HUMANSKILL-4D: TRAINING-FREE LANGUAGE QUERYING OF ANIMATABLE GAUSSIAN AVATARS IN SPACE AND TIME

Anonymous authors
Paper under double-blind review



Figure 1: **Language queries on a Gaussian avatar. (a)** A 3D query selects a region (*Where*). **(b)** A 4D query also states when to select it (*When*). Please see the supplementary video for a demo.

ABSTRACT

We introduce avatar referring segmentation, which uses a language query to select a region of an animatable Gaussian avatar and, when the query specifies a pose or motion, the matching intervals in its driving pose sequence. Such queries can support local editing and motion inspection of digital humans. Adapting existing language Gaussian splatting methods to avatars poses two challenges: adjacent parts and garments are hard to separate and may be referred to indirectly (*region ambiguity*), while fine-grained motions can look alike in RGB (*motion ambiguity*). To address these challenges, we propose HumanSkill-4D, a training-free framework with a multimodal large language model (MLLM). For region ambiguity, it builds a semantic memory linking the avatar's parts and garments to their persistent Gaussian regions and appearance descriptions, from which the MLLM selects target categories, even when referred to by appearance, relation, or function. For motion ambiguity, it stores joint trajectories in a motion memory, and the MLLM composes geometric operators into a motion program that a deterministic executor runs on this memory to find matching intervals. On three human-annotated benchmarks, HumanSkill-4D outperforms prior methods in 3D and 4D referring segmentation and also in part segmentation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Gaussian avatars reconstruct a person's geometry and appearance and are animated by a driving pose sequence (Qian et al., 2024; Kocabas et al., 2024), supporting applications in AR/VR and sports (Hu et al., 2024; Jiang et al., 2025). In these applications, users may want to modify a particular body part or garment, as in garment transfer or appearance editing (*Where*) (Lin et al., 2024; Sunagad et al., 2025). They may also want to inspect that region during a particular pose or motion (*When*). Geometry and pose parameters alone, however, do not link a language query to the relevant Gaussians and intervals. This makes language Gaussian splatting a natural starting point: 3D methods select regions from language queries (Qin et al., 2024; Qiu et al., 2024), while 4D extensions localize the intervals described in language (Li et al., 2025; Fiebelman et al., 2025). We thus introduce avatar referring segmentation: using language to select an avatar region (*3D query*; Fig. 1(a)) and, when needed, the intervals of a specified pose or motion (*4D query*; Fig. 1(b)).

In 3D, language Gaussian methods attach image-language features to the Gaussians by learning a language field (Qin et al., 2024; He et al., 2025) or registering features directly (Kim et al., 2025), and select the regions whose features match a query. In 4D, 4-LEGS (Fiebelman et al., 2025) estimates temporal relevance from video-text features, while 4D LangSplat learns a temporal language field from MLLM-generated captions (Li et al., 2025). For humans, OpenHuman3D (Suzuki et al., 2025a) and OpenHuman4D (Suzuki et al., 2025b) perform open-vocabulary part segmentation.

Despite these advances, enabling language querying on Gaussian avatars presents two challenges. (1) *Region ambiguity.* Human parts and garments lie close together and often overlap, while pose and viewpoint change which boundaries are visible. Yet the class-agnostic SAM masks (Kirillov et al., 2023) used by feature-matching methods (Qin et al., 2024) do not necessarily align with these body-part and garment boundaries. Replacing them with human-parser masks improves LangSplat and Dr. Splat, but better boundaries alone are not enough. Beyond the masks, the generic CLIP (Radford et al., 2021) features that match queries to regions can still confuse body parts with garments (Suzuki et al., 2025a) and do not resolve which region is referred to through relation or function (Tab. 2). (2) *Motion ambiguity.* Even after identifying the target region, determining when a fine-grained pose or motion condition holds remains difficult. For example, selecting upper clothing in Fig. 1(b) depends on arm motion, so tracking the clothing over time (Suzuki et al., 2025b) alone cannot identify the requested intervals. Although RGB video can support temporal localization (Fiebelman et al., 2025; Li et al., 2025), visually similar motions can be difficult to distinguish (Yang et al., 2026), and fine-grained motion understanding remains challenging even for video MLLMs (Hong et al., 2025).

To address these challenges, we introduce HumanSkill-4D, a training-free framework that follows the agent skill approach (Tang et al., 2026) and equips a multimodal large language model (MLLM) with an avatar-specific semantic memory and executable geometric operators. For *region ambiguity*, the semantic memory links body parts and garments to the avatar's Gaussians. A 2D human parser (Khirodkar et al., 2026) labels pixels in multi-view images, and we associate each label with the Gaussians contributing to those pixels. Because animation preserves Gaussian identities across poses (Qian et al., 2024), labels from different views and poses accumulate into per-Gaussian category scores. The MLLM augments these categories with appearance descriptions and uses them to interpret queries by appearance, relation, or function. Selected categories determine the Gaussian region from these scores without learning a language field. For *motion ambiguity*, we store the driving pose sequence as joint trajectories in a motion memory. Operators adapted from PoseScript (Delmas et al., 2022) and MotionScript (Yazdian et al., 2025) measure body configuration and its changes. Given a 4D query, the MLLM reads the text cards of a registry shared across avatars and composes a motion program, which a deterministic executor runs on the motion memory to return matching intervals. Fine-grained timing is thus measured rather than perceived. This compositional design extends to new motion conditions by recombining existing operators or adding new ones.

We make four contributions: (1) HumanSkill-4D, a framework that uses persistent Gaussian identities and driving poses to enable 3D and 4D avatar referring segmentation without task-specific training; (2) an avatar-specific semantic memory that links body parts and garments to Gaussian regions and appearance descriptions, enabling direct and indirect region queries without learning a language field; (3) motion localization through MLLM-composed motion programs executed over joint trajectories, which extends to new motion conditions by recombining existing operators or adding new ones; and (4) evaluation on three human-annotated benchmarks, demonstrating higher

accuracy than the evaluated baselines in 3D and 4D referring segmentation and part segmentation, with compact storage and preparation within minutes after avatar reconstruction.

2 Related Work

3D language Gaussian splatting. Most language Gaussian methods select regions by matching a query to CLIP features that are learned per scene (Qin et al., 2024; Shi et al., 2024; Wu et al., 2024) or registered without per-scene optimization, directly or through an external index (Kim et al., 2025; Ding et al., 2026). Beyond object names, ReLaGS adds relation-guided retrieval in static scenes (Xie et al., 2026), while ReferSplat learns per-scene referring features from annotated expressions (He et al., 2025). Open-vocabulary part segmentation likewise matches text to language-aligned point features of 3D shapes (Ma et al., 2025). For humans, OpenHuman3D fine-tunes a CLIP image encoder on human parsing data (Suzuki et al., 2025a). Whereas these methods match queries to language features, HumanSkill-4D uses an MLLM to interpret various region queries through a semantic memory of avatar parts and garments, supporting both referring and part segmentation.

4D language Gaussian splatting. Recent work extends language Gaussian splatting to dynamic scenes, where a query specifies a region and when it is relevant. 4D LangSplat learns time-agnostic and time-sensitive language fields from MLLM-generated captions (Li et al., 2025), 4-LEGS grounds video-text features in dynamic Gaussians (Fiebelman et al., 2025), and ST4R-Splat studies referring segmentation on 4D Gaussians (Meng et al., 2026), with further work on temporal boundaries and object motion (Xu et al., 2026; Barhdadi et al., 2026). For humans, OpenHuman4D tracks parsed parts over time (Suzuki et al., 2025b), and 4D-DRESS develops a 4D human parsing pipeline (Wang et al., 2024d). However, these methods mainly rely on RGB evidence, in which different motions can look alike (Yang et al., 2026) and fine-grained motion remains hard to capture (Hong et al., 2025). Skeleton-based temporal grounding localizes described motions with learned motion-language features (Wang et al., 2025; Zhou et al., 2026). HumanSkill-4D measures motion conditions on the driving poses and returns the queried region with its intervals.

Agent skills and executable procedures. Large language models (LLMs) increasingly solve complex or fine-grained tasks through supplied procedures rather than task-specific training. Some compose predefined actions or handlers (Pan et al., 2025; Khot et al., 2023; Chen et al., 2026), whereas others generate code over visual or 3D tools (Gupta & Kembhavi, 2023; Surís et al., 2023; Liang et al., 2023; Mi et al., 2025; Luo et al., 2026). More recently, agent skills compile experience into reusable knowledge and procedures that accompany the query (Wang et al., 2024a; Tang et al., 2026), and this paradigm now extends to navigation, 3D spatial reasoning, and 4D spatio-temporal reasoning (Ma et al., 2026; Li et al., 2026a; Sohn et al., 2026). Building on this paradigm, HumanSkill-4D equips an MLLM with avatar-specific memories and executable geometric operators to interpret the semantics and motion of animatable Gaussian avatars, without task-specific training.

3 Preliminaries

Gaussian avatars and motion. 3DGS-Avatar (Qian et al., 2024) represents an avatar as a fixed set of N Gaussians g_i with position, shape, opacity, and color. These Gaussians are defined in a canonical space corresponding to the reference pose of SMPL (Loper et al., 2015), a body model parameterized by shape and joint rotations. A driving pose sequence of T frames specifies the joint rotations and root translation at each frame, which determine the 3D joint positions. Animation deforms the Gaussians with the body into each target pose, so g_i denotes the same Gaussian across frames.

Rendering. Rendering projects the Gaussians posed at frame t into camera view v and blends their colors at pixel u from front to back (Kerbl et al., 2023). Let $\alpha_i^{v,t}(u)$ be the projected opacity of Gaussian g_i at that pixel, and write $j \prec i$ when Gaussian g_j lies in front of g_i along the pixel's ray. The rendering contribution of g_i to pixel u, the coefficient of its color in the blended pixel, is

$$w_i^{v,t}(u) = \alpha_i^{v,t}(u) \prod_{j \prec i} \big(1 - \alpha_j^{v,t}(u)\big), \tag{1}$$

where the product is the transmittance through the Gaussians in front of g_i.

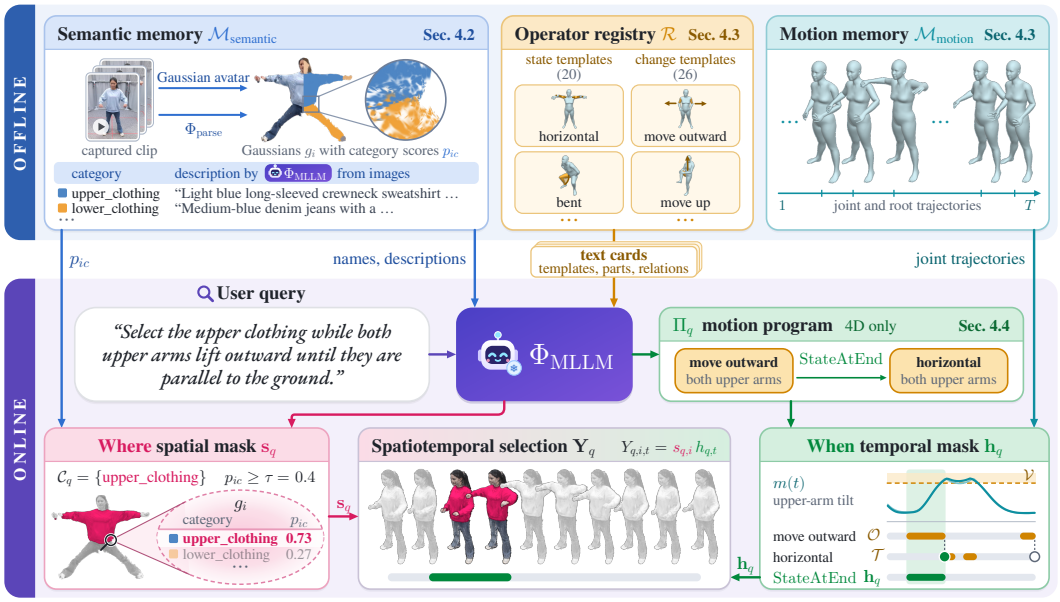


Figure 2: **HumanSkill-4D overview.** **(Offline)** We prepare semantic and motion memories and a shared operator registry. **(Online)** The frozen MLLM selects target categories $\mathcal{C}_q$ from names and descriptions and, for a 4D query, composes a motion program Π_q from registry text cards. In this example, it selects upper clothing, whose Gaussians with $p_{ic} \geq \tau$ form the spatial mask $\mathbf{s}_q$. Applying the motion templates to both upper arms gives four operators: two detect outward movement, and two detect horizontal states, where the upper-arm tilt $m(t)$ lies in the valid range $\mathcal{V}$ (orange band). The temporal relation *StateAtEnd* keeps the movement intervals that end horizontal, giving the temporal mask $\mathbf{h}_q$. The two masks select the garment during these intervals ($\mathbf{Y}_q$).

4 Method

4.1 Overview

Given a language query, HumanSkill-4D selects the described region of an avatar and, for a 4D query, every matching interval in its driving pose sequence. As shown in Fig. 2, the framework prepares reusable memories offline, then uses a frozen MLLM Φ_{MLLM} to interpret queries online.

In the offline phase, we build a semantic memory $\mathcal{M}_{\text{semantic}}$ that links body-part and garment categories to the avatar's Gaussians through category scores p_{ic}. The memory also stores category-level appearance descriptions generated by Φ_{MLLM} from multi-view images (Sec. 4.2). A motion memory $\mathcal{M}_{\text{motion}}$ stores joint and root trajectories for each driving pose sequence. A registry $\mathcal{R}$ shared across avatars provides reusable templates for checking pose and motion conditions (Sec. 4.3).

In the online phase, Φ_{MLLM} reads the query q and the category names and descriptions to select target categories $\mathcal{C}_q$. Their stored scores determine the spatial mask $\mathbf{s}_q$ (*Where*), which is the complete output for a 3D query. For a 4D query, the model also reads the registry's text cards and composes operators into a motion program Π_q. A deterministic executor runs this program on $\mathcal{M}_{\text{motion}}$ to obtain the temporal mask $\mathbf{h}_q$ (*When*). The two masks give the spatiotemporal selection $\mathbf{Y}_q$ (Sec. 4.4).

4.2 Building the Semantic Memory

We first reconstruct a Gaussian avatar with 3DGS-Avatar (Qian et al., 2024) and retain its canonical Gaussians g_i. From the captured images, we select a registration set $\mathcal{D}_{\text{reg}}$ of camera-frame pairs. For each $(v, t) \in \mathcal{D}_{\text{reg}}$, a frozen 2D human parser Φ_{parse} assigns each person pixel u a label $\ell^{v,t}(u) \in \mathcal{K}$, where $\mathcal{K}$ contains the non-background body-part and garment categories (details in Supp. Sec. A.1).

Because each Gaussian retains its identity across poses, we aggregate labels across views and frames using contribution-based Gaussian registration adapted from Dr. Splat (Kim et al., 2025). For each (v, t), we render the posed avatar through camera v and weight each pixel's label by the Gaussian's

rendering contribution $w_i^{v,t}(u)$ from Eq. 1. These weighted labels give the category score

$$p_{ic} = \frac{\sum_{(v,t)\in\mathcal{D}_{\text{reg}}} \sum_u w_i^{v,t}(u)\, \mathbf{1}[\ell^{v,t}(u) = c]}{\sum_{(v,t)\in\mathcal{D}_{\text{reg}}} \sum_u w_i^{v,t}(u)\, \mathbf{1}[\ell^{v,t}(u) \in \mathcal{K}]}. \tag{2}$$

Here, $\mathbf{1}[\cdot]$ denotes the indicator function. The score p_{ic} is the fraction of Gaussian g_i's rendering contribution to non-background person pixels assigned to category c. If the denominator is zero, we set $p_{ic} = 0$ for all c. We retain all category scores and associate g_i with every category whose score meets a shared activation threshold τ. A category is queryable if at least one Gaussian meets τ.

To support appearance-based references such as *"the denim garment"* in Fig. 1(a), we augment category names with avatar-specific descriptions. The frozen Φ_{MLLM} receives the category names and four camera views taken at the frame closest to a T-pose. It summarizes visible color, texture, graphics, and garment coverage into one description per category (instruction in Supp. Tab. I). Together, the category names, descriptions, and Gaussian scores form the reusable per-avatar semantic memory $\mathcal{M}_{\text{semantic}}$ used for online region selection.

4.3 Building the Motion Memory and Operator Registry

Motion memory. The motion memory $\mathcal{M}_{\text{motion}}$ stores SMPL joint positions, root orientation, and root translation for all T frames of the driving pose sequence. For directional measurements, we adapt MotionScript's first-frame normalization (Yazdian et al., 2025) to define a coordinate frame fixed throughout the sequence. The initial root position defines its origin, gravity defines its vertical axis, and the person's initial facing direction defines its forward axis.

Operator registry. The operator registry $\mathcal{R}$, shared across avatars, stores templates for checking pose and motion conditions. A template is a reusable rule that specifies a geometric measurement and a condition to check, without selecting a particular body part. For example, the *horizontal* template in Fig. 2 measures a segment's tilt relative to gravity and checks whether it lies in the range for a horizontal pose. Each template also specifies its coordinate frame and compatible body parts. State templates check conditions at individual frames, whereas change templates check how measurements change over intervals. We adapt these measurements and conditions from PoseScript and MotionScript (Delmas et al., 2022; Yazdian et al., 2025), defining 46 templates—20 state and 26 change templates—in 7 measurement types (Supp. Sec. A.2). The frozen Φ_{MLLM} reads text cards describing each template's measurement, use, and compatible body parts (Supp. Tab. C).

From templates to operators. Applying a template to a compatible body part defines an executable operator ρ. The selected body part determines which SMPL joint trajectories the operator reads from $\mathcal{M}_{\text{motion}}$. For example, applying the *horizontal* template to the left upper arm selects the left shoulder–elbow segment. In Fig. 2, applying the *move outward* and *horizontal* templates to both upper arms gives two outward-motion operators ρ_1, ρ_2 and two horizontal-state operators ρ_3, ρ_4. Let $m_\rho(t)$ denote the measurement of operator ρ at frame t. A state operator selects frames where $m_\rho(t)$ lies in the template's valid range $\mathcal{V}_\rho$. A change operator first identifies candidate intervals $\mathcal{O}_\rho^{\text{cand}}$ over which the measurement changes in the specified direction, then retains those meeting minimum total-change and duration thresholds. The resulting state frames $\mathcal{T}_\rho$ and change occurrences $\mathcal{O}_\rho$ are

$$\begin{aligned} \mathcal{T}_\rho &= \{t :\ m_\rho(t) \in \mathcal{V}_\rho\}, \\ \mathcal{O}_\rho &= \left\{[a,b] \in \mathcal{O}_\rho^{\text{cand}} :\ |m_\rho(b) - m_\rho(a)| \geq \Delta_\rho^{\min},\ b - a + 1 \geq L_{\min}\right\}. \end{aligned} \tag{3}$$

For change operators, $\Delta_\rho^{\min}$ specifies the minimum total change for the measurement type, and $L_{\min}$ specifies the minimum interval length in frames, including both endpoints (details in Supp. Sec. A.3). By separating reusable templates from their body-part-specific operators, this design supports new pose and motion conditions by recombining existing operators or adding templates.

4.4 Answering 3D and 4D Queries

Query interpretation. For both 3D and 4D queries, we provide Φ_{MLLM} with a text-only prompt containing the query q and the category names and descriptions from the semantic memory $\mathcal{M}_{\text{semantic}}$, following skill-based prompting (Tang et al., 2026). For a 4D query, the prompt also includes text cards describing templates in the registry $\mathcal{R}$, their compatible body parts, and temporal

relations that specify how operator outputs are combined over time. Both query types use a few examples for in-context learning (ICL) (Brown et al., 2020) (details in Supp. Sec. A.5). The model returns target categories $\mathcal{C}_q$ and, for a 4D query, a motion program Π_q that combines operators—specified by template–body-part pairs—through temporal relations.

Spatial selection. For both query types, the spatial mask $\mathbf{s}_q \in \{0,1\}^N$ selects canonical Gaussian g_i if its stored category score p_{ic} meets the shared threshold τ for any target category $c \in \mathcal{C}_q$:

$$s_{q,i} = \mathbf{1}[\exists\, c \in \mathcal{C}_q :\; p_{ic} \geq \tau]\,. \tag{4}$$

For a 3D query, this Gaussian mask is the complete output of the system.

Temporal execution. For a 4D query, the executor runs the motion program Π_q on the motion memory $\mathcal{M}_{\text{motion}}$. Each operator ρ returns state frames $\mathcal{T}_\rho$ or change occurrences $\mathcal{O}_\rho$ from Eq. 3. The executor combines these outputs according to the temporal relations in Π_q to obtain the matching interval set $\mathcal{I}_q$ (details in Supp. Sec. A.4). For example, the program first identifies intervals when both upper arms move outward. A temporal relation then retains each movement interval whose final frame has both arms horizontal (*StateAtEnd* in Fig. 2).

We convert the matching intervals in $\mathcal{I}_q$ into a temporal mask $\mathbf{h}_q \in \{0,1\}^T$, marking each frame covered by at least one interval:

$$h_{q,t} = \mathbf{1}[\exists\, I \in \mathcal{I}_q :\; t \in I]\,. \tag{5}$$

Combining the temporal mask $\mathbf{h}_q$ with the spatial mask $\mathbf{s}_q$ gives the spatiotemporal selection $\mathbf{Y}_q$:

$$\mathbf{Y}_q = [Y_{q,i,t}]_{i,t} \in \{0,1\}^{N\times T}, \qquad Y_{q,i,t} = s_{q,i}\, h_{q,t}. \tag{6}$$

Gaussian g_i is selected at frame t when both masks are active. Because $\mathbf{s}_q$ does not depend on t, $\mathbf{Y}_q$ selects the same region in every retained interval and none elsewhere. Selected categories explain which region is selected, while operator measurements and temporal relations explain when.

5 Experiments

5.1 Evaluation Protocol

Benchmarks. To evaluate avatar referring segmentation, we introduce three benchmarks by augmenting existing human datasets with referring queries and human-annotated targets, following prior evaluations (He et al., 2025; Li et al., 2025). ZJU-Refer-3D extends ZJU-MoCap (Peng et al., 2021) with dense part annotations and referring queries covering part names, appearance, wearing relations, and functions. HuMMan-Refer-4D extends HuMMan (Cai et al., 2022) by converting its fine-grained body-part motion descriptions (Zhang et al., 2023) into referring queries, while ZJU-Refer-4D extends ZJU-MoCap with queries in the same format. Benchmark details in Supp. Secs. B.1–B.2.

Metrics. For part segmentation, we report mean per-category IoU (mIoU), overall pixel accuracy (OA), and mean per-category recall (mAcc) (Suzuki et al., 2025a). For 3D referring segmentation, we report mIoU over queries (He et al., 2025). For 4D referring segmentation, vIoU evaluates region accuracy and timing jointly, penalizing both incorrect masks and extra or missed active frames (Li et al., 2025). We additionally report tIoU, precision, and recall to assess temporal selection separately (Fiebelman et al., 2025). tIoU measures overlap between predicted and annotated active frames. Precision is the fraction of selected frames annotated as active, and recall the fraction of annotated active frames selected (Supp. Sec. C.1).

Implementation details. All comparisons use the same reconstructed Gaussian avatars. We use Sapiens2-1B (Khirodkar et al., 2026) as the human parser Φ_{parse} and Qwen3.8-27B (Qwen Team, 2026b) as the MLLM Φ_{MLLM}, with both models frozen. We set the Gaussian activation threshold to $\tau = 0.4$ and the minimum movement interval length to $L_{\min} = 4$ frames. Minimum-change thresholds $\Delta_\rho^{\min}$ are fixed per measurement type and shared across both datasets (Supp. Sec. A.3). We render selected Gaussians as 1 and all others as 0 in held-out views, keeping geometry, opacity, and occlusion unchanged, and binarize the rendering at 0.7 following OpenGaussian (Wu et al., 2024) before comparing it with human annotations. All runs of HumanSkill-4D use an NVIDIA A100 80GB GPU. Further implementation and comparison details appear in Supp. Sec. C.2.

Table 1: **Part segmentation on ZJU-Refer-3D.** OA, mAcc, and mIoU (%) over 16 categories.

Method	Venue	mIoU per subject									Overall		
		313	315	377	386	387	390	392	393	394	OA	mAcc	mIoU
4D-DRESS	CVPR'24	35.9	33.9	34.7	24.7	27.5	20.0	37.5	24.9	24.9	78.6	35.7	31.2
OpenHuman3D	3DV'25	47.7	44.8	32.3	35.6	30.9	25.9	44.3	45.9	26.9	66.6	51.8	41.2
OpenHuman4D	BMVC'25	48.0	37.8	40.3	31.8	27.9	26.0	43.6	41.9	29.6	77.6	52.8	40.0
Find3D	ICCV'25	17.0	13.3	13.1	13.4	8.8	8.1	13.5	12.1	10.7	14.9	31.5	12.7
HumanSkill-4D		**73.4**	**77.6**	**74.5**	**65.6**	**52.1**	**50.4**	**65.9**	**67.8**	**48.0**	**94.2**	**79.5**	**75.0**

5.2 Results on the 3D and 4D Evaluations

Part segmentation. Tab. 1 compares our method with existing human parsing methods, including 4D-DRESS and OpenHuman3D/4D (Wang et al., 2024d; Suzuki et al., 2025a;b), as well as the general-purpose 3D part segmentation model Find3D (Ma et al., 2025). Our method achieves the highest mIoU on all nine avatars. Find3D's lower performance suggests that general-purpose part representations do not readily transfer to the fine-grained body-part and garment distinctions evaluated here. These results show that, although designed for referring segmentation, our semantic memory enables more accurate part segmentation than the evaluated baselines by matching category names to their registered Gaussian regions. See Supp. Fig. B for qualitative results.

Table 2: **3D referring segmentation on ZJU-Refer-3D.** mIoU (%) on part, appearance (App.), and relation/function (Rel.) queries; † uses our parser Φ_{parse} (i.e., Sapiens2).

Method	Part	App.	Rel.	All
LangSplat	57.0	68.0	4.1	56.2
LangSplat†	71.3	72.2	4.8	61.9
ReferSplat	28.6	47.6	5.1	37.4
ReferSplat†	52.1	35.4	12.7	35.3
OpenHuman3D	40.3	54.9	3.1	44.2
Dr. Splat	8.5	16.2	9.1	13.6
Dr. Splat†	54.5	44.0	2.5	39.9
Find3D	26.9	40.6	6.4	32.7
HumanSkill-4D	**78.4**	**89.9**	**55.1**	**82.4**

Table 3: **4D referring segmentation on ZJU-Refer-4D and HuMMan-Refer-4D** (%), averaged over all queries.

Method	vIoU	tIoU	Prec	Rec
4-LEGS	1.9	13.6	17.6	35.7
4D LangSplat	4.9	17.7	22.6	45.3
OpenHuman4D	5.4	22.1	22.1	**100.0**
HumanSkill-4D	**45.0**	**62.2**	**73.4**	86.0

3D referring segmentation. Tab. 2 evaluates queries based on part names (Part), appearance (App.), and relations or functions (Rel.). HumanSkill-4D achieves the highest mIoU across all three groups, with the largest margin on Rel. queries. The comparisons highlight both aspects of region ambiguity. Supplying the same Sapiens2 mask proposals to LangSplat and Dr. Splat (Qin et al., 2024; Kim et al., 2025) improves their Part and App. scores, but their CLIP-based matching still struggles to identify indirectly described targets. OpenHuman3D (Suzuki et al., 2025a), despite using a CLIP encoder adapted to human parsing, also struggles with Rel. queries, as does the general-purpose Find3D (Ma et al., 2025). ReferSplat (He et al., 2025) learns expression-conditioned Gaussian features through mask supervision. Its Sapiens2-supervised variant performs best among the baselines on Rel. queries, yet still substantially underperforms our method. These results suggest that better masks and language-aligned features are insufficient to reliably resolve indirect references among closely packed human parts and garments. Fig. 3 also illustrates this distinction: for "the garment worn below the shirt," LangSplat and OpenHuman3D select the reference shirt, whereas our method selects the requested lower garment. By linking category names and descriptions to Gaussian regions, HumanSkill-4D enables an MLLM to resolve indirect references and select the intended regions without task-specific training. Further qualitative results appear in Supp. Sec. D.2.

4D referring segmentation. Tab. 3 reports joint region and interval selection results averaged over queries from ZJU-Refer-4D and HuMMan-Refer-4D. 4-LEGS (Fiebelman et al., 2025) estimates temporal relevance from video features, while 4D LangSplat (Li et al., 2025) learns a temporal language field from MLLM-generated caption embeddings and selects regions and intervals by matching queries to its learned language fields. Despite incorporating temporal information, both methods struggle to determine when the specified poses and motions occur. In contrast, the human parsing and tracking method OpenHuman4D (Suzuki et al., 2025b) keeps region predictions active throughout the clip, achieving 100% temporal recall but lowering tIoU and joint vIoU by including inac-

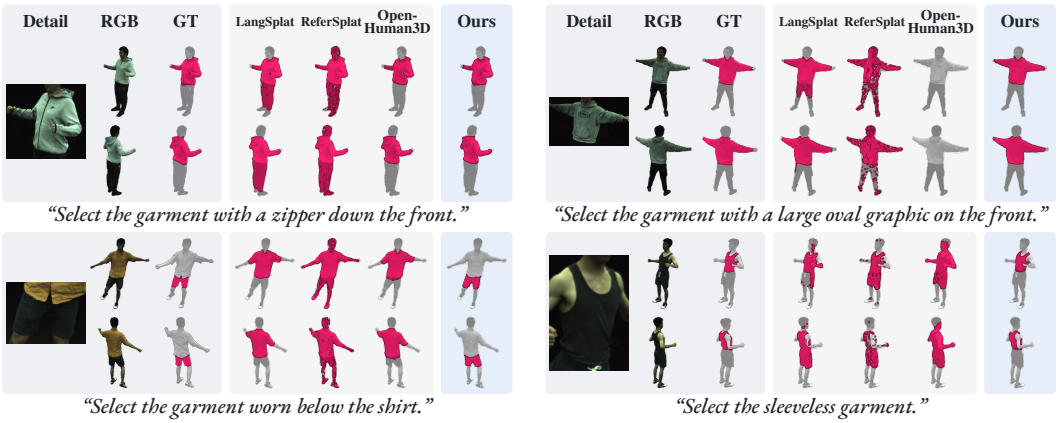


Figure 3: **Qualitative 3D referring segmentation on ZJU-Refer-3D.** Two held-out views and a detail crop per query; selected regions are magenta.

Table 4: **Ablations** on MLLM choice (**left**), semantic category selector (**middle**), and activation threshold τ (**right**). We report 3D mIoU and 4D vIoU (%).

MLLM	mIoU	vIoU
Qwen3.6-27B	**82.4**	44.3
Gemma 4-31B	55.3	**45.0**
Mistral S3.2-24B	58.6	38.8
Qwen3.8-27B	**82.4**	**45.0**

Selector	mIoU	vIoU
$\text{CLIP}_{\text{text}}$	34.9	13.1
E5	41.4	13.3
JINA-v5	63.1	19.4
Qwen3.8-27B	**82.4**	**45.0**

τ	mIoU	vIoU
0.1	80.7	43.2
0.2	82.2	44.6
0.3	**82.6**	44.9
0.4	82.4	**45.0**
0.5	81.9	44.8

tive frames. Fig. 4 also shows spatial errors: 4D LangSplat and OpenHuman4D select unrelated upper-body regions for a lower-clothing query. HumanSkill-4D avoids both errors by resolving target categories through semantic memory and executing MLLM-composed motion programs on the driving poses. This grounds timing in explicit pose and motion measurements, reducing reliance on ambiguous RGB cues and embedding-based matching. The resulting intervals achieve the highest tIoU and temporal precision, and combining them with the selected regions yields the highest vIoU. Further qualitative results appear in Supp. Sec. D.2.

5.3 Ablations and Computational Analysis

Choice of MLLM. We compare choices of Φ_{MLLM} within our framework using the same image inputs and benchmark queries. In Tab. 4 (left), Qwen3.8 ties for the highest score on both tasks, so we use it by default. Qwen3.6 achieves similar performance on both tasks, while Gemma 4 matches its 4D vIoU despite lower 3D mIoU. The variation in 3D mIoU suggests differences in how the models interpret referring queries. Results with smaller MLLMs appear in Supp. Sec. D.1.

Semantic category selector. Tab. 4 (middle) replaces MLLM-based category selection with text-embedding retrieval. CLIP (Cherti et al., 2023), E5 (Wang et al., 2024b), and JINA-v5 (Akram et al., 2026) rank the same category descriptions and select the top three, with temporal predictions held fixed for 4D queries. All three yield lower 3D mIoU and 4D vIoU, showing that the MLLM selects target categories more effectively from the same semantic memory. Supp. Sec. D.2 illustrates the remaining selection errors: JINA correctly identifies the patched garment but includes a forearm when asked to select both hands. This highlights the importance of excluding unrequested categories, even when retrieval finds the target.

Gaussian activation threshold. Tab. 4 (right) varies τ with the memory, selected categories, and intervals fixed. At $\tau = 0.1$, weakly supported Gaussians can leak across part or garment boundaries, consistent with the small drops in mIoU and vIoU. Performance remains stable for $\tau = 0.2$–0.5, including our default 0.4.

Computational costs. Tab. 5 compares preparation through the first query's masks and retained storage for a reconstructed avatar (Supp. Sec. C.3). LangSplat, ReferSplat, 4-LEGS, and 4D LangSplat optimize features when preparing a new avatar for querying. OpenHuman3D/4D retain segmenta-

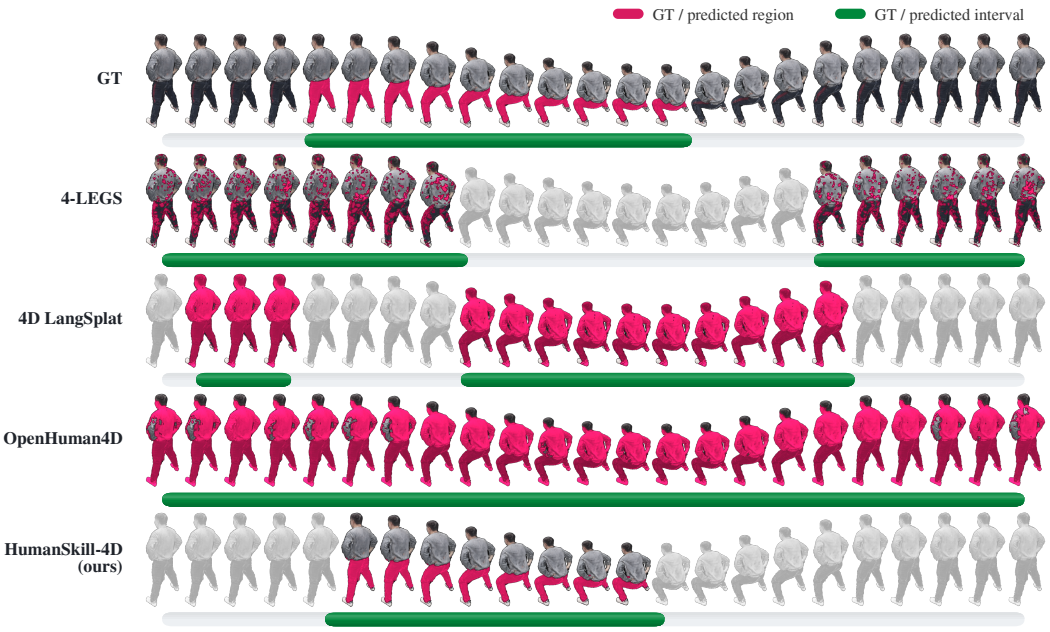


Figure 4: **Qualitative 4D referring segmentation on HuMMan-Refer-4D.** GT and predicted regions are magenta and intervals green; faded frames are unselected.

Table 5: **Computational costs** for one new avatar in 3D (left) and 4D (right). Train.: per-avatar training; Time: minutes from a completed avatar to the first query's masks; Stor.: per-avatar data retained for querying (MB); Score: 3D mIoU or 4D vIoU on the full benchmark (%).

Method	Train.	Time	Stor.	Score
LangSplat	✓	75.1	∼2.4	56.2
ReferSplat	✓	127.8	∼4.4	37.4
OpenHuman3D	×	62.0	∼260	44.2
HumanSkill-4D	×	3.9	∼5	**82.4**

Method	Train.	Time	Stor.	Score
4-LEGS	✓	368.9	∼2,900	1.9
4D LangSplat	✓	208.4	∼94	4.9
OpenHuman4D	×	32.3	∼1,900	5.4
HumanSkill-4D	×	8.0	∼10	**45.0**

tion masks and region embeddings. HumanSkill-4D builds compact memories from Gaussians and driving poses using frozen models, without per-avatar optimization. In the measured cases, preparation through the first query takes 3.9 min in 3D and 8.0 min in 4D, the shortest times. Our memories occupy about 5 MB in 3D, comparable to LangSplat and ReferSplat, and 10 MB in 4D, below all evaluated 4D baselines. Notably, 4-LEGS and 4D LangSplat retain substantially more storage even after feature compression with autoencoders. Exploiting persistent regions and driving poses thus keeps preparation and storage costs low without per-avatar training or learned compression.

6 Conclusion

We introduced avatar referring segmentation and presented HumanSkill-4D, a training-free framework for selecting human regions and the intervals in which queried poses or motions occur. Its semantic memory links body-part and garment categories and appearance descriptions to persistent Gaussian regions, enabling direct and indirect region references to be resolved. For motion localization, the MLLM composes geometric operators adapted from PoseScript and MotionScript into programs that execute on the driving joint trajectories. To evaluate avatar referring segmentation, we introduced three human-annotated benchmarks covering region selection and pose- or motion-conditioned temporal selection. On these benchmarks, HumanSkill-4D outperforms the evaluated baselines in part segmentation and 3D and 4D referring segmentation. By exploiting the avatar's persistent regions and driving poses, the framework achieves these results with compact reusable memories and no additional task-specific training. The same framework supports new motion conditions by composing existing operators or extending the registry. Limitations and future directions are discussed in Supp. Sec. E.

AI USE STATEMENT

We used AI assistants in a supporting role. They were used to polish the wording and grammar of the manuscript, to help build figures and tables from our results, and to assist with code implementation, debugging, and running our evaluation scripts. The MLLM Φ_{MLLM} in HumanSkill-4D is a component of the proposed method (Sec. 4), not an assistant used to prepare this paper.

REPRODUCIBILITY STATEMENT

Every model that HumanSkill-4D calls is a frozen public release identified by its exact checkpoint in Supp. Sec. C.2, including the parser Φ_{parse}, the MLLM Φ_{MLLM}, and the models of each ablation. Supp. Sec. G quotes the core of each MLLM instruction with an in-context example, and Supp. Sec. A defines the semantic categories, the 46 operator templates, their thresholds, and the temporal relations. Sec. 5.1 and Supp. Sec. C.2 give the implementation settings. For evaluation, Sec. 5.1 gives the benchmarks and metrics, Supp. Secs. B–C detail the benchmark construction, the scoring rule of each metric, and the baseline protocols, and Supp. Sec. F lists every query.

REFERENCES

Mohammad Kalim Akram, Saba Sturua, Nastia Havriushenko, Quentin Herreros, Michael Günther, Maximilian Werk, and Han Xiao. jina-embeddings-v5-text: Compact and robust text embedding models using task-targeted distillation. In *SIGIR*, 2026.

Mohamed Rayan Barhdadi, Samir Abdaljalil, Rasul Khanbayov, Erchin Serpedin, and Hasan Kurban. 4D Synchronized Fields: Motion-Language Gaussian Splatting for Temporal Scene Understanding. *arXiv preprint arXiv:2603.14301*, 2026.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.

Zhongang Cai, Daxuan Ren, Ailing Zeng, Zhengyu Lin, Tao Yu, Wenjia Wang, Xiangyu Fan, Yang Gao, Yifan Yu, Liang Pan, Fangzhou Hong, Mingyuan Zhang, Chen Change Loy, Lei Yang, and Ziwei Liu. HuMMan: Multi-Modal 4D Human Dataset for Versatile Sensing and Modeling. In *ECCV*, 2022.

Zeren Chen, Xiaoya Lu, Zhijie Zheng, Pengrui Li, Lehan He, Yijin Zhou, Jing Shao, Bohan Zhuang, and Lu Sheng. Geometrically-constrained agent for spatial reasoning. In *CVPR*, 2026.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *CVPR*, 2023.

Ginger Delmas, Philippe Weinzaepfel, Thomas Lucas, Francesc Moreno-Noguer, and Grégory Rogez. PoseScript: 3D Human Poses from Natural Language. In *ECCV*, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

Jiayu Ding, Xinpeng Liu, Zhiyi Pan, Shiqiang Long, and Ge Li. ExtrinSplat: Decoupling Geometry and Semantics for Open-Vocabulary Understanding in 3D Gaussian Splatting. In *CVPR*, 2026.

Gal Fiebelman, Tamir Cohen, Ayellet Morgenstern, Peter Hedman, and Hadar Averbuch-Elor. 4-LEGS: 4D Language Embedded Gaussian Splatting. *CGF*, 2025.

Gemma Team. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *CVPR*, 2023.

Shuting He, Guangquan Jie, Changshuo Wang, Yun Zhou, Shuming Hu, Guanbin Li, and Henghui Ding. ReferSplat: Referring segmentation in 3D Gaussian splatting. In *ICML*, 2025.

Wenyi Hong, Yean Cheng, Zhuoyi Yang, Weihan Wang, Lefan Wang, Xiaotao Gu, Shiyu Huang, Yuxiao Dong, and Jie Tang. MotionBench: Benchmarking and improving fine-grained video motion understanding for vision language models. In *CVPR*, 2025.

Shoukang Hu, Tao Hu, and Ziwei Liu. GauHuman: Articulated Gaussian Splatting from Monocular Human Videos. In *CVPR*, 2024.

Zeren Jiang, Shaofei Wang, and Siyu Tang. DNF-Avatar: Distilling neural fields for real-time animatable avatar relighting. In *ICCV Workshops*, 2025.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM TOG*, 2023.

Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. LERF: Language Embedded Radiance Fields. In *ICCV*, 2023.

Rawal Khirodkar, He Wen, Julieta Martinez, Yuan Dong, Zhaoen Su, and Shunsuke Saito. Sapiens2. In *ICLR*, 2026.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. In *ICLR*, 2023.

Jun-Seong Kim, GeonU Kim, Yu-Ji Kim, Yu-Chiang Frank Wang, Jaesung Choe, and Tae-Hyun Oh. Dr. Splat: Directly Referring 3D Gaussian Splatting via Direct Language Embedding Registration. In *CVPR*, 2025.

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 Girshick. Segment anything. In *ICCV*, 2023.

Muhammed Kocabas, Jen-Hao Rick Chang, James Gabriel, Oncel Tuzel, and Anurag Ranjan. HUGS: Human Gaussian Splats. In *CVPR*, 2024.

Haoyuan Li, Zhengdong Hu, Jun Wang, Hehe Fan, and Yi Yang. Skill-3D: Evolving scene-aware skills for agentic 3D spatial reasoning. *arXiv preprint arXiv:2606.07436*, 2026a.

Wanhua Li, Renping Zhou, Jiawei Zhou, Yingwei Song, Johannes Herter, Minghan Qin, Gao Huang, and Hanspeter Pfister. 4D LangSplat: 4D Language Gaussian Splatting via Multimodal Large Language Models. In *CVPR*, 2025.

Yue Li, Qi Ma, Runyi Yang, Mengjiao Ma, Bin Ren, Nikola Popovic, Nicu Sebe, Theo Gevers, Luc Van Gool, Danda Pani Paudel, and Martin R. Oswald. Chorus: Multi-Teacher Pretraining for Holistic 3D Gaussian Scene Encoding. In *CVPR*, 2026b.

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.

Siyou Lin, Zhe Li, Zhaoqi Su, Zerong Zheng, Hongwen Zhang, and Yebin Liu. LayGA: Layered Gaussian Avatars for Animatable Clothing Transfer. In *SIGGRAPH*, 2024.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, Alexandre Sablayrolles, Amélie Héliou, Amos You, Andy Ehrenberg, Andy Lo, Anton Eliseev, Antonia Calvi, Avinash Sooriyarachchi, Baptiste Bout, Baptiste Rozière, Baudouin De Monicault, Clémence Lanfranchi, Corentin Barreau, Cyprien Courtot, Daniele Grattarola, Darius Dabert, Diego de las Casas, Elliot Chane-Sane, Faruk Ahmed, Gabrielle Berrada, Gaëtan Ecrepont, Gauthier Guinet, Georgii Novikov, Guillaume Kunsch, Guillaume Lample, Guillaume Martin, Gunshi Gupta, Jan Ludziejewski, Jason Rute, Joachim Studnia, Jonas Amar, Joséphine Delas, Josselin Somerville Roberts, Karmesh Yadav, Khyathi Chandu, Kush Jain, Laurence Aitchison, Laurent Fainsin, Léonard Blier, Lingxiao Zhao, Louis Martin, Lucile Saulnier, Luyu Gao, Maarten Buyl, Margaret Jennings, Marie Pellat, Mark Prins, Mathieu Poirée, Mathilde Guillaumin, Matthieu Dinot, Matthieu Futeral, Maxime Darrin, Maximilian Augustin, Mia Chiquier,

Michel Schimpf, Nathan Grinsztajn, Neha Gupta, Nikhil Raghuraman, Olivier Bousquet, Olivier Duchenne, Patricia Wang, Patrick von Platen, Paul Jacob, Paul Wambergue, Paula Kurylowicz, Pavankumar Reddy Muddireddy, Philomène Chagniot, Pierre Stock, Pravesh Agrawal, Quentin Torroba, Romain Sauvestre, Roman Soletskyi, Rupert Menneer, Sagar Vaze, Samuel Barry, Sanchit Gandhi, Siddhant Waghjale, Siddharth Gandhi, Soham Ghosh, Srijan Mishra, Sumukh Aithal, Szymon Antoniak, Teven Le Scao, Théo Cachet, Theo Simon Sorg, Thibaut Lavril, Thiziri Nait Saada, Thomas Chabal, Thomas Foubert, Thomas Robert, Thomas Wang, Tim Lawson, Tom Bewley, Tom Edwards, Umar Jamil, Umberto Tomasini, Valeriia Nemychnikova, Van Phung, Vincent Maladière, Virgile Richard, Wassim Bouaziz, Wen-Ding Li, William Marshall, Xinghui Li, Xinyu Yang, Yassine El Ouahidi, Yihan Wang, Yunhao Tang, and Zaccharie Ramzi. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A Skinned Multi-Person Linear Model. *ACM TOG*, 2015.

Zhanpeng Luo, Ce Zhang, Silong Yong, Cunxi Dai, Qianwei Wang, Haoxi Ran, Guanya Shi, Katia Sycara, and Yaqi Xie. pySpatial: Generating 3D Visual Programs for Zero-Shot Spatial Reasoning. In *ICLR*, 2026.

Tianyi Ma, Yue Zhang, Zehao Wang, and Parisa Kordjamshidi. Breaking down and building up: Mixture of skill-based vision-and-language navigation agents. In *ACL*, 2026.

Ziqi Ma, Yisong Yue, and Georgia Gkioxari. Find any part in 3D. In *ICCV*, 2025.

Yuming Meng, Dong Wu, and Hongbin Zha. ST4R-Splat: Spatio-Temporal Referring Segmentation in 4D Gaussian Splatting. In *CVPR*, 2026.

Boyu Mi, Hanqing Wang, Tai Wang, Yilun Chen, and Jiangmiao Pang. Language-to-space programming for training-free 3D visual grounding. In *EMNLP*, 2025.

Zhenyu Pan, Haozheng Luo, Manling Li, and Han Liu. Chain-of-action: Faithful and multimodal question answering through large language models. In *ICLR*, 2025.

Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *CVPR*, 2021.

Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016.

Zhiyin Qian, Shaofei Wang, Marko Mihajlovic, Andreas Geiger, and Siyu Tang. 3DGS-Avatar: Animatable Avatars via Deformable 3D Gaussian Splatting. In *CVPR*, 2024.

Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. LangSplat: 3D Language Gaussian Splatting. In *CVPR*, 2024.

Ri-Zhao Qiu, Ge Yang, Weijia Zeng, and Xiaolong Wang. Language-driven physics-based scene synthesis and editing via Feature Splatting. In *ECCV*, 2024.

Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27B dense model. *Qwen Blog*, 2026a.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork. *Qwen Blog*, 2026b.

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. In *ICML*, 2021.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *ICLR*, 2025.

Jin-Chuan Shi, Miao Wang, Hao-Bin Duan, and Shao-Hua Guan. Language embedded 3D Gaussians for open-vocabulary scene understanding. In *CVPR*, 2024.

Tin Stribor Sohn, Maximilian Dillitzer, Jason J. Corso, and Eric Sax. R4: Retrieval-Augmented Reasoning for Vision-Language Models in 4D Spatio-Temporal Space. In *CVPR*, 2026.

Zeyi Sun, Ye Fang, Tong Wu, Pan Zhang, Yuhang Zang, Shu Kong, Yuanjun Xiong, Dahua Lin, and Jiaqi Wang. Alpha-CLIP: A CLIP model focusing on wherever you want. In *CVPR*, 2024.

Basavaraj Sunagad, Heming Zhu, Mohit Mendiratta, Adam Kortylewski, Christian Theobalt, and Marc Habermann. TEDRA: Text-Based Editing of Dynamic and Photoreal Actors. In *3DV*, 2025.

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual Inference via Python Execution for Reasoning. In *ICCV*, 2023.

Keito Suzuki, Bang Du, Girish Krishnan, Kunyao Chen, Runfa Blark Li, and Truong Nguyen. Open-Vocabulary Semantic Part Segmentation of 3D Human. In *3DV*, 2025a.

Keito Suzuki, Bang Du, Runfa Blark Li, Kunyao Chen, Lei Wang, Peng Liu, Ning Bi, and Truong Nguyen. OpenHuman4D: Open-Vocabulary 4D Human Parsing. In *BMVC*, 2025b.

Liyan Tang, Cyrus Rashtchian, Chun-Sung Ferng, Andrew Tomkins, Da-Cheng Juan, and Tu Vu. WikiSkill: Compiling agent experience into persistent knowledge for skill evolution. *arXiv preprint arXiv:2608.27454*, 2026.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *TMLR*, 2024a.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. In *ACL*, 2024b.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024c.

Wenbo Wang, Hsuan-I Ho, Chen Guo, Boxiang Rong, Artur Grigorev, Jie Song, Juan Jose Zarate, and Otmar Hilliges. 4D-DRESS: A 4D dataset of real-world human clothing with semantic annotations. In *CVPR*, 2024d.

Xinghan Wang, Zixi Kang, and Yadong Mu. Text-controlled motion mamba: Text-instructed temporal grounding of human motion. *IEEE TIP*, 2025.

Yi Wang, Yinan He, Yizhuo Li, Kunchang Li, Jiashuo Yu, Xin Ma, Xinhao Li, Guo Chen, Xinyuan Chen, Yaohui Wang, Ping Luo, Ziwei Liu, Yali Wang, Limin Wang, and Yu Qiao. InternVid: A Large-Scale Video-Text Dataset for Multimodal Understanding and Generation. In *ICLR*, 2024e.

Bizhu Wu, Jinheng Xie, Meidan Ding, Zhe Kong, Jianfeng Ren, Ruibin Bai, Rong Qu, and Linlin Shen. FineMotion: A Dataset and Benchmark with Both Spatial and Temporal Annotation for Fine-Grained Motion Generation and Editing. In *ICCV*, 2025.

Yanmin Wu, Jiarui Meng, Haijie Li, Chenming Wu, Yahao Shi, Xinhua Cheng, Chen Zhao, Haocheng Feng, Errui Ding, Jingdong Wang, and Jian Zhang. OpenGaussian: Towards Point-Level 3D Gaussian-based Open Vocabulary Understanding. In *NeurIPS*, 2024.

Yaxu Xie, Abdalla Arafa, Alireza Javanmardi, Christen Millerdurai, Jia Cheng Hu, Shaoxiang Wang, Alain Pagani, and Didier Stricker. ReLaGS: Relational Language Gaussian Splatting. In *CVPR*, 2026.

Yichao Xu, Qiaowei Miao, Jinsheng Quan, Wei Yang, Zhihui Li, and Yawei Luo. LangField4D: Learning Identity-Adaptive and Spatio-Temporal Continuous 4D Language Fields for Dynamic Scenes. In *CVPR*, 2026.

Di Yang, Mahmoud Ali, Xuanlong Yu, Xi Shen, Quan Kong, Gianpiero Francesca, and François Brémond. MoVie: Broaden Your Views with Human Motion for Action Detection. In *CVPR*, 2026.

Payam Jome Yazdian, Rachel Lagasse, Hamid Mohammadi, Eric Liu, Li Cheng, and Angelica Lim. MotionScript: Natural Language Descriptions for Expressive 3D Human Motions. In *IROS*, 2025.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

Mingyuan Zhang, Huirong Li, Zhongang Cai, Jiawei Ren, Lei Yang, and Ziwei Liu. FineMoGen: Fine-grained spatio-temporal motion generation and editing. In *NeurIPS*, 2023.

Yunjiao Zhou, Xinyan Chen, Junlang Qian, Lihua Xie, and Jianfei Yang. Zero-shot open-vocabulary human motion grounding with test-time training. In *AAAI*, 2026.

SUPPLEMENTARY MATERIAL

This supplement provides the definitions, evaluation procedures, controls, and examples needed to interpret and reproduce the results.

A. Method details ..... p. 16

A.1 Semantic categories ..... p. 16

A.2 Operator registry ..... p. 16

A.3 Operator boundaries and thresholds ..... p. 17

A.4 Temporal relations ..... p. 18

A.5 MLLM interface ..... p. 19

B. Benchmarks ..... p. 20

B.1 ZJU-Refer-3D ..... p. 20

B.2 4D referring evaluation ..... p. 20

C. Evaluation details ..... p. 20

C.1 Metrics ..... p. 20

C.2 Implementation and comparison protocols ..... p. 21

C.3 Computational cost accounting ..... p. 23

D. Additional results ..... p. 23

D.1 Smaller MLLMs ..... p. 23

D.2 Qualitative examples ..... p. 24

E. Limitations and future directions ..... p. 28

F. Complete query list ..... p. 29

G. MLLM instructions ..... p. 30

A METHOD DETAILS

This section expands the definitions and execution rules introduced in Sec. 4 of the main paper. We first describe the semantic categories available for region selection, then detail the geometric operators and temporal relations used to localize poses and motions, and finally explain the MLLM interface for selecting and composing these elements. Throughout, we use the query *"Select the upper clothing while both upper arms lift outward until they are parallel to the ground"* from Fig. 1(b) of the main paper as a running example to show how category choices, operators, and temporal relations determine the selected region and intervals.

A.1 SEMANTIC CATEGORIES

The semantic memory uses the fixed category set $\mathcal{K}$ of Φ_{parse} (Sapiens2 (Khirodkar et al., 2026)), which contains 16 foreground categories: `hair`, `face_neck`, `torso`, `left_arm`, `right_arm`, `left_hand`, `right_hand`, `left_leg`, `right_leg`, `upper_clothing`, `lower_clothing`, `left_shoe`, `right_shoe`, `left_sock`, `right_sock`, and `eyeglass`. Separate anatomical left and right labels allow queries to select either side independently.

For each avatar, a category c is available for querying when at least one Gaussian has a category score $p_{ic} \geq \tau$. By Eq. 2 of the main paper, the threshold $\tau = 0.4$ requires at least 40% of that Gaussian's accumulated rendering contribution to non-background person pixels to carry label c. These available categories form $\mathcal{K}_{\text{reg}} \subseteq \mathcal{K}$. Each Gaussian retains all category scores and can be associated with multiple categories that meet the threshold. At query time, Φ_{MLLM} reads the names and descriptions of these available categories to select the target region. For the running query, it selects `upper_clothing`. The upper arms mentioned in the query define its motion condition, which is handled by the operators described next.

A.2 OPERATOR REGISTRY

The operators that check a query's motion conditions are built from the operator registry $\mathcal{R}$, a fixed set of templates shared across avatars. A template pairs one geometric measurement with one condition on it. For example, the distance between two body points gives the *near* template, a state that checks whether the two points are close at a frame, and the *approach* template, a change that checks whether their distance decreases over time. A template does not yet name a body part; applying it to a compatible body part makes it an executable operator.

Tab. A lists the 46 templates of the registry $\mathcal{R}$ by their 7 measurement types, with the 20 state and 26 change templates in separate columns. We define these templates by adapting the geometric pose and motion codes of PoseScript and MotionScript (Delmas et al., 2022; Yazdian et al., 2025) to conditions over stored SMPL trajectories. They include paired directions and pose states, such as *above/below* and *near/far*, as well as joint-angle degrees and vertical and horizontal segment states. The same measurement and condition form one template across compatible body parts. For example, forward and backward head tilt use the *segment forward* and *segment backward* templates applied to the neck-to-head segment. New pose and motion conditions can be supported by recombining existing operators or adding templates with their measurement rules and text cards.

Choosing a body part specifies the SMPL joints on which a template operates. For an elbow, the *bend* template measures the shoulder–elbow–wrist angle and checks when it decreases, whereas the *broad bent* template uses the same joints but checks whether the angle lies in its specified range. For an upper arm, the *horizontal* template measures the shoulder–elbow segment relative to gravity. Other templates use a point and a reference, as the *above* template does for "the hand is above the head", or the distance between two points, as the *near* template does for "the hands are near each other". Paired body parts additionally specify a left-right side.

At query time, the frozen Φ_{MLLM} reads text cards describing the templates and their compatible body parts, then selects the combinations needed for the query (Sec. A.5). For the running query, *lift outward* selects the *segment outward* template (*move outward* in Fig. 2 of the main paper), which tracks the elbow relative to the shoulder along the left–right axis of the fixed frame. The phrase *parallel to the ground* selects the *horizontal* template, which checks whether the shoulder–elbow segment is horizontal. Applying both templates to each upper arm gives four operators: the left and

Table A: **Operator registry $\mathcal{R}$.** Templates grouped by measurement; counts in parentheses.

Measurement type	Definition	State templates (20)	Change templates (26)	Coordinate frame
Joint angle	Angle at a 3-joint hinge (*e.g.*, shoulder–elbow–wrist)	(7) straight, broad bent, and 5 mutually exclusive bent degrees (slight, partial, right angle, deep, full)	(2) bend; straighten	Root-local
Relative position	Point relative to a reference (state) or segment endpoint relative to its start (change), along a fixed-frame axis	(6) above/below, front/behind, left/right	(8) segment up/down, forward/backward, left/right, inward/outward	Fixed
Displacement	Point or body-root travel along a fixed-frame axis	(0) none	(8) up/down, forward/backward, left/right, inward/outward	Fixed
Root rotation	Whole-body pitch and yaw	(1) neutral pitch	(4) lean forward/backward; turn clockwise/ counterclockwise	Fixed
Segment tilt	Angle of a 2-joint segment relative to gravity	(2) vertical; horizontal	(2) toward vertical; toward horizontal	Gravity
Distance	Euclidean distance between 2 body points	(2) near; far	(2) approach; separate	Root-local
Height above lowest joint	Height of a foot or ankle above the lowest of 24 body joints	(2) near the lowest joint; above the lowest joint	(0) none	Gravity

right outward movements ρ_1 and ρ_2, and the left and right horizontal states ρ_3 and ρ_4. The following subsection explains how each operator determines its matching frames or intervals.

A.3 Operator boundaries and thresholds

State ranges. A state operator checks at every frame whether its measurement lies in its valid range $\mathcal{V}_\rho$, and the ranges follow the posecodes of PoseScript and MotionScript (Delmas et al., 2022; Yazdian et al., 2025). For joint angles, where 180° is straight, we use their bin boundaries at 45°, 75°, 105°, 135°, and 160° as fixed values rather than PoseScript's randomized binning. The *straight* template covers angles above 160°, the five bent degrees in Tab. A take the other five bins from *slight* (135°–160°) to *full* (45° or less), and the *broad bent* template covers all five. A point is above, below, in front of, behind, left of, or right of its reference only beyond a 0.15 m dead zone along the corresponding axis of the fixed frame. The root pitch is neutral within 5° of upright. A segment is vertical within 10° of vertical and horizontal beyond 80° from vertical. Two points are *near* within 0.2 m and *far* beyond 0.8 m, the closest and widest of the posecode distance bins. A foot or ankle is near the lowest body joint within 0.1 m, so no scene floor is needed.

Candidate intervals. A change operator splits its measurement $m_\rho(t)$, such as the elbow angle for the *bend* template, into runs of one direction using the frame-to-frame change $m_\rho(t) - m_\rho(t-1)$, and keeps the runs in its requested direction as candidates. A candidate starts at the frame before its first change and ends at the frame before the first change in the opposite direction, where the next run starts; frames without change in between stay in the candidate. If no opposite change follows, the

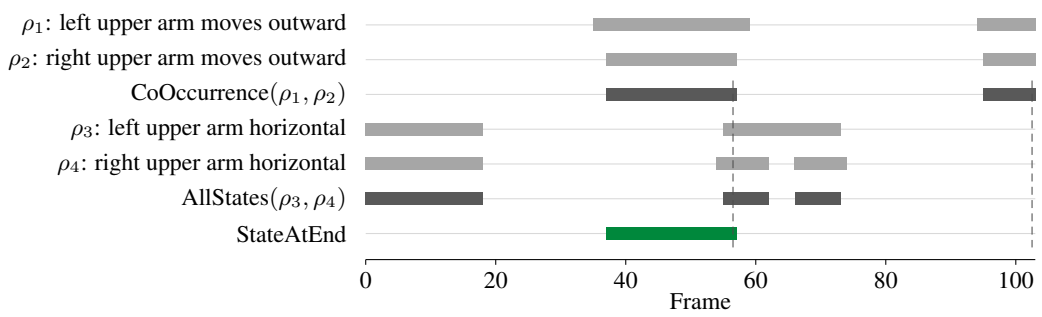


Figure A: **Execution record of the running query.** Bars mark the frames returned by each operator and relation; dashed lines mark the last frame of each co-occurrence.

candidate ends at its last change, so a plateau at the end of the sequence is excluded. For example, if the elbow angle drops at every frame from 10 to 25, holds for three frames, drops again through frame 40, and rises at frame 41, the *bend* template yields the single candidate $[9, 40]$. A candidate becomes an occurrence when it passes the two thresholds of Eq. 3 of the main paper, a minimum total change $\Delta_\rho^{\min}$ and a minimum length of $L_{\min}$ frames. Because the candidate already fixes the direction, the absolute value in that equation measures only the amount of change.

Minimum-change thresholds. We set $\Delta_\rho^{\min}$ from SMPL motion as a filter for changes too small to describe a movement. For each measurement type with change templates, we collect the total change of every run lasting at most 3 frames in the 71 available HuMMan Point sequences that are not used as 4D evaluation clips, and set $\Delta_\rho^{\min}$ to the 99.5th percentile of these changes. The resulting values, shared by HuMMan and ZJU-MoCap, are 1.49° for joint angles, 8.6 mm for relative position, 13.8 mm for displacement, 4.94° for root rotation, 6.16° for segment tilt, and 15.1 mm for distance. They only remove small changes; how far a described motion must go is set by the query, for example through a state such as *parallel to the ground*. The minimum length $L_{\min}$ removes the remaining runs of at most 3 frames whose change exceeds $\Delta_\rho^{\min}$, and we use the shortest such length, 4 frames at 30 fps. Stored as 0.1333 s and rounded up, it also gives 4 frames at 25 fps.

Change magnitudes. A modifier on a change grades how far the measurement moves within an occurrence, following the angular motion codes of MotionScript (Yazdian et al., 2025). If the start and end values lie one or two bins apart, the change is *slight*; three bins make it *moderate*, and four or more *significant*. A change of less than 5°, PoseScript's binning noise for angles and orientations, takes no modifier even if it crosses a boundary. Only joint-angle and root-rotation changes take these modifiers, and root rotation counts the bins of its own rotation axis. Modifiers thus grade changes rather than states, so *slightly bent* is the *slight* bent degree in Tab. A, whereas *slightly bending* is a bend with the *slight* modifier.

Reference frames and body shape. All measurements use joints of the default SMPL shape (Loper et al., 2015) rather than subject-specific shape coefficients, which keeps the thresholds comparable across subjects. The fixed frame is the coordinate frame fixed throughout the sequence in Sec. 4.3 of the main paper; it is set at the first frame and does not rotate when the body later turns. Root-local measurements follow the body root, and gravity-frame measurements use only the vertical axis. The last column of Tab. A gives the frame of each measurement type.

For the running query, ρ_1 and ρ_2 measure each elbow's position relative to its shoulder along the left–right axis of the fixed frame. The left elbow yields four outward candidates and the right elbow five, but only two per arm change by at least 8.6 mm, one in the middle of the clip and one at its end. ρ_3 and ρ_4 return the frames in which each shoulder–elbow segment is more than 80° from vertical, which holds at the start of the clip and again as the first lift ends (Fig. A). The next subsection combines these outputs into the requested interval.

A.4 Temporal relations

State operators return the set of frames in which a state holds, and change operators return a list of occurrences, each an interval covering one movement. Tab. B lists the relations that combine them in a motion program, with the input types each relation accepts and the output it returns.

Table B: **Temporal relations.**

Relation	Inputs	Output
CoOccurrence	2 or more movements	occurrences formed by intervals shared by every movement
AllStates	2 or more states	frames shared by all states
Sequence	movements in order; optional leading state	span from the first to the last movement, linking each last occurrence to its nearest eligible predecessors with no maximum gap; consecutive movements may share a turning frame; a leading state must end before the first movement starts and does not extend the span
Alternation	2 movements A and B	spans of alternating chains of 3 or more occurrences; consecutive movements have disjoint interiors and may share a turning frame
StateAtEnd	a movement and a state	occurrences whose last frame lies in the state
StateThroughout	a movement and a state	occurrences fully covered by the state
During	a movement and a state	each occurrence clipped to every overlapping state interval

Table C: **Text cards used for the running query.**

Card (kind)	Text read by the MLLM
`segment_move_outward` (change)	Measures: the segment endpoint, relative to its start, moves toward its anatomical outer side along the fixed sequence axis. Use when: a limb or head segment moves in a stated direction. Parts: arm, forearm, leg, lower leg, thigh, upper arm.
`segment_horizontal` (state)	Measures: the selected segment is horizontal relative to gravity. Use when: a segment is held horizontal or vertical.
`state_at_end` (relation)	Inputs: a movement, then a state. Meaning: the state describes the pose in which the movement ends. Returns: the movement occurrence whose last frame lies inside the state.

For the running query, Φ_{MLLM} returns the motion program

$$\Pi_q = \text{StateAtEnd}\big(\text{CoOccurrence}(\rho_1, \rho_2),\ \text{AllStates}(\rho_3, \rho_4)\big),$$

where ρ_1 and ρ_2 are the outward movements of the left and right upper arms and ρ_3 and ρ_4 are their horizontal states (Sec. A.2). In Fig. A, *CoOccurrence* keeps the frames in which both upper arms move outward, 37–56 and 95–102, and *AllStates* keeps the frames in which both are horizontal. *StateAtEnd* keeps only the first co-occurrence, whose last frame 56 lies in a shared horizontal state; the second ends at frame 102, where neither arm is horizontal. The program therefore returns frames 37–56, the full interval of the lift that reaches horizontal. Because each of these records can be inspected, an error can be traced to the step that produced it.

A.5 MLLM INTERFACE

At query time, Φ_{MLLM} acts as a 3D selector or a 4D compiler, each with a fixed input and a schema-constrained output, and it never sees Gaussian scores or pose trajectories. The selector and the compiler receive three and five in-context learning (ICL) examples (Brown et al., 2020), respectively, and Sec. G gives the core of each instruction and one ICL example.

3D category selection. For a 3D query (Sec. 4.4 of the main paper), the selector receives the query and the names and descriptions of $\mathcal{K}_{\text{reg}}$, but no images, pose samples, or Gaussian scores. It returns the target categories $\mathcal{C}_q \subseteq \mathcal{K}_{\text{reg}}$, whose Gaussians form the selected region, or an empty list.

4D category-and-motion compilation. For a 4D query, the compiler receives the query, the same category names and descriptions, the text cards of the templates in the registry $\mathcal{R}$ and of the relations in Tab. B, but no video or measured pose values; Tab. C shows the cards used for the running query. It returns the target categories $\mathcal{C}_q$ and a motion program Π_q, or declares the query unsupported. For the running query, one call returned `upper_clothing` and Π_q.

B BENCHMARKS

B.1 ZJU-REFER-3D

Following the evaluation scales of scene-language benchmarks (Kerr et al., 2023; Qin et al., 2024; He et al., 2025; Li et al., 2025), ZJU-Refer-3D extends ZJU-MoCap (Peng et al., 2021) with two forms of human annotation. For dense segmentation, 9 subjects are annotated at 8 frames in 4 held-out cameras, yielding 288 complete semantic ground-truth masks over the 16 foreground categories listed in Sec. A.1. For 3D referring segmentation, ZJU-Refer-3D contains 20 subject-specific queries spanning direct part names, appearance, wearing relations, and functions. Each query is associated with one subject and evaluated on 32 held-out images of that subject, covering eight frames from four cameras. Sec. F lists the exact referring queries.

B.2 4D REFERRING EVALUATION

HuMMan-Refer-4D contains 20 HuMMan queries, one per sequence at 30 fps, and ZJU-Refer-4D contains 20 queries on eight ZJU-MoCap subjects at 25 fps, so the same query text can recur on different subjects. Clip durations range from 3.4 to 20.2 seconds. For HuMMan, we select sequences showing movements of specific body parts with clearly identifiable starts and ends. Annotators convert the provided fine-grained HuMMan motion descriptions (Zhang et al., 2023) into referring queries. The provided temporal annotations serve as an initial reference, but their boundaries do not always align precisely with the query conditions. Annotators therefore review each clip and manually determine the intervals satisfying the specified pose or motion conditions. We follow the same query construction and manual interval annotation procedure for ZJU-MoCap. Each query has a target mask in two held-out cameras for every active frame. Sec. F lists all HuMMan and ZJU-MoCap queries.

C EVALUATION DETAILS

C.1 METRICS

mIoU in part segmentation. For label c, let $n_c^{\cap}$ and $n_c^{\cup}$ be the prediction–GT intersection and union pixel counts (Long et al., 2015). The score is

$$\mathcal{C}_+ = \{c : n_c^{\cup} > 0\}, \qquad \mathrm{mIoU} = \frac{1}{|\mathcal{C}_+|} \sum_{c \in \mathcal{C}_+} \frac{n_c^{\cap}}{n_c^{\cup}}. \tag{C.1}$$

In Tab. 1 of the main paper, each avatar column pools the 32 images of that subject before averaging label IoUs, whereas Overall pools all 288 images. A category absent from an avatar's GT does not affect its mIoU unless predicted; a spurious prediction then contributes 0 IoU.

OA and mAcc in part segmentation. OA is the fraction of valid GT pixels assigned the correct label. mAcc computes recall for each label present in the GT and averages these recalls.

IoU for a target mask. Let M^{pred} and M^{GT} be the predicted and GT Boolean masks for an image. In 3D, pixels outside the valid human annotation are ignored. In 4D, every image pixel is evaluated, so predicted foreground outside the target counts as a false positive. We use the region overlap $\mathcal{J}$ defined by DAVIS (Perazzi et al., 2016):

$$\mathcal{J}(M^{\mathrm{pred}}, M^{\mathrm{GT}}) = \begin{cases} \dfrac{|M^{\mathrm{pred}} \cap M^{\mathrm{GT}}|}{|M^{\mathrm{pred}} \cup M^{\mathrm{GT}}|}, & |M^{\mathrm{pred}} \cup M^{\mathrm{GT}}| > 0, \\ 1, & |M^{\mathrm{pred}} \cup M^{\mathrm{GT}}| = 0. \end{cases} \tag{C.2}$$

Following the official DAVIS evaluator, the score is 1 when both the prediction and GT masks are all false, and 0 when only one mask contains foreground.[1] An all-false GT mask is an annotated empty target, not a missing annotation.

[1] DAVIS 2017 evaluation code, `db_eval_iou`, lines 25–33.

mIoU in 3D referring segmentation. Let $\mathcal{Q}_{3D}$ be the 3D query set and $\mathcal{E}_q$ the held-out images for query q. For image e, M_{qe}^{pred} and M_{qe}^{GT} are the predicted and GT masks. We compute

$$\text{mIoU} = \frac{1}{|\mathcal{Q}_{3D}|} \sum_{q \in \mathcal{Q}_{3D}} \frac{1}{|\mathcal{E}_q|} \sum_{e \in \mathcal{E}_q} \mathcal{J}(M_{qe}^{\text{pred}}, M_{qe}^{\text{GT}}). \tag{C.3}$$

For each of the 20 queries in Tab. 2 of the main paper, we average the 32 held-out images and then average the query scores equally. Four target masks are empty and follow Eq. C.2.

tIoU, precision, and recall. For query q, let $P_q, G_q \subseteq \mathcal{F}_q$ contain the predicted and GT-active frames on source-frame grid $\mathcal{F}_q$. Overlapping intervals are merged before scoring. We use the temporal measures reported by 4-LEGS (Fiebelman et al., 2025):

$$\text{tIoU}_q = \frac{|P_q \cap G_q|}{|P_q \cup G_q|}, \qquad \text{Prec}_q = \frac{|P_q \cap G_q|}{|P_q|}, \qquad \text{Rec}_q = \frac{|P_q \cap G_q|}{|G_q|}. \tag{C.4}$$

tIoU measures temporal overlap. Precision decreases when the prediction includes frames outside the GT intervals, while recall decreases when it omits GT-active frames. A denominator of 0 returns 0. Each measure compares these frame sets without thresholds or occurrence matching and is averaged equally over the 40 queries, 20 from each source.

vIoU in 4D referring segmentation. Following 4D LangSplat (Li et al., 2025), we score each frame by mask IoU rather than by the bounding-box IoU of 4-LEGS (Fiebelman et al., 2025), sum these scores over the temporal intersection, and divide by the temporal union. Let M_{qvt}^{pred} and M_{qvt}^{GT} denote the predicted and GT masks for query q, held-out camera v, and frame t:

$$\text{vIoU}_{qv} = \frac{\sum_{t \in P_q \cap G_q} \mathcal{J}(M_{qvt}^{\text{pred}}, M_{qvt}^{\text{GT}})}{|P_q \cup G_q|}. \tag{C.5}$$

Eq. C.5 thus averages a per-frame score over $P_q \cup G_q$. Frames in both sets score $\mathcal{J}$, whereas extra frames ($P_q \setminus G_q$) and missed frames ($G_q \setminus P_q$) score 0. Because $\mathcal{J} \leq 1$, vIoU_{qv} never exceeds tIoU_q and reaches it only when every overlapping frame matches the target mask exactly. Let $\mathcal{Q}_{4D}$ denote the 4D query set and $\mathcal{W}_q$ its held-out cameras. The score in Tab. 3 of the main paper averages cameras within each query and then averages the queries:

$$\text{vIoU} = \frac{1}{|\mathcal{Q}_{4D}|} \sum_{q \in \mathcal{Q}_{4D}} \frac{1}{|\mathcal{W}_q|} \sum_{v \in \mathcal{W}_q} \text{vIoU}_{qv}, \tag{C.6}$$

where $|\mathcal{Q}_{4D}| = 40$, with 20 queries per source, and $|\mathcal{W}_q| = 2$.

C.2 Implementation and comparison protocols

Our configuration. All models are frozen public releases, namely Sapiens2-1B segmentation[2] (Khirodkar et al., 2026) for Φ_{parse} and Qwen3.8-27B[3] (Qwen Team, 2026b) for Φ_{MLLM}. Each avatar is reconstructed with 3DGS-Avatar (Qian et al., 2024) from eight training cameras at 512×512 resolution, using the SMPL (Loper et al., 2015) fits provided with each dataset, for 15K iterations on ZJU-MoCap and 20K on HuMMan. For the registration set $\mathcal{D}_{\text{reg}}$, we sample four of eight cameras every 32 frames on ZJU-MoCap and six of eight cameras every 2 frames on HuMMan. Foreground masks are the provided ones on ZJU-MoCap and come from Sapiens2 on HuMMan, after which HuMMan frames and masks are center-cropped to a square. Sapiens2 follows its official pipeline, which resizes each input directly to 1024×768. The input is the full undistorted frame on ZJU-MoCap and a square crop around the person on HuMMan, and the predicted labels are resized to match the 512×512 avatar images.

Masks from Gaussian selections. For our method, a Gaussian is active when its category score reaches τ=0.4. Active Gaussians render as 1 and all others as 0, with geometry, opacity, and occlusion unchanged. We map the rendering back to the original camera image and binarize the rendered values at > 0.7, following OpenGaussian's silhouette cutoff (Wu et al., 2024).

[2] facebook/sapiens2-seg-1b
[3] Qwen/Qwen3.8-27B

Ablation models. The MLLM ablation in Tab. 4 (left) of the main paper uses Qwen3.8-27B, Qwen3.6-27B[4] (Qwen Team, 2026a), Gemma 4-31B[5] (Gemma Team, 2026), and Mistral Small 3.2-24B[6]. The smaller MLLMs in Sec. D.1 are Gemma 4 E2B[7] (Gemma Team, 2026), Ministral 3 8B[8] (Liu et al., 2026), Qwen3-VL-8B[9], and Phi-4[10]. Each MLLM receives the same four-view inputs and a 4,096-token generation cap per call; query calls contain no images. The selector ablation uses CLIP[11] (Cherti et al., 2023), E5[12] (Wang et al., 2024b), and JINA-v5[13] (Akram et al., 2026).

OpenHuman3D and OpenHuman4D. OpenHuman3D (Suzuki et al., 2025a) projects multi-view SAM (Kirillov et al., 2023) region proposals onto the shared avatar and fuses their HumanCLIP features, which come from an AlphaCLIP (Sun et al., 2024) image encoder fine-tuned on about 1.38 million human-centric region–caption pairs from four human-parsing datasets. OpenHuman4D (Suzuki et al., 2025b) additionally tracks regions through time with SAM2 (Ravi et al., 2025) and fuses the tracked observations at the evaluation frames. Both assign each region the best-scoring label among the given texts, so for referring segmentation we give them the full query and a fixed competing label, *other*, and select the regions assigned to the query. Because OpenHuman4D has no interval output, its selection stays active over the full clip.

Part segmentation. In Tab. 1 of the main paper, the methods that build a Gaussian field use the same nine avatars and 1,372 training observations, and four separate cameras are held out for evaluation. All methods are scored on the same 16 foreground categories, and text-based methods receive all 16 names at once. Find3D (Ma et al., 2025) predicts point features aligned with SigLIP (Zhai et al., 2023) text features from a point cloud with coordinates, RGB, and normals. As in Chorus's Gaussian-scene comparison with a point-cloud baseline (Li et al., 2026b), we form this input from Gaussian centers, colors, and estimated normals. Each point corresponds to one canonical Gaussian; we average its RGB over training observations and estimate its normal from the 30 nearest centers. We keep Find3D's preprocessing and model, map its point labels back to the corresponding Gaussians, and score the rendered masks against the same held-out annotations. 4D-DRESS (Wang et al., 2024d) parses mesh vertices rather than Gaussians, so we run it on a mesh reconstructed from the same training cameras at each evaluated frame and give each pixel the label of the closest vertex of the mesh triangle it sees. Its labels are mapped to our 16 categories; categories without a matching label, such as the hands and side-specific socks, stay unassigned but remain in the score.

3D referring segmentation. All methods in Tab. 2 of the main paper use the same reconstructed Gaussian avatars, queries, and human annotations. LangSplat (Qin et al., 2024) encodes its SAM (Kirillov et al., 2023) partitions with CLIP (Radford et al., 2021), compresses these features with a per-avatar autoencoder, and fits language fields to them. Dr. Splat (Kim et al., 2025) registers CLIP features on the shared Gaussians with its default settings and compresses them with product quantization; as in our registration, features gathered on posed Gaussians are stored on the corresponding canonical Gaussians. The † variants of LangSplat and Dr. Splat use the same Sapiens2 mask proposals without receiving class names, while retaining their feature encoders and query-matching procedures. With the query as its only text, Find3D leaves a point unlabeled when it does not match. ReferSplat (He et al., 2025) learns referring features for each scene from BERT (Devlin et al., 2019)-encoded phrases paired with target masks. We adapt its released learner to each avatar using five distinct train-only referring expressions per visible target, each paired with one training image; no evaluation query or human GT enters training. For ReferSplat, each expression is given to Grounded-SAM with its image to generate a pseudo mask. For ReferSplat†, the same expressions and images are paired with Sapiens2 fixed-category masks for their named components. Both variants train for 45K iterations, then train a fresh learner for 45K iterations on masks rendered by the first stage for the same training pairs.

[4] Qwen/Qwen3.6-27B
[5] google/gemma-4-31B-it
[6] mistralai/Mistral-Small-3.2-24B-Instruct-2506
[7] google/gemma-4-E2B-it
[8] mistralai/Ministral-3-8B-Instruct-2512
[9] Qwen/Qwen3-VL-8B-Instruct
[10] microsoft/Phi-4-multimodal-instruct
[11] laion/CLIP-ViT-B-16-laion2B-s34B-b88K
[12] intfloat/e5-mistral-7b-instruct
[13] jinaai/jina-embeddings-v5-text-small

Table D: **Method components** for the computational cost comparison in Tab. 5. Segmentation: 2D mask source; AE: autoencoder; ×: none.

Method	Segmentation	Input	Interpretation	Stored information
3D referring segmentation				
LangSplat	SAM	RGB views	CLIP	Field + AE
ReferSplat	Grounded-SAM	RGB views	BERT + attention	Features + attention
OpenHuman3D	SAM	RGB views	HumanCLIP	Masks + embeddings
HumanSkill-4D	Sapiens2	RGB views	MLLM	Scores + text
4D referring segmentation				
4-LEGS	×	RGB video	ViCLIP	Field + AE
4D LangSplat	Sapiens2	RGB video	CLIP + E5	Fields + AEs
OpenHuman4D	SAM/SAM2	RGB video	HumanCLIP	Masks + embeddings
HumanSkill-4D	Sapiens2	RGB + pose	MLLM + operators	Scores + text + motion

4D referring segmentation. In Tab. 3 of the main paper, 4-LEGS (Fiebelman et al., 2025) compresses ViCLIP (Wang et al., 2024e) video-text features with an autoencoder, attaches them to the Gaussians, and scores their relevance to the query text over time. It returns the frames with high relevance as the query intervals and renders the Gaussians with high relevance as the mask, as for our method. 4D LangSplat (Li et al., 2025) has the Qwen2-VL (Wang et al., 2024c) MLLM caption how each region changes over time. It compresses CLIP features of the regions and E5 (Wang et al., 2024b) embeddings of these captions with autoencoders, fits spatial and temporal language fields to them, and matches the query text against both. On the avatar, its regions come from Sapiens2 instead of scene-object tracking, and its MLLM prompt asks which body parts move, because the default prompt describes only whole-body motion. Its temporal field is a continuous deformation, as the original state-based field needs a scene deformation network that the avatar replaces.

C.3 Computational cost accounting

Tab. D summarizes the components of the methods evaluated in Tab. 5; their training, timing, and storage accounting are detailed below.

Training. Train. marks per-avatar field or autoencoder training. OpenHuman3D/4D instead use HumanCLIP, which the OpenHuman authors fine-tuned specifically for human parsing; this shared task-specific training is neither repeated for each avatar nor included in Time. HumanSkill-4D requires no additional task-specific fine-tuning or per-avatar training.

Time. Time starts with completed Gaussians, RGB views, cameras, and poses, before method-specific preparation, and ends when the first query's final masks are saved. It includes per-asset segmentation, feature extraction, training, semantic registration, pose preparation, and query execution; shared reconstruction is excluded. Model loading (e.g., SAM, ViCLIP, MLLMs, and embedding models) and all inference calls are included uniformly across methods. For ReferSplat and 4D LangSplat, Time sums separately measured stage times; for 4D LangSplat, these include feature-map rendering. We use one A100 and *"Select both hands"* on ZJU-MoCap subject 313 (one held-out view) or the HuMMan running query (103 frames, two held-out cameras). These times come from one first query per avatar, whereas scores cover the full benchmarks.

Storage. Storage counts the per-asset semantic data retained for querying: compressed fields and autoencoders, including 4-LEGS' per-frame latent features, attention weights, and neighbor indices; ReferSplat's Gaussian features and attention weights; OpenHuman proposal masks and region embeddings; and our category scores, category descriptions, and motion memory. It excludes shared Gaussians, camera and pose inputs, reusable model weights, training intermediates, and query outputs. For both LangSplat and 4D LangSplat, rendered feature maps can be regenerated from the retained fields and are excluded from storage; their generation is included in Time.

D Additional results

D.1 Smaller MLLMs

In Tab. E, we replace Φ_{MLLM} with four smaller public MLLMs under the same protocol and leave the rest of the method unchanged. For 3D queries, all four yield usable selections, and Qwen3-VL-8B comes close to Qwen3.8-27B in Tab. 4 (left) of the main paper. For 4D queries, Ministral 3 8B and Qwen3-VL-8B still answer part of the queries, whereas Gemma 4 E2B and Phi-4 almost never do. Most of the unanswered queries come from malformed outputs, such as incomplete tool calls, which count as empty predictions. A smaller MLLM such as Qwen3-VL-8B can therefore replace Φ_{MLLM} for 3D queries, whereas 4D queries rely on the reasoning capability of Φ_{MLLM} to interpret the query correctly and to follow the instructions precisely when writing a motion program.

Table E: **Smaller MLLMs.** 3D mIoU and 4D vIoU (%).

Model	mIoU	vIoU
Gemma 4 E2B	60.2	0.0
Ministral 3 8B	63.1	20.4
Qwen3-VL-8B	78.4	16.2
Phi-4	49.9	2.6

D.2 Qualitative examples

Part segmentation. Fig. B renders the stored category assignments of two avatars from two held-out cameras each, without any language query. The labels stay on the same body parts as the view changes, including small regions such as the hands and shoes, because each label belongs to a canonical Gaussian rather than to a single image.

3D referring segmentation. Figs. C and D show two queries from four held-out cameras, marked with magenta for the GT and predicted regions; as in scoring, selections are drawn on annotated person pixels only. The prior methods are LangSplat and Dr. Splat, each with its own SAM masks and with the same Sapiens2 mask proposals (†), together with ReferSplat and OpenHuman3D. The CLIP and JINA columns keep our semantic memory but replace Φ_{MLLM} with embedding selection, as in the selector ablation of Tab. 4 (middle) of the main paper. For *"Select both hands"*, several methods find the hands, but ReferSplat and LangSplat cover most of the body and OpenHuman3D selects the feet. *"Select the garment with the multicolored patch"* separates the methods more clearly: several prior predictions include unrelated clothing or body regions, while JINA selection and our method identify the patched lower garment alone.

4D referring segmentation. Figs. E–H compare 4-LEGS, 4D LangSplat, OpenHuman4D, and our method on four queries, each from a held-out camera with GT. In *"Select the face and neck while the head tilts backward"*, the target itself moves, whereas the other queries select one part under a condition on another, as in *"Select the right hand while the left foot is off the floor"*. In all four cases, our intervals start and end close to the GT intervals, including both occurrences of the held pose in *"Select the upper clothing while the left upper arm is nearly parallel to the ground"*. In contrast, 4-LEGS and 4D LangSplat turn on at many moments outside the GT intervals, and OpenHuman4D, which has no interval output, stays active over the whole clip.

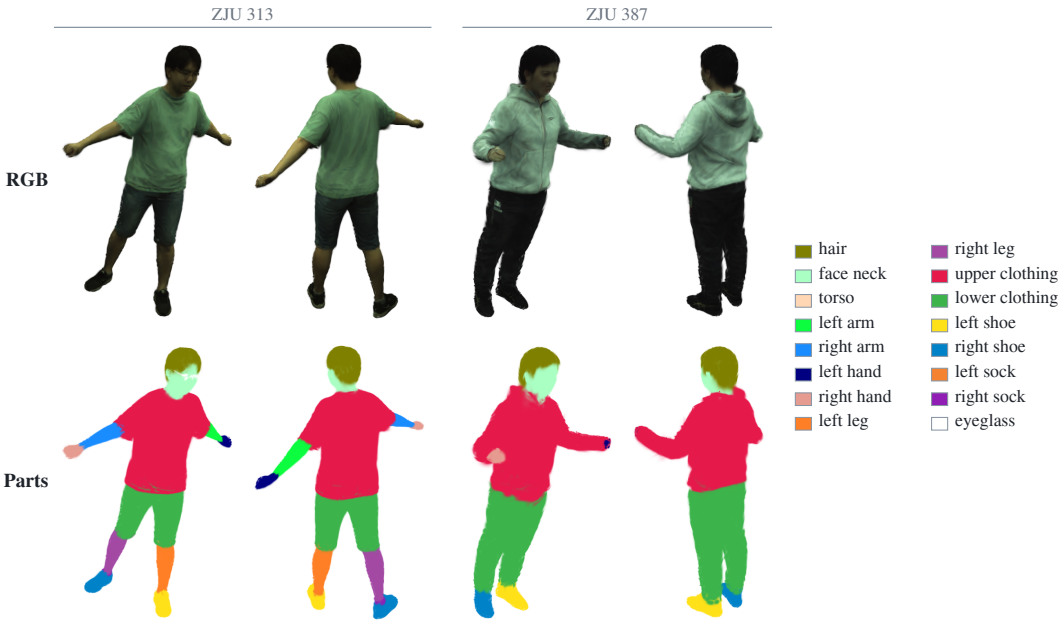


Figure B: **Registered semantic regions on ZJU-Refer-3D.**

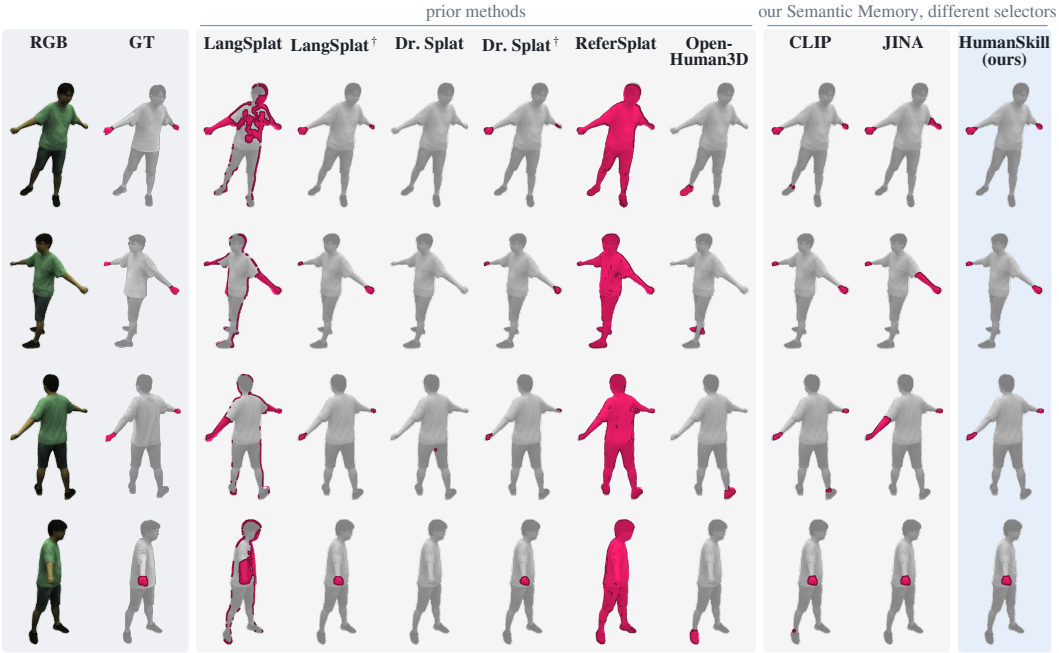


Figure C: **Qualitative 3D referring segmentation on ZJU-Refer-3D.** † uses Sapiens2 masks.

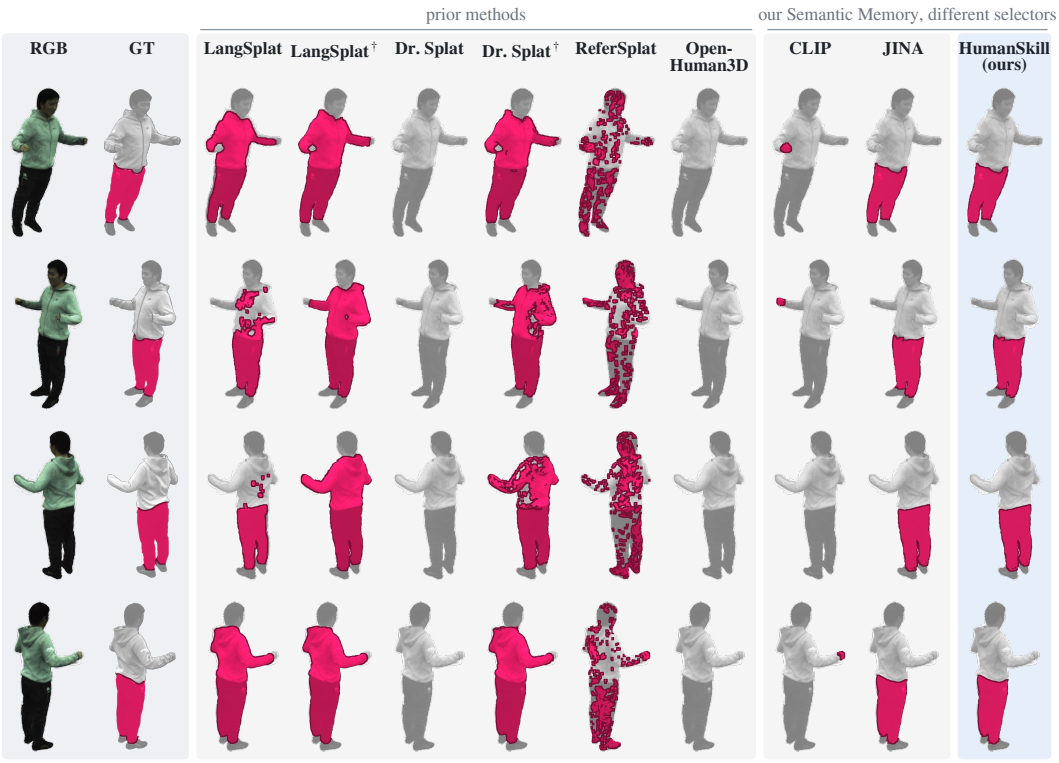


Figure D: **Qualitative 3D referring segmentation on ZJU-Refer-3D.** † uses Sapiens2 masks.

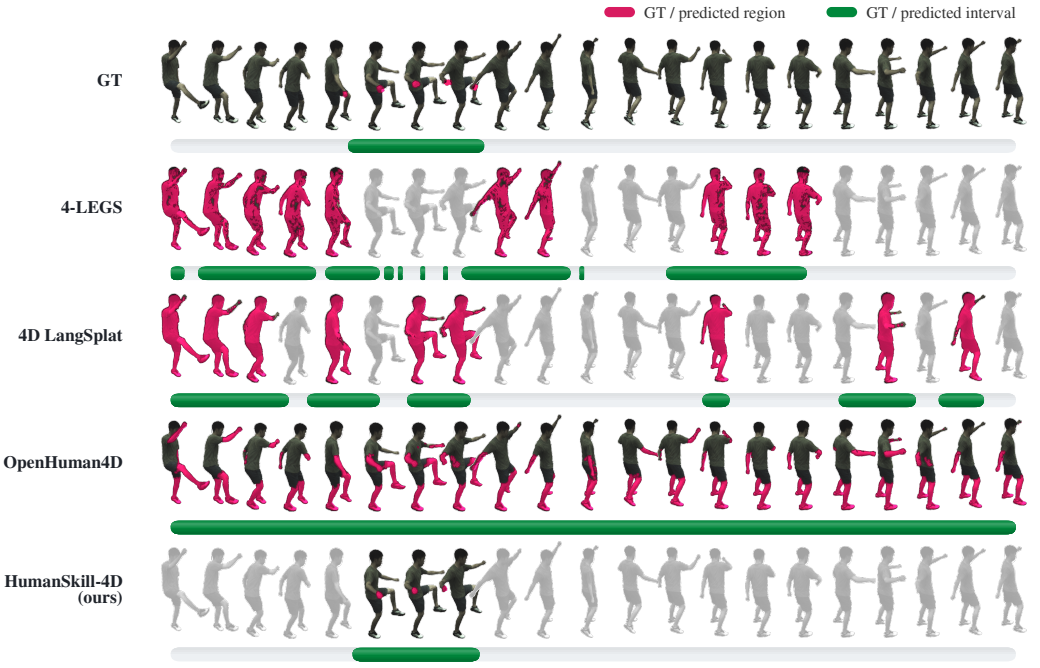


Figure E: **Qualitative 4D referring segmentation on HuMMan-Refer-4D.** Faded frames are unselected.



"Select the upper clothing while the left upper arm is nearly parallel to the ground."

Figure F: **Qualitative 4D referring segmentation on ZJU-Refer-4D.** Faded frames are unselected.



"Select the right hand while the left foot is off the floor."

Figure G: **Qualitative 4D referring segmentation on ZJU-Refer-4D.** Faded frames are unselected.



"Select the upper clothing while the left arm moves upward."

Figure H: **Qualitative 4D referring segmentation on ZJU-Refer-4D.** Faded frames are unselected.

E LIMITATIONS AND FUTURE DIRECTIONS

Region selections follow the 16 categories of Φ_{parse}, so a part within a category, such as a sleeve or a printed logo, is returned as its whole garment. Combining Φ_{parse} with other segmenters can resolve this, since the same registration accumulates their labels on the Gaussians. For motion, coarse action labels (e.g., "while dancing") leave the moving body parts and the start and end of the action unspecified, which makes their intervals ambiguous as ground truth. We therefore follow prior work on fine-grained human motion (Delmas et al., 2022; Yazdian et al., 2025; Zhang et al., 2023; Wu et al., 2025), whose part-level descriptions give precise targets, and define each motion condition by which body parts move, how, and when, so that it can be checked on the driving pose sequence. Coarse action labels and conditions involving objects, such as catching or carrying, are thus outside the current scope. Because Φ_{MLLM} composes motion programs from the registry $\mathcal{R}$, such queries could be supported by extending the same design, for example by decomposing a coarse label into fine-grained operators or by adding object-relative templates given object trajectories, which makes them a promising future direction.

F COMPLETE QUERY LIST

Tab. F lists the ZJU-Refer-3D queries of Tab. 2 of the main paper, and Tabs. G and H list the HuMMan-Refer-4D and ZJU-Refer-4D query subsets of Tab. 3, each exactly as evaluated.

Table F: **ZJU-Refer-3D queries.**

ID	Query
Q01	Select both hands.
Q02	Select both arms.
Q03	Select both shoes.
Q04	Select the upper clothing and lower clothing.
Q05	Select the garment with a zipper down the front.
Q06	Select the yellow garment.
Q07	Select the garment with long sleeves.
Q08	Select the garment with the multicolored patch.
Q09	Select the sleeveless garment.
Q10	Select the denim garment.
Q11	Select the garment worn below the shirt.
Q12	Select the garment with a crown-shaped graphic on the front.
Q13	Select the garment with horizontal stripes across the front.
Q14	Select the garment with a large oval graphic on the front.
Q15	Select the garment with a white drawstring.
Q16	Select the accessory worn on the face.
Q17	Select the items with gold-colored crossing markings.
Q18	Select the garment with a hood and a large front pocket.
Q19	Select the garment with a plain upper section and horizontal stripes below it.
Q20	Select both body parts used to grasp objects.

Table G: **HuMMan-Refer-4D queries.**

ID	Query
F01	Select the upper clothing while both upper arms lift outward until they are parallel to the ground.
F02	Select the left hand while the person lowers into a squat.
F03	Select the lower clothing while the left foot moves forward and the right foot remains on the ground.
F04	Select the lower clothing while the left knee bends.
F05	Select the right hand while the right knee bends and the body lowers.
F06	Select the lower clothing while the right knee lifts.
F07	Select the face and neck while the head tilts backward.
F08	Select the lower clothing while the left knee lifts.
F09	Select the lower clothing while the right knee straightens.
F10	Select the lower clothing while the body lowers.
F11	Select the lower clothing while the right foot moves to the right.
F12	Select the left hand while the left knee bends and the body lowers.
F13	Select the lower clothing while the left foot moves backward.
F14	Select the left hand while it moves toward the right knee as the right knee moves upward.
F15	Select the left hand while the left arm moves forward and rotates toward horizontal.
F16	Select the lower clothing while the left knee straightens.
F17	Select the lower clothing while the right knee bends and the body lowers.
F18	Select the left shoe while the left foot moves to the left.
F19	Select the upper clothing while the left upper arm rotates toward horizontal as the left foot moves to the left.
F20	Select the lower clothing while the left knee bends and the body lowers.

Table H: **ZJU-Refer-4D queries.**

ID	Query
Z01	Select the right leg each time the right knee bends.
Z02	Select the left leg while the left knee bends.
Z03	Select the upper clothing while the left upper arm is nearly parallel to the ground.
Z04	Select the right hand while the right upper arm is nearly parallel to the ground.
Z05	Select the right shoe while the whole right leg lifts until it is nearly parallel to the ground.
Z06	Select the left shoe while the left foot is off the floor.
Z07	Select the left hand while it is above the left shoulder.
Z08	Select the right shoe while the right thigh is nearly parallel to the ground.
Z09	Select the left shoe while the left thigh is nearly parallel to the ground.
Z10	Select the right shoe while the right thigh is nearly parallel to the ground.
Z11	Select the left hand while it is above the left shoulder.
Z12	Select the left hand while it is above the head.
Z13	Select the upper clothing while the left arm moves upward.
Z14	Select the left shoe while the whole left leg lifts until it is nearly parallel to the ground.
Z15	Select the right shoe while the right knee bends.
Z16	Select the right hand while it is above the head.
Z17	Select the left shoe while the left foot is off the floor.
Z18	Select the left hand while it is above the head.
Z19	Select the right shoe while the whole right leg lifts until it is nearly parallel to the ground.
Z20	Select the right hand while the left foot is off the floor.

G MLLM INSTRUCTIONS

Tabs. I–K give the core of each role's instruction and, where used, one of its ICL examples. Text is quoted from the prompts, with function names, field values, and ICL examples in `typewriter`, and [...] marks omitted text; the examples use illustrative inventories rather than evaluation queries.

Table I: **Instruction for the appearance describer.**

Role	You are the Appearance Describer. Record visible appearance for the supplied categories from synchronized images.
Task	Call `submit_multiview_descriptions` once. Describe every supplied category by combining compatible evidence from all supplied views. [...] When no appearance description can be extracted for a category, return its exact class name with status `class_name`.
Rules	Treat them as one observation set and describe only appearance. [...] Combine complementary front, back, and side details [...] Left and right refer to the person's anatomical sides, not the image sides.
Input	{category IDs and names}; {four views from the frame closest to a T-pose}

Table J: **Instruction and ICL example for the 3D selector.**

Role	You are the 3D Semantic Selector. Map the requested static target to categories supported by Semantic Memory.
Task	Call `select_semantic_labels` once with the stored categories that match the full static request, or abstain when no supported match exists.
Rules	Use category names, their ordinary meaning and synonyms for structural identity, and stored descriptions for appearance. [...] Every requested color, material, texture, logo, graphic, shape, or placement detail needs compatible stored evidence. [...] Select only categories present in the supplied inventory.
Input	{query}; {inventory rows: category ID, name, description, status}
ICL input	`{"query": "Find the dark shirt with a small front logo and a large rear graphic.", "semantic_inventory": [{"id": 1, "name": "upper_clothing", "text": "A dark shirt has a small front logo and a large graphic across the back.", ...}, {"id": 2, "name": "lower_clothing", ...}]}`

ICL output	`{"activation_label_ids": [1], "abstain": false, "reason": "The stored upper-clothing description matches both requested graphics."}`

Table K: **Instruction and ICL example for the 4D compiler.**

Role	You are the Fine Motion Binder. First decide whether the exact query is expressible using the supplied Semantic Memory and body measurements, then return its program or the specific missing support. [...]
Task	Return exactly one tool call. Use `submit_component_bindings` for a fully supported query [...] Use `report_query_unsupported` when the supplied inventory or cards cannot express the full request
Rules	The answer target and the subject of a WHEN condition are independent [...] One component describes one movement or one state of a measurable part or bilateral pair. [...] A destination pose qualifies the complete movement that ends in it with `state_at_end`. [...] Report unsupported if the target lacks inventory support, a required anatomical side is missing, no body motion or state is requested, or any required measurement or temporal relation cannot be expressed.
Input	{query}; {inventory}; {template and relation text cards as in Tab. C}
ICL input	`{"query": "Select the buckle while the body turns counterclockwise.", "semantic_inventory": [{"id": 29, "name": "buckle"}]}`
ICL output	`{"target_phrase": "the buckle", "components": [{"component_id": "c01", "kind": "motion", ...}], "root": "c01", "target_label_ids": [29], "bindings_by_component": [{..., "bindings": [{"atom_id": "root_turn_counterclockwise", "part": "body", ...}]}], ...}`