

Supplementary Material of “Synthetic Hands Meet Legacy Data: A Synthetic Dataset for Structured, Controllable, and Multimodal Evaluation”

Menghe Zhang¹, Haley M. So², Mohammad Asadi^{1,2}, Dongfang Zhao¹, Yangwen Liang¹,
Shuangquan Wang¹, Gordon Wetzstein², Kee-Bong Song¹, Donghoon Kim³

¹Samsung Semiconductor Inc., San Diego, USA

²Stanford University, Stanford, USA

³Samsung Electronics, Hwaseong-si, South Korea

{menghe.z, d.zhao, liang.yw, shuangquan.w, keebong.s, dhoon.kim}@samsung.com

{haleyso, masadi, gordonwz}@stanford.edu

This supplementary material complements the main paper by providing additional details and results that further demonstrate the capabilities and utility of the T3DGesture dataset.

1. Additional Experimental Details

1.1. HGR from Keypoints Details

To enable consistent training and evaluation across our dataset, SHREC’22, and SHREC’21 datasets, a unified labeling of hand keypoints and their corresponding locations was necessary. The original SHREC’22 dataset includes annotations with 26 keypoints, while SHREC’21 uses 20 keypoints. Given that the MANO hand model, with 21 keypoints, is the most widely adopted standard in hand representation, we leveraged the unified SHREC’22 and SHREC’21 datasets provided by UniHands [3]. This unified data ensured all datasets were standardized to the 21 keypoints of the MANO model, enabling seamless training and evaluation across datasets.

1.2. HGR from RGB-D Details

The 3D-CNN from [1] was trained following the original author’s training recipe. To achieve the best performance on EgoGesture, the authors first pretrained their models on the Jester dataset [2]. Following suit, we started with their pretrained Jester model and trained on EgoGesture as well as EgoGesture+Our Data to achieve the best results.

1.3. Ablation Study

We also conducted ablation studies to quantify the role of the biomechanical projection layer in refining raw-collected data.

Table 1. **Impact of Intra-finger and Inter-finger Constraints on Hand Pose Data.** Comparison of hand pose metrics with and without biomechanical constraints applied. The constraints improve anatomical plausibility with minimal quantitative changes.

Dataset	$\Delta A_{avg} \downarrow$	$\Delta A_{max} \downarrow$	Thres10(%) \downarrow	$\Delta V_{max} \downarrow$
SHREC-static	0.05	12.6	1.8	2.1
SHREC-dynamic	0.3	22.2	33.1	3.8
EgoGesture	0.3	123.6	21.5	9.5
InterHand2.6M	0.01	11.5	1.9	1.4

Table 2. **Evaluations on the Collision Penetration.** We show the changes of collision metrics and pose for data w/wo collision reduction constraints.

Dataset	$\Delta C_{ratio} \downarrow$	$\Delta C_{dist} \downarrow$	Thres10(%) \downarrow	$\Delta V_{max} \downarrow$
SHREC-static	7.81	4.15	16.67	3.78
SHREC-dynamic	0.61	2.16	1.00	1.35
EgoGesture	0.54	0.32	21.80	4.30

Biomechanical Projection Layer. We assess the biomechanical projection layer by comparing raw-collected hand motions with their biomechanically refined versions on two aspects:

Intra-finger and Inter-finger Constraints. We compare hand poses with and without biomechanical correction, measuring angle changes (ΔA_{avg} , ΔA_{max}), the percentage of changes exceeding 10 degrees (Thres10), and maximum vertex displacement (ΔV_{max}) (Table 1). Our results show that quantitative changes are minor for real-world captures, but motion realism improves significantly, especially in randomly generated poses.

Collision and Self-Penetration Reduction. We evaluated the collision avoidance constraints by comparing hand poses generated with and without this component (Table 2). We evaluated changes of collision ratio (ΔC_{ratio}), collision distance (ΔC_{dist}), percentage of angle changes exceeding 10 degrees (Thres10), and maximum vertex change (ΔV_{max}).

2. Method

This section provides a detailed description of the synthetic data generation pipeline, including hand model rendering, RGB-D simulation, and annotation generation, to complement the implementation details in the main paper.

2.1. NIMBLE and MANO Hand Models

The original MANO model uses a kinematic tree with 16 joints, where each joint’s rotation is represented in axis-angle format, aligned with a wrist-orthogonal basis. To achieve anatomically accurate hand meshes, we adopt A-MANO, which redefines the canonical hand pose to calculate the twist, spread, and bend axes in an anatomically aligned orthogonal basis. The formulation of MANO can be described as:

$$M(\theta_m f, \beta_m) = \mathcal{M}_f(\mathcal{C}(\theta_m c, \beta_m)), \quad (1)$$

where \mathcal{C} denotes the composition function that aligns joint angles with anatomical axes, and \mathcal{M}_f represents the flat-hand layer in MANO. Here, $(\theta_m c, \beta_m)$ are the joint angles and shape parameters in the anatomically aligned space.

The NIMBLE model extends this formulation by incorporating both geometry and appearance modeling:

$$N(\theta_n, \beta_n, \alpha) = \mathcal{G}(\theta_n, \beta_n), \mathcal{A}(\alpha), \quad (2)$$

where \mathcal{G} models the hand geometry, and \mathcal{A} captures the hand’s textural appearance. The parameters $(\theta_n, \beta_n, \alpha)$ correspond to pose, shape, and appearance, respectively. Compared to MANO, NIMBLE’s richer training data provides greater variability in hand shapes and textures.

2.2. CVAE Motion Variation

Pose-phase Encoder The encoder processes a sequence of hand poses, frame-level phase labels, and a sequence-level gesture category label a , generating a latent representation of the motion. Inspired by the ACTOR framework, we include distribution parameters μ_a^{token} and Σ_a^{token} , which are appended to the embedded pose-phase sequence:

$$\mathbf{Z}_{input} = [\mu_a^{token}, \Sigma_a^{token}, z_1, z_2, \dots, z_n] \quad (3)$$

To retain temporal information, positional encodings $PE(i)$ are added to each input embedding:

$$\mathbf{Z}_{input} = \mathbf{Z}_{input} + PE(i) \quad (4)$$

The Transformer encoder processes this sequence, producing the distribution parameters μ and Σ from the output corresponding to the distribution tokens. A latent vector \mathbf{z} is then sampled from the learned distribution:

$$\mathbf{z} \sim \mathcal{N}(\mu, \Sigma) \quad (5)$$

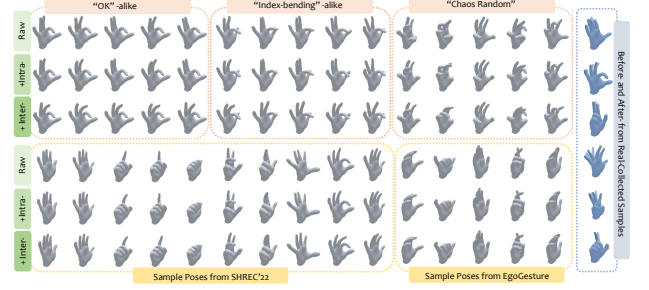


Figure 1. **Qualitative Impact of Intra-finger and Inter-finger Constraints.** Visual comparison of hand poses with and without biomechanical constraints.

Phase-aware Decoder The decoder generates realistic hand motions from \mathbf{z} and a sequence-level gesture category label a' . The latent vector \mathbf{z} serves as the *key* and *value* in the Transformer decoder, while frame-wise sinusoidal positional encodings $PE(t)$ act as the *query*:

$$Q = \{PE(1), PE(2), \dots, PE(T)\} \quad (6)$$

The decoder outputs a sequence of intermediate vectors, which are linearly projected to obtain the predicted hand poses \hat{P}_t and phase labels $\hat{\Phi}_t$. The predicted poses \hat{P}_t are then passed through a differentiable MANO layer, producing 3D joint positions and mesh vertices:

$$\hat{J}_t, \hat{V}_t = \text{MANO}(\hat{P}_t, \beta) \quad (7)$$

2.3. Biomechanical Constraints

Intra- and Inter-Finger Constraints. As shown in Fig. 1, quantitative changes are minor for real-world captures, but motion realism improves significantly, especially in randomly generated poses.

Collision-Guided Anti-Penetration. To ensure anatomically plausible hand poses and prevent unrealistic interpenetrations in the synthetic hand motion data, we implemented a Collision Ratio-Guided Anti-Penetration Algorithm. This algorithm minimizes self-collisions in the hand mesh by iteratively optimizing MANO’s pose parameters while preserving the natural biomechanical constraints of the hand.

The algorithm uses two key measures:

- Signed Distance Function (SDF): the penetration depth, D_{sdf} , for the hand mesh.
- Collision Ratio (R_{col}): the proportion of a submesh involved in self-collision.

The steps are outlined in Algorithm 1.

Fig. 2 provides an example of a resolved self-collision, demonstrating how the algorithm enforces realistic hand configurations by eliminating mesh inter-penetrations. In Fig. 3, we show a comparison of anti-penetration methods.

Algorithm 1 MANO Collision Ratio-Guided Anti-Penetration Algorithm

Require: θ, β \triangleright MANO pose and shape parameters

- 1: Compute $D_{\text{sdf}}, R_{\text{col}}$ for hand mesh
- 2: **if** $D_{\text{sdf}} > \delta$ **then**
- 3: **for** $f \in \{\text{fingers}\}$ **do**
- 4: $R_{\text{col}}^i = V(\text{intersect}(M_i))/V(M_i)$ \triangleright Compute collision ratio for submesh
- 5: **end for**
- 6: $f^* \leftarrow \arg \max_i R_{\text{col}}^f$
- 7: **repeat**
- 8: $\theta \leftarrow \text{optimize}(\theta, \text{minimize } R_{\text{col}}^{f^*})$ \triangleright Minimize collision depth and ratio for selected submesh
- 9: **until** $R_{\text{col}}^{f^*} \leq \epsilon$ and $D_{\text{sdf}} \leq \delta$
- 10: **end if**
- 11: **return** θ

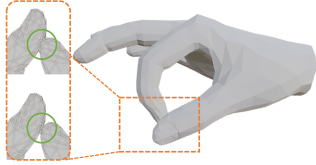


Figure 2. An example of resolving self-collision.

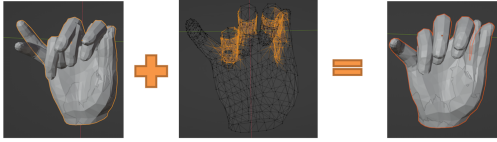


Figure 3. Illustrations of the collision guided anti-penetration. This figure presents a comparative analysis of anti-penetration optimization methods. On the left, we illustrate the traditional method’s before-and-after results. The center displays a collision map used in our approach. On the right, we demonstrate our method’s before-and-after results. Note that while both methods effectively resolve the collision, our method results in fewer alterations to the original configuration.

2.4. Camera Setup

We simulate real-world camera configurations, as outlined in Table 3, including regular cameras with varying focal lengths, XR headsets, and smartphone cameras. Each camera type reflects common devices used in real-world applications, ensuring the dataset is applicable to a wide range of use cases, including XR, HGR, and egocentric interaction studies.

To achieve diverse and comprehensive data, our camera setup, illustrated in Fig. 4, follows a hemispherical arrangement. Cameras are positioned around a hemisphere, with configurations including both static cameras (fixed positions) and dynamic cameras (tracking the hand’s center).

This approach captures hand motions from multiple angles and distances, enabling robust data suitable for training and testing algorithms across various tasks.

Not all 3D hand objects are rendered with every camera. Instead, specific cameras are selected for different components of our dataset as shown in Table 4.

2.5. Dynamic Arm Integration

Below we show the “Cut-and-Stitch” method we designed to dynamically adjust and attach a template arm to hands in any poses of hand shapes.

We identify the wrist boundary vertices of the NIMBLE mesh V_w^{NIM} , and ensure that the up-sampled SMPL-X arm has the same number of vertices V_w^{ARM} :

$$|V_w^{ARM}| = |V_w^{NIM}| \quad (8)$$

The integration process is designed to maintain flexible wrist rotation. Let R_w be the wrist rotation matrix that allows for dynamic hand articulation, while keeping the arm’s attachment stable. We apply R_w to the NIMBLE hand, centered at the wrist cut-center C_w and stitch with the arm template:

$$V_{\text{stitch}} = R_w \cdot (V_w^{NIM} - C_w) + C_w + V^{SMP} \quad (9)$$

Finally, a global transformation, R_g, t_g , is applied to the entire hand-and-arm mesh:

$$V^{\text{final}} = R_g \cdot V^{\text{stitch}} + t_g \quad (10)$$

The “Cut-and-Stitch” method removes overlapping faces at the wrist and blends UVs for consistent skin texture. UV mapping at the wrist is adjusted by interpolating between the NIMBLE hand and SMPL-X arm to prevent visual seams, while NIMBLE UVs are propagated to the arm to ensure uniform skin tone and maintain realism.

3. Dataset Specifications

This section details the specifications of each subset in the dataset generated from different base dataset.

3.1. Annotation Structure

Below, we detail the structure of the available annotations in our dataset.

Overall Camera Specifications. Internal properties of each camera, such as sensor dimensions, focal length, and intrinsic parameters.

Listing 1. Camera Specifications

```
{
  "camera": str, // Camera name
  "sensor_width": float, // Sensor width
```

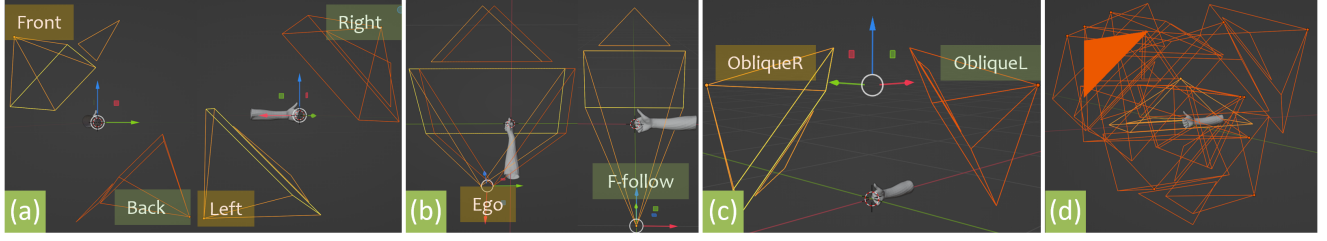


Figure 4. Camera setup showing (a) near static cameras, (b) dynamic cameras, (c) far static cameras, and (d) combined view of all cameras.

Table 3. Camera Setup and Specifications

Camera/Lens	Type	Position	Focal length	Sensor (W x H, mm)	Image Size	Aperture (f)	Special Features
Sony FE 35mm Full-Frame	Static	Front, Back, Left, Right	35mm	36.0×24.0	1200×800	f/1.8	Minimal distortion, balanced perspective
Canon EF 11-24mm f/4L USM Ultra-Wide	Static	Front-Right, Front-Left	18mm	22.3×14.9	1200×800	f/2.8	Wide field of view for diagonal angles
Samsung Galaxy S24 Ultra-Wide	Static	Front-Right, Front-Left	Ultra-Wide (13mm)	6.17×4.55	3200 x 2400	f/2.2	Ego perspective, follows hand motion
Sony FE 85mm	Dynamic	Palm Side	85mm	36.0×24.0	1200×800	f/1.4	Close-up on palm, high detail
Apple Vision Pro Stereo Cameras	Static/Dynamic	Back of Hand (Stereo Pair)	18mm	27.36×24.0	1200×800	f/2	Ego-view, binocular

Table 4. A detailed breakdown of components in the datasets.

Dataset	Base Dataset	# Classes	# Sequences/Class	# Frames/Sequence	# Cameras/Frame	# Images
SHREC-static	SHREC'22 static	6	200	1	8	9.6K
InterHand-static	InterHand2.6M (peak poses)	14	200	1	8	22.4K
InterHand-dynamic	InterHand2.6M (with transitions)	14	200	15	2	84K
SHREC-dynamic	SHREC'22 dynamic	10	200	60	1	120K
EgoGesture-dynamic	EgoGesture	49	100	30	2	294K
InterHand-pro	InterHand-static fingers × SHREC-dynamic wrist	140	20	30	2	168K
SHREC-pro	SHREC-static fingers × SHREC-dynamic wrist	60	20	30	3	108K
EgoGesture-pro	EgoGesture fingers × SHREC-dynamic wrist	490	10	30	2	294K

```

"sensor_height": float, //Sensor height
"lens": int, // Focal length (mm)
"fstop": float // Aperture size
}

```

Per-Image Annotations. Metadata and attributes for each image, including bounding boxes, image dimensions, and camera associations.

Listing 2. Image Metadata

```

{
  "capture_id": str,
  "images": [
    {
      "id": int, // Image ID
      "file_name": str, // File name
      "width": int, // Image width
      "height": int, // Image height
      "capture": int, // Capture ID
      "subject": int, // Subject ID
      "seq_name": str, // Sequence name
      "camera": str, // Camera name
      "sequence_id": int, // Sequence ID
      "bbox": [xmin, ymin, width, height]
    }
  ]
}

```

Camera Pose Annotations. Extrinsic parameters capturing the position and orientation of each camera relative to the scene.

Listing 3. Camera Information

```

{
  "capture_id": str,
  "camera_name": {
    "campos": [x, y, z], // Camera
    position (static or dynamic)
    "camrot": [3x3 matrix] // Camera
    rotation matrix
  }
}

```

3D Sequence Annotations. Frame-by-frame 3D information, including:

1) 3D Keypoints.

Listing 4. 3D Joint Annotations

```

{
  "capture_id": str,
  "frame_idx": str,

```

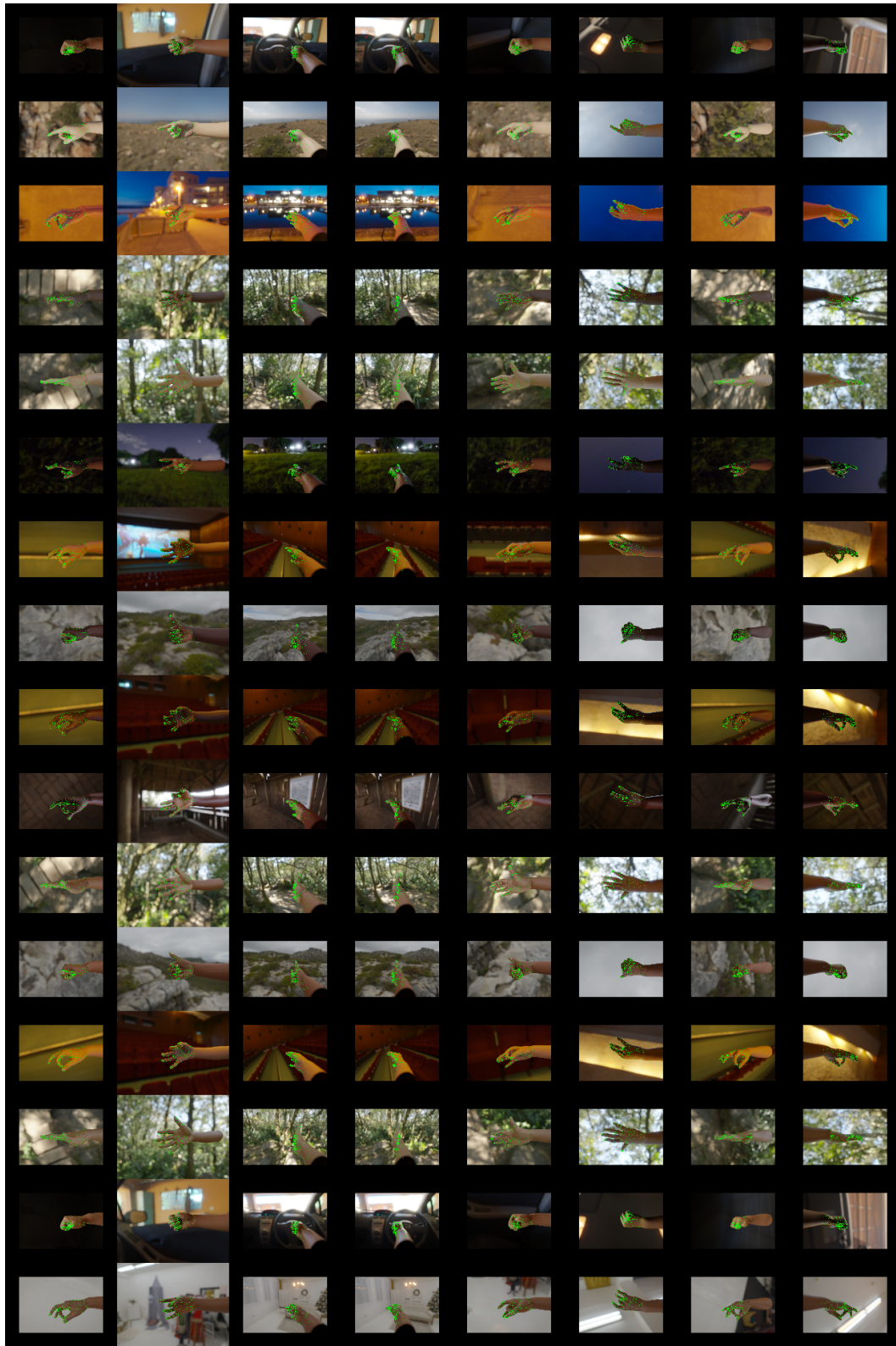



Figure 5. Examples of one hand shooting by eight cameras, annotated with MANO vertices. Each column shows images captured by one camera.

```

"j21_world": [[x, y, z], ...], // 21
    MANO joints derived from vertices in
    meters.
"j25_world": [[x, y, z], ...] // 25
    NIMBLE joints derived from skeleton
    in meters
}

```

2) 3D MANO Ground Truth.

Listing 5. MANO Ground Truth

```

{
  "capture_id": str,
  "frame_idx": str,
  "pose": [48 values], // MANO pose vector
  "shape": [10 values], // MANO shape
    vector
  "trans": [x, y, z] // Translation vector
    in meters
}

```

3) 3D Point Clouds.

Listing 6. 3D Point Cloud Annotations

```

{
  "capture_id": str,
  "frame_idx": str,
  "verts_world": [[x, y, z], ...] // Hand
    and arm vert coordinates in
    millimeters
}

```

A. Gesture Categories

Motions in our dataset are based on widely used gestures in XR applications. Specifically, we selected three popular datasets and processed them to generate “*-gen” subsets. Using the Cartesian product of hand poses and wrist motions, we further derived three “*-pro” subsets, resulting in a total of six subsets. Below, we list the gesture labels for each subset.

A.1. InterHand-gen

We selected 14 single-hand gestures from InterHand2.6M:

Listing 7. InterHand-gen Gesture Labels

```

{
  "good_luck": 0, "thumb_up": 1, "okay":
    2, "fist": 3,
  "one_count": 4, "two_count": 5, "
    three_count": 6,
  "four_count": 7, "five_count": 8, "
    index_tip": 9,
  "middle_tip": 10, "ring_tip": 11, "
    pinky_tip": 12, "capiisce": 13
}

```

A.2. SHREC-gen

SHREC-gen consists of static and dynamic gestures from SHREC’21 and SHREC’22:

Listing 8. SHREC-gen Gesture Labels

```

{
  "Static_Gestures": {
    "ONE": 0, "TWO": 1, "THREE": 2, "FOUR
      ": 3,
    "OK": 4, "MENU": 5
  },
  "Dynamic_Gestures": {
    "GRAB": 0, "PINCH": 1, "LEFT": 2, "
      RIGHT": 3,
    "CIRCLE": 4, "V": 5, "CROSS": 6, "
      DENY": 7,
    "KNOB": 9, "TAP": 10, "EXPAND": 11
  }
}

```

A.3. EgoGesture-gen

We selected only single-hand gestures from EgoGesture, preserving their original numbering:

Listing 9. EgoGesture-gen Gesture Labels

```

{
  1: "Wave_palm_towards_right", 2: "
    Wave_palm_towards_left",
  3: "Wave_palm_downward", 4: "
    Wave_palm_upward",
  5: "Wave_palm_forward", 6: "
    Wave_palm_backward",
  12: "Zoom_in_with_two_fingers", 13: "
    Zoom_out_with_two_fingers",
  14: "Rotate_fingers_clockwise", 15: "
    Rotate_fingers_counter-clockwise",
  16: "Click_with_index_finger", 21: "
    Static_fist",
  22: "Measure_(distance)", 24: "Number_0"
    , 25: "Number_1",
  26: "Number_2", 27: "Number_3", 28: "
    Number_4",
  29: "Number_5", 30: "Number_6", 31: "
    Number_7",
  32: "Number_8", 34: "OK", 35: "
    Another_number_3",
  37: "Shape_C", 38: "Make_a_phone_call",
    39: "Wave_hand",
  40: "Wave_finger", 41: "Knock", 42: "
    Beckon",
  43: "Palm_to_fist", 44: "Fist_to_Palm",
  46: "Trigger_with_index_finger", 48: "
    Grab_(bend_all_five_fingers)",
  50: "Gather_fingers", 51: "Snap_fingers"
    , 54: "Put_two_fingers_together",
  55: "Take_two_fingers_apart", 56: "
    Turn_over_palm",
}

```

```

63: "Thumb_upward", 64: "Thumb_downward"
,
65: "Thumb_towards_right", 66: "
Thumb_towards_left",
67: "Thumbs_backward", 68: "
Thumbs_forward",
74: "Bent_two_fingers", 75: "
Bent_three_fingers",
76: "Dual_fingers_heart", 77: "
Wave_finger_towards_left"
}

```

A.4. SHREC-pro

SHREC-pro extends SHREC-gen by applying the Cartesian product of 5 static gestures with 12 wrist motions, creating a diverse set of combinations:

Listing 10. SHREC-pro Gesture Labels (Partial)

```

{
  "ONE_GRAB": 0, "ONE_PINCH": 1, "ONE_LEFT
  ": 2, "ONE_RIGHT": 3,
  "ONE_CIRCLE": 4, "ONE_V": 5, "ONE_CROSS"
  : 6, "ONE_DENY": 7,
  "ONE_WAVE": 8, "ONE_KNOB": 9, "TWO_GRAB"
  : 10, "TWO_PINCH": 11,
  ...
  "MENU_GRAB": 50, "MENU_PINCH": 51, "
  MENU_LEFT": 52, "MENU_RIGHT": 53,
  "MENU_CIRCLE": 54, "MENU_V": 55, "
  MENU_CROSS": 56, "MENU_DENY": 57,
  "MENU_WAVE": 58, "MENU_KNOB": 59
}

```

A.5. InterHand-pro

InterHand-pro is derived from 14 single-hand gestures in InterHand2.6M, combined with 12 wrist motions, yielding 168 gesture classes:

Listing 11. InterHand-pro Gesture Labels (Partial)

```

{
  "good_luck_GRAB": 0, "good_luck_PINCH":
  1, "good_luck_LEFT": 2,
  "good_luck_RIGHT": 3, "good_luck_CIRCLE"
  : 4, "good_luck_V": 5,
  "good_luck_CROSS": 6, "good_luck_DENY":
  7, "good_luck_WAVE": 8,
  "good_luck_KNOB": 9, "thumb_up_GRAB":
  10, "thumb_up_PINCH": 11,
  ...
  "capisce_WAVE": 138, "capisce_KNOB": 139
}

```

A.6. EgoGesture-pro

EgoGesture-pro extends 49 single-hand gestures from EgoGesture by applying 12 wrist motions, creating 588 gesture variations:

Listing 12. EgoGesture-pro Gesture Labels (Partial)

```

{
  0: "Wave_palm_towards_right_GRAB", 1: "
Wave_palm_towards_right_PINCH",
  2: "Wave_palm_towards_right_LEFT", 3: "
Wave_palm_towards_right_RIGHT",
  4: "Wave_palm_towards_right_CIRCLE", 5:
  "Wave_palm_towards_right_V",
  6: "Wave_palm_towards_right_CROSS", 7: "
Wave_palm_towards_right_DENY",
  8: "Wave_palm_towards_right_WAVE", 9: "
Wave_palm_towards_right_KNOB",
  ...
  488: "Wave_finger_towards_left_WAVE",
  489: "Wave_finger_towards_left_KNOB"
}

```

References

- [1] Okan Köpüklü, Ahmet Gunduz, Neslihan Kose, and Gerhard Rigoll. Real-time hand gesture detection and classification using convolutional neural networks. *arXiv*, 2019. [1](#)
- [2] Joanna Materzynska, Guillaume Berger, Ingo Bax, and Roland Memisevic. The jester dataset: A large-scale video dataset of human gestures. In *IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 2874–2882, 2019. [1](#)
- [3] Menghe Zhang, Joonyeoup Kim, Yangwen Liang, Shuangquan Wang, and Kee-Bong Song. Unihands: Unifying various wild-collected keypoints for personalized hand reconstruction. 2024. [1](#)