

Handout: MediaPipe-Pipeline

1 Schritt-für-Schritt Anleitung

In diesem Handout wird die korrekte Verwendung der erstellten MediaPipe-Pipeline erläutert. Beispielfideos finden sich unter dem Ordner Markerlose-Bewegungserfassung/Videos. In dem Beispiel wird ein einfaches Arm-Modell verwendet, welches auf 3 Markern basiert (Schulter, Ellbogen und Handgelenk). Das verwendete OpenSim Modell ist das im GitHub-Repository befindliche Markerlose-Bewegungserfassung/OpenSim/models/arm_modell.osim.

Detailliertere Erklärungen zu den einzelnen Schritten finden Sie im Kapitel 2.4 des Hauptberichts.

1.1 Repository anlegen

Folgenden Befehl an gewünschter Stelle im Dateisystem ausführen:

```
git clone https://github.com/PhilippSG/Markerlose-Bewegungserfassung.git
```

1.2 Umgebung einrichten

Python Version 3.12.12 verwenden und Pakete mit den in der Sektion Softwareversionen und Module des Berichts dargestellten Versionen installieren.

1.3 Videos erstellen

Mindestens 2 Videos werden benötigt: statische (z.B. T-Pose, ca. 3-5 Sekunden halten) und dynamische Aufnahme mit der gewünschten Bewegung. Genauere Anforderungen an die Videos sind im Bericht in der Sektion Videos beschrieben.

Die Videos müssen unter dem Ordner Markerlose-Bewegungserfassung/Videos abgelegt oder die Pfade im Code entsprechend angepasst werden.

1.4 Videos laden und anzeigen

Führen Sie das Jupyter Notebook Workflow.ipynb bis zur Zelle Execute aus. In der Zelle darunter finden sich die Variablen `static_video_name` und `dyn_video_name`. Setzen Sie diese auf die entsprechenden Videonamen (nur Dateiname, ohne Dateiendung). Die Dateiendung sollte `.mp4` sein, ansonsten muss diese in den Variablen `static_video_path` und `dyn_video_path` entsprechend angepasst werden. Führen Sie nun diese und die darauffolgende Zelle aus, um die Videos über OpenCV zu laden und anzuzeigen. Sollten die Videos nicht in der richtigen Ausrichtung abgespielt werden, so muss die Variable `rotate_video_by_degrees` entsprechend angepasst werden. Rotiert wird dabei im Uhrzeigersinn.

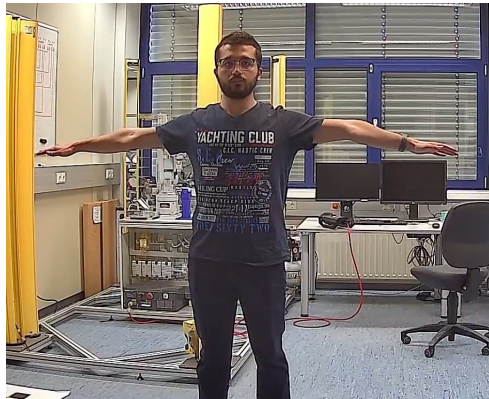
1.5 Erstellen der Landmarks

In der nächsten Zelle findet sich die `landmark_map`. In den Kommentaren finden sich bereits alle von MediaPipe BlazePose verfügbaren Marker. Hier müssen Sie die gewünschten Marker mit den Namen der Marker des OpenSim Modells anpassen. Die Marker des `arm_modell.osim` heißen `r_acromion`, `r_humerus_epicondyle` und `r_radius_styloid` und entsprechen in der Reihenfolge den Markern rechte Schulter, Ellbogen und Handgelenk.

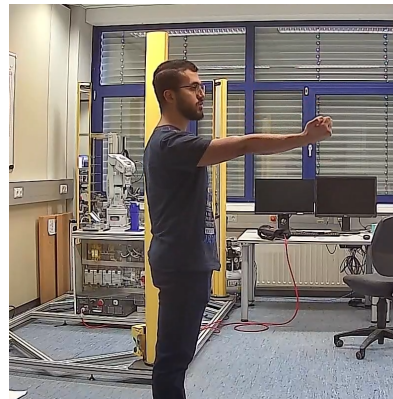
1.6 Ausführen der Pose Estimation

In diesem Schritt findet die eigentliche Pose Estimation von MediaPipe statt. Vor dem jeweiligen Aufruf der `pose_estimation` Funktion findet sich die Variable `rotate_around_y_axis`. Diese sorgt, wie der Name vermuten lässt, für eine Rotation der Daten um die y-Achse. Damit kann die Ausrichtung auf die Ausrichtung des OpenSim Modells angepasst werden, um sicherzustellen, dass die Marker auf die des Modells angepasst sind. Rotiert wird dabei gegen den Uhrzeigersinn.

Die Standardausrichtung des OpenSim Modells entspricht einer Person, die im Video nach rechts schaut. In Abbildung 1 finden sich Beispiele mit der jeweils notwendigen Rotation.



(a) rotate_around_y_axis=90



(b) rotate_around_y_axis=0

Abbildung 1: Beispiel Orientierungen mit notwendiger Rotation

Das Ausführen dieser Zelle kann je nach Modellkomplexität und Dauer der Videos einige Minuten in Anspruch nehmen.

1.7 Skalierung

Anschließend müssen die Daten skaliert werden, um bessere Ergebnisse zu erreichen. Dafür müssen die Körperteile in Gruppen in dem Array `limb_groups` übergeben werden.

Wenn nach einer gemessenen Länge skaliert werden soll, kann diese in der Variablen `reference_length_for_scaling` und das Körperteil in der Variablen `reference_limb_for_scaling` übergeben werden.

Für die Ursprungsverschiebung muss ein Marker gewählt werden, der als Anker für die Verschiebung genommen wird. Tragen Sie den Namen dieses Markers in der Variable `anchor_marker` ein und übergeben Sie die absolute Position dieses Markers im OpenSim Modell in der Variable `anchor_vector`.

1.8 Formatierung zu .trc

Führen Sie zuletzt die letzte Zelle aus, um die bisher verwendeten .csv Dateien in .trc Dateien zu konvertieren, die dann in OpenSim geladen werden können.