Machine Learning Project

AIR GUITAR

Team Ronaldo

Lee Hyunsong, Song Minui, Hyewon Cho

INDEX

Problem Definition

Many people dream of enjoying the experience of playing the guitar, but the high cost and low portability of the instrument often make it difficult to start.

1. An expensive piece of equipment
2. Large and inconvenient to carry

▶ Spontaneous playing and consistent practice challenging

01. Introduction

Air-Guitar

lower these barriers and allow *anyone to enjoy guitar performance* easily by creating an **air guitar system** that works with a standard **Webcam**

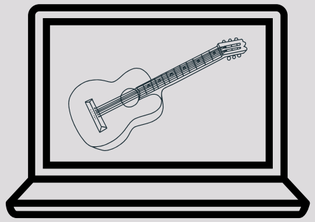
< Project GOAL >

- *Deliver a fun!*
- *Highly accessible!*
- *Portable new style of musical interaction!*

Image generated by ChatGPT

01. Introduction

Air-Guitar

Idea 1		Idea 2		Detection		Result
	+		▶		▶	
Hand Pose Estimation		Virtual Guitar Image		Left hand - Chrods Right hand - Strumming		Hand Movemet into Realistc guitar sound

02. Technical Point

(1) Hand Pose Estimation

OpenCV function - MediaPipe Hands

To track 21 hand landmarks in real time, extracting 2D/3D joint coordinates for chord classification and strumming detection.

Photo of the actual implementation

(2) Guitar Image Systhesis

Overlay a virtual guitar on the webcam feed, updating its position and orientation in real time based on both hands' locations.

02. Technical Point

(3) Chord Classification - *Left Hand*

Coordinate-based classification

Use landmark coordinates to compute distances and angles, then train a **lightweight ML classifier**.

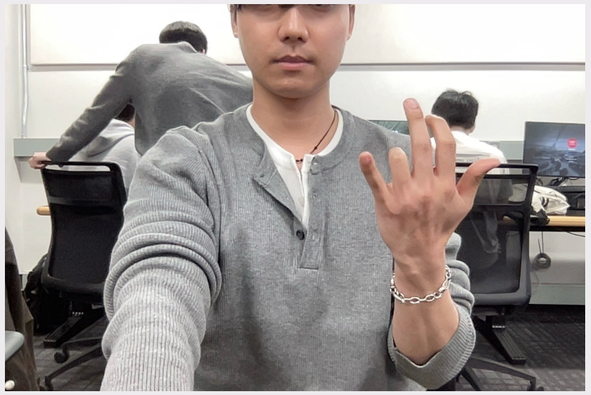
(4) Strumming Detection - *Right Hand*

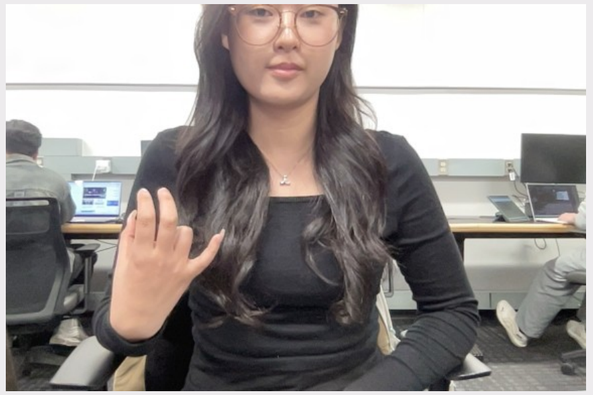
We detect strumming by tracking the right hand’s vertical movement across frames.
Fast, consistent motion is recognized as a strumming event, and optical flow is used to distinguish intentional strums from other hand movements.

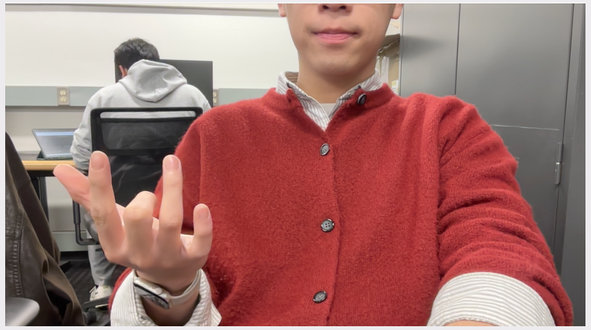
02. Dataset

1. Collect 60 code samples from each team member.

3 chord types × 60 samples × 3 members = 540 total samples.







03. Preprocessing

2. Extract hand landmark coordinates using MediaPipe & Save them to a CSV file

1. Origin Alignment:

Sets the wrist (Landmark 0) as the origin (0, 0). This ensures the model focuses on the hand's shape rather than its absolute position in the image.

2. Scaling:

Adjusts for the hand's distance from the camera. By dividing by the bounding box size, all landmark coordinates are normalized to a 0 to 1 range, making the data size-invariant.

03. Preprocessing Result

2. Extract hand landmark coordinates using MediaPipe and save them to a CSV file

3 coordinates for each finger joint and 6 for the palm
→ a total of 21 coordinates per sample.

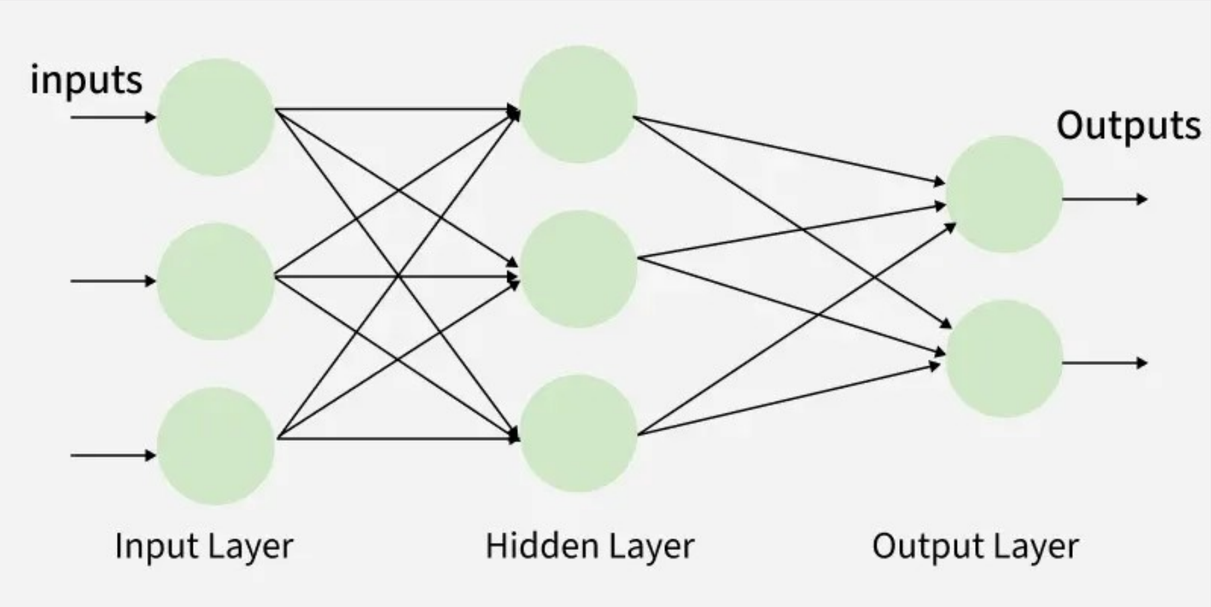
```
filepath,label,label_id,x0,x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12,x13,x14,x15,x16,x17,x18,x19,x20,y0,y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,y12,y13,y14,y15,y16,y17,y18,y
dataset\C\0001.png,C,0,0.0,0.1261470764875412,0.23299585282802582,0.3217525780200958,0.4291890561580658,0.15757417678833008,0.17911511659622192,0.1686359941959381,0.
dataset\C\0002.png,C,0,0.0,0.12228717654943466,0.22315742075443268,0.3193338215351105,0.4488257169723511,0.1395718902349472,0.1251627653837204,0.09942106157541275,0.
dataset\C\0003.png,C,0,0.0,0.07585910707712173,0.17155063152313232,0.28766149282455444,0.4285870492458343 5,0.1652398258447647,0.16359615325927734,0.11401873826980591
dataset\C\0004.png,C,0,0.0,0.119576595723629,0.1948644369840622,0.26168298721313477,0.3668496310710907,0.1290827989578247,0.11881206184625626,0.10925083607435226,0.1
dataset\C\0005.png,C,0,0.0,0.1149563193321228,0.18680229783058167,0.2289631962776184,0.3018203675746918,0.15443530678749084,0.1447911560535431,0.12414032965898514,0.
dataset\C\0006.png,C,0,0.0,0.13530266284942627,0.24870026111602783,0.3212498724460602,0.40745219588279724,0.2706661522388458,0.39021241664886475,0.3576943278312683,0
dataset\C\0007.png,C,0,0.0,0.10402286797761917,0.2673931419849396,0.40253522992134094,0.5308622717857361,0.39814022183418274,0.5518940091133118,0.47243136167526245,0
```

04. Model Training

(1) MLP

Multi-Layer Perceptron

An MLP is a type of artificial neural network composed of multiple layers of neurons.
It uses nonlinear activation functions to learn complex patterns and relationships in data.



04. Model Training

3. Train an MLP model using the coordinates stored in the CSV file

- Hidden layers: (128, 64), (256, 128, 64), (512, 256, 128, 64)
- Dataset split
  - Train : Validation : Test = 8 : 1 : 1

```
pipe = Pipeline([
      ("scaler", StandardScaler()),
      ("mlp", MLPClassifier(
          hidden_layer_sizes=tuple(args.hidden),
          activation="relu",
          solver="adam",
          alpha=1e-4,
          batch_size=64,
          learning_rate_init=1e-3,
          max_iter=500,
          early_stopping=True,
          validation_fraction=1/9, # train:val:test=8:1:1
          n_iter_no_change=20, #  20 epoch 연속으로 향상되지 않으면, 학습이 500
에포크에 도달하지 않았더라도 훈련을 중단
          random_state=args.seed,
          verbose=True
      ))
  ])
```

05. Result

Hidden Layer (128, 64)

```
[TEST] acc = 0.8367
              precision    recall  f1-score   support

           0     0.8667    0.8667    0.8667        15
           1     0.8571    0.7500    0.8000        16
           2     0.8000    0.8889    0.8421        18

    accuracy                         0.8367        49
   macro avg     0.8413    0.8352    0.8363        49
weighted avg     0.8391    0.8367    0.8359        49

Confusion matrix:
 [[13  1  1]
 [ 1 12  3]
 [ 1  1 16]]
```

83%

Hidden Layer (256, 128, 64)

```
[TEST] acc = 0.8571
              precision    recall  f1-score   support

           0     0.8462    0.7333    0.7857        15
           1     0.8235    0.8750    0.8485        16
           2     0.8947    0.9444    0.9189        18

    accuracy                         0.8571        49
   macro avg     0.8548    0.8509    0.8510        49
weighted avg     0.8566    0.8571    0.8551        49

Confusion matrix:
 [[11  3  1]
 [ 1 14  1]
 [ 1  0 17]]
```

86%

Hidden Layer (512, 256, 128, 64)

```
[TEST] acc = 0.9184
              precision    recall  f1-score   support

           0     1.0000    0.7333    0.8462        15
           1     0.8421    1.0000    0.9143        16
           2     0.9474    1.0000    0.9730        18

    accuracy                         0.9184        49
   macro avg     0.9298    0.9111    0.9111        49
weighted avg     0.9291    0.9184    0.9150        49

Confusion matrix:
 [[11  3  1]
 [ 0 16  0]
 [ 0  0 18]]
```

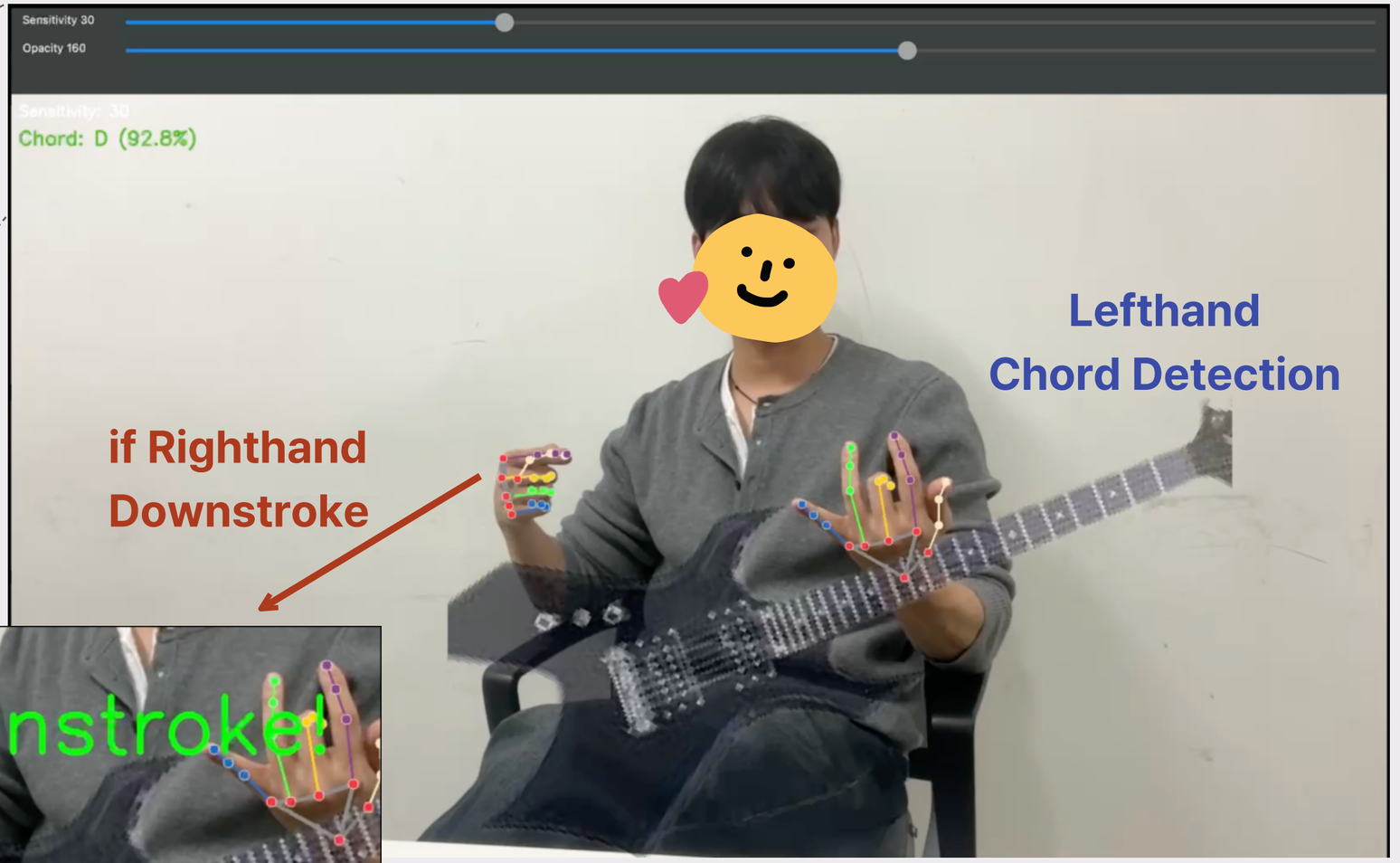
91%

Program execution Screen

Sensitivity & Opacity

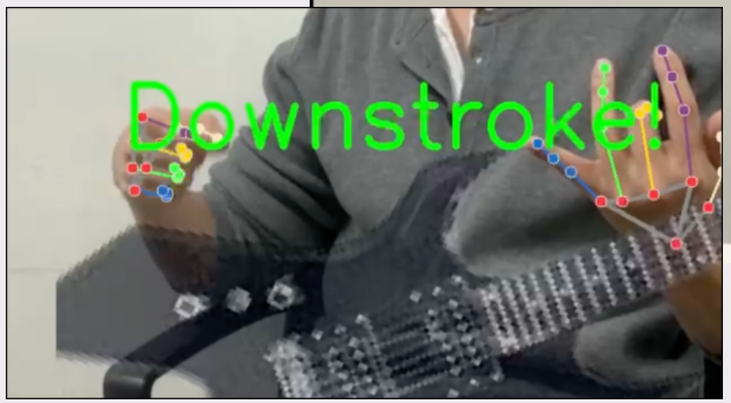
Chord : C / D / G

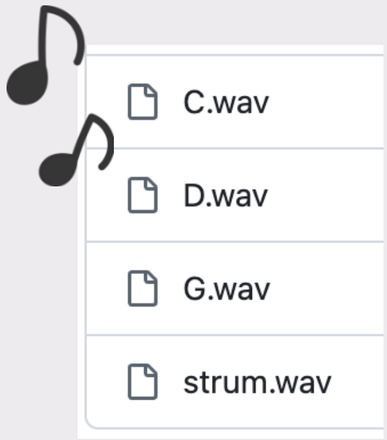
Accuracy



Lefthand
Chord Detection

if Righthand
Downstroke





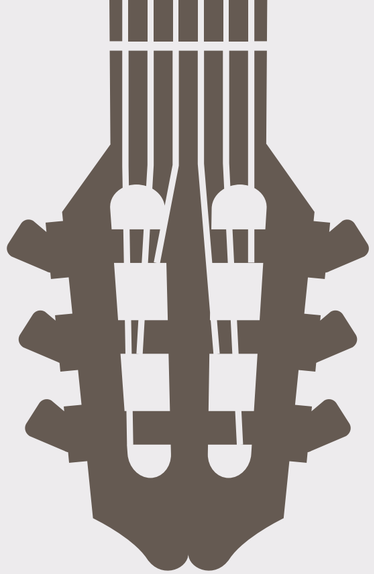
Challenges & Future Work

< Challenge >

Since *no dataset existed* in the first place, we had to create it ourselves.
It’s a bit disappointing because with a wider variety of chords and a larger amount of data, we could have built a much richer and more robust program.

< Future Work >

- We would like to enable **upstroke** detection as well.
- We also want to implement support for recognizing **multiple users** simultaneously so that more than one person can play the guitar together.
- I would like to extend this to **various instruments** and create a real virtual band.



Thank you