ArtHOI: Taming Foundation Models for Monocular 4D Reconstruction of Hand-Articulated-Object Interactions

Supplementary Material

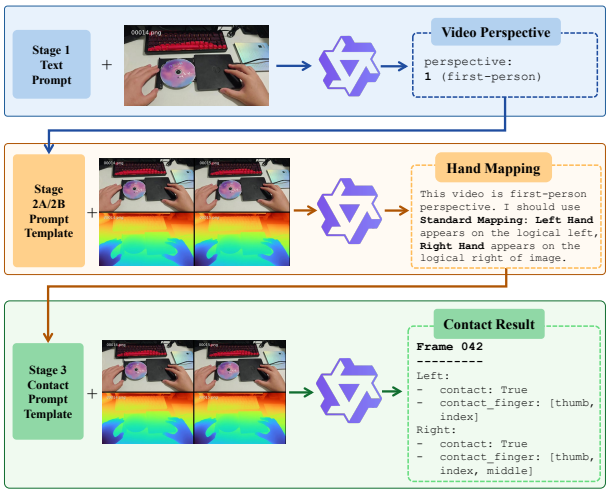

Figure A. Demonstration of our MLLM contact reasoning pipeline. For clarity, we merge 2 neighbouring frames, but in practice, it's typically set to 3. The top row shows RGB frames, the bottom row shows colorized depth maps. The MLLM analyzes visual and depth cues across frames to determine contact status and engaged fingers for each hand.

A. Implementation Details

A.1. Coarse Metric Scale Estimation of Object

We detail the coarse scale estimation introduced in Sec. 3.2. Given estimated metric depth maps, we first back-project them into 3D space using camera intrinsics $\mathbf{K}$ and the object mask. To suppress boundary noise, the mask is eroded prior to back-projection, followed by a Statistical Outlier Removal (SOR) filter to further clean the point cloud. We then compute the bounding boxes of both the normalized canonical object and the back-projected depth point cloud. The coarse metric scale s^o_{coarse} is obtained as the maximum ratio between their extents along the x- and y-axes. The z-axis (depth direction) is excluded because the back-projected point cloud only captures the visible object surface and is typically more noisy and unreliable in depth.

A.2. Object Part Segmentation

Sec. 3.3 describes the reconstruction of part-wise motion for articulated objects. Here, we provide additional details on the part partition process. We begin by applying PartField [3] to extract per-vertex feature fields, followed by agglomerative clustering to obtain vertex group labels.

Table A. Comparison of contact accuracy (Acc.) and false positive rate (FP) between our MLLM-based contact reasoning and a rule-based mask-intersection heuristic. While both methods perform similarly on the controlled RSRD dataset, the heuristic degrades notably on in-the-wild videos, whereas the MLLM remains robust. ArtHOI-RGBD is excluded due to its near-perfect accuracy.

	RSRD [1]		ArtHOI-Wild	
Contact Judge	Acc. ↑	FP ↓	Acc. ↑	FP ↓
Mask Intersection	0.86	0.14	0.76	0.23
MLLM	**0.89**	**0.11**	**0.87**	**0.10**

The object is then rendered in its canonical pose using PyTorch3D [6] to produce a 2D label map. Vertex groups are merged according to part masks, after which the mesh is finally split into individual parts.

A.3. MLLM Contact Reasoning

We adopt an image-text question-answer strategy to extract contact information for each frame of input video. The primary challenge of this task lies in suppressing false positives: in real-world videos, both humans and models often confuse near-contact with genuine physical contact, while clear separation is seldom misidentified as contact, making false negatives comparatively rare. To mitigate this, we augment RGB frames with colorized depth, incorporate neighboring-frame sampling to strengthen spatio-temporal cues, and explicitly instruct the MLLM to be cautious about false positives. Furthermore, because the input videos may be egocentric or exocentric, we identify video perspective beforehand to reduce hallucinations on hand laterality when reasoning about bimanual contact. Figures C, D, and E demonstrate the full prompt templates used in our pipeline.

Input and Output Format To provide richer contextual cues, we concatenate k neighboring frames ($k = 3$ in practice) along with their colorized depth maps into a single large image prompt, which the MLLM can jointly analyze for spatio-temporal consistency. The depth maps are visualized with a color gradient (blue for near, red for far), making depth discontinuities visually salient to the model. The output is a structured JSON containing: (i) frame count and which hands appeared in the video; (ii) for each frame, binary contact flags for left and right hands; (iii) lists of contacting fingers for each hand-frame pair, empty if no contact. This structured format enables downstream optimiza-

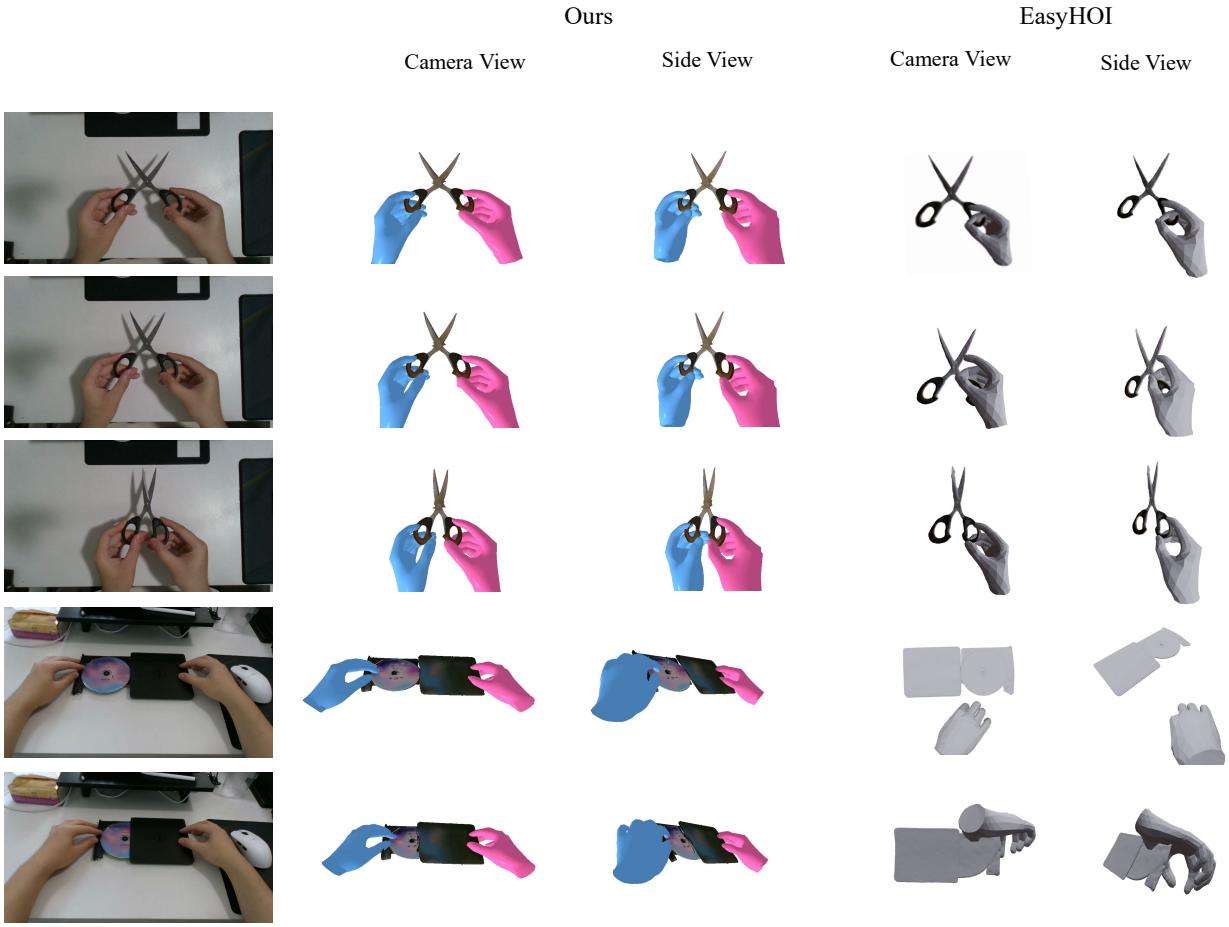

Figure B. Qualitative comparison between our method and EasyHOI [4] on ArtHOI-RGBD. EasyHOI often fails to recover articulated object scale and pose, and exhibits inconsistent hand-object alignment across frames.

tion to directly parse and apply contact constraints.

Three-Stage Prompting Strategy The MLLM contact reasoning pipeline consists of three carefully designed stages, as shown in Figures C, D, and E.

Stage 1: Perspective Detection. Video perspective (egocentric vs. exocentric) significantly affects hand laterality interpretation. In first-person perspective, a single visible hand is automatically from the operator's viewpoint, and spatial relationships are relatively straightforward. In third-person perspective, MLLM must infer the operator's orientation and account for mirror effects to correctly identify hands. By first explicitly determining the perspective, we reduce hallucinations on hand identity in subsequent reasoning stages.

Stage 2: Hand Mapping. After identifying perspective, hand mapping disambiguates left and right hands through perspective-specific heuristics. For first-person videos (Stage 2a), spatial positioning and thumb direction provide direct cues. For third-person videos (Stage 2b), the strategy shifts to analyzing relationship between camera and the operator's body. In this stage, the MLLM can map visible hands to left or right labels.

Stage 3: Frame-wise Contact Reasoning. Given correct hand identity, Stage 3 performs detailed frame-by-frame contact analysis. For each visible hand, the prompt guides the MLLM through a structured reasoning chain. The prompt emphasizes caution: uncertain cases should be marked as no-contact (`false`) to suppress false positives. This conservative bias aligns with our observation that false positive predictions in real-world contact cases are more often than false negatives.

B. Computational Performance

For a video sequence of 150 frames at a resolution of 960×540, preprocessing (mask segmentation, metric depth estimation, frame inpainting, hand estimation, and mesh re-

Stage 1 Perspective Detection Prompt (`prompt_perspective`)

You are given a set of images sampled from a video about a human manipulating an articulated object.
Please determine whether this video is from a **first-person perspective** or a **third-person perspective**.

If it is first-person perspective, output only the number 1.
If it is third-person perspective, output only the number 3.
Do not output any other text, explanation, or thoughts.

First-person: filmed from the operator's point of view; arms/hands extend from the bottom or sides of the frame; viewpoint aligned with the operator's head direction.
Third-person: filmed from an observer's point of view; shows the whole/most of the operator's body; viewpoint not aligned with the operator's head direction.

Judgment principles:
1. If only one hand appears, it must be first-person perspective.
2. If a human face appears, it must be third-person perspective.
3. In first-person perspective, the hand(s) usually occupy a large area of the image.

Figure C. Stage 1: Perspective Detection Prompt. This prompt determines whether the input video is from a first-person or third-person viewpoint, which is essential for correctly identifying hand laterality in subsequent stages.

Stage 2a Hand Mapping — First-Person

Step 1: Perspective Analysis & Hand Mapping (Reasoning Chain of Thought Example)
This video is **First-person** perspective. The camera mimics the operator's eyes. I need to determine carefully about hand side.
1. If there's two hands, then the hand on the left side of the image is the Left Hand, and the hand on the right side is the Right Hand. If only one hand appears, I need to determine carefully.
1. **Standard Mapping**: Usually, the Left Hand enters from the logical left, and the Right Hand from the logical right.
2. But I also need to examine the thumb direction to determine. Thumb of left hand is pointing right, and thumb of right hand is pointing left.

So I can check the thumb direction to determine the hand side, especially when only one hand appears.
3. **Apply Mapping**: Use these cues to strictly confirm the identity of any visible hands before proceeding.

Stage 2b Hand Mapping — Third-Person

Step 1: Perspective Analysis & Hand Mapping (Chain of Thought)
This video is **Third-person** perspective. You must determine the exact camera angle to identify hands correctly:
1. **Analyze Operator Orientation**: Look at the person's body/head in the RGB frames.

2. **Determine View Type & Arm Connectivity**:
-- **Frontal/Side-Front View**: Camera faces the person. → **Logic**: Mirror effect (Left Hand is on Right, Right Hand is on Left).
-- **Rear/Side-Rear View**: Camera looks at the person's back or side-back. Use **Arm Connectivity** to identify hands:
* **Right Side-Rear**: Camera observes from the operator's right-back. The **Right Arm** is visibly connected to the body on the right. → **Logic**: The hand connected to this visible right arm is the **Right Hand**. The other hand is the Left Hand.
* **Left Side-Rear**: Camera observes from the operator's left-back. The **Left Arm** is visibly connected to the body on the left. → **Logic**: The hand connected to this visible left arm is the **Left Hand**. The other hand is the Right Hand.

3. **Apply Mapping**: Based on the arm connections, strictly assign 'Left Hand' and 'Right Hand' labels before proceeding.

Figure D. Stage 2: Hand Mapping Prompt. This stage identifies and maps visible hands to left/right labels. Stage 2a handles first-person perspective videos using spatial positioning and thumb direction cues. Stage 2b handles third-person perspective videos by analyzing camera angle relative to the operator's body and arm connectivity patterns.

construction with HunYuan3D [2]) requires approximately 10 to 15 minutes. Optimizing the canonical object metric scale and pose takes less than 2 minutes. Part-wise motion recovery is the most time-consuming stage and takes roughly 30 minutes; during this stage, our pipeline could concurrently perform MLLM contact reasoning to obtain HOI contact information. Finally, aligning the separately reconstructed hands and the articulated object requires up to 5 minutes, yielding the final result. Overall, the full pipeline runtime is dominated by the coarse-to-fine part-wise motion reconstruction, which can be accelerated with a more optimized implementation.

For comparison, RSRD [1] reports similar overall runtime: about 40 minutes to reconstruct and segment the 3D part model from pre-scanned video, roughly 7 minutes for part-motion reconstruction and 4D hand estimation, yet it does not perform any hand-object joint optimization.

C. Additional Results

Qualitative Comparison with EasyHOI We compare our approach with EasyHOI [4], a monocular image HOI

Stage 3 Contact Reasoning Prompt (`prompt`, appended after Stage 2)

```
Step 2: Frame-by-Frame Contact Reasoning
The image contains K frames (horizontally merged) separated by black bars.
 -- Top row: RGB frames. -- Bottom row: Depth frames (blue=near, red=far).

For each frame, analyze the 'Left Hand' and 'Right Hand' (identified in Step 1) separately using the following Chain of
Thought:
 A. Visibility Check: Is the hand visible? If not, skip.

   (A.1) Left or Right hand may be occluded by the articulated object. I need to identify partial, occluded hand around
object, not missing them.
 B. Object contact estimation: Is the hand close enough to contact the articulated object (but not background) in the RGB
frame?
   -- If the hand is clearly distant from the object or merely contact, mark contact:false and skip to next.
   (B.1) I need to carefully identify the hand appearance, fully utilizing both RGB and depth.
   (B.2) I need to carefully determine if the hand is contacting the articulated object in a solid state, or it's in mere
contact (which I should output FALSE).

 C. Depth Map Verification (Critical Phase):
   -- Look at the bottom row (Depth map) corresponding to the hand's position.
   -- Does the hand's depth color seamlessly merge with the object's at the interaction point?
   -- Is there a sharp edge or color contrast separating them? If YES -> FALSE CONTACT.
   -- Mark contact: true only if depth values merge without discontinuity.
 D. Finger Analysis:
   -- If contact: true, identify specific fingers (thumb, index, middle) involved.
   -- If a finger is occluded or ambiguous, exclude it.

Step 3: Consistency & Final Decision
 -- Review your frame-by-frame findings.
 -- Ensure the contact status transitions logically (e.g., hand approaches -> touches -> leaves) and is fully supported by the
visual evidence in each frame independently.
 -- Combining the neighbouring frames, make sure judge merely contact frame as FALSE contact.

Step 4: Output Generation
IMPORTANT: If not 100% sure about contact ⇒ output false.

Do not simply output the same status for all frames; look for changes.
Please only output the JSON without any additional text or markdown format. output it just using text.

Output format (JSON only; r_/l_fingers: valid fingers only, empty list if no contact):

{ "frames_cnt": K,
  "appeared": ["left", "right"], // list only hands that appeared
  "contacts": [

    { "frame": <frame_number noted on the corner of each frame, output the number using int, without .png or output string>,
      "r_contact": <bool>, "l_contact": <bool>,
      "r_fingers": ["thumb","index","middle"], // list valid fingers only, empty if no contact
      "l_fingers": [] },
    ...
  ]
}
```

Figure E. Stage 3: Frame-wise Contact Reasoning Prompt. This stage performs detailed analysis of each frame to determine contact state and identify engaged fingers. The critical depth map verification step (Phase C) distinguishes true physical contact from mere proximity using depth discontinuity analysis.

reconstruction method that also leverages foundation models. Since EasyHOI accepts only single images, we evaluate it frame by frame. For a fair comparison, we use the same foundation models as in our pipeline: WiLoR [5] for hand reconstruction and HunYuan3D [2] for object shape reconstruction.

Figure B shows EasyHOI results on ArtHOI-RGBD using single-frame input. EasyHOI generalizes poorly to articulated manipulation because it assumes a fixed object scale and 6-DoF pose, and instead optimizes camera parameters and object pose to fit each image. While this image-based paradigm can be efficient for isolated frames, it clearly fails to produce coherent results on videos.

Moreover, EasyHOI struggles to maintain consistent hand-object alignment across frames. It optimizes contact by considering the entire plausible hand interaction region, which is sufficient for rigid-object grasps, but without specifying contacting fingers, its performance degrades in articulated interactions. The frame-wise reconstruction paradigm also makes video processing computationally infeasible: reconstructing a 100 frame sequence requires roughly 3 hours or more. Finally, EasyHOI assumes a single-hand setting and cannot be easily extended to bimanual scenes without substantial code modifications.

Effect of MLLM Contact Reasoning We evaluate the effectiveness of our MLLM-based contact reasoning against a simple rule-based baseline that determines contact via mask intersection. As shown in Table A, while the mask-intersection heuristic shows slightly inferior performance on controlled lab datasets, its accuracy drops substantially on casually captured in-the-wild videos. In contrast, the MLLM leverages broader visual and semantic knowledge, enabling more reliable contact judgments under challenging real-world conditions.

References

[1] Justin Kerr, Chung Min Kim, Mingxuan Wu, Brent Yi, Qianqian Wang, Ken Goldberg, and Angjoo Kanazawa. Robot see robot do: Imitating articulated object manipulation with monocular 4d reconstruction. In *8th Annual Conference on Robot Learning*, 2024. 1, 3

[2] Zeqiang Lai, Yunfei Zhao, Haolin Liu, Zibo Zhao, Qingxiang Lin, Huiwen Shi, Xianghui Yang, Mingxin Yang, Shuhui Yang, Yifei Feng, et al. Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details. *arXiv preprint arXiv:2506.16504*, 2025. 3, 4

[3] Minghua Liu, Mikaela Angelina Uy, Donglai Xiang, Hao Su, Sanja Fidler, Nicholas Sharp, and Jun Gao. Partfield: Learning 3d feature fields for part segmentation and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9704–9715, 2025. 1

[4] Yumeng Liu, Xiaoxiao Long, Zemin Yang, Yuan Liu, Marc Habermann, Christian Theobalt, Yuexin Ma, and Wenping Wang. Easyhoi: Unleashing the power of large models for reconstructing hand-object interactions in the wild. In *CVPR*, pages 7037–7047, 2025. 2, 3

[5] Rolandos Alexandros Potamias, Jinglei Zhang, Jiankang Deng, and Stefanos Zafeiriou. Wilor: End-to-end 3d hand localization and reconstruction in-the-wild. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12242–12254, 2025. 4

[6] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 1