

Segmentation and Key Frame Extraction for Analyzing Educational Research Videos

Group: DKHTNI

Members: Xin Cai, Boyuan Zou, Jonquil Liao

“**D**ucks **K**ee**P** **H**unting **T**iny **N**ymphs **I**ntently.”

“**D**atabases **K**ee**P** **H**uge **T**ables **N**eatly **I**ndexed.”

Project Background

The Need for Video Analysis Tools in Educational Research

- Educational Psychology Research
- Video Analysis
- Students' gestures as answering math problems
- No tools automatically support for this

Task: Identify specific type of gestures

- **Examples: *Representational gesture, (Non-)Dynamic gesture, etc.***
- **Highly abstract concepts in educational psychology**

Problem Statement

Developing an Intermediate Layer to Bridge Low-Level Data with High-Level Analytical Needs

High-level Needs (Identification of Abstract Gesture Type)

Intermediate Layer (Construction of Gesture Chunk) **Our Project**

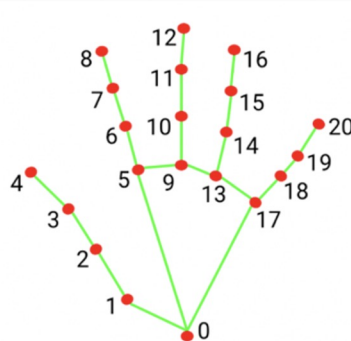
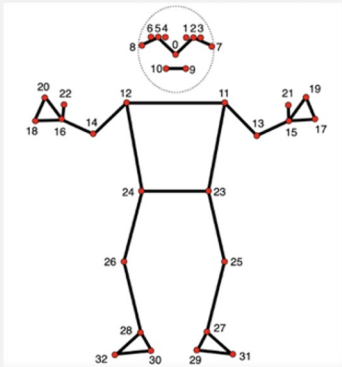
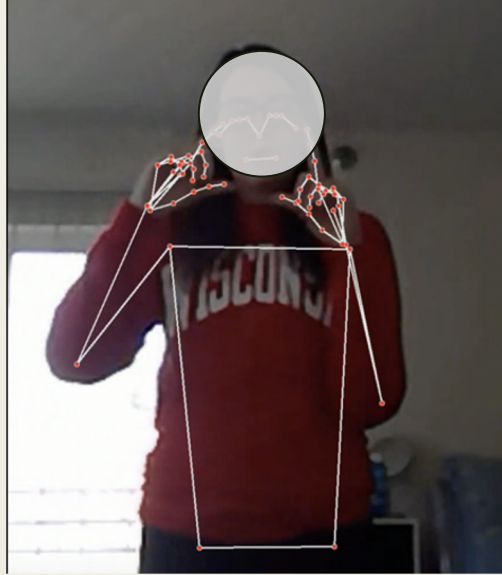
Low-level Data (Coordinates of Body/Hands Landmarks)

Data set

- 127 Video files (102 GB)
- Research in Embodied Geometry
- Magic Lab
- Department of Edu Psych

Tools

- OpenCV
- Mediapipe
- Stick figure model



- | | |
|-----------------------|-----------------------|
| 0. WRIST | 11. MIDDLE_FINGER_DIP |
| 1. THUMB_CMC | 12. MIDDLE_FINGER_TIP |
| 2. THUMB_MCP | 13. RING_FINGER_MCP |
| 3. THUMB_IP | 14. RING_FINGER_PIP |
| 4. THUMB_TIP | 15. RING_FINGER_DIP |
| 5. INDEX_FINGER_MCP | 16. RING_FINGER_TIP |
| 6. INDEX_FINGER_PIP | 17. PINKY_MCP |
| 7. INDEX_FINGER_DIP | 18. PINKY_PIP |
| 8. INDEX_FINGER_TIP | 19. PINKY_DIP |
| 9. MIDDLE_FINGER_MCP | 20. PINKY_TIP |
| 10. MIDDLE_FINGER_PIP | |

Low Level

Data:

Coordinates of
Body/Hands
Landmarks



High-Level Needs

```
# Unit of analysis
# Carrier of gesture-related features
struct Gesture Chunk {
    # Identify 'idle' and
    # 'active' sections
    segmentation: TODO

    # Recognition hand gesture
    # for each hand in each frame
    simple_hand_gestures: TODO

    # Extract keyframes to summarize
    # current chunk
    keyframes: TODO
}
```

Low-Level Data

**Intermediate
Layer:**
**Construction
of Gesture
Chunk**



Temporal Segmentation: Existing Algorithms

Papers

Key concepts

[1] Jiang, Feng, et al. "Multi-layered gesture recognition with Kinect." *J. Mach. Learn. Res.* 16.1 (2015): 227-254.



Quantity of Movement
+ thresholding

[2] Peng, Xiaojiang, et al. "Action and gesture temporal spotting with super vector representation." *European Conference on Computer Vision*. Cham: Springer International Publishing, 2014.



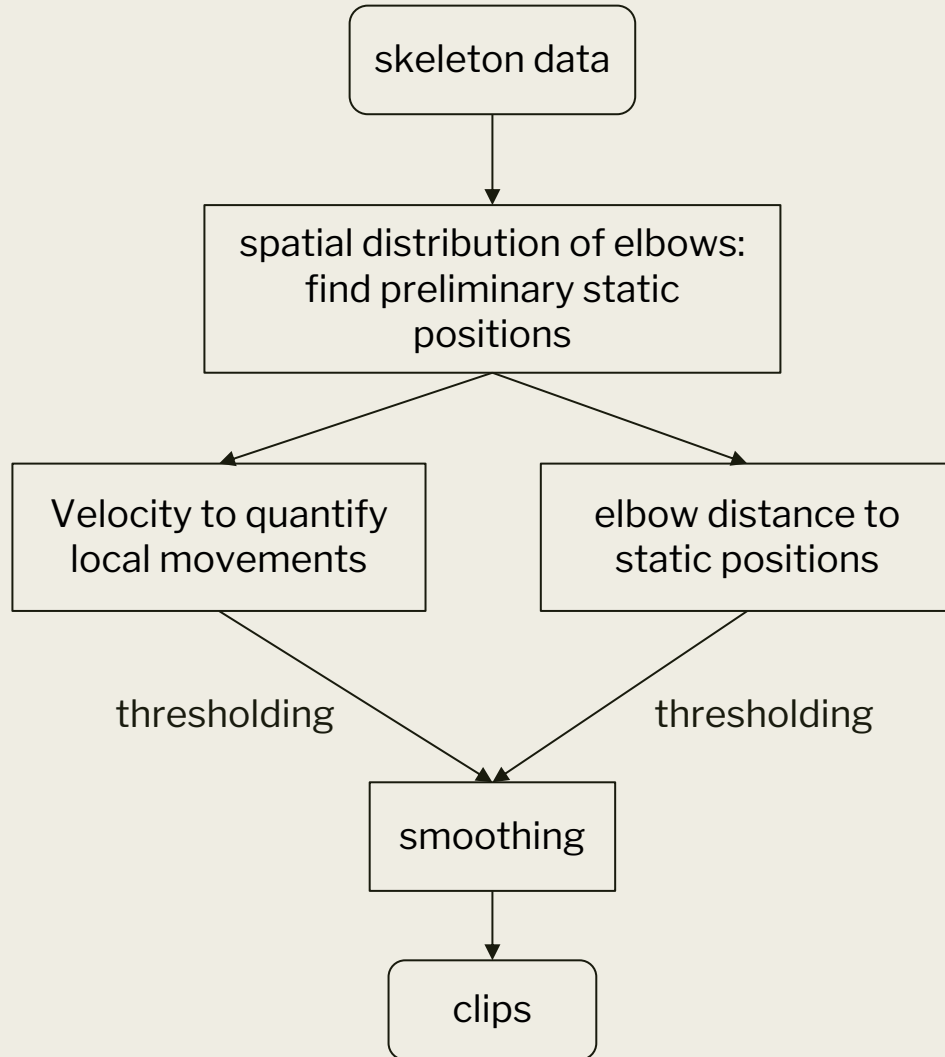
Spatial distribution of hand positions
+ distance of current hand position to
static position

[3] Madeo, Renata CB, Clodoaldo AM Lima, and Sarajane M. Peres. "Gesture unit segmentation using support vector machines: segmenting gestures from rest positions." *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. 2013.

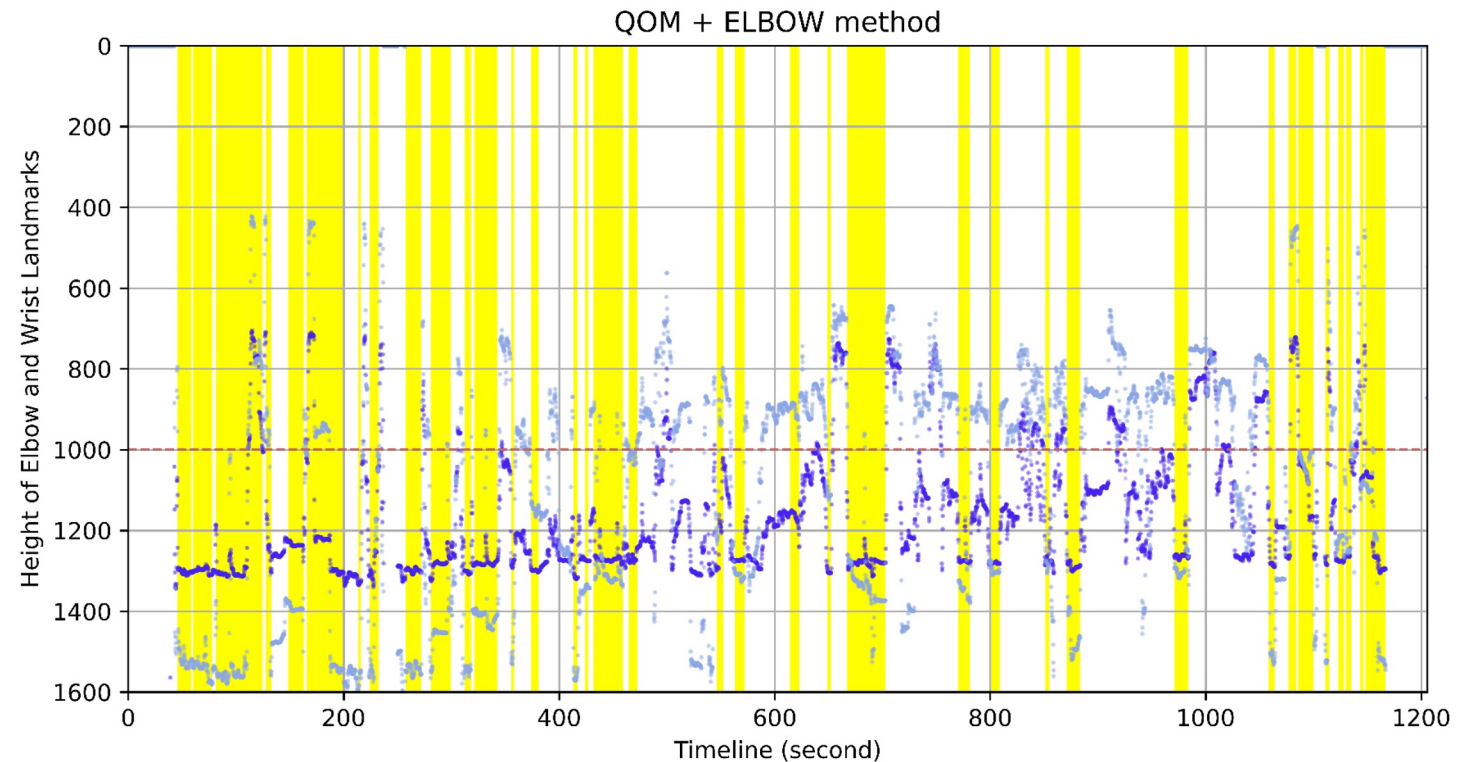
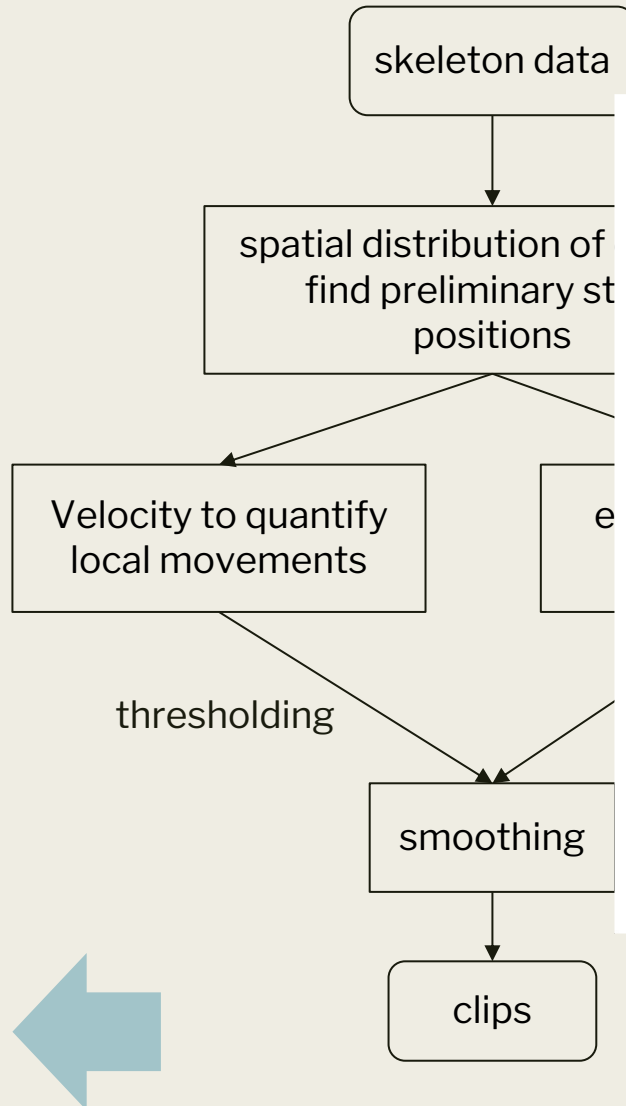


Velocity and acceleration as feature
vector + support vector machine
classification

Temporal Segmentation: Our Algorithms



Temporal Segmentation: Our Algorithms



Simple Hand-Gesture Detection

One important feature of active gesture chunk: Gesture of Hand

8 distinct categories:

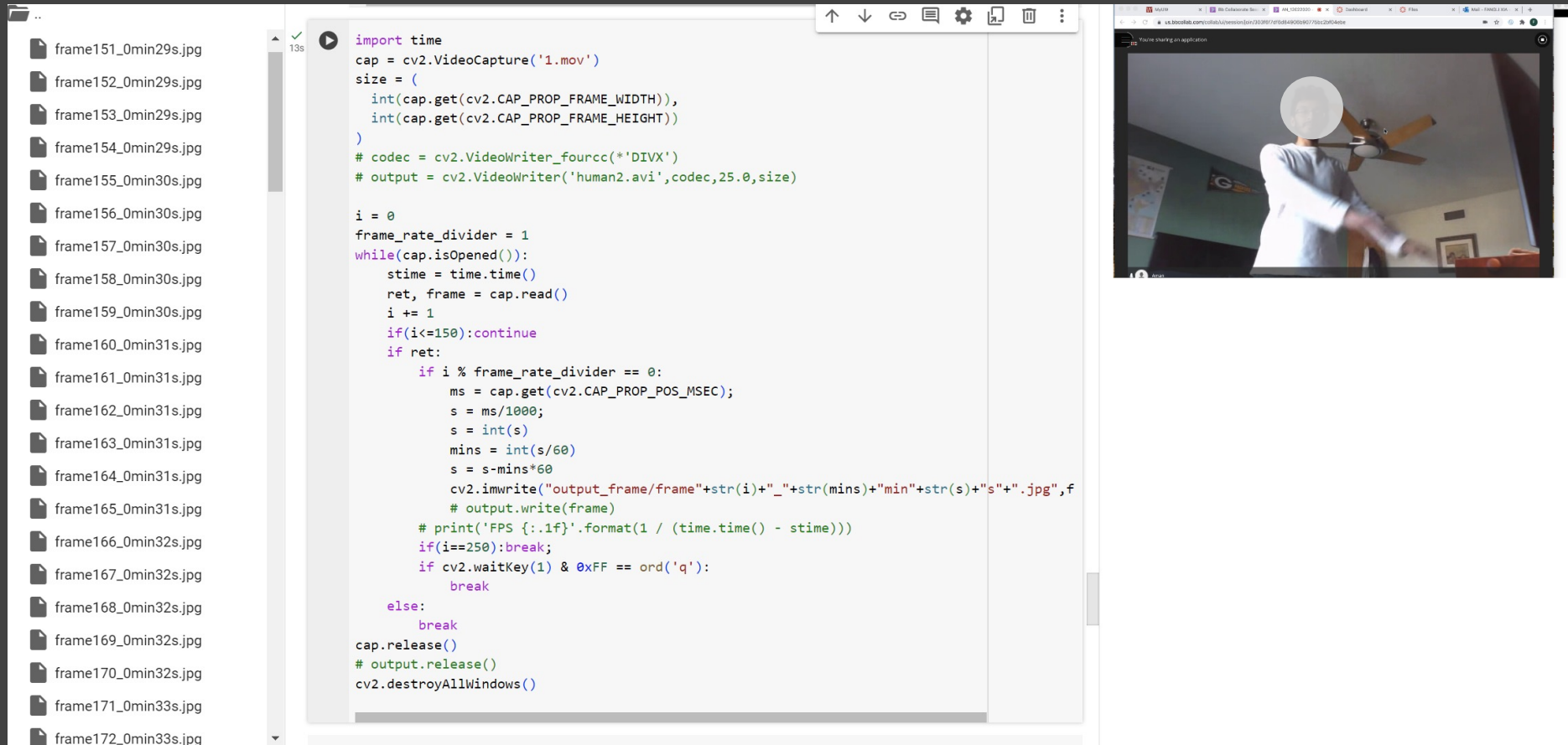
- Palm
- Fist
- Thumb Up
- Indexing
- L-shape
- Relax
- Other
- Missing



Other

Method:

1) Extract Frame



The image displays a workflow for video frame extraction. On the left, a file explorer shows a list of generated frames: frame151_0min29s.jpg through frame172_0min33s.jpg. In the center, a code editor shows a Python script that captures video from '1.mov', reads frames, and saves them as individual JPEG files. On the right, a video player shows the source video of a person in a white shirt, with a white circle highlighting the head area.

```
import time
cap = cv2.VideoCapture('1.mov')
size = (
    int(cap.get(cv2.CAP_PROP_FRAME_WIDTH)),
    int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
)
# codec = cv2.VideoWriter_fourcc(*'DIVX')
# output = cv2.VideoWriter('human2.avi', codec, 25.0, size)

i = 0
frame_rate_divider = 1
while(cap.isOpened()):
    stime = time.time()
    ret, frame = cap.read()
    i += 1
    if(i<=150):continue
    if ret:
        if i % frame_rate_divider == 0:
            ms = cap.get(cv2.CAP_PROP_POS_MSEC);
            s = ms/1000;
            s = int(s)
            mins = int(s/60)
            s = s-mins*60
            cv2.imwrite("output_frame/frame"+str(i)+"_"+str(mins)+"min"+str(s)+"s"+" .jpg", frame)
            # output.write(frame)
            # print('FPS {:.1f}'.format(1 / (time.time() - stime)))
            if(i==250):break;
            if cv2.waitKey(1) & 0xFF == ord('q'):
                break
        else:
            break
    cap.release()
    # output.release()
    cv2.destroyAllWindows()
```

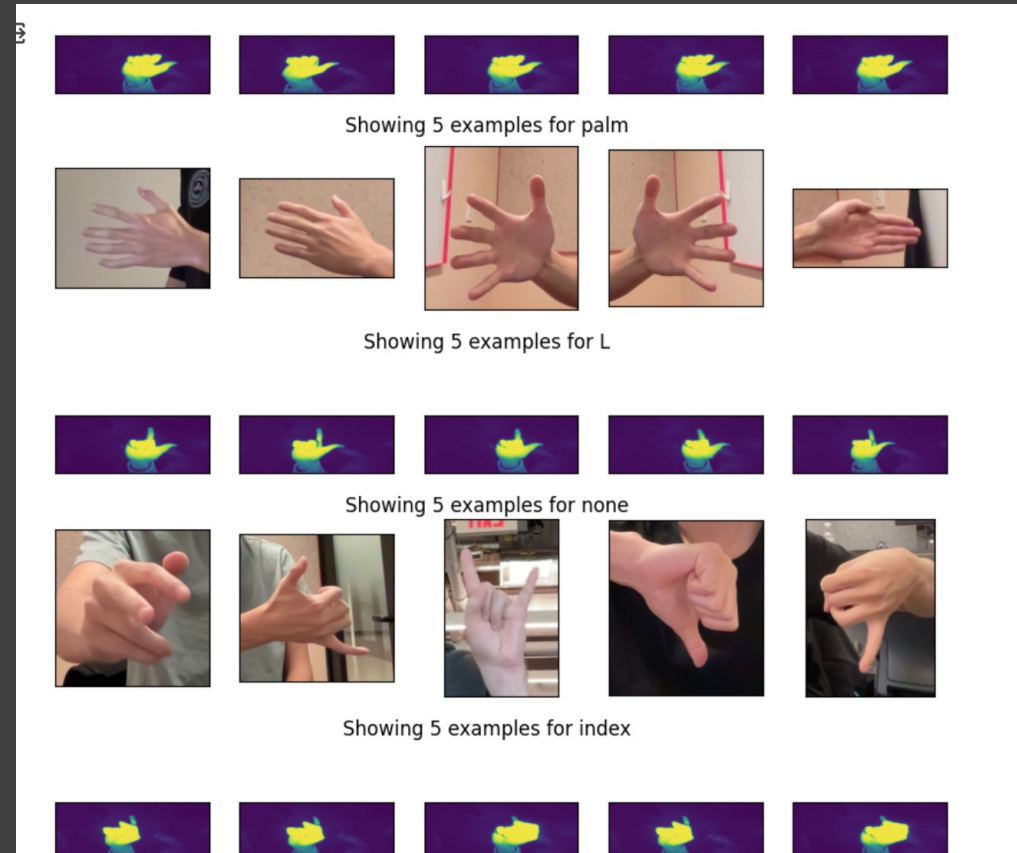
Simple Hand-Gesture Detection

2) Crop Out Hand using MediaPipe



Simple Hand-Gesture Detection

3) Train Gesture Classification
and apply: 80% for train 10%
for val, and 10 for test



Simple Hand-Gesture Detection

3) Train Gesture Classification and apply

```
# Train the model
hparams = gesture_recognizer.HParams(export_dir="rock_paper_scissors_model", batch_size=8,
    epochs=20,)
options = gesture_recognizer.GestureRecognizerOptions(hparams=hparams)
model = gesture_recognizer.GestureRecognizer.create(
    train_data=train_data,
    validation_data=validation_data,
    options=options
)
```

Model: "model"

Layer (type)	Output Shape	Param #
hand_embedding (InputLayer)	[(None, 128)]	0
batch_normalization (Batch Normalization)	(None, 128)	512
re_lu (ReLU)	(None, 128)	0
dropout (Dropout)	(None, 128)	0
custom_gesture_recognizer_out (Dense)	(None, 4)	516

=====
Total params: 1028 (4.02 KB)
Trainable params: 772 (3.02 KB)
Non-trainable params: 256 (1.00 KB)
=====
None
Epoch 1/20
47/47 [=====] - 4s 52ms/step - loss: 1.2161 - categorical_accuracy: 0.2420 - val_loss: 0.7115 - val_categorical_accuracy: 0.5
Epoch 2/20
47/47 [=====] - 1s 16ms/step - loss: 0.6147 - categorical_accuracy: 0.5452 - val_loss: 0.3462 - val_categorical_accuracy: 0.7
Epoch 3/20
47/47 [=====] - 1s 16ms/step - loss: 0.4304 - categorical_accuracy: 0.6941 - val_loss: 0.2212 - val_categorical_accuracy: 0.8
Epoch 4/20
47/47 [=====] - 1s 16ms/step - loss: 0.3449 - categorical_accuracy: 0.7420 - val_loss: 0.1632 - val_categorical_accuracy: 0.9
Epoch 5/20
47/47 [=====] - 1s 16ms/step - loss: 0.2725 - categorical_accuracy: 0.7872 - val_loss: 0.1332 - val_categorical_accuracy: 0.9

Simple Hand-Gesture Detection

3) Train Gesture Classification and apply

```
[ ] loss, acc = model.evaluate(test_data, batch_size=1)
    print(f"Test loss:{loss}, Test accuracy:{acc}")
```

```
48/48 [=====] - 1s 2ms/step - loss: 0.0857 - categorical_accuracy: 0.9375
Test loss:0.08573845773935318, Test accuracy:0.9375
```

Simple Hand-Gesture Detection

To simplify our project, we aim to recognize hand gestures in eight distinct categories:

- **Palm**
- **Fist**
- **Thumb Up**: extending the thumb
- **Indexing**: extending the index finger
- **L-shape**: extending both the thumb and index finger to form an L-shape
- **Relax**: the hand in a relaxed state
- **Other**: any other hand positions
- **Missing**: insufficient hand landmarks detected for classification



Keyframe Extraction

- **Purpose**

- Use less frames to represent a gesture chunk

- **Algorithm**

- (Wrists) Velocity and position-based filtering

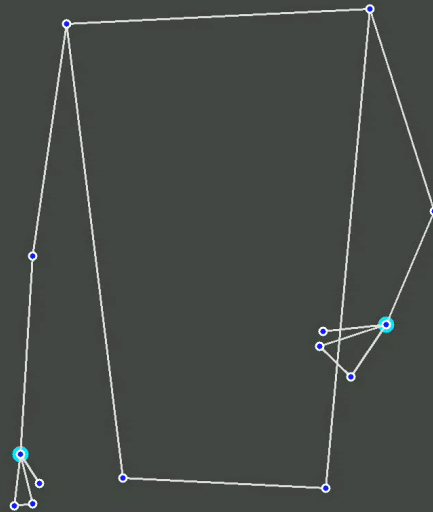
- **Example Results**

- Scenario: a student explaining whether “the lengths of the two diagonals of a rectangle are the same”
 - Duration: 30 sec
 - Original Frames: **129**
 - Extracted frames: **15**

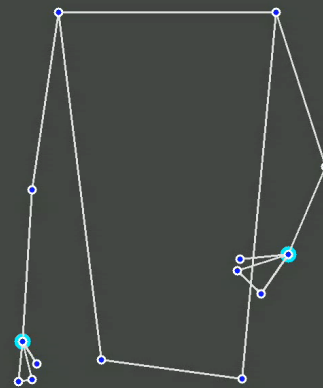


Video from 129 Original Frames

Question: “Are the lengths of the two diagonals of a rectangle the same?”



Video from 15 Extracted Keyframes



Challenges

- **Label:** Abstract nature of targeted gestures
- **Noise:** Presence of various noise types
- **Feature:** Absence of an intermediate feature construction layer
- **Data:** Limited dataset size post-'idle' state removal and inaccuracies in landmark detection by MediaPipe

Future Directions

- Enhance baseline algorithms for better **feature** extraction
- Strengthen algorithm **robustness** by incorporating edge cases
- **Integrate** keyframe extraction with hand gesture recognition
- Construct models using gesture chunks for **simple** gesture identification
- Advance the model to recognize more **abstract** gestures

Thank You!

From Group

“Dreams Kindle Hope,
Transforming New Ideas.”