# MultiPLY: A Multisensory Object-Centric
# Embodied Large Language Model in 3D World

Yining Hong[2,3], Zishuo Zheng[1], Peihao Chen[1], Yian Wang[1], Junyan Li[1], Chuang Gan[1,3]
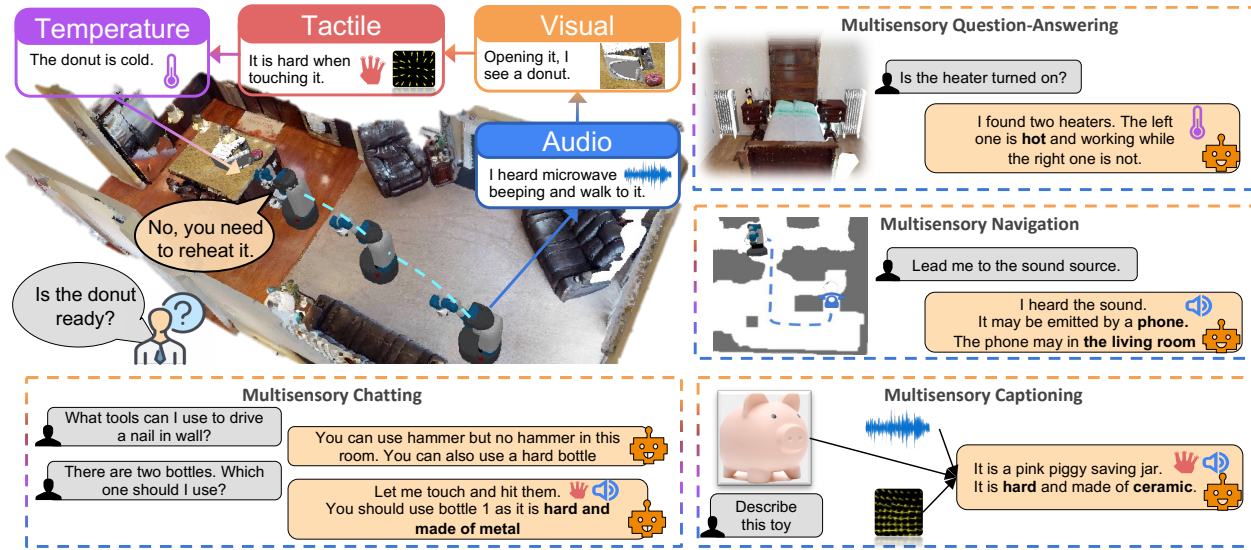
[1]UMass Amherst, [2] UCLA, [3]MIT-IBM Watson AI Lab

Figure 1. We propose MultiPLY, a multisensory embodied LLM that encodes object-centric multisensory representations (*e.g.,* visual, audio, tactile, and thermal), by deploying an embodied agent to engage with the 3D environment. MultiPLY excels at multiple tasks including multisensory captioning, question answering, dialogue, manipulation, navigation, tool use, task decomposition, and so on.

## Abstract

*Human beings possess the capability to multiply a mélange of multisensory cues while actively exploring and interacting with the 3D world. Current multi-modal large language models, however, passively absorb sensory data as inputs, lacking the capacity to actively interact with the objects in the 3D environment and dynamically collect their multisensory information. To usher in the study of this area, we propose MultiPLY, a multisensory embodied large language model that could incorporate multisensory interactive data, including visual, audio, tactile, and thermal information into large language models, thereby establishing the correlation among words, actions, and percepts. To this end, we first collect Multisensory Universe, a large-scale multisensory interaction dataset comprising 500k data by deploying an LLM-powered embodied agent to engage with the 3D environment. To perform instruction tuning with pretrained LLM on such generated data, we first encode the 3D scene as abstracted object-centric representations, and then introduce action tokens denoting that the embodied agent takes certain actions within the environment, as well as state tokens that represent the multisensory state observations of the agent at each time step. In the inference time, MultiPLY could generate action tokens, instructing the agent to take the action in the environment and obtain the next multisensory state observation. The observation is then appended back to the LLM via state tokens to generate subsequent text or action tokens. We demonstrate that MultiPLY outperforms baselines by a large margin through a diverse set of embodied tasks involving object retrieval, tool use, multisensory captioning, and task decomposition.*

# 1. Introduction

Human beings inhabit an extraordinary multisensory world - one in which we constantly explore and interact with the 3D environment, collecting and analyzing a mélange of sensory data to accomplish various tasks [56]. Picture yourself situated within an embodied environment depicted as Figure 1. To reason about the question "is the donut ready for eating", you begin by hearing the microwave beep. Subsequently, you decide to investigate whether the donut is inside the microwave. Once you locate the donut, you may touch it, sensing its hardness and coldness, leading you to the conclusion that the donut is not yet ready.

Existing multi-modal large language models (*e.g.,* LLaVA [39], Flamingo [1], BLIP-2 [37], PaLM-E [12]) excel at numerous vision-language tasks. However, they mainly focus on 2D scene understanding, struggling to reason about and interact with 3D environments. Recent works such as 3D-LLM [32] take preliminary steps to encode holistic 3D point clouds as inputs and show impressive results on 3D reasoning tasks, while suffering from expensive training and inefficient reasoning for objects. More importantly, these models fall short of the ability to capture multisensory information that goes beyond vision and language.

Efforts have been made to bind representations from different modalities [28], and adapt them to pre-trained LLMs [31, 40]. However, they often focus on a single object [30] or 2D image [28], unable to encode a large 3D environment and *interact* with the 3D embodied environment. For example, to address a question illustrated in Figure 1, a human would need to touch the donut to sense its softness and temperature, a capability well beyond the current scope of multi-modal LLMs.

Looking ahead, challenges inevitably exist for building embodied multisensory large language models. The first challenge resides in the paucity of multisensory interaction data for training such an LLM. The next challenge lies in the appropriate representations of the 3D scenes and multisensory information of the objects. Humans could hold a coarse impression of the scene by abstracting the scene as an object-centric representation and attending to the object details when further interacting with the objects. It's essential for LLMs to also be able to flexibly switch between an abstracted object-centric representation and detailed multisensory information of the objects. Lastly, existing LLMs are not tailored for instruction tuning with interaction data. They often take passive data as inputs and generate single-step outputs, incapable of connecting the words, actions, and percepts to engage with an embodied environment.

To this end, we propose MultiPLY, a multisensory embodied LLM that could encode multisensory object-centric representations, including visual, audio, tactile, and thermal information, by deploying an LLM-powered agent to engage with the 3D environment. We first collect Multisen-

sory Universe, a large-scale multisensory dataset comprising 500k data collected by an agent actively engaging with 3D embodied environments. We utilize the 3D environments from Habitat-Matterport 3D (HM3D) dataset [46], and enrich the environments by adding interactive objects with rich sensory data from ObjectFolder [20] and Objaverse [11]. We prompt ChatGPT to create the input and output data of tasks ranging from multisensory captioning, question answering, dialogue, manipulation, task decomposition, and so on. An embodied agent explores the environment and interacts with the objects in the environment to get multisensory observations of these tasks.

To perform instruction tuning on such generated data, we first encode the 3D scene as an abstracted object-centric representation, informing the LLM of what objects are in the scene. We further devise an additional set of action tokens such as NAVIGATE, OBSERVE (for obtaining object point cloud), TOUCH (for tactile and thermal information), HIT (for getting the impact sound) to denote that the agent takes the actions to explore the environment and interacts with the objects. By interacting with the objects, more detailed multisensory information could be unveiled as outcomes of the actions and encoded via a set of state tokens. All sensory observations are encoded by different sensor encoders and connected to the LLM using sensor-to-image adapters.

In the inference time, MultiPLY could generate a series of action tokens through the LLM, instructing the agent to take the action and receive the outcome of the action as the next-state multisensory observation. The observation is then appended back to the LLM, enclosed by a set of state tokens, facilitating the next-step generation. Our MultiPLY, trained on Multisensory Universe, outperforms baseline models by a large margin on object retrieval, tool use, multi-modal captioning, and task decomposition.

To sum up, the contributions of this paper are:

- We propose Multisensory Universe, a large-scale multisensory dataset comprising 500k data collected by an agent engaging with the 3D embodied environment, covering a diverse set of tasks involving multisensory captioning, question answering, dialogue, manipulation, task decomposition, and so on.
- We propose MultiPLY, a multisensory embodied LLM that could encode multisensory object-centric representations with a novel set of action tokens and state tokens for the end-to-end instruction tuning of a pre-trained LLM.
- Experimental results on object retrieval, tool use, multisensory captioning, and task decomposition show that MultiPLY outperforms baselines by a large margin.

## 2. Related Works

**Multisensory Learning** Multisensory learning aims to learn from information from different sensors, including cameras, microphones, tactile sensors, etc. For visual-audio

learning, the datasets collecting visual-audio pairs in real-world [10, 43] or rendering sounds in simulators [6, 8, 17] promote the development of this field of research. Earlier works seek to combine audio and visuals information for audio-visual event localization [27, 57, 60, 61], sound source localization in visual frame [14, 16, 19, 66, 67], visual-guided sound editing [7, 18, 25], and visually-aligned sound generation [9, 15, 44, 50]. As for visual-tactile learning, many works focus on building realistic tactile simulation system [41, 58] or collecting tactile data of real objects [23, 24]. With these tactile data, researchers combine visual and tactile data for cross-modal retrieval [3, 21], robotic manipulation [4, 5, 36], and 3D reconstruction [48, 49, 52]. Different from the previous works, our MultiPLY aims to combine visual, audio, tactile, and thermal information in an interactive 3D environment for diverse embodied tasks.

**Multi-modal Large Language Models** LLMs [42, 53, 55, 65] demonstrate prowess across numerous domains. Recent works [1, 37, 39] attempt to empower LLMs with visual understanding ability using large-scale image-text pair data and apply the trained models on downstream tasks like visual question-answering, image captioning, and multi-modal dialogue. Researchers [32, 51, 62, 64] also focus on incorporating 3D visual information into LLMs to empower spatial reasoning abilities. In addition to incorporating visual information into LLMs, recent works [30, 31] attempt to enable LLMs to understand multi-modal information. AnyMAL [40] presents a unified model that aligns multi-modal information including text, image, video, audio, and IMU motion reading. However, these works process passive information rather than actively interact with the environment. In contrast, our work focuses on an embodied large language model, which could actively interact with the multi-modal 3D world by navigating in the environment, touching objects to get tactile and thermal information, hitting objects to get impact sound, etc.

# 3. The Multisensory-Universe Dataset

In this section, we illustrate the process of collecting the Multisensory-Universe dataset. As presented in Figure 2, we begin by explaining how we input interactive objects into the scene to construct object-centric 3D scenes for our dataset in Section 3.1. Subsequently, we outline the methodology for obtaining sensor data from these objects in Section 3.2. Moving on to Section 3.3, we describe the deployment of an embodied agent tasked with proposing tasks and exploring the environment to solve them. The resulting interaction data are collected as paired interaction-language data, which serves as training input for the LLM.
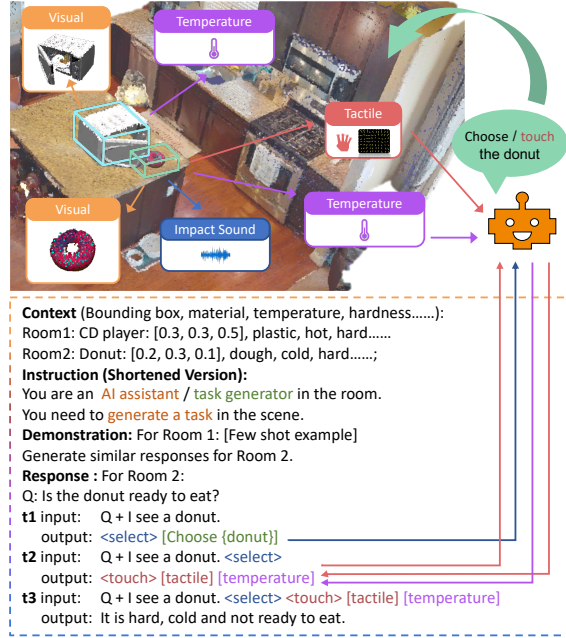


Figure 2. **Multisensory-Universe Generation Pipelines**. We first add a set of new interactive objects in the embodied environments, then prompt ChatGPT to generate diverse tasks about the environment. An embodied agent interacts with the objects to retrieve the multisensory information and construct interaction data.

## 3.1. Inputting Interactive Objects into 3D Scenes

We build our scenes on top of the Habitat-Matterport 3D (HM3D) semantics dataset [46, 63], which has 216 3D spaces and 3,100 rooms within those spaces. However, the existing objects in HM3D scenes, with insufficient sensor data and limited diversity, are not interactive in Habitat-sim [47]. Thus, we propose to add new interactive objects to the scenes, allowing agents to interact with them using Habitat-sim. The objects we add to the scenes are from two sources: 1) ObjectFolder [20, 22], which contains 1k object meshes, with impact sounds of these objects stored in implicit neural fields, and annotated with object materials. 2) Objaverse [11] is a universe of 800K 3D objects spanning rich categories. We select the objects that could appear in indoor scenes.

We ask ChatGPT [42] to choose 1-10 new objects from ObjectFolder and Objaverse, and generate the proper bounding boxes for these newly-added objects. ChatGPT is also required to specify objects' material categories (*e.g.,* ceramic, plastic, steel) and properties(*e.g.,,* deformation, elasticity hardness), as well as temperature labels (*e.g.,* whether the objects are hot, cold, or the same as room temperature). Our prompt to GPT contains all existing objects in HM3D scenes and their bounding boxes, as well as several preferences: 1) Select some similar objects. For example, choose two bottles of similar appearances and specify one of them as plastic and the other one as steel. In this way,

information from different sensors needs to be collected to resolve the ambiguity. 2) Select objects that are compatible with the environment and can be utilized together for interesting tasks. For instance, in a kitchen environment, we could put ingredients and tools for cooking. We also give some few-shot prompting examples to GPT.

## 3.2. Object Sensor Data Acquisition

We illustrate how we collect sensor data of added objects.
- **Tactile** We use DiffTactile [2] which leverages MLS-MPM [33] to simulate rigid, elastic, elasto-plastic objects. We put meshes of added objects into DiffTactile, which uses the bubble gripper with several position markers to touch the objects at pre-defined positions. The tactile readings are the initial and final positions of the markers, which represent how much the bubble deforms.
- **Ambient Sound** Each object could emit ambient sound to facilitate navigation or reasoning, or serve as cues for informing the agents what's going on in the environment. We prompt ChatGPT to match the sounds from AudioSet [26] with the semantic labels of the added objects. Given the Audioset description, ChatGPT needs to select objects in the candidate list that are possible to make this sound.
- **Impact Sound** Impact sound represents the sound that we hear when we strike or hit an object, which is crucial for identifying the material of an object. We get the impact sounds of ObjectFolder objects by querying their implicit sound fields given a hitting position and a force.
- **Temperature** Given the temperature label of the object, we ask ChatGPT for a proper temperature of each object.

## 3.3. Embodied Agents for Data Collection

Inspired by [59], we utilize LLM-powered embodied agents to collect the data in the constructed scenes. We first prompt ChatGPT to propose tasks. Then we place an embodied agent to interact with the objects in 3D environments to perform the task and collect interaction data.

**Generating Task Proposals** We follow the box-demonstration-instruction-based prompting method proposed by [32], and prompt ChatGPT to generate tasks. In addition to the ground-truth bounding boxes of objects, we also input the ground-truth materials, deformability, and hardness, as well as the ground-truth temperature labels of all objects. ChatGPT is provided with a list of actions to be performed in the environment. Then it generates specific tasks requiring interactions with objects, a sequence of words representing pseudo ground-truth actions, and language reasoning outputs which are deduced from the ground-truth feedback labels of the objects (note that ChatGPT has access to all material and temperature labels, so that it could generate a sentence like "it feels cold" after the "touch" action). We cover a diverse set of tasks including multisensory captioning, question answering,

embodied dialogue, navigation, object manipulation, tool use, rearrangement, task decomposition, and so on. We append all prompts in Supplementary Material.

**Interaction Data Collection** The embodied agent first randomly explores the environments to collect initial RGBD environment data. Given the actions, the agent executes the actions to interact with the objects in the environment and obtains the sensory feedback. For example, when the action is "touching an object", the agent returns the tactile and temperature readings of it. We store all the interaction results of the actions. From one interaction, we could incrementally construct several input-output data, denoting the interaction at different steps, as shown in Figure 2.

## 4. MultiPLY

In this section, we introduce the MultiPLY framework. As in Figure 3, we first encode the scene as an abstracted object-centric representation, while multisensory details of objects are unveiled only when the agent executes an action and interacts with them. We devise a set of action tokens denoting the actions of agents to interact with the environment. Interaction results are appended back to the LLM via state tokens to generate subsequent text or action tokens.

### 4.1. Object-Centric Scene Representations

Our model first takes the features of the 3D environment explored by the agent as inputs to form an initial impression of what the scene looks like. We follow 3D-LLM and utilize 2D features to construct 3D scene features, so that the visual features could be seamlessly fed into a pre-trained vision-language model without adaption. However, the point cloud encoding of 3D-LLMs makes it hard for LLMs to process thousands of points at a time. Alternatively, when humans explore a 3D environment, we abstract over the scene and roughly form an idea of objects and their locations without remembering all the details. Likewise, we propose to represent the environment as an abstracted object-centric representation. We use concept graphs [29] powered with a CLIP [45] encoder to first encode the objects in the observed images, and fuse the outputs in images to 3D by multi-view association. We also add position embeddings to the visual features of objects. We finally get $\mathcal{O} \times 1024$ features as an abstracted object-centric scene representation, where $\mathcal{O}$ is the number of objects. If there's an ambient sound emitted by an object in the 3D environment, we encode the sound using the CLAP [13] audio encoder and get a 1024-dim feature. The object-centric scene representation and ambient sound representation serve as the initial inputs to the LLM, enclosed by tokens as <SCENE>, </SCENE> and <AMBIENT_SOUND>, </AMBIENT_SOUND>.
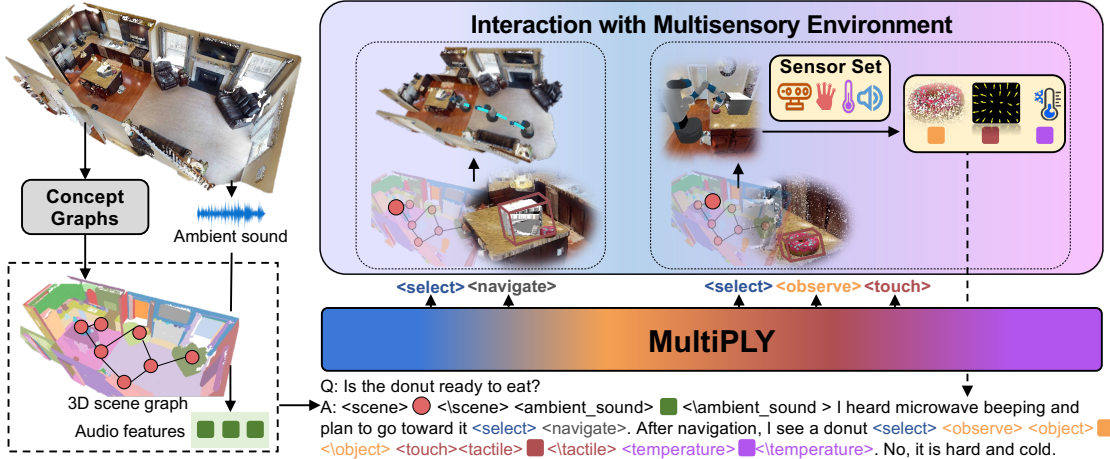
Figure 3. **Overview of our MultiPLY.** We first encode the scene as an abstracted object-centric representation, while multisensory details of objects can only be unveiled when the agent executes an action and interacts with them. We devise a set of action tokens denoting the actions of agents to interact with the environment. The interaction results are appended back to the LLM via state tokens.

## 4.2. Action Tokens

We devise a set of action tokens that denote the agent's interaction with the environment, which are listed below:

- **<SELECT>** token selects an object to interact with. The object is chosen by the attention between the language features (*i.e.*, the last hidden state of the LLM of the SELECT token), and the CLIP visual features of the objects in the environment. It selects the object with the maximum attention score.
- **<NAVIGATE>** token asks an agent to navigate to the selected object. Note that the navigation action could be executed by any pre-defined pathfinder module and is not the research focus of this paper.
- **<OBSERVE>** token asks an agent to scrutinize an object that is chosen and get the object details (in the form of the detailed point cloud of the object).
- **<TOUCH>** token allows the agent to touch the object that is chosen, to get the tactile and temperature information.
- **<HIT>** token allows the agent to hit the chosen object to get the impact sound.
- **<PICK-UP>**, **<PUT-DOWN>** tokens enable the agent to pick up or put down a chosen object.
- **<LOOK-AROUND>** token allows the agent to rotate its head and get nearby objects.

## 4.3. State Tokens

We devise another set of state tokens to feed the interaction results back to the LLM.

- **<OBJECT>** encodes the obtained object points when the agent <OBSERVE>s an object. Specifically, we get the 3D features aggregated from 2D CLIP features [32] and add position embeddings to the 3D features. We build $\mathcal{N} \times 1024$ object point cloud features where $\mathcal{N}$ is the number of points.

- **<IMPACT_SOUND>** encodes the obtained impact sound when the agent <HIT>s an object. We use CLAP audio encoder to encode the sound and get 1024-dim impact sound representation. Since the CLAP features are not aligned with the LLM, we use a sound projector (one linear layer) to map to the feature space of the LLM.
- **<TACTILE>** encodes the obtained tactile information when an object is being <TOUCH>ed by an agent. We transform the tactile reading as a heatmap and use CLIP to encode the heatmap. We mean-pool over the patches and get 1024-dim temperature features. We use a tactile projector (one linear layer) to map to the feature space of the LLM.
- **<TEMPERATURE>** encodes the obtained temperature. We transform the temperature reading as a heatmap and use CLIP to encode the heatmap. We mean-pool over the patches and get 1024-dim temperature features. We use a temperature projector (one linear layer) to map to the feature space of the LLM.

## 4.4. Training & Inference

**Model Architecture** We use LLaVA [38] as our backbone multi-modal large language model. Since our visual features have been aligned to the same embedding space as LLaVA using ConceptGraphs [29], we could directly use LLaVA's vision-to-language projector without pretraining on vision-language data. For other sensor modalities, we leverage a lightweight adapter, which is a one-layer linear projector to project the sensor features into the text token embedding space of LLaVA.

**Modality Alignment** As stated above, the tactile, sound, and temperature representations are not aligned with the language features. In the first stage, we train the sensor-to-language adapter for multisensory feature alignment. For audio-language alignment, we use AudioSet [26] and Au-

dioCaps [34]. For impact sound, tactile, and thermal data, we use ChatGPT to generate a one-sentence caption describing the material and the alignment between each sensor modality and language. We freeze the weight of the image encoder and the LLM for faster convergence and maintenance of language reasoning abilities.

**Instruction tuning with Multisensory Universe** In the second stage, we tune LLaVA with our multisensory dataset. Our training loss consists of two parts. The first one is the LLM loss which is the same as the original LLaVA model. We add one more loss that forces the model to select the right object to attend to. Specifically, we calculate the attention between the last hidden state of the LLM of the SELECT token, and each abstracted object feature. The feature goes through a Sigmoid layer, and is optimized with a binary cross entropy (BCE) loss. We unfreeze the whole model for the training of this stage. We use FSDP on 128 V100 GPUS for efficient training.

**Inference** At the inference time, our MultiPLY first takes the task prompt and abstracted scene representation as inputs and generates subsequent tokens. Once an action token is generated, an embodied agent is instructed to take the action in Habitat-sim [47] and interact with the environment. The observation outcome of the agent is sent back to the LLM as inputs via state tokens. The LLM further generates next tokens based on the current state inputs.

## 5. Experiments

After training on our collected Multisensory Universe, we perform an evaluation in the simulator, where an agent could actually interact with the environment when the action tokens are generated by MultiPLY. Then, the LLM waits for the agent to complete the actions and send back the observations via state tokens to generate the next token. We provide four experimental settings: object retrieval, tool use, multisensory captioning, and task decomposition, and provide detailed task descriptions, baselines, and analysis for each task. We ensure that no scenes and objects in the Multisensory Universe appear in the evaluation setup. Due to space limits, we attach more ablative studies in the Supplementary Material, where we experiment with each possible combination of sensory inputs from different modalities, with or without interaction with the environment.

### 5.1. Object Retrieval

**Task Decription** We devise the object retrieval task where several similar objects are present in the 3D scene, and the agent needs to use multiple sensor data to retrieve the correct object. For example, the task input could be like "retrieve the soft paper cup with hot water", while there could be distracting objects like "hard paper cup with hot water", "soft paper cup with hot water", "soft plastic bowl with hot water" or "soft paper bowl with hot water", etc. The

scene setup is different from the Multisensory Universe as we place more distracting objects to retrieve from (while in Multisensory Universe most scenes have two similar objects), and we include different sensor attribute combinations from Multisensory Universe objects. For example, in the training set, we saw a ceramic cup and a paper bowl, and in the evaluation, we query about a paper cup.

**Baselines** We include a set of cross-modality retrieval models as our baselines, which return the similarity between aligned sensor embeddings. They can be categorized into 1) single-sensor language models, such as CLIP and CLAP. 2) 2D multisensory models, for which the embeddings of other modalities have been mapped to the same as 2D images like ImageBind [28]. 3) 3D multisensory models, in which the embeddings of object point clouds are binded to other modalities, like PointBind [30]. We first explore the environment and use concept graphs to represent the scene as a set of object features like MultiPLY, where the object features are visual embeddings from these retrieval models. The select action could be achieved by calculating the similarity between the object embedding and the language embedding, and the object with the highest score will be retrieved. As these models cannot interact with the environment to get the tactile, impact sound, and temperature data, we refine three setups for the baselines: 1) No interaction, and retrieve the object with the highest retrieval score. (For CLAP we assume that we have impact sounds of all objects) 2) Interact with the environment using oracle interactive actions. That is, we first retrieve the objects of interest via visual-language similarity, then we manually control the agent to interact with the objects to get impact sound, tactile and temperature information. The embeddings of all sensors are averaged and calculate the similarities with the language query, and the object with the highest score is retrieved. Since the action tokens are pre-defined and not generated, this oracle setting makes it easier to compete with MultiPLY. 3) Finetuned with a modified version of our Multisensory Universe tailored for multi-modal alignment and retrieval. Specifically, we first align the sensor data of the objects in Multisensory Universe to visual modality (like in ImageBind and PointBind), then we further align them with the modified language data in Multisensory Universe.

For LLM-based methods, we include Pointbind-LLM, which uses the pointbind representations and performs instruction tuning with LLaMA [54]. We also experiment with MultiPLY-2D, a 2D variant of our model, where we replace 3D features with 2D single-view features.

**Analysis** Table 1 shows the object retrieval results. We could come to several conclusions. First, models that take multiple sensory inputs outperform models that handle single modality inputs by a large margin. CLIP, CLAP, as well as models that use the initial visual embeddings have a very low score in object retrieval, emphasizing the importance

| Model | Retrieval Accuracy |
|---|---|
| ConceptGraph+CLAP | 14.5 |
| ConceptGraph+CLIP | 18.7 |
| ConceptGraph+ImageBind | 20.3 |
| ConceptGraph+ImageBind-I | 24.7 |
| ConceptGraph+ImageBind-I (Finetuned) | 36.7 |
| MultiPLY-2D | 44.6 |
| ConceptGraph+PointBind | 19.5 |
| ConceptGraph+PointBind-I | 22.7 |
| ConceptGraph+PointBind-I (Finetuned) | 40.4 |
| PointBind-LLM (Finetuned) | 48.9 |
| MultiPLY | **56.7** |

Table 1. **Experimental Results of Object Retrieval.** -I denotes the models utilize oracle action tokens to interact with the environment. (Finetuned) means finetuned on Multisensory Universe.

of integrating multisensory data for reasoning. Second, 3D-based models surpass 2D models, mainly because single-view images sometimes fail to provide enough information to reason about the objects due to view inconsistency and occlusion. Third, LLMs outperform similarity-based retrieval models. The reason could be that retrieval models fuse the multisensory embeddings into a whole, and do not disentangle the representation, or interact with the different sensors step by step. In general, our MultiPLY outperforms the baseline models a lot. That's probably because one weakness of the binding-based methods is that they bind everything to the visual modality, while one visual attribute could be mapped to several attributes from another modality (*e.g.,* from the appearance of a cup, we could not tell whether it's made of ceramic or plastic, unable to align to different impact sounds for alignment). Our MultiPLY resolves ambiguity by interacting with and reasoning about the different sensor data individually.

## 5.2. Tool Use

**Task Description** In an embodied environment, multisensory data are crucial for finding an appropriate tool to solve a problem. One example is that when we are injured, we need to retrieve warm compresses or ice packs depending on the injured parts and how long we've been injured. We could also find substitute tools if the common ones are not present. For example, we could use a steel spoon to replace the can opener, but we can't use a plastic spoon. Similar to the object retrieval task, we place some objects from different categories, and also objects from the same categories but with different materials/haptic/thermal information in the environment. We use one sentence to describe the current situation and the goal to be done, and ask the agent to retrieve the correct tool for dealing with the situation.

**Baselines** We use the same baselines as the object retrieval experiment for tool retrieval. For LLM-based methods, we

also need to give reasons when we select the tools.

**Analysis** Table 2 shows the results of tool use. We could see that the binding-based methods have a very poor performance in tool use. It might be because that they treat the object sensory data as a whole, unable to disentangle the individual sensory information such as material from the representation, let alone reasoning about how this property could be utilized as a tool, and how to analyze and deduce the functionality of an object when the multisensory information is integrated.

| Model | Accuracy |
|---|---|
| ConceptGraph+CLIP | 10.1 |
| ConceptGraph+ImageBind | 7.4 |
| ConceptGraph+ImageBind-I | 8.2 |
| ConceptGraph+ImageBind-I (Finetuned) | 16.4 |
| MultiPLY-2D | 36.3 |
| ConceptGraph+PointBind | 11.5 |
| ConceptGraph+PointBind-I | 13.2 |
| ConceptGraph+PointBind-I (Finetuned) | 18.7 |
| PointBind-LLM (Finetuned) | 32.1 |
| MultiPLY | **41.6** |

Table 2. **Experimental Results of Tool Use.**

## 5.3. Multisensory Captioning

**Task Description** Different from traditional single-modality captioning tasks, multisensory captioning requires the model to describe the object in all senses. By giving semantic information about an object or ambient sound emitted by the object, the agent must first navigate to the object to interact with it and describe it.

**Baselines** For baseline models, we include LLaVA, which takes a holistic scene image as input and generates a caption about the queried object. 3D-LLM takes the scene point cloud as inputs, and uses dense captioning to describe the object. Both methods only use visual information. PointBind-LLM first retrieves the objects by modality alignment, and then interacts with the objects and integrates multisensory information to describe the queried object.

| | BLEU1 | BLEU4 | METEOR |
|---|---|---|---|
| LLaVA | 9.5 | 0.6 | 7.1 |
| LLaVA (Finetuned) | 28.6 | 10.1 | 10.4 |
| 3D-LLM | 14.4 | 1.5 | 9.5 |
| 3D-LLM (Finetuned) | 31.2 | 12.1 | 12.4 |
| PointBind-LLM | 16.5 | 2.3 | 7.7 |
| PointBind-LLM (Finetuned) | 36.7 | 14.5 | 15.1 |
| MultiPLY | **48.9** | **20.1** | **24.2** |

Table 3. **Experimental Results of Multisensory Captioning.**

**Analysis** Table 3 shows the result. From the table, we could see that 3D-based LLMs overall outshine 2D VLMs.
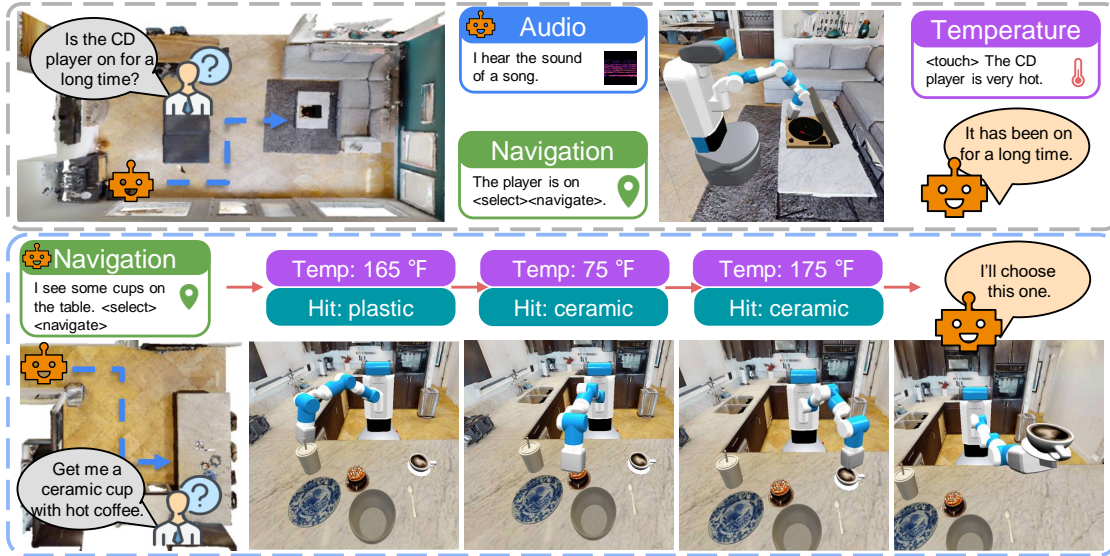
Figure 4. **Qualitative Examples of our MultiPLY**. MultiPLY could interact with the objects in the embodied environments and gather multisensory information.

LLaVA and 3D-LLM take the holistic representation as inputs, and thus fail to compete with models that could interact with the models to switch between representations. MultiPLY outshines Pointbind-LLM, probably because PointBind binds the representations of different modalities, making it difficult to disentangle the senses.

## 5.4. Task Decomposition

**Task Definition** Task decomposition focuses on decomposing a high-level task into smaller actions. In our setting, we focus on retrieving different things to prepare for a task. For example, to prepare for dinner, we need to first detect available foods in the kitchen, and gauge its temperature. If it's cold, we need to heat it in the microwave so we also need to retrieve a ceramic or glass container which is microwave-safe. We also need to prepare the utensils of the appropriate materials. In our setting, we place several possible choice combinations in the environment, we also place object combinations unseen from the Multisensory Universe. As long as the agent retrieves one of the correct combinations, the task is marked as success.

**Baselines** We include LLaVA, a minimal 2D image version of our model. We output an image of the scene and ask the model to decompose the tasks into actions. We also utilize 3D-LLM since it's capable of performing task decomposition. In the original paper, we take the whole point cloud as input and generate low-level actions. Note that there is a domain gap between the task decomposition data 3D-LLM was trained on and our setting, which yields almost zero success rates of 3D-LLM without finetuning. Therefore, we finetune all models as baselines. For each baseline we have two variants: 1) wo Interaction: generate all actions all at once, and execute the actions sequentially in the environment; 2) w Interaction: generate an action one at a time, take the action feedback and generate the next action.

| | success rate |
|---|---|
| LLaVA wo Interaction | 4.0 |
| LLaVA w Interaction | 14.5 |
| 3D-LLM wo Interaction | 8.7 |
| 3D-LLM w Interaction | 22.4 |
| MultiPLY | **30.2** |

Table 4. **Experimental Results of Multisensory Captioning.**

**Analysis** Table 4 shows the task decomposition results. From the table, we observe that models without interaction have very poor results, probably because vision-language models have hallucination to a great extent. For example, the models could generate "retrieve a bread" when there's no bread in the scene. MultiPLY outperforms the baseline models by a large margin. One reason could be that MultiPLY leverages multisensory information while the other two leverage visual information. The other reason might be that baseline models take the whole scene as inputs, thus could not attend to the nuanced object in the scene.

## 5.5. Qualitative Examples

Qualitative Examples are shown in Figure 4, demonstrating the power of MultiPLY to interact with objects in the embodied environments and gather multisensory information. More examples can be found in the **supplementary materials**.

## 6. Conclusion

In this paper, we propose MultiPLY, a multisensory LLM that could incorporate multisensory interactive data into large language models. We introduce Multisensory Universe, a dataset comprising 500k multisensory data collected by an agent actively exploring and interacting with an environment. One limitation of our model is that currently MultiPLY does not involve detailed navigation and control policy, but utilizes pre-defined policies for carrying out the actions. We think that such aspects are orthogonal to our study, and could be explored and seamlessly integrated into our framework in the future.

## References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022. 2, 3

[2] Anonymous. DIFFTACTILE: A physics-based differentiable tactile simulator for contact-rich robotic manipulation. In *Submitted to The Twelfth International Conference on Learning Representations*, 2023. under review. 4, 13

[3] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. See, hear, and read: Deep aligned representations. *arXiv preprint arXiv:1706.00932*, 2017. 3

[4] Roberto Calandra, Andrew Owens, Manu Upadhyaya, Wenzhen Yuan, Justin Lin, Edward H Adelson, and Sergey Levine. The feeling of success: Does touch sensing help predict grasp outcomes? *arXiv preprint arXiv:1710.05512*, 2017. 3

[5] Roberto Calandra, Andrew Owens, Dinesh Jayaraman, Justin Lin, Wenzhen Yuan, Jitendra Malik, Edward H Adelson, and Sergey Levine. More than a feeling: Learning to grasp and regrasp using vision and touch. *IEEE Robotics and Automation Letters*, 3(4):3300–3307, 2018. 3

[6] Changan Chen, Unnat Jain, Carl Schissler, Sebastia Vicenc Amengual Gari, Ziad Al-Halah, Vamsi Krishna Ithapu, Philip Robinson, and Kristen Grauman. Soundspaces: Audio-visual navigation in 3d environments. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 17–36. Springer, 2020. 3

[7] Changan Chen, Ruohan Gao, Paul Calamia, and Kristen Grauman. Visual acoustic matching. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[8] Changan Chen, Carl Schissler, Sanchit Garg, Philip Kobernik, Alexander Clegg, Paul Calamia, Dhruv Batra, Philip Robinson, and Kristen Grauman. Soundspaces 2.0: A simulation platform for visual-acoustic learning. *Advances in Neural Information Processing Systems*, 35:8896–8911, 2022. 3

[9] Peihao Chen, Yang Zhang, Mingkui Tan, Hongdong Xiao, Deng Huang, and Chuang Gan. Generating visually aligned sound from videos. *IEEE Transactions on Image Processing*, 29:8292–8302, 2020. 3

[10] Samuel Clarke, Ruohan Gao, Mason Wang, Mark Rau, Julia Xu, Mark Rau, Jui-Hsien Wang, Doug James, and Jiajun Wu. Realimpact: A dataset of impact sound fields for real objects. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[11] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects, 2022. 2, 3

[12] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model, 2023. 2

[13] Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. Clap: Learning audio concepts from natural language supervision, 2022. 4

[14] Chuang Gan, Hang Zhao, Peiaho Chen, David Cox, and Antonio Torralba. Self-supervised moving vehicle tracking with stereo sound. In *International Conference on Computer Vision (ICCV)*, 2019. 3

[15] Chuang Gan, Deng Huang, Peihao Chen, Joshua B Tenenbaum, and Antonio Torralba. Foley music: Learning to generate music from videos. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 758–775. Springer, 2020. 3

[16] Chuang Gan, Deng Huang, Hang Zhao, Joshua B Tenenbaum, and Antonio Torralba. Music gesture for visual sound separation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10478–10487, 2020. 3

[17] Chuang Gan, Yi Gu, Siyuan Zhou, Jeremy Schwartz, Seth Alter, James Traer, Dan Gutfreund, Joshua B Tenenbaum, Josh H McDermott, and Antonio Torralba. Finding fallen objects via asynchronous audio-visual integration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10523–10533, 2022. 3

[18] Ruohan Gao and Kristen Grauman. 2.5d visual sound. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[19] Ruohan Gao, Rogerio Feris, and Kristen Grauman. Learning to separate object sounds by watching unlabeled video. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 35–53, 2018. 3

[20] Ruohan Gao, Yen-Yu Chang, Shivani Mall, Li Fei-Fei, and Jiajun Wu. Objectfolder: A dataset of objects with implicit visual, auditory, and tactile representations. *ArXiv*, abs/2109.07991, 2021. 2, 3, 13

[21] Ruohan Gao, Yen-Yu Chang, Shivani Mall, Li Fei-Fei, and Jiajun Wu. Objectfolder: A dataset of objects with implicit visual, auditory, and tactile representations. *arXiv preprint arXiv:2109.07991*, 2021. 3

[22] Ruohan Gao, Zilin Si, Yen-Yu Chang, Samuel Clarke, Jeannette Bohg, Li Fei-Fei, Wenzhen Yuan, and Jiajun Wu. Objectfolder 2.0: A multisensory object dataset for sim2real transfer. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10588–10598, 2022. 3

[23] Ruohan Gao*, Zilin Si*, Yen-Yu Chang*, Samuel Clarke, Jeannette Bohg, Li Fei-Fei, Wenzhen Yuan, and Jiajun Wu. Objectfolder 2.0: A multisensory object dataset for sim2real transfer. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[24] Ruohan Gao, Yiming Dou, Hao Li, Tanmay Agarwal, Jeannette Bohg, Yunzhu Li, Li Fei-Fei, and Jiajun Wu. The objectfolder benchmark: Multisensory learning with neural and real objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17276–17286, 2023. 3

[25] Rishabh Garg, Ruohan Gao, and Kristen Grauman. Geometry-aware multi-task learning for binaural audio generation from video. In *British Machine Vision Conference (BMVC)*, 2021. 3

[26] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780, 2017. 4, 5

[27] Tiantian Geng, Teng Wang, Jinming Duan, Runmin Cong, and Feng Zheng. Dense-localizing audio-visual events in untrimmed videos: A large-scale benchmark and baseline. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22942–22951, 2023. 3

[28] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all, 2023. 2, 6

[29] Qiao Gu, Alihusein Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, Chuang Gan, Celso Miguel de Melo, Joshua B. Tenenbaum, Antonio Torralba, Florian Shkurti, and Liam Paull. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning, 2023. 4, 5, 16

[30] Ziyu Guo, Renrui Zhang, Xiangyang Zhu, Yiwen Tang, Xianzheng Ma, Jiaming Han, Kexin Chen, Peng Gao, Xianzhi Li, Hongsheng Li, and Pheng-Ann Heng. Point-bind & point-llm: Aligning point cloud with multi-modality for 3d understanding, generation, and instruction following, 2023. 2, 3, 6

[31] Jiaming Han, Renrui Zhang, Wenqi Shao, Peng Gao, Peng Xu, Han Xiao, Kaipeng Zhang, Chris Liu, Song Wen, Ziyu Guo, Xudong Lu, Shuai Ren, Yafei Wen, Xiaoxin Chen, Xiangyu Yue, Hongsheng Li, and Yu Qiao. Imagebind-llm: Multi-modality instruction tuning, 2023. 2, 3

[32] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models, 2023. 2, 3, 4, 5

[33] Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (TOG)*, 37:1 – 14, 2018. 4

[34] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In *North American Chapter of the Association for Computational Linguistics*, 2019. 6

[35] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross B. Girshick. Segment anything. *ArXiv*, abs/2304.02643, 2023. 16

[36] Hao Li, Yizhi Zhang, Junzhe Zhu, Shaoxiong Wang, Michelle A Lee, Huazhe Xu, Edward Adelson, Li Fei-Fei, Ruohan Gao, and Jiajun Wu. See, hear, and feel: Smart sensory fusion for robotic manipulation. *arXiv preprint arXiv:2212.03858*, 2022. 3

[37] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023. 2, 3

[38] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. 5

[39] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023. 2, 3

[40] Seungwhan Moon, Andrea Madotto, Zhaojiang Lin, Tushar Nagarajan, Matt Smith, Shashank Jain, Chun-Fu Yeh, Prakash Murugesan, Peyman Heidari, Yue Liu, Kavya Srinet, Babak Damavandi, and Anuj Kumar. Anymal: An efficient and scalable any-modality augmented language model, 2023. 2, 3

[41] Yashraj Narang, Balakumar Sundaralingam, Miles Macklin, Arsalan Mousavian, and Dieter Fox. Sim-to-real for robotic tactile sensing via physics-based simulation and learned latent projections. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6444–6451. IEEE, 2021. 3

[42] OpenAI. GPT-4 technical report. *ArXiv*, abs/2303.08774, 2023. 3

[43] Andrew Owens, Phillip Isola, Josh McDermott, Antonio Torralba, Edward H Adelson, and William T Freeman. Visually indicated sounds. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2405–2413, 2016. 3

[44] Qiutang Qi, Haonan Cheng, Yang Wang, Long Ye, and Shaobin Li. Rd-fgfs: A rule-data hybrid framework for fine-grained footstep sound synthesis from visual guidance. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 8525–8533, 2023. 3

[45] 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, 2021. 4

[46] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 2, 3

[47] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 3, 6

[48] Edward Smith, Roberto Calandra, Adriana Romero, Georgia Gkioxari, David Meger, Jitendra Malik, and Michal Drozdzal. 3d shape reconstruction from vision and touch. *Advances in Neural Information Processing Systems*, 33: 14193–14206, 2020. 3

[49] Edward Smith, David Meger, Luis Pineda, Roberto Calandra, Jitendra Malik, Adriana Romero Soriano, and Michal Drozdzal. Active 3d shape reconstruction from vision and touch. *Advances in Neural Information Processing Systems*, 34:16064–16078, 2021. 3

[50] Kun Su, Kaizhi Qian, Eli Shlizerman, Antonio Torralba, and Chuang Gan. Physics-driven diffusion models for impact sound synthesis from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9749–9759, 2023. 3

[51] Chunyi Sun, Junlin Han, Weijian Deng, Xinlong Wang, Zishan Qin, and Stephen Gould. 3d-gpt: Procedural 3d modeling with large language models. *arXiv preprint arXiv:2310.12945*, 2023. 3

[52] Sudharshan Suresh, Zilin Si, Joshua G Mangelson, Wenzhen Yuan, and Michael Kaess. Shapemap 3-d: Efficient shape mapping through dense touch and vision. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7073–7080. IEEE, 2022. 3

[53] 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. 3

[54] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. 6

[55] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 3

[56] Mark T. Wallace. The development of multisensory processes. *Cognitive Processing*, 5:69–83, 2004. 2

[57] Hao Wang, Zheng-Jun Zha, Liang Li, Xuejin Chen, and Jiebo Luo. Context-aware proposal–boundary network with structural consistency for audiovisual event localization. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. 3

[58] Shaoxiong Wang, Mike Lambeta, Po-Wei Chou, and Roberto Calandra. Tacto: A fast, flexible, and open-source simulator for high-resolution vision-based tactile sensors. *IEEE Robotics and Automation Letters*, 7(2):3930–3937, 2022. 3

[59] Yufei Wang, Zhou Xian, Feng Chen, Tsun-Hsuan Wang, Yian Wang, Zackory Erickson, David Held, and Chuang Gan. Robogen: Towards unleashing infinite data for automated robot learning via generative simulation, 2023. 4

[60] Yan Xia and Zhou Zhao. Cross-modal background suppression for audio-visual event localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19989–19998, 2022. 3

[61] Haoming Xu, Runhao Zeng, Qingyao Wu, Mingkui Tan, and Chuang Gan. Cross-modal relation-aware networks for audio-visual event localization. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 3893–3901, 2020. 3

[62] Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. Pointllm: Empowering large language models to understand point clouds. *arXiv preprint arXiv:2308.16911*, 2023. 3

[63] Karmesh Yadav, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Theo Gervet, John Turner, Aaron Gokaslan, Noah Maestre, Angel Xuan Chang, Dhruv Batra, Manolis Savva, et al. Habitat-matterport 3d semantics dataset. *arXiv preprint arXiv:2210.05633*, 2022. 3

[64] Jianing Yang, Xuweiyi Chen, Shengyi Qian, Nikhil Madaan, Madhavan Iyengar, David F Fouhey, and Joyce Chai. Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent. *arXiv preprint arXiv:2309.12311*, 2023. 3

[65] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022. 3

[66] Hang Zhao, Chuang Gan, Andrew Rouditchenko, Carl Vondrick, Josh McDermott, and Antonio Torralba. The sound of pixels. In *The European Conference on Computer Vision (ECCV)*, 2018. 3

[67] Hang Zhao, Chuang Gan, Wei-Chiu Ma, and Antonio Torralba. The sound of motions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1735–1744, 2019. 3

# Contents

## A. Dataset

### A.1. More details on Scene Construction

In figure 5, we show how we add new objects to the HM3D scenes. Specifically, ChatGPT is asked to generate: 1) object bounding boxes; 2) object material and material properties; 3) temperatures.

```
messages=[{"role": "system" , "content": "You're an AI assistant that can analyze a 3D scene."
          "A room is given with its bounding box in format '<room>: [[x min, y min, z min],[x max, y max, z max]]'. " \
       "All object instances in this 3D scene are given with their bounding boxes in format 'obj_name: [x min, y min, z min],[x max, y
max, z max]]'. \n" \
          "You need to select 1-10 objects possible to appear in this 3D scene from the candidate_objects. " \
          "You need to specify whether the object is rigid, elastic, plastic, cloth or liquid. If the object is elastic, you could specify whether
the object is hard or soft"
          "You need to specify the material of the object: plastic, ceramic, steel, polycarbonate and so on"
          "You could select some ambiguous objects (like two objects of the same category, one of them is wood and one of them is
ceramic), so interesting tasks could be proposed about the objects. " \
          "You could specify whether it's hot or cold. you could add the same object, one is hot and one is cold.\n" \
          "You also need to output a proper bounding box to place the selected object with correct size and location. You need to ensure that
there's no collision between the existing objects and added objects. They also don't collide with each other. You need to ensure that the
bounding box makes sense so the object does not float in the air. Give Reason why you select the objects. \n" \"
          "Remember, Do not copy coordinates from input data." \
          "The coordinate of the object should be inside the room!" \
          "You DON'T choose objects that are already in the room. (for example, if there's a chair, you don't want a chair again!)"

for sample in fewshot_samples:
    messages.append({"role": "user", "content": '\n'.join(sample['scene'])})
    messages.append({"role": "assistant", "content": sample['response']})

messages.append({"role": "user", "content": '\n'.join(new_scene)})
```

Figure 5. Prompts for adding objects to the scene

### A.2. More details on Sensor Data Acquisition

In this section, we elaborate on how we get the sensor data of the objects in details.

#### A.2.1 Tactile

DiffTactile [2] requires us to provide a set of parameters for tactile simulation of different objects. In addition to telling to the model whether we are inputting a rigid, elastic, or elasto-plastic object, we also need to specify the parameters such as Young's modulus, Poisson's ratio, Yield Strength and so on.

As in the main paper, when ChatGPT adds objects to the scene, it also specifies what kinds of objects (*e.g*, rigid, elastic, plastic) and the softness / deformability (in the description of language) of each object. In order to get the parameters required by DiffTactile, we prompt ChatGPT with the type and the softness / deformability description, as well as detailed definition of each parameter, and the possible values of the parameters of several few-shot examples. ChatGPT is asked to return the detailed parameter combinations of the given objects. For example, a soft bread corresponds to a smaller young's modulus, while a harder one corresponds to a larger young's modulus. We add the prompt in getting the parameters in Figure 6.

We input the object into DiffTactile, normalize the shape of the gripper according to the object. We record the 2D initial position and final position of the markers in the gripper. And we turn the tactile readings into a 2D image, by drawing an arrowed line from the initial position to the final position. We show some examples of tactile images in Figure 7. We sample 16 touching positions of each object. In training and evaluation, we randomly return one image of the object.

#### A.2.2 Impact Sound

ObjectFolder [20] stores the multi-modal information all in implicit fields. That is, by inputting a striking location to the sound implicit field of an object, we could get the impact sound of striking the object at the specific location. For each object,

```
messages=[{"role": "system" , "content": "You are a chemist and material analyzer that could analyze the materials of
any objects. Given the object \"%s\", please define the following things:\n \
`                  Young's Modulus: Provide a value for this specific object\n \
            The definition of Young's Modulus: quantifies the relationship between tensile or compressive stress σ \sigma (force per unit area)
and axial strain ε \varepsilon (proportional deformation) in the linear elastic region of a material. It's equal to exerted force / deformation
length under the force. The lowest values of Young's modulus are for materials like natural rubber, at 0.01–0.1 GPa, whereas the highest
values are typically for carbon nanotube materials (up to 1,000 GPa) \
                 Poisson Ratio: Please choose a value between 0.0 and 0.5 for this specific object\n \
            The definition of Poissons ratio: ν \nu (nu) is a measure of the Poisson effect, the deformation (expansion or contraction) of a
material in directions perpendicular to the specific direction of loading. The value of Poisson's ratio is the negative of the ratio of transverse
strain to axial strain. For small values of these changes, ν \nu is the amount of transversal elongation divided by the amount of axial
compression. Most materials have Poisson's ratio values ranging between 0.0 and 0.5. For soft materials, such as rubber, where the bulk
modulus is much higher than the shear modulus, Poisson's ratio is near 0.5. For open-cell polymer foams, Poisson's ratio is near zero, since
the cells tend to collapse in compression. Many typical solids have Poisson's ratios in the range of 0.2–0.3.\n \
            Yield Strength: Provide a value for this specific object\n \
            The yield strength or yield stress is a material property and is the stress corresponding to the yield point at which the material
begins to deform plastically. The yield strength is often used to determine the maximum allowable load in a mechanical component, since it
represents the upper limit to forces that can be applied without producing permanent deformation. The yield strength of steel ranges from as
low as 220 MPa (hot-rolled A36 steel) to as high as 1570 MPa (4140 alloys, oil-quenched and tempered)\n""}]

for sample in fewshot_samples:
    messages.append({"role": "user", "content": '\n'.join(sample['object'])})
    messages.append({"role": "assistant", "content": sample['response']})

messages.append({"role": "user", "content": '\n'.join(new_object)})
```

Figure 6. Prompts for getting the material parameters for the objects
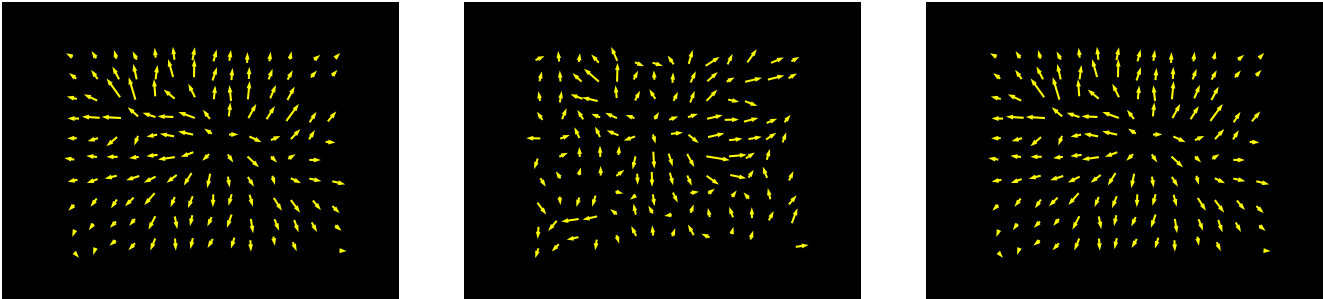


Figure 7. Examples of tactile images.

we randomly sample 10 locations in the mesh points to get the impact sound. In training and evaluation, we randomly return
one impact sound of the object.

### A.2.3 Ambient Sound

AudioSet is paired with objects to represent ambient sound. The AudioSet ontology is organized in a hierarchy structure.
From the root node to the leaf node, the description granularity becomes finer (e.g., Music - Musical instrument - Keyboard
- Piano - Electric piano). Each ontology entry is attached with a description (e.g., "Glass: sounds associated with the non-
crystalline amorphous solid that is often transparent and has widespread practical, technological, and decorative uses"). Each
audio is labeled with multiple ontology entries tracing from the child node to the root node (e.g., the sound of the piano will
be labeled with "Piano", "Keyboard", "Musical Instrument", and "Music", but without "Electric piano" since this piano is
not electric). We prompt ChatGPT to match each ontology entry with object categories (Figure 8).

### A.2.4 Temperature

We add the prompt in getting the temperature in Figure 9.

14

```
System:
You are an AI audio assistant that can analyze descriptions and tags of sounds. The input has three fields. 'tags' are some labels about the
sound. 'description' is the text description of the sound. 'objects' are a list of candidate objects. You need to infer what kind of objects can
make the sound based on 'tags' and 'descriptions'. You need to select ALL objects that are possible to make this sound from the 'objects' list.
Remember, the object MUST be found in a normal INDOOR environment. Do not include objects that do not exist in the 'objects' list.
Return [] if no object satisfies the condition.

Example Questions:
tags=['Frying (food)', 'Domestic sounds, home sounds', 'Sounds of things'],
description='The sound of cooking food in oil or another fat.',
objects=[poncho, pool_table, pop_(soda), popsicle, postbox_(public), pan_(for_cooking), postcard, poster, pot, potato, potholder]

Example Answers:
[pot, pan_(for_cooking)]
```

Figure 8. Prompts to match AudioSet with Objects

```
messages=[{"role": "system" , "content": "You are a temperature analyzer that can analyze a 3D scene.
          You need to assign a temperature (celsius) for the input object. The default room temperature is 26 degree celsius. \
          Pay attention to the hot and cold label of the objects. For example, cup_hot can be as hot as 85 celsius, and cup_cold can be as
cold as 5 celsius."}]

for sample in fewshot_samples:
    messages.append({"role": "user", "content": '\n'.join(sample['object'])})
    messages.append({"role": "assistant", "content": sample['response']})

messages.append({"role": "user", "content": '\n'.join(new_object)})
```

Figure 9. Prompts on generating temperature for each object

## A.3. More details on Task Construction

In Figure 10, we illustrate the prompts for generating the language task data for Multisensory-Universe. Specifically, the actions could return the expected observation in the form of language (*e.g.,* tactile map of and object when touching). We insert that into the state tokens for placeholder, and after the agent has executed the actions in the space and gets the observations, we append the observations back to the state tokens.

| Ablative Model | Acc |
|---|---|
| MultiPLY Vision | 21.0 |
| MultiPLY Audio | 13.2 |
| MultiPLY Tactile | 10.5 |
| MultiPLY Temperature | 11.2 |
| MultiPLY Vision, Audio | 31.8 |
| MultiPLY Vision, Tactile | 24.3 |
| MultiPLY Vision, Temperature | 25.7 |
| MultiPLY Audio, Tactile | 20.6 |
| MultiPLY Audio, Temperature | 23.4 |
| MultiPLY Tactile, Temperature | 18.9 |
| MultiPLY Vision, Audio, Tactile | 45.3 |
| MultiPLY Vision, Tactile, Temperature | 41.4 |
| MultiPLY Vision, Audio, Temperature | 45.3 |
| MultiPLY Audio, Tactile, Temperature | 37.7 |
| MultiPLY | 56.7 |

Table 5. Ablative Study of MultiPLY

15

```
messages=[{"role": "system" , "content": "You are an AI assistant / task generator in the room. All object instances in
this 3D scene are given, along with their bounding boxes and ids." \
        "The bounding boxes are represented by a 3D coordinate (x, y, z) with units of meters. " \
        "If the object emits a sound, it will have a 'emit' label." \
        "If the object could be hit, it will have a 'hit' label." \
        "You could use the actions to interact with the environment. They are:"
            "<SELECT>: which returns the id of the object"
            "<NAVIGATE>: which navigates to the object selected"
            "<OBSERVE>: which returns the visual details of the object"
            "<TOUCH>: which returns tactile and temperature information of the object"
            "<HIT>: which returns the impact sound of the object"
            "<PICK UP>: pick up the object"
            "<PUT DOWN>: put down the object"
            "<LOOK AROUND>: retrieves objects ids and categories near the object"

Using the provided object instance information and selected objects, you need to generate a task that could be performed in the scene.
Exempler tasks include captioning, question answering, dialogue, manipulation, task decomposition, rearrangement. For example:
        "Captioning: you need to choose one object, describing its information of all modalities, and also its relationships to the other
objects. "
        "Question Answering: you need to generate several question-answering pairs about the 3D scene. The questions must be
answered by exploring the room using the above actions."
        "Manipulation: You need to generate some manipulation tasks which you use the actions to manipulate the objects"
        "Task Decomposition: You need to design a task that could be performed in this room and decompose it into 3-10 sub-tasks.
The task must be completed using the actions."
        "Rearrangement: If the objects are in a weird position, move them to a suitable location using the actions."
You also need to decompose the description process by several actions to interact with the environment using the tokens above. You need
to also specify what's the observation / feedback you could get by executing the action. For example <SELECT> -> returns apple(65),
where 65 is the object id, or touch -> returns tactile map and temperature of apple(65). You need to output your reasoning processes like "I
need to touch it " or conclusions like "it's hot"
for sample in fewshot_samples:
    messages.append({"role": "user", "content": '\n'.join(sample['scene'])})
    messages.append({"role": "assistant", "content": sample['response']})

messages.append({"role": "user", "content": '\n'.join(new_scene)})
```

Figure 10. Prompts for task construction

# B. Experiments

## B.1. Experimental Details

We tune the model based on the llava-v1.5-7b checkpoint of the LLaVA model. We use Adam optimizer with learning rate of
1e-6. We train the model on 4*132 V100s. We use a batch size of 2112. The training of multi-modal adapters takes 2 hours,
while the whole finetuning takes less day 1 day to complete.

We use the mm projector of the original LLaVA for adapting scene representations and object point clouds to the LLM.
The sound, tactile and temperature adapters are all one linear layer with input size 1024 and output size 1024.

We use the default CLIP vision encoder of LLaVA to encode all objects, point clouds, tactile and temperature images.
Specifically, for objects, we use segment anything [35] to get the objects out of 2D objects, mask out other objects and
background, and crop the image to the size of the object, and use CLIP encoder to encode the object. We follow ConceptGraph
[29] to merge the objects from 2D to 3D. For scene construction, each object has one CLIP feature. For object details (point
cloud), we project the 2D pixels of the objects to 3D, and get the point clouds of the objects.

## B.2. Ablative Studies

In Table 5, we show additional experimental results where we explore MultiPLY with single, double or triple modalities.

## B.3. More Qualitative Examples



Figure 11. More qualitative examples of MultiPLY