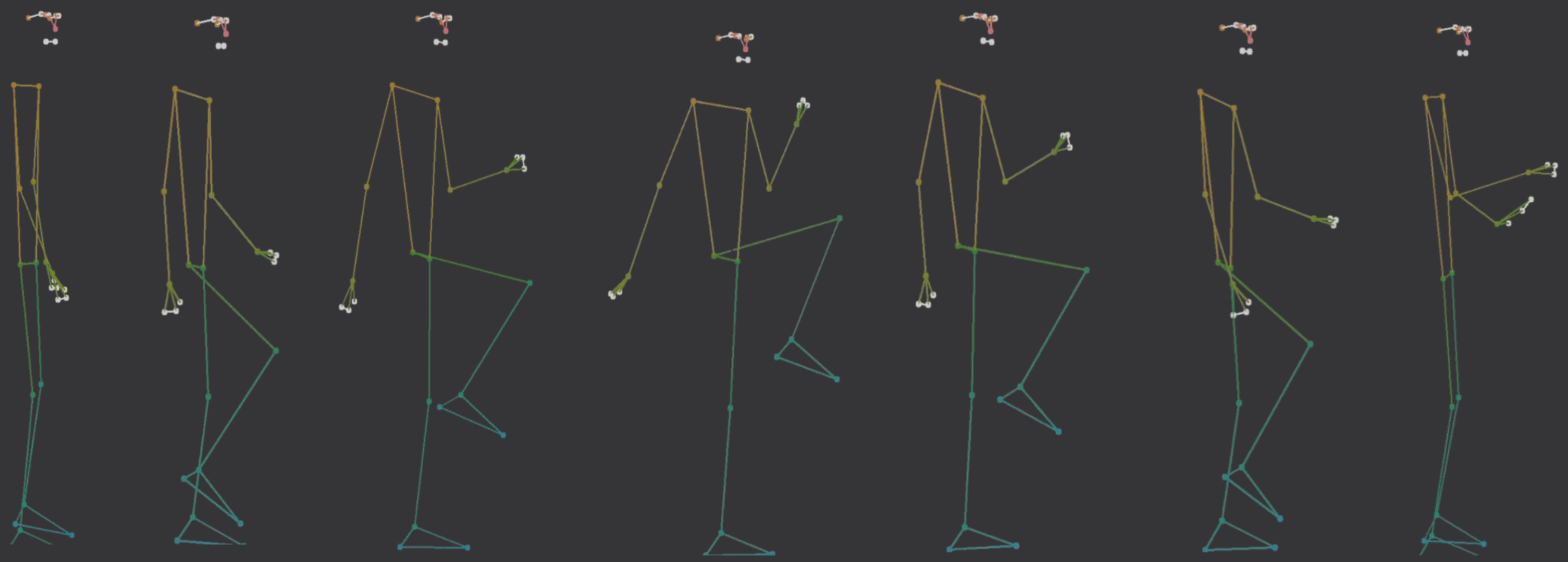


Statistics In Motion

A Python-based Analysis of the 2-Minute Step Test – A M.Sc. SEDS Data Science Project

Jördis Luisa Strack

joerdis.strack@uni-konstanz.de

Contents

Introduction & Motivation

Task & Relevance

Rise in popularity of at-home workouts during COVID-19

- supported by digital tools e.g., YouTube and fitness apps.

Older adults face challenges with access to real-time feedback and fitness specialists

- home fitness support crucial for maintaining physical independence.

2-Minute Step Test (2MST):

Functional fitness test to assess aerobic endurance and lower-body strength, especially for older adults

- test involves raising each knee to a point midway between the kneecap and hip within two minutes.

Project goal:

Provide personalized feedback on 2MST performance using a Python-based heuristic and a benchmark comparison

Introduction & Motivation

Task Description

Data Generation:

- analyze 2MST performance using video data transformed into coordinate form.

Preprocessing:

- apply Gaussian smoothing (sigma 2) to reduce positional fluctuations.

Data Analysis

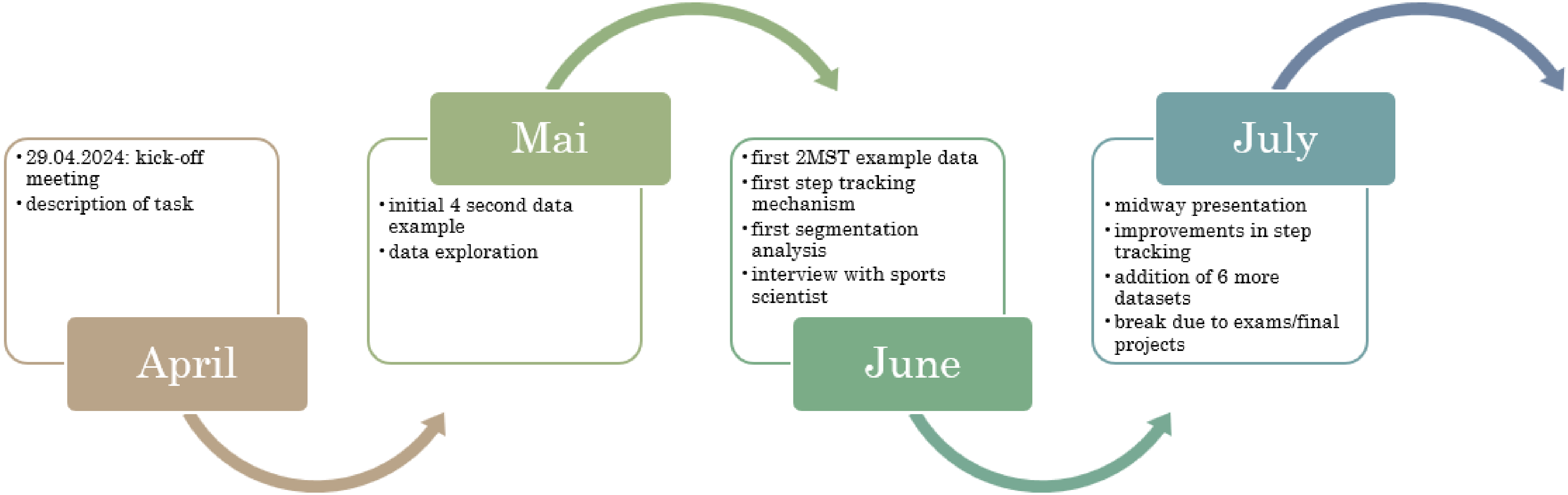
- calculate velocity from timestamp differences and define valid steps based on knee thresholds and ground intervals.
- check for cross-body coordination (right leg-left arm and vice versa movement).
- segment performance into four 30-second intervals to track consistency over time.

Output:

- provide individualized feedback to help improve form and endurance.

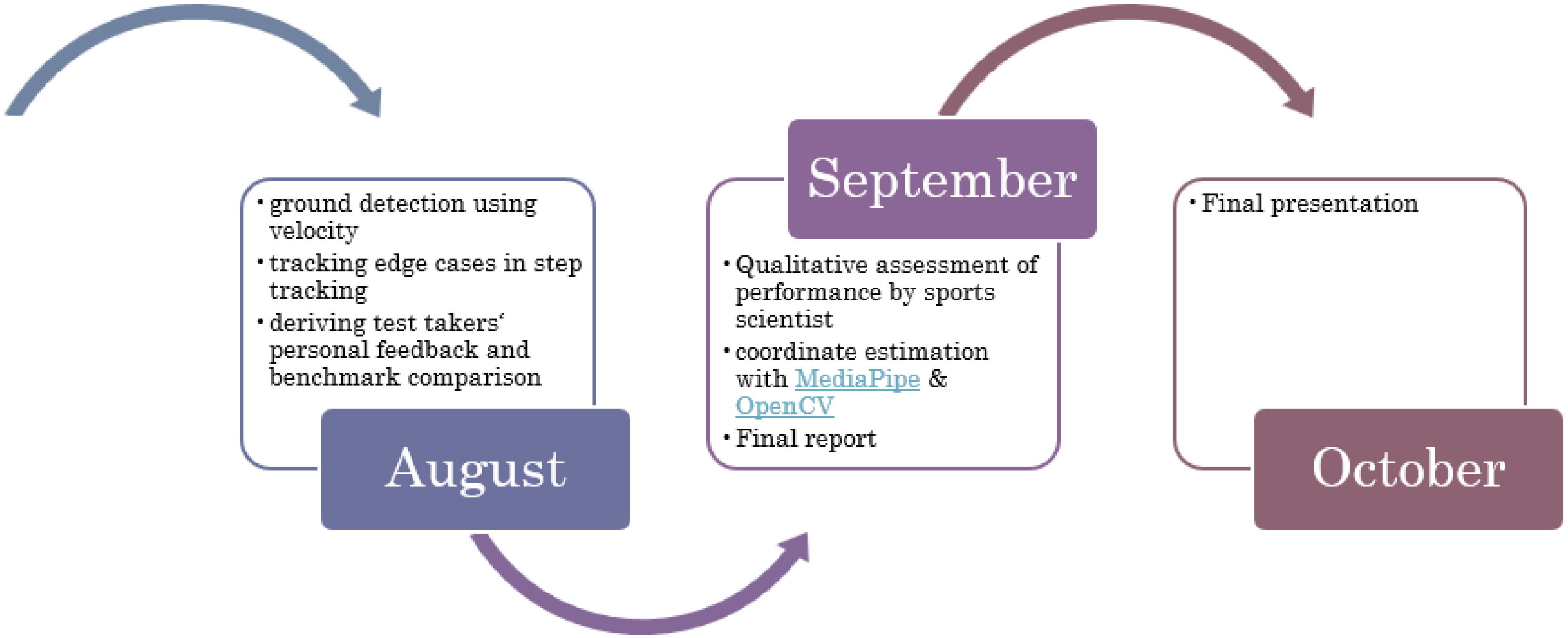
Introduction & Motivation

Timeline of Development



Introduction & Motivation

Timeline of Development



Algorithm | Coordinate Generation

Generate Coordinates from any mp4 file using Python's OpenCV and MediaPipe

1. processes video frame-by-frame to extract 32 body landmarks offered by MediaPipe
2. ensures valid FPS, compute timestamp and convert frames from BGR to RGB
3. Extracts x, y, z coordinates for each landmark per frame
4. standardize coordinates to [0, 1] for comparability across test takers

OpenCV Team. (2024). *Open Source Computer Vision Library*. https://opencv.org
Lugaresi, C., et al. (2019). *MediaPipe: A Framework for Building Perception Pipelines*. Google AI Blog. https://google.github.io/mediapipe

Algorithm | Preprocessing

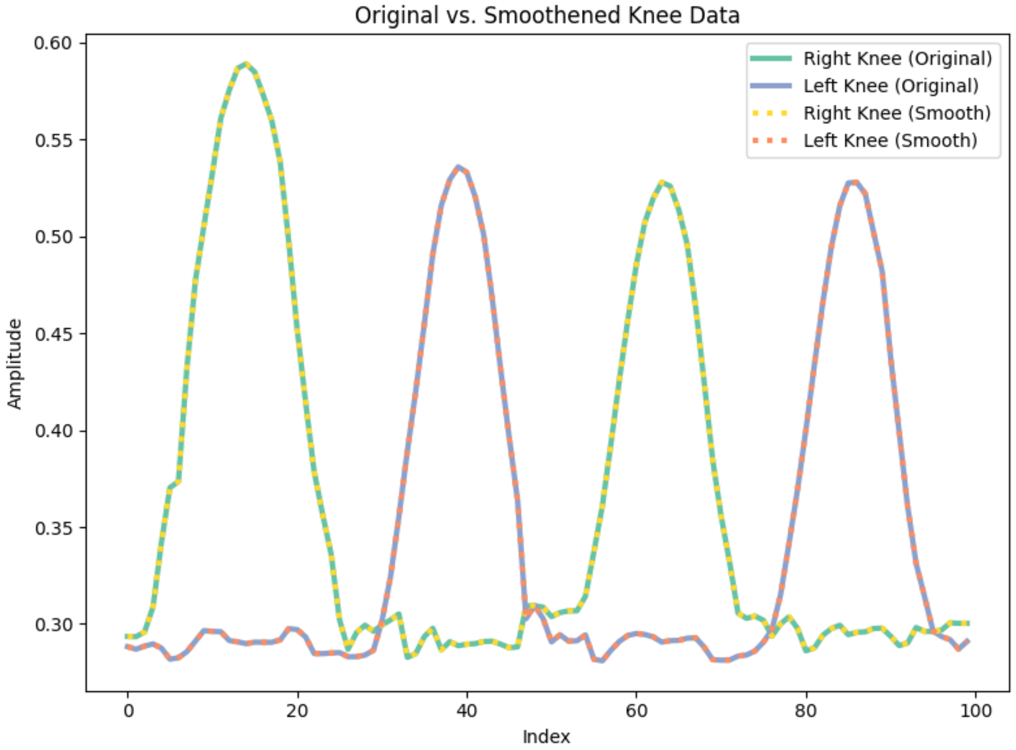
Handling Noise

Original data was distorted by noise

- caused by oversensitive coordinate generation
- under MediaPipe approach no longer present

Apply SciPy's Gaussian filter to smooth out any remaining noise

- sd. of Gaussian Kernel Sigma = 2 to smooth out coordinates moderately
- avoid over-smooth data and loss of information!



SciPy v1.10.1. (2024). *scipy.ndimage.gaussian_filter*. SciPy Documentation

Algorithm | Tracking Steps

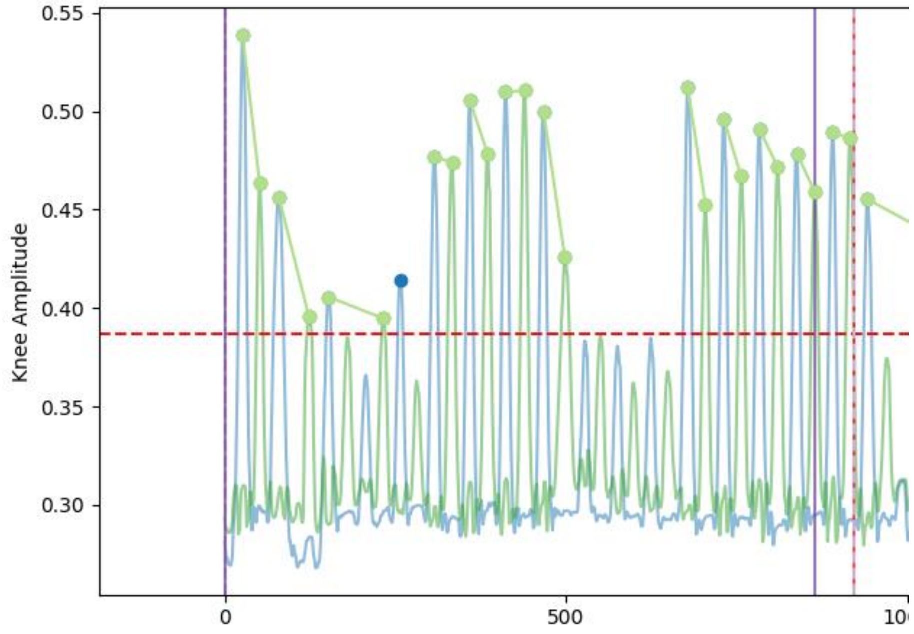
What is a Step?

A step is a sequence of the following events:

- right knee is lifted above knee threshold while left foot is on the ground
- left knee is lifted above knee threshold while right foot is on the ground

How to track Steps?

1. define height threshold for knees
2. use velocity and marker information to detect ground contact for feet
3. use both requirements to identify valid step sequences



Algorithm | Tracking Steps

Finding the Knee-threshold

Knee threshold reflects height threshold between right hip and knee used in 2MST

- identify global minimum of right foot marker
- obtain index at which right leg is most extended

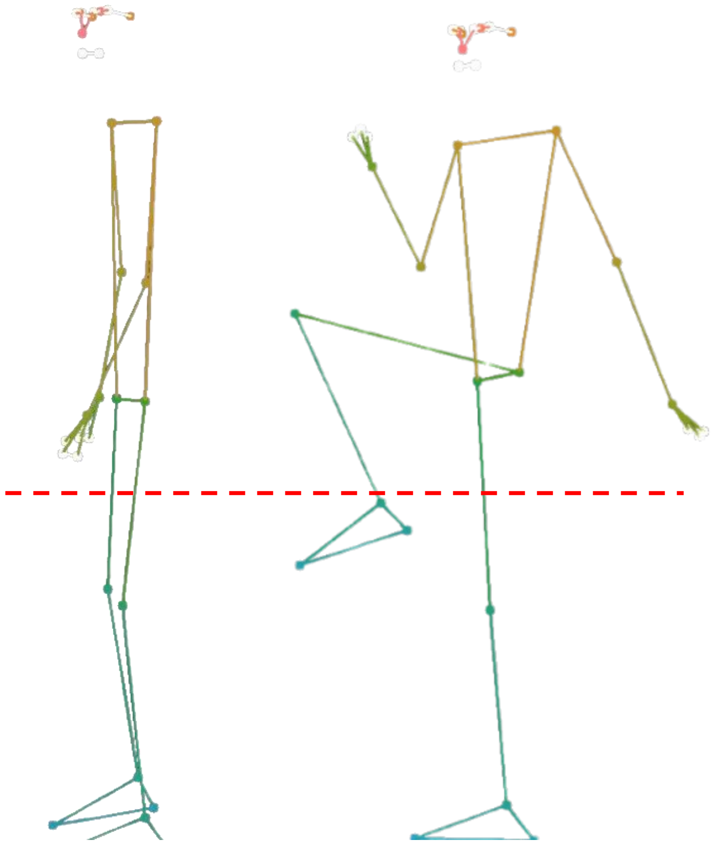
Function ğîŋđ lŋêê ţĥsêşĥộľđ computes Euclidian distance between right hip and knee

- identify point exactly in between both markers

Threshold Significance

The knee threshold differentiates valid steps from non-valid attempts as the most important requirement

- function ŵắľîđắţê şţêř şêṛụêŋçê applies knee threshold



An illustrative sketch of the knee threshold based on the test taker‘s right leg markers

Algorithm | Tracking Steps

Autonomous detection of Ground Contact

The function çsêǻʧê ĝsộụŋđ çộŋʧǻçʧ îŋʧêsŵǻľ

- inverts data to identify local minima in the Y-coordinates of both feet
- calculates ground contact intervals based on positions of foot and ankle markers
  → ankle marker as threshold to avoid overestimation
- assesses initial stance during first three seconds before first knee peak
  → account for potential offsets in foot positions due to bending or relaxed stances

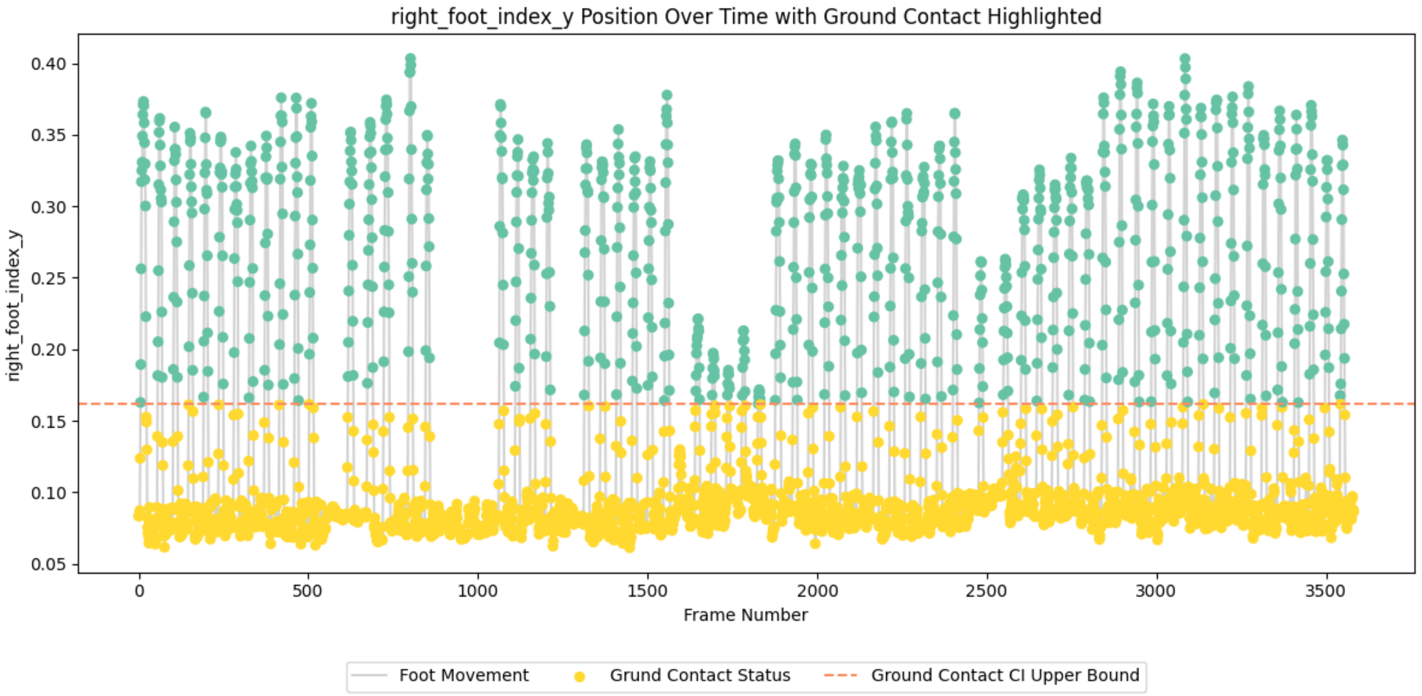
The function đêʧêçʧ ĝsộụŋđ çộŋʧǻçʧ

- evaluates foot coordinates for lower bound fixed at zero and upper bound dynamically determined
- combines foot marker positions with velocity to detect moments of ground contact
- utilizes finite differences to compute instantaneous velocity
- applies threshold based on 95th percentile to define inactivity within the ground contact interval

Algorithm | Tracking Steps

Autonomous detection of Ground Contact

A slight overestimation in ground contact is present, due to oversensitivity of y-coordinates in both feet of the initial data used during the first stages of development



Algorithm | Tracking Steps

Valid Step Sequence Identification

Step Validation:

The ŵắľîđắʧê ṣʧêř ṣêṛụêŋçê function

- flags the last right knee peak to ensure the final step is included in the analysis
- iterates through right knee peaks to find corresponding valid left knee peaks
  → sequences with consecutive raised peaks are deemed invalid.
- assesses ground contact for both feet between the identified knee peaks, ensuring valid steps meet both height and ground contact conditions.

Output:

- returns a list of valid step sequences along with the total count of valid steps detected

Algorithm | Tracking Steps

Posture and Coordination

The `sêŵîşêđ şţêř şêŗųêŋçê ắŋắľỳşîş` function

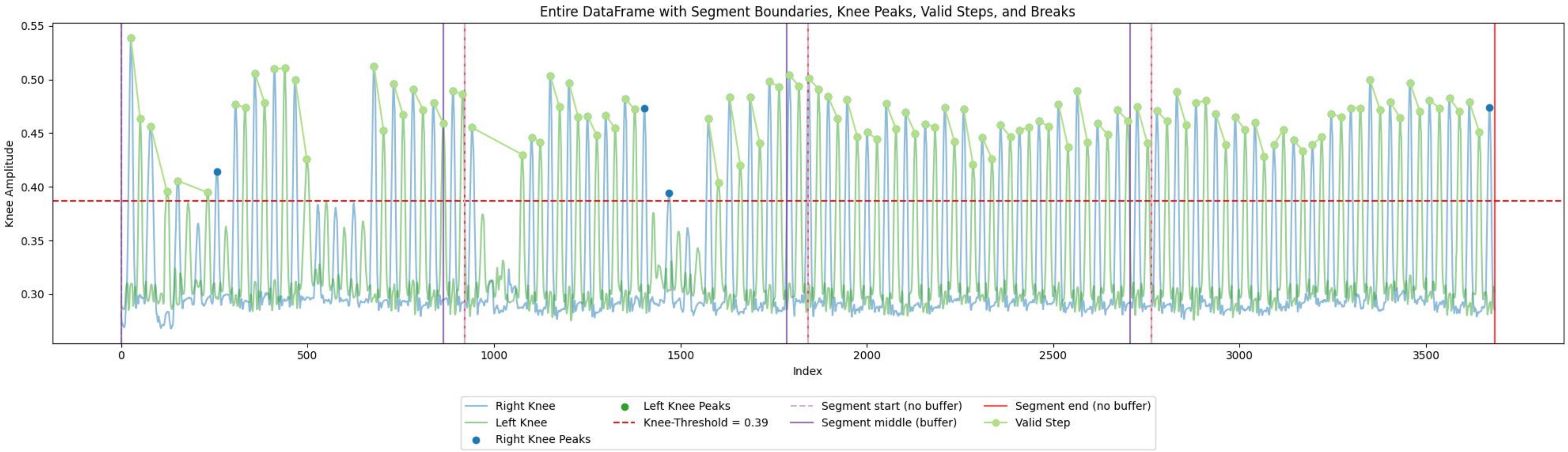
- evaluates cross-body arm coordination during valid steps
- checks for the lifting of the opposite arm corresponding to knee lifts
  → time window of 200 milliseconds to account for minor delays
- reports number of coordinated sequences, including total step count and count of steps showing correct arm-leg coordination
  → allows for analysis of posture during movement

Good posture is not a requirement for 2MST but of interest to evaluate balance and posture over time

Algorithm | Step Analysis

Test Takers' Personal Performance Metrics:

- main focus is number of valid steps taken during 2MST
- segmentation analysis of performance data can reveal additional feedback



Algorithm | Step Analysis

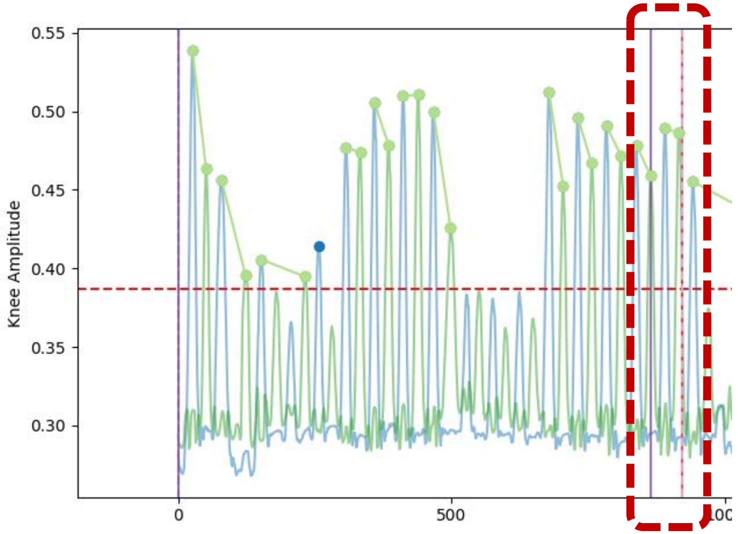
Buffer Calculation for Steps that are carried over Segments

- in some cases, steps are started in one segment and completed in the following segment
  → this can cause steps to be counted twice!

- a buffer between segments
  - captures steps that start in one segment and end in the next
  - searches for unpaired right knee peaks at the end of one segment if the subsequent segment begins with an unpaired left peak

- buffer is computed dynamically based on all knee peak attempts (valid and invalid)
  → buffer is average time between peaks to create small buffer window in which a right knee peak can be expected

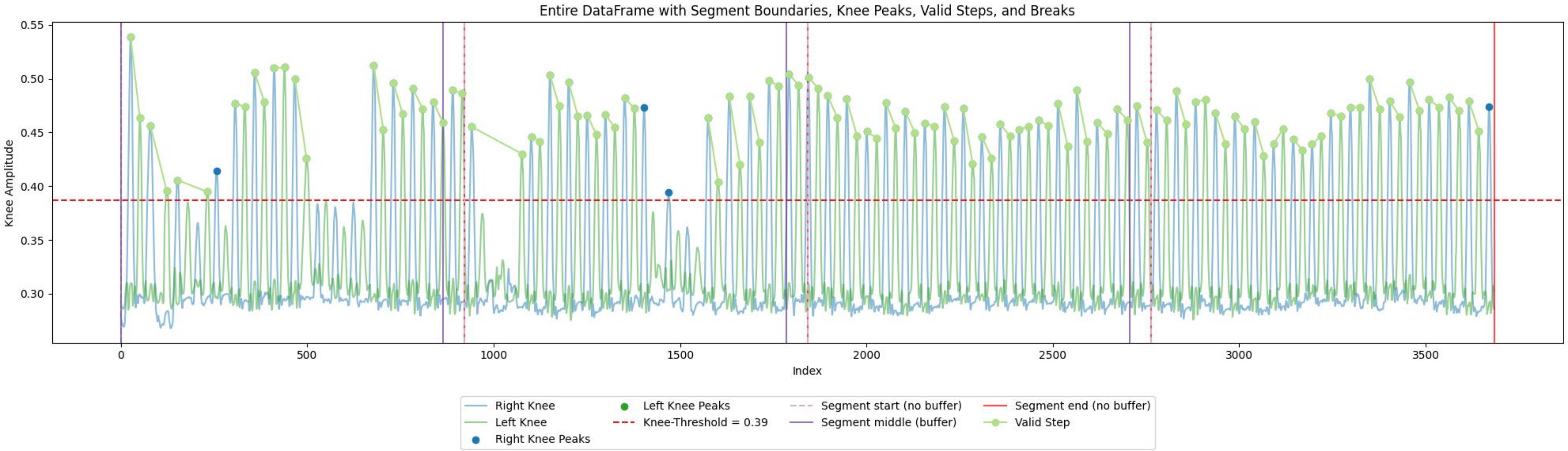
This is the buffer between segments 1 and 2:



Algorithm | Step Analysis

Segmentation and Consistency Analysis

- ǧîŋđ şêĝņêŋţ čųğğês splits data into four segments of equal length, using dataset specific buffer
- ǎŋǎľỳćê şêĝņêŋţş inspects each segment and aggregates results:
  - number of valid steps and steps under good posture
  - variation in knee peak amplitudes overall and for left and right knee
  - time between knee peaks and overall number of breaks that lasted at least three seconds



Personal Feedback

Benchmark Comparison

- normative data from Rikli and Jones (1999a), N = 7,183 (2,135 men and 5,048 women) aged 60-94 years.
- original benchmarks for ages 60-94 in white
- interpolated age ranges for participants above 94 based on observed trends in grey
- notably high standard deviations in the data due to smaller demographic subgroups.

Table 1: Interpolated average count and standard deviation of steps by age and gender based on Rikli and Jones (1999a); N = 7,183; age range = 60-94 years, including normative data for men (n = 2,135) and women (n = 5,048)

Age Range	Start Age	Steps (Male)	SD (Male)	Steps (Female)	SD (Female)
15 - 19	15	119	24	110	22
20 - 24	20	117	24	108	22
25 - 29	25	115	23	105	23
30 - 34	30	112	23	103	23
35 - 39	35	110	22	100	23
40 - 44	40	108	22	98	23
45 - 49	45	106	22	96	24
50 - 54	50	103	21	93	24
55 - 59	55	101	21	91	24
60 - 64	60	101	21	91	24
65 - 69	65	101	23	90	26
70 - 74	70	95	23	84	25
75 - 79	75	91	27	84	24
80 - 84	80	87	24	75	23
85 - 89	85	75	24	70	22
90 - 94	90	69	26	58	21
95 - 99	95	64	27	53	21
100 - 104	100	60	27	49	20
105 - 109	105	55	28	44	20
110 - 114	110	51	29	39	19
115 - 119	115	46	30	34	19
120 - 124	120	42	30	30	18

Personal Feedback

Interpolation

- linear interpolation used to extend step count benchmarks for participants aged 95 to 120
  → based on natural physical performance rate of decline
- additional benchmarks derived for younger test-takers, aged 18-59 using rate of increase
  → check of plausibility using data from Nogueira et al. (2021)

Physical Independence

- for test takers who are at least 60 years old, number of valid steps used to determine status of physical independence
  → if number of valid steps is greater than threshold, test taker is classified as independent

Age Range	Start Age	Threshold (Male)	Threshold (Female)
60 - 64	60	106	97
65 - 69	65	101	93
70 - 74	70	95	89
75 - 79	75	88	84
80 - 84	80	80	78
85 - 89	85	71	70
90 - 94	90	60	60
95 - 99	95	53	55
100 - 104	100	47	49
105 - 109	105	40	44
110 - 114	110	34	39
115 - 119	115	27	34
120 - 124	120	21	28

Table 2: Interpolated cutoff threshold values by age and gender based on Rikli and Jones (2013); N = 2,140; age range = 60-94 year

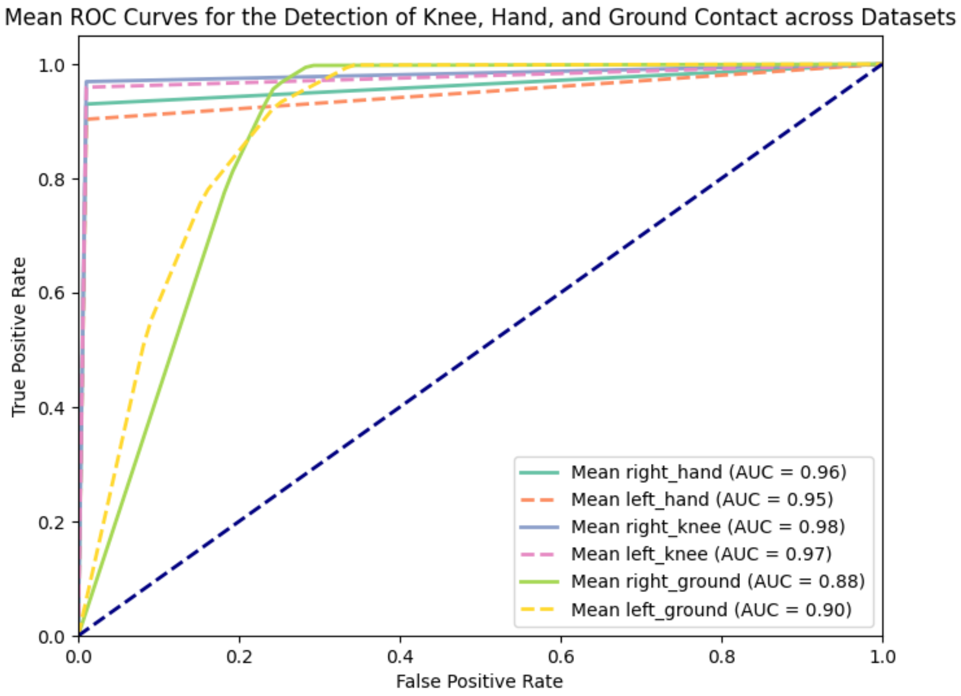
Results & Discussion

Performance Metrics

- **knee and hand detection** show high precision, recall, and accuracy across datasets
  → AUC values are above 0.95 indicate robustness

- **ground contact detection** shows lower precision but retains high recall
  → ground detection is overestimated due to changes in data after early development

- imprecise **human-annotated labels** likely effect performance metrics of hand and knee peak detection
  → algorithm relies on SciPy's find_peaks, which is more precise than any human annotation
  → even one typo in labels distorts true performance metrics



Conclusion

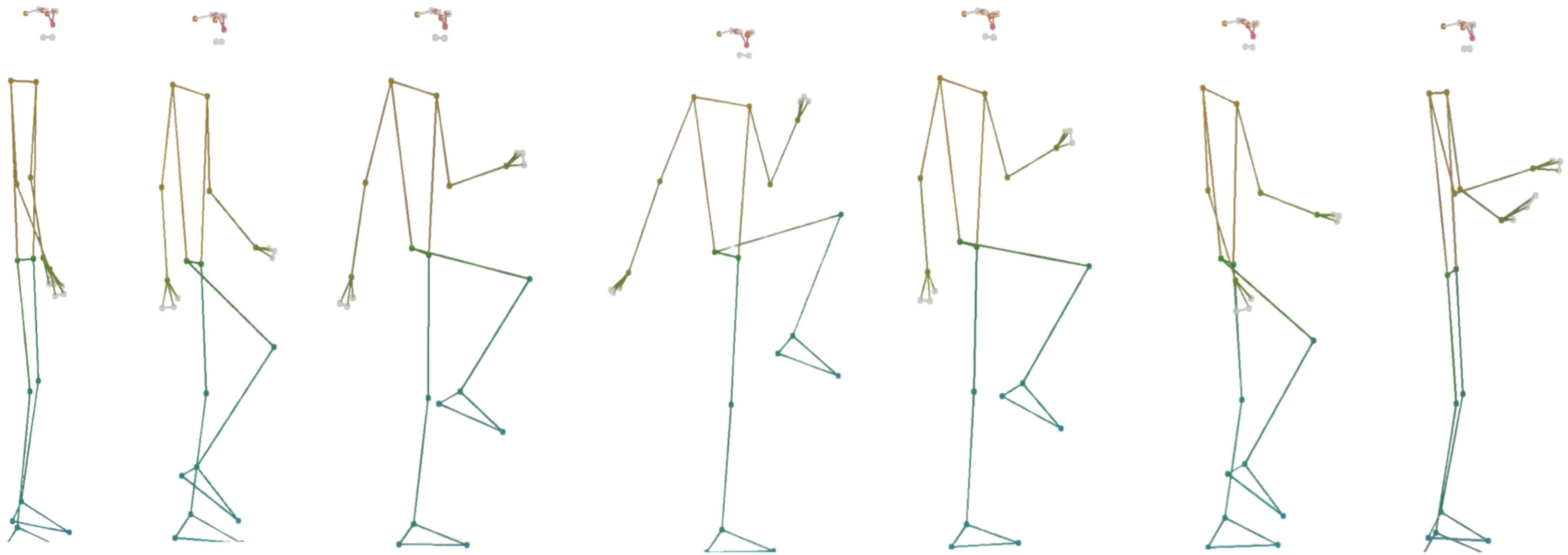
The algorithm

- is innovative as one of the first of its kind and enables automatic scoring of the 2MST
- is built entirely using open-source libraries making it accessible for interdisciplinary research and at-home use
- effectively detects peaks in hand and foot coordinates with perfect precision and high recall
  → due to minor inconsistencies in human-annotated both precision and recall are likely perfect
- performs effectively even with partial occlusion or low video quality
  → verified by ROC and AUC metrics

Based on an interview with a sports scientist, the project fully meets the current requirements of sports science for an analysis program for the 2MST.

Thank you for your Attention!

Do you have any Questions?



Literature

- Nogueira, M. A., Almeida, T. D. N., Andrade, G. S., Ribeiro, A. S., Rˆego, A. S., Dias, R. D. S., Ferreira, P. R., Penha, L. R. L. N., Pires, F. D. O., Dibai-Filho, A. V., & Bassi-Dibai, D. (2021). Reliability and Accuracy of 2-Minute Step Test in Active and Sedentary Lean Adults. Journal of Manipulative and Physiological Therapeutics, 44 (2), 120–127. https ://doi.org/10.1016/j.jmpt.2020.07.013

- Rikli, R. E., & Jones, C. J. (1999a). Development and Validation of a Functional Fitness Test for Community-Residing Older Adults. Journal of Aging and Physical Activity, 7 (2), 129–161. https://doi.org/10.1123/japa.7.2.129

- Rikli, R. E., & Jones, C. J. (1999b). Functional Fitness Normative Scores for Community- Residing Older Adults, Ages 60-94. Journal of Aging and Physical Activity, 7 (2), 162–181. https://doi.org/10.1123/japa.7.2.162

- Rikli, R. E., & Jones, C. J. (2013). Development and validation of criterion-referenced clinically relevant fitness standards for maintaining physical independence in later years [Publisher:Oxford University Press]. The gerontologist, 53 (2), 255–267.

- Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., . . . SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17, 261–272. https://doi.org/10.1038/s41592-019-0686-2