MotIF: Motion Instruction Fine-tuning

Minyoung Hwang[1], Joey Hejna[2], Dorsa Sadigh[2], Yonatan Bisk[3]

***Abstract*—While success in many robotics tasks can be determined by only observing the final state and how it differs from the initial state – e.g., if an apple is picked up – many tasks require observing the full motion of the robot to correctly determine success. For example, brushing hair requires repeated strokes that correspond to the contours and type of hair. Prior works often use off-the-shelf vision-language models (VLMs) as success detectors; however, when success depends on the full trajectory, VLMs struggle to make correct judgments for two reasons. First, modern VLMs are trained only on single frames, and thus cannot capture changes over a full trajectory. Second, even if we provide state-of-the-art VLMs with an aggregate input of multiple frames, they still fail to correctly detect success due to a lack of robot data. Our key idea is to fine-tune VLMs using abstract representations that are able to capture trajectory-level information such as the path the robot takes by overlaying keypoint trajectories on the final image. We propose motion instruction fine-tuning (MotIF), a method that fine-tunes VLMs using the aforementioned abstract representations to semantically ground the robot's behavior in the environment. To benchmark and fine-tune VLMs for robotic motion understanding, we introduce the MotIF-1K dataset containing 653 human and 369 robot demonstrations across 13 task categories. MotIF assesses the success of robot motion given the image observation of the trajectory, task instruction, and motion description. Our model significantly outperforms state-of-the-art VLMs by at least twice in precision and 56.1% in recall, generalizing across unseen motions, tasks, and environments. Finally, we demonstrate practical applications of MotIF in refining and terminating robot planning, and ranking trajectories on how they align with task and motion descriptions.**

Project page: https://motif-1k.github.io/

I. Introduction

Measuring success in robotics has focused primarily on *what* robots should do, not *how* they should do it. Concretely, *what* is determined by the *final state* of an object, robot, or end-effector [1], [2]. However, not all trajectories that achieve the same final state are *equally successful*. When transporting a fragile object, a path through safer terrain could be considered *more successful* than a shorter yet riskier route (Fig. 1 a). Similarly, in the presence of humans a robot's actions when navigating, holding objects, or brushing human hair (Fig. 1 b-d) can cause surprise, discomfort, or pain, making such motions *less successful*.

Success detectors play an important role in robot learning since they evaluate whether or not a robot has completed a task. However, most overlook the importance of "*how*" the task is accomplished, focusing on the initial and final states of the trajectory [2], [3]. This simplification fails to account for tasks that fundamentally require evaluating the entire trajectory to assess success. As we incorporate robots into everyday scenarios, the manner in which they complete tasks will become increasingly important given the context of a scene and its semantic grounding (*e.g.,* moving safely to avoid collisions with chairs). Therefore, a more holistic approach to success detection is needed that considers the entire trajectory, the task, and how the agent should move to complete said task.

While modern vision-language models (VLMs) have recently been used as promising tools for success detection [2], [3], they are unable to capture complex notions of how a task is completed. This challenge stems from the fact that VLMs are generally designed to reason over single images, while success detection in robotics is inherently sequential. To bridge this gap, we explore the choice of abstract motion representations, such as visualizing trajectories or aggregating multiple frames in a single image, to better leverage the capabilities of VLMs. In this paper, we propose a trajectory based visual motion representation which overlays a robot's trajectory on the final frame, capturing both the path shape and its semantic connections to the environment. This approach leverages the world knowledge encoded in VLMs and refines it to assess robotic behaviors more effectively.

We propose **motion instruction fine-tuning**, a method that fine-tunes pre-trained VLMs to equip the capability to distinguish nuanced robotic motions with different shapes and semantic groundings. Given a single frame with a robot's trajectory overlaid and its corresponding task and motion specifications, we query VLMs for success detection. Specifically, our model outputs a binary value indicating whether the motion is *correct* (1) or *incorrect* (0).

We collect the **MotIF-1K** dataset to fine-tune VLMs, due to limited public robot data with diverse semantically grounded motions. We find that co-training mostly on human data with limited robot data enables the model to transfer its knowledge to robotic motion understanding effectively. Based on this insight, **MotIF-1K** contains a variety of motions with 653 human and 369 robot demonstrations across 13 task categories, offering extensive coverage of both the *what* and the nuanced *how* of motion, complete with detailed annotations. It identifies common types of motions featuring varying degrees of semantic grounding, such as the robot's relationship with objects or humans in the environment. The dataset also captures diverse path shapes, in terms of directionality, concavity, and oscillation. For instance, paths in Fig. 1 (a) differ in terms of directionality and spatial relationship with the environment, where it might be undesirable for a robot to pass over the grass. Paths in Fig. 1 (d) demonstrate how straight and curly hairs might require different brushing techniques regarding horizontal oscillations. Notably, MotIF-1K includes subtle motions that are often indistinguishable solely by their start and end states (Fig. 4).

MotIF, a motion discriminator developed by fine-tuning on MotIF-1K, shows further improved success detection on nuanced robot motions. We evaluate MotIF on the validation

[1]Massachusetts Institute of Technology, [2]Stanford University, and [3] Carnegie Mellon University. myhwang@mit.edu, jhejna@stanford.edu, dorsa@stanford.edu, ybisk@cs.cmu.edu

(a) Outdoor Navigation (b) Indoor Navigation (c) Object-Interactive Manipulation (d) User-Interactive Manipulation

Fig. 1: Different robotic motions for various tasks. For each task, we visualize two different motions (path 1 and 2) from real robot demonstrations, where the trajectories share the same initial and final states. Most existing success detectors ignore intermediate states, thereby cannot distinguish them.

and test splits of MotIF-1K and demonstrate generalization to unseen motions, tasks, and environments. We significantly outperform state-of-the-art VLMs (*e.g.* GPT-4o, GPT-4V and Gemini-1.5 Pro), with at least twice and 56.1% higher precision and recall, respectively. Additionally, we demonstrate three applications of using MotIF: (1) using it as a feedback to finish robot planning, (2) using it as a feedback to refine planning without additional human effort, and (3) using the model output to rank trajectories. In summary, our main contributions include:

- **Motion Representation**: We propose overlaying a robot's trajectory on the final frame to capture the path shape and semantic context, enhancing VLMs' ability to assess success on nuanced robotic behaviors.
- **MotIF-1K Dataset**: We introduce a dataset with 653 human and 369 robot demonstrations across 13 tasks, providing extensive coverage of various motion types for effective VLM fine-tuning.
- We develop **MotIF**, a motion discriminator fine-tuned on MotIF-1K, achieving significant improvements in success detection precision and recall over state-of-the-art VLMs.

II. Related Work

With the recent development of large language models (LLMs) and VLMs, foundation models have been used to understand the environment and critique agent behaviors. Additionally, the increasing use of visual observations in robotics has brought attention to motion-centric visual representations.

Success Detection Using Foundation Models. [4], [5], [6], [7], [8] use LLMs to generate reward functions that evaluate agent behaviors. However, as LLMs cannot understand how a robot's actions are visually grounded in a scene, such methods require everything including the state of the environment to be translated through language. VLMs have been used to understand and evaluate agent behaviors given visual and language information. [9] use a VLM critic for general vision-language tasks, and closed API-based models have been used as behavior critics in robotics, even if inaccurate [10], [3]. Prior work [11], [10], [12], [13] have also used VLMs as zero-shot reward models for training downstream policies. [2] trains a VLM success detector for evaluating what was achieved from the robot, but does not consider *"how"* the agent solves the task. [14], [15] use representation learning to train reward models as behavior critics. Other studies [16], [17], [18] train VLMs to be physically or spatially grounded, but focus on static environment understanding, not dynamic motions. [19] uses robot joint states to generate motion descriptions in language, but their limited vocabulary is restricted to short horizon, non-grounded motions (e.g., move arm up, rotate arm right). In contrast, we fine-tune VLMs to understand and evaluate grounded motions (e.g., make a detour to the left of the table) and consider more complicated, long-horizon, object or human interactive motions. Our work extends these foundations by specifically using VLMs as success detectors in the context of grounded motion in various tasks.

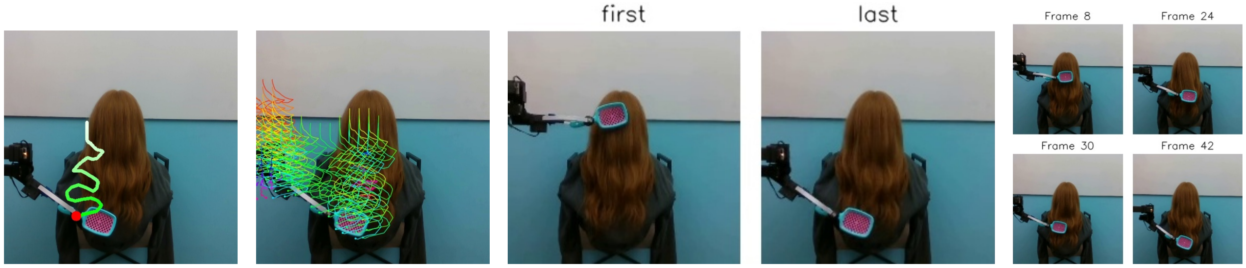
Motion-Centric Visual Representations. A growing interest in motion-centric visual representations in robotics has led to recent work in egocentric trajectory representation [20], [21] and point tracking using optical flow [22], [23] or learned predictors [24], [25], [26]. Prior works summarize trajectory or actionable choices in a single image frame by visualizing waypoints [20], keypoints in the environment [27], [28], or desirable paths [21], [29], [30]; however, the focus of these works is mostly on conditioning the policy on these representations to improve policy learning rather than evaluation of nuanced motions and behaviors. While Egocentric views are easy to visualize robot trajectories in the camera frame, they often fail to provide comprehensive environmental context and understanding of the robot's movement in the global coordinate. In this paper, we mainly employ visual representations of trajectories overlaid on single images from an exocentric view, which can offer a broader perspective, crucial for analyzing the robot's interactions and movements comprehensively.

III. Motion Instruction Fine-tuning (MotIF)

In this section we first broaden the definition of success by including motion as a core component. We then discuss why pre-trained models are insufficient for motion-based success detection in robotics. Finally, we introduce MotIF for fine-tuning VLMs to be motion-aware success detectors.

A. Problem Statement

Success detection has been an integral part of recent robotics literature. Typically, success detection is defined as a binary function of the final state conditioned on the task, $y = f(o_T|\text{task}) \in \{0, 1\}$ [2], where y is the binary success label and o_T is the image observation of the agent and the environment at the final time step T. This restrictive assumption prevents success detectors from criticizing *how* a task is completed. For many tasks like collision-aware navigation, the final state does not provide sufficient information to capture the



(a) Single Keypoint **(b)** Optical Flow **(c)** 2-frame Storyboard **(d)** 4-frame

Fig. 2: Visual Motion Representations We explore three visual motion representations: (a) single keypoint tracking, (b) optical flow, and (c-d) multi-frame storyboard. For single keypoint tracking, temporal changes are shown with color gradient from white to green, ending with a red circle. For optical flow, we visualize the flow of all keypoints with rainbow colors. We sample N keyframes for N-frame storyboard.

robot's interaction with objects or humans in the environment. Such tasks are often described by both their objective (i.e. bring me lemonade) and their execution (i.e. avoid going over the grass). In these scenarios, success cannot be determined by just the final state. Perhaps the simplest approach is modeling the entire trajectory, $y = f(o_1, \ldots, o_T | \text{objective, execution})$; however, this is computationally costly and often redundant. We instead propose using an abstract visualization of the trajectory $\tau = (o_1, \ldots, o_T)$. Notably, $I(\tau)$ outputs a 2D representation of the full trajectory overlaid on the last image o_T.

Suppose a task instruction T and motion description M corresponds to the robot's trajectory. Our goal is to discern how aligned different types of motions are given a task specification to be able to assess success based on *how* the task is done. Doing so requires assessing motion or path shape and, if the task requires object or human interaction, semantically grounding the robot's motion in the environment. Thus, we define the set of motions based on two criteria: *path shape* and *semantic grounding* (see Fig. 5). First, motions are distinguished based on properties of *path shapes* such as directions of translations, rotations, oscillations, repeated motions, and the convexity of curves. Second, *semantic grounding* in the environment involves understanding the context of the scene and the robot's interaction with the environment. For instance, moving over or making a detour towards an object implies the robot is aware of that instance. Similarly, we consider the relative distance and orientation of the agent with respect to the key objects in the scene (see App. Table I). In the following sections, we develop a VLM that acts as a success detector $y = f(o_1, \ldots, o_T | T, M)$.

B. Visual Motion Representations

Can foundation models be used for success detection? While VLMs have demonstrated a strong understanding of physical and causal commonsense reasoning [31], [32] and semantic grounding [17], [16], [33], they typically work with static images as inputs, and cannot reason about sequential inputs necessary for dynamic tasks in robotics. Understanding motion requires not only isolating the most meaningful aspects of the scene but also identifying which changes that occurred due to the robot's motion are semantically relevant to the task. Without the ability to understand the changes over time, VLMs may struggle to detect success over robotic motions.

While the ideal model would simply extract semantic content from videos, the strongest current VLMs instead rely on detecting differences between multiple frames, often in the form of storyboards (see Fig. 2 (c-d) and the example of GPT-4o in Section C). Though straightforward, this approach often struggles as storyboards lead to lower resolution images. Instead, we borrow insights from prior work [21], [29] that show VLMs can effectively leverage diagrams or abstract representations on top of image observations.

Representing a robot's motion in a single image. Due to the limitations of storyboards, we explore what representations effectively capture a robot's motion in a single frame. To construct diagrams of robotic motions, we overlay a robot's trajectory on the image observation as shown in Fig. 2 (a-b). One solution is to detect K keypoints $\{(x_0^1, y_0^1), \ldots, (x_0^K, y_0^K)\}$ in the initial frame I_0, and track the movement of each keypoint until the final frame I_T (see Fig. 2 (b)). Here, x_t^k and y_t^k denote the x and y coordinates of the k^{th} keypoint in a 2D image observation at timestep t, respectively. The detected visual traces, i.e., optical flow, represent the apparent motions of the robot and how the environment changes accordingly. While this solution helps a single image representation contain the information of multiple keypoints, the full optical flow with trajectories of all keypoints may obscure a large portion of the background and important objects in the scene. Additionally, visualizing many keypoints often results in indistinguishable or overlapping trajectories, which may create visual clutter and reduce the clarity of the motion.

Therefore, the proposed method, MotIF, visualizes the trajectory of the most representative keypoint (see Fig. 2 (a)), as a simplified yet more interpretable visual motion representation. We call the keypoint as point of interest, where a point of interest is typically chosen as a point on the end effector's surface by human annotators. Compared to prior work [21] that visualizes the center of mass of the end effector, our approach ensures that the selected keypoint is visually recognizable and not occluded. Details on labeling the point of interest and visualizing its trajectory are provided in Section IV.

C. Fine-Tuning VLMs

While we can directly pass a single image visualizing the robot's trajectory into a model zero-shot, non-fine-tuned models struggle to understand complex robotic motions. Section V empirically shows that existing state-of-the-art VLMs often

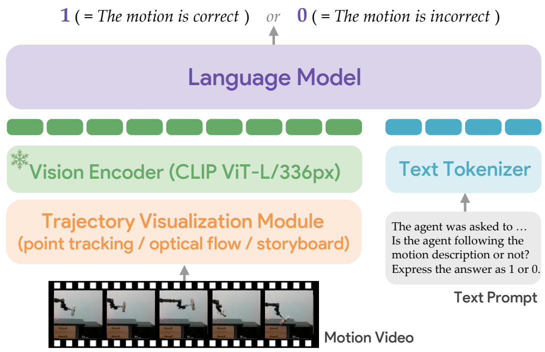


Fig. 3: Network Architecture. Given a visual motion representation of a robot's trajectory and its corresponding task and motion specifications, our model outputs a binary value indicating whether the motion is correct (1) or incorrect (0).

infer undesirable motions as correct (false positives) or successful motions as incorrect (false negatives). Prior works [16], [17] have shown promising results fine-tuning VLMs to improve their grounding capabilities and understanding of a scene. Similarly, we also fine-tune VLMs with the different representations proposed earlier.

Model Architecture. Fig. 3 shows the network architecture of our model. Given a trajectory representation $I(\tau)$ as an image and a text prompt consisting of a task instruction T and motion description M followed by the question *"Is the agent following the motion description or not? Express the answer as 1 or 0."*, our model outputs a binary success prediction y. The prediction y indicates whether the motion is *correct* (1) or *incorrect* (0) with respect to the task and motion specification (T, M). During fine-tuning, only the language model is updated, while the visual encoder is frozen.

Constructing Training Data. Effective fine-tuning requires a well-structured dataset that includes both positive and negative samples. Positive samples consist of trajectories paired with their corresponding task instructions and motion descriptions. This encourages the model to associate visual motion representations with their correct task specification. For the i^{th} trajectory τ_i in the training set we generate a single image I_i representing the trajectory based on the chosen visual motion representation. To construct positive samples for training, we pair the set of K images with their corresponding task instructions and motion descriptions, forming the set $\mathbf{S}^+_{\text{train}} = \{(I_1, T_1, M_1), \ldots, (I_K, T_K, M_K)\}$, where T_i and M_i are the task instruction and motion description for image I_i, respectively. To construct negative samples for image I_i, we choose the N_{neg} least similar motion descriptions. The similarity between motion descriptions is calculated using SentenceTransformer [34] embeddings. The set of negative samples is then constructed as $\mathbf{S}^-_{\text{train}} = \bigcup_{i=1}^{K} \{(I_i, T_i, M^-_{i,1}), \ldots, (I_i, T_i, M^-_{i,N_{\text{neg}}})\}$, where $M^-_{i,j}$ is the j^{th} least similar motion description to M_i. We set $N_{\text{neg}} = 10$ in our experiments. The full training dataset D is $\mathbf{S}^+_{\text{train}} \cup \mathbf{S}^-_{\text{train}}$.

Co-training with Human and Robot Data. Based on prior work [35], fine-tuning VLMs require a huge amount of training data. Ideally our dataset D would cover all motions that a robot would execute across a wide variety of tasks. Unfortunately, robot demonstrations are time-consuming and often difficult to collect. This makes scaling the size of the dataset hard when we only rely on robot data for fine-tuning VLMs. Moreover, robots may have physical constraints which prevent a model trained with a specific robot embodiment from generalizing to other robots with different dynamics. On the other hand, human demonstrations have a high degree of freedom and are often easier and more intuitive to collect, especially when looking for a diversity of motion. Thus, we opt to train on a mixture of human and robot demonstrations, D_h and D_r respectively. By doing so, we facilitate easy data collection while also ensuring the downstream VLM is more robust to embodiment and motion types.

IV. MotIF-1K Dataset

To benchmark and improve VLMs for motion understanding, we release the MotIF-1K dataset containing 653 human and 369 robot demonstrations across 13 tasks. As in Fig. 1 and Fig. 4, each task has demonstrations for 2 to 5 distinct motions which vary during the intermediate steps of a trajectory. For instance, a motion's path shape and semantic relationship with nearby objects may be different across demonstrations. This captures the diversity of *how* a task can be achieved, reflecting the nuanced and complex motions present in real-world scenarios. For instance, when shaking a boba drink, one person might use vigorous vertical movements, while another might use careful side-to-side movements to avoid spillage after inserting a straw. Moving a cup near a laptop via the shortest path is preferred if the cup is empty, but a detour is necessary if the cup contains water. Motion diversity is essential in grounded settings where motions need to adapt to varying environmental contexts. Fig. 4 shows examples of diverse motions. By collecting a diverse set of context-dependent motions with different intermediate trajectories, we ensure that our dataset challenges VLMs to consider the full trajectory for success detection.

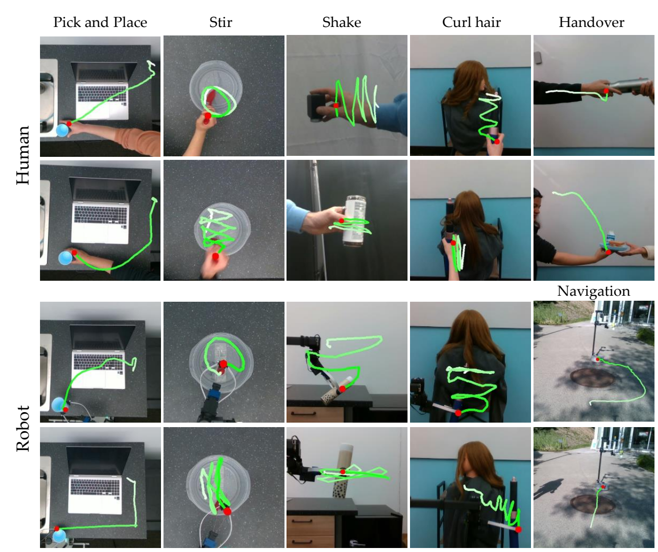


Fig. 4: Trajectory Visualizations. We visualize two different motions for solving the same task with the same embodiment.

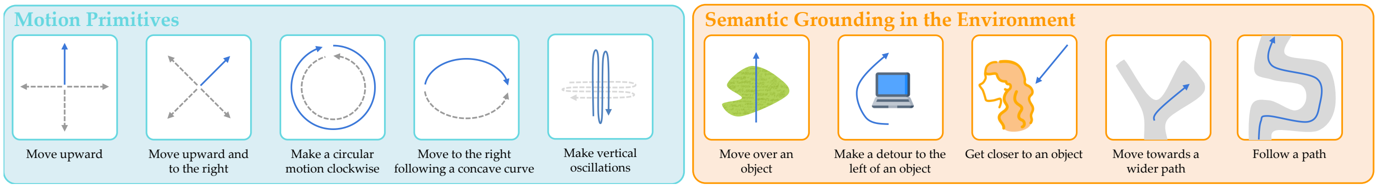


Fig. 5: Motion Diversity. 10 canonical motions are described with blue arrows and their corresponding descriptions. In the blue box, motion primitives are categorized based on the shape of the ideal path. Gray dashed arrows denote variants of the blue arrow, lying in the same category. The orange box shows motions that involve grounding in the environment, where the relationship between the robot and an instance in the environment is considered.

A. Collecting Human and Robot Demonstrations

As addressed in Section III-C, we use a mixture of human and robot demonstrations. In this section, we will explain how we collect human and robot demonstrations. Fig. 4 visualizes the final frames of the collected human and robot demonstrations. Our human demonstrations are collected by six different people to ensure ample variation in motion. For robot data, a single human expert teleoperates a Stretch robot with a Meta Quest 3 VR controller for manipulation tasks and a gamepad for navigation tasks. We record the agent's joint states and image observations with a fixed exocentric RGBD camera for visual consistency. For the pick and place, stir, shake, brush hair, and tidy hair tasks we collect trajectories in two orthogonal camera views to support future 3D motion understanding using multiview image observations. Since we use VLMs that input RGB images, we treat observations from different camera viewpoints as separate trajectories and focus on effectively representing the agent's motion on a 2D image frame. Section IV-B explains how we annotate the trajectories with task instructions and fine-grained motion descriptions.

We preprocess the collected demonstration observations using three motion representation methods (see Fig. 2):

- **optical flow** [22], [23]: visualizing the trajectories of all visible keypoints with rainbow colors. For each keypoint, its trajectory is drawn with a single color.
- **single keypoint tracking (MotIF)**: visualizing a trajectory of a single keypoint. For single point tracking on human data, we use mediapipe [36] and track the center of the hand pose. For robot data, we annotate 2D keypoints to identify point of interests in the initial frame of each episode, which is either a keypoint on the object that is being manipulated or the initial position of the robot's end effector. Then, we choose the keypoint nearest to the point of interest. For both human and robot data, temporal changes are shown with color gradient from white to green, ending with a red circle.
- **N-frame storyboard ($N = 2, 4, 9$)**: sampling N keyframes and stacking those frames into a single image. We use K-means clustering on the image embeddings of all frames to sample keyframes that are sufficiently different in latent space. Frame indices are annotated above each frame image.

B. Grounded Motion Annotations

In this section, we use "agent" to refer to either human or robot demonstrating the task. We use human annotators to label the motion in each video. While automated motion labeling using proprioception has previously been used, it is only applicable to short-horizon motions (¡10 frames), whereas our dataset contains long-horizon motions (¿300 frames). Compared to previous datasets which do not capture any information about *how the motions are grounded in the environment or to the user*, we consider motion diversity in two different axes: (1) path shape, and (2) semantic grounding in the environment.

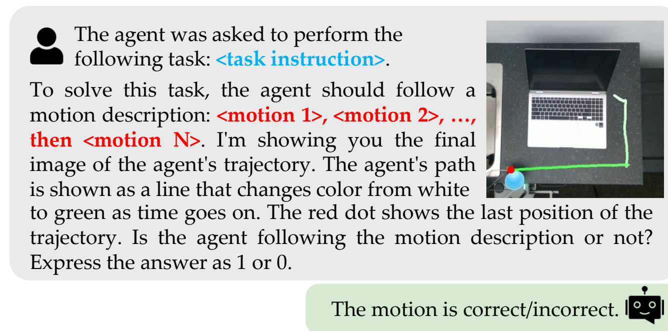


Fig. 6: Text Prompt Template shows how a motion composed of multiple primitives is described in the text input of a VLM.

Path Shape. As illustrated in Fig. 5, we first set a vocabulary of motion primitives in terms of path shape; direction and convexity of translation (e.g., move upward, follow a convex curve), direction of rotation (e.g., make a circular motion clockwise), and oscillatory movement (e.g., move up and down). Motion often consists of multiple motion primitives with different path shapes, such as moving right then downward, or moving right while oscillating vertically. When annotating such motions in language, we first list the primitives in the temporal order and prioritize the dominant ones among those that happen simultaneously.

References to Objects and Humans. Another important aspect of grounded motions are references to objects or humans in the scene. The orange box in Fig. 5 illustrates five common examples of motions in terms of semantic grounding. These motions gain specific meaning in relation to objects or humans in the environment. Based on the task specification regarding an object (e.g., avoid damaging the laptop) or a user (e.g., focus brushing the bottom part of the hair that is tangled), an agent's motion can be distinguished accordingly. For instance, a motion of shaking salt over food is defined not only by the path shape of the shaking movement but also by the spatial relationship between the agent and the food. We use a set vocabulary (e.g., move over, make a detour, get closer/farther, follow path) to annotate the grounded motions in language.

The final motion descriptions in MotIF-1K are constructed as combinations of descriptions in terms of path shape and references to objects and humans. With the annotated task instructions and motion descriptions, we construct text prompts that could be used to describe the agent's motion. Using this text prompt and the image representation of the trajectory as inputs, we train VLMs to predict the alignment between the text and image (see Section III-C). For the example in Fig. 6, the task instruction is "pick up the cup and place it to the lower left of the laptop" and the motion description could be "move downward and farther from the laptop, then move to the left". Here, two motion primitives, "move downward" and "move to the left", are combined with a grounded motion annotation,"move farther from the laptop". Given the text prompt and trajectory representation $I(\tau)$ in Fig. 6, our fine-tuned VLM outputs a binary value indicating whether the motion is correct or not.

V. Experiments

In this section we seek to answer the following questions: 1) How does MotIF compare to start-of-the-art models? 2) How important is robot data in understanding motion? and finally 3) What is the effect of visual motion representation? We compare our approach with state-of-the-art models and assess the benefits of co-training on human and robot data. We investigate the impact of different visual motion representations. All models are evaluated on the validation and test splits in MotIF-1K. See Section C for detailed analyses.

Baselines. We evaluate against GPT-4o, GPT-4V [37], and Gemini-1.5 Pro [38] as state-of-the-art API-based baselines. We also compare to the best performing pre-trained open LLaVA [35] models with various sizes (7B, 13B, 34B). To evaluate different visual motion representations, we compare our proposed single point tracking to full optical flow, N-frame storyboard ($N = 2, 4, 9$), and a single-frame image. All learned baselines are fine-tuned from the LLaVA-v1.5 7B model [35], [39], which was pre-trained on the Vicuna visual question answering (VQA) dataset [40].

Training Details. We train on all human and 100 robot demonstrations from MotIF-1K. For each demonstration, we construct one positive and 10 negative samples of image-text pairs. When fine-tuning the VLMs, we freeze the weights of the pre-trained visual encoder, CLIP ViT-L/14 [41] with an input size of 336px by 336px. We fine-tune the projection layer and the language model in the VLM using low-rank adaptation (LoRA) with cross-entropy loss for 30 epochs with a learning rate of $5e^{-5}$ and a batch size of 32. We use a single A100 GPU for fine-tuning.

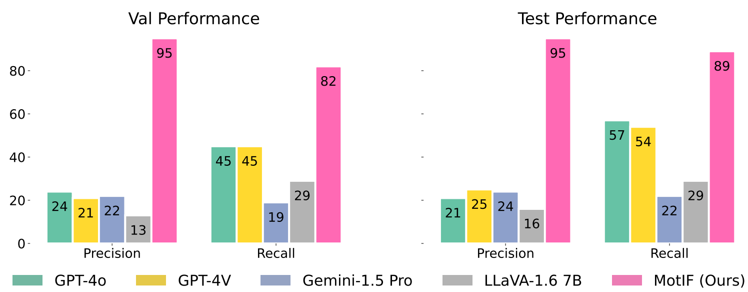
Evaluation Set and Metrics. All VLMs output a binary label indicating whether or not the agent's motion in the image aligns with the given task and motion descriptions. For off-the-shelf models, we convert natural language responses into their corresponding binary labels. We evaluate models on the validation and test split of MotIF-1K containing 129 and 134 robot demonstrations, respectively. The test split contains a set of unseen trajectories which vary in terms of camera viewpoints, motions, tasks, and environment in comparison to the training and validation split. Similar to the training data, we construct one positive and 10 negative samples per demonstration. For each experiment, we evaluate the performance of models using precision $(= \mathrm{TP}/(\mathrm{TP}+\mathrm{FP}))$ and recall $(= \mathrm{TP}/(\mathrm{TP}+\mathrm{FN}))$[1]. These metrics show the reliability and robustness of model outputs. High precision minimizes false positives, and high recall ensures most valid motions are identified.

How does MotIF compare to state-of-the-art VLMs? Our model is developed by fine-tuning a 7B LLaVA model with the MotIF-1K dataset. While LLaVA is pre-trained on general VQA tasks answering questions given a static image, we fine-tune the model using a motion representation that overlays a single keypoint trajectory on the final image. As shown in Fig. 7 (a), MotIF outperforms GPT-4o, GPT-4V, Gemini-1.5 Pro, and LLaVA in both precision and recall in the test split, by at least 280.0% and 56.1%, respectively. We include results in the Appendix that show our model robustly works in data out of training domain, such as unseen tasks and environments.

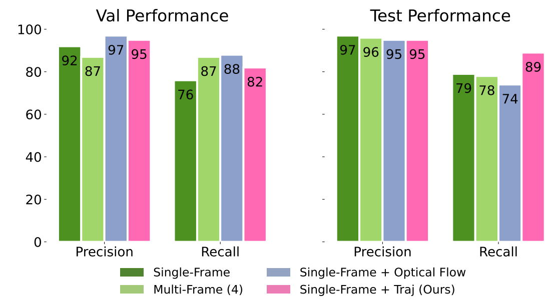
How does robot data impact performance? Fig. 8 shows how our model results in positive transfer from human to robot data, by co-training on full human data with 653 trajectories and a small subset of robot data. The plots show how precision and recall improve when adding 0 to 100 robot demos. Simply adding 20 robot demos improved performance significantly in recall by 151.5% and slightly in precision by 3.3%. With 50 and 100 robot demos, performance improves in all evaluation metrics. Interestingly, co-training on human data and 20 robot demos outperforms training solely on 20 robot demos by 63.8% and training on 100 robot demos by 6.7%. This implies that human data can be used to learn representations of grounded motions which can transfer to robots, despite the large embodiment gap. However, some robot data is still necessary; training only on human data shows very poor performance (33% recall) which is worse than random guessing. This might be because fine-tuning exclusively on human data specializes the model towards the human domain, which may cause performance decrease on robot data. Adding even a small amount of robot data significantly improves performance by encouraging the model to learn more generalized, embodiment-agnostic representations, demonstrating the crucial role of robot data in achieving robust performance.

How does motion representation affect performance? Fig. 7 (b) compares different visual motion representations trained on all human data and 100 robot trajectories used in the co-training experiment. Using the last frame without any motion representation obtains recall scores lower than 80% in validation and test splits. In contrast, using MotIF (single frame + Traj) improves recall by 7.9% and 12.7% on the validation and test split, respectively. Using full optical flow is effective in the validation split, but it generalizes poorly to test split, performing worse than MotIF. Perhaps the simplest baseline, inputting multiple frame storyboard, also fails to outperform MotIF. Performance degradation with multi-frame and optical flow is likely due to reduced image quality from either reduced image resolution or visual clutter from optical flow that covers relevant information.

[1]TP: True Positive, FP: False Positive, TN: True Negative, FN: False Negative



(a) Comparison with SOTA off-the-shelf VLMs



(b) Motion Representations

Fig. 7: Performance on MotIF-1K. (a) shows that our models outperforms state-of-the-art (SOTA) off-the-shelf models in validation and test splits. (b) We explore motion representations in terms of single-frame vs. multi-frame and the effectiveness of trajectory drawing. Single-frame with trajectory drawing demonstrates the highest recall in the test split, while other motion representations falter. Our approach identifies valid motions effectively and generalizes better than baselines. Among the three multi-frame representations ($N = 2, 4, 9$), we report the performance of the model that performs the best in the test split ($N = 4$).

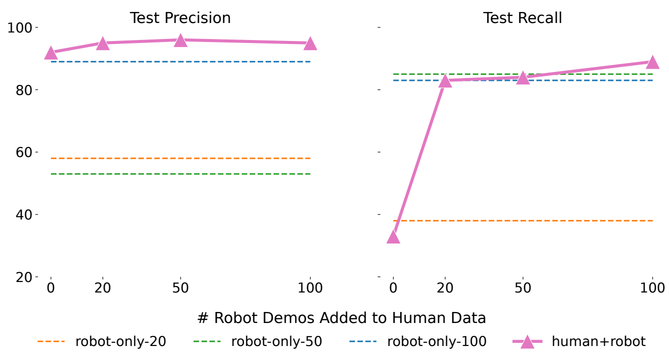


Fig. 8: Co-training. Performance improves with more robot demonstrations. Dashed lines indicate performance with robot-only data. Performance of the model trained only with human data is shown as human+robot with #robot=0 (the leftmost pink triangle).

VI. Discussion

Summary. Task specification in robotics often goes beyond simply stating *what* the objective is, and additionally consists of *how* a task should be done. As a step in this direction, we introduce a dataset, MotIF-1K, alongside a unique representation method and training technique, MotIF. Our findings demonstrate that our system can effectively provide assessments of nuanced robotic actions. The MotIF-1K dataset consisting of human and robot trajectories, captures the diverse ways in which tasks can be executed. We propose MotIF that uses this data to fine-tune open-source VLMs to detect a more nuanced notion of success. Our results demonstrate that MotIF can effectively assess success over these nuanced motions by leveraging a simple visual motion representation; overlaying the robot's trajectory on the image observation.

MotIF can determine the success of trajectories generated by any policy evaluating if the motions align with the task instruction and motion description. This can provide a signal for when to terminate an episode or to further refine the policy. We include examples of the following uses of MotIF on LLM generated policies [42]: MotIF as a success detector (App. Fig. 10), providing correction feedback to the policy, terminating or adapting robot policies (Fig. 9 (a)), and ranking trajectories (Fig. 9 (b)).

Limitations and Future Directions. One limitation of MotIF is the dependency on 2D visual motion representations, which might not capture all aspects of complex 3D motions. Future work could incorporate RGB-D and multi-view images, or 3D visual representations overlaid on the image for a more comprehensive understanding of 3D space using our dataset. Another direction is to use MotIF as a reward signal for reinforcement learning. MotIF can act as a motion discriminator when training RL policies that can potentially more accurately reflect user preferences and contextual appropriateness. Future work might create a VLM that outputs natural language responses, which could be used to not only evaluate binary success but also perform reasoning on robotic motions.

References

[1] Sriram Yenamandra, Arun Ramachandran, Karmesh Yadav, Austin Wang, Mukul Khanna, Theophile Gervet, Tsung-Yen Yang, Vidhi Jain, Alexander William Clegg, John Turner, et al. Homerobot: Open-vocabulary mobile manipulation. *arXiv:2306.11565*, 2023.

[2] Yuqing Du, Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, and Serkan Cabi. Vision-language models as success detectors. *arXiv:2303.07280*, 2023.

[3] Lin Guan, Yifan Zhou, Denis Liu, Yantian Zha, Heni Ben Amor, and Subbarao Kambhampati. " task success" is not enough: Investigating the use of video-language models as behavior critics for catching undesirable agent behaviors. *arXiv:2402.04210*, 2024.

[4] 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. *arXiv:2310.12931*, 2023.

[5] Minae Kwon, Sang Michael Xie, Kalesha Bullard, and Dorsa Sadigh. Reward design with language models. *arXiv:2303.00001*, 2023.

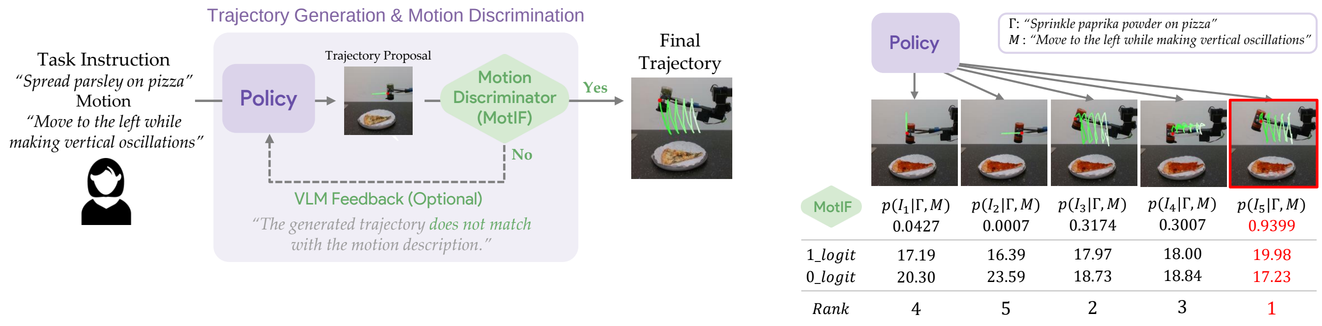
[6] Hengyuan Hu and Dorsa Sadigh. Language instructed reinforcement learning for human-ai coordination. In *ICML*, 2023.

[7] Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis. *Arxiv preprint arXiv:2306.08647*, 2023.

[8] Minyoung Hwang, Luca Weihs, Chanwoo Park, Kimin Lee, Aniruddha Kembhavi, and Kiana Ehsani. Promptable behaviors: Personalizing multi-objective rewards from human preferences. *arXiv:2312.09337*, 2023.

[9] Seongyun Lee, Seungone Kim, Sue Hyun Park, Geewook Kim, and Minjoon Seo. Prometheus-vision: Vision-language model as a judge for fine-grained evaluation. *arXiv:2401.06591*, 2024.

[10] Yufei Wang, Zhanyi Sun, Jesse Zhang, Zhou Xian, Erdem Biyik, David Held, and Zackory Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. *arXiv:2402.03681*, 2024.



(a) Refining and Terminating Robot Planning

(b) Ranking Trajectories

Fig. 9: Examples of using MotIF in robot control. (a) MotIF can close the loop of any existing open-loop controlled system by automatically determining success on executing proper motion and giving this as a feedback to the system. We use an LLM as the policy that generates the sequence of the robot joint states. (b) We can use MotIF to rank trajectories, which is a useful application for sampling based planners. $p(I_k|\Gamma, M)$ denotes how likely the motion in the k^{th} image corresponds to the given task instruction Γ and motion description M.

[11] Sumedh Sontakke, Jesse Zhang, Séb Arnold, Karl Pertsch, Erdem Bı yık, Dorsa Sadigh, Chelsea Finn, and Laurent Itti. Roboclip: One demonstration is enough to learn robot policies. In *NeurIPS*, 2023.

[12] Juan Rocamonde, Victoriano Montesinos, Elvis Nava, Ethan Perez, and David Lindner. Vision-language models are zero-shot reward models for reinforcement learning. *arXiv:2310.12921*, 2023.

[13] Kate Baumli, Satinder Baveja, Feryal Behbahani, Harris Chan, Gheorghe Comanici, Sebastian Flennerhag, Maxime Gazeau, Kristian Holsheimer, Dan Horgan, Michael Laskin, et al. Vision-language models as a source of rewards. *arXiv:2312.09187*, 2023.

[14] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv*, 2022.

[15] Pierre Sermanet, Kelvin Xu, and Sergey Levine. Unsupervised perceptual rewards for imitation learning. *arXiv:1612.06699*, 2016.

[16] Jensen Gao, Bidipta Sarkar, Fei Xia, Ted Xiao, Jiajun Wu, Brian Ichter, Anirudha Majumdar, and Dorsa Sadigh. Physically grounded vision-language models for robotic manipulation. *arXiv:2309.02561*, 2023.

[17] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Danny Driess, Pete Florence, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. *arXiv:2401.12168*, 2024.

[18] Zawalski Michał, Chen William, Pertsch Karl, Mees Oier, Finn Chelsea, and Levine Sergey. Robotic control via embodied chain-of-thought reasoning. *arXiv:2407.08693*, 2024.

[19] 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*, 2024.

[20] Jacob Krantz, Aaron Gokaslan, Dhruv Batra, Stefan Lee, and Oleksandr Maksymets. Waypoint models for instruction-guided navigation in continuous environments. In *CVPR*, 2021.

[21] Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. *arXiv:2311.01977*, 2023.

[22] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adria Recasens, Lucas Smaira, Yusuf Aytar, Joao Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 2022.

[23] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement. In *CVPR*, 2023.

[24] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation. *arXiv:2405.01527*, 2024.

[25] Mel Vecerik, Carl Doersch, Yi Yang, Todor Davchev, Yusuf Aytar, Guangyao Zhou, Raia Hadsell, Lourdes Agapito, and Jon Scholz. Robotap: Tracking arbitrary points for few-shot visual imitation. *arXiv:2308.15975*, 2023.

[26] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv:2401.00025*, 2023.

[27] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv:2406.10721*, 2024.

[28] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. *arXiv:2403.03174*, 2024.

[29] Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. *arXiv:2402.07872*, 2024.

[30] Dantong Niu, Yuvan Sharma, Giscard Biamby, Jerome Quenum, Yutong Bai, Baifeng Shi, Trevor Darrell, and Roei Herzig. Llarva: Vision-action instruction tuning enhances robot learning. *arXiv:2406.11815*, 2024.

[31] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

[32] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *ACL*, 2019.

[33] Minae Kwon, Hengyuan Hu, Vivek Myers, Siddharth Karamcheti, Anca Dragan, and Dorsa Sadigh. Toward grounded social reasoning. *arXiv:2306.08651*, 2023.

[34] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

[35] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *NeurIPS*, 2023.

[36] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. Mediapipe: A framework for building perception pipelines. *arXiv:1906.08172*, 2019.

[37] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv:2303.08774*, 2023.

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

[39] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. *arXiv:2310.03744*, 2023.

[40] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *https://vicuna.lmsys.org*, 2023.

[41] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[42] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *ICRA*, 2023.

[43] Carl Doersch, Pauline Luc, Yi Yang, Dilara Gokay, Skanda Koppula, Ankush Gupta, Joseph Heyward, Ignacio Rocco, Ross Goroshin, João Carreira, and Andrew Zisserman. BootsTAP: Bootstrapped training for tracking any point. *arXiv*, 2024.

Category	Task	Motion Description Examples	Demonstrations Human	Demonstrations Robot
Non-Interactive	Outdoor Navigation	move in the shortest path make a detour to the left and follow the walkway, avoiding moving over the grass	-	✓
	Indoor Navigation	move in the shortest path make a detour to the right of the long table, avoiding collision with chairs	-	✓
	Draw Path	make a triangular motion clockwise move upward and to the right	-	✓
Object-Interactive	Shake	move up and down 4 times completely flip the object to the right and flip it back to its initial state	✓	✓
	Pick and place	move downward and to the left move downward while getting farther from <obstacle>, then move to the left	✓	✓
	Stir	make 2 circular motions counter-clockwise move upward, then move downward while making diagonal oscillations	✓	✓
	Wipe	move to the right and move to the left, repeating this sequence 2 times move to the right, making diagonal oscillations	✓	✓
	Open/Close the cabinet	move to the right move upward and to the left	✓	-
	Spread Condiment	move to the left and to the right move to the left while making back and forth oscillations	-	✓
User-Interactive	Handover	move upward and to the left move downward and to the right following a concave curve	✓	-
	Brush hair	move downward while making horizontal oscillations make 5 strokes downward, increasing the starting height of each stroke	✓	✓
	Tidy hair	move downward and to the right following a convex curve make a circular motion clockwise, move upward, then move downward and to the right	✓	✓
	Style hair	move to the right shortly, then move to the left following a concave curve make a circular motion clockwise, gradually increasing the radius of the circle	✓	✓

App. Table I: List of Tasks and Motion Descriptions. The collected dataset contains 653 human and 369 robot demonstrations across 13 task categories, 36 task instructions, and 239 motion descriptions. Checkmarks denote which agent (human/robot) demonstrations exist for each task. The table provides two motion description examples for each task.

APPENDIX

We provide additional details and analyses of MotIF and MotIF-1K in the appendix. Section A provides details for how we construct and annotate the dataset. Section B provides implementation details of the proposed motion discriminator, MotIF. Section C provides additional experiment results and detailed analyses on the results in the paper. Section D provides the implications of MotIF in real robot planning, with detailed examples.

A. MotIF-1K Dataset Details

A.1 Grounded Motion Annotations

Path Shape. For translation motions, we consider the direction of the movement such as *move upward* or *move upward and to the right*. As motions could be forming a curve or part of a shape, we also consider the convexity of the path or compare the path to figures (e.g., circle, triangle, and square) that resemble the shape being traced. For cyclic motions, the direction such as clockwise/counter-clockwise are considered. Many trajectories may be a composition of simpler motions. Side-to-side motions could be described as *move to the right and to the left N times* or vice versa. When one motion component dominates the overall agent behavior among multiple motion components, we describe the dominating motion first and include the remaining details as modifiers. For instance, a motion can be a combination of downward translation and side-to-side motion, thereby resulting in *move downward, making horizontal oscillations*.

Grounding in the Environment. Building on top of VLMs' semantic reasoning capabilities by leveraging their world knowledge, we address how robotic motions can be grounded in the environment. The most basic grounded motion is moving over or avoiding an instance in the environment. As shown in the second leftmost motion in the orange box of Fig. 5, to avoid an obstacle or a hazardous region, the agent can make a detour to the left or right of the corresponding instance. Another example could be getting closer to or farther from an instance, even when the instance is not essentially a target. This could happen when people want legibility or predictability of the agent's motion. The instance could also be a path as shown in the two rightmost motions, such as walkways, drawings on a whiteboard, or spills on a table.

B. Implementation and Experiment Details

B.1 Image Preprocessing

To preprocess images, we first crop all images to be square. Next, depending on the visual motion representation, the images are sampled and aggregated into a storyboard, or have trajectories of single or multiple keypoints overlaid.

Multi-frame Storyboard. For constructing multi-frame storyboards, we investigate $N = 2, 4, 9$ numbers of keyframes, where the keyframes are sampled from K-means clustering of N clusters based on the ResNet18 embeddings of the images at each timestep of a given visual trajectory.

Optical flow. To extract optical flow from a robot's motion video, we use the BootsTAPIR [43], [23] algorithm. We run

Parameter	Value
Inlier Point Threshold	0.01
Minimum Fraction of Inlier Points	0.95
# Refinement Passes	2
Foreground Inlier Threshold	0.01
Minimum Fraction of Foreground Inlier Points	0.60

App. Table II: Optical Flow Parameters.

Category	Task	Closed Model						Open Model		Ours	
		GPT-4o		GPT-4V		Gemini-1.5 Pro		LLaVA-1.6-7B		MotIF	
Non-Interactive	navigation	0.50	1.00	1.00	1.00	1.00	1.00	0.00	0.00	1.00	1.00
Object-Interactive	shake	0.24	0.27	0.33	0.34	0.35	0.24	0.10	0.18	0.98	0.91
	stir	0.25	0.45	0.08	0.18	0.00	0.00	0.28	0.45	0.79	1.00
	pick and place	0.00	0.00	0.08	0.29	0.00	0.00	0.14	0.43	1.00	1.00
	spread condiment	0.20	1.00	0.15	0.55	0.00	0.00	0.00	0.00	1.00	0.82
User-Interactive	brush hair	0.26	0.48	0.18	0.58	0.30	0.23	0.20	0.45	0.96	0.74
	style hair	0.26	0.85	0.23	0.46	0.25	0.42	0.00	0.00	1.00	0.77
	tidy hair	0.17	0.30	0.38	0.80	0.40	0.22	0.18	0.70	1.00	0.40
Average		0.24	0.45	0.21	0.45	0.22	0.19	0.13	0.29	**0.95**	**0.82**

App. Table III: Comparison with SOTA off-the-shelf VLMs in MotIF-1K val split. We report precision ($= TP/(TP + FP)$) and recall ($= TP/(TP + FN)$) in gray and white cells, respectively.

Category	Task	Closed Model						Open Model		Ours	
		GPT-4o		GPT-4V		Gemini-1.5 Pro		LLaVA-1.6-7B		MotIF	
Object-Interactive	shake	0.33	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.75	0.75
	stir	0.00	0.00	0.17	0.50	0.00	0.00	0.00	0.00	0.33	1.00
	spread condiment	0.20	0.75	0.17	0.52	0.15	0.11	0.11	0.13	0.98	0.89
User-Interactive	brush hair	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00
	style hair	0.19	0.44	0.23	0.32	0.18	0.36	0.07	0.20	1.00	0.96
	tidy hair	0.23	0.50	0.42	0.72	0.43	0.26	0.28	0.54	1.00	0.85
Average		0.21	0.57	0.25	0.54	0.24	0.22	0.16	0.29	**0.95**	**0.89**

App. Table IV: Comparison with SOTA off-the-shelf VLMs in MotIF-1K test split. We report precision ($= TP/(TP + FP)$) and recall ($= TP/(TP + FN)$) in gray and white cells, respectively.

random sample consensus (RANSAC) with parameters in App. Table II.

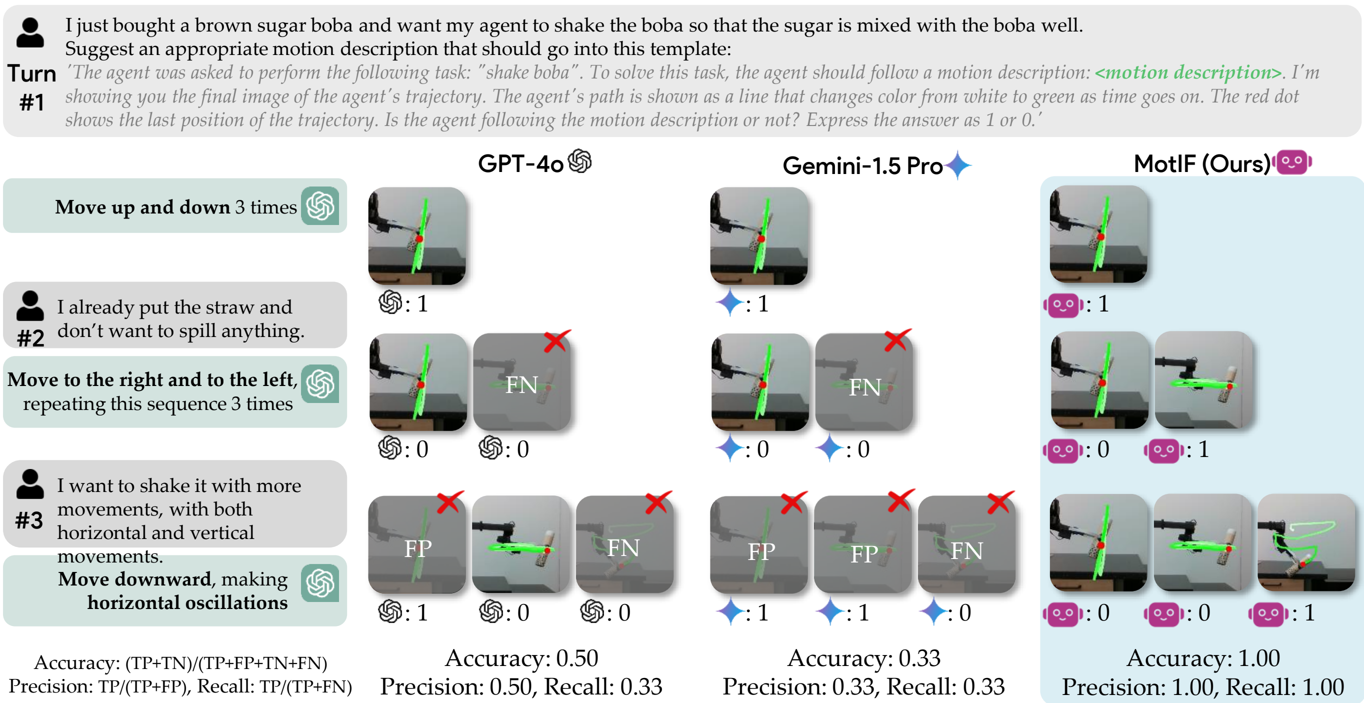
Evaluation Details. For interpretability, the scale of the y-axis in Fig. 7 and Fig. 8 are 100 times larger than the actual scale of evaluation metrics; precision and recall.

C. Additional Experiments and Analyses

C.1 Detailed Analysis

Comparison with off-the-shelf VLMs. App. Table III and App. Table IV show the performance of off-the-shelf VLMs and MotIF for each task category. The results show that our model outperforms existing VLMs in most tasks. In the val split of MotIF-1K, API-based closed VLMs show the weakest performance in 'pick and place' and the strongest performance in 'navigation'. In the test split of MotIF-1K, closed VLMs show the weakest performance in 'brush hair'. On the other hand, our model scores high precision and recall in both val and test splits, scoring 0.95 precision and 0.89 recall in the test split in average. Since the test split contains numerous data samples with unseen motions, tasks, and environments, the results demonstrate the generalization capability of our model.

App. Fig. 10 shows a toy scenario comparing GPT-4o, Gemini-1.5 Pro, and MotIF, along a conversation between a user and an LLM (ChatGPT). Given a user's task specification (*e.g.*, I don't want to spill anything), an LLM can generate a motion description that corresponds to the user's situation or preference. When motion description is specified from the LLM, we retrieve the best corresponding trajectory from MotIF-1K, based on the judgement of a human expert. Then, we ask each VLM to determine whether the robot's motion corresponds to the motion description or not. At each turn, we ask the VLMs to match the motion description proposed from the LLM with the current and all previously retrieved motions. After three turns of conversation, we calculate the performance of motion discrimination using the VLMs. Results in App. Fig. 10 show that our model achieves the highest accuracy, precision, and



App. Fig. 10: Comparison between MotIF and state-of-the-art closed VLMs. We compare three VLMs: our model, GPT-4o, and Gemini-1.5 Pro, along a conversation between a user and an LLM. The user specifies the task and the LLM generates an appropriate motion description. The performance of each VLM is measured by predicting if the robot motions align (VLM response: 1) with motion descriptions suggested from the LLM or not (VLM response: 0), where the images are not included in training our model. Comparing the accuracy, precision, and recall for each model, MotIF shows the highest performance in all metrics.

recall, successfully understanding robotic motions in all cases.

Another interesting aspect of off-the-shelf VLMs is that they can refuse to answer or output a neutral response. For instance, Gemini-1.5 Pro often outputs "The answer is: **Cannot be determined**.", "We cannot tell, so we cannot provide a 1 or 0 answer.", or "it's impossible to tell if the agent followed the instruction". GPT-4V frequently outputs "I'm sorry, I can't provide assistance with that request." or "I'm sorry, but I cannot provide assistance with tasks or interpret content that includes tracking or analyzing the movements".

Co-training on Human and Robot Datasets. App. Table V and Fig. 8 show detailed results of how co-training on human and robot data improves performance and boosts positive transfer. To focus on the effect of co-training, we fix the visual motion representation as single point tracking. Adding just 20 robot demonstrations to all human data, the recall score significantly improves by 169.7% in the test split.

Visual Motion Representations. App. Table VI shows detailed performance values of models trained with different visual motion representations. Using the raw last frame shows the highest precision in the test split, while the recall score is 11.2% lower than using the proposed motion representation, drawing the trajectory of a single keypoint. This implies that without the information of the trajectory, the VLM can easily output false negatives. Bringing insights from GPT-4o generating storyboard when analyzing videos, we also demonstrate the performance using multi-frame storyboards with 2, 4, and 9 keyframes. Interestingly, higher number of frames does not consistently improve performance, which might be due to the lower resolution of images when multiple images are concatenated.

Model	Training Data		Val Performance		Test Performance	
	# human demos	# robot demos	Precision	Recall	Precision	Recall
Human Only	653	0	0.63	0.50	0.92	0.33
Robot Only	653	20	0.55	0.26	0.58	0.38
	653	50	0.68	0.87	0.53	0.85
	653	100	0.83	**0.89**	0.89	0.83
Human+Robot Co-trained	653	20	0.77	0.68	0.95	0.83
	653	50	0.92	**0.89**	**0.96**	0.84
	653	100	**0.95**	0.82	0.95	**0.89**

App. Table V: Co-training Performance in MotIF-1K Val and Test splits. We report precision $(= TP/(TP + FP))$ and recall $(= TP/(TP + FN))$ in gray and white cells, respectively.

Visual Motion Representation			Val Performance		Test Performance	
Type	Trajectory Drawing	# frames	Precision	Recall	Precision	Recall
Raw Image	✗	1	0.92	0.76	**0.97**	0.79
Multi-Frame Storyboard	✗	2	0.89	0.81	0.92	0.86
	✗	4	0.87	0.87	0.96	0.78
	✗	9	0.87	0.87	0.95	0.83
Optical Flow	✓	1	**0.97**	**0.88**	0.95	0.74
MotIF (Ours)	✓	1	0.95	0.82	0.95	**0.89**

App. Table VI: Comparison of Visual Motion Representations in MotIF-1K Val and Test splits. We report precision $(= TP/(TP + FP))$ and recall $(= TP/(TP + FN))$ in gray and white cells, respectively.

C.2 Visualization of MotIF Outputs

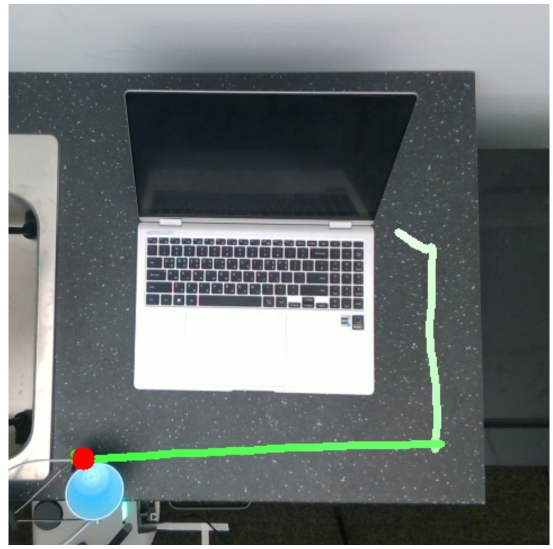
In this section, we visualize four trajectories in the val and test splits of MotIF-1K. For each trajectory, we analyze the output of MotIF given four different motion descriptions.

App. Table VII visualizes the 106^{th} trajectory in the val split of MotIF. While motion description 1 is included in the training data, our VLM generalizes to understanding unseen motion descriptions (2, 3, and 4). This may be due to the effect of having diverse human demonstrations on grounded robot motions, since there are similar motion descriptions in human data such as "move downward, getting closer to the laptop, and then move to the left".

App. Table VIII shows an example of our model understanding a robotic motion from an unseen camera viewpoint. Despite the change of the camera height and angle, our model effectively understands a paraphrased instruction (motion description 2) and an instruction with a small tweak on the core directional component (motion description 4).

App. Table IX demonstrates our model understands robotic motions even when the interacting object is changed. While parmesan cheese is the only object for this task in the training data, MotIF does accurately determine ground truth and paraphrased motion descriptions align with the robot's motion.

App. Table X shows a trajectory of a robot delivering lemonade to the door, while avoiding moving over the manhole. This example demonstrates the grounded motion understanding capability of our model such as moving over or making a detour to avoid a specific instance (e.g., manhole) in the environment.



Task Instruction: pick up the cup and place it to the lower left of the laptop	**MotIF Output**	**Prediction Correctness**
Motion Description 1: move downward, then move to the left	1	✔
Motion Description 2: move farther from the laptop, move downward, then move to the left	1	✔
Motion Description 3: move downward and to the left, passing over the laptop	0	✔
Motion Description 4: move over the laptop	0	✔

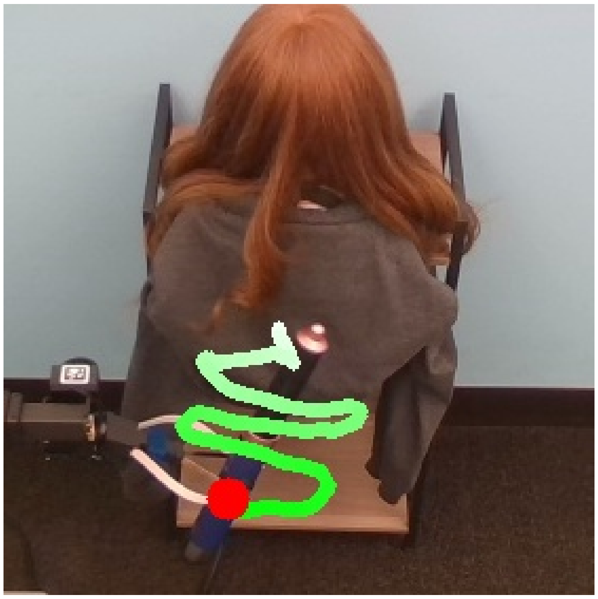
App. Table VII: Trajectory Visualization of a Grounded Motion. For the 106^{th} robot motion in the validation split of MotIF-1K, we get the output of MotIF on 4 different motion descriptions. The results show that our model effectively understands how the robot's motion is grounded in the environment, especially considering the spatial relationship with the laptop.

C.3 Qualitative Analysis on Off-the-shelf VLMs

To check the motion understanding capabilities of off-the-shelf VLMs, we perform a toy experiment on asking several questions to the VLMs, given a video or an image of the robot's motion brushing hair (see App. Fig. 11). The results in this section show that off-the-shelf VLMs such as GPT-4V and GPT-4o fail to describe the robot's motion and its semantic grounding in the environment.

App. Fig. 12 shows the response of GPT-4V and GPT-4o, given the robot's motion video, without any trajectory overlaid, and a simple text prompt 'Describe the robot's motion'. While GPT-4V fails to analyze the video, GPT-4o generates a long detailed response explaining the robot's motion. Specifically, GPT-4o first extracts keyframes with its internal algorithm and creates a storyboard of the keyframes. Then, it detects the trajectory of the robot's end effector. Although the model's approach is logically valid, the model eventually describes the robot's motion as "The robot's end effector moves in a non-linear path with variations in both the x and y directions, suggesting a complex motion.", which implies neither a specific shape of the motion or how it is grounded in the scene.

We also check if off-the-shelf VLMs can better understand



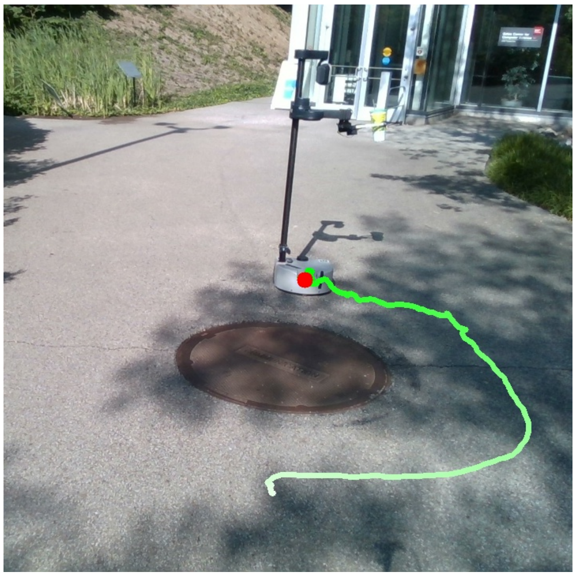
Task Instruction: curl hair	MotIF Output	Prediction Correctness
Motion Description 1: move downward, while making horizontal oscillations	1	✔
Motion Description 2: move downward, while making side-to-side movements	1	✔
Motion Description 3: move downward	0	✔
Motion Description 4: move downward, while making vertical oscillations	0	✔

App. Table VIII: Trajectory Visualization of a Motion in an Unseen Camera Viewpoint. For the 14^{th} robot motion in the test split of MotIF-1K, we get the output of MotIF on 4 different motion descriptions. Our model effectively understands the robot's motion in an unseen camera viewpoint.

robotic motions given the proposed visual motion representation, by overlaying the trajectory of a single keypoint in the image. App. Fig. 13 shows an example of passing a video with the robot's trajectory visualized. While GPT-4o outputs a response with detailed explanation, the described motion with multiple downward strokes does not match with the actual motion as shown in App. Fig. 11, which is moving downward while making horizontal oscillations. This implies that existing VLMs might rely on the language prior, because making multiple downward strokes is a common brushing technique in general.

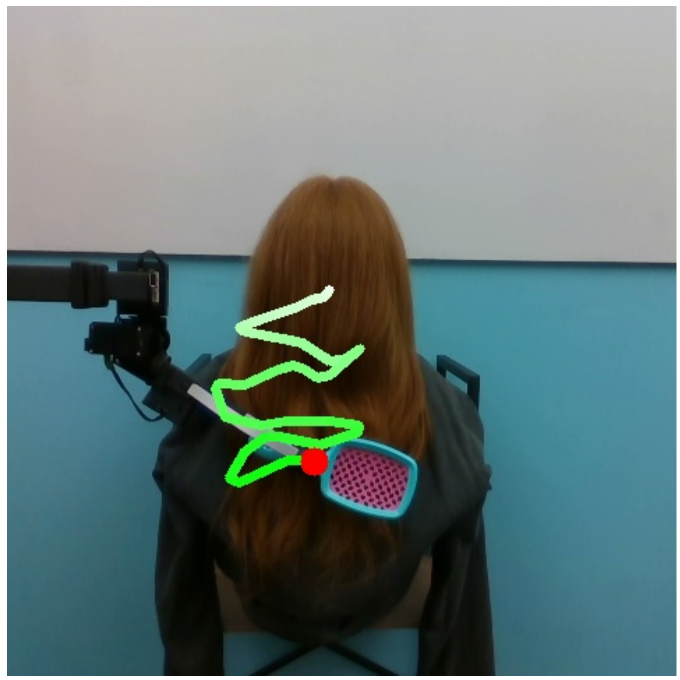
Task Instruction: sprinkle parsley on pizza	MotIF Output	Prediction Correctness
Motion Description 1: move to the left, while making vertical oscillations and alternating rotations	1	✔
Motion Description 2: move to the left, while making vertical oscillations	1	✔
Motion Description 3: move to the left, while making vertical shaking movements	1	✔
Motion Description 4: move to the left in a straight line	0	✔

App. Table IX: Trajectory Visualization of a Motion with an Unseen Object. For the 166^{th} robot motion in the test split of MotIF-1K, we get the output of MotIF on 4 different motion descriptions. While the training data only contains motions of spreading condiments on pizza with parmesan cheese, our model effectively understands the robot's motion with an unseen object, parsley.



Task Instruction: deliver lemonade	MotIF Output	Prediction Correctness
Motion Description 1: make a detour to the right of the manhole	1	✔
Motion Description 2: move forward, making a detour to the right of the manhole	1	✔
Motion Description 3: move forward in the shortest path	0	✔
Motion Description 4: move forward in a straight line, moving over the manhole	0	✔

App. Table X: Trajectory Visualization of an Unseen Grounded Motion. For the 181^{th} robot motion in the val split of MotIF-1K, we get the output of MotIF on 4 different motion descriptions. The results show that our model effectively understands the semantic grounding of the robot's motion in a navigation task.



App. Fig. 11: Example Trajectory of a robot brushing hair. The robot is brushing the hair by moving downward, while making horizontal oscillations.

```
# Question
Describe the robot's motion.
```

```
# GPT-4V Answer
It seems you've uploaded a video file related to the robot's motion. Let's analyze the content
of this video to describe the motion in detail. I'll review the video and provide a
description of the robot's movements. Please hold on for a moment.

The video is ready for viewing. Please watch the robot's motion in the video above. After
viewing, I can help with any specific analysis or description you need about the robot's
movements. If you have specific aspects of the motion you're interested in, such as speed,
trajectory, or interaction with objects, let me know!
```

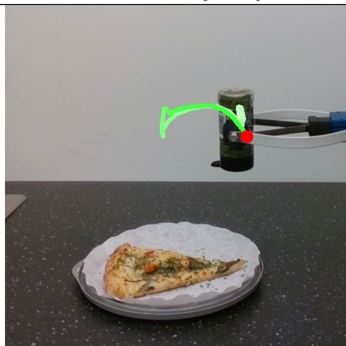
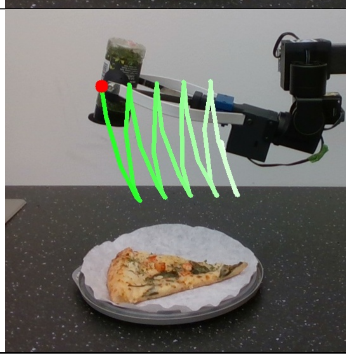
```
# GPT-4o Answer
The first frame of the video shows the side view of a robot performing a task. To describe the
robot's motion, I'll need to analyze the frames in the video to identify the movement pattern.
I'll extract and analyze key frames to provide a detailed description of the motion.

Let's proceed by extracting key frames and tracking the movement of the robot's end effector.

The extracted key frames from the video show the robot's motion over time. To provide a
detailed description of the robot's motion, let's analyze these frames:

The robot's end effector appears to be moving in a specific path throughout the video.
The trajectory includes several distinct positions and orientations at different times.
To describe the motion more precisely, I will track the movement of the robot's end effector
across these frames. This will help in visualizing and understanding the pattern of movement.

The tracked positions of the robot's end effector across the frames indicate its motion
pattern. Here is the detailed description of the robot's motion:

Initial Position: The robot starts with the end effector near the top left quadrant of the
frame.

Motion Path: The end effector follows a somewhat curved path, moving diagonally towards the
bottom right quadrant.

Key Positions:
Frame 1: (146, 166)\n Frame 2: (174, 173)\n Frame 3: (303, 221)\n Frame 4: (260, 236)
Frame 5: (296, 245)\n Frame 6: (265, 225)\n Frame 7: (300, 266)\n Frame 8: (245, 282)
Frame 9: (159, 258)\n Frame 10: (289, 319)\n Frame 11: (262, 353)\n Frame 12: (318, 336)

Overall Trajectory: The robot's end effector moves in a non-linear path with variations in
both the x and y directions, suggesting a complex motion.
The movement is likely part of a task requiring precise positioning and orientation changes.
If you have any specific aspects of the motion or additional details you'd like to focus on,
please let me know!
```

App. Fig. 12: Example prompt and answer where GPT-4o is asked to describe the robot's motion. The model is given a video of the robot's motion with the question prompt. The video does not include any trajectory visualization. Given the video, the model fails to describe the semantic meaning of the robot's motion.

```
# Question
Describe the robots motion. The task is 'brushing hair'. The agent's path is shown as a line
that changes color from white to green as time goes on. THe red dot shows the last position of
the trajectory.
```

```
# GPT-4o Answer
I've loaded the video of the robot's motion for the task of brushing hair. To provide a
detailed description of the motion, I will analyze the video and describe the path and color
change over time, along with the final position indicated by the red dot.

Let's start by extracting key frames and analyzing the motion trajectory.
<12x1 storyboard displayed>

The key frames extracted from the video provide a visual representation of the robot's motion
for the task of brushing hair. Here's a detailed description of the motion:

Initial Position: The robot starts with the brush positioned near the top of the hair, as
indicated by the white color at the beginning of the trajectory.

Motion Path: The robot moves the brush in a downward stroke, following a smooth and continuous
path. The color of the trajectory gradually changes from white to green, indicating the
progression of time.

Path Continuation: The robot completes the first downward stroke and moves the brush back up
to the starting position, preparing for another stroke. The color of the line continues to
change from white to green, showing the ongoing motion.

Red Dot: The red dot at the end of the trajectory indicates the last position of the brush at
the end of the video. This dot signifies the completion of the brushing motion.

Overall, the robot's motion is characterized by a repetitive up-and-down brushing movement,
with the color-changing line visualizing the time progression of the motion. The red dot marks
the final position of the brush.
```

App. Fig. 13: Example prompt and answer where GPT-4o is asked to describe the robot's motion. The model is given a video of the robot's motion with the robot's trajectory overlaid, and the question prompt.

Turn	Trajectory	Policy	MotIF output	Prob. of "1"
1		horizontal_shaking$(2, 0.1, (0.1, 1.0), 0.15, 2, roll_angle = 3.14)$	0	0.392
2		vertical_shaking$(2, 0.1, (0.2, 1.0), 0.1, 4)$	1	0.999

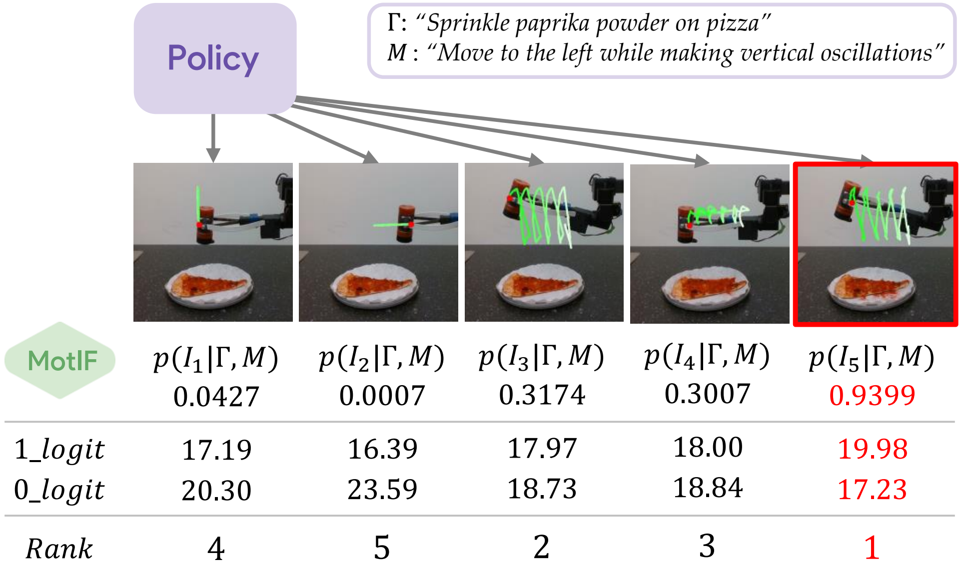
App. Table XI: Refining and Terminating Policy with MotIF Output as Feedback. For task instruction "sprinkle parsley on pizza" and motion description "move to the left while making vertical oscillations", an LLM suggests to use horizontal_shaking function with wrist roll rotations in the first turn. Using the output of MotIF or the probability of output "1" as a signal, we can terminate when the output is "1" or the probability is greater than a threshold, and refine the policy otherwise.

D. Examples of Using MotIF

In this section, we provide a detailed examination of the qualitative results demonstrating the utility of MotIF in evaluating policy-generated trajectories. Specifically, we focus on how MotIF acts as a critical component in assessing whether the motions of a robot are in alignment with the given task instructions and motion descriptions. This capability is essential for determining appropriate moments to terminate episodes or to offer corrective feedback to the policy module responsible for trajectory generation. We illustrate various applications of MotIF, including its role as a success detector, providing correction feedback, terminating or adapting robot policies. Additionally, we discuss how MotIF can be used to rank trajectories, as depicted in App. Fig. 14 and App. Fig. 15. These examples underscore the method's potential in various robotic applications.

D.1 Refining and Terminating Robot Planning

Policy. Bringing an insight from code-as-policies [42], we use an LLM (ChatGPT) to generate the agent policy as a code. We use the prompts in App. Fig. 16 to generate a function that outputs the sequence of end effector states. We make the functions to be parameterized, so that we can utilize the output of MotIF as a correction feedback to adjust the parameters. In App. Table XI, we show an example of refining a policy to generate a robot's motion corresponding to "move to the left while making vertical oscillations".

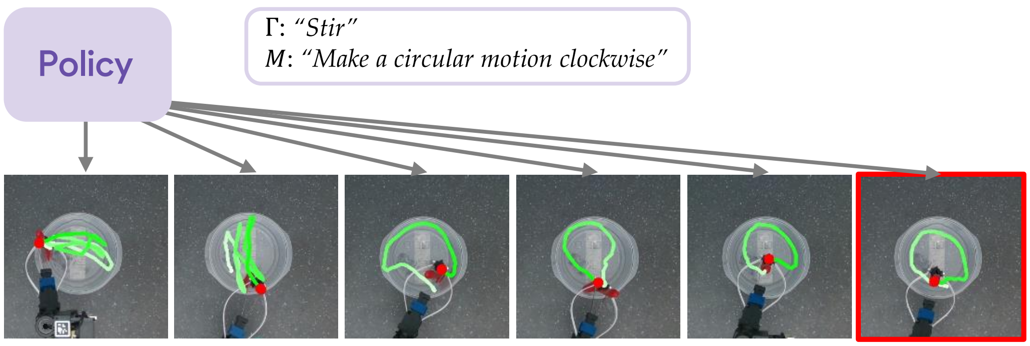


App. Fig. 14: Ranking Trajectories - Case 1. Unseen Object. We can use the probability of the output token being "1" or the logits for tokens "1" and "0" as a criterion to rank the trajectories. The bottom row shows the rank determined by the probability in the first row.

D.2 Ranking Trajectories with MotIF

Another usage of MotIF can be ranking trajectories, as shown in App. Fig. 14 and App. Fig. 15. To rank trajectories, we can use the probability of the output token being "1" or the logits for tokens "1" and "0" as criteria. While these different criteria make a consensus on choosing the best trajectory in both examples (the rightmost trajectories in App. Fig. 14 and App. Fig. 15), the ranking of the other trajectories change depending on the criteria. For instance, in App. Fig. 14, while the rank from the first to the last trajectories is $(4, 5, 2, 3, 1)$ using the probability, it becomes $(4, 5, 3, 2, 1)$ using the logit for token "1". Despite a small difference on the rankings, the two leftmost trajectories always rank the lowest. This is a desirable result since those trajectories only show a simple translation motion, without any oscillations.

App. Fig. 15 shows how we can reduce human effort in choosing high-quality demonstrations. In this example, robot motions get higher scores as the shape becomes closer to a perfect circle. Also, the two leftmost trajectories with repeated vertical or horizontal movements are accurately determined as failed trajectories.



MotIF	$p(I_1\|\Gamma, M)$	$p(I_2\|\Gamma, M)$	$p(I_3\|\Gamma, M)$	$p(I_4\|\Gamma, M)$	$p(I_5\|\Gamma, M)$	$p(I_6\|\Gamma, M)$
	0.0003	0.00003	0.9960	0.7800	0.9990	0.9993
1_logit	16.16	15.23	21.59	18.75	22.53	22.66
0_logit	24.41	25.75	16.08	17.48	15.63	15.44
$Rank$	5	6	3	4	2	1

App. Fig. 15: Ranking Trajectories - Case 2. Unseen Motions. We can use the probability of the output token being "1" or the logits for tokens "1" and "0" as a criterion to rank the trajectories. The bottom row shows the rank determined by the probability in the first row.

```
# Generate motion primitive functions
I want to generate videos of a 2d agent moving by following a given path with specific shapes.
for instance, an agent can follow a linear line from point (0,0) to (1,1), while another agent
can follow a sine wave from (0,0.5) to (1,0.5). Set the range of x,y coordinates from 0 to 1.
Write down a python code to generate N different trajectories and save those as .mp4 videos
and also .jpg images. Also, save the information of trajectories in a .json file, including
the shape type of the path, starting point, and ending point.
```

```
# Generate a robot's motion moving from point A to point B in a straight line
def draw_line_position_mode(n, time_dt=1.5, start=(0,0.5), end=(0.5,1.0)):
    x = np.linspace(start[0], end[0], n, endpoint=True)
    y = np.linspace(start[1], end[1], n, endpoint=True)
    mat = np.c_[x, y]
    episode_info = {"type": "straight_line", "n": n, "time_dt": time_dt, "start": start,
    "end": end}
    return mat, episode_info

# Generate a robot's motion moving from point A to point B, while making horizontal
oscillations
def horizontal_shaking(n, time_dt=1.5, start=(0,0.5), amplitude=0.1, frequency=2,
roll_angle=0):
    start_point = start
    end_point = (start[0] + amplitude, start[1])
    x = np.linspace(start_point[0], end_point[0], n, endpoint=True)
    y = np.linspace(start_point[1], end_point[1], n, endpoint=True)
    mat = np.c_[x, y]
    horizontal_shaking_mat = []
    for i in range(frequency):
        if i % 2 == 0:
            horizontal_shaking_mat.append(mat)
        else:
            horizontal_shaking_mat.append(mat[::-1])
    # add wrist_roll for vibration
    wrist_roll = np.array([roll_angle, 0.0] * (n*frequency))
    episode_info = {"type": "horizontal_shaking", "n": n, "time_dt": time_dt, "start": start,
    "amplitude": amplitude, "frequency": frequency, "wrist_roll": wrist_roll}
    return np.concatenate(horizontal_shaking_mat), episode_info

# Generate a robot's motion moving from point A to point B, while making vertical oscillations
def vertical_shaking(n, time_dt=1.5, start=(0,0.5), amplitude=0.1, frequency=2, roll_angle=0):
    start_point = start
    end_point = (start[0], start[1] + amplitude)
    x = np.linspace(start_point[0], end_point[0], n, endpoint=True)
    y = np.linspace(start_point[1], end_point[1], n, endpoint=True)
    mat = np.c_[x, y]
    vertical_shaking_mat = []
    for i in range(frequency):
        if i % 2 == 0:
            vertical_shaking_mat.append(mat)
        else:
            vertical_shaking_mat.append(mat[::-1])
    # add wrist_roll for vibration
    wrist_roll = np.array([roll_angle, 0.0] * (n*frequency))
    episode_info = {"type": "vertical_shaking", "n": n, "time_dt": time_dt, "start": start,
    "amplitude": amplitude, "frequency": frequency, "wrist_roll": wrist_roll}
    return np.concatenate(vertical_shaking_mat), episode_info
```

App. Fig. 16: Example prompt and generated policies when ChatGPT is asked to generate policies as codes.