.2 \pm 5.3$  min long. Since we are interested in very long histories of robot experience, we randomly combine them to form histories of up to 100 episodes. We also randomize dates and times for each episode, ensuring realistic sequences by picking one to five episodes per day, avoiding nighttimes, and occasionally skipping some days; the longest histories thus span nearly two months. Based on these histories, we generate QA pairs by adjusting the generation grammar from [7]. Specifically, we generate ten types of questions. These ask for: a list of high-level summaries of episodes (task descriptions); a detailed description of one particular episode in a history; a summary of an episode that happened either before or after a particular episode; a list of episodes in which a particular object was seen or action performed; a summary of an episode that occurred at a given time or a specified number of days ago; a list of times or number of days ago at which a given task was performed. Questions can refer to specific episodes either by date and time, or by the task performed (if it is unique). From the TEACH “valid unseen” set, we generate test sets with 10 histories per sequence

length (combining  $|h| = 5, 15, 25, 50$ , and 100 episodes). Each history  $h$  is annotated with ten QA pairs, making up 100 samples per history length  $|h|$ .

**Evaluation Metrics:** Evaluation of free-form EMV answers is hard since there can be many ways to formulate the correct answer, questions can be underspecified, and verifying abstract statements by grounding them in the history tree is a research question in itself. Following [1], we define a semantic categorization of a model’s output  $o$  given the GT  $g$  and question  $q$ : *correct* when  $o$  is semantically equivalent to  $g$ ; *correctly summarized* if  $o$  is a correctly summarized version of  $g$ , still containing all relevant facts (in context of  $q$ ); *correct TMI* (too much information) if  $o$  is correct but overly specific; *partially correct TMI* if parts of  $o$  are correct, but there are TMI parts and these are wrong; *partially correct missing* if parts of  $o$  are correct, but relevant facts from  $g$  are missing; *wrong* when  $o$  could be an answer to  $q$  but is none of the above; and *no answer* if  $o$  is empty, completely irrelevant to  $q$ , or the model threw an error. Counting *correct* (incl. correct summarized, correct TMI) samples  $C$  and *at least partially correct* samples  $P$  ( $C \subseteq P$ ), we report their percentages  $S_c$  and  $S_p$ , respectively. Since categorizing each evaluated sample by hand is prohibitively expensive, we prompt GPT-4o [56] to perform this evaluation. For this, we started by annotating 60 samples by hand, and use these as a database to retrieve few-shot samples based on maximal marginal relevance [57]. We further tuned the prompts on a validation set of 100 hand-categorized model outputs (generated from TEACH train).

We evaluate the agreement of the LLM’s predicted categories with manually annotated ones on 200 model results from our test data, resulting in an aggregated category accuracy of 88%, and per-class f-scores of  $F_1(\text{correct}) = 0.89$ ,  $F_1(\text{partially\_correct}) = 0.84$ ,  $F_1(\text{wrong}) = 0.91$ . The LLM categorizes correct and wrong samples very well and has the most difficulties on the *partially correct* labels. However, these categories are also defined imprecisely, and the inter-annotator agreement [58] between the first two authors has only a value of Cohen’s  $\kappa = 0.66$  ( $n = 110$ ), vs.  $\kappa = 0.91$  ( $n = 68$ ) when only considering correct/wrong. Thus, while not perfect, we use the LLM to automatically obtain reasonable score estimates. We do not report surface-level automated metrics such as BLEU [59] or ROUGE [60] as these metrics struggle to capture the variety of correct answers in the EMV task, e.g., for a “when”-question, both of the following answers are correct, but have no word overlap at all: “at 4 PM” vs. “in the afternoon”.

**Settings and Baselines:** We evaluate under three settings: First, vision-only can act solely on the visual data stream. As a baseline, we prompt Gemini 1.5 Pro [11] in one pass with the sequence of images along with timestamps and the question (*img 1-pass*). We choose Gemini because we require extremely long requests; however, despite its 2M token limit, we need to sample every 2nd frame so that longer histories fit into the context window. We few-shot-prompt with one static example history of five episodes including ten QA samples for this history. For a fair comparison, we use the same LLM

for ReMEmbR and H-EMV (although we can achieve better results with GPT-4o, as shown in Sec. IV-C). H-EMV does not take raw images, but constructs history trees by inferring objects using YOLO-World [61] and actions using a LongT5 transformer model [62] fine-tuned on TEACH train. This vision model limits our system with an action classification accuracy of 62% (which can be partially circumvented by the summarization LLM’s commonsense knowledge). Few-shot prompting Gemini to infer the sequence of actions for building the tree had a worse accuracy of around 27%. We also compare with one-pass zero-shot prompting the LLM with this tree limited to L3 node (i.e., no hierarchical summarization) in text form (*text 1-pass*). Second, vision + speech enriches the visual information with the dialog data from TEACH episodes, representing natural language commands given to the robot. This is simply added to the prompt for *img 1-pass*, and inserted into the history tree for H-EMV and *text 1-pass*. Finally, full multimodal uses the recorded (GT) actions and goals from the TEACH episodes, as this information is typically available when a robot executes some actions. This setting also uses GT object information to compare system performance assuming perfect vision components. We compare H-EMV with one-shot and zero-shot prompting of the LLM agent. For preparing the few-shot samples, we use histories built from episodes in TEACH train, and record traces of manually using the Python console interface and the defined API to interact with the history tree until the given GT answer becomes evident. While we collect two to three samples per question type this way, making up 21 samples in total, we select only the top-1 sample when prompting the LLM, based on semantic similarity of the user’s questions. Semantic similarity is determined after asking `gpt-4o-mini` to cross out the task-specific words from the question so that an example from the same question type is retrieved (instead of an irrelevant example just mentioning the same objects or activities). We further ablate the hierarchical summarization: H-EMV *0-shot L3* is our method without LLM-generated summaries (L4+), still using the interactive agent to explore an initially collapsed list of L3 nodes. Last, *text 1-pass 0-shot L3* is a baseline presenting the fully expanded tree (L3 and lower) to the LLM along with the question in a single prompt, thus not using the LLM as an agent. We also compare to ReMEmbR [28], with their method/prompting minimally adjusted to our setting.

**Results:** Results of our TEACH experiments can be found in Table I. First, we can observe that every method’s performance decreases with increasing  $|h|$ .

Focusing on the vision-only and vision + speech results, the Gemini 1-pass baseline outperforms H-EMV for shorter histories. This is reasonable, as the baseline can directly access the full stream of visual information, whereas our hierarchical system suffers from error propagation and is limited by pretrained vision components. In particular, the history tree could contain incomplete or wrong information or our method could fail by expanding the wrong nodes of the tree, which cannot happen to the 1-pass baseline. However, token costs scale linearly with history length for

TABLE I

RESULTS ON SIMULATED HOUSEHOLD EPISODES FROM TEACH [17]

method	ICL	Hier.	$ h  = 5$			$ h  = 15$			$ h  = 25$			$ h  = 50$			$ h  = 100$		
			$S_c$	$S_p$	T	$S_c$	$S_p$	T	$S_c$	$S_p$	T	$S_c$	$S_p$	T	$S_c$	$S_p$	T
vision-only																	
img 1-pass	FS	flat	24	49	164	13	43	334	17	41	406	11	35	512	10	30	1256
text 1-pass	0	L3	5	18	65.4	8	15	336	5	15	270	5	25	789	6	14	1234
H-EMV	1	full	9	40	10.6	9	37	10.2	10	42	10.2	15	36	11.6	15	35	11.9
vision + speech																	
img 1-pass	FS	flat	57	83	165	30	73	336	39	69	418	25	60	861	17	54	1591
text 1-pass	0	L3	21	43	68.9	19	43	217	22	50	287	15	41	740	7	28	1304
H-EMV	1	full	28	64	9.2	33	63	9.9	31	63	9.7	27	57	10.9	19	51	13.3
full multimodal (objects + speech + actions)																	
ReMEmbR			10	30	12.2	6	21	12.5	11	28	12.5	7	22	12.6	5	18	12.7
text 1-pass	0	L3	32	65	120	37	63	372	35	69	422	24	56	1055	OOC		
H-EMV	0	L3	28	61	19.7	20	62	44.2	29	65	19.8	16	64	116	21	52	249
H-EMV	0	full	51	71	3.1	45	76	4.1	46	73	15.0	32	58	4.9	25	60	5.6
H-EMV	1	full	48	74	8.5	48	72	9.6	55	74	23.7	37	68	10.2	34	62	10.4

$S_c, S_p$ : semantically categorized (partially) correct in %, T: number of 1K prompt token.

OOC: out of context, ICL: in-context learning, FS: few-shot

1-pass, while it stays approximately constant for H-EMV. The performance also drops faster for 1-pass, with H-EMV reaching comparable or better semantic scores for  $|h| \geq 25$ . In contrast, when circumventing the limitations of perception components by using GT object detection and action information (full multimodal setting), H-EMV outperforms the 1-pass system with a token budget two orders of magnitudes smaller (see Fig. 4). Further, 1-shot prompting helps for longer histories, but 0-shot works with half the token costs and comparable performance for short histories. Ablating the hierarchical higher-level summaries notably increases token cost and leads to worse performance, especially for longer histories (when the LLM sometimes expands all nodes). While ReMEmbR keeps token cost low due to its use of the LLM as an agent similar to H-EMV, its retrieval-augmented generation approach and lack of summarization capability has difficulties with our QA data, as it typically retrieves single actions (“I picked up a mug”) instead of higher-level task information (“I prepared breakfast”).

### B. Egocentric Human Videos

In addition to verbalizing robot experience, EMV can be applied to human egocentric recordings, e.g., in the context of smart wearables. Here, the system does not summarize and answer questions about its own actions, but the actions of its user.

**Data:** To evaluate our system under this setting, we use Ego4D [18]. Randomly concatenating episodes (as done above for TEACH) generates histories that are not cohesive, thus restricting automatic summarization to bare enumeration instead of abstraction of related events. Therefore, we perform a small-scale evaluation on very long recordings from Ego4D. Specifically, we manually select two very long Ego4D videos (6:43h and 4:28h) showing diverse and interesting actions in a tourist scenario. Additionally, we construct one history by concatenating shorter episodes from similar scenarios, selected to ensure some level of cohesiveness and plausibility (in contrast to random sequences). We manually write 40 challenging QA samples.

**Method:** To construct history trees from Ego4D videos, we apply VideoReCap [10] which produces low-level narrations

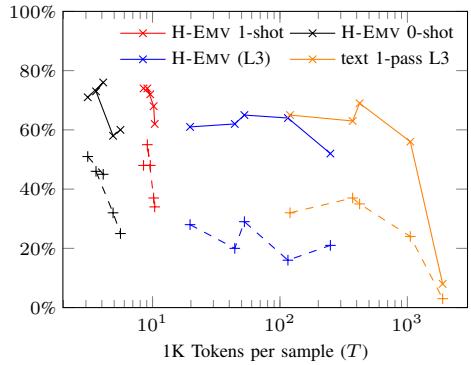


Fig. 4. Token costs vs. performance for different history lengths (TEACH full multimodal). Solid lines are  $S_p$ , dashed lines  $S_c$ . H-EMV retains better performance at lower costs.

at 1 fps and mid-level summaries for each minute. We map these to action (L2) and goal (L3) nodes of our hierarchy, respectively, converting texts to first-person perspective using meta-llama3-8b [63]. For constructing higher-level (L4+) summaries, we generated few-shot samples for the group-and-summarize LLM (see Sec. III-A) using Ego4D-HCap [10]. To populate the L1 scene graph with objects, we apply YOLO-World [61], an open-vocabulary object detection approach, which we prompt with classes obtained through a Socratic Models [8] approach: First, we select the top-100 classes according to cosine similarity of the mean CLIP [64] image embedding within one L3 node and the CLIP text embeddings of all LVIS [65] labels. Further, we prompt meta-llama3-8b to propose  $\approx 20$  objects that might occur given the current L3 goal annotation produced by VideoReCap. We then apply YOLO-World with the combined set of classes on each image within this L3 node and store the detected objects in the respective L1 nodes. The EMV agent for QA is instantiated zero-shot.

**Results:** The results of applying our method and manually categorizing the results (following Sec. IV-A) can be seen on the left of Table II. Due to the very challenging nature of our QA samples and the limitations of the used vision components, the overall performance is low. Low performance of the 1-pass baseline can partially be explained by it seeing only a flat version of the L2 events without access to the images. However, it also fails on most of the samples that could be answered from the text history. This may be explained by the noisy text history inferred from vision, which H-EMV can handle better because of hierarchical summarization and selective expansion of nodes, whereas the 1-pass baseline observes all the noise at once. We did not directly apply the 1-pass baseline on the images (which would be possible only with aggressive sub-sampling) for cost reasons. H-EMV L3 performs slightly worse than H-EMV, with double the costs.

### C. Real-World Robot Recordings

Finally, we apply our method on our real-world humanoid robot ARMAR-7. To obtain an EM, we record multiple robot sessions of typical household tasks, spanning a total duration

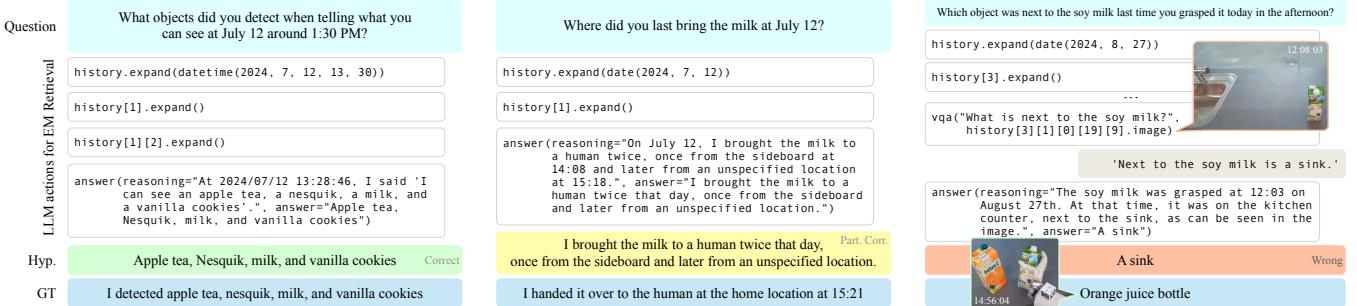


Fig. 5. Example traces from our ARMAR-7 evaluation. *Left*: success case. *Middle*: partially correct, LLM misinterpreting the question. *Right* (shortened): wrong answer, expanded the wrong node (not the *last* time grasping the milk). More examples at [hierarchical-env.github.io](https://hierarchical-env.github.io)

TABLE II  
EXPERIMENTS ON HUMAN & ROBOTIC REAL-WORLD DATA (0-SHOT)

Method	Hierarchy	LLM	Ego4D			ARMAR-7		
			$S_c$	$S_p$	T	$S_c$	$S_p$	T
1-pass	flat	Gemini	5	13	438	43	53	227
H-EMV	L3	Gemini	18	38	33.1	40	70	29.2
H-EMV	full	Gemini	30	40	14.4	33	73	10.5
H-EMV	L3	GPT-4o	25	43	57.6	30	47	68.4
H-EMV	full	GPT-4o	28	53	21.9	43	70	12.8

of 3.3 hours of robot actions over the scope of two months. We record all entries made to the memory system introduced in [24], in particular: vision (RGB and depth images), robot state (proprioception), skill events (executed actions and goals), speech (speech-to-text output and text-to-speech input), symbolic scene (objects and their relations). From such recordings, we build up a history tree by populating L0 with images, speech, and proprioception, L1 scene graphs with the symbolic scene information, L2 and L3 with robot action events (where L2 contains low-level actions and L3 contains actions that themselves invoke other actions). Note that L3 nodes can be nested in this case (goals and their subgoals). Higher levels (L4+) are constructed dynamically by an LLM as described in Sec. III-A, with two manually created few-shot samples.

Subsequently, we annotate the recordings with 30 QA-pairs, apply our method, and again manually categorize the results. Results can be seen on the right part of Table II, with examples in Fig. 5. In general, our task is very challenging, and the 1-pass Gemini baseline which has direct access to the complete stream of episodic data (without images) scores only 43%/53% of correct/partially correct samples. Compared to the Ego4D experiment, the quality of the text history is better, as most content (esp. current action, goal) is not inferred from vision. Our interactive hierarchical system achieves better  $S_p$  and slightly worse  $S_c$ , with 1/21 of the token costs. While the experiments with Gemini and limited tree depth (L3) perform slightly better, the best results are achieved by the full system with GPT-4o. Here, L3 has lower performance with more than five times the token cost. See the supplementary video for a demonstration of our system in action, enabling ARMAR-7 to answer questions about its past interactively. We also applied H-EMV (with no further dataset-specific prompt tuning) on NaVQA [28], but obtained

an accuracy of only 32% on the descriptive questions, which we attribute to their dataset being focused on very specific details, whereas our method strives at tasks that require broader context and summarization capabilities.

#### D. Failure analysis

To better understand the potential for future improvement of our system, we analyze its failure modes. We broadly categorize failures by three sources: EM construction, EM retrieval, and LLM QA reasoning. For the simulated TEACH evaluation, full multimodal setting, both the generated QA and the history tree are derived from the simulation state, therefore the requested information is guaranteed to be contained in the tree, eliminating the EM construction failure reason. During QA generation, we also annotate the target episode this question asks for, and then evaluate whether our model correctly expands the history tree node representing this episode. Specifically, we calculate retrieval precision  $p$  and recall  $r$  as follows: Given the final state of the tree when the LLM decides to answer the question, we extract the set of visible (expanded) nodes  $V$ . From these, we pick the lowest-level visible nodes:  $L = \{v \in V \mid \text{children}(v) \cap V = \emptyset\}$ . Every node  $v$  has an associated time span  $t_v = (s_v, e_v)$  with start and end datetime  $s_v, e_v$ . Given a reference time span  $t_g = (s_g, e_g)$  that the question refers to (one question may refer to multiple time spans, e.g., “what did you do before...”), we calculate the length of the intersection  $i_{g,v} = |t_g \cap t_v|$ . This intersection is normalized relative to the duration of the node and the baseline intersection  $b$  of the root node  $v_r$  with the reference, defined as  $b = \frac{|t_g|}{|t_{v_r}|}$ . The normalized intersection is given by  $n_{g,v} = \max(0, \frac{i_{g,v}}{|t_v|} - b) \cdot \frac{1}{1-b}$ . Recall is defined as the average of the maximum normalized intersection rate per reference time span:  $r = \text{avg}_{t_g \in G}(\max_{v \in V} n_{g,v})$ . Precision is calculated by taking the maximum normalized intersection rate for each leaf  $l \in L$  over all reference spans, and then averaging across all leaves:  $p = \text{avg}_{l \in L}(\max_{t_g \in G} n_{g,l})$ . Eventually,  $F_1 = \frac{p \cdot r}{p + r}$ .

Fig. 6 shows the retrieval  $F_1$  of H-EMV for different history lengths. Evidently, the retrieval process is a major source of problems for the EMV task: When taking into account only samples that were answered (partially) correct, the retrieval performance is notably better than for the samples that were answered wrong, underlining that correct

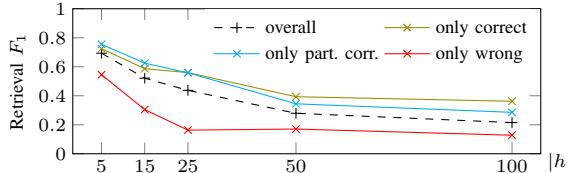


Fig. 6. Retrieval  $F_1$  score by history length  $|h|$  (TEACH H-EMV 1-shot)

retrieval is essential for overall QA performance. The non-zero retrieval performance for wrong samples stems from partial retrieval (not expanded deep enough) as well as other failures due to QA reasoning (despite correct retrieval).

For our real-world (Ego4D and ARMAR-7) experiments, we manually inspect the execution traces of H-EMV (Gemini) for each (partially) wrong sample and attribute it with one or multiple of the above failure reasons. As for the simulated evaluation, the majority of failures is related to retrieval (18 of 28 failure samples for Ego4D, 19/20 for ARMAR-7). Especially for the real-robot experiments, the tree is very deeply nested, and retrieval therefore challenging. On the other hand, tree building issues are more prevalent for Ego4D (13/28), as the system is relying purely on visual information there. However, issues can also occur on ARMAR-7 (7/20), for instance when the LLM-generated summaries are misleading or too broad (“performed various actions”). Further, both approaches sometimes struggle with reasoning or correctly using the tree exploration API (e.g., getting stuck in repetitive behavior), which is most likely related to the difficulty of the questions and the general LLM capabilities.

## V. CONCLUSION & DISCUSSION

We present H-EMV, a system for verbalization of life-long robot experience. The multimodal, hierarchical representation of EM is interactively accessed by an LLM to answer user questions, keeping token costs low even for extremely long histories. Despite the promising results and versatility of our system, it has some limitations: First, as a modulated approach, it is limited by the performance of each component and can suffer from error propagation. While the interactive tree search improves interpretability, there are no performance guarantees. Moreover, our system could integrate more modalities and tools, e.g., joint angle proprioception data could be rendered in simulation and then verbalized by a VLM. Episodic memories should refer back to previous events to enable contextual or comparative descriptions [66]. Adding personalization, both to EM and verbalization, is desirable for improved human-robot interactions. We hope our code and data will foster research on EMV, and will continue addressing these challenges in future work.

## REFERENCES

- [1] L. Bärmann, F. Peller-Konrad, S. Constantin, T. Asfour, and A. Waibel, “Deep Episodic Memory for Verbalization of Robot Experience,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 6, no. 3, pp. 5808–5815, 2021. Cited on pages 1, 2, 4, and 5.
- [2] S. Rosenthal, S. P. Selvaraj, and M. Veloso, “Verbalization: Narration of Autonomous Robot Experience,” in *International Joint Conferences on Artificial Intelligence by AAAI*, 2016, pp. 862–868. Cited on pages 1 and 2.
- [3] Q. Zhu, V. Perera, M. Wächter, T. Asfour, and M. M. Veloso, “Autonomous narration of humanoid robot kitchen task experience,” in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, 2017, pp. 390–397. Cited on pages 1 and 2.
- [4] C. DeChant and D. Bauer, “Toward robots that learn to summarize their actions in natural language: a set of tasks,” in *Conference on Robot Learning (CoRL)*, 2021. Cited on pages 1 and 2.
- [5] K. Katuwandeniya, L. Tian, and D. Kulic, ““What did the Robot do in my Absence?” Video Foundation Models to Enhance Intermittent Supervision,” *IEEE Robotics and Automation Letters (RA-L)*, pp. 1–8, 2025. Cited on page 1.
- [6] E. Tulving, “Episodic and semantic memory,” in *Organization of Memory*, E. Tulving and W. Donaldson, Eds. Cambridge, MA: Academic Press, 1972, vol. 1, pp. 381–403. Cited on pages 1 and 2.
- [7] C. DeChant, I. Akinola, and D. Bauer, “Learning to summarize and answer questions about a virtual robot’s past actions,” *Autonomous Robots*, 2023. Cited on pages 1, 2, and 4.
- [8] A. Zeng, M. Attarian, B. Ichter, K. M. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. S. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, and P. Florence, “Socratic Models: Composing Zero-Shot Multimodal Reasoning with Language,” in *International Conference on Learning Representations (ICLR)*, 2023. Cited on pages 1, 2, and 6.
- [9] Z. Liu, A. Bahety, and S. Song, “REFLECT: Summarizing Robot Experiences for Failure Explanation and Correction,” in *Conference on Robot Learning (CoRL)*, 2023. Cited on pages 1, 2, and 3.
- [10] M. M. Islam, N. Ho, X. Yang, T. Nagarajan, L. Torresani, and G. Bertasius, “Video ReCap: Recursive Captioning of Hour-Long Videos,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 18 198–18 208. Cited on pages 1, 2, and 6.
- [11] Gemini Team, “Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context,” 2024. Cited on pages 1 and 5.
- [12] C.-P. Hsieh, S. Sun, S. Kriman, S. Acharya, D. Rekesh, F. Jia, Y. Zhang, and B. Ginsburg, “RULER: What’s the Real Context Size of Your Long-Context Language Models?” 2024. Cited on page 1.
- [13] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, “Lost in the Middle: How Language Models Use Long Contexts,” *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 157–173, 2024. Cited on page 1.
- [14] L. Bärmann, R. Kartmann, F. Peller-Konrad, J. Niehues, A. Waibel, and T. Asfour, “Incremental Learning of Humanoid Robot Behavior from Natural Interaction and Large Language Models,” *Frontiers in Robotics and AI*, vol. 11, 2024. Cited on pages 1 and 4.
- [15] K. Rana, J. Haviland, S. Garg, J. Abou-Chakra, I. Reid, and N. Suennderhauf, “SayPlan: Grounding Large Language Models using 3D Scene Graphs for Scalable Robot Task Planning,” in *Conference on Robot Learning (CoRL)*, 2023. Cited on page 2.
- [16] Q. Xie, S. Y. Min, P. Ji, Y. Yang, T. Zhang, K. Xu, A. Bajaj, R. Salakhutdinov, M. Johnson-Roberson, and Y. Bisk, “Embodied-RAG: General Non-parametric Embodied Memory for Retrieval and Generation,” 2025. Cited on page 2.
- [17] A. Padmakumar, J. Thomason, A. Shrivastava, P. Lange, A. Narayan-Chen, S. Gella, R. Piramuthu, G. Tur, and D. Hakkani-Tur, “TEACH: Task-Driven Embodied Agents That Chat,” *AAAI Conference on Artificial Intelligence*, vol. 36, no. 2, pp. 2017–2025, 2022. Cited on pages 2, 4, and 6.
- [18] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, M. Martin, T. Nagarajan, I. Radosavovic, S. K. Ramakrishnan, F. Ryan, J. Sharma, M. Wray, M. Xu, E. Z. Xu, C. Zhao, S. Bansal, D. Batra, V. Cartillier, S. Crane, T. Do, M. Doulaty, A. Erappalli, C. Feichtenhofer, A. Fragomeni, Q. Fu, A. Gebrselassie, C. González, J. Hillis, X. Huang, Y. Huang, W. Jia, W. Khoo, J. Kolář, S. Kottur, A. Kumar, F. Landini, C. Li, Y. Li, Z. Li, K. Mangalam, R. Modhugu, J. Munro, T. Murrell, T. Nishiyasu, W. Price, P. Ruiz, M. Ramazanova, L. Sari, K. Somasundaram, A. Southerland, Y. Sugano, R. Tao, M. Vo, Y. Wang, X. Wu, T. Yagi, Z. Zhao, Y. Zhu, P. Arbeláez, D. Crandall, D. Damen, G. M. Farinella, C. Fuegen, B. Ghanem, V. K. Ithapu, C. V. Jawahar, H. Joo, K. Kitani, H. Li, R. Newcombe, A. Oliva, H. S. Park, J. M. Rehg, Y. Sato, J. Shi, M. Z. Shou, A. Torralba, L. Torresani, M. Yan, and J. Malik, “Ego4D: Around the World in 3,000 Hours of Egocentric Video,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 18 995–19 012. Cited on pages 2 and 6.

- [19] T. Asfour, R. Dillmann, N. Vahrenkamp, M. Do, M. Wächter, C. Mandery, P. Kaiser, M. Kröhnert, and M. Grotz, “The Karlsruhe AR-MAR Humanoid Robot Family,” in *Humanoid Robotics: A Reference*. Springer Netherlands, 2017, pp. 1–32. Cited on page 2.
- [20] L. Bärmann and A. Waibel, “Where Did I Leave My Keys? - Episodic-Memory-Based QA on Egocentric Videos,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2022, pp. 1560–1568. Cited on pages 2 and 3.
- [21] A. Waibel, H. Steusloff, R. Stiefelhagen, et al., “CHIL: Computers in the human interaction loop,” 2005. Cited on page 2.
- [22] J. Rothfuss, F. Ferreira, E. E. Aksoy, Y. Zhou, and T. Asfour, “Deep Episodic Memory: Encoding, Recalling, and Predicting Episodic Experiences for Robot Action Execution,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 4, pp. 4007–4014, 2018. Cited on page 2.
- [23] C. DeChant, I. Akinola, and D. Bauer, “In search of the embgram: forming episodic representations in a deep learning model,” in *Cognitive Computational Neuroscience 2024*, 2024. Cited on page 2.
- [24] F. Peller-Konrad, R. Kartmann, C. R. G. Dreher, A. Meixner, F. Reister, M. Grotz, and T. Asfour, “A Memory System of a Robot Cognitive Architecture and Its Implementation in ArmarX,” *Robotics and Autonomous Systems*, vol. 164, p. 104415, 2023. Cited on pages 2 and 7.
- [25] M. Beetz, D. Bessler, A. Haidu, M. Pomarlan, A. K. Bozcuoglu, and G. Bartels, “KnowRob 2.0 — A 2nd Generation Knowledge Processing Framework for Cognition-Enabled Robotic Agents,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018. Cited on page 2.
- [26] J. Plewnia, F. Peller-Konrad, and T. Asfour, “Forgetting in Robotic Episodic Long-Term Memory,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 6711–6717. Cited on page 2.
- [27] A. Yuguchi, S. Kawano, K. Yoshino, C. T. Ishi, Y. Kawanishi, Y. Nakamura, T. Minato, Y. Saito, and M. Minoh, “Butsukusa: A Conversational Mobile Robot Describing Its Own Observations and Internal States,” in *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2022, pp. 1114–1118. Cited on page 2.
- [28] A. Anwar, J. Welsh, J. Biswas, S. Pouya, and Y. Chang, “ReMEmbR: Building and Reasoning Over Long-Horizon Spatio-Temporal Memory for Robot Navigation,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2025. Cited on pages 2, 5, and 7.
- [29] Z. Wang, B. Liang, V. Dhat, Z. Brumbaugh, N. Walker, R. Krishna, and M. Cakmak, “I Can Tell What I am Doing: Toward Real-World Natural Language Grounding of Robot Experiences,” in *Conference on Robot Learning (CoRL)*, 2024. Cited on page 2.
- [30] M. Han, M. Kang, H. Jung, and S. J. Hwang, “Episodic Memory Reader: Learning What to Remember for Question Answering from Streaming Data,” in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019, pp. 4407–4417. Cited on page 2.
- [31] V. Araujo, A. Soto, and M.-F. Moens, “A Memory Model for Question Answering from Streaming Data Supported by Rehearsal and Anticipation of Coreference Information,” in *Findings of the Association for Computational Linguistics (ACL)*, 2023, pp. 13124–13138. Cited on page 2.
- [32] S. Wang, Y. Fang, Y. Zhou, X. Liu, and Y. Ma, “ArchRAG: Attributed Community-based Hierarchical Retrieval-Augmented Generation,” 2025. Cited on page 2.
- [33] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, D. Metropolitan, R. O. Ness, and J. Larson, “From Local to Global: A Graph RAG Approach to Query-Focused Summarization,” 2025. Cited on page 2.
- [34] X. Chen, P. Gao, J. Song, and X. Tan, “HiQA: A Hierarchical Contextual Augmentation RAG for Multi-Documents QA,” 2024. Cited on page 2.
- [35] Y. Zhong, W. Ji, J. Xiao, Y. Li, W. Deng, and T.-S. Chua, “Video Question Answering: Datasets, Algorithms and Challenges,” in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022, pp. 6439–6455. Cited on page 2.
- [36] Y. Song, G. Byrne, T. Nagarajan, H. Wang, M. Martin, and L. Torresani, “Ego4D Goal-Step: Toward Hierarchical Understanding of Procedural Activities,” in *Conference on Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023. Cited on page 2.
- [37] K. Mangalam, R. Akshulakov, and J. Malik, “EgoSchema: A diagnostic benchmark for very long-form video language understanding,” in *Conference on Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023. Cited on page 2.
- [38] I. Balazevic, Y. Shi, P. Papalampidi, R. Chaabouni, S. Koppula, and O. J. Henaff, “Memory Consolidation Enables Long-Context Video Understanding,” in *International Conference on Machine Learning (ICML)*, 2024. Cited on page 2.
- [39] E. Song, W. Chai, G. Wang, Y. Zhang, H. Zhou, F. Wu, H. Chi, X. Guo, T. Ye, Y. Zhang, Y. Lu, J.-N. Hwang, and G. Wang, “MovieChat: From Dense Token to Sparse Memory for Long Video Understanding,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 18221–18232. Cited on page 2.
- [40] R. Tan, X. Sun, P. Hu, J.-h. Wang, H. Deilamsalehy, B. A. Plummer, B. Russell, and K. Saenko, “Koala: Key Frame-Conditioned Long Video-LLM,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 13581–13591. Cited on page 2.
- [41] S. Di and W. Xie, “Grounded Question-Answering in Long Egocentric Videos,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 12934–12943. Cited on page 2.
- [42] Y. Fan, X. Ma, R. Wu, Y. Du, J. Li, Z. Gao, and Q. Li, “VideoAgent: A Memory-augmented Multimodal Agent for Video Understanding,” 2024. Cited on pages 2 and 3.
- [43] Z. Wang, S. Yu, E. Stengel-Eskin, J. Yoon, F. Cheng, G. Bertasius, and M. Bansal, “VideoTree: Adaptive Tree-based Video Representation for LLM Reasoning on Long Videos,” 2024. Cited on pages 2 and 3.
- [44] X. Wang, Y. Zhang, O. Zohar, and S. Yeung-Levy, “VideoAgent: Long-form Video Understanding with Large Language Model as Agent,” 2024. Cited on pages 2 and 3.
- [45] J. Min, S. Buch, A. Nagrani, M. Cho, and C. Schmid, “MoReVQA: Exploring Modular Reasoning Models for Video Question Answering,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 13235–13245. Cited on page 2.
- [46] Y. Wang, Y. Yang, and M. Ren, “LifelongMemory: Leveraging LLMs for Answering Queries in Egocentric Videos,” 2023. Cited on page 2.
- [47] R. Choudhury, K. Niinuma, K. M. Kitani, and L. A. Jeni, “Zero-Shot Video Question Answering with Procedural Programs,” 2023. Cited on page 2.
- [48] C. Zhang, T. Lu, M. M. Islam, Z. Wang, S. Yu, M. Bansal, and G. Bertasius, “A Simple LLM Framework for Long-Range Video Question-Answering,” in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024, pp. 21715–21737. Cited on page 2.
- [49] Z. Huang, Y. Ji, X. Wang, N. Mehta, T. Xiao, D. Lee, S. Vanvarkenburg, S. Zha, B. Lai, L. Yu, N. Zhang, Y. J. Lee, and M. Liu, “Building a Mind Palace: Structuring Environment-Grounded Semantic Graphs for Effective Long Video Analysis with LLMs,” 2025. Cited on page 2.
- [50] G. Goletto, T. Nagarajan, G. Averta, and D. Damen, “AMEGO: Active Memory from Long EGOCentric Videos,” in *European Conference on Computer Vision (ECCV)*, 2024, pp. 92–110. Cited on page 2.
- [51] X. Wang, Y. Chen, L. Yuan, Y. Zhang, Y. Li, H. Peng, and H. Ji, “Executable Code Actions Elicit Better LLM Agents,” in *International Conference on Machine Learning (ICML)*, 2024. Cited on page 3.
- [52] S. Ren, L. Yao, S. Li, X. Sun, and L. Hou, “TimeChat: A Time-sensitive Multimodal Large Language Model for Long Video Understanding,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 14313–14323. Cited on page 3.
- [53] N. Vahrenkamp, M. Wächter, M. Kröhnert, K. Welke, and T. Asfour, “The robot software framework ArmarX,” *it – Information Technology*, vol. 57, no. 2, pp. 99–111, 2015. Cited on page 3.
- [54] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, W. X. Zhao, Z. Wei, and J. Wen, “A survey on large language model based autonomous agents,” *Frontiers of Computer Science*, vol. 18, no. 6, p. 186345, 2024. Cited on page 3.
- [55] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weih, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, “AI2-THOR: An Interactive 3D Environment for Visual AI,” *arXiv*, 2017. Cited on page 4.
- [56] OpenAI, “GPT-4 Technical Report,” *arXiv:2303.08774*, 2023. Cited on page 5.
- [57] X. Ye, S. Iyer, A. Celikyilmaz, V. Stoyanov, G. Durrett, and R. Pasunuru, “Complementary Explanations for Effective In-Context Learning,” in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023, pp. 4469–4484. Cited on page 5.

- [58] J. Cohen, “A Coefficient of Agreement for Nominal Scales,” *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960. Cited on page 5.
- [59] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a Method for Automatic Evaluation of Machine Translation,” in *ACL*, 2002, pp. 311–318. Cited on page 5.
- [60] C.-Y. Lin, “ROUGE: A Package for Automatic Evaluation of Summaries,” in *Text Summarization Branches Out*, 2004, pp. 74–81. Cited on page 5.
- [61] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, “YOLO-World: Real-Time Open-Vocabulary Object Detection,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 16901–16911. Cited on pages 5 and 6.
- [62] M. Guo, J. Ainslie, D. Uthus, S. Ontanon, J. Ni, Y.-H. Sung, and Y. Yang, “LongT5: Efficient Text-To-Text Transformer for Long Sequences,” in *Findings of the Association for Computational Linguistics (ACL)*, July 2022, pp. 724–736. Cited on page 5.
- [63] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, A. Goyal, A. Hartshorn, A. Yang, A. Mitra, A. Srivankumar, A. Korenev, A. Hinsvark, A. Rao, A. Zhang, A. Rodriguez, A. Gregerson, A. Spataru, B. Roziere, B. Biron, B. Tang, B. Chern, C. Caucheteux, C. Nayak, C. Bi, *et al.*, “The Llama 3 Herd of Models,” 2024. Cited on page 6.
- [64] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, “Learning Transferable Visual Models From Natural Language Supervision,” in *International Conference on Machine Learning (ICML)*, vol. 139, 18–24 Jul 2021, pp. 8748–8763. Cited on page 6.
- [65] A. Gupta, P. Dollar, and R. Girshick, “LVIS: A Dataset for Large Vocabulary Instance Segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. Cited on page 6.
- [66] V. Perera, “Language-Based Bidirectional Human And Robot Interaction Learning For Mobile Service Robot,” Ph.D. dissertation, School of Computer Science, Carnegie Mellon University, 2018. Cited on page 8.