ESCA: Contextualizing Embodied Agents via Scene-Graph Generation

Jiani Huang *
University of Pennsylvania
jianih@seas.upenn.edu

Amish Sethi *
University of Pennsylvania
asethi04@seas.upenn.edu

Matthew Kuo *
University of Pennsylvania
mkuo@seas.upenn.edu

Mayank Keoliya
University of Pennsylvania
mkeoliya@seas.upenn.edu

Neelay Velingker
University of Pennsylvania
neelay@seas.upenn.edu

JungHo Jung
University of Pennsylvania
j76jung@seas.upenn.edu

Ser-Nam Lim
University of Central Florida
sernam@ucf.edu

Ziyang Li
Johns Hopkins University
ziyang@cs.jhu.edu

Mayur Naik
University of Pennsylvania
mhnaik@seas.upenn.edu

Abstract

Multi-modal large language models (MLLMs) are making rapid progress toward general-purpose embodied agents. However, current training pipelines primarily rely on high-level vision-sound-text pairs and lack fine-grained, structured alignment between pixel-level visual content and textual semantics. To overcome this challenge, we propose ESCA, a new framework for contextualizing embodied agents through structured spatial-temporal understanding. At its core is SGClip, a novel CLIP-based, open-domain, and promptable model for generating scene graphs. SGClip is trained on 87K+ open-domain videos via a neurosymbolic learning pipeline, which harnesses model-driven self-supervision from video-caption pairs and structured reasoning, thereby eliminating the need for human-labeled scene graph annotations. We demonstrate that SGClip supports both prompt-based inference and task-specific fine-tuning, excelling in scene graph generation and action localization benchmarks. ESCA with SGClip consistently improves both open-source and commercial MLLMs, achieving state-of-the-art performance across two embodied environments. Notably, it significantly reduces agent perception errors and enables open-source models to surpass proprietary baselines.

1 Introduction

Recent advances in large-scale pretraining have enabled foundation models to assist with a wide range of tasks, from language and vision understanding [2, 14, 47, 77] to mathematical problem solving [90, 17] and code generation [20, 56, 50]. However, it remains an open challenge to realize embodied agents that are capable of doing household chores, training alongside humans in physical activities, or providing care for the aging [36, 12]. A critical first step toward this goal is equipping agents with fine-grained perception to ground abstract goals in physical interactions.

Despite progress, multi-modal large language models (MLLMs) still struggle to build spatially and temporally grounded world models. Their inability to reliably ground visual features with spatial-temporal relations creates a disconnect between conceptual semantics and pixel-level observations [83,

*These authors contributed equally to this work

Preprint. Under review.

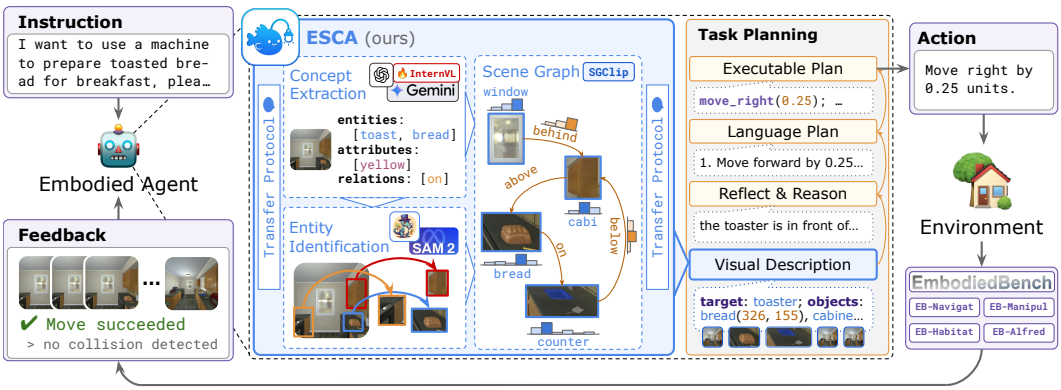


Figure 1: An overview of embodied agent pipeline augmented with ESCA. In each cycle, the agent takes in an instruction and the environmental feedback and outputs a concrete executable action, through a sequence of perception, reasoning, and planning. The action is then executed and the environment will provide the next state. Notably, ESCA contextualizes the task planner with grounded visual features represented as a scene graph.

55]. This lack of structured, fine-grained scene understanding severely limits their effectiveness in embodied environments. In fact, our empirical analysis shows that up to 69% of agent failures stem from perception errors, highlighting the need for frameworks that can bridge this gap.

To address this challenge, we propose integrating structured scene graphs into the perception, reasoning, and planning pipelines of MLLM-based embodied agents. While prior work has explored enhancing MLLMs with external visual grounding modules, these approaches typically rely on open-domain object detection models such as Grounding DINO [51] and YOLO [26]. However, these models are primarily designed for object identification and often overlook semantic attributes, inter-object relationships, and temporal consistency.

In this work, we introduce **ESCA** (**E**mbodied and **S**cene-Graph **C**ontextualized **A**gent), a framework designed to contextualize MLLMs through open-domain scene graph generation (Figure 1). Much like the bioluminescent lure of a deep-sea anglerfish, which illuminates its surroundings to reveal otherwise hidden prey, ESCA provides structured visual grounding that helps MLLMs make sense of complex and ambiguous sensory environments. A key feature of ESCA is selective grounding: rather than injecting full scene graphs, which may degrade performance, the MLLM first identifies the subset of objects, attributes, and relations most pertinent to the instruction, then determines the essential entities for task completion. This mechanism is supported by our transfer protocol, which performs probabilistic reasoning over object names, attributes, and spatial relations to construct prompts enriched with the most relevant scene elements. At its core is SGClip, a CLIP-based model that captures semantic visual features, including entity classes, physical attributes, actions, interactions, and inter-object relations.

Through experiments on four challenging embodied environments, we demonstrate that ESCA consistently improves the performance of all evaluated MLLMs, including both open-source and proprietary models. By providing structured and grounded scene graphs, ESCA significantly reduces perception errors, laying the foundation for more reliable reasoning and planning. Beyond its integration with MLLM-based agents, we show that SGClip, when evaluated independently, exhibits strong zero-shot generalization, is promptable for task-specific scene understanding, and remains fine-tunable for downstream tasks such as action recognition.

In summary, our contributions are as follows: (1) we present ESCA, a general framework for contextualizing MLLM-based embodied agents through selective scene graph generation; (2) we introduce the transfer protocol for enriching prompts with probabilistically inferred scene-specific information for diverse embodied benchmarks; (3) we introduce SGClip, a generalizable and fine-grained scene graph generation model, along with ESCA-Video-87K, an MLLM annotated dataset; (4) we conduct extensive evaluations demonstrating the effectiveness and versatility of ESCA and SGClip across both embodied agent and scene understanding tasks.

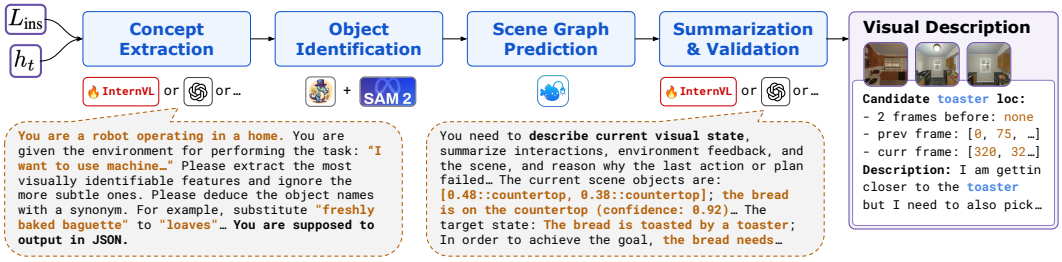


Figure 2: A detailed illustration of the visual description module, which involves concept extraction, object identification, scene graph prediction, and visual summarization. We also illustrate sample MLLM prompts used in a kitchen environment for the concept extraction and summarization steps.

2 ESCA: A Framework for Embodied Agents

2.1 Background

Embodied Environments. Recent research on embodied agents has been accelerated by the availability of simulated environments such as VisualAgentBench [52] and EmbodiedBench [85], which provide rich, multimodal task suites covering navigation, manipulation, and interaction across diverse scenarios. These benchmarks pose challenges that require agents to operate in both a) low-level action spaces such as continuous control signals over robot joints, and b) high-level action spaces such as programmatic instructions and skills that abstract over low-level actions.

These tasks are typically modeled as Partially Observable Markov Decision Processes (POMDPs), represented as 7-tuple $(S, A, \Omega, \mathcal{T}, \mathcal{O}, L_{\text{ins}}, \mathcal{R})$, where S is the unobservable state space, A is the task-specific action space, Ω is the visual perception space where $I_t \in \Omega$ is an image frame at time t, $\mathcal{T} : S \times A \to S$ is the state transition dynamics, $\mathcal{O} : S \to \Omega$ relates the underlying state to the observations, L_{ins} is the natural language instruction for the agent, and $\mathcal{R} : S \to \{0, 1\}$ is the reward function indicating whether the task has been completed.

Embodied Agents and MLLM-Based Embodied Agents. A typical embodied agent interacts with the environment by maintaining a history of observations and actions: $h_t = (I_0, a_0, I_1, \ldots, a_{t-1}, I_t)$, where a_t is the action taken by the agent at time t. The agent selects the next action by conditioning on the instruction L and history h_t via a policy $\pi(a_t \mid L_{\text{ins}}, h_t)$. An *MLLM-based embodied agent* realizes such a policy by leveraging a multi-modal model that processes both: a) imagery data I_t, and b) textual data, including the instruction L_{ins} and textual representations of actions $a \in A$.

Established by [85], recent MLLM-based agent architectures often decompose the policy evaluation process into structured stages inspired by cognitive reasoning workflows (Figure 1): 1) *Visual Description*: extracting and summarizing visual inputs; 2) *Reflection*: integrating observations with historical context to build situational awareness; 3) *Reasoning*: inferring task-relevant insights based on the combined context; 4) *Language Plan*: generating high-level plans or action sequences in natural language; 5) *Executable Plan*: translating plans into concrete actions executable by the agent.

Challenges of MLLM-Based Embodied Agents. Despite recent progress, MLLM-based embodied agents remain fragile in complex environments due to compounded errors across perception, reasoning, and planning [36, 12]. *Perception errors* include hallucinated objects, misrecognized entities or actions, and incorrect spatial relationships. *Reasoning errors* arise when agents fail to correctly infer spatial relations or recognize task termination states. These issues propagate into *planning*, where agents may skip critical steps or generate invalid plans due to inaccurate state estimation.

2.2 The ESCA Framework

At the core of ESCA is a simple yet effective idea: contextualizing MLLM-based agents with grounded, structured scene-graph information to improve visual description. Specifically, given a language instruction L_{ins} and interaction history h_t, the agent generates a multi-modal message sequence of images and text. While existing approaches rely on end-to-end MLLMs to implicitly perform this step, ESCA decomposes the process into four modular stages below (Figure 2).

Selective Concept Extraction. This module extracts concepts with MLLM guided by carefully designed prompts based on the instruction L_{ins} and the history h_t. Rather than producing only free-form natural descriptions, ESCA requires the MLLM to explicitly extract structured concepts that are most relevant to the query, which can be classified as follows. a) *entity classes* such as (`car`, `knife`, `person`, `cup`); b) *attributes* including physical properties (`red`, `small`, `broken`) and semantic states (`close-by`, `far`, `moving`, `sitting`); c) *relations* covering spatial relations (`behind`, `above`) and interactions (`cutting`, `cooking`).

Concept extraction is guided by two signals: 1) the instruction L_{ins}, which highlights target entities, attributes, or relations the agent should focus on, and 2) the visual feedback I_t, which provides spatial context such as environmental structures (e.g., barriers, pathways) and dynamic interactions (e.g., objects being manipulated). Overall, this MLLM-generated structured output provides a targeted concepts set $\bar{c} = \{c_1, c_2, \dots\}$ for subsequent object identification and scene graph generation.

Object Identification. Given the extracted concepts and the agent's visual feedback, the second module grounds these concepts to specific image segments $\bar{\sigma} = \{\sigma_1, \sigma_2, \dots\}$ each represented by a bit-mask. This step isolates visual elements in the frame that correspond to classified entities or attributes, enabling downstream semantic inference over localized visual units rather than raw pixels.

The object identification module is implemented using a multi-stage pipeline. First, Grounding DINO (GD) [51], a vision-language detection model, takes the extracted concepts and the visual input to predict bounding boxes for the mentioned entities. These bounding boxes are further refined into precise segmentation masks using SAM2 [62], a segmentation model that produces pixel-accurate regions. Leveraging the GD + SAM2 pipeline improves computational efficiency and offers better generalization to both attribute and relational concepts, as further detailed in the Appendix.

Scene Graph Prediction. We then construct a scene graph (SG) [35, 81, 31] by grounding the extracted concepts into structured visual elements. Specifically, the SG contains two types of probabilistic facts: 1) unary facts of the form $p :: c_i(\sigma_j)$, each describing an entity σ_j with their classes, attributes, or states represented as c_i; 2) relational facts of the form $p :: c_i(\sigma_j, \sigma_k)$, each representing that a relation or an interaction c_i between a pair of grounded entities σ_j and σ_k. Notably, each predicted fact is associated with a confidence score denoted as p. This probabilistic formulation allows the SG to capture uncertainty and distributions over possible scene interpretations.

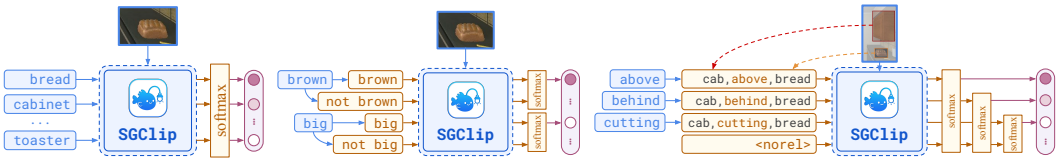
Implementation-wise, we build SGClip, a CLIP-based model [60] fine-tuned for open-domain scene graph prediction. The design of SGClip is guided by three aforementioned key desiderata. First, it supports open-domain concept coverage, enabling it to generalize beyond fixed taxonomies. Second, it is adaptable to extracting diverse types of information, including entity classes, attributes, and inter-object relations. Third, it produces probabilistic predictions, allowing the model to capture uncertainties. We provide a more detailed discussion of SGClip and its training in Section 3.

Visual Summarization and Validation. This module distills these multi-modal signals into a list of messages to contextualize the agent's reasoner and planner. Concretely, the summarizer takes a prompt and is responsible for transforming the structured scene graph into natural descriptions, while also validating the consistency between the visual feedback and the underlying structured scene graph. While the summary is customizable via our transfer protocol, the set of messages include 1) the current view (and potentially historic ones) augmented with visualized bounding boxes served as markers, 2) image segments corresponding to key entities, 3) the textual description of scene graph and segments, and 4) an analysis of history actions and how they caused the current scene.

2.3 Transfer Protocol

To enable ESCA to generalize across different downstream tasks, we define a general transfer protocol based on the customization of two prompt templates, positioned at the entry and exit points of the entire visual description module. The goal of this unified transfer protocol is to maximize adaptability of ESCA across tasks with diverse planning strategies, action spaces, and reasoning requirements, while maintaining a consistent interface.

Specifically, the first prompt, the *Concept Extraction Prompt*, specifies the required JSON output format and indicates, or enumerates if possible, task-specific concepts that the agent should focus on. The second prompt, the *Visual Summarization Prompt*, guides the model to produce a contextualized summary that integrates both grounded image segments and task-specific textual elements, such



(a) Entity classes (b) Attributes (c) Binary relations

Figure 3: Illustration of the inference modes of SGClip for three types of concepts: entity classes, attributes, and binary relations. While the model stays the same, the three inference modes perform different pre- and post-processing for a more accurate semantic estimation of probabilities.

as target objects, desired states, and environmental constraints. Together, these prompts provide a principled way to adapt ESCA to diverse embodied AI tasks without retraining the core system. We present a case study of the transfer protocol in Figure 2 and provide further details in Section 4.

3 SGClip Model

To enable the generation of spatial-temporal scene graphs in an open-domain setting, especially the embodied environments, we develop SGClip, a CLIP-based foundation model [60] for structured scene understanding. SGClip is designed to operate in an open-domain fashion, recognizing a wide and extensible set of concepts. Secondly, it must adapt to different types of concepts, while be capable of generalizing to unseen visual and textual domains. Finally, it must produce probabilistic predictions to capture uncertainty.

To meet these goals, SGClip builds on CLIP's vision-language architecture, which naturally supports joint reasoning over images and textual phrases. However, deploying CLIP directly is insufficient, as it lacks specialization for structured scene graph prediction. To bridge this gap, we fine-tune SGClip to balance adaptability and generalizability. In this section, we describe 1) how to handle different types of concepts through inference time adaptation (Section 3.1), 2) how to overcome the lack of data by collecting a model-driven self-supervision dataset (Section 3.2), and 3) how to learn without relying on human annotations via a self-supervised neurosymbolic learning pipeline (Section 3.3).

3.1 Model Architecture and Inference Time Adaptation

At its core, SGClip (Scene Graph CLIP) is a single CLIP-based model designed to score concept relevance within an image. Formally, it operates as $\text{SGClip}(\sigma, \bar{c}) \in \mathbb{R}^{|\bar{c}|}$, where σ is the input image and $\bar{c}$ is the a set of candidate concepts, producing a logit score for each concept that reflects the model's confidence in its presence. SGClip supports three distinct inference modes to handle different types of concepts: entity classes $\bar{c}_{\text{class}}$, attributes $\bar{c}_{\text{attr}}$, and binary relations $\bar{c}_{\text{rela}}$. Illustrated in Figure 3, each mode requires a specialized formulation of the input concepts and scoring process, effectively allowing SGClip to operate flexibly across these concept types:

Entity classes. In this setting, SGClip is used to identify the most likely entity class presented in an image segment. Let $\bar{c}_{\text{class}}$ denote the list of candidate entity classes. Since an entity is typically assumed to belong to a single class, we apply softmax normalization over the logit scores produced by SGClip for these candidates: $\text{softmax}(\text{SGClip}(\sigma, \bar{c}_{\text{class}}))$.

Attributes. To estimate the likelihood that a specific segment σ possesses a particular attribute c, we construct a binary contrast between the attribute and its negation by evaluating $\text{softmax}(\text{SGClip}(\sigma, \{c, \neg c\}))$, where $\neg c$ denotes the negated textual phrase (e.g., "not red" for the attribute "red"). The first element of the resulting probability distribution corresponds to the model's estimated likelihood. To improve computational efficiency, we perform batched evaluation by merging all attribute-contradiction pairs into a single concept set $\bar{c}^*_{\text{attr}} = \bar{c}_{\text{attr}} \cup \{\neg c \mid c \in \bar{c}_{\text{attr}}\}$.

Binary relations. For binary relation prediction, the goal is to determine whether a relation c holds between two segments σ_i and σ_j. To do this, we first compute a bounding region σ^*_{ij} that tightly encloses both segments. Within this region, we apply distinct color tinting to σ_i and σ_j to indicate their directional roles (subject and object). To provide additional relational context, especially for interactions like "`cutting`," we augment the relation phrase by including the predicted entity classes

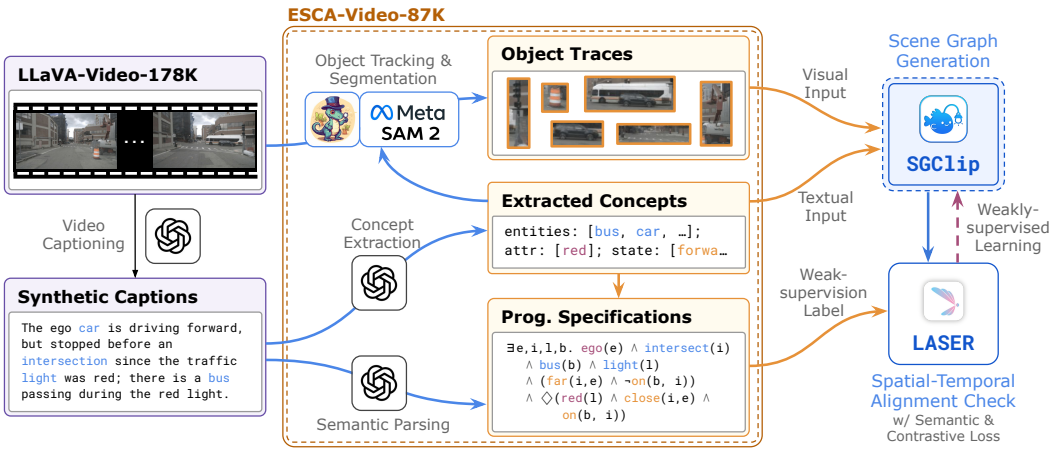


Figure 4: Illustration of the construction of ESCA-Video-87K dataset and the model-driven self-supervised fine-tuning pipeline of our SGClip model. In addition to videos and their natural language captions, ESCA-Video-87K includes object traces, open-domain concepts, and programmatic specifications for 87K video-caption pairs. The dataset is then used to train SGClip via LASER [31], a neurosymbolic learning procedure based on spatial-temporal alignment.

of the subject and object, generating "(`robot`, `cutting`, `cabbage`)". Relation predictions are thus conditioned on the classes of both the subject and the object. Specifically, for each segment σ_i, we compute its most likely class ν_i by selecting the top prediction from the entity class:

$$\nu_i = c_{\text{class}\,u}, \quad \text{where } u = \text{argmax}_{u \in 1\ldots|\bar{c}_{\text{class}}|} \text{SGClip}(\sigma_i, \bar{c}_{\text{class}})_u.$$

We then form the augmented relation phrase as (ν_i, c, ν_j). Similar to attribute prediction, we contrast the candidate relation with a special token `<norel>` denoting "no relation," and compute $\text{softmax}(\text{SGClip}(\sigma^*_{ij}, \{(\nu_i, c, \nu_j), \texttt{<norel>}\}))$, to obtain the probability of whether the relation c holds between object i and j. In practice, this process is batched over all segment pairs and relation concepts to maximize efficiency: $\bar{c}^*_{\text{rela}} = \{(\nu_i, c, \nu_j) \mid i, j \in 1\ldots|\bar{\sigma}|, c \in \bar{c}_{\text{rela}}\} \cup \{\texttt{<norel>}\}$.

3.2 ESCA-Video-87K Dataset

We adopt the neurosymbolic weak-supervision pipeline introduced in LASER [31], which enables learning fine-grained STSGs from *weak supervision signals* derived from spatial-temporal programmatic specifications, eliminating the need for costly manual annotations. While the details of this learning pipeline are provided in Section 3.3, we begin by introducing the ESCA-Video-87K dataset, the dataset we curate and use to train SGClip.

The ESCA-Video-87K dataset is constructed on top of the publicly available LLaVA-Video-178K dataset [91], and consists of 87K short video clips, each paired with natural language captions generated by GPT-4 [32]. As illustrated in Figure 4, these captions are first processed to extract relevant concepts which are then fed into GD [51] and SAM2 [62] to obtain object traces, which are sequences of object segmentations that evolve across multiple video frames. In addition, these concepts are also used to assist in generating spatial-temporal programmatic specifications, expressed in a linear temporal logic-based language. To construct these specifications, we develop a semantic parsing pipeline, again leveraging GPT-4, which converts high-level captions into structured temporal statements. These specifications formally describe how the semantics of object traces evolve, capturing temporal relations using operators such as "until", "finally", or "always".

In summary, each data point in ESCA-Video-87K is represented as a 5-tuple $(\bar{I}, L_{\text{cap}}, \Sigma, \bar{c}, \phi)$, where $\bar{I} = \{I_1, I_2, \ldots\}$ is the video, L_{cap} is the associated natural language caption, $\Sigma = \{\bar{\sigma}_1, \bar{\sigma}_2, \ldots\}$ is the set of object traces, $\bar{c} = \{c_1, c_2, \ldots\}$ is the set of extracted concepts, and ϕ is the spatial-temporal programmatic specification. This rich, multi-level annotation enables training models like SGClip without requiring manual scene graph labeling. We defer additional details about the dataset construction process and data statistics to the Appendix.

3.3 Neurosymbolic Learning Pipeline

Given the ESCA-Video-87K, our goal is to fine-tune the SGClip model using the provided object traces, concepts, and spatial-temporal programmatic specifications. This is achieved by aligning the scene graphs generated by SGClip with the expected specifications [31], where the degree of alignment serves as the learning signal (Figure 4). To perform this alignment in a differentiable manner, we leverage the Scallop programming language [46], enabling symbolic alignment checks to be integrated into end-to-end gradient-based learning.

Specifically, the alignment loss computation mirrors the inference-time adaptation procedure described in Section 3.1, where different types of concepts are processed differently but unified under the same model. The pipeline is further enhanced with *semantic losses*, derived from evaluating common-sense and temporal constraint satisfaction, as well as a *contrastive loss* that encourages the model to distinguish between matched and unmatched scene graph–specification pairs. Additional details of the training process are provided in the Appendix.

4 Embodied Environments and Transfer Protocol Setup

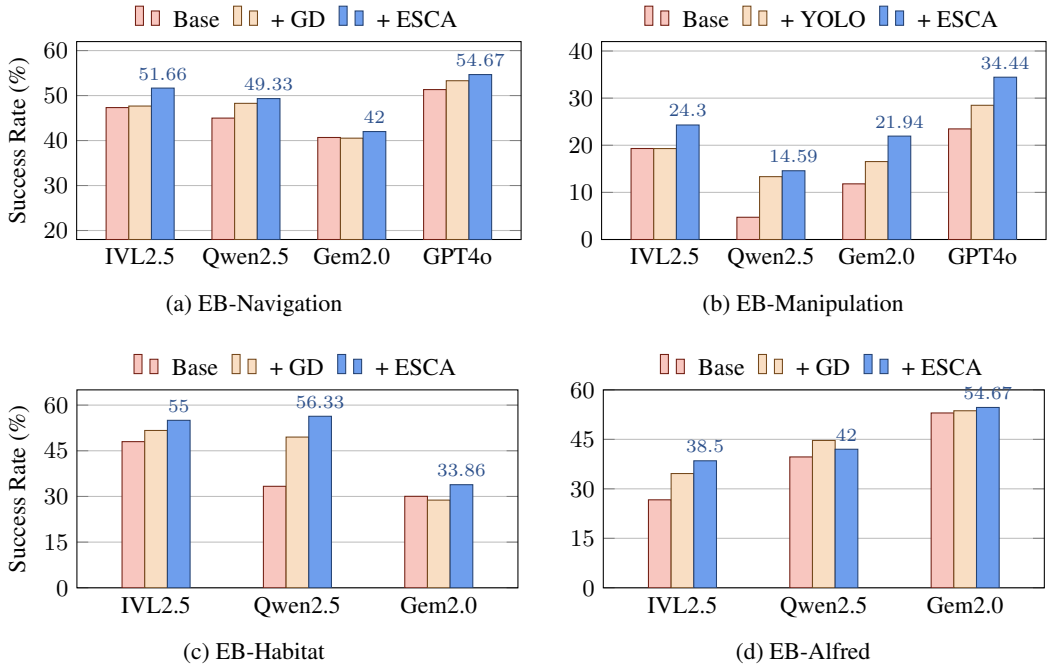
We evaluate our approach on EmbodiedBench [42], a benchmark suite designed to assess MLLM-based embodied agents. We focus on two environments: EB-Navigation and EB-Manipulation, each of which requires different levels of perception, reasoning, and control, and may benefit from a contextualized visual description. To adapt ESCA to these tasks, we apply our transfer protocol by designing two specialized prompts for each environment. Full prompt templates are provided in the Appendix; here, we summarize the core challenges and how ESCA addresses them.

EB-Navigation is built on AI2-THOR [39] and focuses on visual navigation tasks where the agent must locate target objects based on language instructions, such as "navigate to the laptop." The agent relies solely on egocentric visual input and textual feedback, navigating through a space using eight low-level movement and rotation actions. Existing models often fail by generating correct high-level plans but producing incorrect low-level actions due to poor spatial grounding from an egocentric perspective. ESCA addresses this by generating accurate scene graphs that capture spatial relations and object positions. With the scene graphs, we design prompts that incorporate *numerical bounding box data* and *temporal movement cues*, helping the agent to localize targets more precisely.

EB-Manipulation extends VLMBench [93] for evaluating low-level robotic manipulation. The agent controls a 7-DoF robotic arm using discretized action spaces, with additional signals such as YOLO bounding boxes [26] and global 3D object pose estimates to assist manipulation. Existing models struggle to ground object concepts into actionable spatial representations, leading to perception failures that disrupt downstream planning and control. ESCA improves this by grounding target features into precise visual segments, enabling prompts that describe object attributes, semantic relations, and *3D spatial coordinates*, giving the agent more reliable geometric context.

EB-Habitat builds upon Language Rearrangement task [72], simulated via Habitat 2.0 [71], and primarily evaluates high-level task decomposition and planning capabilities. The action space is limited to atomic high-level actions, such as `navigate`/`pick`/`place`/`open`/`close`, from which the agent is instructed to complete tasks such as "Find a toy airplane and move it to the right counter". Common failure cases of existing models include invalid actions arising from failure to identify and remember object displacements in the scene. Our model improves this by managing scene graphs of the current and desired state of the target object, which improves awareness of task progression and limits the number of focus objects.

EB-Alfred is based on the ALFRED dataset [67] and AI2-THOR [39]. It evaluates agents on high-level household tasks involving eight skill types like "pick up" or "turn off." The agent receives egocentric observations and textual feedback on action validity, performing actions on objects. While agents receive egocentric observations and action feedback, existing models tend to repeat the same mistakes because they fail to reflect on how past actions influence the current state. ESCA addresses this by generating scene graphs that describe both the current and target states *symbolically*, enabling prompts that support *causal reasoning*. This allows the agent to recognize how previous actions led to the current situation and to deduce the necessary state changes to achieve the task goal.



(a) EB-Navigation

(b) EB-Manipulation

(c) EB-Habitat

(d) EB-Alfred

Figure 5: The overall performance on EB-Navigation and EB-Manipulation environments. We show the performance of four base models, InternVL-2.5-38B-MPO (IVL2.5), Genimi-2.0-flash (Gem2.0), Qwen2.5-VL-72B-Instruct (Qwen2.5), and GPT-4o (GPT4o), as well as their performance when accompanied with baseline visual grounding modules such as Grounding DINO (GD) or YOLO. With ESCA and SGClip, all models consistently outperform the baselines.

5 Empirical Evaluation

Our experiments are designed to address two key research questions: (1) How effectively does ESCA, together with SGClip, improve embodied agent performance through structured scene graph generation? and (2) How generalizable and adaptable is SGClip when evaluated independently on open-domain, zero-shot, and downstream transfer tasks? We now detail our experimental setup and present empirical results addressing both questions.

Experimental Setup. We evaluate ESCA in EB-Navigation and EB-Manipulation, two environments that demand fine-grained perception to support low-level control. To assess the general applicability of ESCA, we integrate it with four diverse MLLMs: InternVL-2.5-38B-MPO [13] , Qwen2.5-VL-72B-Ins [3], Gemini-2.0-flash [57], and GPT-4o [32]. For each MLLM experiment, we use the model for both the concept extraction and visual summarization steps (Figure 2). To further benchmark ESCA's impact, we compare to performance of MLLMs augmented with existing visual grounding modules, including Grounding DINO [51] and Ultralytics-YOLO11 [26].

For evaluating SGClip independently, we consider out-of-domain scene graph benchmarks, including OpenPVSG [84], Action Genome [33], and VidVRD [65], comparing SGClip against strong baselines such as CLIP [60], InternVL-6B [14], BIKE [79], and Text4Vis [78]. To assess SGClip's downstream adaptability beyond structured scene graph prediction, we further test the fine-tunability on the ActivityNet action recognition dataset by applying a transfer protocol.

ESCA **for Embodied Agents.** As shown in Figure 5, ESCA-augmented MLLMs consistently outperform their non-contextualized baselines across both EB-Navigation and EB-Manipulation. Remarkably, on EB-Navigation, even the open-source InternVL-2.5, when augmented with ESCA, surpasses the base performance of the proprietary GPT-4o model. While integrating Grounding DINO or YOLO improves baseline models, ESCA provides additional, substantial gains. For example,

[1]The three top-level error types are Perception, Reasoning, and Planning. The second-level errors are Hallucination, Wrong Recognition, Spatial Understanding, Spatial Reasoning, Reflection Error, Inaccurate Action, and Collision. For clarity, the figure uses these acronyms to label the different error types.

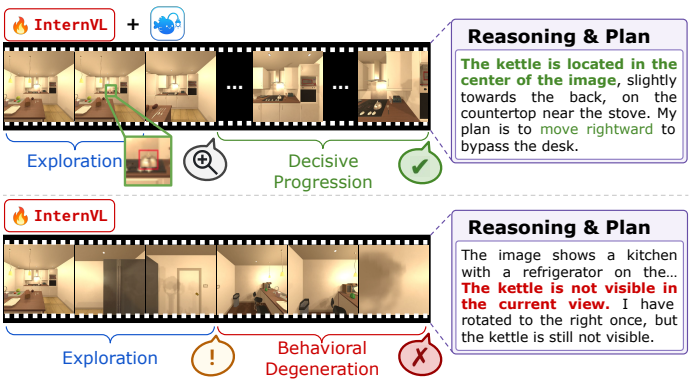


Figure 6: Two traces by InternVL with and without ESCA, on the task of "navigate to the kettle on the stove". With ESCA, the kettle can be marked in the image and textual description, encouraging the agent to decisively move towards the goal.

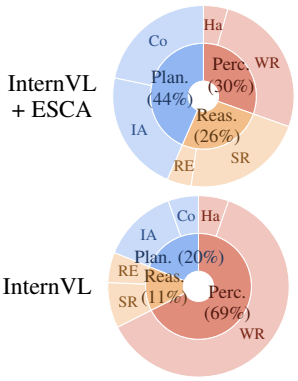
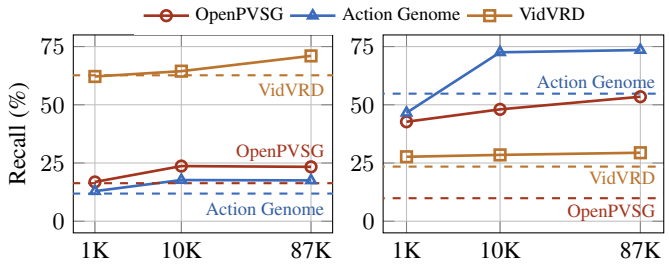


Figure 7: Error decomposition[1] of InternVL with or without ESCA, manually inspected on 60 EB-Navigation tasks.



(a) Entity class prediction R@1 (b) Binary relation prediction R@10

Figure 9: The zero-shot performance of SGClip compared to CLIP (shown in dashed lines) on OpenPVSG, Action Genome, and VidVRD datasets. We showcase the Recall@1 metrics on entity class prediction, as well as the Recall@10 metrics on binary relation prediction. To illustrate data-efficiency, we include the performance of checkpoints of SGClip when trained on 1K, 10K, or 87K (full) portion of ESCA-Video-87K.

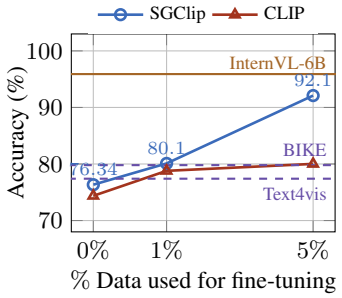


Figure 10: Down-stream fine-tunability on action recognition, evaluated on ActivityNet dataset. We also illustrate zero-shot baselines (BIKE and Text4vis) as well as a fully-supervised baseline (InternVL-6B).

Gemini-2.0 with ESCA achieves over 10% improvement, while GPT-4o, already boosted by YOLO, still benefits from an additional 6% performance gain on EB-Manipulation.

Through qualitative analysis of end-to-end agent behaviors, we observe that ESCA consistently improves the agent's perceptual grounding, leading to more effective task execution. As illustrated in Figure 6, an agent powered by InternVL with ESCA successfully identifies the kettle early in the episode and navigates directly toward it. In contrast, the base InternVL model fails to recognize the target and ultimately collapses onto the wall. This observation is further supported by the error decomposition analysis shown in Figure 7, where we find that ESCA reduces the overall perception error rate from 69% to 30%. We provide additional detailed experimental results in the Appendix.

Generalizability and Adaptability of SGClip. Evaluating SGClip's zero-shot generalization, Figure 9 shows that SGClip trained on ESCA-Video-87K consistently outperforms CLIP on OpenPVSG, Action Genome, and VidVRD, demonstrating strong out-of-domain robustness. Further, SGClip shows strong adaptability, achieving notable improvements when fine-tuned on VidVRD (details provided in the Appendix). Beyond scene graph tasks, Figure 10 highlights SGClip's downstream transferability to action recognition on ActivityNet. Fine-tuned with only 1% of the training data, SGClip outperforms state-of-the-art zero-shot video recognition baselines. With 5% of the data (approximately 800 videos), SGClip achieves 92.10% accuracy, approaching the performance of InternVideo2-6B with end-to-end finetuning on the ActivityNet dataset.

6 Related Works

Scene Graph in planning. Scene graphs [35, 81] are symbolic representations that encode the semantic structure of an image or video by identifying objects as nodes and their relationships as edges [53, 40]. They play a central role in a variety of vision-related tasks, including visual question answering [41, 27, 59], image captioning [86, 94], and image generation [25, 43]. More recently, scene graphs have been increasingly adopted in the domain of robotic planning, for enhanced robustness [29, 75], or verifiable planning [34, 61, 15]. To enable seamless integration with multimodal large language model (MLLM) agents, ESCA constructs scene graphs from 2D image inputs and dynamically updates them through embodied interaction with the environment.

Embodied Agents. Embodied agents [21, 92] are autonomous systems that perceive, reason, and interact within physical or simulated environments. To evaluate their capabilities, varies benchmarks [42, 80] have been proposed, ranging from vision-language navigation [63, 4, 16] and object manipulation [18, 48] to interactive instruction following and long-horizon planning [67, 68, 19]. Recent advances in large language models (LLMs) [7, 32, 73, 74] and multimodal large language models (MLLMs) [2, 14, 47, 77] are driving progress toward general-purpose embodied agents [1, 5, 8, 30, 88, 54]. Furthermore, recent MLLMs have been trained end-to-end to directly generate low-level numerical control commands [9, 6]. ESCA introduces a general framework for augmenting vision-driven, MLLM-based agents with structured scene graph information.

Neurosymbolic Methods with LLM and MLLM. A growing trend for enhancing the reasoning capabilities and robustness of large language models (LLMs) is to incorporate structured representations and leverage symbolic algorithms to reason over them [37, 22, 87, 45, 44]. These efforts span diverse domains, including code generation [20, 56, 50], mathematical problem solving [90, 17], and verifiable planning [70, 49, 11, 89]. Recent work has extended this paradigm to the low level control domain, training MLLMs with structured scene graphs in an end-to-end manner [6, 66, 58]. ESCA follows this neurosymbolic direction by introducing SGClip, which is trained in MLLM-augmented self-supervised manner enabled by neuro-symbolic methodology, using structured scene graphs as an intermediate representation to guide learning.

7 Conclusion and Limitations

We introduced ESCA, a framework for contextualizing embodied agents through scene graph generation, powered by SGClip, a promptable, open-domain scene graph model. Through a general transfer protocol, ESCA adapts to diverse tasks and consistently improves agent performance across multiple environments and MLLMs. Beyond embodied tasks, SGClip demonstrates strong generalization and adaptability on open-domain scene graph and action recognition benchmarks.

Limitations. Despite its strong performance, our framework has several limitations. First, the use of large language models for high-level planning introduces latency, making it unsuitable for real-time low-level control. Second, the system relies on 2D visual inputs and lacks support for 3D representations like point clouds, limiting depth-aware reasoning and spatial precision. Finally, while ESCA leverages MLLMs to generate coherent plans, it lacks formal mechanisms for verifying intermediate and final states during execution.

Acknowledgements. This research was supported by the ARPA-H program on Safe and Explainable AI under award #D24AC00253-00, the NSF under award #2313010, a gift from Google, and a gift from AWS AI to ASSET–Penn Engineering Center on Trustworthy AI.

References

[1] Anurag Ajay, Seungwook Han, Yilun Du, Shuang Li, Abhi Gupta, Tommi Jaakkola, Josh Tenenbaum, Leslie Kaelbling, Akash Srivastava, and Pulkit Agrawal. Compositional foundation models for hierarchical planning. *Advances in Neural Information Processing Systems*, 36:22304–22325, 2023.

[2] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 1(2):3, 2023.

[3] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025.

[4] Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*, 2020.

[5] Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024.

[6] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π0: A vision-language-action flow model for general robot control, 2024. *URL https://arxiv. org/abs/2410.24164*.

[7] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[8] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[9] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[10] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970, 2015.

[11] Alessio Capitanelli and Fulvio Mastrogiovanni. A framework for neurosymbolic robot action planning using large language models. *Frontiers in Neurorobotics*, 18:1342786, 2024.

[12] Yanan Chen, Ali Pesaranghader, Tanmana Sadhu, and Dong Hoon Yi. Can we rely on llm agents to draft long-horizon plans? let's take travelplanner as an example, 2024.

[13] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, Lixin Gu, Xuehui Wang, Qingyun Li, Yimin Ren, Zixuan Chen, Jiapeng Luo, Jiahao Wang, Tan Jiang, Bo Wang, Conghui He, Botian Shi, Xingcheng Zhang, Han Lv, Yi Wang, Wenqi Shao, Pei Chu, Zhongying Tu, Tong He, Zhiyong Wu, Huipeng Deng, Jiaye Ge, Kai Chen, Kaipeng Zhang, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling, 2025.

[14] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 24185–24198, 2024.

[15] Zhirui Dai, Arash Asgharivaskasi, Thai Duong, Shusen Lin, Maria-Elizabeth Tzes, George Pappas, and Nikolay Atanasov. Optimal scene graph planning with large language model guidance, 2024.

[16] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, et al. Robothor: An open simulation-to-real embodied ai platform. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3164–3174, 2020.

[17] Aniket Didolkar, Anirudh Goyal, Nan Rosemary Ke, Siyuan Guo, Michal Valko, Timothy Lillicrap, Danilo Jimenez Rezende, Yoshua Bengio, Michael C Mozer, and Sanjeev Arora. Metacognitive capabilities of llms: An exploration in mathematical problem solving. *Advances in Neural Information Processing Systems*, 37:19783–19812, 2024.

[18] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Manipulathor: A framework for visual object manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4497–4506, 2021.

[19] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35:18343–18362, 2022.

[20] James Finnie-Ansley, Paul Denny, Brett A Becker, Andrew Luxton-Reilly, and James Prather. The robots are coming: Exploring the implications of openai codex on introductory programming. In *Proceedings of the 24th Australasian computing education conference*, pages 10–19, 2022.

[21] Stan Franklin. Autonomous agents as embodied ai. *Cybernetics & Systems*, 28(6):499–520, 1997.

[22] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR, 2023.

[23] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

[24] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18995–19012, 2022.

[25] Roei Herzig, Amir Bar, Huijuan Xu, Gal Chechik, Trevor Darrell, and Amir Globerson. Learning canonical representations for scene graph to image generation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16*, pages 210–227. Springer, 2020.

[26] Priyanto Hidayatullah, Nurjannah Syakrani, Muhammad Rizqi Sholahuddin, Trisna Gelar, and Refdinal Tubagus. Yolov8 to yolo11: A comprehensive architecture in-depth comparative review, 2025.

[27] Marcel Hildebrandt, Hang Li, Rajat Koner, Volker Tresp, and Stephan Günnemann. Scene graph reasoning for visual question answering. *arXiv preprint arXiv:2007.01072*, 2020.

[28] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, Adriane Boyd, et al. spacy: Industrial-strength natural language processing in python. 2020.

[29] Joy Hsu, Jiayuan Mao, Josh Tenenbaum, and Jiajun Wu. What's left? concept grounding with logic-enhanced foundation models. *Advances in Neural Information Processing Systems*, 36:38798–38814, 2023.

[30] Cassie Huang and Li Zhang. On the limit of language models as planning formalizers, 2025.

[31] Jiani Huang, Ziyang Li, Mayur Naik, and Ser-Nam Lim. LASER: A neuro-symbolic framework for learning spatio-temporal scene graphs with weak supervision. In *The Thirteenth International Conference on Learning Representations*, 2025.

[32] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[33] Jingwei Ji, Ranjay Krishna, Li Fei-Fei, and Juan Carlos Niebles. Action genome: Actions as compositions of spatio-temporal scene graphs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10236–10247, 2020.

[34] Ziyuan Jiao, Yida Niu, Zeyu Zhang, Song-Chun Zhu, Yixin Zhu, and Hangxin Liu. Sequential manipulation planning on scene graph. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8203–8210. IEEE, 2022.

[35] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3668–3678, 2015.

[36] Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. Llms can't plan, but can help planning in llm-modulo frameworks, 2024.

[37] Nora Kassner, Oyvind Tafjord, Hinrich Schütze, and Peter Clark. Beliefbank: Adding memory to a pre-trained language model for a systematic notion of belief. *arXiv preprint arXiv:2109.14723*, 2021.

[38] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[39] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: an interactive 3d environment for visual AI. *CoRR*, abs/1712.05474, 2017.

[40] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123:32–73, 2017.

[41] Soohyeong Lee, Ju-Whan Kim, Youngmin Oh, and Joo Hyuk Jeon. Visual question answering over scene graph. In *2019 First International Conference on Graph Computing (GC)*, pages 45–50. IEEE, 2019.

[42] Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Erran Li Li, Ruohan Zhang, et al. Embodied agent interface: Benchmarking llms for embodied decision making. *Advances in Neural Information Processing Systems*, 37:100428–100534, 2024.

[43] Yikang Li, Tao Ma, Yeqi Bai, Nan Duan, Sining Wei, and Xiaogang Wang. Pastegan: A semi-parametric method to generate image from scene graph. *Advances in Neural Information Processing Systems*, 32, 2019.

[44] Ziyang Li, Saikat Dutta, and Mayur Naik. Iris: Llm-assisted static analysis for detecting security vulnerabilities. In *The Thirteenth International Conference on Learning Representations*, 2025.

[45] Ziyang Li, Jiani Huang, Jason Liu, Felix Zhu, Eric Zhao, William Dodds, Neelay Velingker, Rajeev Alur, and Mayur Naik. Relational programming with foundational models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10635–10644, 2024.

[46] Ziyang Li, Jiani Huang, and Mayur Naik. Scallop: A language for neurosymbolic programming, 2023.

[47] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023.

[48] Xingyu Lin, Yufei Wang, Jake Olkin, and David Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, pages 432–448. PMLR, 2021.

[49] Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. Llm+p: Empowering large language models with optimal planning proficiency, 2023.

[50] Fang Liu, Yang Liu, Lin Shi, Houkun Huang, Ruifeng Wang, Zhen Yang, Li Zhang, Zhongqi Li, and Yuchi Ma. Exploring and evaluating hallucinations in llm-powered code generation. *arXiv preprint arXiv:2404.00971*, 2024.

[51] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding dino: Marrying dino with grounded pre-training for open-set object detection, 2024.

[52] Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, Hanyu Lai, Xinyi Liu, Hanlin Zhao, Jiadai Sun, Xinyue Yang, Yu Yang, Zehan Qi, Shuntian Yao, Xueqiao Sun, Siyi Cheng, Qinkai Zheng, Hao Yu, Hanchen Zhang, Wenyi Hong, Ming Ding, Lihang Pan, Xiaotao Gu, Aohan Zeng, Zhengxiao Du, Chan Hee Song, Yu Su, Yuxiao Dong, and Jie Tang. Visualagentbench: Towards large multimodal models as visual foundation agents, 2024.

[53] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 852–869. Springer, 2016.

[54] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *ICLR*, 2024.

[55] Damiano Marsili, Rohun Agrawal, Yisong Yue, and Georgia Gkioxari. Visual agentic ai for spatial reasoning with a dynamic api, 2025.

[56] Theo X Olausson, Alex Gu, Benjamin Lipkin, Cedegao E Zhang, Armando Solar-Lezama, Joshua B Tenenbaum, and Roger Levy. Linc: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. *arXiv preprint arXiv:2310.15164*, 2023.

[57] Sundar Pichai, Demis Hassabis, and Koray Kavukcuoglu. Introducing gemini 2.0: Our new ai model for the agentic era, Dec 2024.

[58] Jianing Qian, Yunshuang Li, Bernadette Bucher, and Dinesh Jayaraman. Task-oriented hierarchical object decomposition for visuomotor control. *arXiv preprint arXiv:2411.01284*, 2024.

[59] Tianwen Qian, Jingjing Chen, Shaoxiang Chen, Bo Wu, and Yu-Gang Jiang. Scene graph refinement network for visual question answering. *IEEE Transactions on Multimedia*, 25:3950–3961, 2022.

[60] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[61] Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning. *arXiv preprint arXiv:2307.06135*, 2023.

[62] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos, 2024.

[63] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019.

[64] Xindi Shang, Donglin Di, Junbin Xiao, Yu Cao, Xun Yang, and Tat-Seng Chua. Annotating objects and relations in user-generated videos. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, pages 279–287. ACM, 2019.

[65] Xindi Shang, Tongwei Ren, Jingfan Guo, Hanwang Zhang, and Tat-Seng Chua. Video visual relation detection. In *Proceedings of the 25th ACM International Conference on Multimedia*, pages 1300–1308, 2017.

[66] Junyao Shi, Jianing Qian, Yecheng Jason Ma, and Dinesh Jayaraman. Composing pre-trained object-centric representations for robotics from" what" and" where" foundation models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15424–15432. IEEE, 2024.

[67] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749, 2020.

[68] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

[69] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 510–526. Springer, 2016.

[70] Tom Silver, Varun Hariprasad, Reece S Shuttleworth, Nishanth Kumar, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Pddl planning with pretrained large language models. In *NeurIPS 2022 foundation models for decision making workshop*, 2022.

[71] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems*, 34:251–266, 2021.

[72] Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Rin Metcalf, Walter Talbott, Natalie Mackraz, R Devon Hjelm, and Alexander T Toshev. Large language models as generalizable policies for embodied tasks. In *ICLR*, 2024.

[73] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[74] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[75] Renhao Wang, Jiayuan Mao, Joy Hsu, Hang Zhao, Jiajun Wu, and Yang Gao. Programmatically grounded, compositionally generalizable robotic manipulation. *arXiv preprint arXiv:2304.13826*, 2023.

[76] Yi Wang, Yinan He, Yizhuo Li, Kunchang Li, Jiashuo Yu, Xin Ma, Xinhao Li, Guo Chen, Xinyuan Chen, Yaohui Wang, et al. Internvid: A large-scale video-text dataset for multimodal understanding and generation. *arXiv preprint arXiv:2307.06942*, 2023.

[77] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023.

[78] Wenhao Wu, Zhun Sun, and Wanli Ouyang. Revisiting classifier: Transferring vision-language models for video recognition, 2023.

[79] Wenhao Wu, Xiaohan Wang, Haipeng Luo, Jingdong Wang, Yi Yang, and Wanli Ouyang. Bidirectional cross-modal knowledge exploration for video recognition with pre-trained vision-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6620–6630, 2023.

[80] Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9068–9079, 2018.

[81] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5410–5419, 2017.

[82] Hongwei Xue, Tiankai Hang, Yanhong Zeng, Yuchong Sun, Bei Liu, Huan Yang, Jianlong Fu, and Baining Guo. Advancing high-resolution video-language representation with large-scale video transcriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5036–5045, 2022.

[83] Jihan Yang, Shusheng Yang, Anjali W. Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces, 2024.

[84] Jingkang Yang, Wenxuan Peng, Xiangtai Li, Zujin Guo, Liangyu Chen, Bo Li, Zheng Ma, Kaiyang Zhou, Wayne Zhang, Chen Change Loy, and Ziwei Liu. Panoptic video scene graph generation, 2023.

[85] Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, et al. Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. *arXiv preprint arXiv:2502.09560*, 2025.

[86] Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-encoding scene graphs for image captioning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10685–10694, 2019.

[87] Hanlin Zhang, Jiani Huang, Ziyang Li, Mayur Naik, and Eric Xing. Improved logical reasoning of language models via differentiable symbolic programming. *arXiv preprint arXiv:2305.03742*, 2023.

[88] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. *arXiv preprint arXiv:2307.02485*, 2023.

[89] Li Zhang, Peter Jansen, Tianyi Zhang, Peter Clark, Chris Callison-Burch, and Niket Tandon. Pddlego: Iterative planning in textual environments. *arXiv preprint arXiv:2405.19793*, 2024.

[90] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pages 169–186. Springer, 2024.

[91] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data, 2024.

[92] Zhonghan Zhao, Wenhao Chai, Xuan Wang, Boyi Li, Shengyu Hao, Shidong Cao, Tian Ye, and Gaoang Wang. See and think: Embodied agent in virtual environment. In *European Conference on Computer Vision*, pages 187–204. Springer, 2024.

[93] Kaizhi Zheng, Xiaotong Chen, Odest Chadwicke Jenkins, and Xin Eric Wang. Vlmbench: A compositional benchmark for vision-and-language manipulation, 2022.

[94] Yiwu Zhong, Liwei Wang, Jianshu Chen, Dong Yu, and Yin Li. Comprehensive image captioning via scene graph decomposition. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 211–229. Springer, 2020.

[95] Luowei Zhou, Chenliang Xu, and Jason J Corso. Procnets: Learning to segment procedures in untrimmed and unconstrained videos. *arXiv preprint arXiv:1703.09788*, 2(6):7, 2017.

[96] Bin Zhu, Bin Lin, Munan Ning, Yang Yan, Jiaxi Cui, HongFa Wang, Yatian Pang, Wenhao Jiang, Junwu Zhang, Zongwei Li, Wancai Zhang, Zhifeng Li, Wei Liu, and Li Yuan. Languagebind: Extending video-language pretraining to n-modality by language-based semantic alignment, 2024.

Algorithm 1: Video Mask Propagation with New Object Discovery

Input: Video $V = \{I_0, \ldots, I_T\}$, mask generator $\mathcal{G}$, propagation model $\mathcal{P}$, state S
Input: Batch size B, thresholds: $\tau_{\text{iou}}, \tau_{\text{score}}, \tau_{\text{inner}}$, stride s
Output: Mask dictionary `video_segments` : `Dict[Frame][ObjectID] = Mask`

`video_segments` $\leftarrow \emptyset$, `now_frame` $\leftarrow 0$;
while *not* `saturated` **do**
  $I \leftarrow V$[`now_frame`]; `// Load current image frame`
  $\mathcal{M} \leftarrow \mathcal{G}(I)$; `// Generate masks for current frame`
  `// Find new object masks not yet propagated from this frame`
  $\mathcal{M}_{new}$ = filter($\mathcal{M}$, `video_segements[now_frame]`, $\tau_{\text{iou}}, \tau_{\text{score}}, \tau_{\text{inner}}$);
  `// Register the new masks as prompts`
  $\mathcal{P}_{\text{buf}}$.update($\mathcal{M}_{\text{new}}$);
  `// Prompt the mask generator with updated masks as initialize condition`
  $\mathcal{P}$.`reset_state`(S);
  foreach *(oid, frame_id, mask) in* $\mathcal{P}_{buf}$ **do**
    $\mathcal{P}$.`add_new_mask_prompt`($S, frame\_id, o, m$);
  end
  `// Propagate the mask prompts through the whole video`
  foreach $(f, \{o_i\}, \{\ell_i\}) \in \mathcal{P}.propagate(S)$ **do**
    if $f \notin$ `video_segments` **then**
      `video_segments`$[f] \leftarrow \{\}$;
    end
    foreach i **do**
      $m_i \leftarrow$ `binarize`(ℓ_i);
      `video_segments`$[I_f][o_i] \leftarrow m_i$;
    end
  end
  `// Find the next frame to process with least mask coverage`
  Compute coverage ρ_f over $f \in \{$`now_frame` $+ 1, \ldots T\}$;
  `saturated`, `now_frame` = check_saturation(ρ)
end
return `video_segments`

A ESCA-Video-87K

A.1 Video and Caption Source

We build upon the LLaVA-Video-178K dataset [91], which comprises diverse video sources and GPT-generated, detailed video descriptions.

ESCA-Video-87K contains a total of $87,045$ datapoints, each consisting of a video clip ranging from 0 to 30 seconds in length, along with its corresponding caption. The videos are drawn from ten primary sources, spanning a wide range of content—including egocentric and household activities, as well as crowd-sourced videos from YouTube. Specifically, these ten sources are HD-VILA-100M [82], InternVid-10M [76], VidOR [64], VIDAL (YouTube Shorts) [96], YouCook2 [95], Charades [69], ActivityNet [10], Kinetics-700 [38], Something-Something v2 [23], and Ego4d [24]. The captions were generated using GPT-4 with a dense frame sampling technique [91].

The two main contributions of ESCA-Video-87K are the inclusion of object trajectories and executable programmatic specifications. We leverage SAM2 [62] to generate object trajectories and design a custom algorithm to handle objects that do not appear in the first frame. To obtain the specifications, we use GPT-4 and develop a transcompiler that converts them into executable programs.

These two additional components enable fine-grained spatio-temporal alignment between the video content and the concepts expressed in the specifications [31]. Notably, the only source of supervision is the video itself; all other annotations are derived using multimodal large language models (MLLMs). We refer to this approach as *model-driven self-supervision*.

Algorithm 2: Bounding Box Prompt Buffer Class

Output: Prompt dictionary for propagation; object-to-class mapping

Initialize fields: `oid`, `frame2bboxes`, `frame2masks`, `valid_prompts`, `overlapped_prompts`;

// Add predicted bounding box and mask

Function `AddPrompt(`$frame\_id$, box, $mask$,`)`

  if *not valid(m)* **then**

    return

  end

  `frame2bboxes[`$frame\_id$`]`.append(box); `frame2masks[`$frame\_id$`]`.append($mask$)

end

Function `PopMostDensePromptFrame()`

  // Find frame f^* with largest mask area

  $f^* \leftarrow \arg\max_f \sum$ `frame2masks`$[f]$;

  // Assign object IDs starting from oid for each bbox in f^*

  foreach *box in frame2bboxes[*f^**]* **do**

    `valid_prompts[`f^*`][oid]` $\leftarrow$ box; oid += 1;

  end

  // Remove prompts from to process list

  `frame2bboxes`.pop(f^*); `frame2masks`.pop(f^*);

  return f^*, `valid_prompts`$[f^*]$

end

Function `RemoveDuplicatePrompts(`*video_segments*`)`

  foreach $frame\_id$ *in* *video_segments* **do**

    if $frame\_id$ *not in* *frame2masks* **then**

      continue

    end

    Compute IoU between predicted and prompt masks in $frame\_id$;

    // Filter prompts already covered

    Remove overlapping segments from `frame2bboxes` and `frame2masks`

  end

end

A.2 Object Trajectory Generation

We aim to generate object trajectories from videos using Segment Anything 2 (SAM2). A key challenge is discovering new objects that do not appear in the first frame, which is not natively supported by SAM2. To address this, we design an iterative algorithm that identifies the next frame most likely to contain a new object and propagates its mask throughout the video, as illustrated in Algorithm 1. To further leverage concepts generated by GPT-4, we extend this algorithm to incorporate arbitrary frame-based bounding box generators, such as Grounding DINO [51] and YOLO [26]. This extended version uses a prompt scheduler that prioritizes frames containing the highest number of grounded objects and iteratively propagates them across the video, as shown in Algorithm 2 and Algorithm 3.

Algorithm 3: Mask Generation via Frame-wise Bounding Box Grounding

Input: Grounding model $\mathcal{G}$, SAM predictor $\mathcal{P}$, SAM mask generator $\mathcal{M}$, video frames $\{I_0, \ldots, I_T\}$, label set $\mathcal{C}$
Input: Box and text thresholds τ_b, τ_t, target FPS, max propagation steps k
Output: Per-frame mask segments `video_segments`, object-to-class mapping `oid_pred`

Initialize `prompt_memory` $\leftarrow$ `PromptBuffer()`;
// Step 1: Grounding-based object detection
foreach *frame* I_f *in video* **do**
  bboxes = $\mathcal{G}(I_f, \mathcal{C})$;
  `video_boxes`$[f] \leftarrow$ bboxes ;
end
// Step 2: Generate masks for all bounding boxes
foreach $(f, \textit{bboxes}) \in \textit{video\_boxes}$ **do**
  masks = $\mathcal{M}(I_f,$ bboxes) ;
  foreach *(box, mask) in zip(bboxes, masks)* **do**
    `prompt_memory.add_prompt`$(f, box, mask)$
  end
end
// Step 3: Iterative propagation from dense prompt frame
f_{start}, `prompt` $\leftarrow$ `prompt_memory.pop_most_dense_fid_prompt()`;
`video_segments` $\leftarrow \emptyset$, $t \leftarrow 0$;
while $\textit{prompt} \neq \emptyset$ *and* $f_{start} \neq -1$ *and* $t < k$ **do**
  Reset predictor state $S \leftarrow S_0$;
  foreach $(f, \textit{frame\_prompt}) \in \textit{prompt\_memory.valid\_prompts}$ **do**
    foreach $(o, b) \in \textit{frame\_prompt}$ **do**
      Add prompt (o, b) at frame f into $\mathcal{P}$ with state S;
    end
  end
  // Propagate forward
  foreach $(f, \{o_i\}, \{\ell_i\}) \in \mathcal{P}.\textit{propagate}(S)$ **do**
    `video_segments`$[f] \leftarrow$ binarized masks $\{\ell_i > 0\}$ mapped to o_i
  end
  // Propagate backward
  foreach $(f, \{o_i\}, \{\ell_i\}) \in \mathcal{P}.\textit{propagate}(S, reverse = True)$ **do**
    `video_segments`$[f] \leftarrow$ binarized masks $\{\ell_i > 0\}$ mapped to o_i
  end
  `prompt_memory.remove_dup_prompt(video_segments)`;
  f_{start}, `prompt` $\leftarrow$ `prompt_memory.pop_most_dense_fid_prompt()`;
  $t \leftarrow t + 1$;
end
return `video_segments`

A.3 Concept Generation

To construct low-level supervision from high-level captions, we leverage GPT-4 to generate spatio-temporal specifications. To mitigate potential hallucinations from the language model, we design a compiler that verifies the validity of the generated programs. We use a few-shot prompt with three examples.

Prompt 1: Role Definition

```
You are a super user in logic programming.
You are also an expert at structured data extraction.
```

Prompt 2: General Task Instruction

```
You will describe the event length and location in
both natural language and fraction of the video.
```

```
The natural language description of the locations in the video
    can be: early, mid, late.
The natural language description of the durations of the event
    can be: long, medium, short
Examples of precise video locations: [1/4, 1/2], [2/3, 1].
Examples of event durations: 1/4, 2/3, 1.
```

Prompt 3: Few-shot Examples

Listing 1: "One of the few-shot example for concept extraction"

```
Caption: A man carries a child and walks to the left from
    behind a woman holding another child.
Video_ID: "0be30efe",
Action json:
{"video_id": "0be30efe",
 "sequential descriptions": [ "man A carry child B, women C
     hold child D, man A is behind women C", "man A walk", "man
      A at left" ],
  "time stamps":
    { "1": { "description": [ "man A carry child B", "women C
       hold child D", "man A is behind women C" ], "
       programmatic": [ "carrying(A, B)", "name(A, man)", "
       name(B, child)", "holding(C, D)", "name(C, women)", "
       name(D, man)", "behind(A, C)",  ], "duration": "short",
        "duration precise": "1/4", "video location": "early",
        "video location precise": "[0, 1/4]" }, "2": { "
       description": [ "man A walk" ], "programmatic": [ "walk
       (A)", ], "duration": "medium", "duration precise": "1/2
       ", "video location": "mid", "video location precise":
       "[1/4, 3/4]" }, "3": { "description": [ "man A at left"
        ], "programmatic": [ "left(A)" ], "duration": "short",
        "duration precise": "1/4", "video location": "late", "
       video location precise": "[3/4, 1]"}}}
```

Prompt 4: Concept Definition

```
Note all the relations are name, unary or binary.
A name relation takes in two arguments, the first is always a
    variable, and the second argument could be noun ("apple"),
    noun phrase ("ancient_building"), location("dark_forest"),
    etc. For example "name(A, "apple")" means the variable A
    refers to an apple. Please ensure no space occur in the
    second argument.
A unary relation takes in one variable as its argument. For
    example, close(A) means A is close to the camera.
A binary relation takes in two variables as its arguments. For
    example, above(A, B) means A is above B.
The entity in the binary and unary relation are variables in
    the form of capitalized letters (A, B).
The predicate of the unary relation can be adjectives, verbs,
    and name.
The predicate of the binary relation can be preposition, and
    verb.
Please include the name relation any time if applicable.
Please make the preposition and adjectives into separate two
    relation.
```

```
For each time stamp, please only describe the events that are
    happening at the same time, if any sequential events occur,
     put them into multiple time stamps.
For example, instead of 'person A enter from left, person A
    walk to center, person A move to couch' in one time stamp,
put it into three different time stamps: "person A enter from
    left", "person A walk to center", "person A move to couch".
Please only describe one single event in sequential description
     per time stamp.
Please use as many relations as possible to precisely describe
    the action.
Please generate the action json programs for the following
    captions in the following format:
{"actions": {caption_id: action json programs}}
IMPORTANT: Please REMOVE all new line characters and extra
    spaces in the generated json!
```

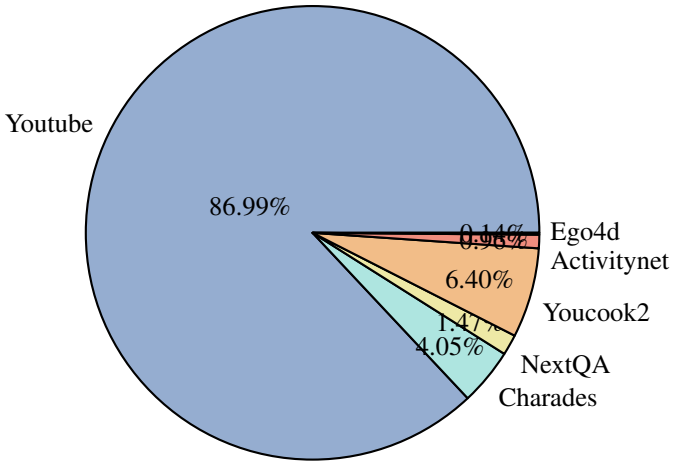
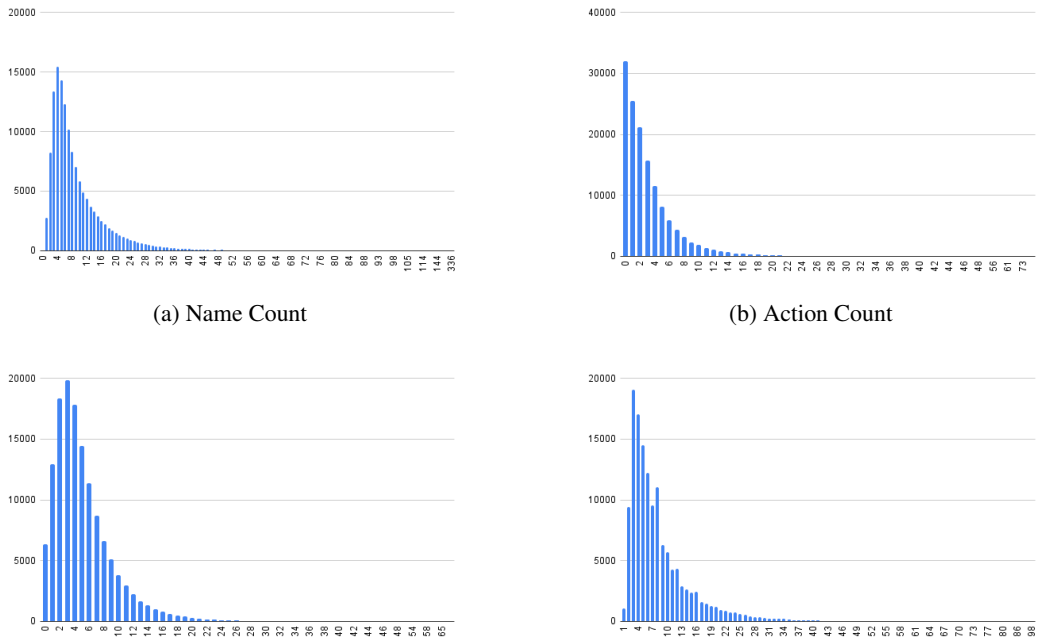


Figure 11: Video sources for ESCA-Video-87K.

A.4 Dataset Statistics

We present the statistics of ESCA-Video-87K through three perspectives: the composition of video sources, the complexity of extracted specifications, and the word clouds of associated concepts. We selected 0–30 second clips from the LLaVA-Video-178K dataset CITO, with the resulting source distribution shown in Figure 11. Our extracted spatio-temporal specifications exhibit considerable complexity. As illustrated in Figure 12, a single specification can contain multiple names, actions, relations, and events. In addition, we visualize the vocabulary diversity using word clouds for names, actions, and relations. In total, our dataset includes 220,905 unique names, 57,930 unique actions, and 35,415 unique relation keywords.



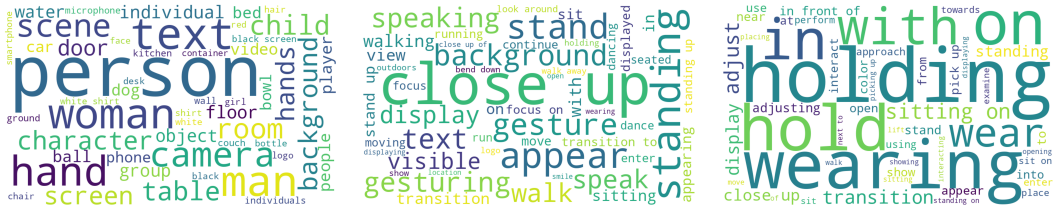
(a) Name Count

(b) Action Count

(c) Relation Count

(d) Event Count

Figure 12: Distribution of Names, Attributes, Relations, and Events



(a) Name (b) Action (c) Relation

Figure 13: Word cloud visualizations of the top-50 most frequent names, attributes, and relations.

B SGClip Training Details

We build our training pipeline following the LASER work [31], which uses a weakly-supervised approach to align video with its caption. Note that the caption used in SGClip is also generated from the video with multi-modal large language model (GPT-4o), and this leads to all labels are generated from the video itself, thus, we consider this approach as a model-driven self-supervised method. In this section, we elaborate on the details of how SGClip is trained and providing details in the training procedure.

B.1 Compute

All our experiments are carried out on a device with (1) 128 32-core Intel(R) Xeon(R) Gold 6338 CPU @ 2.00GHz (2) 10 NVIDIA H100 PCIe GPUs. SGClip takes 10 days to finetune on the $87K$ datapoints for 3 epochs.

B.2 Loss

We train the SGClip with three different losses: contrastive loss, temporal loss, and semantic loss.

Contrastive Loss. To help SGClip distinguish between concepts that appear in a video and those that do not, we adopt a contrastive loss design. For each video V, we pair it with its corresponding specification ϕ and randomly sample an unrelated specification ϕ' from the dataset. The objective is to maximize the alignment score between the matching pair (V, ϕ) toward 1, while minimizing the score between the mismatched pair (V, ϕ') toward 0.

A key challenge arises from the fact that specifications are exceptionally long—each containing an average of 8.24 events per datapoint—making it computationally expensive to ground the entire specification in the video. To address this, we introduce a chunked event training strategy. Each specification is divided into smaller chunks containing at most 3 events, and we align each chunk with the full video. When sampling mismatched pairs, we similarly draw from these smaller chunks to maintain alignment granularity and training efficiency.

Temporal Loss. To improve alignment accuracy between each event and its corresponding location in the video, we query GPT for a fine-grained estimation of the event's temporal span. For all satisfiable event sequences, we assign higher rewards to those aligned with the desired temporal location, and reduced rewards to those outside the target range.

Semantic Loss. We further incorporate common-sense negations to improve the model's understanding of what is unlikely to occur in a given scenario. For example, if a scene is outdoors, it is unlikely to contain a bed; if a person is "standing," they are unlikely to be "sleeping" at the same time; if a person is "holding" an object, it is improbable that the object is "hitting" the person simultaneously.

To introduce this notion of negation, we first identify the top $5,000$ most frequent keywords across names, actions, and relations. Using word vectors from SpaCy [28], we compute the 50 most semantically distant keywords within this subset for each current scenario. During training, we sample 5 keywords per category (name, action, relation) from these 50 distant terms to compute a semantic loss that penalizes implausible pairings.

B.3 Hyperparameters

We use a learning rate of 1×10^{-6} and a batch size of 2. The video is sampled at a target frame rate of 1 FPS. For the semantic loss, we sample 5 negative keywords per instance and set the semantic loss weight to 0.1. In the provenance setting for Scallop, we use difftopkproofs with a top-k value of 3 for proof extraction. We fine tune from the CLIP model with a total of 3 epochs, and we evaluate and ensemble on the best performing models from different epochs.

Model	Strategy	Subset					Avg
		Base	Common	Complex	Visual	Long	
InternVL-2.5-38B-MPO	Base	55.00	60.00	51.67	40.00	30.00	47.33
	+ GD	60.00	61.67	56.67	48.33	11.66	47.67
	+ ESCA	58.33	61.67	53.33	58.33	26.66	**51.66**
Gemini-2.0-flash	Base	56.67	46.67	48.41	36.67	15.00	40.68
	+ GD	55.00	53.33	50.00	38.33	6.00	40.53
	+ ESCA	56.67	41.67	43.33	46.67	21.66	**42.00**
Qwen2.5-VL-72B-Ins	Base	58.34	48.30	48.30	36.70	33.33	44.99
	+ GD	65.00	55.00	58.00	43.33	20.00	48.27
	+ ESCA	60.00	48.33	60.00	46.67	31.67	**49.33**
GPT-4o	Base	55.00	60.00	58.33	60.00	23.33	51.33
	+ GD	63.33	63.33	65.00	53.33	21.66	53.33
	+ ESCA	66.67	62.00	63.33	55.00	26.67	**54.67**

Table 1: Detailed performance on EB-Navigation, decomposed by subsets of tasks.

C Additional Experimental Results

C.1 EmbodiedBench: Navigation

We demonstrate that ESCA enhances agent performance in navigating to target objects on embodied benchmarks, as shown in Table 1. Across four base multimodal large language models, ESCA consistently outperforms both the base models and variants using only Grounding-DINO. We outline the design of the transfer protocol and provide qualitative studies in the following section.

Transfer Protocol: Grounding Dino The overall pipeline of ESCA with Grounding DINO consists of three main steps: concept extraction, object identification, and visual summarization and validation. Rather than performing full scene graph prediction, we directly use Grounding DINO to extract candidate target objects based on their name and attributes. Each concept input to Grounding DINO follows the format `<attribute> <object name>`, such as "a grey rectangular object." The ESCA pipeline then passes the augmented image and the newly constructed query back to the MLLM to predict subsequent actions. Note that, based on our study, Grounding DINO tends to generate false positive candidate objects, which leads to a performance drop on the long-horizon subset. This is the only task subset where the target object is not visible in the initial scenario. The box threshold for grounding DINO is 0.2, and the text threshold is 0.1.

Transfer Protocol: ESCA The overall pipeline of ESCA with SGClip consists of four main steps: concept extraction, object identification, scene graph generation, and visual summarization and validation. We begin by using Grounding DINO to extract candidate target objects and their related objects based on object names. Next, SGClip predicts the attributes and relationships of the identified objects and aggregates predictions across names, attributes, and relationships. To ensure precision, we select only the top-1 object identified as the target and apply a confidence threshold of 0.3. For Grounding DINO, we use a box threshold of 0.1 and a text threshold of 0.1.

Qualitative Studies

We present two tasks to qualitatively analyze performance on the EB-Navigation environment. Each task is solved using three configurations: GPT-4o alone, GPT-4o augmented with Grounding DINO, and GPT-4o augmented with ESCA. Compared to GPT-4o alone, both Grounding DINO and ESCA provide additional information that aids task completion. However, ESCA results in significantly fewer false negatives than Grounding DINO, leading to improved performance on long-horizon tasks.

Figure 14: We present two qualitative examples comparing the original MLLM (GPT-4o), its variant augmented with Grounding DINO, and the full ESCA pipeline. Our observations show that ESCA improves the precision and quality of target object recognition.

Model	Strategy	Subset					Avg
		Base	Common	Complex	Visual	Spatial	
InternVL-2.5-38B-MPO	Base	10.42	25.00	20.83	27.78	12.50	19.31
	+ YOLO	14.58	14.58	25.00	19.40	22.92	19.30
	+ ESCA	31.25	21.00	14.58	27.78	27.08	**24.30**
Gemini-2.0-flash	Base	10.42	10.42	8.33	11.11	18.75	11.81
	+ YOLO	14.60	8.30	14.60	13.90	31.30	16.54
	+ ESCA	18.75	21.00	20.80	22.22	27.08	**21.94**
Qwen2.5-VL-72B-Ins	Base	2.08	6.25	8.33	4.16	2.78	4.72
	+ YOLO	18.80	20.80	4.20	8.30	14.60	13.34
	+ ESCA	18.80	10.41	16.67	16.67	10.42	**14.59**
GPT-4o	Base	16.67	25.00	20.83	35.41	19.44	23.47
	+ YOLO	39.60	29.20	29.20	19.40	25.00	28.48
	+ ESCA	33.33	31.25	37.50	38.89	31.25	**34.44**

Table 2: Detailed performance on EB-Manipulation, decomposed by subsets of tasks.

C.2 EmbodiedBench: Manipulation

EB-Manipulation evaluates an agent's ability to perform fine-grained object manipulation using visual input and language instructions. Unlike typical embodied AI settings, it provides full 3D spatial information (X, Y, Z coordinates) for all objects, removing the need for spatial inference and shifting the focus to semantic grounding and planning.

To construct structured object representations, we use the provided coordinates to generate point-based prompts for SAM 2.1, which produces segmentation masks that are converted into bounding boxes. Each object is annotated with a unique integer label starting from 1. we then prompt a vision-language model (VLM) to generate textual descriptions in the form <color> <object> (e.g., "yellow star"), using handcrafted templates to ensure consistency and match the number of detected objects.

Since the VLM outputs are unordered, we use ESCA, which treats predicted entities as categorical labels and outputs probability distributions over them for each bounding box. To resolve the correspondence between entities and boxes, we apply an entropy-based matching algorithm (Figure 3), which iteratively assigns the most confident (lowest-entropy) pairings while enforcing uniqueness constraints. The resulting assignments are used to generate a structured natural language description (e.g., "Object 1 is a yellow star"), which is appended to the original prompt and passed to the VLM to support precise and context-aware manipulation planning.

Qualitative Studies We present two tasks to qualitatively analyze performance on the EB-Manipulation environment. Each task is solved using three configurations: GPT-4o alone, GPT-4o augmented with YOLO, and GPT-4o augmented with ESCA. Compared to GPT-4o alone, both YOLO and ESCA provide additional information that aids task completion. However, ESCA results in more precise recognition than using YOLO alone.

Error Analysis As shown in Figure 17, we conducted an error analysis on the EB-Manipulation task. For each subtask category, we randomly sampled 10 erroneous tasks from both the GPT-4o baseline and GPT-4o augmented with ESCA. The results show that ESCA significantly reduces the perception error component.

[2]The three top-level error types are Perception, Reasoning, and Planning. The second-level errors are Hallucination, Wrong Recognition, Spatial Understanding, Spatial Reasoning, Reflection Error, Inaccurate Action, and Collision. For clarity, the figure uses these acronyms to label the different error types.

Algorithm 4: Entropy-Guided Assignment of Entity Names to Objects

Input: Objects $O = \{o_1, \ldots, o_n\}$;
Entities $E = \{e_1, \ldots, e_m\}$ (with possible repetitions);
Assignment scores $\mathcal{S} : O \times E \to \mathbb{R}$, $\mathcal{S}(o, e)$ denotes the score of assigning entity e to object o
Output: Assignment map $A : O \to E$, assigning one entity name to each object

Initialize assignment map $A \leftarrow \emptyset$;
Initialize name availability counter $C : E \to \mathbb{N}$. how many times each entity name appears in E;
while *there exists unassigned object* $o \in O$ **do**
  foreach *unassigned object* $o \in O$ **do**
    Let $D_o = \{\mathcal{S}(o, e) \,|\, C[e] > 0\}$; // Available assignment scores
  end
  Select object $o^* = \arg\min_o \text{Entropy}(D_o)$; // Most confident object
  Let $n^* = \arg\max_{e \in E : C[e] > 0} \mathcal{S}(o^*, e)$; // Highest scoring available name
  Assign $A[o^*] \leftarrow n^*$ and decrement $C[n^*] \leftarrow C[n^*] - 1$;
end
return A;

Figure 15: Entropy-guided assignment algorithm that maps entity names to objects based on assignment scores and availability. At each step, it selects the object with the lowest entropy over valid assignments, then assigns the highest scoring available entity. This process continues until all objects are assigned.

	k=1		k=5		k=10	
Model	Prec	Rec	Prec	Rec	Prec	Rec
SGClip-CLIP	0.469 (75)	0.085 (75)	0.321 (70)	0.250 (70)	0.246 (70)	0.353 (70)
SGClip	**0.495 (45)**	**0.087 (60)**	**0.350 (45)**	**0.270 (75)**	**0.278 (45)**	**0.385 (55)**

Table 3: Maximum test **precision**@k and **recall**@k on the VidVRD dataset, evaluated every five epochs during 50 epochs of finetuning. Each cell shows the score followed by the epoch at which that score was achieved.

C.3 Downstream Task: Scene Graph Generation

We further analyze the abilities of SGClip by applying it to the task of relation tagging of scene graphs. In this task, the ground truth bounding boxes of objects are provided, and the objective is to correctly classify their respective object class as well as the binary predicates between them. To demonstrate SGClip on this task, we use the VidVRD [65] dataset, comprised of 1,000 videos with 35 object categories and 132 predicate categories.

For our evaluation, we finetune two versions of SGClip on the VidVRD training set for 75 epochs: the first is the SGClip architecture initialized to the original CLIP backbone, denoted as SGClip-CLIP; the second is SGClip. The only difference between the two is that the latter has had training from the neurosymbolic learning pipeline denoted in 3.3. We next test these models on 196 videos from the test (4 are ignored for causing an OOM error on a single H100 in both models.) This test evaluation happens every five epochs; we report the maximum test precision@k and recall@k over all test evaluations for each evaluation in table 3. As the table shows, SGClip outperforms SGClip-CLIP across both metrics for $k = 1, 5, 10$, demonstrating that the proposed neurosymbolic training pipeline provides an advantageous initialization for relational tagging finetuning on VidVRD.

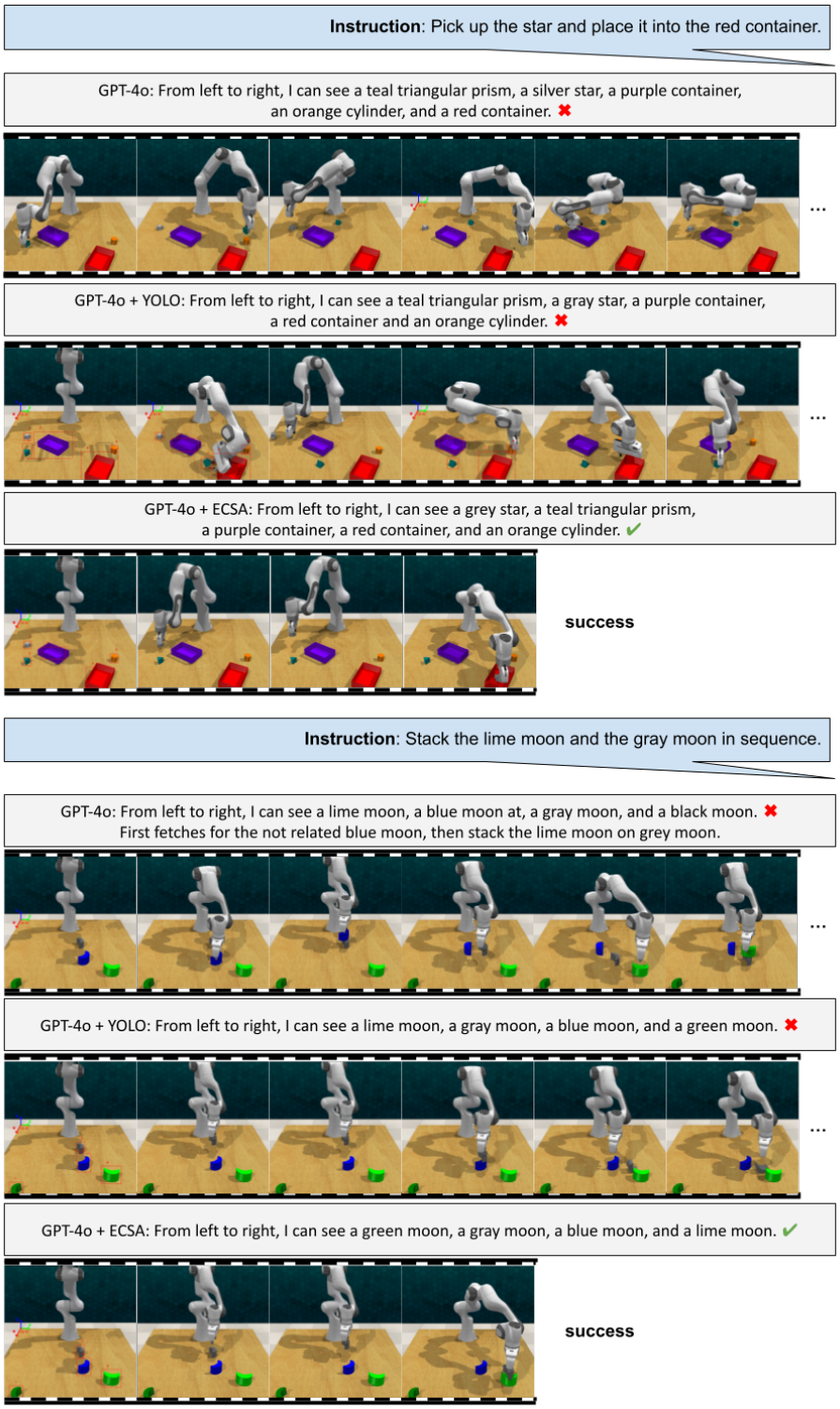


Figure 16: We present two qualitative examples comparing the original MLLM (GPT-4o), its variant augmented with YOLO, and the full ESCA pipeline. Our observations show that ESCA improves the precision and quality of scene graph recognition.

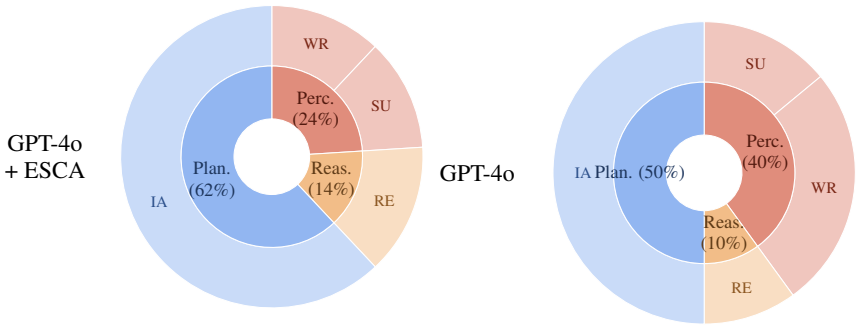


Figure 17: Error decomposition[2] of GPT-4o with and without ESCA, based on manual inspection of 50 EB-Manipulation tasks, with 10 randomly sampled from each subtasks. As we can see, ESCA has reduced perception error by a large margin.

Category	Model	ActivityNet Accuracy (%)
Zero-shot	SGClip	76.34
	CLIP	74.37
	BIKE	80.00
	Text4vis	77.40
	ResT	26.30
	E2E	20.00
Few-shot (1%)	SGClip	80.10
	CLIP	78.79
Few-shot (5%)	SGClip	86.05
	CLIP	80.02
Fully finetuned	InternVL-6B	95.90

Table 4: ActivityNet accuracy for zero-shot, few-shot, and fully finetuned models. Zero-shot includes both external baselines and our models evaluated without training.

C.4 Down-stream Task: Action Recognition

To evaluate the generalization and finetunability of our model beyond structured scene graph understanding, we consider the task of action recognition, where each video is annotated with a single activity label. We use a combined version of ActivityNet 1.2 and 1.3, which together span 200 unique action classes and approximately 20,000 untrimmed videos across training, validation, and test splits. The action categories range from simple atomic activities such as swimming and rock climbing to more complex, composite tasks like starting a campfire or performing a basketball layup drill.

Similar to Embodied Bench, we adopt a transfer protocol tailored to the action recognition setting. For each second of the video, we generate the top-4 predicted actions along with their confidence scores. We then apply thresholding to discard low-confidence predictions, followed by temporal smoothing to aggregate results into a single continuous segment per video. The final action label is selected based on a weighted combination of its average confidence score and temporal frequency. The output is a single scored segment per video, defined by the merged temporal boundaries of the selected label.

The results, summarized in Table 3, demonstrate the strong fine-tuning capability of ESCA under varying supervision levels. ESCA consistently outperforms CLIP across 0%, 1%, and 5% training subsets, highlighting its superior ability to recognize fine-grained actions. Remarkably, ESCA achieves 92.0% accuracy using only 5% of the training data (approximately 800 videos), approaching the performance of InternVL-6B, which is trained on the full dataset. This indicates that ESCA generalizes effectively from limited supervision, whereas InternVL-6B's advantage stems in part from its exposure to the entire training corpus.