THETA: Triangulated Hand Estimation for Teleoperation and Automation to Enhance Compliance in Robotic Hand Control using Multi-view Vision-based Segmentation and Deep Learning

SR-ROB-006
AzSEF 2025

All figures created by researchers unless otherwise specified

Introduction

Background

The global teleoperated robotics market is **projected to grow from $40.17 billion to $171.91 billion by 2032**, driven by rising automation demands in all sorts of fields **[1]**. **Teleoperation, the remote control of robots,** enables safe task execution in hazardous, inaccessible, or precision-critical areas, such as medical procedures, industrial operations, agricultural monitoring, or those with disabilities.

However, **high costs**, **complexity**, and **limited accessibility** of teleoperation technology restrict widespread adoption, highlighting the need for affordable, effective solutions to enhance robotic integration across industries.

Research Gap

- Current robotic teleoperation techniques rely heavily on costly **infrared depth cameras** and **embedded sensor** gloves.
- **Depth cameras** like Intel RealSense D455 ($350), Microsoft Azure Kinect ($400), and high-end systems such as Vicon (~$10,000+) significantly raise costs, limiting accessibility for the everyday user.
  - **Google MediaPipe**, a vision-based prediction system for joint angles using trigonometry and vector math, loses accuracy when the hand is curled, perpendicular, or flexed due to landmark occlusion.
- **Sensor gloves** like Manus Prime X ($5,000+) and SenseGlove Nova ($4,500) further increase expenses and complexity.

Existing methods lack a cost-effective, vision-based alternative capable of accurately estimating joint angles in real-time without expensive hardware or occlusion-prone hand tracking systems.

Figure 1: Existing teleoperation technologies: (1) Manus Prime X glove ($5000+), (2) Intel RealSense D455 camera ($350+), (3) Google MediaPipe (free software, single-camera based), (4) Vicon system (~$10,000+). Sourced from Shutterstock.

Objectives and Proposed Solution

We present **THETA[5]**, a **novel, cost-effective** method utilizing three **triangulated** inexpensive **webcams ($15 each)** for multi-view tracking to **estimate** relative **joint angles (θ)** in human fingers. Our approach integrates **DeepLabV3** for precise **hand segmentation** and **MobileNetV2** for robust **joint angle classification**, trained on a manually annotated dataset to enhance accuracy. These predictions are seamlessly transmitted to an **Arduino-controlled, low-cost (~$250), and open-sourced robotic hand**, enabling real-time, precise joint movement replication and significantly reducing system costs and complexity while maintaining **accurate**, **responsive** teleoperation.

Novelty and Advancements

- **Pioneered a novel webcam-based triangulation approach** for teleoperation, achieving high-precision joint angle estimation at a fraction of the cost ($45) compared to traditional infrared depth cameras and sensor gloves ($400+).
- **Introduced** a **first-of-its-kind 360° joint angle recognition system** using multi-view RGB input, eliminating the need for hands to remain parallel to a front-facing camera—overcoming the limitations of existing landmark-based and joint-location tracking methods for joint-angle recognition.
- **Restructured & optimized CNN** (MobileNetV2) layers to enhance joint angle detection rather than generic image classification tasks.
- **Engineered a low-cost, dexterous robotic hand (~$250)** to validate the effectiveness of THETA, setting a new benchmark for affordability and adaptability in teleoperation.

Experimental Design

1	Robotic Hand Development & ROS2 Control	Built a **servo-driven DexHand robotic hand** with modified hardware using 3D-printed parts, servos, and springs. **Developed ROS 2 software pipeline** for joint control and Arduino serial communication for hand actuation.
2	Multi-View Data Collection, Annotation & Segmentation	Collected **synchronized images of hand gestures** from multiple webcam angles; applied DeepLabV3 segmentation (ResNet-50 backbone) to isolate hands; **manually measured and annotated finger joint angles for supervised learning.**
3	Segmentation Preprocessing & THETA Joint Angle Classification	**Preprocessed segmented images** and trained a lightweight, efficient **MobileNetV2-based classifier** to accurately predict finger joint angles, optimizing performance using advanced deep-learning methods.
4	THETA Joint Angle Prediction & Real-Time Inference	**Evaluated THETA model performance**, achieving high accuracy and strong generalization across diverse conditions. **Implemented real-time inference w/ serial communication** for precise robotic hand actuation.

Methodology

1 DexHand Robotic Hand Design, Assembly, and ROS2 Control Integration

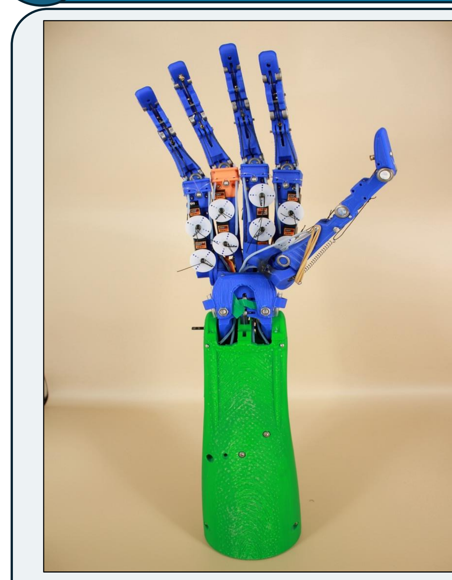


Figure 2: Constructed & Modified DexHand.

- 3D Hand CAD model and wrist mechanism open-sourced from **DexHand** by **The RobotStudio [2].**
  - Hand comprised entirely of **3D-prints, fishing line, bearings, springs, mini servos, & screws.**
- Phalanges, knuckle joint, and metacarpal bones **fastened w/ 80-lb fishing line** + 2mm spring.
- 3x Emax ES3352 12.4g mini servos (4.8-6V) and 1 spring **actuates fingers**
  - 2x servos for **abduction/adduction** and finger base **flexion**, 1x servo for **fingertip flexion.**
- 1 spring for **fingertip** (distal) and **base** (proximal) **extension.**

- **Ubuntu VM w/ USB passthrough** used as the environment for **ROS2**
- Arduino pipeline relied on **two main ROS2 nodes** to facilitate robotic hand movement [3].
  - **Gesture Controller Node**: Generates and manages high-level hand joint angles by publishing an array of 15 servo angles on the *dexhand_hw_command* Python topic.
    - Developed new function to modify servo angles dynamically
  - **USB Serial Node**: Acts as a **bridge** between **ROS 2** and the **Arduino**, converting high-level commands into **serial messages** that the Arduino Mega **interprets** to control the **servos** on the robotic hand.

Figure 3: Fingertip flexion by pulling on ligament. Spring (tip extension) circled.

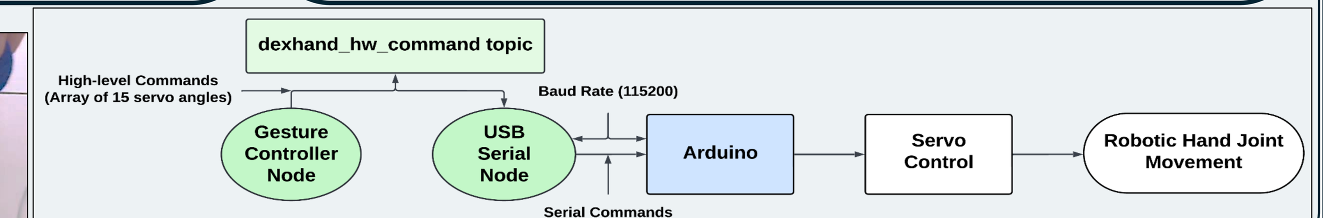


Figure 4: ROS2-Arduino Joint Angle Transmission pipeline for robotic hand servos actuation.

2 THETA Architectural Pipeline: Multi-View Data Collection, Annotation & Segmentation

A Standardized Gesture Dataset: Definition, Joint Angle Mapping, and Database Integration

Gesture Id	Gesture Name	Index MCP Angle	Index PIP Angle	Index DIP Angle	Middle MCP Angle
1	Closed Fist	90 (±5°)	90 (±5°)	110 (±5°)	90 (±5°)
2	Open Palm	180 (±5°)	180 (±5°)	180 (±5°)	180 (±5°)
3	Number One	180 (±5°)	180 (±5°)	180 (±5°)	90 (±5°)

Figure 5: Example entries from the "gesture joint angles" dataset, which defines 40 standardized hand gestures and maps their corresponding 15 joint angles (MCP, PIP, DIP) for each finger.

A **"Ground Truth gesture joint angles" dataset was manually created** by measuring 15 joint angles across 40 distinct hand gestures using a protractor. The angles of the three finger joints were recorded:

- **Metacarpophalangeal (MCP) joint:** flexion/extension, abduction/adduction at the knuckle
- **Proximal Interphalangeal (PIP) joint:** mid-finger bending
- **Distal Interphalangeal (DIP) joint:** fingertip actuation

B Multi-View RGB Data Collection for Hand Tracking

Figure 6: Joint Data Collection Diagram.

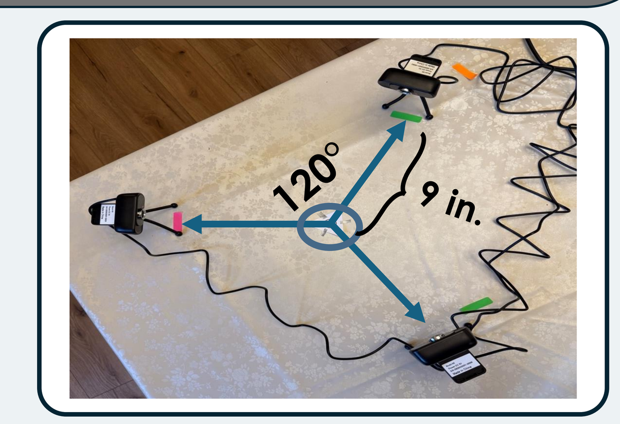


Figure 7: Triangulation Data Collection Setup.

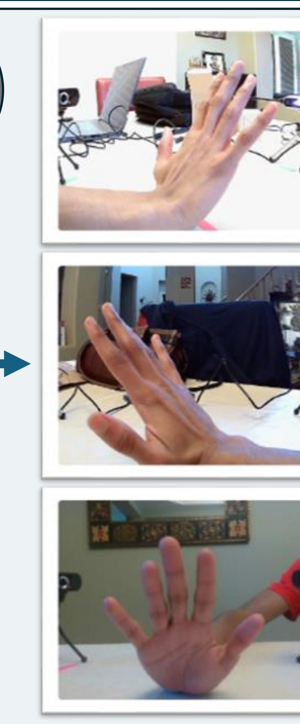


Figure 8: Synchronized RGB images captured from multiple viewpoints (front, left, and right).

Synchronized RGB images (640×480, 30 FPS) are captured from three webcam angles, front, right, and left, while performing the selected hand gesture. The corresponding joint angles are recorded, with a ±5-degree perturbation applied per frame to enhance variability and improve generalization.

C Hand Segmentation: Data Processing Pipeline and Mask Generation

Image Preprocessing
- Resize images to 224×224 for model input
- $I_{resized} \in \mathbb{R}^{224 \times 224 \times 3}$
- Normalize pixel values

$$\mu = \begin{bmatrix} 0.485 & 0.485 & \dots & 0.485 \\ 0.456 & 0.456 & \dots & 0.456 \\ 0.406 & 0.406 & \dots & 0.406 \end{bmatrix} \quad \sigma = \begin{bmatrix} 0.229 & 0.229 & \dots & 0.229 \\ 0.224 & 0.224 & \dots & 0.224 \\ 0.225 & 0.225 & \dots & 0.225 \end{bmatrix}$$

- Convert images to tensors

Feature Extraction
- Pass images through DeepLabV3 with ResNet-50 backbone **[3].**
- Apply Atrous Spatial Pyramid Pooling for multi-scale feature extraction

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Segmentation Prediction
- Optimize masks using BCEWithLogitsLoss
- Track accuracy using Mean IoU
- Convert soft masks into binary segmentation masks
- Apply erosion/dilation for noise removal and mask refinement

Post-Processing & Refinement
- Resize masks using nearest-neighbor interpolation
- Convert final segmentation masks to binary format
- Extract segmented hand regions for THETA model input

Figure 9: Feature extraction adjacency matrix: Nodes represent feature extraction stages, and edges (1s) denote transformations or feature extraction steps.

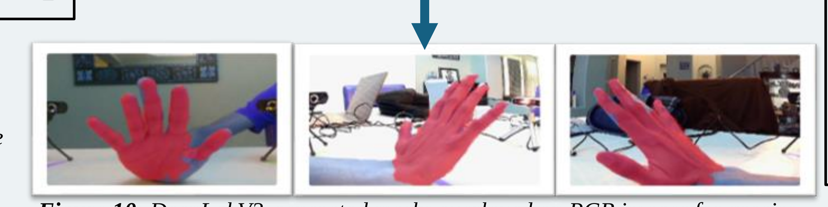


Figure 10: DeepLabV3-generated masks overlayed on RGB images for precise hand localization across different perspectives.

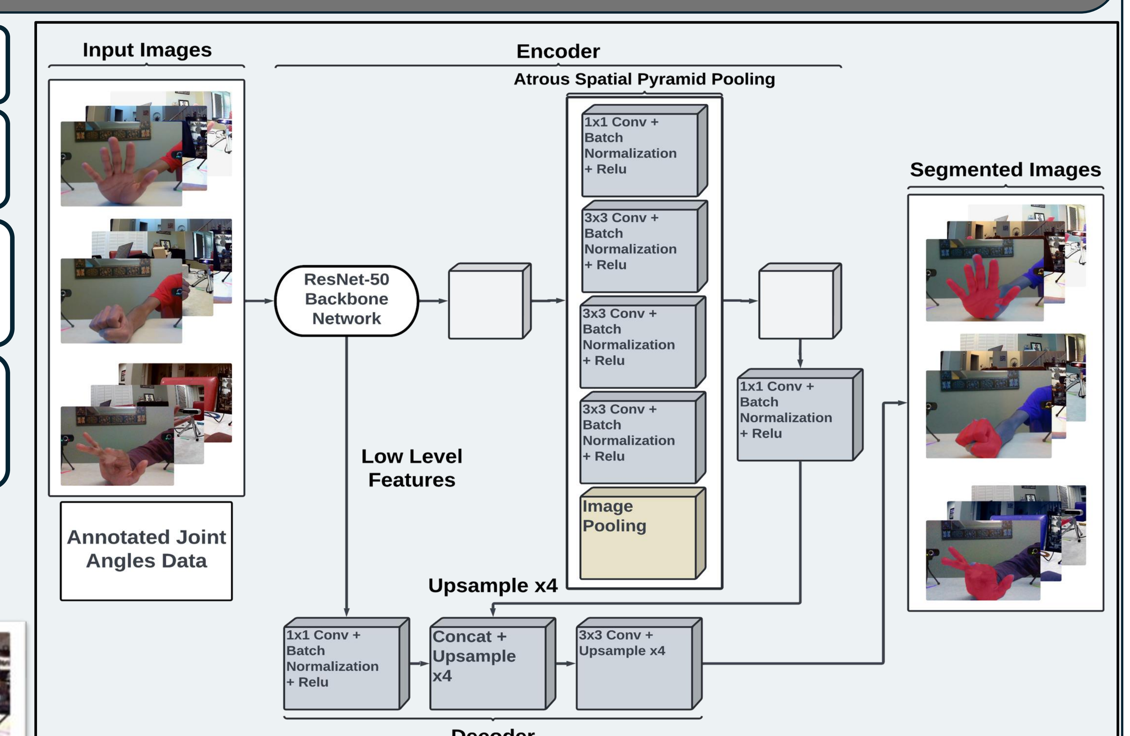
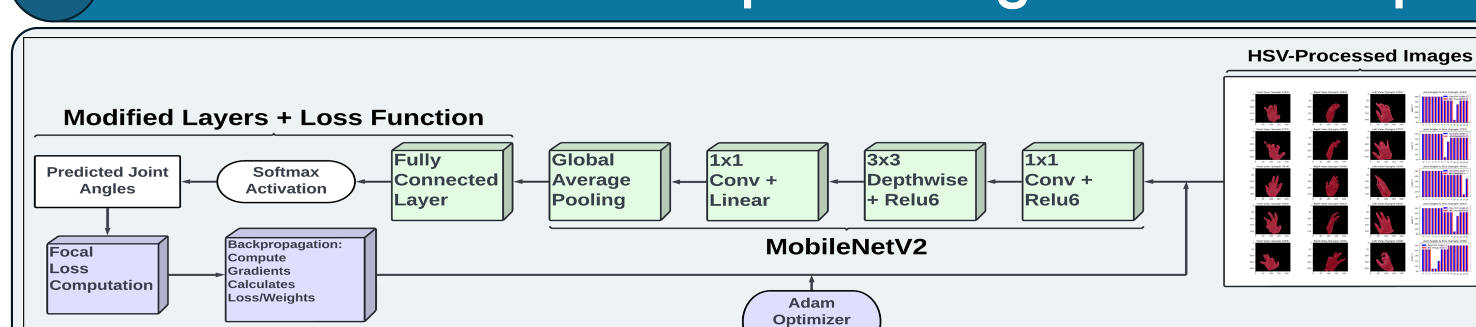


Figure 11: THETA End-to-End Hand Segmentation Pipeline: Multi-View RGB Image Segmentation Using DeepLabV3 for Image Preprocessing, Feature Extraction, Segmentation Prediction, and Mask Generation.

3 THETA Architectural Pipeline: Segmentation Preprocessing & Joint Angle Classification



H	S	V
0 – 10	120 – 255	70 – 255
170 – 180	120 – 255	70 – 255

Figure 13: HSV color thresholding matrices for hand segmentation. The predefined lower and upper red thresholds effectively isolate hand regions from the background.

- **HSV Segmentation:** Isolates hands via red color thresholds.
- **Data Processing:** Normalizes, resizes (224×224), and bins joint angles.
- **Feature Extraction:** MobileNetV2 classifies joint angles from multi-view images.
- **Optimization:** Focal Loss (γ=2.0), Adam (LR=0.001), 10-epoch training.
- **Evaluation:** Predicts joint angles, tracks accuracy, refines probabilities.
- **Modified Last Layer:** Reshapes to (batch, 15, 10), applies T=2.0 scaling, and softmax.

Figure 12: Multi-view RGB input undergoes HSV processing and MobileNetV2 joint angle classification for accurate, cost-effective robotic hand control.

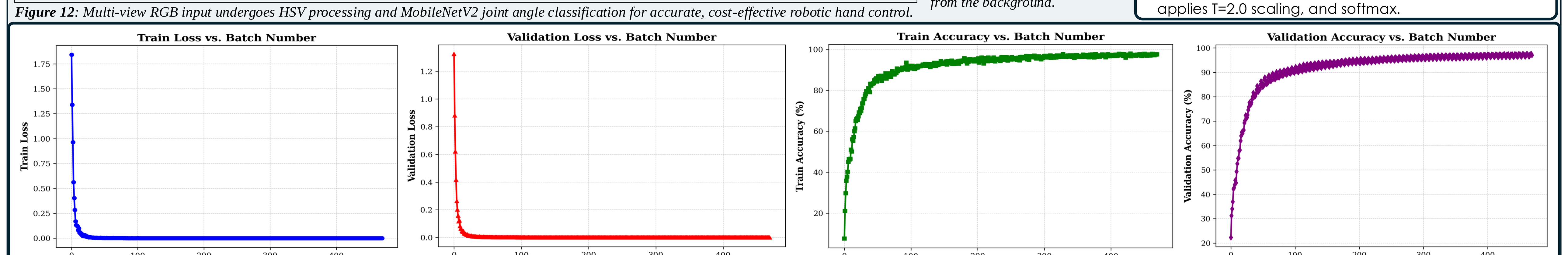


Figure 14: 97.50% training accuracy and 97.03% validation accuracy with loss convergence to 0.0001, demonstrating strong generalization, minimal overfitting, and reliable joint angle classification for real-time robotic teleoperation.

Conclusion

4 Joint Angle Prediction & Inference

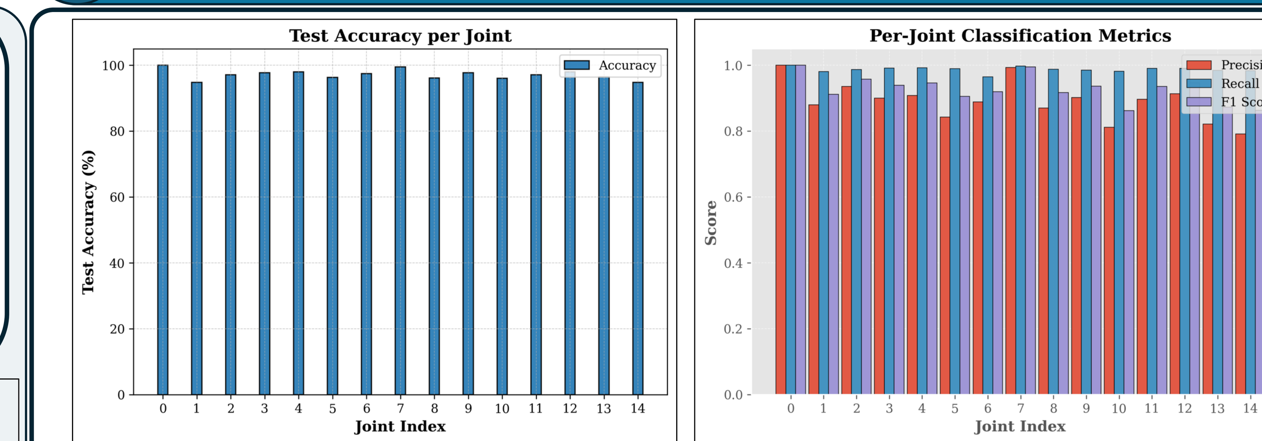


Figure 15: THETA achieves 97.18% accuracy, 0.9274 F1-score, 0.8906 precision, and 0.9872 recall in joint angle classification, ensuring precise hand pose estimation for robust motion analysis.

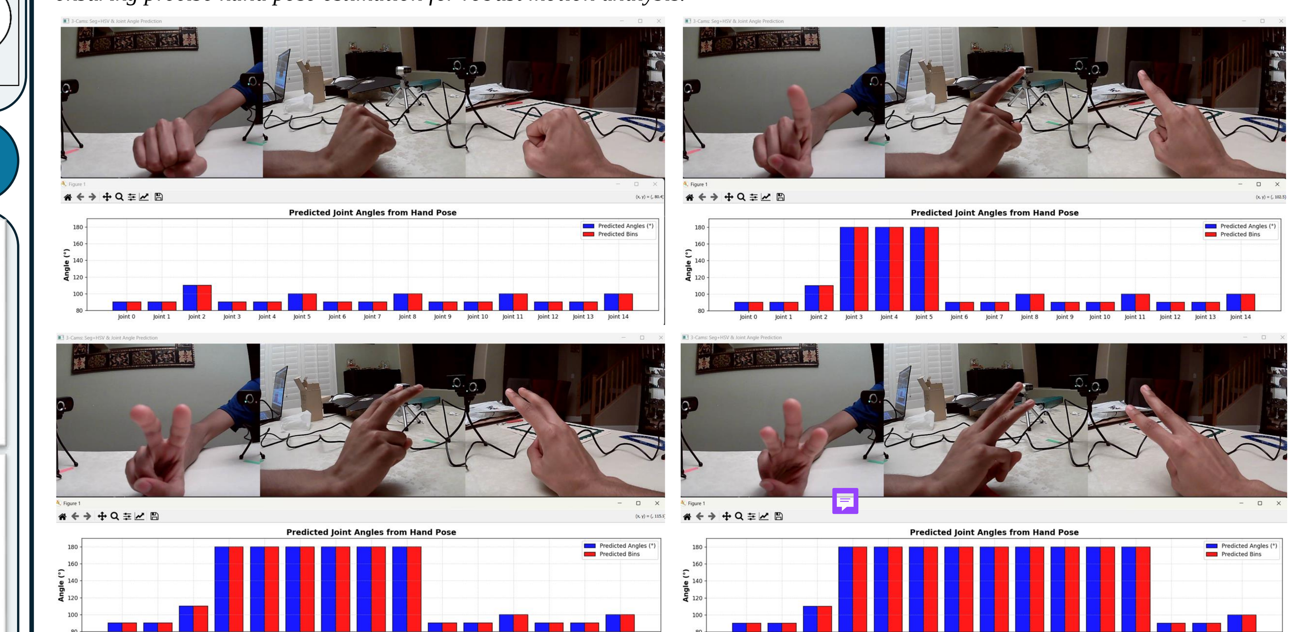


Figure 16: Real-time joint angle inference using THETA's multi-view triangulation for precise and responsive robotic hand control.

Result Evaluation

- THETA achieved **97.18% accuracy on the testing set**, demonstrating strong generalization in predicting joint angles.
  - The model attained an F1 score of **0.9274**, precision of **0.8906**, and recall of **0.9872**, ensuring **precise hand pose estimation** for robust motion analysis.
- In our project video, the THETA-DexHand pipeline successfully mimicked triangulated joint angles, validating real-world applicability.

Ultimately, THETA's simple setup and robustness has the potential to increase the accessibility of high-compliant teleoperated robotic hands, with implications for countless real-life fields.

Limitations

- **Despite having over 48,000 training images**, THETA's data sample size remains limited due to the slow and costly nature of training and computation on cloud GPUs.
- **THETA is not entirely accurate**, as it can sometimes misclassify certain joint angles, such as the joint angles distinguishing a peace sign with three fingers up.
- **As the dataset size increases, THETA can transition from a classification-based approach to a regression model**, enabling more precise and continuous joint-angle predictions

Future Research & Applications

Develop adaptive learning models that continuously refine and enhance joint angle recognition through **weighted user feedback.**

Integrate LLM reasoning, logic, and **image capabilities** to enhance compliance and awareness for situational contexts.

Optimize deep learning pipelines to minimize latency and boost real-time responsiveness of physical robotic hand.

Applications

- **Household Prosthetics:** Improve automation and AI functionalities in household prosthetics, especially for those with disabilities.
- **Medical Field:** Support remote surgical procedures with advanced gesture recognition technology.
- **Linguistics:** Facilitate remote or automated sign language interpretation and gestures.
- **Space Exploration:** Enable the manipulation of extraterrestrial objects during space missions.

References

1. "Robotics Market Size, Growth Trends, Report and Forecast 2022-2027." *www.imarcgroup.com*, www.imarcgroup.com/robotics-market.
2. Open, An. "DexHand - an Open Source Dexterous Humanoid Robot Hand." *DexHand - an Open Source Dexterous Humanoid Robot Hand*, 2023, www.dexhand.org.
3. iotdesignshop. "GitHub - Iotdesignshop/Dexhand_ros2_meta: Metapackage for DexHand ROS 2 Packages. Used to Manage the Collection of ROS 2 Packages and Environment for the DexHand Humanoid Robot Hand." *GitHub*, 2023, www.github.com/iotdesignshop/dexhand_ros2_meta.
4. guglielmocamporese. "GitHub - Guglielmocamporese/Hands-Segmentation-Pytorch: A Repo for Training and Finetuning Models for Hands Segmentation." *GitHub*, 4 Aug. 2022, www.github.com/guglielmocamporese/hands-segmentation-pytorch/tree/master.
5. smokyfishy. "GitHub - Smokyfishy/THETA." *GitHub*, 2025, github.com/smokyfishy/THETA.