Flying Drones with Gestures

Laleh Safari | Pascal Schambach | Christian Wegmann

AGENDA

AGENDA

DRONE STEERING: A USE CASE OF GESTURE RECOGNITION

GESTURE RECOGNITION USE CASES

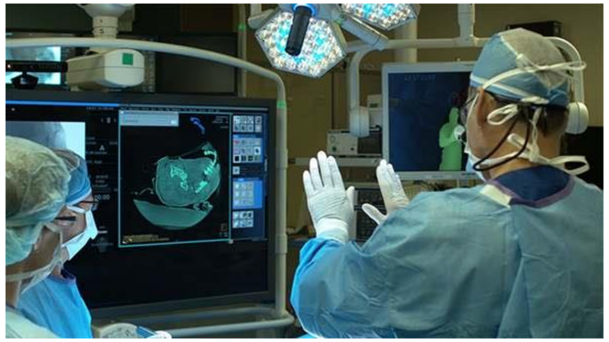
Gesture recognition technology can be integrated in perceptual user interfaces.

It can refer to any non-verbal communication that is intended to communicate a specific massage.

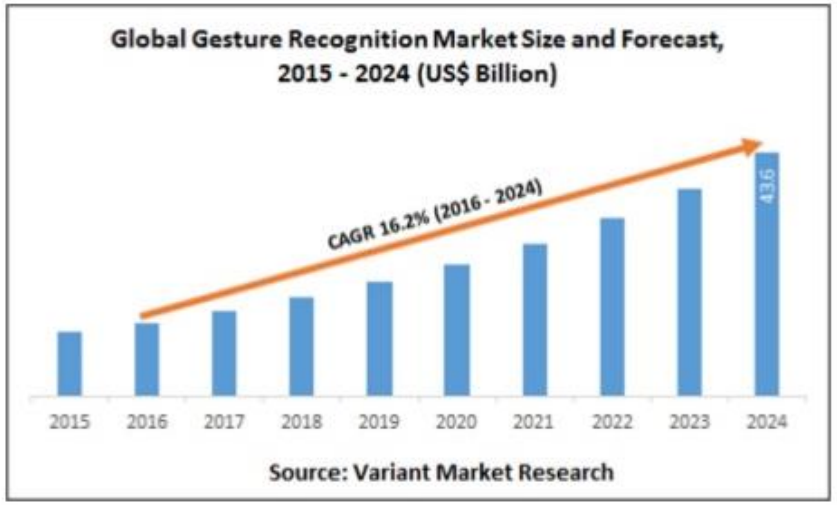
Car onboard system

Media use

Gaming


Medicine

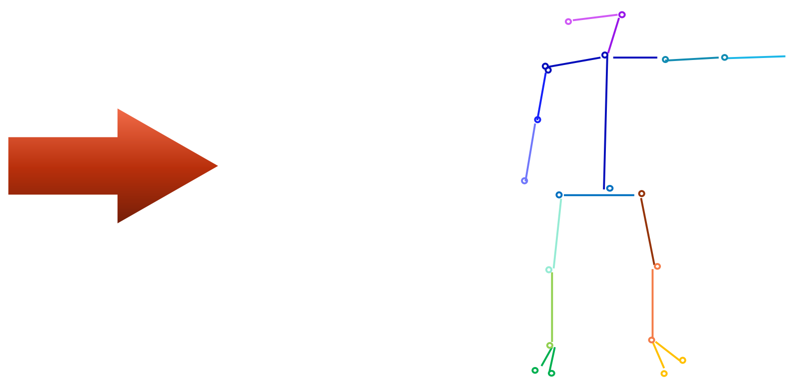
The global gesture recognition market is estimated to triple by 2024.



POSTURE (STATIC)

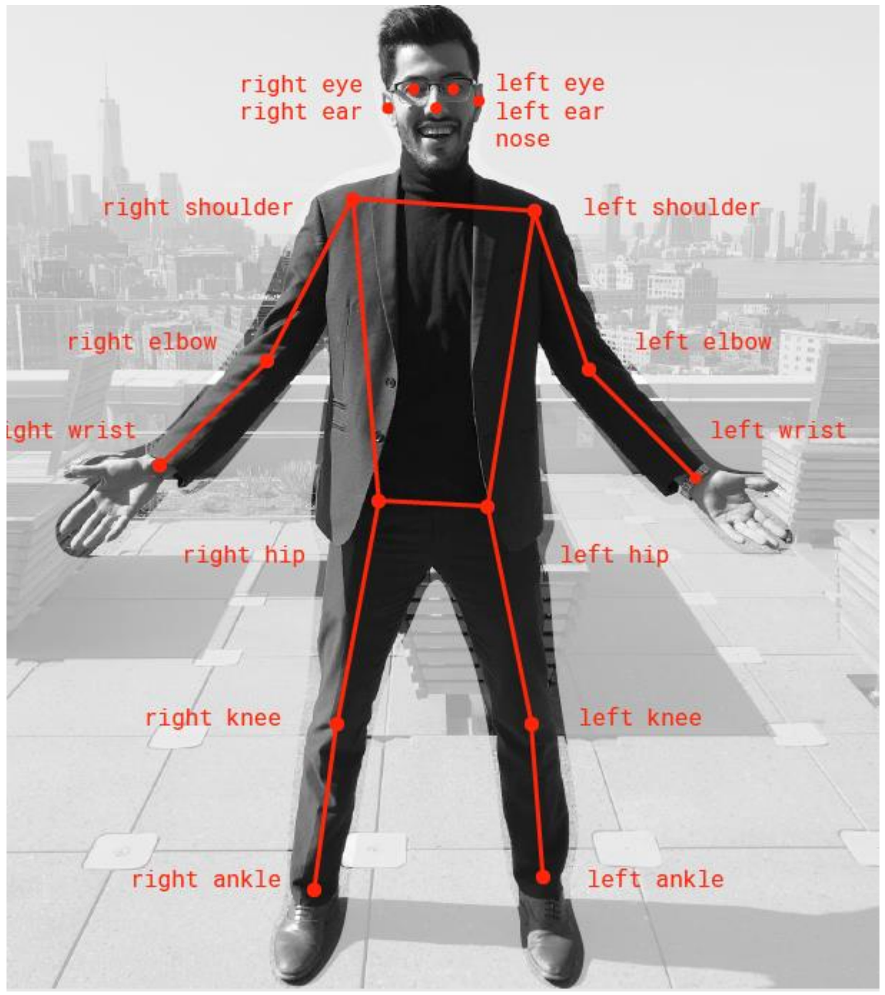
Postures is an image consist of x/y coordinates of human body joints.

Since a posture is static it is simple to handle and live predictions are faster. A posture can be used as simple command in perceptual user interfaces.

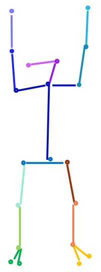


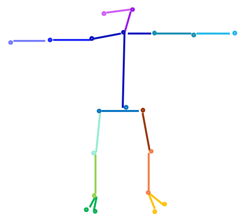
Label = right

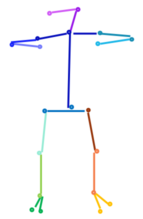
features = x_i, y_i, i = body joints

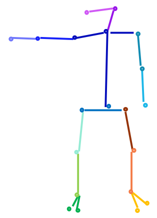


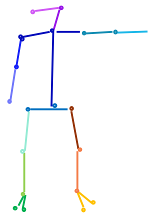
POSTURES (SEVEN CLASSES)

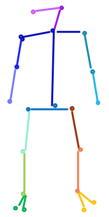

take_off


move


flip


left


right

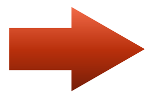

STOP
land

We classified everything which does not match exactly one of the above postures as 'no posture'.

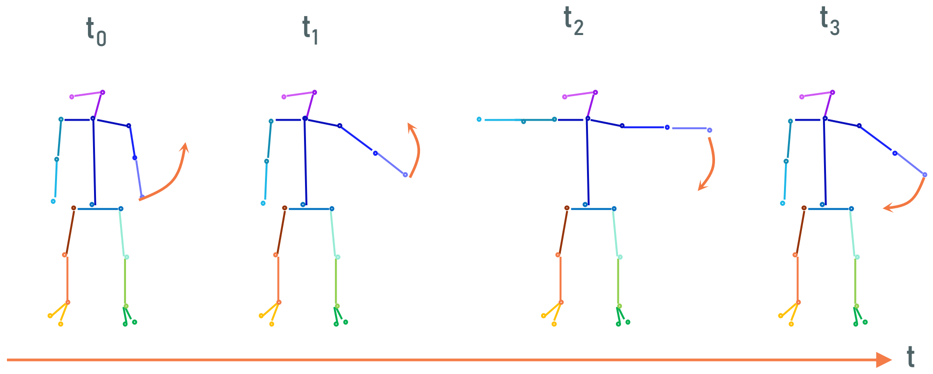
GESTURE (DYNAMIC)

On the other hand, a gesture is a time record of a set of poses.

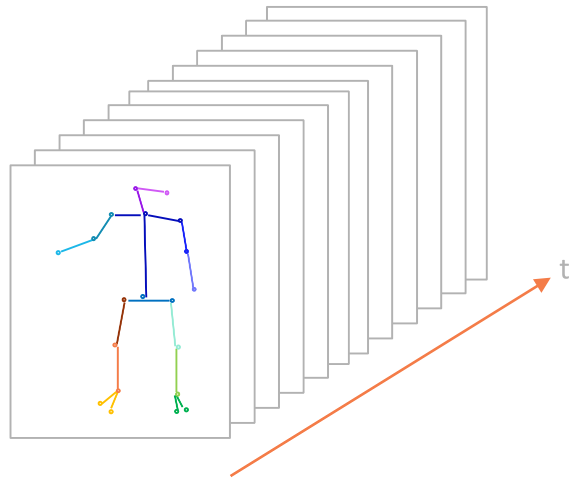
Gestures allow us to be much more expressive (e.g. sign language of the deaf) than with postures – but it also more difficult to detect and interpret.



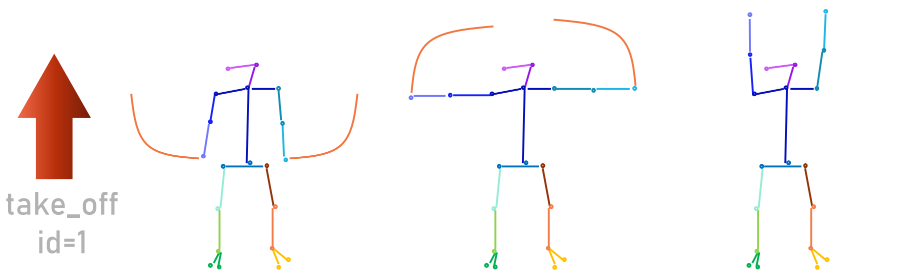
Label = right

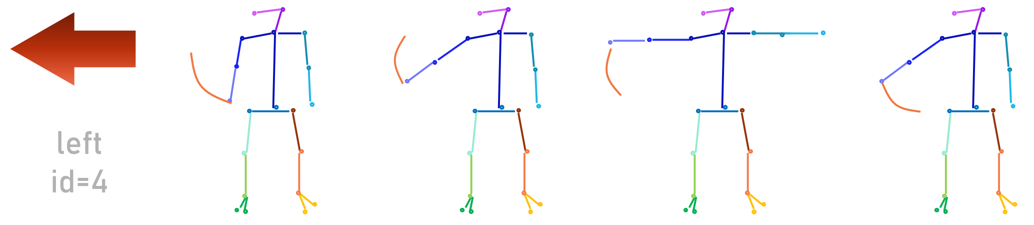


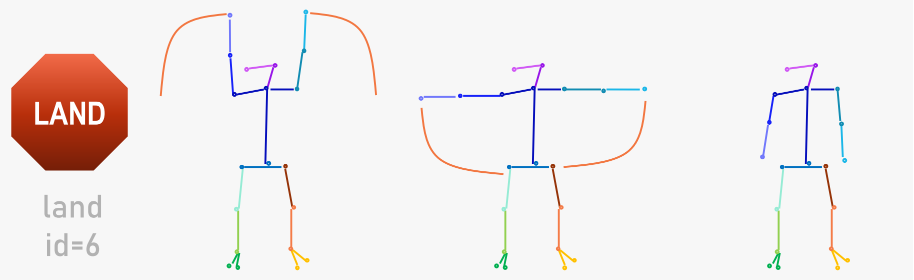
Features = x_i, y_i, t_i, i = body joints



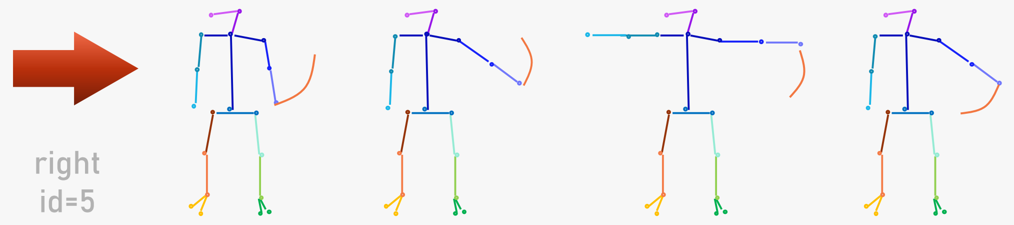
GESTURES (SEVEN CLASSES)

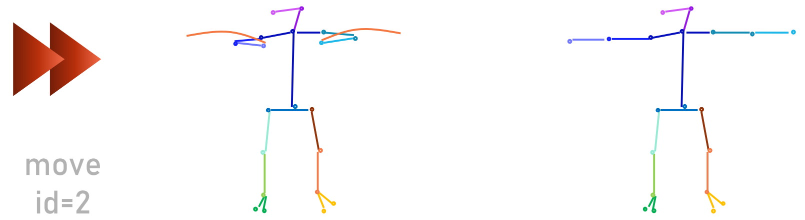

take_off
id=1

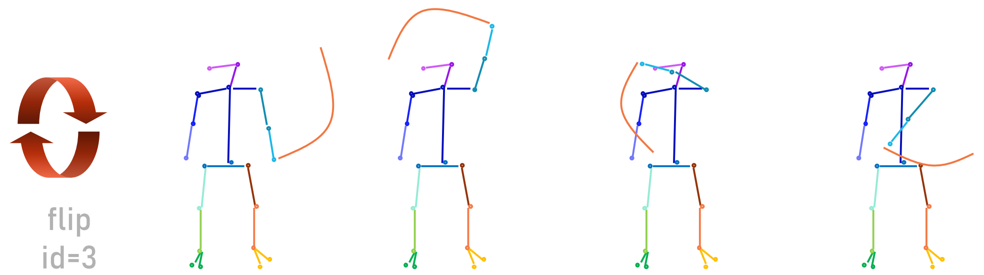

left
id=4



land
id=6


right
id=5


move
id=2

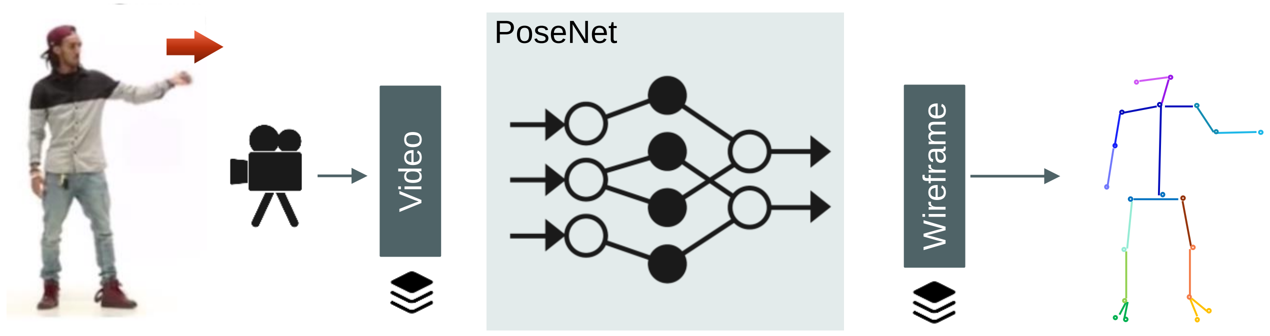

flip
id=3

We classified everything which does not match exactly one of the above gesture as ‘no gesture’.

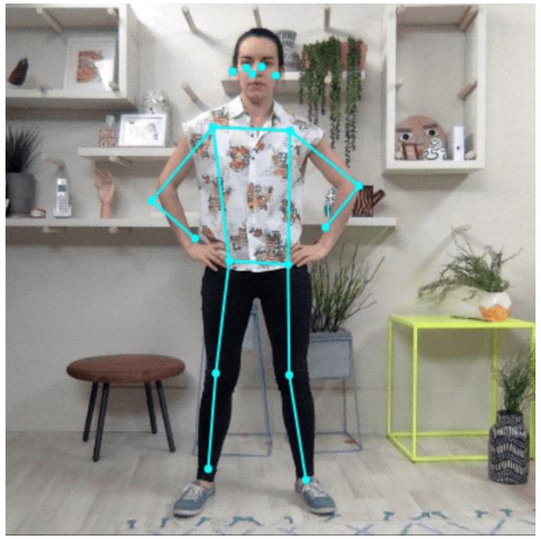
POSENET

In May 2018 the Google Creative Lab released a tensorflow.js version of PoseNet (see blog post). PoseNet is a convolutional neural network which in essence extracts the x/y coordinates of body parts from an input picture or video.

We did a form of transfer learning by building our gesture detection model on top of the PoseNet output, i.e. the body part coordinates.



…and this is how it looks:

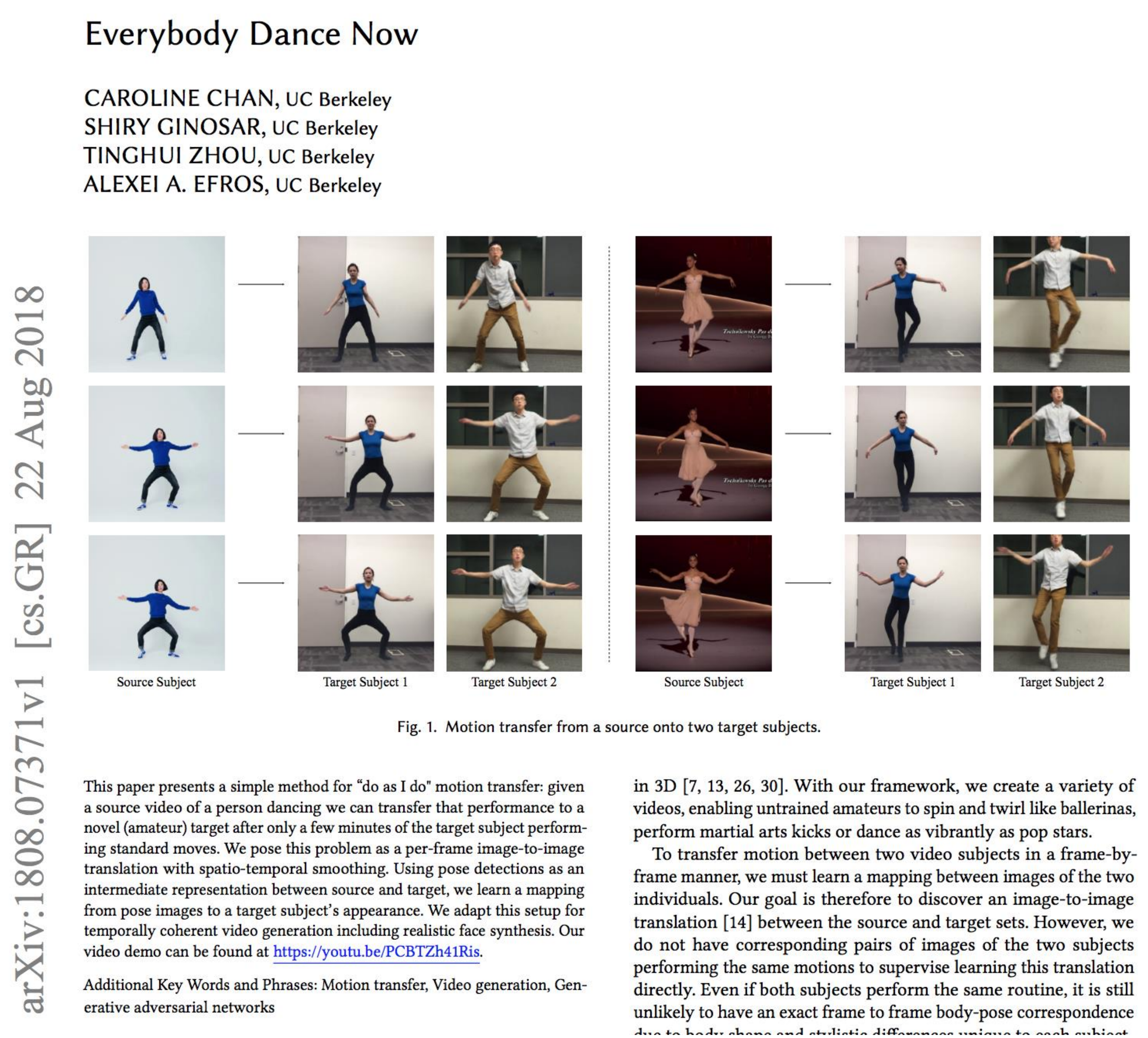


POSENET

In May 2018 the Google Creative Lab released a tensorflow.js version of PoseNet (see blog post). PoseNet is a convolutional neural network which in essence extracts the x/y coordinates of body parts from an input picture or video.

We did a form of transfer learning by building our gesture detection model on top of the PoseNet output, i.e. the body part coordinates.

Despite PoseNet being relatively new there are already a few very interesting research papers available, e.g. for live video adaptation.



Everybody Dance Now

CAROLINE CHAN, UC Berkeley
SHIRY GINOSAR, UC Berkeley
TINGHUI ZHOU, UC Berkeley
ALEXEI A. EFROS, UC Berkeley

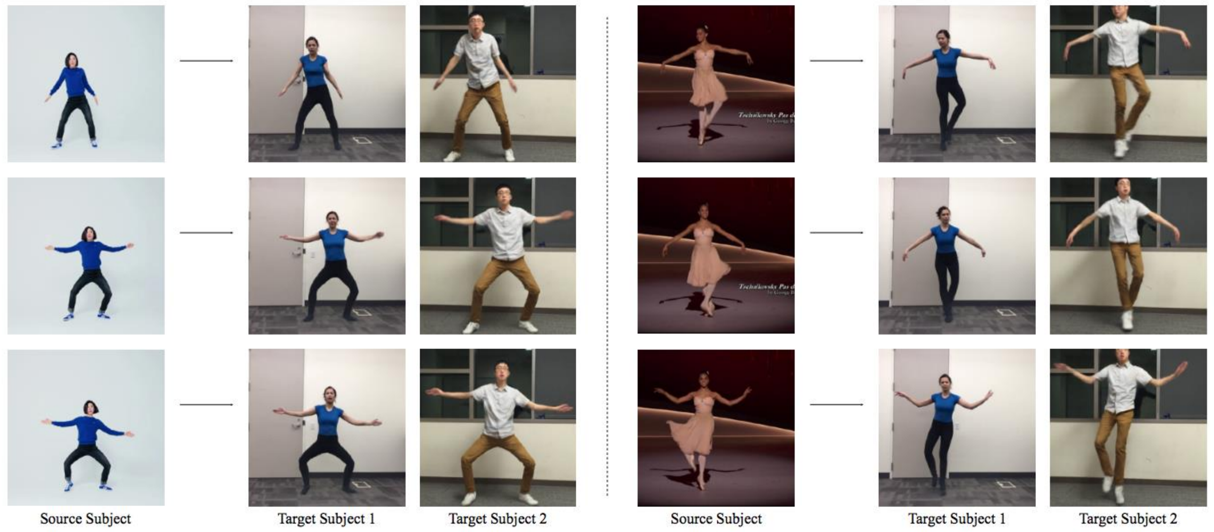


Fig. 1. Motion transfer from a source onto two target subjects.

This paper presents a simple method for "do as I do" motion transfer: given a source video of a person dancing we can transfer that performance to a novel (amateur) target after only a few minutes of the target subject performing standard moves. We pose this problem as a per-frame image-to-image translation with spatio-temporal smoothing. Using pose detections as an intermediate representation between source and target, we learn a mapping from pose images to a target subject's appearance. We adapt this setup for temporally coherent video generation including realistic face synthesis. Our video demo can be found at https://youtu.be/PCBTZh41Ris.

Additional Key Words and Phrases: Motion transfer, Video generation, Generative adversarial networks

in 3D [7, 13, 26, 30]. With our framework, we create a variety of videos, enabling untrained amateurs to spin and twirl like ballerinas, perform martial arts kicks or dance as vibrantly as pop stars.

To transfer motion between two video subjects in a frame-by-frame manner, we must learn a mapping between images of the two individuals. Our goal is therefore to discover an image-to-image translation [14] between the source and target sets. However, we do not have corresponding pairs of images of the two subjects performing the same motions to supervise learning this translation directly. Even if both subjects perform the same routine, it is still unlikely to have an exact frame to frame body-pose correspondence

AGENDA

MUCH MORE THAN A STANDALONE MACHINE LEARNING MODEL

We had to overcome several challenges:

Real-time Requirements

Flying a drone means every millisecond counts – or we crash into a wall. The model and data pipes therefore need to be stable and performant.

Python | JavaScript | Hardware

JS PY

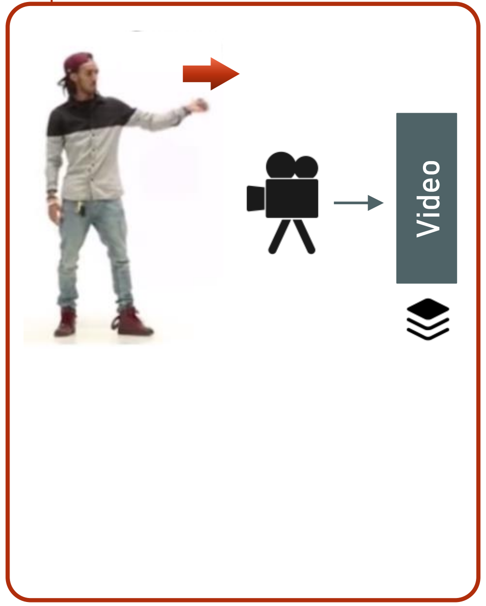
PoseNet runs in JavaScript. Training neural networks is much more mature in Python. Connecting to a drone API is needed. Bringing it all together is not simple.

Needs to Work Offline

A drone has its own wireless network. The native PoseNet runs on Google servers – we therefore had to make it run locally, including all libraries.

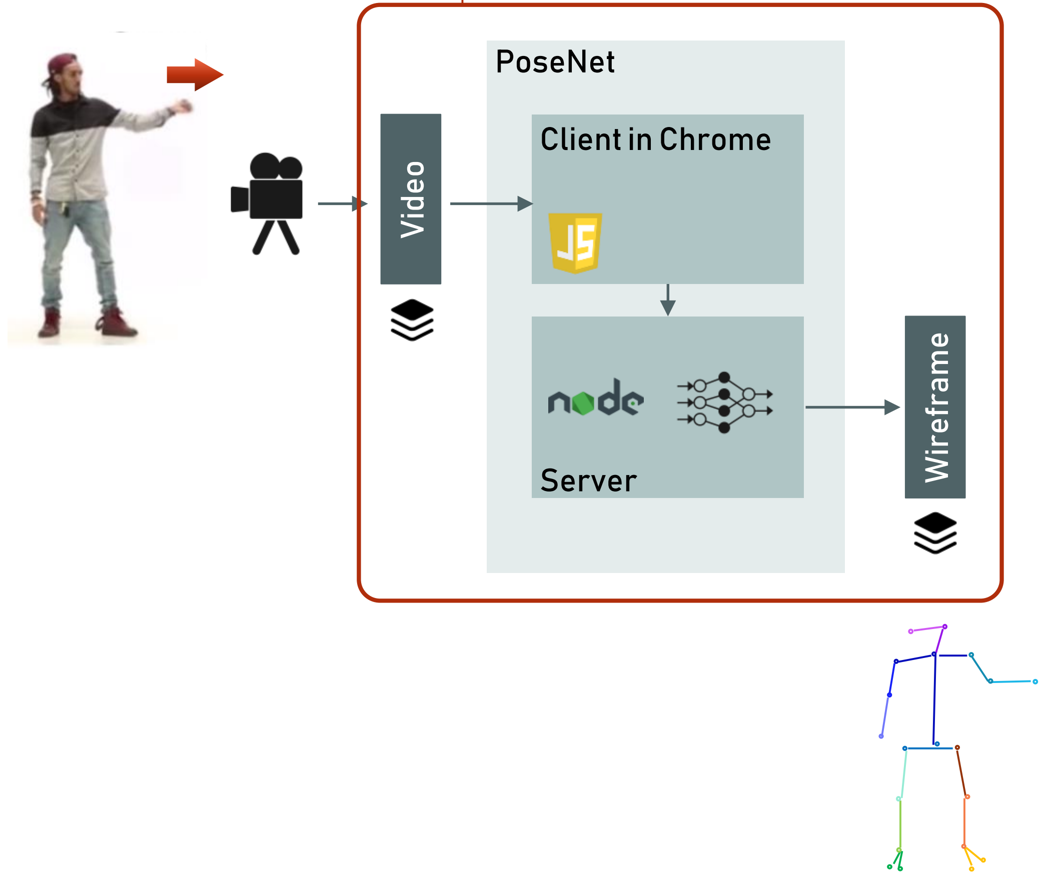
SYSTEM COMPONENTS

We use a webcam to record gestures from users in real-time.



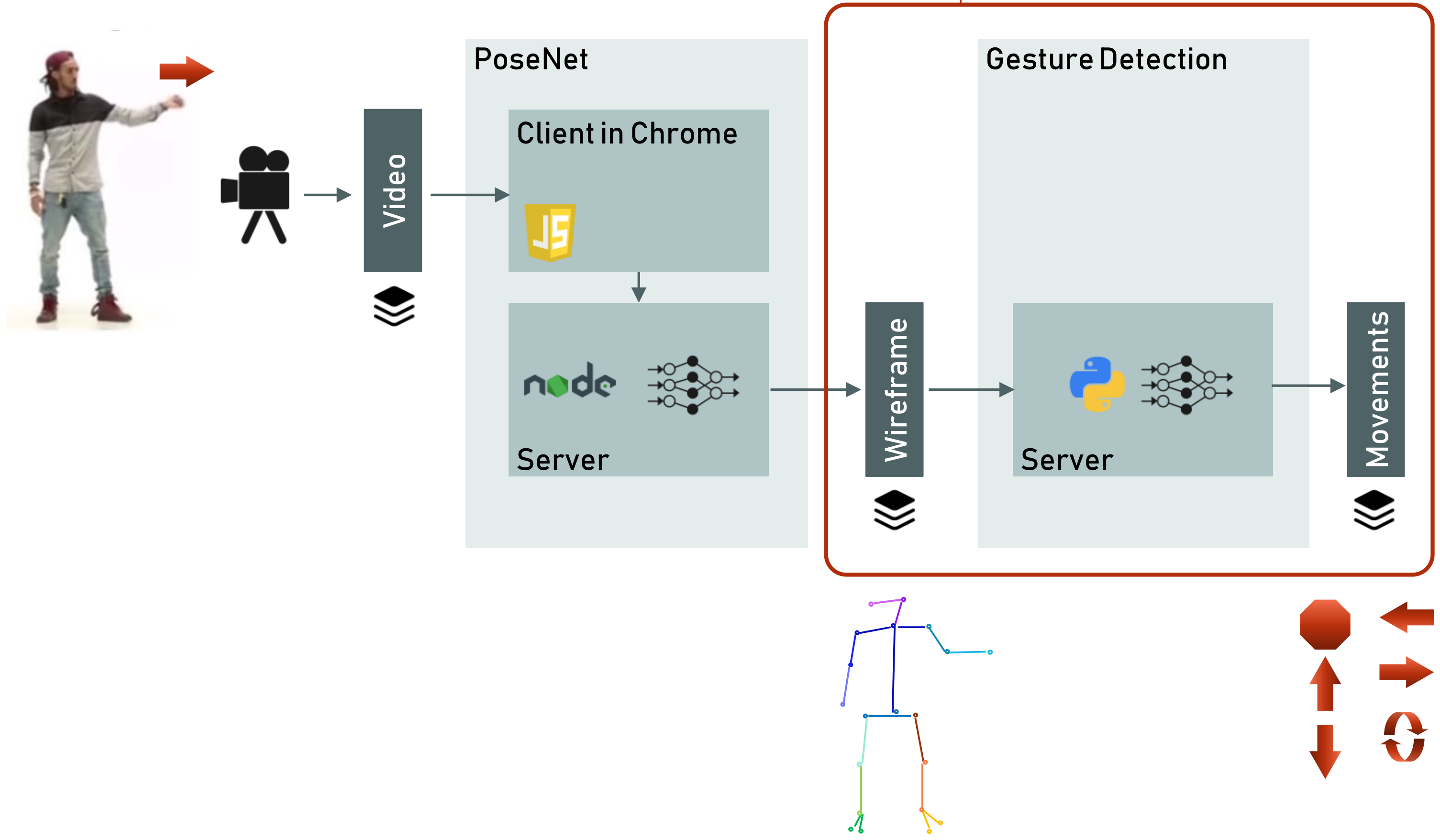
SYSTEM COMPONENTS

PoseNet runs on a local node.js server and is exposed via a website. It translates a video stream into a wireframe stream.



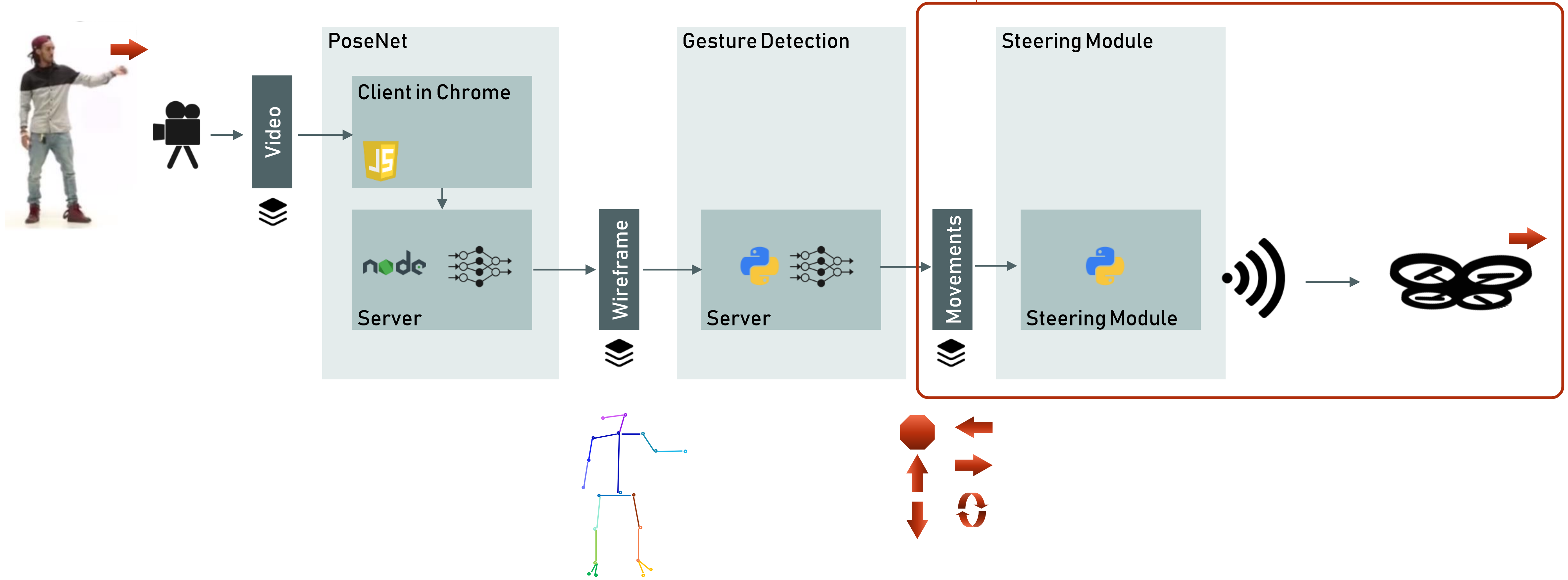
SYSTEM COMPONENTS

A Python websocket server consumes the wireframe stream and feeds it into a gesture model. This model extracts movements like take-off or left from the wireframe stream.



SYSTEM COMPONENTS

A steering module connect the drone API with the movement stream from the gesture model and ensures multi-threading and realtime movements.

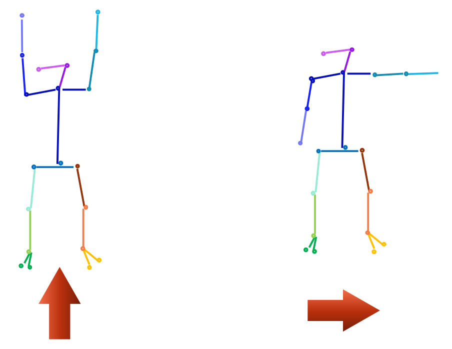


AGENDA

STARTING WITH (STATIC) POSTURES

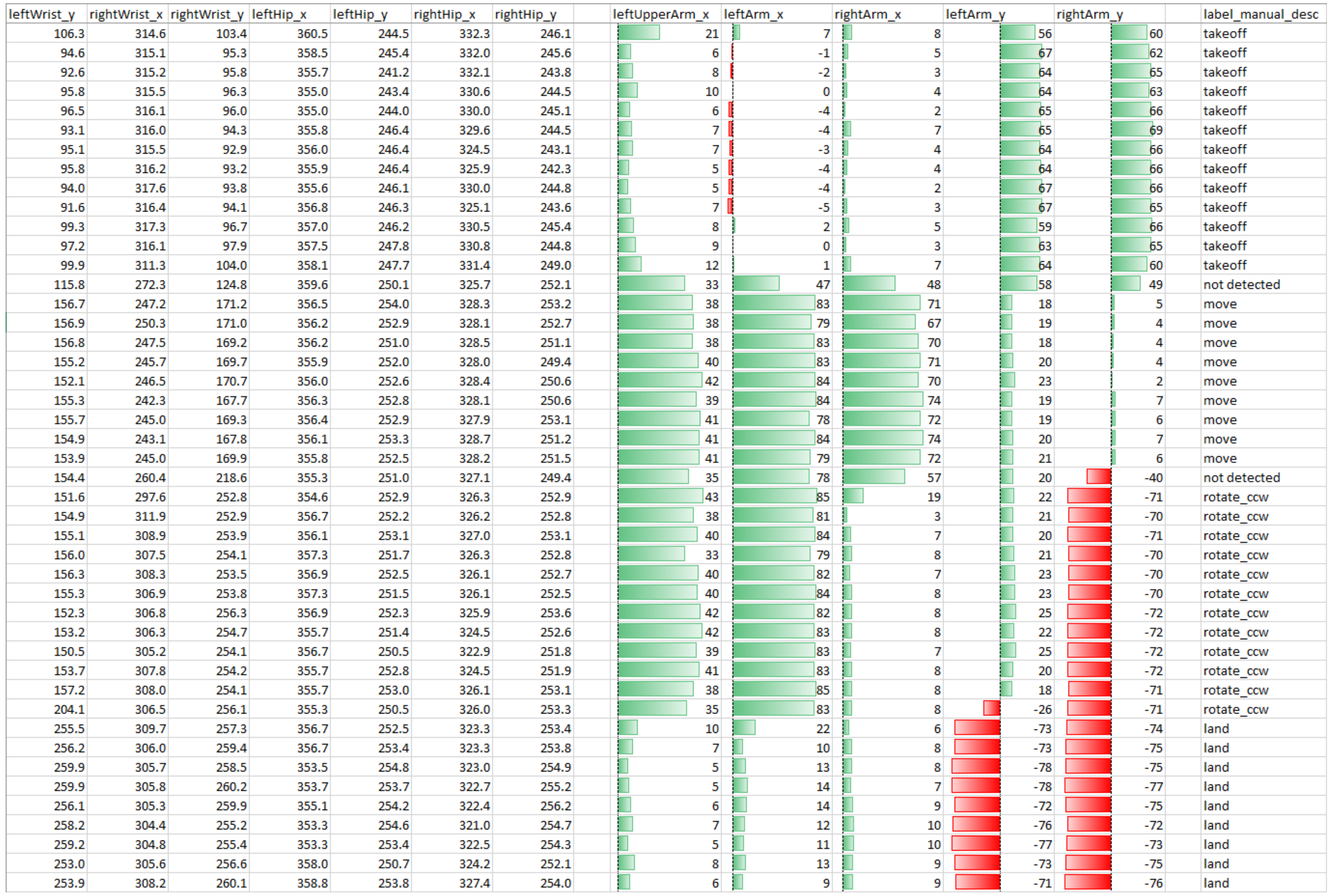
A Soft Start…

Static postures are much easier to generate, label and predict. We thus started a prototype with simple postures like the ones below.



The Data

PoseNet streams the x/y coordinates of body parts. We used these to easily label movements. We do not need to care about the duration between points in the stream or their sequential order.



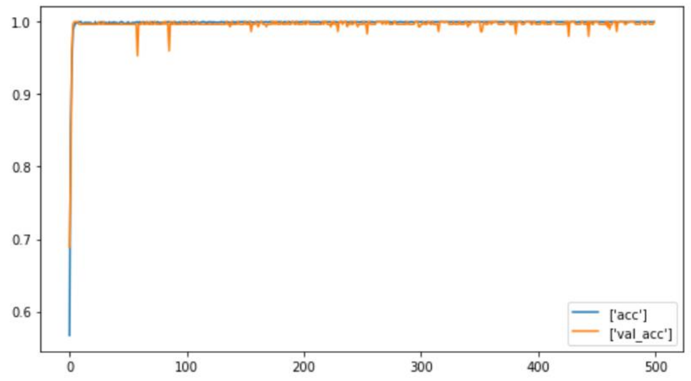
leftWrist_y	rightWrist_x	rightWrist_y	leftHip_x	leftHip_y	rightHip_x	rightHip_y	leftUpperArm_x	leftArm_x	rightArm_x	leftArm_y	rightArm_y	label_manual_desc
106.3	314.6	103.4	360.5	244.5	332.3	246.1	21	7	8	56	60	takeoff
94.6	315.1	95.3	358.5	245.4	332.0	245.6	6	-1	5	67	62	takeoff
92.6	315.2	95.8	355.7	241.2	332.1	243.8	8	-2	3	64	65	takeoff
95.8	315.5	96.3	355.0	243.4	330.6	244.5	10	0	4	64	63	takeoff
96.5	316.1	96.0	355.0	244.0	330.0	245.1	6	-4	2	65	66	takeoff
93.1	316.0	94.3	355.8	246.4	329.6	244.5	7	-4	7	65	69	takeoff
95.1	315.5	92.9	356.0	246.4	324.5	243.1	7	-3	4	64	66	takeoff
95.8	316.2	93.2	355.9	246.4	325.9	242.3	5	-4	4	64	66	takeoff
94.0	317.6	93.8	355.6	246.1	330.0	244.8	5	-4	2	67	66	takeoff
91.6	316.4	94.1	356.8	246.3	325.1	243.6	7	-5	3	67	65	takeoff
99.3	317.3	96.7	357.0	246.2	330.5	245.4	8	2	5	59	66	takeoff
97.2	316.1	97.9	357.5	247.8	330.8	244.8	9	0	3	63	65	takeoff
99.9	311.3	104.0	358.1	247.7	331.4	249.0	12	1	7	64	60	takeoff
115.8	272.3	124.8	359.6	250.1	325.7	252.1	33	47	48	58	49	not detected
156.7	247.2	171.2	356.5	254.0	328.3	253.2	38	83	71	18	5	move
156.9	250.3	171.0	356.2	252.9	328.1	252.7	38	79	67	19	4	move
156.8	247.5	169.2	356.2	251.0	328.5	251.1	38	83	70	18	4	move
155.2	245.7	169.7	355.9	252.0	328.0	249.4	40	83	71	20	4	move
152.1	246.5	170.7	356.0	252.6	328.4	250.6	42	84	70	23	2	move
155.3	242.3	167.7	356.3	252.8	328.1	250.6	39	84	74	19	7	move
155.7	245.0	169.3	356.4	252.9	327.9	253.1	41	78	72	19	6	move
154.9	243.1	167.8	356.1	253.3	328.7	251.2	41	84	74	20	7	move
153.9	245.0	169.9	355.8	252.5	328.2	251.5	41	79	72	21	6	move
154.4	260.4	218.6	355.3	251.0	327.1	249.4	35	78	57	20	-40	not detected
151.6	297.6	252.8	354.6	252.9	326.3	252.9	43	85	19	22	-71	rotate_ccw
154.9	311.9	252.9	356.7	252.2	326.2	252.8	38	81	3	21	-70	rotate_ccw
155.1	308.9	253.9	356.1	253.1	327.0	253.1	40	84	7	20	-71	rotate_ccw
156.0	307.5	254.1	357.3	251.7	326.3	252.8	33	79	8	21	-70	rotate_ccw
156.3	308.3	253.5	356.9	252.5	326.1	252.7	40	82	7	23	-70	rotate_ccw
155.3	306.9	253.8	357.3	251.5	326.1	252.5	40	84	8	23	-70	rotate_ccw
152.3	306.8	256.3	356.9	252.3	325.9	253.6	42	82	8	25	-72	rotate_ccw
153.2	306.3	254.7	355.7	251.4	324.5	252.6	42	83	8	22	-72	rotate_ccw
150.5	305.2	254.1	356.7	250.5	322.9	251.8	39	83	7	25	-72	rotate_ccw
153.7	307.8	254.2	355.7	252.8	324.5	251.9	41	83	8	20	-72	rotate_ccw
157.2	308.0	254.1	355.7	253.0	326.1	253.1	38	85	8	18	-71	rotate_ccw
204.1	306.5	256.1	355.3	250.5	326.0	253.3	35	83	8	-26	-71	rotate_ccw
255.5	309.7	257.3	356.7	252.5	323.3	253.4	10	22	6	-73	-74	land
256.2	306.0	259.4	356.7	253.4	323.3	253.8	7	10	8	-73	-75	land
259.9	305.7	258.5	353.5	254.8	323.0	254.9	5	13	8	-78	-75	land
259.9	305.8	260.2	353.7	253.7	322.7	255.2	5	14	7	-78	-77	land
256.1	305.3	259.9	355.1	254.2	322.4	256.2	6	14	9	-72	-75	land
258.2	304.4	255.2	353.3	254.6	321.0	254.7	7	12	10	-76	-72	land
259.2	304.8	255.4	353.3	253.4	322.5	254.3	5	11	10	-77	-73	land
253.0	305.6	256.6	358.0	250.7	324.2	252.1	8	13	9	-73	-75	land
253.9	308.2	260.1	358.8	253.8	327.4	254.0	6	9	9	-71	-76	land

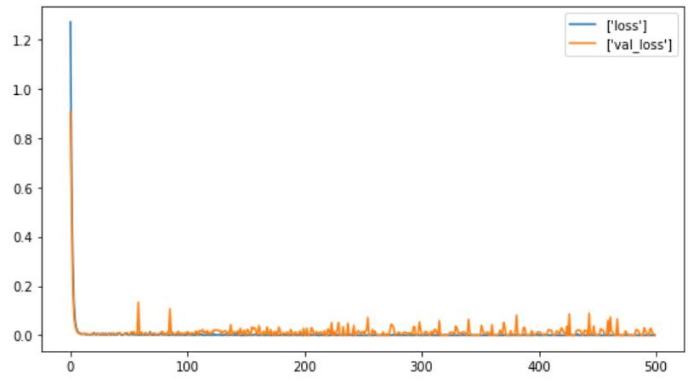
STARTING WITH (STATIC) POSTURES

Using a Neural Network for Postures

We used a fully connected neural network to detect postures. As expected, the model detected postures almost perfectly (see accuracy plot).

Using a neural network for postures feels like using a sledgehammer for cracking a nut. We did not tune the model further and started with gestures.



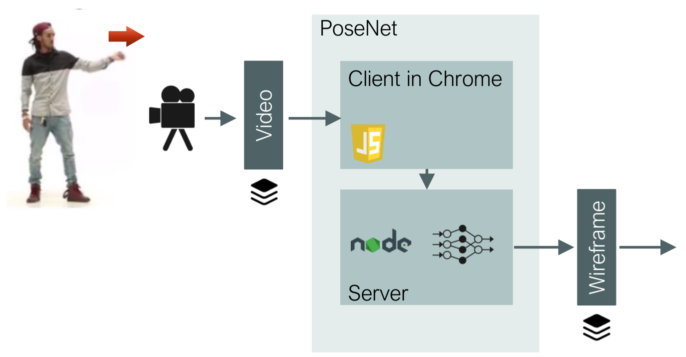


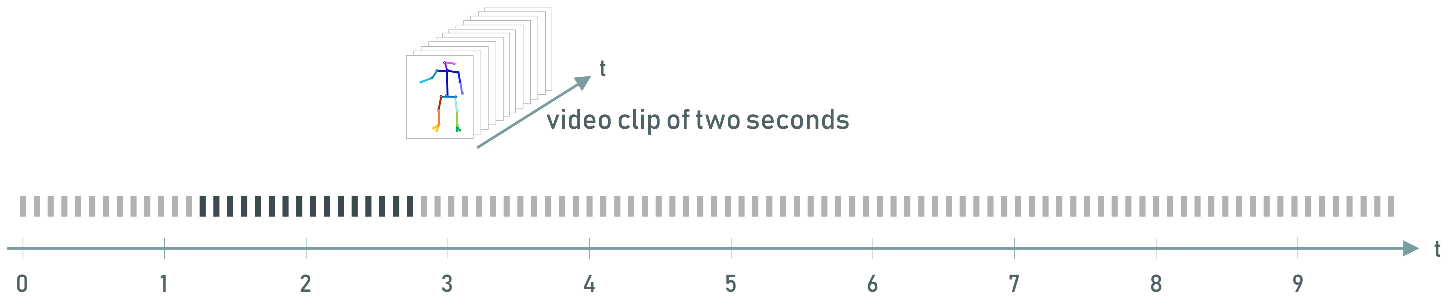
What Did we Learn?

- We set up the data preparation pipelines properly
- We got a feeling that gestures actually might work
- We had a model that helped us connecting the other components

But now: let’s see how we approached the more interesting part of dynamic gestures!

LABELLING



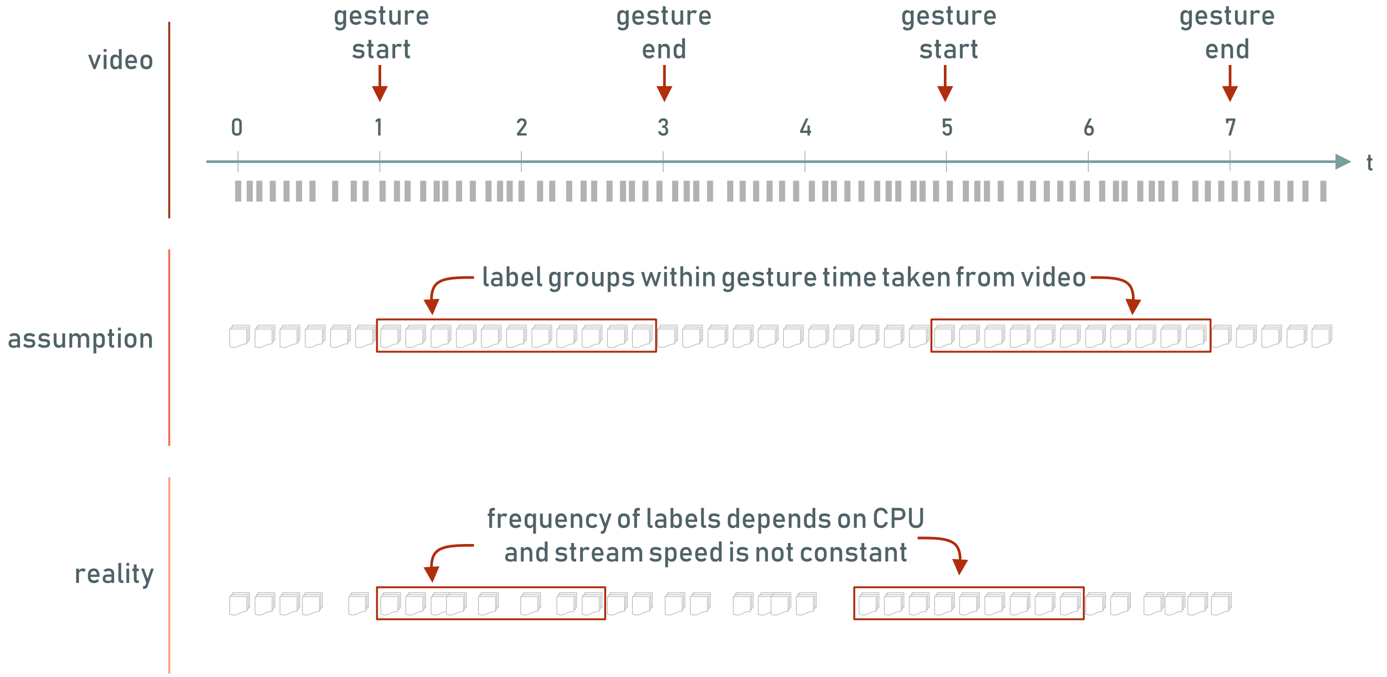


First Try: Time-Based Labelling

We noted down start and end of a gesture in the video and applied this to the frames of the stream.

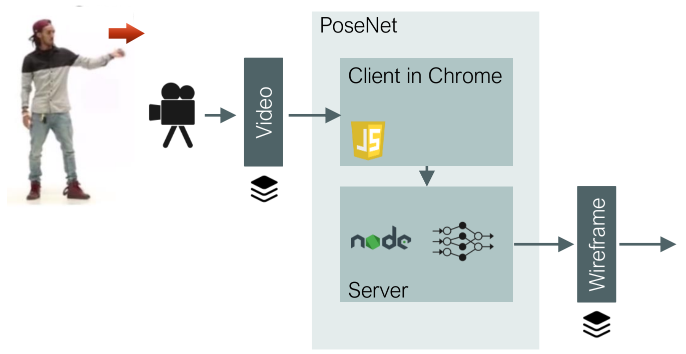
We struggled with two issues:

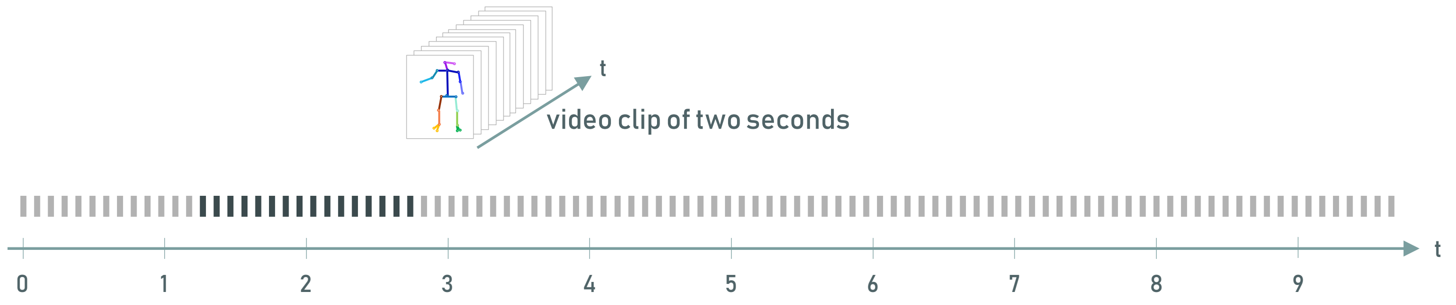
- Stream speed variations made allocation of frames to point in time difficult
- Even overall length of stream did not match the video length



video
gesture start
gesture end
gesture start
gesture end
0
1
2
3
4
5
6
7
t
assumption
label groups within gesture time taken from video
reality
frequency of labels depends on CPU and stream speed is not constant

LABELLING

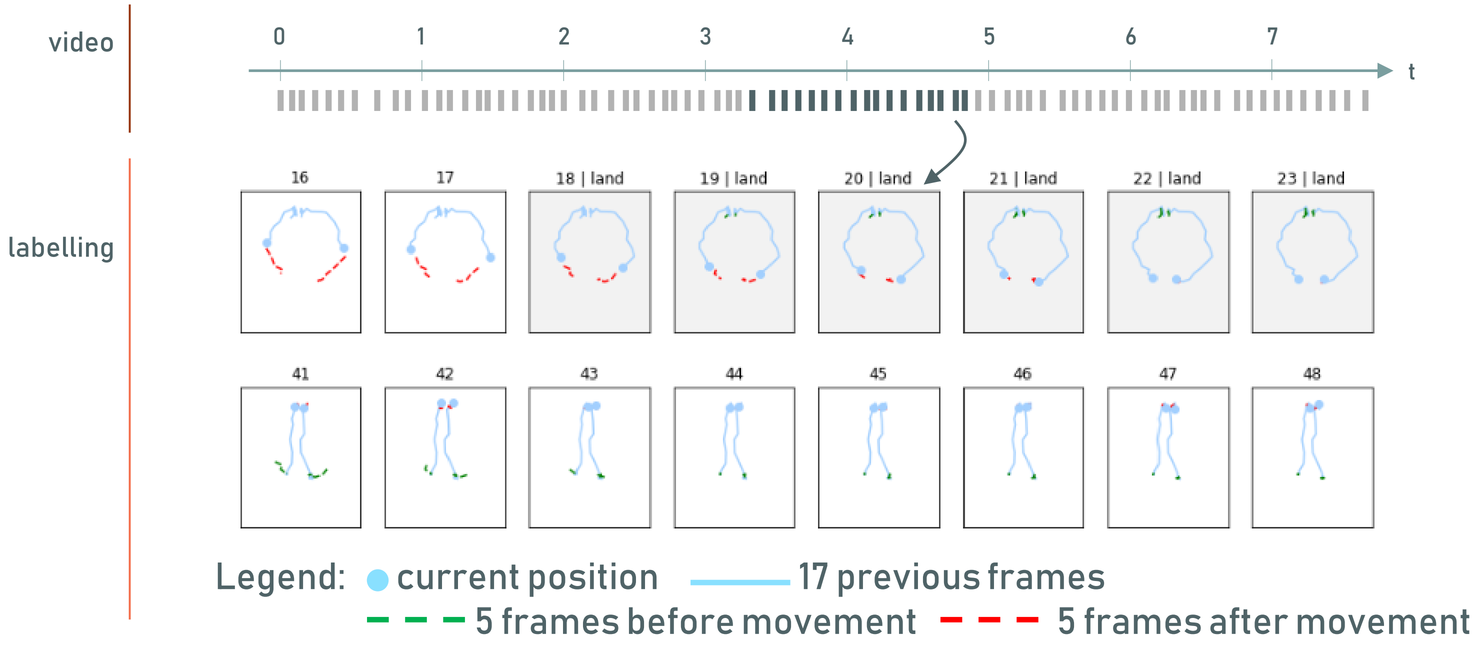




Second Try: Frame-Based Labelling

We looked at frame sets one by one and labelled them, with no regard for the actual time. For this we needed to visualize a video clip in a picture.

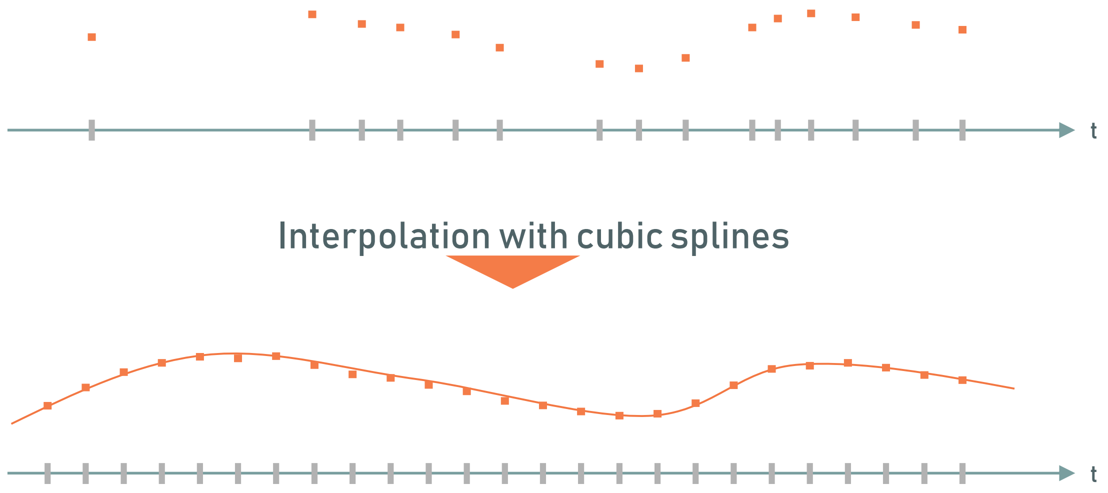
Here we see the position of the left and right wrist over time; grey means labelled.



DATA IMPROVEMENT 1 / 2

1 Interpolation

PoseNet sends coordinates with changing intervals. We though interpolating them makes it easier for the model.

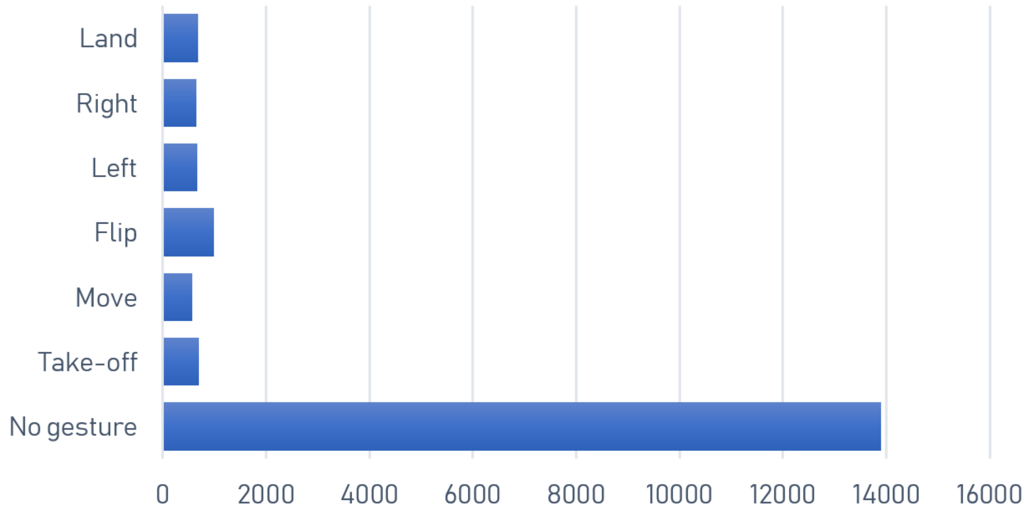


While the theoretical accuracy increased we did not notice any improvement in practical tests with live streaming data.

▶ We decided to not use interpolation to reduce complexity

2 Up- / Downsampling

We had unbalanced data: ~75% of our data was labelled as 'no gesture'.



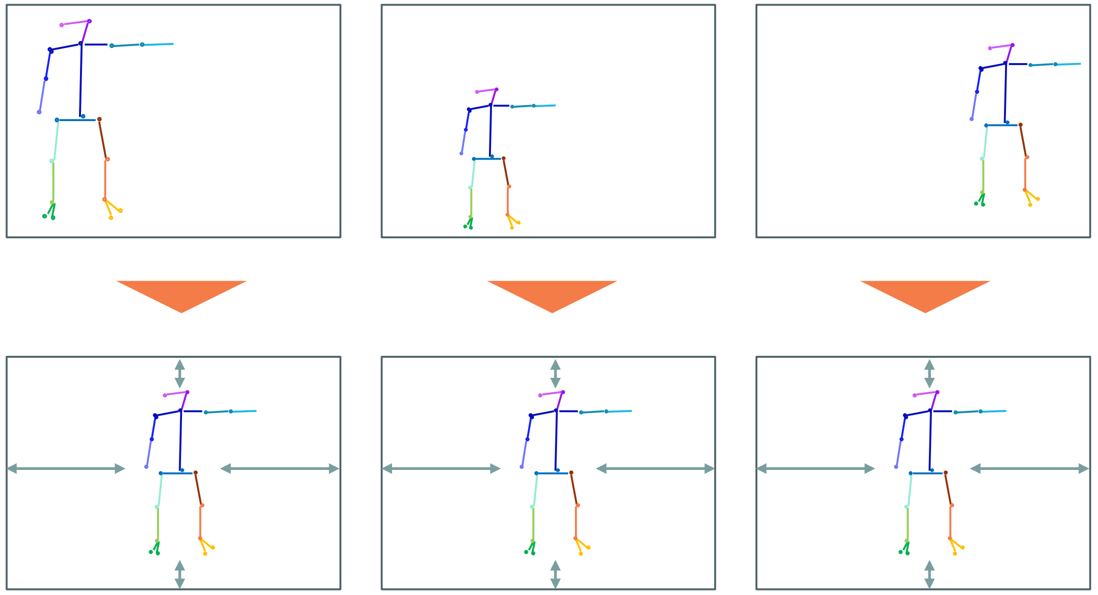
- We downsampled the 'no gesture' data by removing ambiguous ones
- We upsampled the other data using replacement sampling

▶ Sampling improved our accuracy noticeably

DATA IMPROVEMENT 2 / 2

3 Centralizing / Scaling

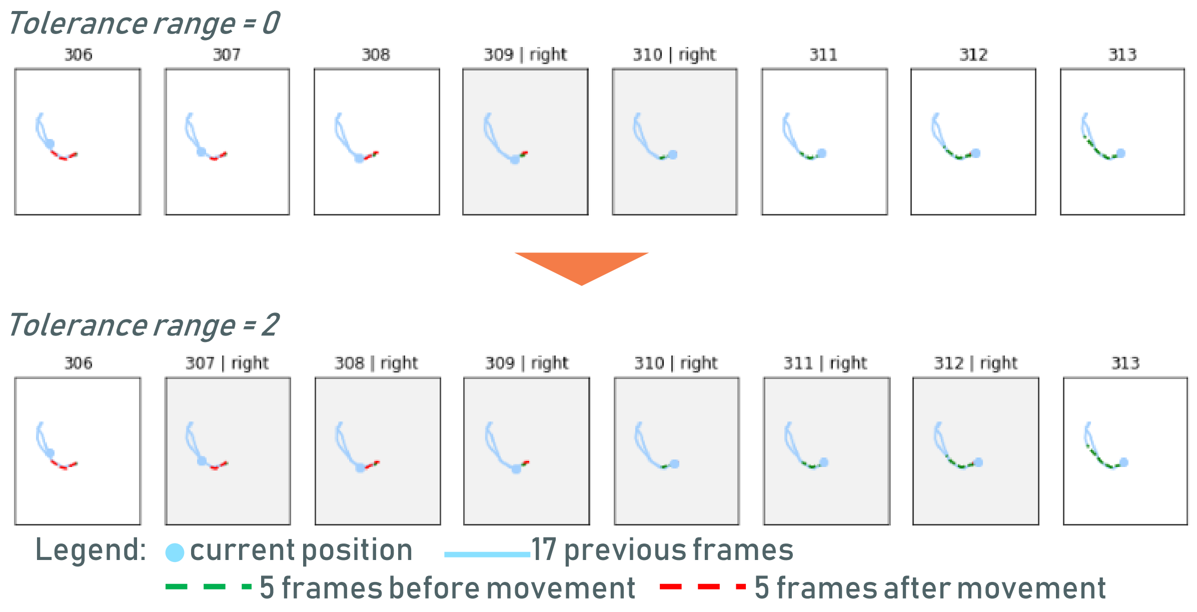
PoseNet returns absolute coordinates. As the subject moves around, the measurements differ. We thus centralized and scaled the coordinates.



▶ Normalization proved to be the single most effective data improvement

4 Tolerance range adaptation

We labelled movements exactly. The result was that the model required very precise inputs. We relaxed the requirement and added a tolerance range.



▶ Substantial impact on model performance

MODEL SELECTION

1	Random Forest	Usually robust, generalize well and do not need a lot of hyperparameter tuning. Candidate for baseline classifier	Test Accuracy: 0.95
2	Support Vector Machine	Known to work well for classification tasks when there is not a lot of data	Test Accuracy: 0.93
3	Long Short-Term Memory (LSTM)	Type of recurrent neural network (RNN) that can store past information Suited for sequence classification	Test Accuracy: 0.97

In theory all our Models achieved high accuracy. However, this did not automatically translate to high accuracy on classifying live streaming data.

AGENDA

CONCLUSION

- We implemented a minimum viable product in the field of gesture recognition
- We used knowledge transfer by building on top of PoseNet
- Using PoseNet outcomes we build our own machinery for data acquisition
- We used several feature engineering technics to compensate for lack of data
- We achieved a high theoretical accuracy ~ 97 %

POSSIBLE IMPROVEMENTS AND NEXT STEPS

Data
- Collect more (and more representative) training data
- Train more complex and different length gestures
- Use data augmentation for time series[1]
- Extract wireframes several times for a given video

Drone
- Use advanced drone
- Capture video signal directly from drone

Performance
- Use better hardware
- Optimize runtime performance for live prediction
- Use more sophisticated model[2]

1 Data augmentation using synthetic data for time series classification with deep residual networks [arXiv:1808.02455v1 [cs.CV] 7 Aug 2018]
2 LSTM Fully Convolutional Networks for Time Series Classification

Potential for Commercialization

Search & Rescue

Toys

Camera Drones