Tom Hetto* and Markus Reischl

Automatic Labeling of Multi-Modal Sensor Training Data for Hand Gesture Analysis

https://doi.org/10.1515/cdbme-2025-0180

Abstract: Hand gesture recognition is an important task in human-machine interaction, enabling intuitive and accessible control methods across various applications, including assistive technologies, virtual reality or sign language translation. The gesture recognition task relies on sensory input from different modalities and in order to be processed, the data needs to be labeled with the corresponding hand gesture classes. This is often done manually which remains a major bottleneck in developing robust gesture recognition systems. This paper presents an Auto-Labeling Pipeline designed to automate the annotation process for multi-modal sensor data, reducing the time and effort required for labeling input data. The proposed system uses data from the Ultraleap Leap Motion Controller, and classifies predefined gesture models based on inter-joint angles and the cosine similarity between their pose vectors. The pipeline proposes a post-processing step to filter misclassifications and enhance gesture recognition reliability. The developed open-source toolbox enables researchers to collect and label gesture datasets efficiently, making hand gesture recognition more accessible and scalable.

Keywords: hand gesture recognition, automatic labeling, multi-modal sensor data, machine learning, human-computer interaction, gesture classification

1 Introduction

Hand gesture recognition is a natural form of human interaction and can be used in various ways if detected reliably. Human-Machine Interfaces (HMI) can rely on hand gestures, as they increase acceptance due to ease of use and accessibility [1] but are also used in many other domains such as sign language translation [2–4]. Due to the wide range of applications for hand gesture recognition, a variety of sensors are commonly used as input sources, such as electromyography [5], electrocardiography, other biosignals, camera images, gyroscope and accelerometer data or any other sensor. When performing a hand gesture, each sensor produces a time series input corresponding to this gesture. In order to process the collected data, it has to be labeled with the according gesture, as depicted in Figure 1. This labeling process consists of attributing the gesture performed at each time step to the set of input data, which can be a combination of multiple sensors. In some domains, researchers have started to generate synthetic data to eliminate the labeling step, but this has not yet been done for hand gestures [6, 7]. Thus, in order to get a high quality annotated dataset, manual labeling is often still the solution, e.g., time series need to be mapped to each other as well as to a performed gesture which serves as label in order to be usable as training data. It can be done manually by either recording a video of the test person performing the gesture and later-on labeling the data by hand, or by displaying instructions given to the user. The first solution is time-consuming while the latter is error prone if the user does not precisely times his gestures. Additionally, traditional data collection often introduces a predefined gesture sequence, in order to reduce the labeling efforts, which can introduce an unintended gesture dependency in the dataset. If in a predefined sequence, one gesture always follows another, motion artifacts can transition into the following gesture, and thus pollute the training data.

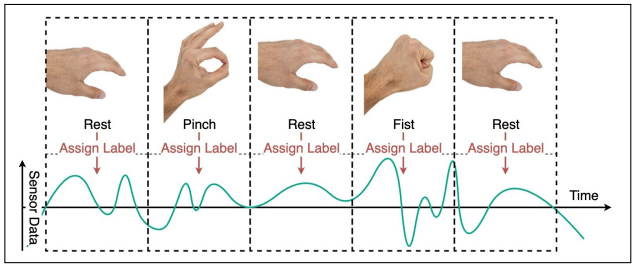


Fig. 1: Exemplary labeling process of a time-series of unspecified data

The goal of this article is to develop a solution to this labeling problem and make hand gesture recognition research more accessible to researchers who do not have the time to manually label their data. We present a data collection toolbox with an integrated Auto-Labeling Pipeline, which is made open-source and publicly available. First, it makes it easier to test and innovate on new multi-modal sensory input solutions by combining multiple sensor sources. Secondly, it facilitates

***Corresponding author: Tom Hetto,** HMT Engineering SARL, L-6143 Junglinster, Luxembourg & Institute for Automation and Applied Informatics, Karlsruhe Institute of Technology (KIT), D-76344 Karlsruhe, Germany, e-mail: tomhetto@gmail.com
Markus Reischl, Institute for Automation and Applied Informatics, Karlsruhe Institute of Technology (KIT), D-76344 Karlsruhe, Germany

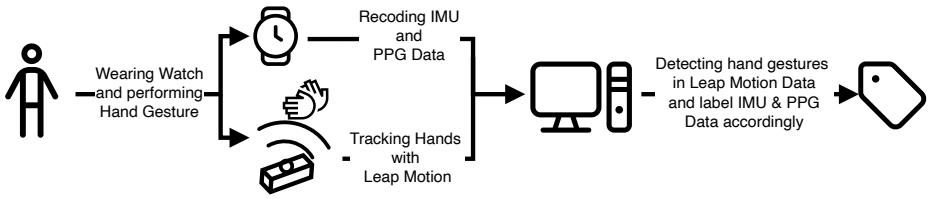


Fig. 2: Exemplary Auto-labeling pipeline used for labeling IMU and PPG training data.

further research on hand gesture detection by easing and automating the labeling process. Using the pipeline described in Section 3, it is possible to collect a dataset of hand gestures and the corresponding sensor data without tediously labeling the data manually. Third, it is easily extendable for other hand gestures and any modality of input. And finally the presented pipeline will also reduce errors due to motion artifacts pollution because no gesture sequence is predefined, and the data collection can be different for each subject and repetition, which is demonstrated in Section 4 where we will briefly evaluate the data collection process by training a neural network with two differently collected datasets.

2 Gestures and Classification

Hand gestures are defined by the position of each finger, and more specifically the angle of each joint in each finger. Thus, in order to classify gestures, calculating the angles of each joint gives a mathematical description of a hand gesture. The direction of the metacarpal and proximal phalanx as well as the middle and distal phalanx of the fingers are taken into consideration as they are providing the necessary positional information to determine the selected grasps. To classify gestures on the joint angles, an algorithm is proposed which computes joint angles between its corresponding bones and the palm's orientation to derive meaningful features for gesture classification. The angle between the direction vectors of the individual bones is calculated for each finger and these computed angles and pinch metrics are used to classify the hand pose into one of several predefined categories. Table 1 presents an exemplary list of gestures and their corresponding joint angles. To classify a newly performed gesture based on the prerecorded ones, a 14-dimensional feature vector is calculated based on the angles between the joints:

$$\mathbf{G_t} = \begin{pmatrix} t_{mc-pp} & t_{pp-dp} \end{pmatrix} \tag{1a}$$

$$\mathbf{G_i} = \begin{pmatrix} i_{mc-pp} & i_{pp-mp} & i_{md-dp} \end{pmatrix} \tag{1b}$$

$$\mathbf{G_m} = \begin{pmatrix} m_{mc-pp} & m_{pp-mp} & m_{md-dp} \end{pmatrix} \tag{1c}$$

$$\mathbf{G_r} = \begin{pmatrix} r_{mc-pp} & r_{pp-mp} & r_{md-dp} \end{pmatrix} \tag{1d}$$

$$\mathbf{G_p} = \begin{pmatrix} p_{mc-pp} & p_{pp-mp} & p_{md-dp} \end{pmatrix} \tag{1e}$$

$$\mathbf{G_{gesture}} = \begin{pmatrix} \mathbf{G_t} & \mathbf{G_i} & \mathbf{G_m} & \mathbf{G_r} & \mathbf{G_p} \end{pmatrix} \tag{1f}$$

Each finger has a joint angle vector, which contains the angles between the bones of the finger as described in Table 1. These vectors are then combined to a gesture vector $\mathbf{G_{gesture}}$ which is a 14-dimensional vector describing a specific gesture. The cosine similarity s of this vector is calculated towards the predefined gestures [8], and the most similar gesture is selected:

$$s = \cos(\theta) = \frac{\mathbf{G_{gestureA}} \cdot \mathbf{G_{gestureB}}}{\|\mathbf{G_{gestureA}}\| \|\mathbf{G_{gestureB}}\|} \tag{2}$$

Cosine similarity is often used in multi-dimensional vector spaces as it allows a fast calculation of a similarity measurement in complex search spaces [9, 10]. It is a value between 0 and 1 and in addition to selecting the most similar gesture, it also allows to discard gestures that are not part of the gesture set by setting a minimum similarity threshold. This similarity measurement is used in the following part to classify gestures.

3 Auto-Labeling Pipeline

The process of capturing input data from a multi-modal sensor source and automatically assigning the correct class to the time series is defined as Auto-Labeling Pipeline and visualized exemplarily in Figure 2. The auto-detection pipeline is constituted of a multi-step algorithm. It makes use of the angular relationships between the bones of the hand and the orientation of the palm to classify hand poses into predefined categories. This approach is designed to work with data provided by Ultraleap's Leap Motion Controller v1 (or v2) using two infrared cameras to track multiple hands [11]. The Leap Motion sensor provides detailed skeletal data for each hand, including the positions of joints and the orientation of the palm through its Gemini LeapC SDK written in C. For this, the Python Bindings are used in order to integrate the hand tracking data into our auto-labeling pipeline [12]. Additionally, Leap Motion directly provides a pinch strength value from 0 to 1, which indi-

	T MC-PP	T PP-DP	I MC-PP	I PP-MP	I MP-DP	M MC-PP	M PP-MP	M MP-DP	R MC-PP	R PP-MP	R MP-DP	P MC-PP	P PP-MP	P MP-DP
Fist	30°	30°	70°	85°	35°	80°	80°	40°	80°	85°	40°	80°	85°	40
Pinch	20°	20°	50°	55°	30°	25°	30°	20°	20°	20°	15°	10°	15°	10°
I. Tap	5°	5°	65°	65°	30°	15°	10°	5°	5°	5°	5°	20°	0°	5°
Rest	5°	5°	5°	10°	10°	10°	20°	15°	10°	20°	15°	15°	20°	15°

Tab. 1: Exemplary Inter-Joint Angle Gesture Mapping Table for each Finger Thumb(T), Index(I), Middle (M), Ring (R), Pinkie (P), with Metacarpal (MC), Proximal Phalanx (PP), Middle Phalanx (MP), Distal Phalanx (DP)

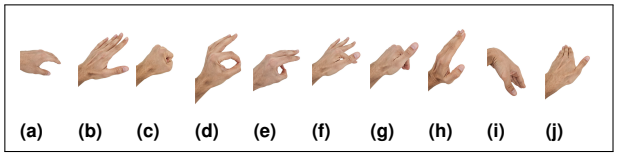


Fig. 3: Pre-implemented Gesture Set of 10 Gestures: Resting (a), Flat (b), Fist (c), Pinch (d), Pinkie Pinch (e), Index Tap (f), All Finger Tap (g), Wrist Flick Up (h), Wrist Flick Down (i), Writs Flick Out (j)

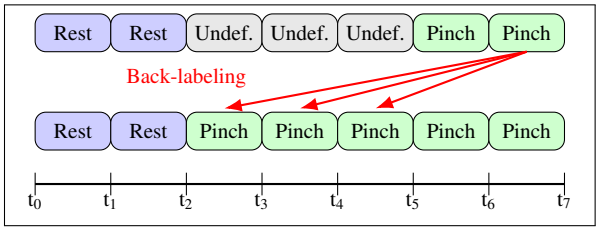


Fig. 4: Pre-Gesture Back-Labeling Post-Processing

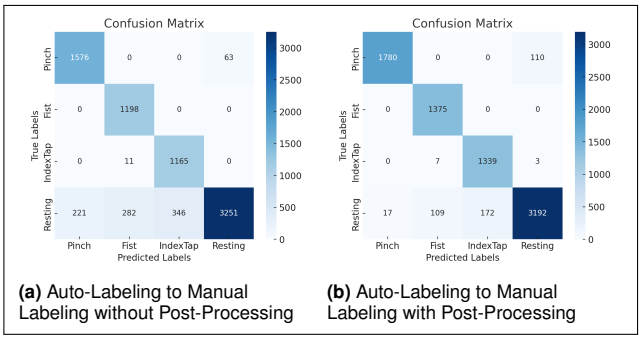
cates how strong the index and thumb are pinching, or if they are not pinching at all.

3.1 AutoGesture Toolbox

In addition to the sensory input chosen by the applying researcher, the pipeline makes use of infrared camera feeds provided by an Ultraleap Leap Motion Controller v1. Using two infrared cameras to track hands, the controller detects and calculates the position of the bones of each finger. For each gesture which needs to be detected a shape of angle ranges per joint as shown in Table 1 is defined. The pipeline receives the joint angles for each finger, compares these to the predefined shapes by calculating the cosine similarity according to Equation 2 and classifies the gesture accordingly. The detected gesture is then assigned as label for the collected raw sensor input. If no gesture is recognized, the data gets assigned a "No gesture" label in order to have training data for this case as well. To implement this pipeline for data labeling, a toolbox called AutoGesture is put at disposal [13] and consists of a collection of Python scripts. By using AutoGesture, researchers define their incoming data streams (e.g. IMU data) and they provide a defined list of hand gestures and their corresponding finger positions. The toolbox comes with 10 predefined gestures shown in Figure 3, but can easily be adjusted or extended by any other needed gesture with the provided Gesture Recording Tool. With this information, a pipeline is put in place which allows the researcher to start collecting data, which is then automatically labeled. The toolbox allows to record the hands joint angle data inside Python and further process either the raw data or the classified gesture labels. Additionally, the frames of the raw hand tracking data are saved to be able to verify the classifications.

3.1.1 Post-Processing

Due to the high framerate of the Leap Motion Device (up to 200 frames per second) it can happen that our pipeline falsely detects intermediate positions as gestures while the fingers are still moving in order to reach their final position and classification is performed at each frame. To filter these misclassifications, a post-processing procedure has been developed to achieve robust and responsive classifications. In a sliding window of a length of 100ms with a step-size of one frame, a majority voting is performed to determine the most probable gesture inside the selected window. This filters most of the unwanted misdetections due to intermediate positions. The pipeline only detects gestures once the fingers are in their final predefined position. For some applications or sensory inputs, the transition motion itself can greatly influence the sensor data and contain the most important information. To account for this, the post-processing pipeline implements a pre-window smoothing which is of variable length. Once a transition from "Resting" is starting, an undefined state is detected. As soon as a gesture has been recognized, the previous frames with undefined state are back-labeled with the respective gesture. Thus, the whole motion from "Resting" to the finished position gets labeled accordingly. This process is visualized in



(a) Auto-Labeling to Manual Labeling without Post-Processing

(b) Auto-Labeling to Manual Labeling with Post-Processing

Fig. 5: Hands detected by Leap Motion Controller and the corresponding Bone Representation

Figure 4 and can be disabled in the toolbox if not needed for the specific use case.

4 Results and Conclusion

The pipeline described in this study provides an efficient and scalable method for collecting a large dataset of hand gestures and corresponding sensory data without tediously labeling the data manually, which is often time-consuming and error-prone. The toolbox is easily extendable for other hand gestures and sensory input which ensures that the system can be adapted to various research needs and applied to a range of applications. In order to demonstrate the effectiveness of the pipeline, we recorded and auto-labeled a series of gestures and manually labeled the same dataset. By comparing the raw labeling of the pipeline to the manual labels, an accuracy of 89 % is achieved. The confusion matrix is displayed in Figure 5a. It shows that many misclassifications happen in the Resting class, which demonstrates the problem described in Section 3.1.1. By applying the there described post-processing, the accuracy increases to 95% and eliminates most of these misclassifications as shown in Figure 5b. These accuracy values are measured towards manual labeling, which in itself is not error-free. It can be assumed that the ground-truth is somewhere in between the manual labels and the automatic ones. Thus we can conclude that the Auto-Labeling Pipeline simplifies and accelerates the process of collecting large datasets of hand gestures and corresponding sensor data. Overall, this work contributes to the advancement of hand gesture recognition by making the data collection and labeling process fast and less error-prone. The ease of extending the pipeline to different gestures and sensory modalities opens new possibilities for future research, enabling further advancements in multimodal sensory input solutions for human-computer interaction, virtual reality, assistive technologies, and other research fields.

In future work, one can analyze the ability of removing the gesture dependency of predefined gesture sequences as detailed in Section 1 with the here presented pipeline.

References

[1] S. Adelé and E. Brangier, “Evolutions in the human technology relationship: rejection, acceptance and technosymbiosis,” *IADIS International Journal on www/Internet*, vol. 11, pp. 46–60, Jan. 2013.

[2] T. Zhao, J. Liu, Y. Wang, H. Liu, and Y. Chen, “Towards Low-Cost Sign Language Gesture Recognition Leveraging Wearables,” *IEEE Transactions on Mobile Computing*, vol. 20, pp. 1685–1701, Apr. 2021. Conference Name: IEEE Transactions on Mobile Computing.

[3] J. Hou, X.-Y. Li, P. Zhu, Z. Wang, Y. Wang, J. Qian, and P. Yang, “SignSpeaker: A Real-time, High-Precision SmartWatch-based Sign Language Translator,” in *The 25th Annual International Conference on Mobile Computing and Networking*, MobiCom ’19, (New York, NY, USA), pp. 1–15, Association for Computing Machinery, Aug. 2019.

[4] S. Jiang, P. Kang, X. Song, B. Lo, and P. Shull, “Emerging Wearable Interfaces and Algorithms for Hand Gesture Recognition: A Survey,” *IEEE Reviews in Biomedical Engineering*, vol. 15, pp. 85–102, 2022.

[5] O. Schill, R. Wiegand, B. Schmitz, R. Matthies, U. Eck, C. Pylatiuk, M. Reischl, S. Schulz, and R. Rupp, “OrthoJacket: an active FES-hybrid orthosis for the paralysed upper extremity,” vol. 56, pp. 35–44, Feb. 2011. Publisher: De Gruyter Section: Biomedical Engineering / Biomedizinische Technik.

[6] R. Bruch, F. Keller, M. Böhland, M. Vitacolonna, L. Klinger, R. Rudolf, and M. Reischl, “Synthesis of large scale 3D microscopic images of 3D cell cultures for training and benchmarking,” *PLOS ONE*, vol. 18, p. e0283828, Mar. 2023. Publisher: Public Library of Science.

[7] A. Orth, H. Höfer, A. Nefedov, M. Jalali, C. Wöll, and M. Reischl, “ML-Based XPS Quantification Supported by Synthetic Dataset Generation,” *Current Directions in Biomedical Engineering*, vol. 10, pp. 482–485, Dec. 2024. Publisher: De Gruyter.

[8] “Clustering based on Cosine,” *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCE*, vol. 2, no. 3, 2012.

[9] A. R. Lahitani, A. E. Permanasari, and N. A. Setiawan, “Cosine similarity to determine similarity measure: Study case in online essay assessment,” in *2016 4th International Conference on Cyber and IT Service Management*, pp. 1–6, Apr. 2016.

[10] G. Sidorov, A. Gelbukh, H. Gómez-Adorno, and D. Pinto, “Soft Similarity and Soft Cosine Measure: Similarity of Features in Vector Space Model,” *Computación y Sistemas*, vol. 18, pp. 491–504, Sept. 2014. Publisher: Instituto Politécnico Nacional, Centro de Investigación en Computación.

[11] U. Limited, “Leap Motion Controller,” Mar. 2025.

[12] U. Limited, “Gemini LeapC Python Bindings.”

[13] T. Hetto, “AutoGesture: A data collection toolbox with an integrated Auto-Labeling Pipeline.”