HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

MEDIEN

ActEV - HSMW_TUC Team

TRECVID 2019

Tony Rolletschke - University of Applied Science Mittweida

12/11/19

Introduction - Team

Our ActEv approach with object detection and custom tracking algorithm

Who is hiding behind ‘our’?

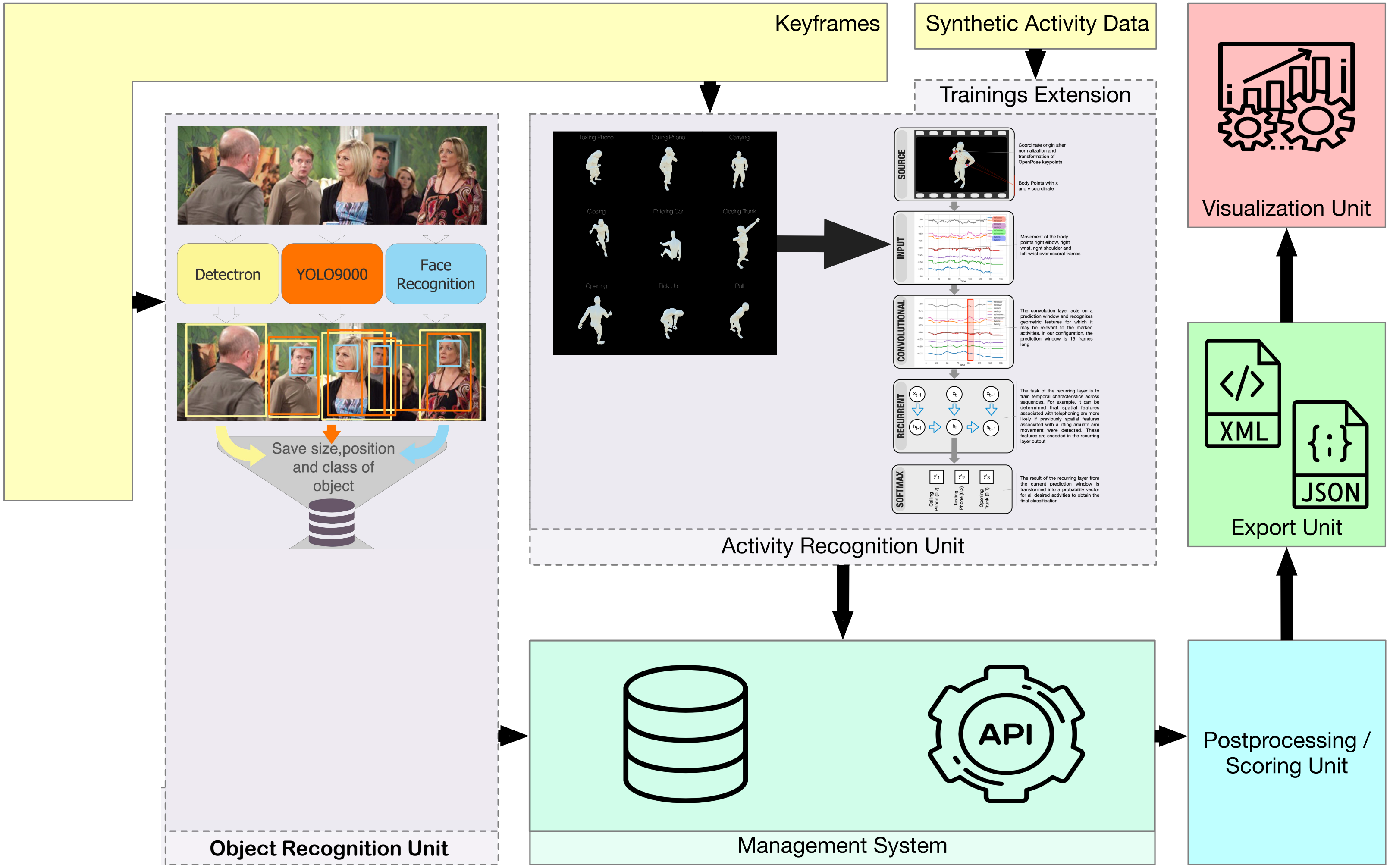
- Rico Thomanek
- Christian Roschke
- Benny Platte
- Tony Rolletschke
- Tobias Schlosser
- Manuel Heinzig
- Danny Kowerko
- Matthias Vodel
- Frank Zimmer
- Maximilian Eibl
- Marc Ritter

TECHNISCHE UNIVERSITÄT CHEMNITZ

HOCHSCHULE MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

System - Architecture

Holistic server-client approach



Keyframes

Synthetic Activity Data

Trainings Extension

Detectron

YOLO9000

Face Recognition

Save size,position and class of object

Object Recognition Unit

SOURCE

Coordinate origin after normalization and transformation of OpenPose keypoints

Body Points with x and y coordinate

INPUT

Movement of the body points right elbow, right wrist, right shoulder and left wrist over several frames

CONVOLUTIONAL

The convolution layer acts on a prediction window and recognizes geometric features for which it may be relevant to the marked activities. In our configuration, the prediction window is 15 frames long

RECURRENT

The task of the recurring layer is to train temporal characteristics across sequences. For example, it can be determined that spatial features associated with telephoning are more likely if previously spatial features associated with a lifting arcuate arm movement were detected. These features are encoded in the recurring layer output

SOFTMAX

The result of the recurring layer from the current prediction window is transformed into a probability vector for all desired activities to obtain the final classification

Activity Recognition Unit

Visualization Unit

Export Unit

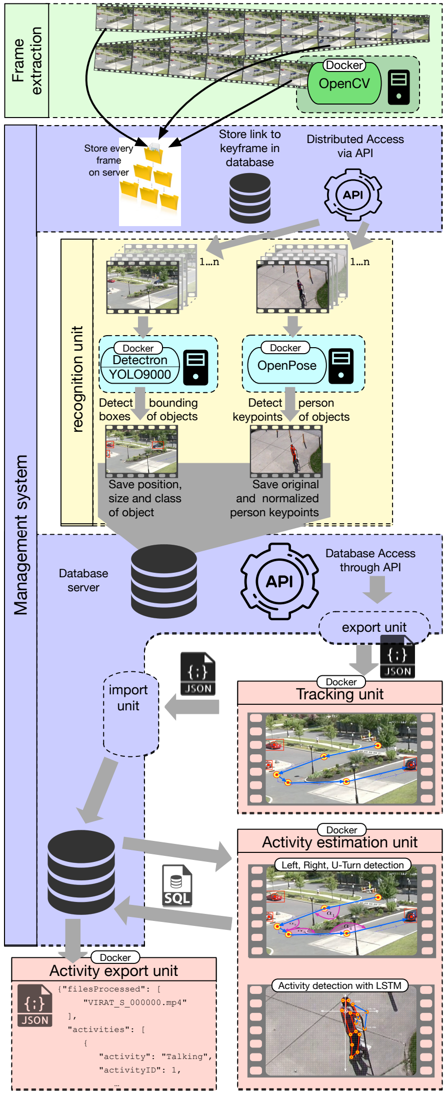
Management System

Postprocessing / Scoring Unit

Workflow

First step

- From the provided video material each frame was extracted
- Those frames were generated using *OpenCV*
- All frames are stored in the central file system
- Each image is provided with the original video title and a frame ID
- The meta information and references are stored in the database

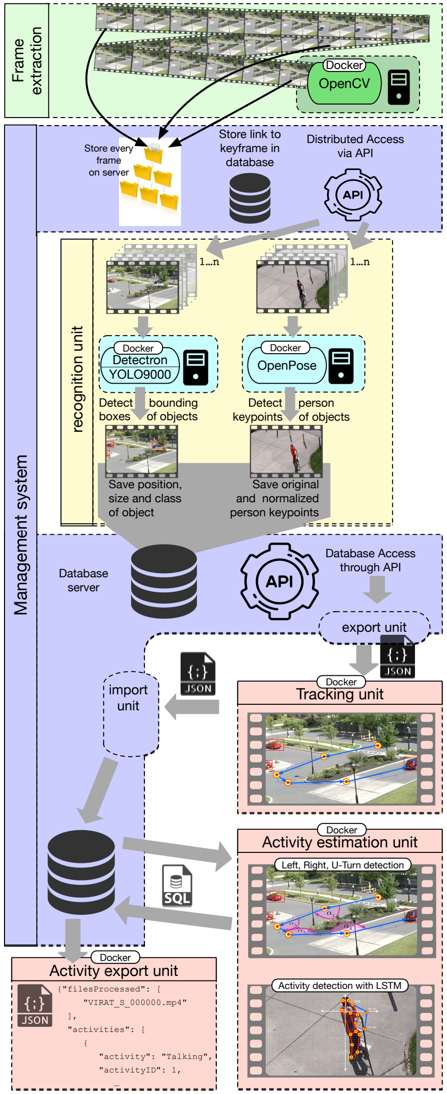

Frame extraction
Docker
OpenCV
Management system
Store every frame on server
Store link to keyframe in database
Distributed Access via API
API
recognition unit
1…n
1…n
Docker
Detectron
YOLO9000
Docker
OpenPose
Detect bounding boxes of objects
Detect person keypoints of objects
Save position, size and class of object
Save original and normalized person keypoints
Database server
API
Database Access through API
export unit
JSON
import unit
JSON
Docker
Tracking unit
Docker
Activity estimation unit
Left, Right, U-Turn detection
Activity detection with LSTM
SQL
Docker
Activity export unit
JSON

```
{"filesProcessed": [
     "VIRAT_S_000000.mp4"
  ],
  "activities": [
     {
        "activity": "Talking",
        "activityID": 1,
          …
```

Workflow

Second step

- Several clients in network compute state-of-art-frameworks
- With the usage of *Detectron* and *Yolo9000* objects and persons were detected
- The extraction of body-key-points is executed with *OpenPose*
- All outputs for each frame are stored in the database

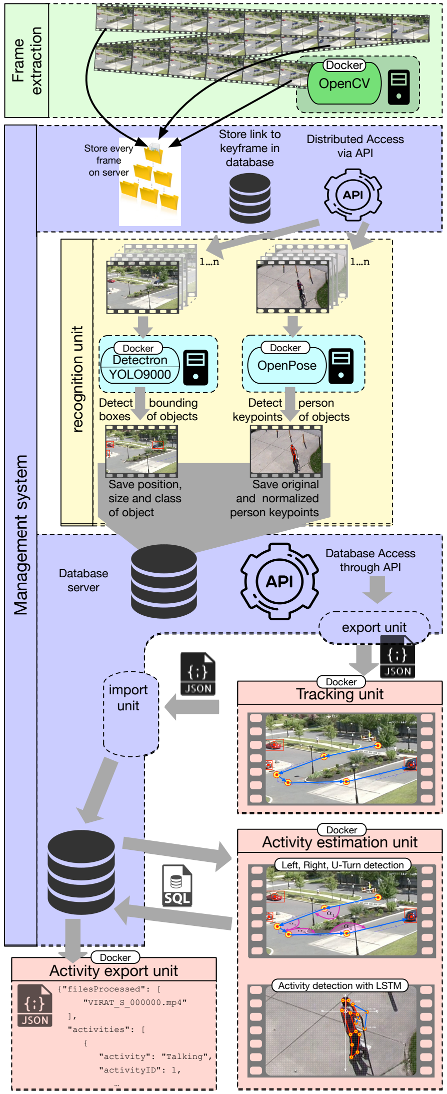

Frame extraction
Docker
OpenCV
Store every frame on server
Store link to keyframe in database
Distributed Access via API
API
Management system
recognition unit
1...n
1...n
Docker
Detectron
YOLO9000
Docker
OpenPose
Detect bounding boxes of objects
Detect person keypoints of objects
Save position, size and class of object
Save original and normalized person keypoints
Database server
API
Database Access through API
export unit
JSON
JSON
import unit
Docker
Tracking unit
Docker
Activity estimation unit
Left, Right, U-Turn detection
Activity detection with LSTM
SQL
Docker
Activity export unit
JSON

```
{"filesProcessed": [
    "VIRAT_S_000000.mp4"
  ],
  "activities": [
    {
      "activity": "Talking",
      "activityID": 1,
      …
```

Workflow

Third step

- The tracking results for all detected objects were estimated
- The activity recognition unit estimate the activities
- The results can be exported in a suitable exchange format



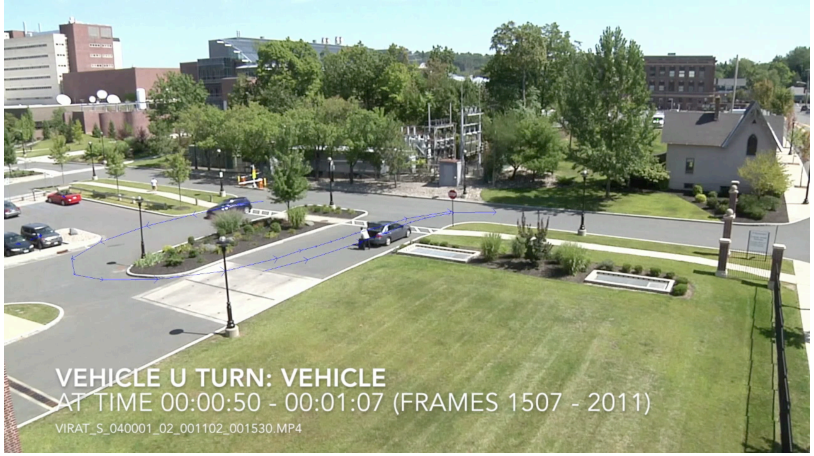
Frame extraction
Docker
OpenCV
Management system
Store every frame on server
Store link to keyframe in database
Distributed Access via API
API
recognition unit
1…n
1…n
Docker
Detectron
YOLO9000
Docker
OpenPose
Detect bounding boxes of objects
Detect person keypoints of objects
Save position, size and class of object
Save original and normalized person keypoints
Database server
API
Database Access through API
export unit
JSON
JSON
import unit
Docker
Tracking unit
Docker
Activity estimation unit
Left, Right, U-Turn detection
Activity detection with LSTM
SQL
Docker
Activity export unit
JSON

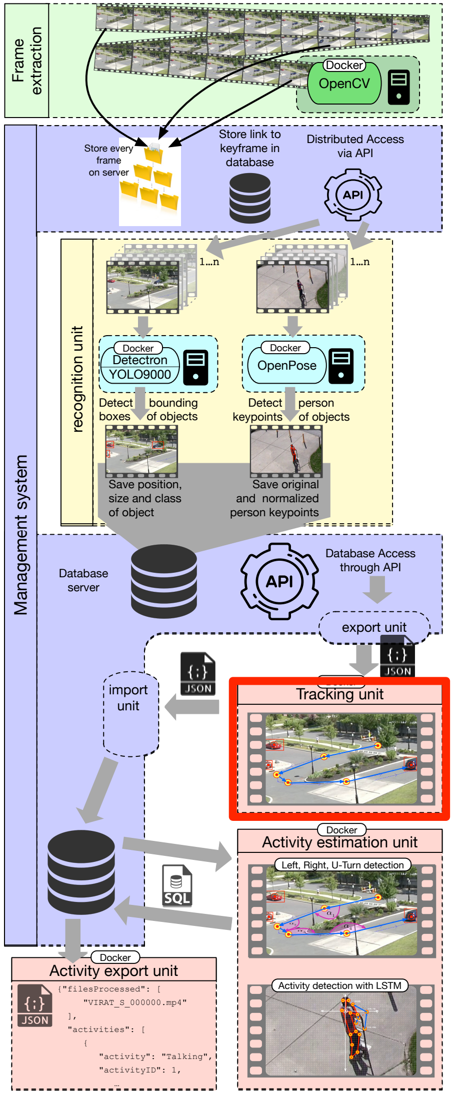
```
{"filesProcessed": [
     "VIRAT_S_000000.mp4"
  ],
  "activities": [
     {
        "activity": "Talking",
        "activityID": 1,
          …
```

Workflow - Tracking Unit

Tracking

- We use the tracking algorithm introduced last year
- As a result, unique id, direction, speed, and motion vectors estimated for a given time window

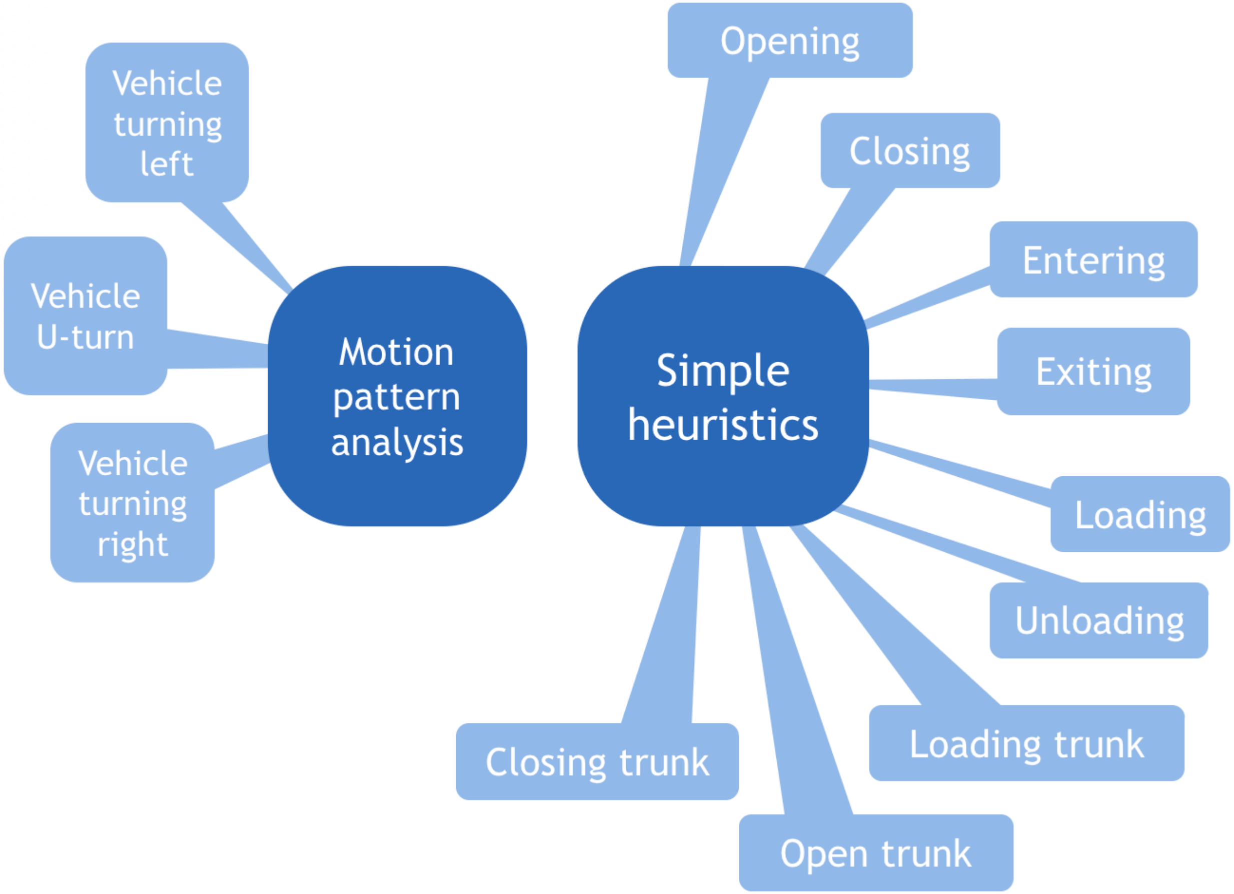

VEHICLE U TURN: VEHICLE
AT TIME 00:00:50 - 00:01:07 (FRAMES 1507 - 2011)
VIRAT_S_040001_02_001102_001530.MP4


Frame extraction
Docker
OpenCV
Management system
Store every frame on server
Store link to keyframe in database
Distributed Access via API
API
recognition unit
1...n
1...n
Docker
Detectron
YOLO9000
Docker
OpenPose
Detect bounding boxes of objects
Detect person keypoints of objects
Save position, size and class of object
Save original and normalized person keypoints
Database server
API
Database Access through API
export unit
JSON
import unit
JSON
Tracking unit
Docker
Activity estimation unit
Left, Right, U-Turn detection
SQL
Activity detection with LSTM
Docker
Activity export unit
JSON

```
{"filesProcessed": [
    "VIRAT_S_000000.mp4"
  ],
  "activities": [
    {
      "activity": "Talking",
      "activityID": 1,
        ...
```

Workflow - Activity Estimation Unit

Motion pattern analysis & simple heuristics



Motion pattern analysis

Vehicle turning left

Vehicle U-turn

Vehicle turning right

Simple heuristics

Opening

Closing

Entering

Exiting

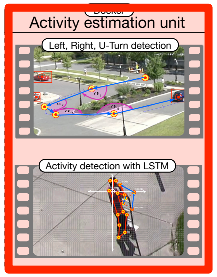
Loading

Unloading

Loading trunk

Open trunk

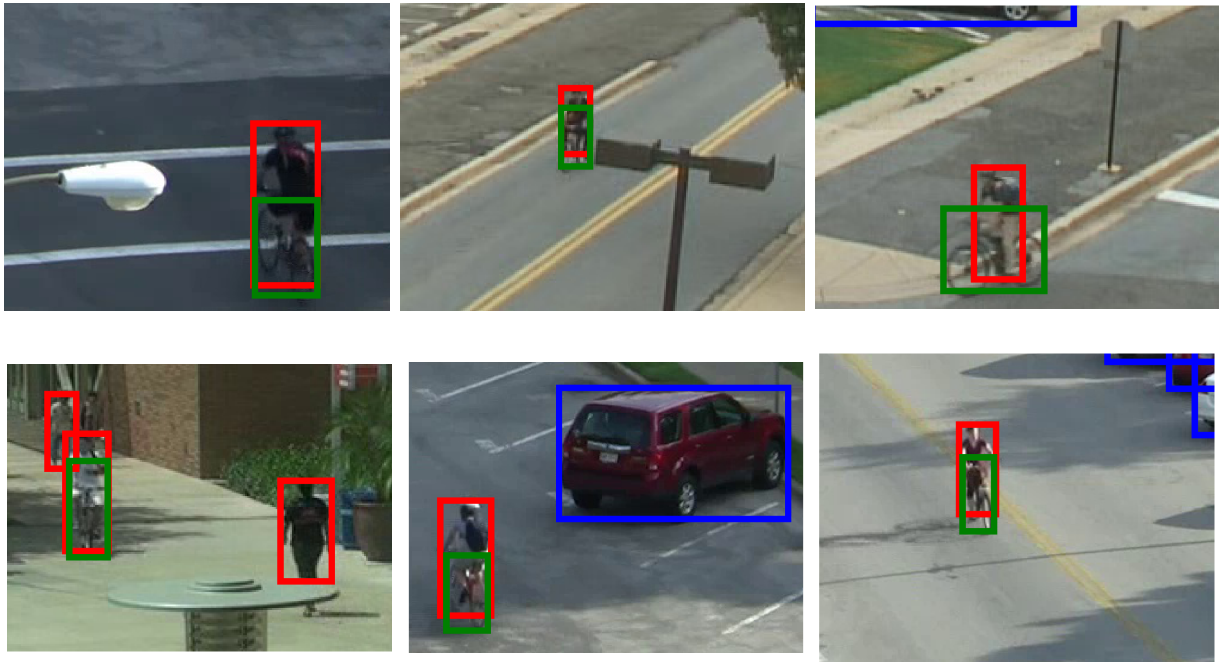
Closing trunk

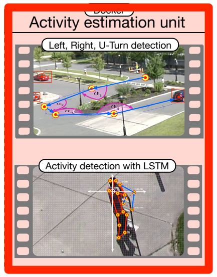


Workflow - Activity Estimation Unit

Motion pattern analysis & simple heuristics

- Bounding box interaction for a specific period

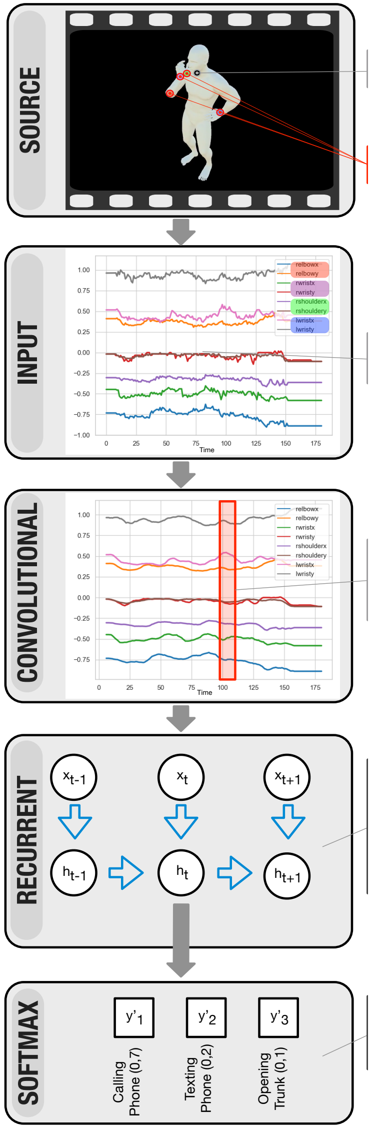




Workflow - Activity Estimation Unit

Activity classifier

- Generate an synthetic ground-truth-dataset with the unity game engine
- Body-key-points extract as feature-vectors with *OpenPose*
- Convolutional layers extract geometric temporal features from a single prediction window
- Recurrent layers extract temporal features over time
- A probability vector for all desired activities to obtain the final classification



Coordinate origin after normalization and transformation of OpenPose keypoints

Body Points with x and y coordinate

Movement of the body points right elbow, right wrist, right shoulder and left wrist over several frames

The convolution layer acts on a prediction window and recognizes geometric features for which it may be relevant to the marked activities. In our configuration, the prediction window is 15 frames long

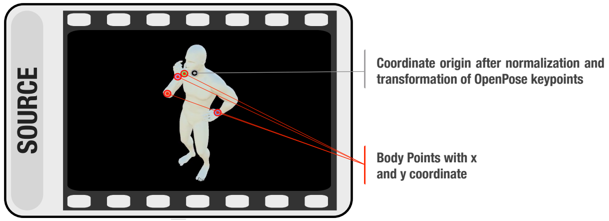
The task of the recurring layer is to train temporal characteristics across sequences. For example, it can be determined that spatial features associated with telephoning are more likely if previously spatial features associated with a lifting arcuate arm movement were detected. These features are encoded in the recurring layer output

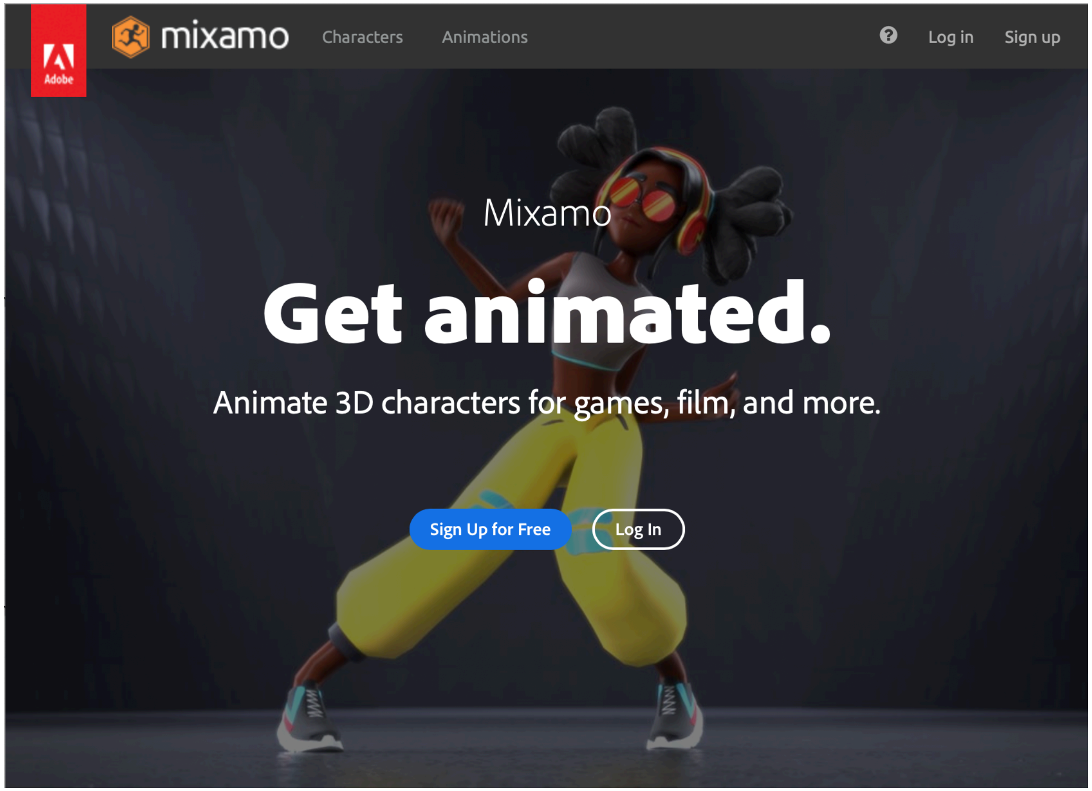
The result of the recurring layer from the current prediction window is transformed into a probability vector for all desired activities to obtain the final classification

Workflow - Ground Truth Data

Generate an synthetic ground-truth-dataset

- Download the animations from „*Mixamo*“


SOURCE
Coordinate origin after normalization and transformation of OpenPose keypoints
Body Points with x and y coordinate



Mixamo

Get animated.

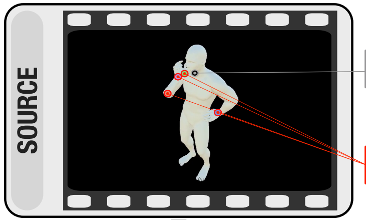
Animate 3D characters for games, film, and more.

Source: https://www.mixamo.com

Workflow - Ground Truth Data

Generate an synthetic ground-truth-dataset

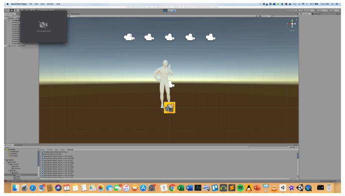
- Download the animations from „*Mixamo*“



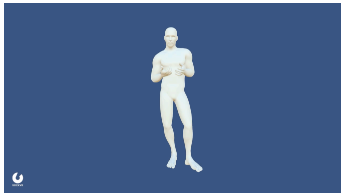
Coordinate origin after normalization and transformation of OpenPose keypoints

Body Points with x and y coordinate

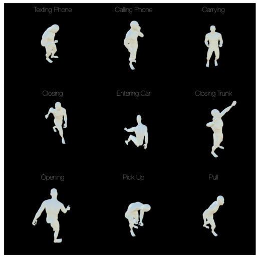
- Simultaneously recording activities from 10 different perspectives



- Multiple variances of activity animations are possible

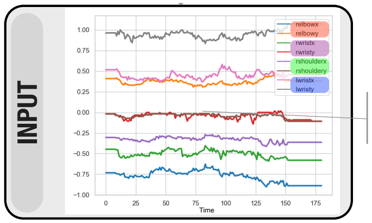


- 5535 synthetic animations were generated and decomposed into 536517 frames



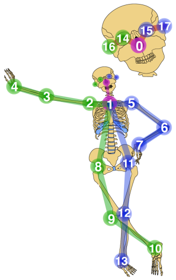
Workflow - Feature Extraction

Body-key-points extract as feature-vectors with *OpenPose*

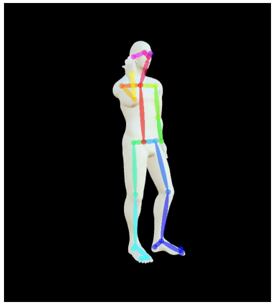


Movement of the body points right elbow, right wrist, right shoulder and left wrist over several frames

- The COCO model of *OpenPose* provides 18 body-key-points



Nr	Name
0	Nose
1	Neck
2	RShoulder
3	RElbow
4	RWrist
5	LShoulder
6	LElbow
7	LWrist
8	RHip
9	RKnee
10	RAnkle
11	LHip
12	LKnee
13	LAnkle
14	REye
15	LEye
16	REar
17	LEar

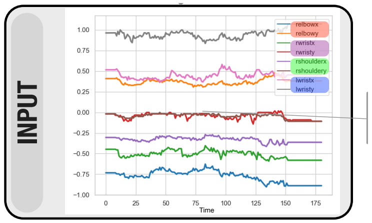


- This body-key-points were extracted from all animations and stored in the database

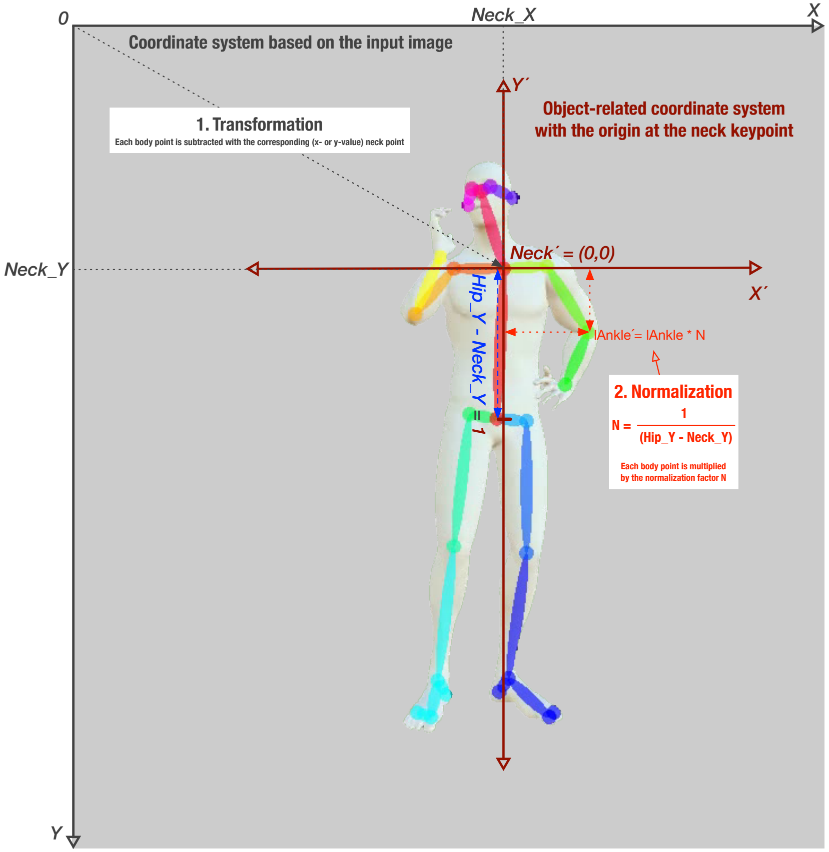
Workflow - Feature Extraction

Normalization of body-key-points

- Transform the image coordinates to a body-centered point
- Neck is the origin of coordinates
- Body-points also must be normalized
- Distance between neck and hip



Movement of the body points right elbow, right wrist, right shoulder and left wrist over several frames



Coordinate system based on the input image

1. Transformation

Each body point is subtracted with the corresponding (x- or y-value) neck point

Object-related coordinate system with the origin at the neck keypoint

2. Normalization

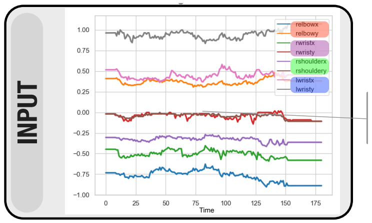
$$N = \frac{1}{(Hip\_Y - Neck\_Y)}$$

Each body point is multiplied by the normalization factor N

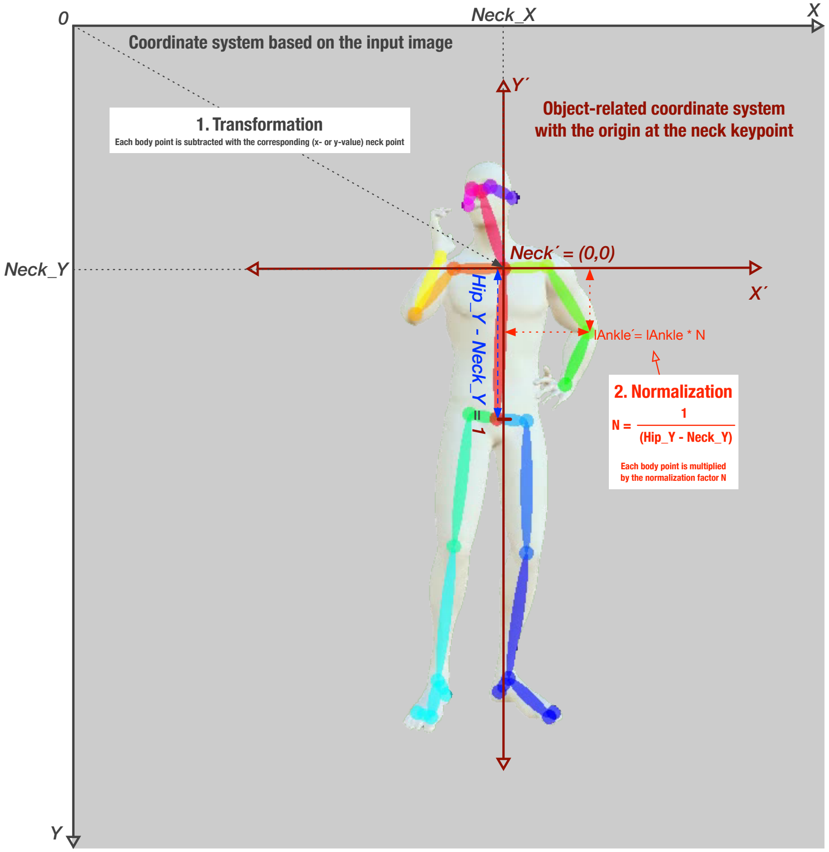
Workflow - Feature Extraction

Normalization of body-key-points

- The normalized and transformed body-key-points are independent from the image resolution and format



Movement of the body points right elbow, right wrist, right shoulder and left wrist over several frames



Coordinate system based on the input image

Object-related coordinate system with the origin at the neck keypoint

1. Transformation

Each body point is subtracted with the corresponding (x- or y-value) neck point

2. Normalization

$$N = \frac{1}{(Hip\_Y - Neck\_Y)}$$

Each body point is multiplied by the normalization factor N

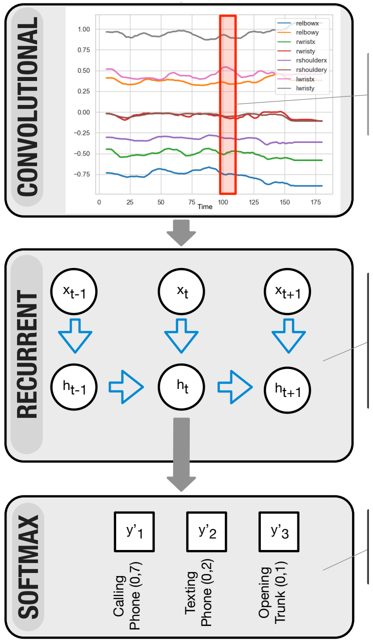
Activity: nothing (0.9208972454071045)

Activity: nothing (0.9854370951652527)

Workflow - Creating Classifier

Convolutional layers extract geometric temporal features from a single prediction window

- We get 36 individual „sensor values“
- Sensor values are labeled with an activity and sorted chronologically in ascending order
- Training of the activity classifier
- Prediction window was set to 15 frames
- The detected activities are stored in the database with a probability value



The convolution layer acts on a prediction window and recognizes geometric features for which it may be relevant to the marked activities. In our configuration, the prediction window is 15 frames long

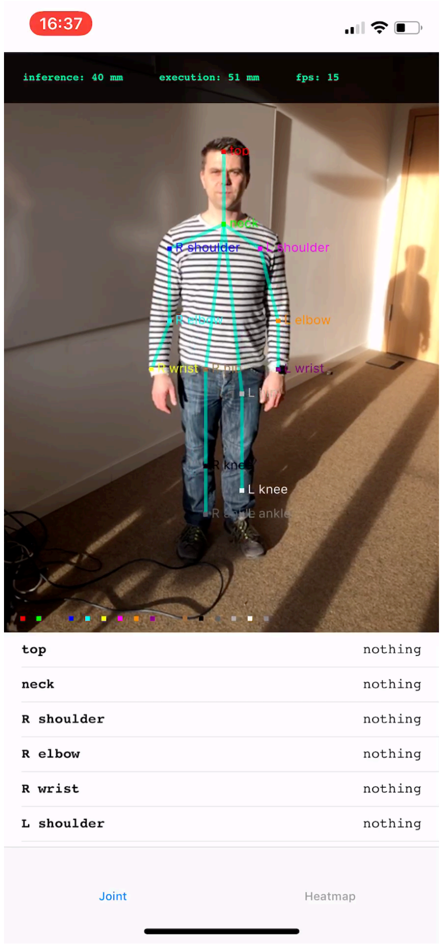
The task of the recurring layer is to train temporal characteristics across sequences. For example, it can be determined that spatial features associated with telephoning are more likely if previously spatial features associated with a lifting arcuate arm movement were detected. These features are encoded in the recurring layer output

The result of the recurring layer from the current prediction window is transformed into a probability vector for all desired activities to obtain the final classification

Mobile Application

Development of an mobile iOS application for activity recognition

- Live captured camera stream
- Extract body-key-points with version of *OpenPose*, which is optimized for mobile devices
- Normalized body-key-points
- Predict activities with the ActEV activity classier trained with *Turi Create*


16:37

```
inference: 40 mm      execution: 51 mm      fps: 15
```

top	nothing
neck	nothing
R shoulder	nothing
R elbow	nothing
R wrist	nothing
L shoulder	nothing

Conclusion

Our ActEV approach with object detection, custom tracking algorithm and custom actvity classifier

- We significantly improved performance
- We find a easy way to generate ground for video based activity recognition
- We proof that the model trained with synthetic data is able to classify real data
- We integrate the new activity recognition unit in our system architecture

Future Work

Our ActEV approach with object detection, custom tracking algorithm and custom actvity classifier

- We will use different kinds of person models for training
- We still working on a approach to export the body-key-points directly out of the game engine
- In addition we working on a approach for multiple person realtime activity recognition
- General optimization and evaluation of our new approach