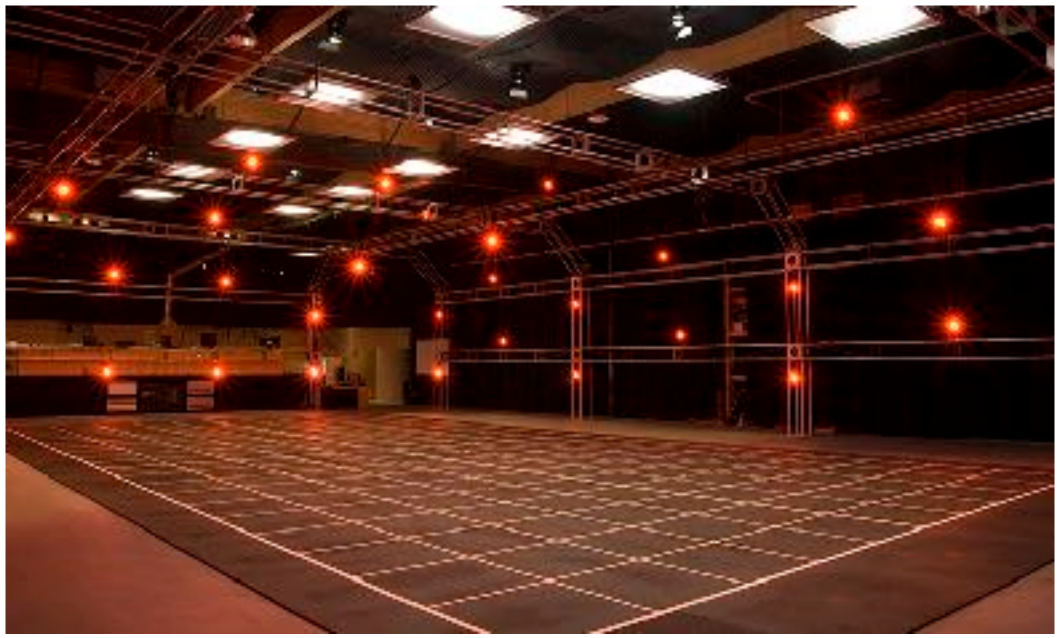
Integrating Human Body MoCaps into Blender using RGB Images

Jordi Sanchez-Riera and Francesc Moreno-Noguer
{jsanchez,fmoreno}@iri.upc.edu
IRI CSIC-UPC

Outline

- MoCap systems and applications
- Blender MoCap integration
    - From RGB image to 3D body joints
    - Match 3D human model to 3D skeleton movement
- Offline results with synthetic and real data

MoCap studio



https://www.activision.com/company/locations/activision-capture

https://anotherindiestudio.com/diccionario-o-vocabulario/que-es-la-captura-de-movimiento-o-motion-capture/

- Installation with multiple cameras
- Needs dedicated space

} → Expensive

- Actor to track needs to wear a suit with markers
- Retrieved data needs to be processed

\+ Obtain high quality motion files

MoCap from suit

https://www.rokoko.com/en/products/smartsuit-pro

- Sensors are incorporated in the suit
- More cheap, but still not affordable for everyone

+ Can be used outdoors
+ Quality of recorded action is high
+ Usually comes with software to process data

MoCap from RGBD

Commercial

https://www.reallusion.com/iclone/mocap/

Free

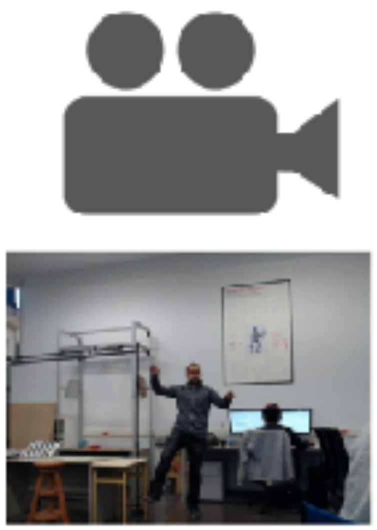
https://brekel.com/kinect-3d-scanner/

- Quality of motion is more unstable
- Indoor actions between 0.5-2 meters of the camera
- Usually only compatible with Windows OS
+ Commercial and open source versions
+ Affordable for broad public

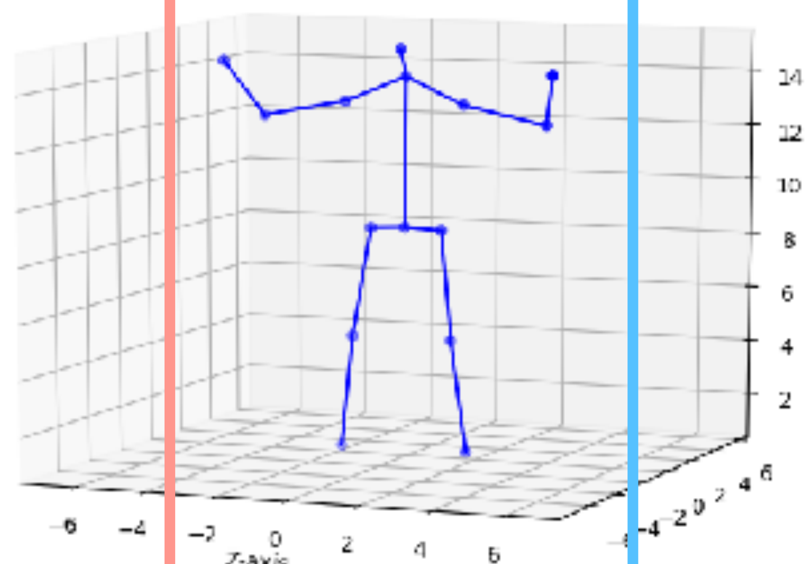
Outline

- MoCap systems and applications
- Blender MoCap integration
  - From RGB image to 3D body joints
  - Match 3D human model to 3D skeleton movement
- Offline results with synthetic and real data

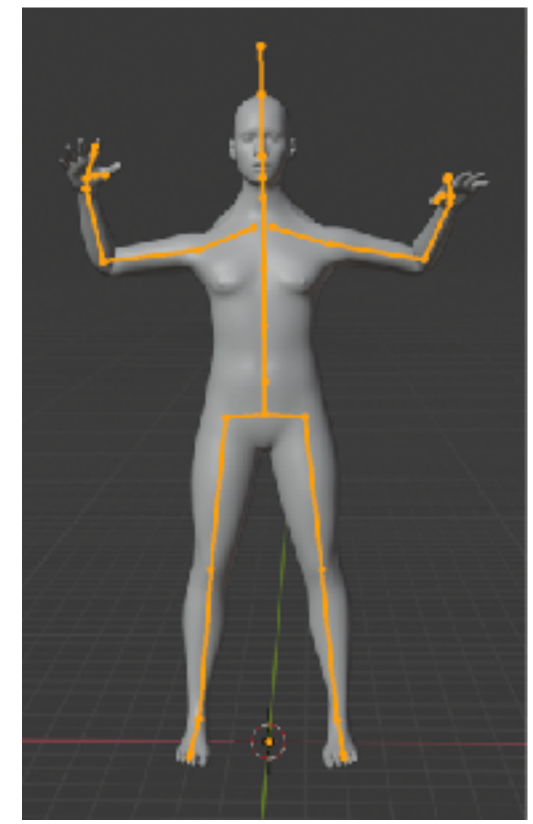
MoCap system



1



2



Get 3D joint coordinates

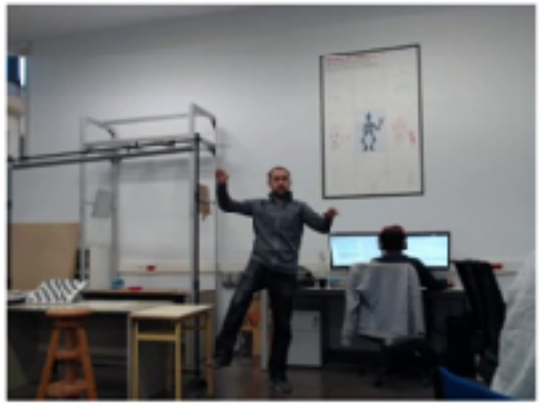
Transfer motion

Pose estimation

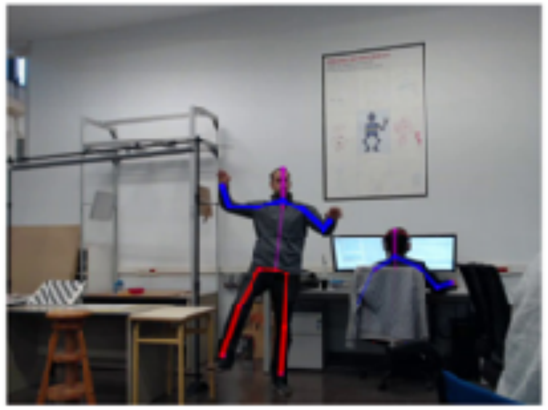
Pose transfer

Human 3D pose estimation

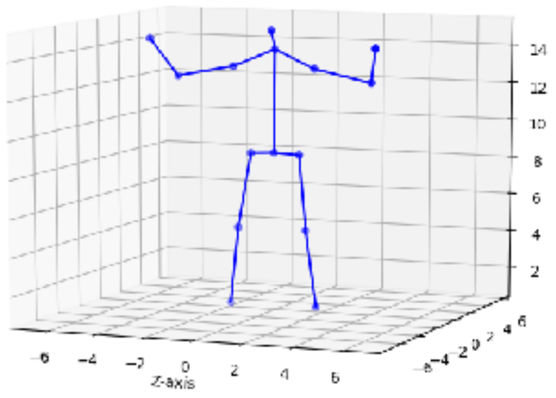
RGB



2D joints



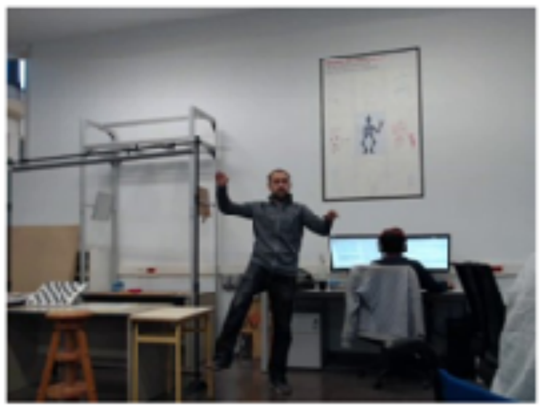
3D joints



- Standard approach
- Allows fast inference
- Reliable detections, we can use already trained weights
- Easy customisable

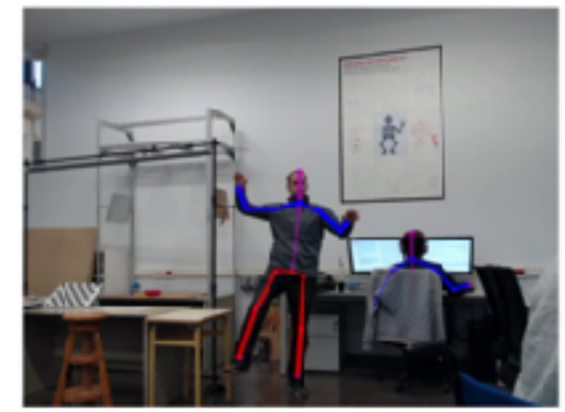
Obtain 2D joint coordinates

RGB



AlphaPose*

2D joints



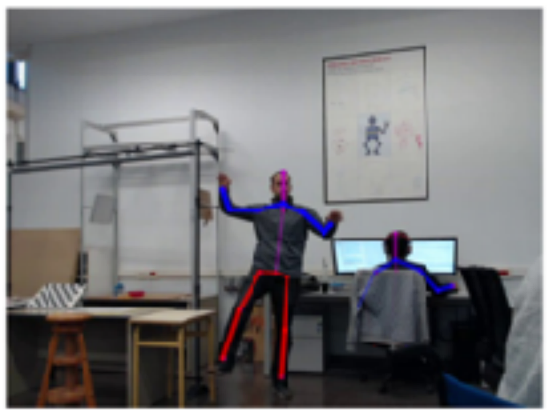
- Top down approach
- Rely on human pose detection → {
  - Complications if many people in scene
  - We expect usually on person in scene
- Very fast to perform inference

- Skeleton similar to our human model
- Better performance OpenPose
- Use default weights

* H.-S. Fang, S. Xie, Y.-W. Tai, and C. Lu, “RMPE: Regional multiperson pose estimation,” in ICCV, 2017.

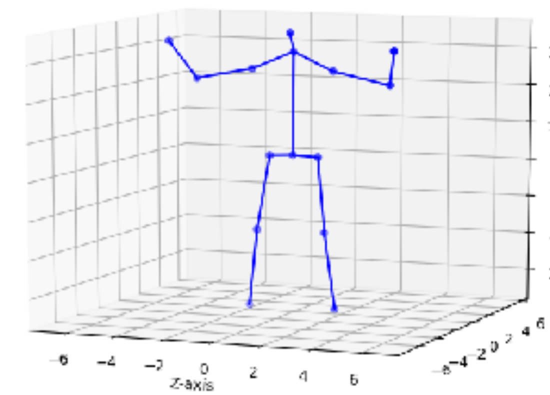
Obtain 2D joint coordinates

2D joints



Martinez et al. *

3D joints



- **Fast to train and fast to perform inference (ms)**
- **Network simple linear model**
- **Can customise with our own data world reference coordinates**

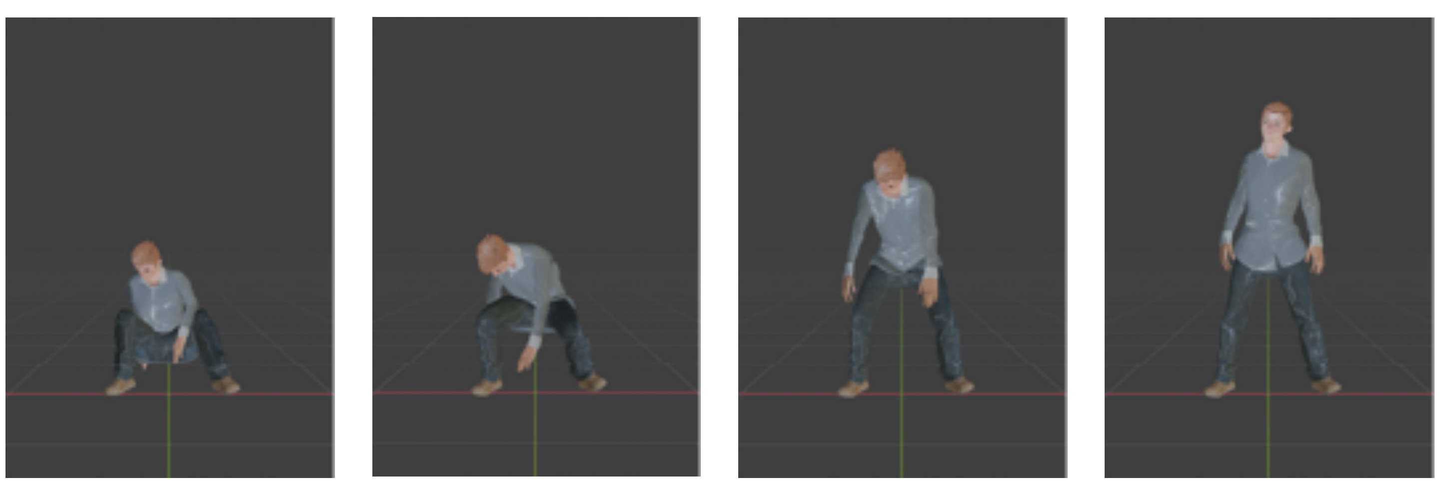
* J. Martinez, R. Hossain, J. Romero, and J. J. Little, “A simple yet effective baseline for 3d human pose estimation,” in ICCV, 2017.

Training dataset

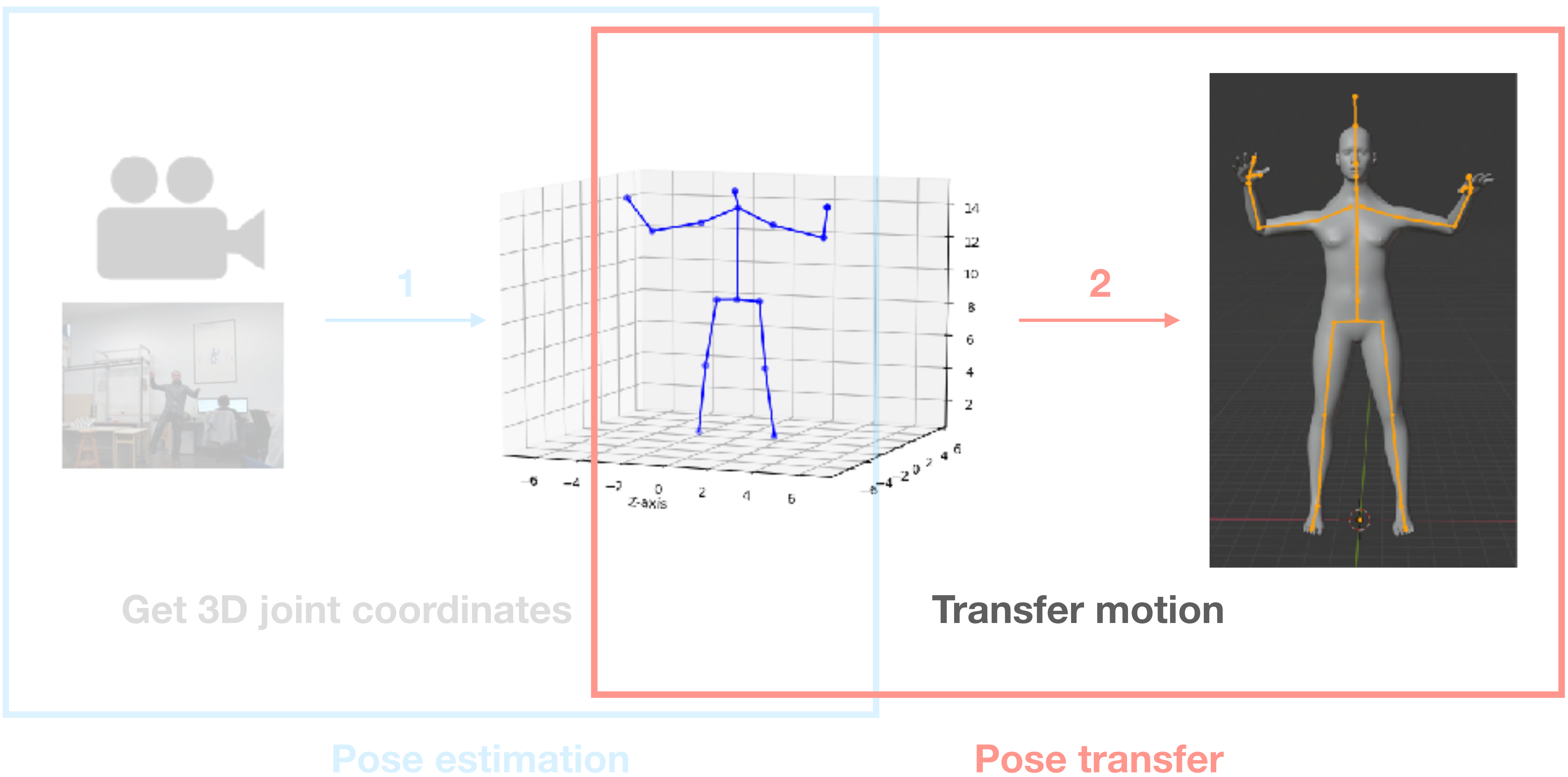
- Create a small dataset to train 2D-3D module
  - 6 human models of different sizes
  - 54 actions
  - 1 camera 640x480
- We extract 2D and 3D joint coordinates

3D human models

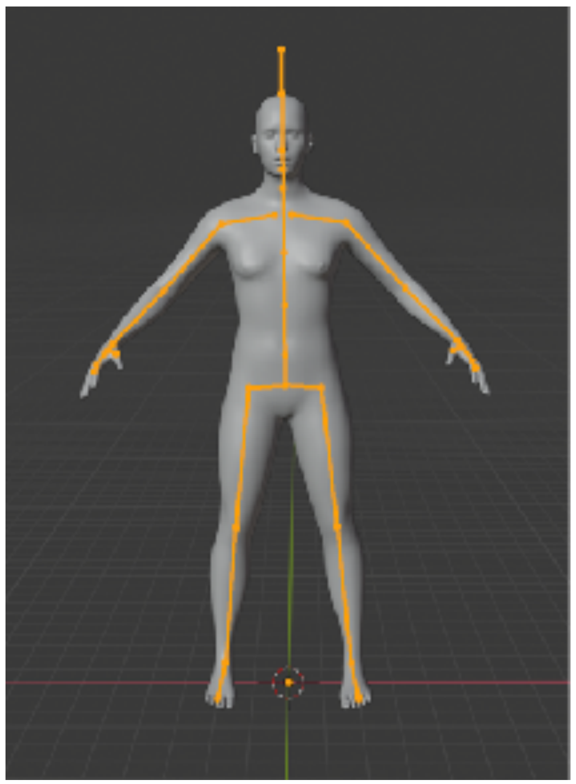
stand up action



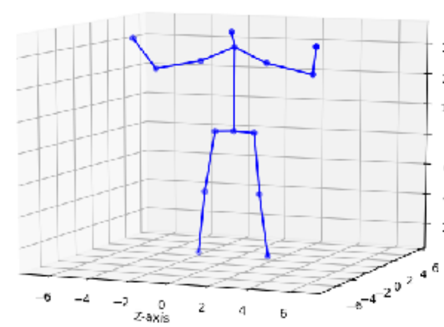
MoCap system



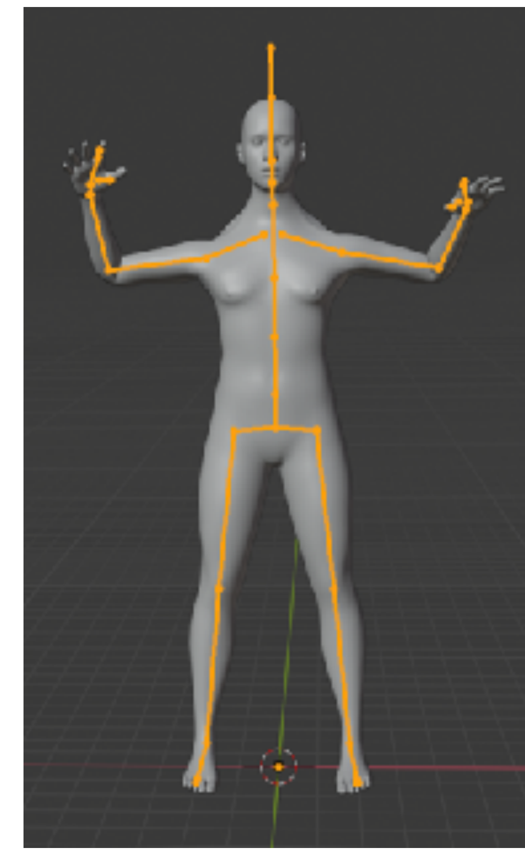
Calculate rotations



Reference



Detected 3D joints

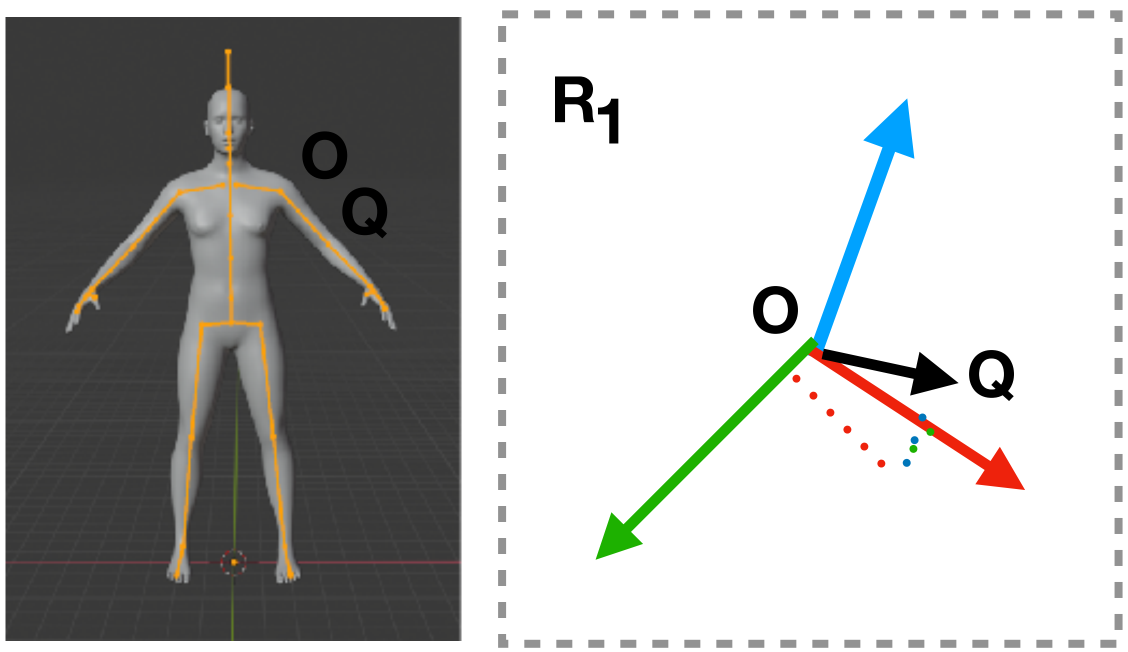


Retarget pose

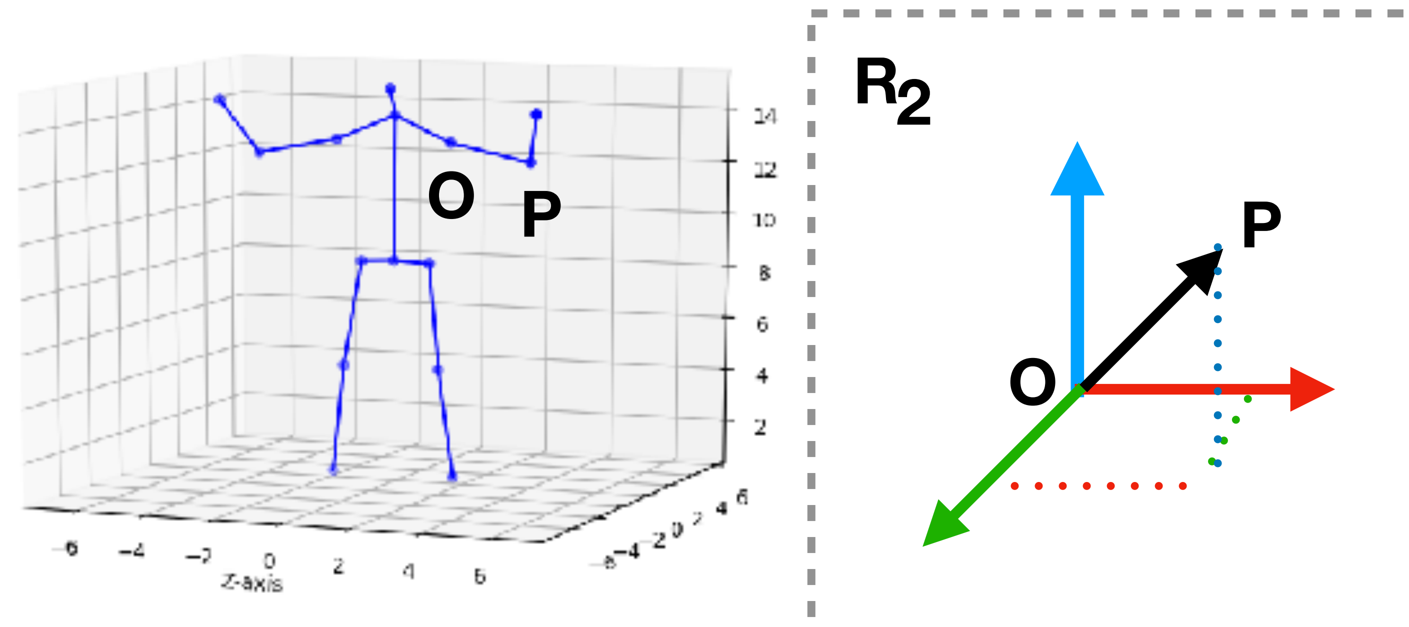
- Want to transform reference pose to the detected 3D joint coordinates
- Calculate rotations for each joint to match both skeletons
- Rotations need to be calculated in a hierarchical manner

Calculate rotations

- We want to find rotation angle between the different two reference systems:
- Need to scale and align human model with 3D joints

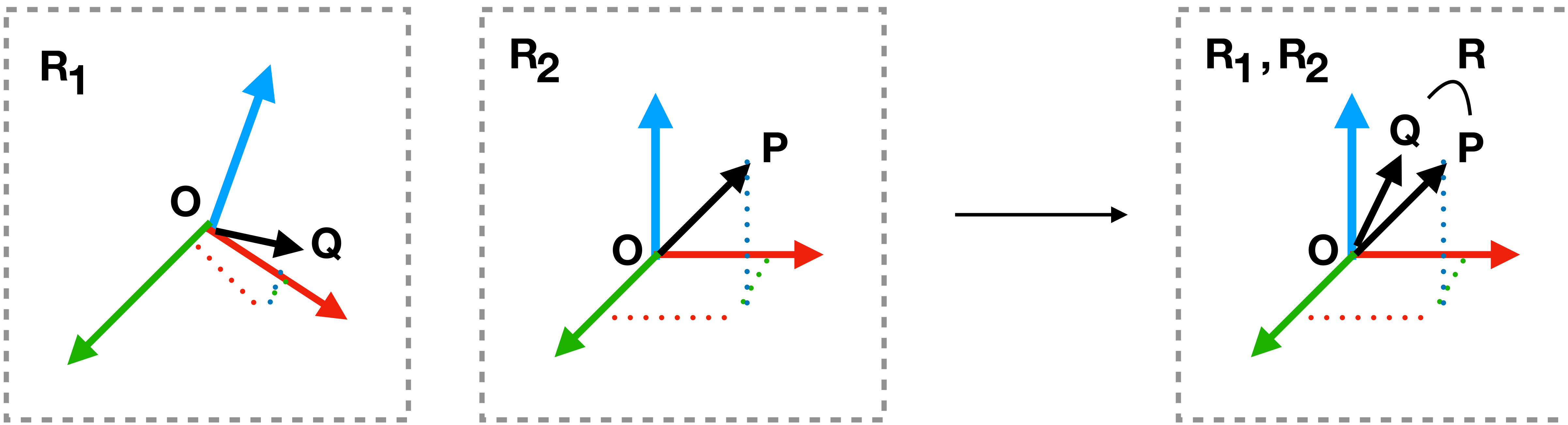


Joints from 3D human model



Detected 3D points from image

Calculate rotations

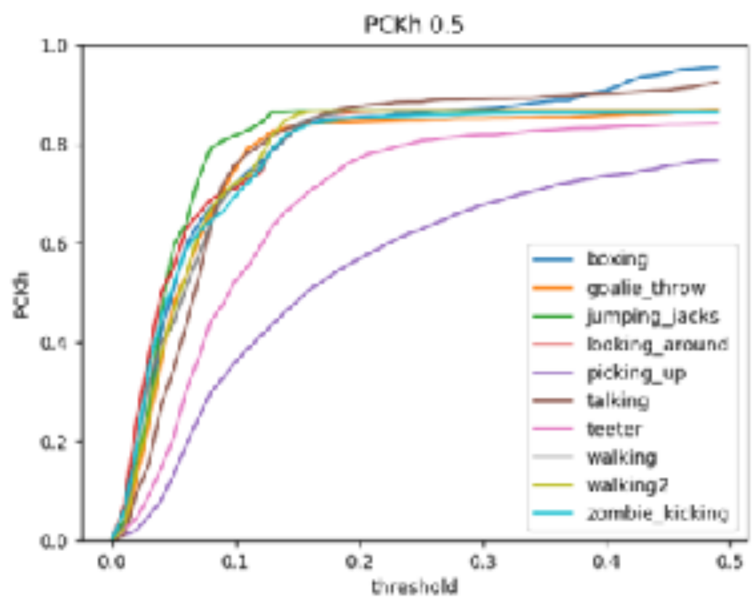


- We need to align reference system R1 and R2
- Then, rotation matrix R to go from Q to P can be found
- Finally, we need to convert global rotation R to local coordinates of the joint

Outline

- MoCap systems and applications
- Blender MoCap integration
    - From RGB image to 3D body joints
    - Match 3D human model to 3D skeleton movement
- Offline results with synthetic and real data

Quantitative results



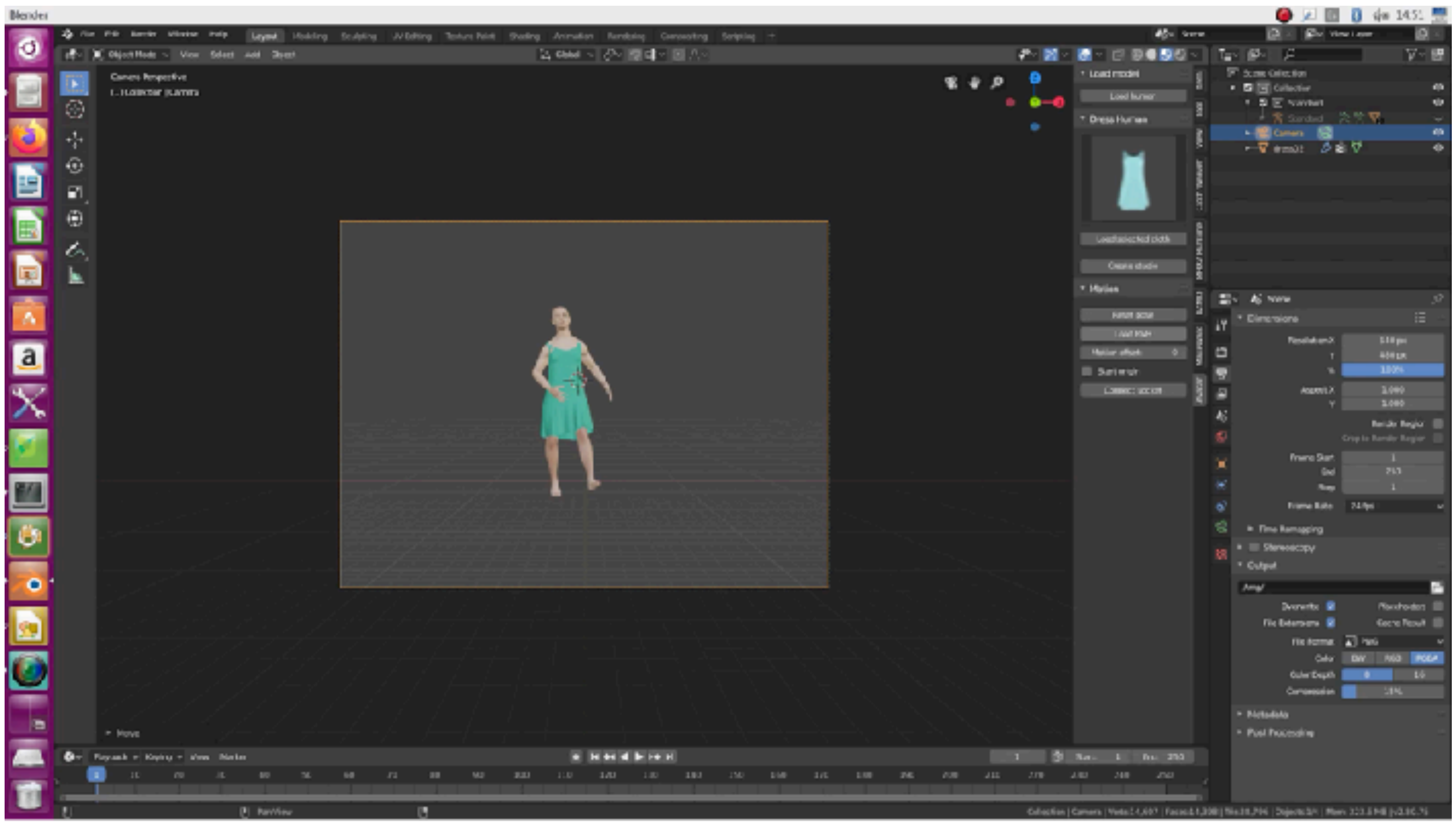
2D module

Action	Error (m)
Boxing	0.0961
Goalie throw	0.0989
Jumping jacks	0.1299
Look around	0.1585
Pick up	0.1399
Talking	0.0952
Teeter	0.1416
Walking	0.1621
Walking 2	0.1535
Zombie kicking	0.1288
Average	0.1304

3D module

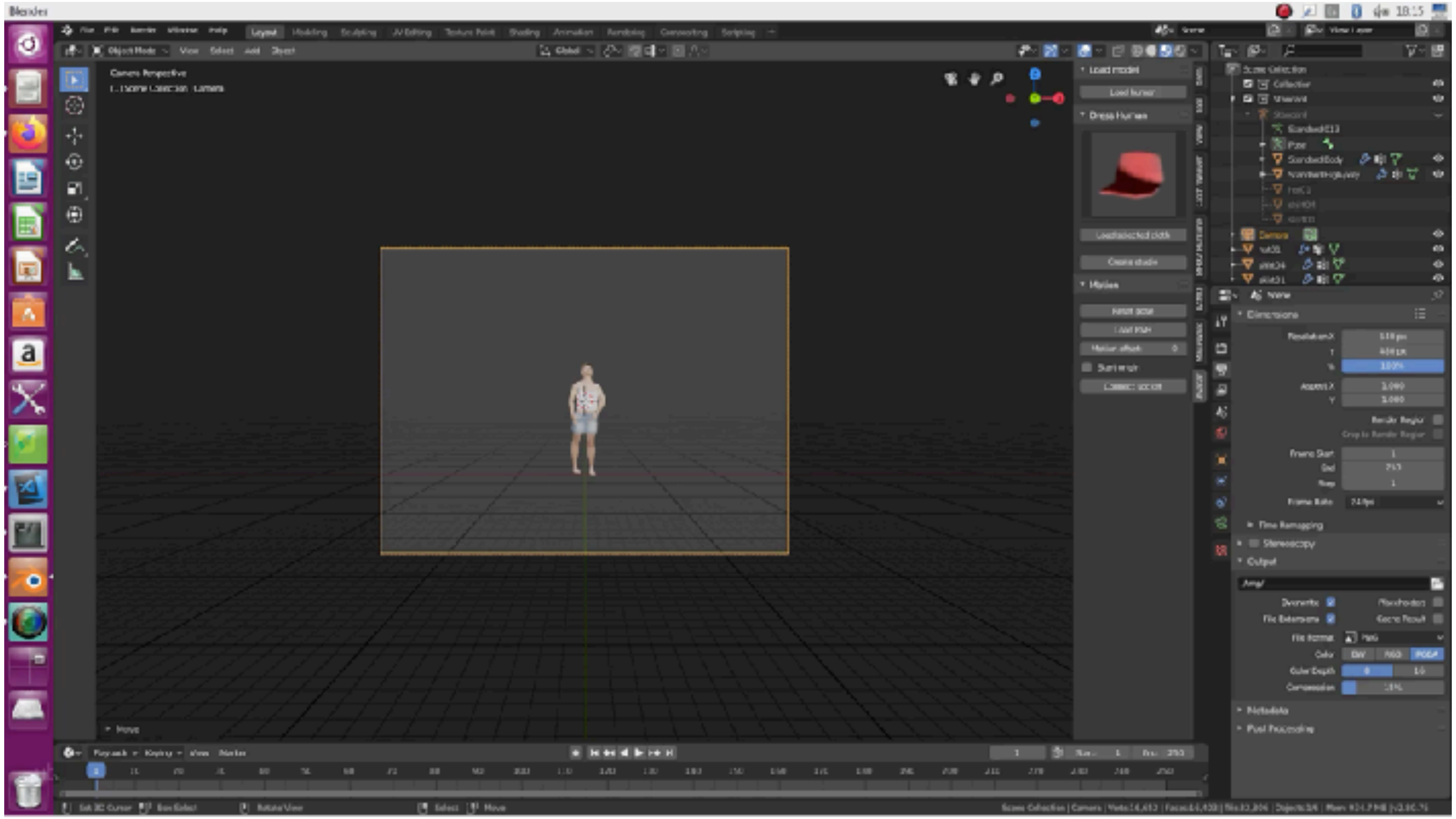
- **Tests performed on synthetic dataset**
- **Show only few actions**
- **Worst performing actions in 2D module have more self-occlusions**
- **Worst performing actions in 3D module have more variability**

Qualitative results



Synthetic sequence

Qualitative results



Real sequence

Thank you for your attention!!