پیوند بین پایتون و یونیتی

این برنامه یک کد پایتون است که از کتابخانه‌های mediapipe و cv2 استفاده می‌کند. در زیر توضیحات خط به خط این کد آمده است:

۱. import mediapipe as mpe : وارد کردن کتابخانه mediapipe با نام مستعار mpe .

۲. import cv2 : وارد کردن کتابخانه cv2 برای پردازش تصاویر و ویدئو.

۳. import threading : وارد کردن کتابخانه threading برای استفاده از رشته‌ها و همزمان سازی.

۴. import time : وارد کردن کتابخانه time برای کنترل زمان در برنامه.

۵. import struct : وارد کردن کتابخانه struct برای کار با ساختارهای داده پیچیده.

۷. TERMINATE_THREADS = False : تعریف یک متغیر بولین به نام TERMINATE_THREADS که مقدار اولیه آن False است.

۸. DEBUGIN = True : تعریف یک متغیر بولین به نام DEBUGIN که مقدار اولیه آن True است.

۹. EMPLOY_CUSTOMIZED_CAMERA_OPTIONS = False : تعریف یک متغیر بولین به نام EMPLOY_CUSTOMIZED_CAMERA_OPTIONS که مقدار اولیه آن False است.

۱۰. FRAMES_PER_SECOND = ۶۰ : تعریف یک متغیر عددی به نام FRAMES_PER_SECOND که مقدار آن برابر با ۶۰ است.

۱۱. WIDTH_ONE = ۳۲۰ : تعریف یک متغیر عددی به نام WIDTH_ONE که مقدار آن برابر با ۳۲۰ است.

۱۲. HEIGHT_TOW = ۲۴۰ : تعریف یک متغیر عددی به نام HEIGHT_TOW که مقدار آن برابر با ۲۴۰ است.

۱۳. LEVEL_OF_MODEL_COMPLEXITY = ۱ : تعریف یک متغیر عددی به نام LEVEL_OF_MODEL_COMPLEXITY که مقدار آن برابر با ۱ است.

۱۵. class Crisp_image_capture(threading.Thread) : تعریف یک کلاس به نام Crisp_image_capture که از کلاس threading.Thread ارثبری می‌کند.

۱۶. Lid = None : تعریف یک متغیر به نام Lid که اولاً برابر با None است.

۱۷. ret = None : تعریف یک متغیر به نام ret که اولاً برابر با None است.

۱۸. frame_video = None : تعریف یک متغیر به نام frame_video که اولاً برابر با None است.

۱۹. jogging = False : تعریف یک متغیر بولین به نام jogging که مقدار اولیه آن False است.

۲۰. numberer = 0 : تعریف یک متغیر عددی به نام numberer که مقدار اولیه آن برابر با ۰ است.

۲۱. Time_control = 0,0: تعریف یک متغیر عددی به نام Time_control که مقدار اولیه آن برابر با ۰,۰ است.

۲۳. def run(self): : تعریف یک تابع به نام run در داخل کلاس Crisp_image_capture .

۲۴. self.Lid = cv2.VideoCapture : ایجاد یک شی از کلاس cv2.VideoCapture و ذخیره‌ی آن در متغیر Lid .

۲۵. if EMPLOY_CUSTOMIZED_CAMERA_OPTIONS: : بررسی شرطی که در صورت برقرار بودن، تنظیمات دلخواه دوربین را اعمال می‌کند.

۲۶. self.Lid.set(cv2.CAP_PROP_FPS, FRAMES_PER_SECOND) : تنظیم نرخ فریم دوربین با استفاده از متغیر FRAMES_PER_SECOND .

۲۷. self.Lid.set(cv2.CAP_PROP_FRAME_WIDTH, WIDTH_ONE) : تنظیم عرض فریم دوربین با استفاده از متغیر WIDTH_ONE .

۲۸. self.Lid.set(cv2.CAP_PROP_FRAME_HEIGHT, HEIGHT_TOW) : تنظیم ارتفاع فریم دوربین با استفاده از متغیر HEIGHT_TOW .

۲۹. else: : بخش دیگر شرطی که در صورت برقرار نبودن شرط قبلی اجرا می‌شود.

۳۰. time.sleep(2) : تاخیر ۱ ثانیه.

۳۲. while not TERMINATE_THREADS: : حلقه‌ای که تا زمانی که متغیر TERMINATE_THREADS برابر با False باشد، اجرا می‌شود.

۳۳. self.ret, self.frame_video = self.Lid.read : خواندن یک فریم از ویدئوی دوربین و ذخیره‌ی تصویر در متغیرهای ret و frame_video.

.jogging = True : تنظیم مقدار True برای متغیر jogging.

۳۵. if DEBUGIN: : بررسی شرطی که در صورت برقرار بودن، بخش مشخصات را نمایش می‌دهد.

۳۶. self.numberer = self.numberer + 1 : افزایش مقدار متغیر numberer به ازای هر تکرار حلقه.

۳۷. if time.time() - self.Time_control >= 3: : بررسی شرطی که در صورتی که از زمان کنترلی ۳ ثانیه گذشته باشد، مقدار متغیر numberer را صفر می‌کند.

۳۸. self.numberer = 0 : تنظیم مقدار صفر برای متغیر numberer.

۳۹. self.Time_control = time.time : به روزرسانی زمان کنترل به زمان فعلی.

۴۱. class Detection_of_body_movements(threading.Thread) : تعریف یک کلاس به نام Detection_of_body_movements که از کلاس threading.Thread ارثبری می‌کند.

۴۲. data_nl=: تعریف یک متغیر رشته‌ای به نام data_nl با مقدار اولیه خالی.

۴۳. line = None : تعریف یک متغیر به نام line که اولاً برابر با None است.

۴۴. since_last_connection_check = 0 : تعریف یک متغیر عددی به نام since_last_connection_check که مقدار اولیه آن برابر با ۰ است.

۴۵. Details_of_the_time_elapsed_since_publication = 0 : تعریف یک متغیر عددی به نام Details_of_the_time_elapsed_since_publication که مقدار اولیه آن برابر با ۰ است.

۴۷. def run(self): : تعریف یک تابع به نام run در داخل کلاس Detection_of_body_movements .

۴۸. mechanical_drawing = mpe.solutions.drawing_utils : تعریف یک متغیر به نام mechanical_drawing که به drawing_utils از کتابخانه mediapipe اشاره می‌کند.

۴۹. Parliamentary_Pose = mpe.solutions.pose : تعریف یک متغیر به نام Parliamentary_Pose که به pose از کتابخانه mediapipe اشاره می‌کند.

۵۰. acquire_fresh = Crisp_image_capture() : ایجاد یک شی از کلاس Crisp

_image_capture و ذخیره‌ی آن در متغیر acquire_fresh .

۵۱. acquire_fresh.start() : فراخوانی تابع start() بر روی شی acquire_fresh ، که باعث شروع اجرای رشته مربوطه می‌شود.

۵۳. with Parliamentary_Pose.Pose() : شروع بخش with که از Parliamentary_Pose.Pose استفاده می‌کند.

۵۴. min_detection_confidence=0.8: تنظیم حداقل اطمینان تشخیص به ۰٫۸۰.

۵۵. min_tracking_confidence=0.5 : تنظیم حداقل اطمینان پیگیری به ۰٫۵.

۵۶. model_complexity=LEVEL_OF_MODEL_COMPLEXITY : تنظیم پیچیدگی مدل با استفاده از متغیر LEVEL_OF_MODEL_COMPLEXITY .

۵۷. static_image_mode=False : تنظیم حالت تصویر ثابت به False .

۵۸. enable_segmentation=True : فعال کردن قطعه‌بندی.

۵۹.) :as pose_new : پایان بخش with و ذخیره‌ی نتیجه در متغیر pose_new .

۶۱. while not TERMINATE_THREADS and acquire_fresh.jogging == False: : حلقه‌ای که تا زمانی که متغیر TERMINATE_THREADS برابر با False و متغیر jogging شی acquire_fresh برابر با False باشد، اجرا می‌شود.

۶۲. (Wait for the camera to read)print : چاپ پیام Wait for the camera to read .

۶۳. time.sleep(0.5) : تاخیر ۰٫۵ ثانیه.

۶۵. (...start)print : چاپ پیام start… .

۶۶. :()while not TERMINATE_THREADS and acquire_fresh.Lid.isOpened : حلقه‌ای که تا زمانی که متغیر TERMINATE_THREADS برابر با False و دوربین متعلق به شی acquire_fresh باز است، اجرا می‌شود.

۶۷. image_new = acquire_fresh.frame_video : اختصاص دادن تصویر جدید از دوربین به متغیر image_new .

۶۸. (image_new, 1)cv2.flip = image_new : تغییر جهت تصویر با استفاده از تابع flip از کتابخانه cv2 .

۶۹. image_new.flags.writeable = DEBUGIN : تنظیم دسترسی نوشتن برای تصویر بر اساس مقدار DEBUGIN .

۷۰. (image_new)pose_new.process = results_new : پردازش تصویر با استفاده از تابع process از شی pose_new و ذخیره‌ی نتیجه در متغیر results_new .

۷۲. :if DEBUGIN : بررسی شرطی که در صورت برقرار بودن، بخش مشخصات را نمایش می‌دهد.

۷۳. :if time.time() - self.since_last_connection_check >= 1 : بررسی شرطی که در صورتی که از زمان بررسی ارتباط قبلی ۱ ثانیه گذشته باشد، اجرا می‌شود.

۷۴. ()self.since_last_connection_check = time.time : به روزرسانی زمان بررسی ارتباط به زمان فعلی.

۷۶. :if results_new.pose_landmarks : بررسی شرطی که در صورت برقرار بودن، مشخصات لندمارک‌های قابل ردیابی را نمایش می‌دهد.

۷۷. mechanical_drawing.draw_landmarks(image_new, results_new.pose_landmarks, Parliamentary_Pose.POSE_CONNECTIONS, mechanical_drawing.DrawingSpec(color=(۰, ۱۰۰, ۰), thickness=۲, circle_radius=۲), mechanical_drawing.DrawingSpec(color=(۰, ۲۵۵, ۲۵۵), thickness=۲, circle_radius=۲)) : رسم لندمارک‌ها بر روی تصویر با استفاده از تابع draw_landmarks از شی mechanical_drawing با استفاده از مشخصات داده شده.

۷۸. cv2.imshow('tracking', image_new) : نمایش تصویر با استفاده از تابع imshow از کتابخانه cv۲ .

۷۹. cv2.waitKey(3) : انتظار برای ورود یک کلید از کیبورد به مدت ۳ میلی‌ثانیه.

۸۱. if self.line == None and time.time() - self.since_last_connection_check >= 1: : بررسی شرطی که در صورتی که self.line برابر با None باشد و از زمان بررسی ارتباط قبلی ۱ ثانیه گذشته باشد، اجرا می‌شود.

۸۲. try: : بخش try برای دستکاری خطا.

۸۳. self.line = open(r'\\.\pipe\Unity', 'r+b', 0) : باز کردن لوله ارتباطی با نرم‌افزار دیگر.

۸۴. except FileNotFoundError: : بخش except برای کمک به برطرف کردن خطاهای فایل پیدا نشد.

۸۵ self.line = None : تنظیم مقدار None برای متغیر self.line .

۸۶. self.Details_of_the_time_elapsed_since_publication = time.time() : به روزرسانی زمانی که از زمان انتشار گذشته است.

۸۸. if self.line != None: : بررسی شرطی که در صورت برقرار بودن، داده‌ها را به صورت رشته در self.data_nl ذخیره می‌کند.

۸۹. for i in range(0,33) : حلقه‌ای برای اعداد i در بازه ۰ تا ۳۳.

۹۰. self.data_nl += f
{i}|{hand_world_landmarks_new.landmark[i].x}|{hand_world_landmarks_new.landmark[i].y}|{hand_world_landmarks_new.landmark[i].z}\n : الحاق رشته به self.data_nl با استفاده از مقادیر i و مختصات لندمارک‌ها.

۹۲. D = self.data_nl.encode('ascii') : رمزگذاری رشته self.data_nl با استفاده از رمزگذاری ASCII.

۹۴. self.line.write(struct.pack('I', len(D)) + D) : نوشتن طول رشته و رشته رمزگذاری شده در لوله ارتباطی.

۹۵. self.line.seek(0) : تنظیم نقطه قرارگیری در لوله ارتباطی به ابتدا.

۹۷. except Exception as ex: : بخش except برای کمک به برطرف کردن خطاهای استثنا.

۹۸. self.line = None : تنظیم مقدار None برای متغیر self.line .

۱۰۱. self.line.close() : بستن لوله ارتباطی.

۱۰۲. acquire_fresh.Lid.release() : آزاد کردن منابع مرتبط با دوربین.

۱۰۳. cv2.destroyAllWindows() : بستن همه‌ی پنجره‌های متعلق به cv2 .

۱۰۵. thread = Detection_of_body_movements().start() : ایجاد یک شی از کلاس Detection_of_body_movements و فراخوانی تابع start() بر روی آن، که باعث شروع اجرای رشته مربوطه می‌شود و نتیجه در متغیر thread ذخیره می‌شود.

۱۰۶. i_new = input() : خواندن ورودی از کاربر و ذخیره‌ی آن در متغیر i_new .

۱۰۷. TERMINATE_THREADS = True : تنظیم مقدار True برای متغیر TERMINATE_THREADS .

۱۰۸. time.sleep(0,5) : تاخیر ۰,۵ ثانیه.

۱۰۹. exit() : خروج از برنامه.

این برنامه یک سیستم تشخیص حرکات بدن است. در اینجا ماژول‌های mediapipe و cv۲ برای تشخیص حرکات بدن و کنترل دوربین استفاده شده است. در این برنامه از چندین نخ استفاده شده است تا فرآیندها به صورت همزمان اجرا شوند.

ابتدا متغیرها و پارامترهای برنامه تعریف شده‌اند. سپس یک کلاس به نام Crisp_image_capture تعریف شده است که یک نخ راه‌اندازی می‌کند و تصاویری که از دوربین دریافت می‌شوند را ضبط می‌کند. این نخ همچنین تنظیمات دوربین را بررسی و تنظیم می‌کند.

سپس کلاسی به نام Detection_of_body_movements تعریف شده است که نخ دیگری راه‌اندازی می‌کند و حرکات بدن را تشخیص می‌دهد. این کلاس از ماژول mediapipe برای تشخیص حرکات بدن استفاده می‌کند. ابتدا یک نخ دیگر به نام Crisp_image_capture راه‌اندازی می‌شود تا تصاویر از دوربین دریافت شوند. سپس در حلقه‌ای که تا زمانی که برنامه متوقف نشده و دوربین باز است ادامه می‌یابد، تصاویر گرفته شده از دوربین را به ماژول mediapipe می‌دهد تا حرکات بدن را تشخیص دهد. سپس نتایج را رسم کرده و در پنجره‌ای به نام tracking نمایش می‌دهد. همچنین، در این بخش، اتصالی با نرم‌افزار دیگری برقرار می‌کند تا اطلاعات حرکات بدن را برای آن ارسال کند.

در انتها، نخ Detection_of_body_movements راه‌اندازی شده و سپس منتظر ورودی از کاربر می‌شود. در صورتی که ورودی دریافت شود، متغیر TERMINATE_THREADS به مقدار True تغییر می‌کند و به طور متوالی به کلیه نخ‌ها دستور متوقف شدن داده می‌شود.

به علاوه، در انتهای برنامه دوربین آزاد شده و پنجره‌های باز نیز بسته می‌شوند.