Install and Import Dependencies

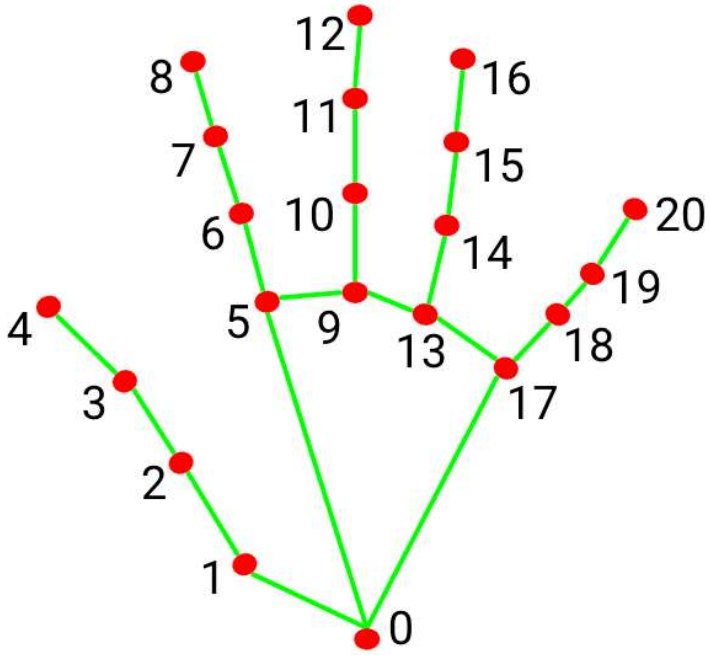
```
!pip install mediapipe opencv-python
```

```
Collecting mediapipe
  Downloading mediapipe-0.10.10-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (34.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 34.8/34.8 MB 38.7 MB/s eta 0:00:00
Requirement already satisfied: opencv-python in /usr/local/lib/python3.10/dist-packages (4.8.0.76)
Requirement already satisfied: absl-py in /usr/local/lib/python3.10/dist-packages (from mediapipe) (1.4.0)
Requirement already satisfied: attrs>=19.1.0 in /usr/local/lib/python3.10/dist-packages (from mediapipe) (23.2.0)
Requirement already satisfied: flatbuffers>=2.0 in /usr/local/lib/python3.10/dist-packages (from mediapipe) (23.5.26)
Requirement already satisfied: jax in /usr/local/lib/python3.10/dist-packages (from mediapipe) (0.4.23)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from mediapipe) (3.7.1)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from mediapipe) (1.25.2)
Requirement already satisfied: opencv-contrib-python in /usr/local/lib/python3.10/dist-packages (from mediapipe) (4.8.0
Requirement already satisfied: protobuf<4,>=3.11 in /usr/local/lib/python3.10/dist-packages (from mediapipe) (3.20.3)
Collecting sounddevice>=0.4.4 (from mediapipe)
  Downloading sounddevice-0.4.6-py3-none-any.whl (31 kB)
Requirement already satisfied: CFFI>=1.0 in /usr/local/lib/python3.10/dist-packages (from sounddevice>=0.4.4->mediapipe
Requirement already satisfied: ml-dtypes>=0.2.0 in /usr/local/lib/python3.10/dist-packages (from jax->mediapipe) (0.2.0
Requirement already satisfied: opt-einsum in /usr/local/lib/python3.10/dist-packages (from jax->mediapipe) (3.3.0)
Requirement already satisfied: scipy>=1.9 in /usr/local/lib/python3.10/dist-packages (from jax->mediapipe) (1.11.4)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mediapipe)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mediapipe) (0.
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mediapipe
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mediapipe
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mediapipe)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mediapipe) (9
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mediapipe)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mediap
Requirement already satisfied: pycparser in /usr/local/lib/python3.10/dist-packages (from CFFI>=1.0->sounddevice>=0.4.4
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotl
Installing collected packages: sounddevice, mediapipe
Successfully installed mediapipe-0.10.10 sounddevice-0.4.6
```

```
import mediapipe as mp
import cv2
import numpy as np
import uuid
import os


mp_drawing = mp.solutions.drawing_utils
mp_hands = mp.solutions.hands
```

Draw Hands



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

```
cap = cv2.VideoCapture(0)

with mp_hands.Hands(min_detection_confidence=0.8, min_tracking_confidence=0.5) as hands:
    while cap.isOpened():
        ret, frame = cap.read()

        # BGR 2 RGB
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        # Flip on horizontal
        image = cv2.flip(image, 1)

        # Set flag
        image.flags.writeable = False

        # Detections
        results = hands.process(image)

        # Set flag to true
        image.flags.writeable = True

        # RGB 2 BGR
        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

        # Detections
        print(results)

        # Rendering results
        if results.multi_hand_landmarks:
            for num, hand in enumerate(results.multi_hand_landmarks):
                mp_drawing.draw_landmarks(image, hand, mp_hands.HAND_CONNECTIONS,
                                          mp_drawing.DrawingSpec(color=(121, 22, 76), thickness=2, circle_radius=4),
                                          mp_drawing.DrawingSpec(color=(250, 44, 250), thickness=2, circle_radius=2),
                                           )


        cv2.imshow('Hand Tracking', image)

        if cv2.waitKey(10) & 0xFF == ord('q'):
```

```
            break

cap.release()
cv2.destroyAllWindows()


mp_drawing.DrawingSpec??
```

Output Images

```
os.mkdir('Output Images')


cap = cv2.VideoCapture(0)

with mp_hands.Hands(min_detection_confidence=0.8, min_tracking_confidence=0.5) as h
    while cap.isOpened():
        ret, frame = cap.read()

        # BGR 2 RGB
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        # Flip on horizontal
        image = cv2.flip(image, 1)

        # Set flag
        image.flags.writeable = False

        # Detections
        results = hands.process(image)

        # Set flag to true
        image.flags.writeable = True

        # RGB 2 BGR
        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
```

```
        # Detections
        print(results)

        # Rendering results
        if results.multi_hand_landmarks:
            for num, hand in enumerate(results.multi_hand_landmarks):
                mp_drawing.draw_landmarks(image, hand, mp_hands.HAND_CONNECTIONS,
                                          mp_drawing.DrawingSpec(color=(121, 22, 76),
                                          mp_drawing.DrawingSpec(color=(250, 44, 250)
                                           )

        # Save our image
        cv2.imwrite(os.path.join('Output Images', '{}.jpg'.format(uuid.uuid1())), i
        cv2.imshow('Hand Tracking', image)

        if cv2.waitKey(10) & 0xFF == ord('q'):
            break

cap.release()
cv2.destroyAllWindows()
```