PalmPilot: Drone Control using Live Hand Signal Detection

Chris Kay
Stanford University
kayro@stanford.edu

Matt Mahowald
Stanford University
mcm2018@stanford.edu

Carlos Hernandez
Stanford University
carlosh6@stanford.edu

Abstract

Hand gesture recognition offers a compelling avenue for intuitive and accessible human-computer interaction, particularly in scenarios where traditional controllers may be impractical. In this project, we investigate real-time hand gesture recognition as a mechanism for drone control, with an emphasis on enabling lightweight and responsive inference on edge devices. We present a real-time pipeline that (i) uses MediaPipe Hands to extract 21 three-dimensional landmarks from an RGB stream and (ii) classifies those 63-D landmark vectors with an ultra-lightweight two-layer MLP. We collected an 11-gesture, 8,250-frame dataset and split it 70/15/15. The "MLP-Small" (35k parameters) attains 89.3% test accuracy at >700 FPS on a desktop CPU; the "MLP-Large" variant (130k parameters) pushes accuracy to 93.1%. All results in this milestone are obtained on a PC; on-device deployment and temporal modeling will be tackled during the remaining three weeks. We describe the data-capture setup, training procedure, preliminary results, and next steps toward mobile inference and live drone integration.

1. Introduction

Hand gesture recognition has emerged as a powerful modality for natural and non-verbal interaction with computing systems. Applications range from sign language interpretation and AR/VR interfaces to robot and drone control. This work focuses on the latter—developing a real-time, embedded-friendly hand gesture recognition system designed for drone operation. Compared to traditional physical controllers or voice commands, gesture-based input enables more fluid and intuitive control, especially in outdoor or hands-busy environments.

Our goal is to deliver a *fully on-device* pipeline that converts a single RGB frame into an actionable gesture label with strict constraints on latency (<15 ms), model size (<0.5 MB) and power.

1.1. Pipeline overview.

We adopt a two-stage architecture. *Stage 1* uses **MediaPipe Hands** [3] to locate the hand and output 21 three-dimensional landmarks; this strips away background clutter at negligible cost. *Stage 2* operates on the resulting 63-D vector. For the milestone we replace heavyweight CNNs with a pair of compact multilayer perceptrons. Both networks are trained with AdamW for 30 epochs on an 8,250-frame, 11-gesture dataset we recorded for this project.

2. Problem Statement

We aim to build a fast, compact, and accurate hand gesture classification system designed for drone control. The system should operate reliably in real-time using only RGB camera input, and be lightweight enough for deployment on edge devices in future iterations (e.g., Raspberry Pi or onboard drone hardware). The classification must be robust to varying hand orientations, lighting conditions, and backgrounds.

The input to our method is a single video frame containing a hand. We use the MediaPipe Hands pipeline [3] to extract 21 3D landmarks per frame, producing a 63-dimensional feature vector. This landmark vector is then passed to a neural classifier, which outputs one of 11 gesture labels corresponding to flight commands (e.g., "move forward," "rotate left," "hover," "land").

For this milestone, our classifier is a multilayer perceptron (MLP) trained on a custom 8,250-frame dataset across 11 gesture classes. We implemented two versions: a compact MLP-Small (35k parameters) and a larger MLP-Large (130k parameters). Both variants achieve strong test accuracy and run at over 600 FPS on desktop CPU, satisfying the latency and size constraints for real-time drone control.

2.1. Baseline and future work.

Previous gesture recognition pipelines often rely on CNNs or transformers trained on full-frame RGB input, which are too heavy for embedded deployment. Our baseline discards raw pixels after landmark extraction and classifies directly from geometry. For the final report, we plan

to introduce temporal modeling via 1D CNNs or GRUs and explore on-device deployment using TensorFlow Lite.

2.2. Related Work.

Our project draws on three core contributions from the literature:

- **MediaPipe Hands** [3]: This real-time hand-tracking framework achieves fast, accurate 2.5D landmark estimation from RGB input using a two-stage pipeline—palm detection followed by landmark regression. We use it to preprocess each frame, converting pixel input into a pose-invariant 63-D representation.

- **MobileNetV2** [1]: Designed for mobile efficiency, MobileNetV2 introduces inverted residual blocks with linear bottlenecks, achieving strong performance with minimal compute. While we do not use MobileNetV2 in the milestone model, its architecture informed our design goals for lightweight classification. We plan to incorporate it in a future frame-based classifier for temporal gesture modeling.

- **EfficientNet** [2]: This family of models uses compound scaling to balance width, depth, and resolution. While our MLPs do not yet use this approach, EfficientNet's scaling philosophy guides our plan to extend from per-frame MLPs to compact temporal CNNs in later phases of the project.

3. Dataset

We constructed a custom dataset tailored for drone control applications, comprising 11 distinct hand gesture classes corresponding to intuitive drone commands: `forward`, `backward`, `left`, `right`, `rotate_left`, `rotate_right`, `takeoff`, `land`, `hover`, `stop`, and `test`. The dataset was recorded using the MediaPipe Hands framework [3], which outputs 3D coordinates for 21 hand landmarks (63 dimensions per frame).

Each gesture was performed for 30 seconds across multiple recording sessions under varied lighting and background conditions to ensure robustness. The resulting dataset contains over 8,250 labeled examples.

To enable model training, we implemented a preprocessing pipeline that performs the following operations: (1) Splits raw gesture clips into individual frame-level examples. (2) Normalizes landmark coordinates. (3) Performs a 70/15/15 stratified split into train, validation, and test sets.

3.1. Limitations and Planned Data Collection

While the dataset supports initial model training and evaluation, it is not yet sufficient for final deployment. The current data was collected by two users in a relatively static indoor environment. As a result, the model may overfit to personal hand shape, gesture speed, or background context.

To improve generalizability, we plan to expand the dataset over the next three weeks by: (1) Recording additional users with varied hand sizes and skin tones. (2) Capturing data across multiple environments, including outdoor lighting conditions. (3) Introducing gesture variations such as different gesture speeds and orientations. (4) Collecting more samples for underrepresented classes (e.g., *land*, *test*).

4. Technical Approach

4.1. System Overview

Our hand gesture recognition system follows a two-stage pipeline optimized for real-time classification and compatibility with edge deployment:

Landmark Extraction. We use *MediaPipe Hands* [3] to extract 3D hand landmark positions from RGB video streams. For each frame, the detector produces 21 hand keypoints in 3D camera space. These are output as $\mathbb{R}^{63}$ vectors representing (x_i, y_i, z_i) for each joint i.

Gesture Classification. The normalized landmark vector is then classified by a compact multilayer perceptron (MLP), which maps input features to one of 11 command classes.

4.2. Normalization

To improve invariance to translation and scale, landmarks are centered and normalized. Specifically, we subtract the wrist coordinate from all points and divide by the maximum distance from wrist to fingertip joints. This ensures consistency across different users and camera setups.

4.3. Model Architecture

We implement two baseline classifiers: a small MLP with three hidden layers (63–128–64–32–11) and a larger variant with five hidden layers and increased capacity (63–256–128–64–32–11). Both use ReLU activations and dropout for regularization.

Each model is trained using a categorical cross-entropy loss:

$$\hat{p} = f(x) \in \Delta^{C-1}, \quad \mathcal{L}_{\text{CE}} = -\sum_{c=1}^{C} y_c \log \hat{p}_c$$

where $\hat{p}_c$ is the softmax probability of class c, and y_c is the one-hot ground truth.

4.4. Baseline and Training

To evaluate the tradeoff between speed and accuracy, we compare both MLP variants on our 11-class dataset. Both models are trained on a PC using the Adam optimizer

with learning rate 1×10^{-3}, batch size 64, and 30 epochs. Data augmentations such as random horizontal flipping and Gaussian noise are supported but not yet tuned.

4.5. Model Evaluation

Models are evaluated using accuracy and macro F1 score on the test split. Confusion matrices are generated to identify failure modes and inter-class confusion (see Figure 1 and Figure 2). MLP-Large outperforms MLP-Small, particularly on harder-to-distinguish gestures like `takeoff` and `land`, at the cost of increased parameter count.

4.6. Next Steps

We plan to expand the classification backbone by implementing a MobileNetV2-inspired inverted residual network [1], and experiment with EfficientNet-style compound scaling [2] to balance model depth, width, and resolution. Additionally, we aim to deploy the trained model to a Raspberry Pi using TensorFlow Lite for live drone demonstrations.

5. Intermediate/Preliminary Results

To assess model performance, we trained and evaluated two variants of our gesture classifier: a compact MLP ("MLP-Small") and a deeper MLP with additional hidden layers ("MLP-Large"). Both were trained on our custom 11-class hand gesture dataset using 8,250 landmark frames, split into training, validation, and test sets with a 70/15/15 ratio.

5.1. Quantitative Metrics

Evaluation on the test split yielded strong results for both models. MLP-Large achieved an accuracy of 93.1% and a macro-averaged F1 score of 0.915, while MLP-Small achieved 89.3% accuracy and an F1 score of 0.875. These metrics suggest that even our lightweight baseline is capable of reliable classification across a diverse gesture vocabulary.

Model	Accuracy	Macro F1 Score
MLP-Small	89.3%	0.875
MLP-Large	93.1%	0.915

Table 1. Performance metrics on the held-out test set.

5.2. Confusion Matrices

Figure 1 and Figure 2 show confusion matrices for the MLP-Small and MLP-Large models, respectively. Most gestures are classified with high precision and recall, particularly for "forward", "rotate_right", and "stop". However, "takeoff", "land", and "test" showed more frequent confusion with other dynamic gestures, which we attribute to limited training samples and inter-class visual similarity.

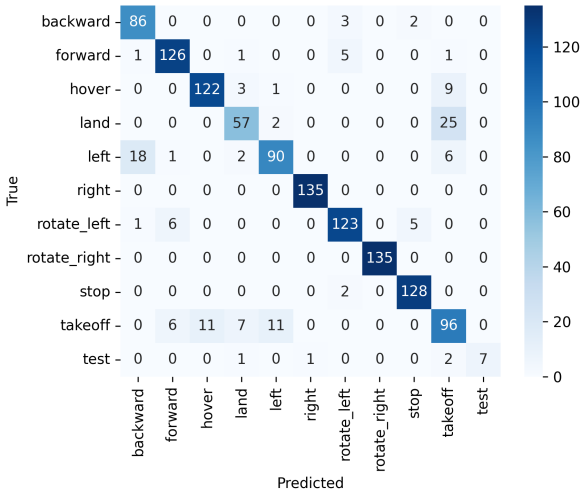

Figure 1. Confusion matrix for MLP-Small.

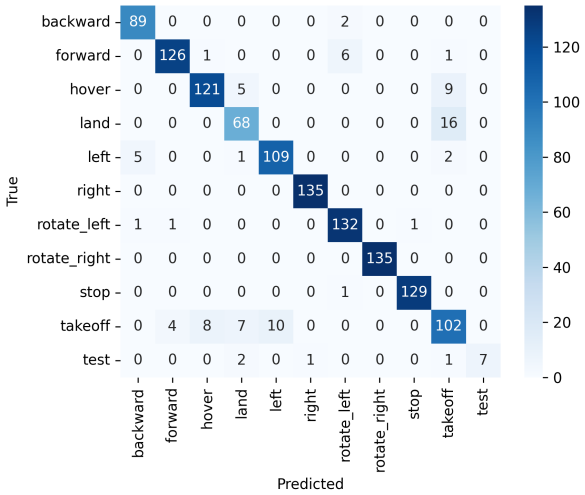

Figure 2. Confusion matrix for MLP-Large.

5.3. Failure Modes and Final Steps

While MLP-Large outperforms MLP-Small overall, it incurs a modest increase in parameter count and inference latency. Misclassifications predominantly occur between spatially adjacent gestures (e.g., "left" vs. "takeoff"), highlighting the need for finer temporal smoothing or gesture segmentation.

Next milestones: (1) Expand dataset diversity as outlined in Section 3.1. (2) Explore temporal modeling via GRUs or 1D CNNs. (3) Deploy the best model to a Raspberry Pi via TensorFlow Lite. (4) Demonstrate live drone gesture control in final presentation.

References

[1] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018.

[2] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, pages 6105–6114, 2019.

[3] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann. Mediapipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*, 2020.