Python les-materialen

Usage example of MediaPipe Pose Solution API in Python (see also http://solutions.mediapipe.dev/pose).

!pip install mediapipe

Upload any image that that has a person. We take two example images from the web: https://unsplash.com/photos/v4zceVZ5HK8 and https://unsplash.com/photos/e\_rhazQLaSs.

from google.colab import files  
uploaded = files.upload()

import cv2  
from google.colab.patches import cv2\_imshow  
import math  
import numpy as np  
  
DESIRED\_HEIGHT = 480  
DESIRED\_WIDTH = 480  
def resize\_and\_show(image):  
 h, w = image.shape[:2]  
 if h < w:  
 img = cv2.resize(image, (DESIRED\_WIDTH, math.floor(h/(w/DESIRED\_WIDTH))))  
 else:  
 img = cv2.resize(image, (math.floor(w/(h/DESIRED\_HEIGHT)), DESIRED\_HEIGHT))  
 cv2\_imshow(img)  
  
# Read images with OpenCV.  
images = {name: cv2.imread(name) for name in uploaded.keys()}  
# Preview the images.  
for name, image in images.items():  
 print(name)   
 resize\_and\_show(image)

david-hofmann-e\_rhazQLaSs-unsplash.jpg

png

thao-le-hoang-v4zceVZ5HK8-unsplash.jpg

png

All MediaPipe Solutions Python API examples are under mp.solutions.

For the MediaPipe Pose solution, we can access this module as mp\_pose = mp.solutions.pose.

You may change the parameters, such as static\_image\_mode and min\_detection\_confidence, during the initialization. Run help(mp\_pose.Pose) to get more informations about the parameters.

import mediapipe as mp  
mp\_pose = mp.solutions.pose  
mp\_drawing = mp.solutions.drawing\_utils   
mp\_drawing\_styles = mp.solutions.drawing\_styles  
  
help(mp\_pose.Pose)

# Run MediaPipe Pose and draw pose landmarks.  
with mp\_pose.Pose(  
 static\_image\_mode=True, min\_detection\_confidence=0.5, model\_complexity=2) as pose:  
 for name, image in images.items():  
 # Convert the BGR image to RGB and process it with MediaPipe Pose.  
 results = pose.process(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))  
   
 # Print nose landmark.  
 image\_hight, image\_width, \_ = image.shape  
 if not results.pose\_landmarks:  
 continue  
 print(  
 f'Nose coordinates: ('  
 f'{results.pose\_landmarks.landmark[mp\_pose.PoseLandmark.NOSE].x \* image\_width}, '  
 f'{results.pose\_landmarks.landmark[mp\_pose.PoseLandmark.NOSE].y \* image\_hight})'  
 )  
  
 # Draw pose landmarks.  
 print(f'Pose landmarks of {name}:')  
 annotated\_image = image.copy()  
 mp\_drawing.draw\_landmarks(  
 annotated\_image,  
 results.pose\_landmarks,  
 mp\_pose.POSE\_CONNECTIONS,  
 landmark\_drawing\_spec=mp\_drawing\_styles.get\_default\_pose\_landmarks\_style())  
 resize\_and\_show(annotated\_image)

Downloading model to /usr/local/lib/python3.7/dist-packages/mediapipe/modules/pose\_landmark/pose\_landmark\_heavy.tflite  
Nose coordinates: (182.0577621459961, 255.44222474098206)  
Pose landmarks of david-hofmann-e\_rhazQLaSs-unsplash.jpg:

png

Nose coordinates: (206.92650318145752, 202.37878853082657)  
Pose landmarks of thao-le-hoang-v4zceVZ5HK8-unsplash.jpg:

png

# Run MediaPipe Pose and plot 3d pose world landmarks.  
with mp\_pose.Pose(  
 static\_image\_mode=True, min\_detection\_confidence=0.5, model\_complexity=2) as pose:  
 for name, image in images.items():  
 results = pose.process(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))  
  
 # Print the real-world 3D coordinates of nose in meters with the origin at  
 # the center between hips.  
 print('Nose world landmark:'),  
 print(results.pose\_world\_landmarks.landmark[mp\_pose.PoseLandmark.NOSE])  
   
 # Plot pose world landmarks.  
 mp\_drawing.plot\_landmarks(  
 results.pose\_world\_landmarks, mp\_pose.POSE\_CONNECTIONS)

Nose world landmark:  
x: -0.6137464046478271  
y: 0.04928246885538101  
z: -0.4365992546081543  
visibility: 0.9999992847442627

png

Nose world landmark:  
x: -0.04833628237247467  
y: -0.5876999497413635  
z: -0.37148377299308777  
visibility: 0.9999998807907104

png

# Run MediaPipe Pose with `enable\_segmentation=True` to get pose segmentation.  
with mp\_pose.Pose(  
 static\_image\_mode=True, min\_detection\_confidence=0.5,   
 model\_complexity=2, enable\_segmentation=True) as pose:  
 for name, image in images.items():  
 results = pose.process(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))  
  
 # Draw pose segmentation.  
 print(f'Pose segmentation of {name}:')  
 annotated\_image = image.copy()  
 red\_img = np.zeros\_like(annotated\_image, dtype=np.uint8)  
 red\_img[:, :] = (255,255,255)  
 segm\_2class = 0.2 + 0.8 \* results.segmentation\_mask  
 segm\_2class = np.repeat(segm\_2class[..., np.newaxis], 3, axis=2)  
 annotated\_image = annotated\_image \* segm\_2class + red\_img \* (1 - segm\_2class)  
 resize\_and\_show(annotated\_image)

Pose segmentation of david-hofmann-e\_rhazQLaSs-unsplash.jpg:

png

Pose segmentation of thao-le-hoang-v4zceVZ5HK8-unsplash.jpg:

png