

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

School of Computer Science and Statistics

Assessing Bicycle Saddle Height from Smartphone Video of Passing Cyclists

Kate O'Neill

May 2026

A Report submitted in partial fulfilment
of the requirements for the degree of
BA(Mod) in Science in Computer Science

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Kate O'Neill

May 4, 2026

Assessing Bicycle Saddle Height from Smartphone Video of Passing Cyclists

Kate O'Neill, BA(Mod) in Science in Computer Science

University of Dublin, Trinity College, 2026

Supervisor: Kenneth Dawson-Howes

Every published system for assessing bicycle saddle height from video assumes a stationary cyclist or a fixed camera. This dissertation asks whether the same assessment can work on a hand-held smartphone recording of a cyclist riding past outdoors. The system described here has six stages. A fine-tuned YOLO detector finds the cyclist and both wheels in each frame; the relative geometry of the wheel bounding boxes filters out frames that are not roughly side-on. OpenPose extracts joint coordinates from the remaining frames, and a Savitzky–Golay filter smooths the resulting knee angle series. A representative peak angle is then compared against the accepted 145–155° range to produce a seat-height verdict and cadence estimate. Ten participants were filmed in three outdoor locations and on a static trainer. On trainer footage, cadence mean absolute error (MAE) was 0.4 revolutions per minute (RPM) and verdicts were consistent across cadences for every participant. On outdoor footage, cadence MAE was 8.5 RPM and the seat-height verdict matched the trainer reference in 16 of 19 recordings (84%). The three mismatches trace back to perspective distortion or borderline angles near the verdict threshold. The side-angle filter reduced the number of frames sent to OpenPose by a factor of 7.5.

Acknowledgments

I would like to thank Dr. Kenneth Dawson-Howe for the weekly support that he offered throughout the course of this project.

Kate O'Neill

University of Dublin, Trinity College
May 2026

Contents

Abstract ii

Acknowledgments iii

List of Acronyms vii

Chapter 1 Introduction 1

Chapter 2 State of the Art 3
2.1 Background . . . 3
2.1.1 Saddle-Height Biomechanics . . . 3
2.1.2 Markerless Pose Estimation . . . 4
2.1.3 Object Detection . . . 5
2.1.4 Kinematic Signal Processing . . . 7
2.2 Related Work . . . 7
2.3 Summary . . . 8

Chapter 3 Design 9
3.1 Problem Formulation . . . 9
3.1.1 Identified Challenges . . . 9
3.1.2 Proposed Work . . . 10
3.2 Overview of the Design . . . 10
3.3 Summary . . . 11

Chapter 4 Implementation 13
4.1 Overview of the Solution . . . 13
4.1.1 Stage Structure and Intermediate JavaScript object notation (JSON) 13
4.1.2 Shared Utilities . . . 13
4.2 Side-Angle Frame Selection . . . 14
4.2.1 Custom YOLO Model . . . 14
4.2.2 Side-Angle Selection . . . 16
4.2.3 *Burst* Window Selection . . . 17

4.2.4 Outputs . . . 17
4.3 Pose Estimation . . . 17
4.3.1 Frame Preparation . . . 17
4.3.2 OpenPose Body25 Inference . . . 18
4.3.3 Cyclist Box Estimation . . . 18
4.3.4 Output Schema . . . 19
4.4 Knee Analysis . . . 19
4.4.1 Direction Detection . . . 19
4.4.2 Run Segmentation . . . 20
4.4.3 Knee Angle Computation . . . 20
4.4.4 Savitzky-Golay Filter and Peak Detection . . . 20
4.4.5 Autocorrelation . . . 20
4.4.6 Output Schema . . . 21
4.5 Seat Height Assessment . . . 21
4.5.1 Peak Angle Extraction . . . 21
4.5.2 Verdict Logic . . . 21
4.6 Cadence Estimation . . . 22
4.6.1 Cadence Calculation . . . 22
4.6.2 Autocorrelation revolutions per minute (RPM) . . . 22
4.7 Annotation . . . 23
4.8 Summary . . . 23

Chapter 5 Evaluation **24**
5.1 Dataset . . . 24
5.2 Cyclist and Wheel Detection . . . 25
5.2.1 Detection Accuracy . . . 25
5.2.2 Class Discrimination . . . 26
5.3 Frame Selection . . . 26
5.3.1 Selection Rate . . . 26
5.3.2 Processing Time Reduction . . . 26
5.3.3 Frame Selection Thresholds . . . 27
5.3.4 Filming Location Comparison . . . 28
5.4 Pose Estimation . . . 28
5.4.1 Detection Rate in Selected Frames . . . 29
5.4.2 Joint Confidence . . . 29
5.5 Knee Angle Analysis . . . 29
5.5.1 Savitzky-Golay Window and Peak Prominence Evaluation . . . 30
5.5.2 Consistency between Filming Conditions . . . 31
5.6 Cadence . . . 33
5.6.1 RPM Accuracy . . . 33
5.7 Seat-Height . . . 33

5.7.1 Verdict Agreement 34
5.7.2 Percentile Selection Sensitivity 34
5.8 Failure Cases 34
5.9 Comparison with Related Work 35
5.10 Summary 36

Chapter 6 Conclusions & Future Work **37**
6.1 Conclusions 37
6.2 Future Work 38

Bibliography **39**

List of Acronyms

RPM revolutions per minute

BPM beats per minute

YOLO "You Only Look Once"

COCO Common Objects in Context

MUGA multi use games area

MAE mean absolute error

ROI region of interest

CLAHE contrast-limited adaptive histogram equalisation

JSON JavaScript object notation

mAP mean average precision

Chapter 1

Introduction

A bicycle saddle set at the wrong height increases the risk of knee injury and reduces pedalling efficiency [1]. The accepted way to check saddle height is to measure the knee extension angle at the bottom of the pedal stroke: an angle between 145° and 155° indicates a correct setup [2]. Markerless pose estimators such as OpenPose [3] can extract this angle from video with errors of 4–8° [4], but every published system that does so assumes a stationary cyclist on a trainer or a fixed camera. This dissertation asks whether the same assessment can be carried out from a hand-held smartphone recording of a cyclist riding past outdoors.

Moving outdoors with a hand-held phone introduces problems that do not arise on a trainer: the cyclist must be detected in the frame, only near-side-on frames give usable joint positions, observation windows are short, and distance and motion blur lower pose-estimator confidence. Chapter 3 analyses these challenges in detail.

To address them, this dissertation presents a six-stage system that detects the cyclist and filters for side-on frames, runs markerless pose estimation on the selected frames, extracts peak knee extension angles from the smoothed angle series, and compares a representative angle against the 145–155° range to produce a seat-height verdict and cadence estimate.

The system was tested on 10 participants filmed in three outdoor locations as well as on a static trainer. On controlled trainer footage, cadence mean absolute error (MAE) was 0.4 RPM and every participant received a consistent verdict across two cadences. On uncontrolled outdoor footage, cadence MAE was 8.5 RPM and the seat-height verdict matched the controlled reference in 16 of 19 recordings (84%). The three mismatches trace back to perspective distortion or borderline angles near the threshold. The fine-tuned detector scored an mean average precision (mAP)50 of 0.876 on held-out frames, and the side-angle filter reduced the number of frames sent to OpenPose by a factor of 7.5.

Outdoor saddle-height assessment from a smartphone is therefore feasible, though not as accurate as a controlled trainer setup. The system does not replace a professional

bike fit, but it can give a cyclist a rough, equipment-free check. Perspective distortion remains the main source of error and the clearest target for future work.

The remainder of this report is organised as follows. Chapter 2 reviews the biomechanics of saddle height, markerless pose estimation, object detection, and signal processing, and identifies the gap in the literature. The design chapter (Chapter 3) turns that gap into a concrete engineering problem and presents the system architecture. Implementation details for each stage follow in Chapter 4. The evaluation in Chapter 5 tests the system against ground-truth cadence measurements and controlled-condition verdicts. Chapter 6 draws conclusions and outlines future work.

Chapter 2

State of the Art

This chapter reviews the background knowledge and existing work that this dissertation builds on. Section 2.1 covers the four technical areas needed to assess saddle height from video: cycling biomechanics, markerless pose estimation, object detection, and signal processing. Section 2.2 looks at published systems that apply these techniques to cycling and identifies the gap this dissertation addresses.

2.1 Background

Assessing saddle height from video draws on several fields. This section covers the relevant biomechanics, how joint angles can be pulled from video, how objects of interest are found in a frame, and how noisy angle measurements are smoothed into usable data.

2.1.1 Saddle-Height Biomechanics

A saddle that is too low produces a knee angle below 145° at the six-o'clock crank position [5], increasing patellofemoral compression and reducing power output [1]. A saddle that is too high pushes the knee angle above 155°, over-extending the knee and causing hip rocking, which hurts efficiency and risks injury [1].

Methods for setting saddle height fall into two groups. Anthropometric methods base the height on a body measurement: the Hamley and Thomas method [6] uses 1.09 times the inseam length, and the LeMond method [7] uses 0.883 times the inseam. Both need only a tape measure but ignore differences in crank arm length and pedalling style. Kinematic methods instead measure joint angles while the rider is on the bicycle. The Holmes method [2] targets a knee-flexion angle of 25–30° at the six o'clock crank position, equivalent to 150–155° of knee extension.

Bini, Hume, and Croft [1] reviewed sixty-two studies and recommended the Holmes

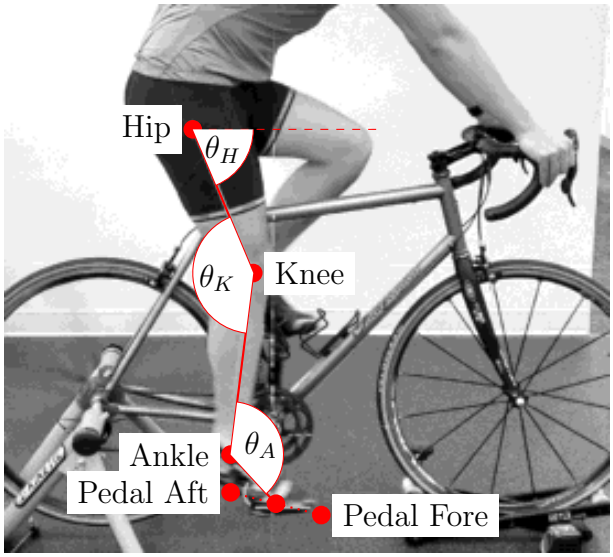


Figure 2.1: Hip-knee-ankle angle described in [9].

range, measured statically with a goniometer. Bini and Hume [8] later compared static and dynamic measurements in thirty cyclists. At six-o'clock the hip, knee, and ankle angles differed by $5 \pm 1°$, $8 \pm 2°$, and $9 \pm 2°$ respectively; at three-o'clock the differences were not significant. The static six-o'clock measurement endorsed in the earlier review therefore underestimates the dynamic knee angle by roughly $8°$.

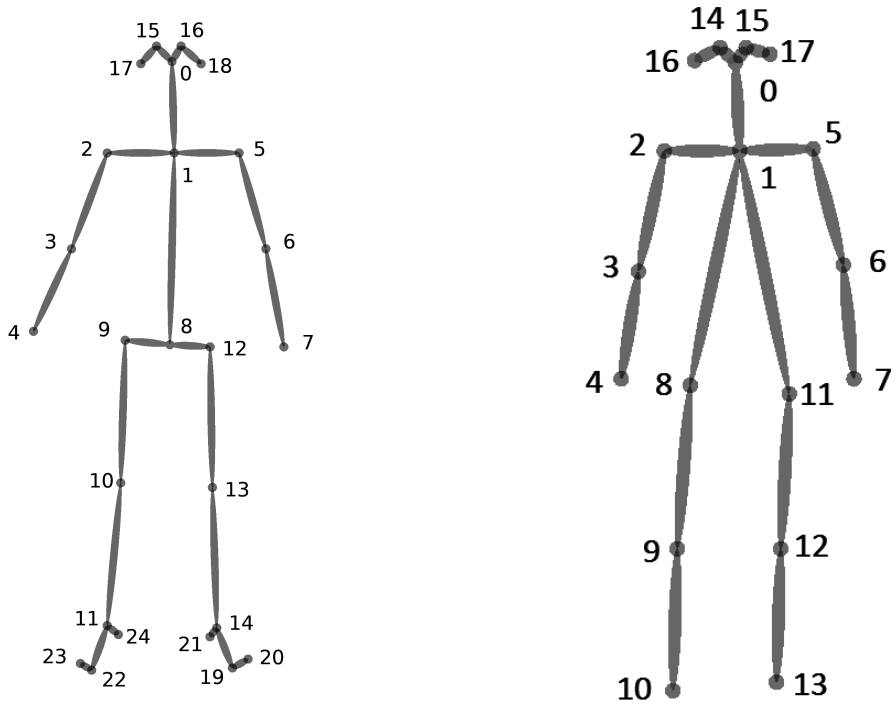
2.1.2 Markerless Pose Estimation

Pose estimation locates a person's joints in an image or video frame. Marker-based motion capture, where reflective markers are placed on the skin and tracked by a calibrated multi-camera system, remains the gold standard for biomechanical analysis [10], but the laboratory setup puts it out of reach for most cyclists. Depth-sensor systems such as the Microsoft Kinect removed the need for markers but still required specialised hardware [11].

Markerless estimators take one of two approaches. *Bottom-up* methods find every visible joint in the frame first and then group them into skeletons; this scales well to crowds but has to solve a harder association problem. *Top-down* methods first detect each person with a bounding box and then estimate joints within that crop, which makes the pose problem simpler but ties runtime to the number of people in the scene.

Three markerless estimators are relevant to this work. OpenPose [3] is a bottom-up method that finds every joint in the frame and groups them into person-level skeletons using Part Affinity Fields. Its Body25 output format (Figure 2.2a) produces twenty-five keypoints per person. MediaPipe Pose [12], built on the BlazePose architecture,

is a lightweight top-down estimator designed for on-device real-time use; it produces thirty-three keypoints but only handles single-person scenes. YOLO-Pose [13] is a single-stage estimator built on the "You Only Look Once" (YOLO) detection framework. It reports a higher AP50 on the Common Objects in Context (COCO) benchmark [14] than OpenPose, but has seen far less independent validation: at the time of writing OpenPose has over 8,400 Google Scholar citations compared with roughly 680 for YOLO-Pose. That larger body of independent work gives more confidence in OpenPose's accuracy.



(a) Body25 Skeleton. (b) COCO Skeleton.

Figure 2.2: OpenPose Output Formats.

2.1.3 Object Detection

Object detection finds and classifies regions of interest in an image, returning a bounding box and class label for each instance. Two-stage detectors like Faster R-CNN [**renFasterRCNN** first propose candidate regions and then classify each one, giving high accuracy but slower inference. YOLO, introduced by Redmon et al. [15], reframed detection as a single regression problem: one forward pass through a convolutional network predicts bounding boxes and class probabilities at the same time, trading some accuracy for much faster inference.

The YOLO family has gone through twelve major revisions since 2016. Early versions split the input image into a fixed grid and gave each cell a set of pre-defined anchor boxes whose dimensions were adjusted during training (Figure 2.3). YOLOv8 dropped this anchor-based design and instead predicts object centres directly, simplifying training and improving accuracy on small objects. Later versions added attention-based feature fusion, which lets the network focus on the most informative parts of the feature map when combining information across scales [16].

These models typically ship with weights pre-trained on large-scale benchmarks like COCO, which contains eighty object classes [14]. Transfer learning [17] makes it possible to adapt a pre-trained network to a new task by replacing the final classification layer and fine-tuning on a smaller, task-specific dataset. Because the early convolutional layers already capture general visual features like edges, textures, and shapes, only the later layers need to learn new representations. This cuts both the amount of labelled data and the training time needed.

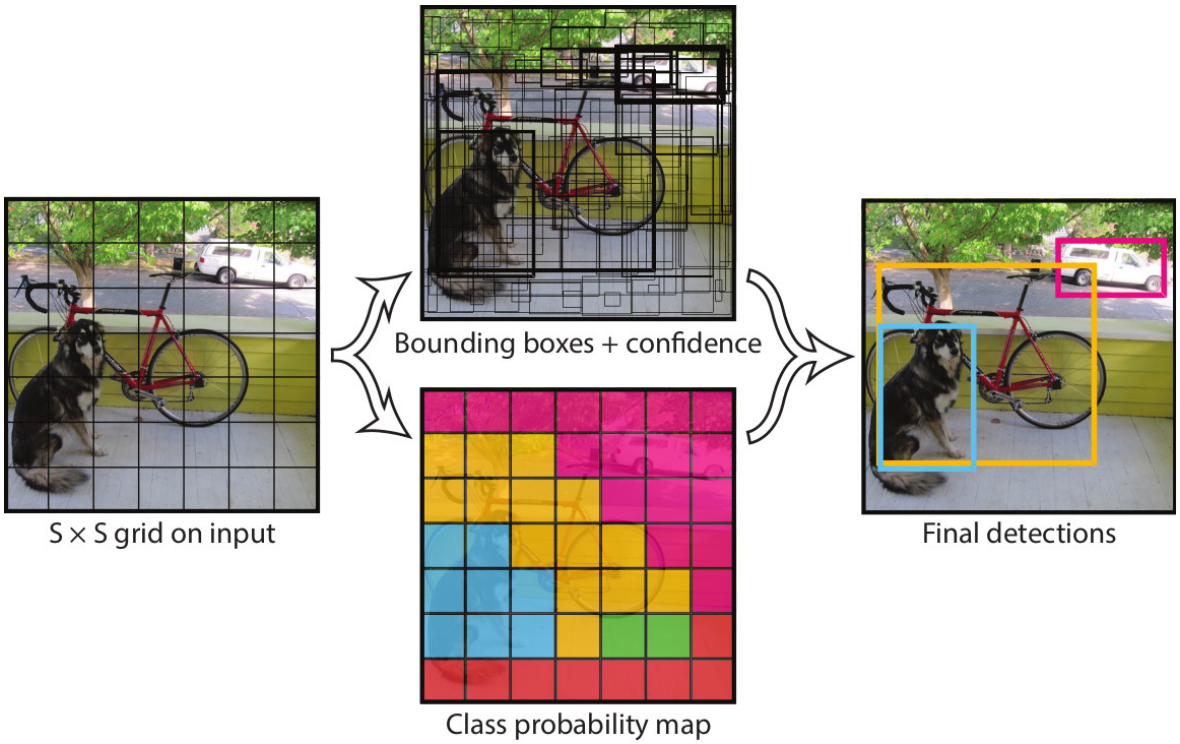


Figure 2.3: Early YOLO detection model [15].

Cyclist detection in unconstrained imagery is well established, but existing models and datasets are aimed at autonomous driving and traffic safety. Saranya et al. [18], for example, applied Tiny YOLOv2 to the Tsinghua-Daimler cyclist benchmark, a dataset captured from a vehicle-mounted camera. No published model or dataset was found

for detecting a cyclist riding past from a bystander's roadside perspective.

2.1.4 Kinematic Signal Processing

Kinematic time series, whether from marker-based systems or markerless pose estimators, contain noise from measurement error and distortion. Smoothing is therefore a standard preprocessing step in biomechanical analysis [19]. Common choices include moving-average filters, which replace each sample with the mean of its neighbours, and Gaussian filters, which weight neighbours with a bell-shaped kernel. Both reduce high-frequency noise but also flatten sharp peaks, shifting the true extrema of the signal. The Savitzky–Golay filter [20] works differently: it fits a low-order polynomial to a sliding window of samples by least-squares regression and takes the polynomial's value at the centre of the window. Because the polynomial can follow the curvature of a peak rather than averaging across it, the filter keeps peak shape and amplitude intact while still suppressing noise. This makes it a good fit for kinematic signals where the goal is to find the extremum of a periodic waveform, such as maximum knee extension during a pedal cycle.

2.2 Related Work

Markerless pose estimation has been applied to cycling biomechanics in several studies, but all assume a controlled filming environment. Bini et al. [21] compared OpenPose and MSRA-net against manually digitised markers in fourteen cyclists on a stationary trainer and found a mean OpenPose bias of 0.03–1.81° at the three-o'clock and six-o'clock crank positions. In a follow-up, Bini et al. [4] looked at the full pedal cycle and reported 4–8° differences at the hip and knee, rising to 15–22° at the ankle. Pagnon et al. [22] presented Pose2Sim, an open-source system that triangulates 2D poses from multiple synchronised cameras into 3D joint coordinates and feeds them into OpenSim, a musculoskeletal modelling toolkit. They tested the pipeline against marker-based capture across walking, running, and cycling, reporting a mean joint-angle error of 4.0° for cycling. Kakavand et al. [23] evaluated OpenCap, a markerless system that uses two synchronised iPhones and automatic musculoskeletal scaling to estimate joint kinematics without physical markers, and found per-joint errors below 4° on a stationary trainer.

Two recent systems use a single camera to assess bike fit on a trainer. Nasution et al. [24] used a fixed webcam and MediaPipe Pose to classify a road-bike fit as suitable or unsuitable, reporting relative angle errors below 1.8% at the elbow, hip, knee, and ankle. Zhang and Zhao [25] described a similar OpenPose-based system but gave no quantitative results. Ummu Habeeba et al. [26] combined MediaPipe and Keypoint R-CNN for cyclist pose estimation and reported a relative knee-angle error of 2.3% on

indoor footage.

Of these, Nasution et al. [24] is closest to this work: both use a single camera and a pose estimator to produce a saddle-height verdict. However, Nasution assumes a fixed webcam and a stationary trainer. Every system reviewed here shares at least one of two assumptions: the cyclist is stationary, or the camera is fixed. No published system operates on a hand-held smartphone recording of a cyclist passing the camera outdoors.

2.3 Summary

Several systems can now estimate cycling joint angles from video with errors of 4–8°, but all rely on a controlled environment. This dissertation addresses the gap by building a system that works on hand-held outdoor footage. The next chapter turns this into a concrete engineering problem. Table 2.1 compares the reviewed systems.

System	Filming	Pose model	Reported error
Bini et al.[21]	fixed, trainer	OpenPose	0.03–1.81°
Bini et al.[4]	fixed, static	OpenPose	4–8°
Pagnon et al.[22]	multiple	Pose2Sim	4.0°
Kakavand et al.[23]	multiple, trainer	OpenCap	<4°
Nasution et al.[24]	fixed, trainer	MediaPipe	<1.8%
Zhang & Zhao[25]	fixed, static	OpenPose	None
Ummu Habeeba et al.[26]	fixed, static	MediaPipe, R-CNN	2.3%
This work	hand-held, moving	OpenPose	Ch. 5

Table 2.1: Comparison of related systems.

Chapter 3

Design

This chapter turns the gap identified in Chapter 2 into a concrete engineering problem and presents a system design to address it. §3.1 sets out the problem and the challenges that come with filming outdoors. §3.2 walks through the six-stage pipeline designed to handle them.

3.1 Problem Formulation

As discussed in Chapter 2, saddle height can be judged by measuring the knee extension angle at the bottom of the pedal stroke and checking whether it falls within the 145–155° range [2]. Markerless pose estimators like OpenPose can already do this from video with errors of 4–8° [4], but every existing system assumes a stationary cyclist or a fixed camera. The question this dissertation addresses is whether the same verdict can be produced from a hand-held smartphone recording of a cyclist riding past outdoors.

3.1.1 Identified Challenges

Filming outdoors with a hand-held phone rather than a fixed camera on a trainer introduces several challenges that existing systems do not face.

1. **Cyclist detection.** The cyclist must first be found in the frame. No published detection model or dataset is designed for the bystander perspective used here; existing cyclist detectors are built for autonomous-driving scenarios [18].

2. **Viewing angle.** Measuring knee angle requires a side-on view, but a cyclist passing the camera is only roughly side-on for a brief part of the pass. Frames from other angles give distorted joint positions and must be discarded.

3. **Short observation windows.** A single pass produces far fewer frames than a trainer session and may capture only one or two full pedal cycles. Cadence estimation and peak-angle extraction must still work with this limited data.

4. **Reduced pose quality.** At greater distance the cyclist occupies fewer pixels, and lateral motion blur lowers keypoint confidence, especially at the ankle.

5. **Direction of travel.** The cyclist may ride in either direction, so the system needs to work out which leg faces the camera before choosing the correct joints.

3.1.2 Proposed Work

To tackle these challenges, this dissertation proposes a six-stage pipeline that takes a smartphone video and produces a saddle-height verdict and cadence estimate. The pipeline first detects the cyclist and works out their viewing angle so that only side-on frames reach the pose estimator (challenges 1, 2, and 5). It then extracts peak knee extension angles from what are often short and noisy observation windows (challenges 3 and 4). Finally, the representative angle is checked against the 145–155° range to produce a verdict. §3.2 walks through each stage.

3.2 Overview of the Design

Figure 3.1 shows the six stages and the data that flows between them. Each stage is a standalone script that reads its predecessor's output and writes an intermediate JSON file, so stages can be run and tested on their own.

Cyclist and Wheel Detection. A fine-tuned YOLO model runs on every frame and returns bounding boxes for the cyclist, front wheel, and back wheel. The relative position of the two wheels tells the system which direction the cyclist is travelling. A side-angle filter then keeps only those frames where the bounding-box aspect ratio and wheel separation suggest a near-side-on view. Consecutive qualifying frames are grouped into *burst* windows, scored, and only the best windows are kept.

Pose Estimation. Each selected frame is cropped to the cyclist bounding box, resized, and fed to OpenPose, which returns Body25 keypoints with per-joint confidence scores. Joints below a confidence threshold are dropped.

Knee Analysis. The hip, knee, and ankle coordinates are used to compute the knee extension angle via the law of cosines. A Savitzky–Golay filter smooths the resulting angle series and a peak detector picks out the bottom-of-stroke extension maxima. If

too few peaks are found, an autocorrelation fallback estimates the dominant period instead.

Seat-Height Verdict. A representative knee extension angle is taken from the smoothed series using a percentile-based method and compared against the 145–155° range: below 145° gives a `too_low` verdict, above 155° gives `too_high`, and values in between give `optimal`.

Cadence Estimation. Cadence in RPM comes from the time intervals between consecutive peaks. When the autocorrelation fallback was used, cadence is derived from the dominant period instead.

Annotation. The verdict, cadence, and skeleton overlay are drawn onto each frame of the original video to produce an annotated output file.

3.3 Summary

This chapter set out five challenges that come with moving saddle-height assessment outdoors and proposed a six-stage pipeline to handle them. Each stage produces an intermediate output for the next, so they can be developed and tested independently. The next chapter covers how each stage is implemented.

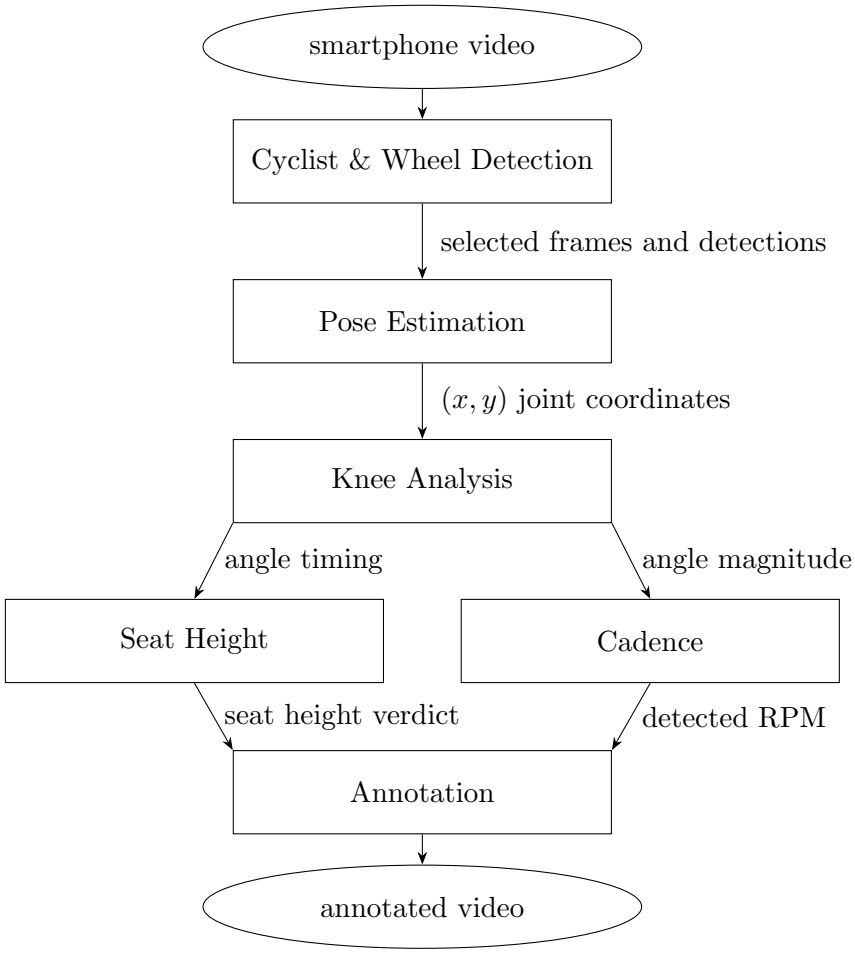


Figure 3.1: System diagram with each step of the designed system and their inputs and outputs.

Chapter 4

Implementation

This chapter walks through the implementation of the six-stage system designed in Chapter 3, covering the decisions behind each stage and the intermediate JSON files that connect them. The sections follow execution order, from the custom YOLO model and side-angle filter (§4.2), through pose estimation (§4.3) and knee analysis (§4.4), to the seat-height verdict (§4.5), cadence estimation (§4.6), and video annotation (§4.7). Chapter 5 evaluates the choices made here.

4.1 Overview of the Solution

Each stage from §3.2 lives in its own Python file. Stages are linked by intermediate JSON files and share a small set of utility functions.

4.1.1 Stage Structure and Intermediate JSON

Each stage is a standalone Python script that reads its predecessor's JSON output and writes its own to `output/<stem>/`. Because some stages are much slower than others, any stage can be run on its own without re-running the ones before it.

4.1.2 Shared Utilities

`utils.py` exports one constant and three functions used by every stage. `CONF_MIN = 0.1` is the minimum OpenPose confidence for a joint to count. It is deliberately low: the goal is to throw out joints that OpenPose missed entirely, not borderline detections, since downstream smoothing handles residual noise. `get_xy(keypoints, idx)` returns the pixel coordinates of joint `idx` if it passes `CONF_MIN`, or `None` otherwise. `calc_angle(A, B, C)` returns the interior angle at vertex B via the law of cosines (Equation 4.5). `video_stem(path, suffix)` derives the output stem from a file path.

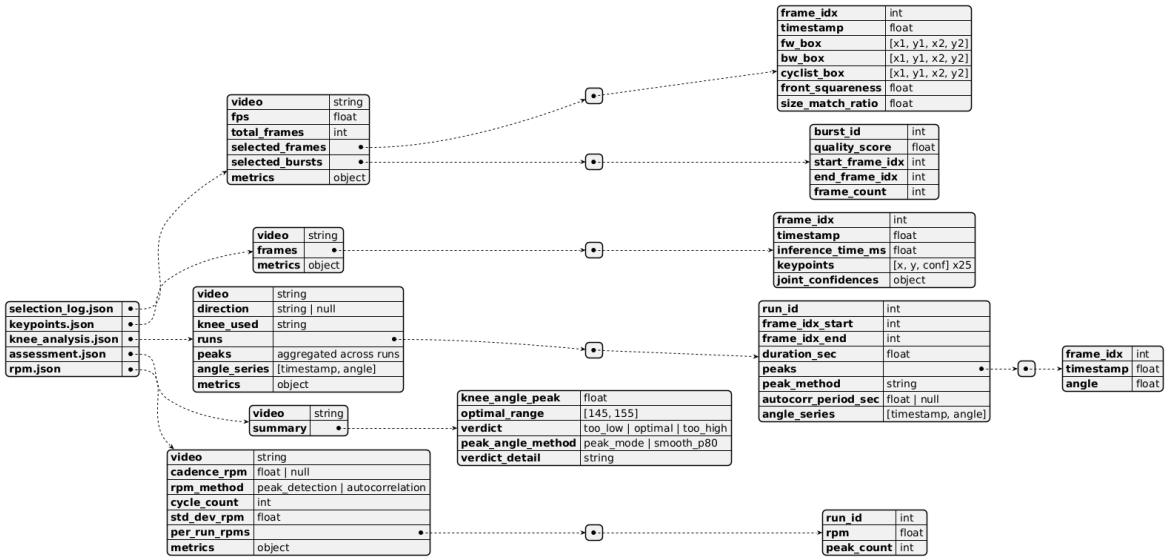


Figure 4.1: Intermediate JSON File Schemas.

4.2 Side-Angle Frame Selection

Stage 1 runs a custom YOLO model on every frame to find the ones where the cyclist is roughly side-on. Qualifying frames are grouped into *burst* windows, scored for quality, and only the best windows are kept for Stage 2.

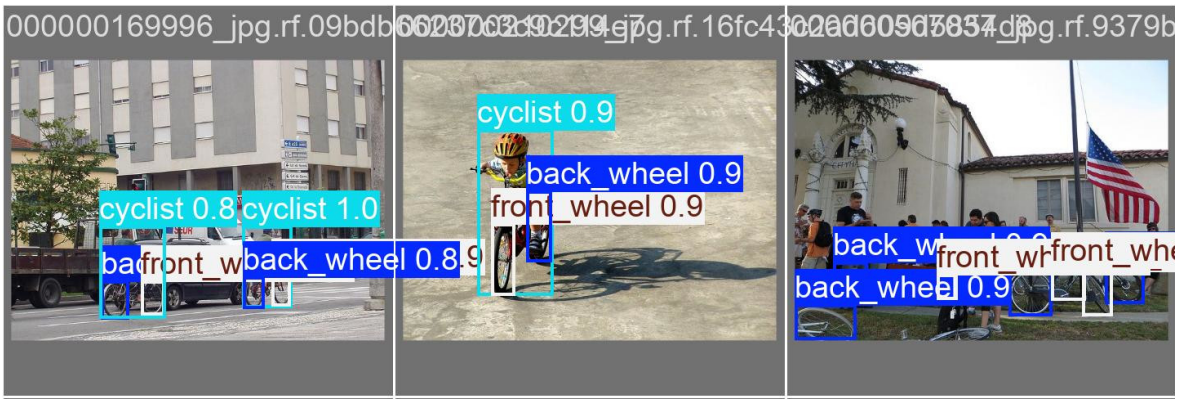
4.2.1 Custom YOLO Model

The model was trained on three classes: `cyclist`, `front_wheel`, and `back_wheel`. Training images came from the COCO dataset, filtered for annotations containing both `person` and `bicycle`. Wheel bounding boxes were drawn on all visible wheels regardless of whether the bicycle was being ridden. The final dataset had 1,754 images; 39 of these were taken from one controlled-footage video to give the model some exposure to that filming condition. Augmentation added three variants per image through randomised horizontal and vertical flips, which helped generalisation across cyclist orientations.

Training ran for 50 epochs, with peak mAP50 of 0.808 at epoch 44. Final-epoch precision was 0.844 and recall 0.698. The gap between them means the model is conservative: it rarely fires on something that is not there, which is what the system needs, since a false detection at frame selection is more disruptive than a missed one. The trained weights are saved as `best.pt`.



(a) Labels.



(b) Predictions.

Figure 4.2: Qualitative results from COCO validation batch.

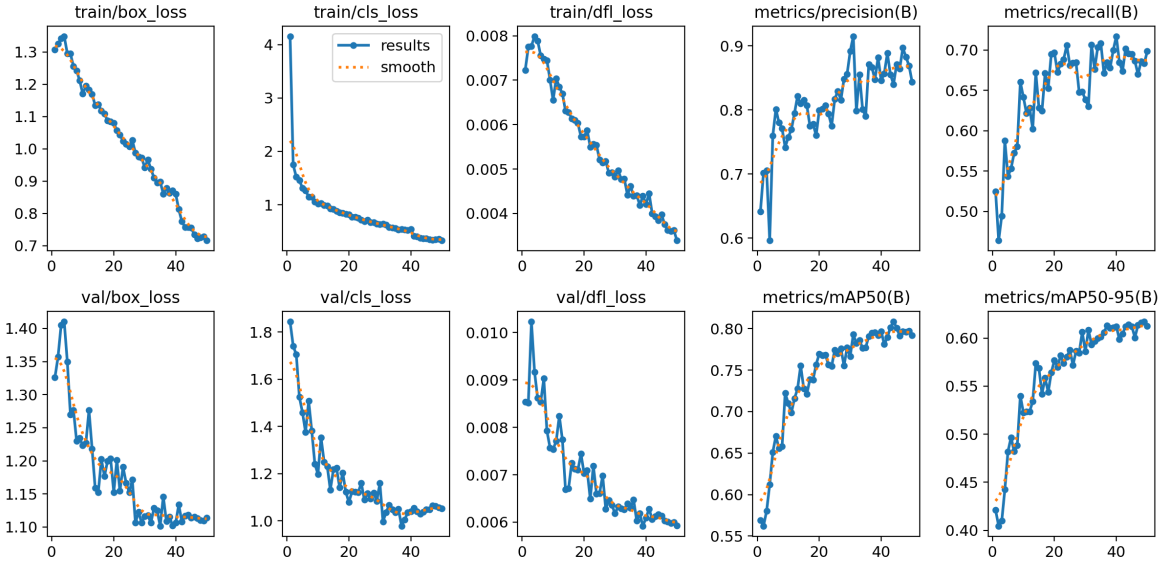


Figure 4.3: YOLO training curves over 50 epochs.

4.2.2 Side-Angle Selection

Every frame goes through the custom model at YOLO's default confidence threshold of 0.5. A frame counts as side-on only when both `front_wheel` and `back_wheel` are detected and two geometric conditions are met.

The first is *wheel squareness*. A circular wheel seen side-on projects as a circle; any off-axis angle foreshortens it and elongates the bounding box. Squareness is defined as

$$s = \frac{\min(w, h)}{\max(w, h)} \tag{4.1}$$

where w and h are the bounding-box dimensions. A perfect circle gives $s = 1$; both wheels must satisfy $s \geq 1 - \texttt{SQUARE_TOL} = 0.85$.

The second is *wheel size-match*. When the camera is truly side-on, both wheels are the same distance away and project to equal areas. Size-match is defined as

$$r = \frac{\min(A_f, A_b)}{\max(A_f, A_b)} \tag{4.2}$$

where A_f and A_b are the front and back wheel bounding-box areas; the threshold $r \geq 1 - \texttt{SIZE_TOL} = 0.80$ is applied. This is more lenient than squareness because projected wheel area shifts with lens distortion and distance even when the angle is genuinely side-on. §5.3.3 evaluates the effect of varying `SQUARE_TOL`.

4.2.3 *Burst* Window Selection

Frames that pass the filter are grouped into *bursts*: runs of strictly consecutive accepted frames. A single-frame gap splits a burst into two separate windows.

Each window gets a quality score:

$$Q = N \times \bar{s} \times \bar{r} \times \bar{h} \tag{4.3}$$

where N is the number of frames in the window, $\bar{s}$ is the mean squareness averaged across both wheels, $\bar{r}$ is the mean size-match ratio, and $\bar{h} = \overline{h_{\text{cyc}}/H_{\text{frame}}}$ is the mean normalised cyclist height, defined as the height of the cyclist bounding box divided by the frame height, averaged over all frames in the window for which a `cyclist` detection was available.

All windows with $Q \geq$ `QUALITY_FRACTION` $\times Q_{\text{best}}$ and $N \geq 5$ are retained, where `QUALITY_FRACTION` is set to 0.5. §5.3.3 evaluates the effect of varying this threshold across values from 0.5 to 1.0.

4.2.4 Outputs

Each accepted frame is written as a JPEG to `selected_frames/`. Frames from windows that do not survive the quality threshold are deleted after scoring.

A JSON log is written alongside the frames; its schema is shown in Figure 4.1. Stage 2 reads `cyclist_box` from this log to crop each frame to the cyclist region without a second YOLO pass.

4.3 Pose Estimation

Stage 2 preprocesses each selected frame and runs OpenPose to extract a 25-joint Body25 skeleton (Figure 2.2a). The keypoints are transformed back to original-frame coordinates and written to `<stem>_keypoints.json` for Stage 3.

4.3.1 Frame Preparation

Each selected frame goes through four preprocessing steps before OpenPose, in the order shown in Figure 4.4.

First, the frame is cropped to a region of interest (ROI) defined by the cyclist bounding box from the selection log. A 5% margin (based on the larger of box width and height) is added on each side and clamped to frame bounds, so the cyclist ends up filling roughly 80% of the crop.

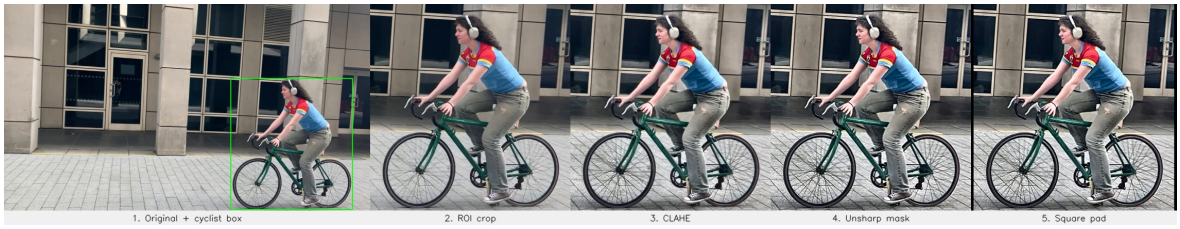


Figure 4.4: Preprocessing applied to selected frames before OpenPose inference.

contrast-limited adaptive histogram equalisation (CLAHE) is then applied to the L-channel of the LAB colour space with OpenCV's defaults (clip limit 2.0, tile grid 8×8) to sharpen limb boundaries without shifting hue.

Next, an unsharp mask with $\sigma = 3$ and strength 0.5 ($\mathbf{I}' = 1.5\,\mathbf{I} - 0.5\,\mathrm{blur}(\mathbf{I})$) partially restores edge sharpness lost to motion blur. Both values were set by visual inspection of preprocessing montages.

Finally, the crop is padded with black pixels to a square (storing offsets Δx_{left} and Δy_{top}) and resized to 656×368 pixels, the default resolution for OpenPose [27]. Resizing a non-square crop directly would distort the aspect ratio and bias all subsequent angle calculations.

After inference, keypoint coordinates are mapped back to the original frame:

$$x_{\text{orig}} = x_{\text{pad}} - \Delta x_{\text{left}} + x_{\text{roi}}, \qquad y_{\text{orig}} = y_{\text{pad}} - \Delta y_{\text{top}} + y_{\text{roi}} \tag{4.4}$$

where $(x_{\text{roi}}, y_{\text{roi}})$ is the top-left corner of the ROI crop. This lets the annotation stage draw keypoints directly onto the source video.

4.3.2 OpenPose Body25 Inference

OpenPose runs in Body25 mode, producing a 25-joint skeleton per detected person. Six joints are used in the following stages: right hip (`RHip`), right knee (`RKnee`), right ankle (`RAnkle`), left hip (`LHip`), left knee (`LKnee`), and left ankle (`LAnkle`).

4.3.3 Cyclist Box Estimation

YOLO sometimes detects both wheels but misses the cyclist. Rather than skip these frames, the system builds a cyclist bounding box from the wheel positions. The width spans the outer edges of both wheel boxes. The height is that width scaled by the median height-to-width ratio of all frames in the selection log that do have a cyclist detection.

```
def estimate_cyclist_box(entry, aspect):
```

```
    fw_box = entry["fw_box"]
    bw_box = entry["bw_box"]
    x_left = min(fw_box[0], bw_box[0])
    x_right = max(fw_box[2], bw_box[2])
    wheel_bottom = max(fw_box[3], bw_box[3])
    width = x_right-x_left
    height = int(width*aspect)
    return [x_left, wheel_bottom-height, x_right, wheel_bottom]
```

Listing 4.1: Cyclist bounding box estimated from wheel positions when YOLO produces no cyclist detection. `aspect` is the median height-to-width ratio of frames that do have a detected box.

4.3.4 Output Schema

Stage 2 writes `<stem>_keypoints.json` (schema in Figure 4.1). Stage 3 reads `joint_confidences` from this file, filtering through the `CONF_MIN` gate in `utils.py` before computing knee angles.

4.4 Knee Analysis

Stage 3 reads `<stem>_keypoints.json` and produces `<stem>_knee_analysis.json`, which both Stage 4 and Stage 5 read directly.

4.4.1 Direction Detection

Direction is worked out per window by `load_direction_map`: for each frame the horizontal difference between front- and back-wheel centres is computed, and the median across the window gives the travel direction. A positive median picks the right hip-knee-ankle triplet; a negative one picks the left. Each window has its own assignment, so clips with passes in both directions are handled correctly.

```
def get_direction(entries):
    deltas = []
    for entry in entries:
        fb = entry.get("fw_box")
        bb = entry.get("bw_box")
        if fb and bb:
            fx = (fb[0]+fb[2])/2
            bx = (bb[0]+bb[2])/2
            deltas.append(fx-bx)
```

```
    if not deltas:
        return None
    deltas.sort()
    return "right" if deltas[len(deltas)//2] > 0 else "left"
```

Listing 4.2: Direction of travel resolved from the median difference between front-wheel and back-wheel horizontal coordinates across all frames in a window.

4.4.2 Run Segmentation

Within each *burst* window, a frame is kept only if the hip, knee, and ankle joints all pass `CONF_MIN` (confidence ≥ 0.1 and positive pixel coordinates). These frames are grouped into contiguous *runs*; a new run starts whenever the gap between consecutive valid frames exceeds `MAX_FRAME_GAP` $= 30$ frames. Smoothing and peak detection run separately per *run* so that interpolation across gaps does not create false peaks.

4.4.3 Knee Angle Computation

For each frame in a run, the knee extension angle is computed via the law of cosines on the hip-knee-ankle triangle. For hip H, knee K, and ankle A:

$$\theta = \arccos\left(\frac{|HK|^2 + |KA|^2 - |HA|^2}{2\,|HK|\,|KA|}\right) \tag{4.5}$$

4.4.4 Savitzky-Golay Filter and Peak Detection

Each run's knee angle series is smoothed with a Savitzky-Golay filter (window 11, polynomial order 2). Series shorter than 11 frames are left unsmoothed.

Peaks are found using `scipy.signal.find_peaks` with two constraints. Minimum prominence is $\max(25°, \sigma_{\text{run}})$, where σ_{run} is the standard deviation of the smoothed series; this raises the bar for noisier runs. A minimum inter-peak distance enforces an upper cadence bound of 130 RPM, or roughly 0.46 seconds between peaks. Both values are evaluated in § 5.5.1.

4.4.5 Autocorrelation

Runs with fewer than two peaks but at least `AUTOCORR_MIN_FRAMES` $= 10$ frames fall back to autocorrelation. The smoothed series is correlated against shifted copies of itself; the first secondary peak above a normalised height of 0.1 gives the period. The period is converted to RPM and checked against a 20–130 RPM range; §5.6 reports the MAE for this path versus peak detection.

4.4.6 Output Schema

Stage 3 writes `<stem>_knee_analysis.json` (schema in Figure 4.1). Stages 4 and 5 both read `runs[]` from this file.

4.5 Seat Height Assessment

Stage 4 reads peak angles from `<stem>_knee_analysis.json` and assigns a seat-height verdict by checking the peak knee extension at the bottom of the pedal stroke against the thresholds from §2.1.1.

4.5.1 Peak Angle Extraction

The peak angle is picked by one of two methods depending on how many peaks were found across all *runs*.

If the count is $\geq$ `PEAK_MODE_MIN_PEAKS` = 10, the angles are rounded to the nearest degree and the mode is taken (`peak_mode`).

With fewer peaks, the 80th percentile of the smoothed angle series across all runs is used instead. Both the percentile and the `PEAK_MODE_MIN_PEAKS` threshold are evaluated in §5.7.2.

```
if len(usable_peaks) >= PEAK_MODE_MIN_PEAKS:
    angles = np.array([p["angle"] for p in usable_peaks])
    values, counts = np.unique(
        np.round(angles).astype(int), return_counts=True
    )
    peak_angle = float(values[np.argmax(counts)])
    peak_angle_method = "peak_mode"
```

Listing 4.3: Peak angle computed as the mode of bottom-of-stroke angles rounded to the nearest degree.

4.5.2 Verdict Logic

The peak angle is then checked against the thresholds from §2.1.1: below 145° is `too_low`, above 155° is `too_high`, and anything in between is `optimal`. A `verdict_detail` string records the measured angle and which bound it violated.

Stage 4 writes `<stem>_assessment.json`. The `summary` object holds the frame count with valid angles, mean and standard deviation of the smoothed series, the peak angle and how it was derived, the optimal range, the verdict, and the detail string.

4.6 Cadence Estimation

Stage 5 reads `<stem>_knee_analysis.json` and turns the inter-peak intervals from Stage 3 into a cadence estimate in RPM, written to `<stem>_rpm.json`.

4.6.1 Cadence Calculation

For each run with at least two peaks, the time gaps between consecutive peak timestamps are collected. These inter-peak periods are pooled across all usable runs; the mean gives the average pedal stroke period $\overline{T}$, and cadence follows as:

$$\text{RPM} = \frac{60}{\overline{T}} \tag{4.6}$$

```
for run in runs:
    peaks = run.get("peaks",[])
    if len(peaks)>=2:
        ts = [p["timestamp"] for p in peaks]
        periods = [ts[i+1]-ts[i] for i in range(len(ts)-1)]
        per_run_rpms.append({
            "run_id": run["run_id"],
            "rpm": round(60.0/(sum(periods)/len(periods)),1),
            "peak_count": len(peaks),
        })
        all_periods.extend(periods)
```

Listing 4.4: Inter-peak periods collected from every usable run and aggregated into a single list for mean-period cadence estimation.

Pooling across runs makes sense because the cyclist holds roughly constant cadence within a recording, so peaks from different runs are all estimating the same thing. A per-run RPM is also stored in `per_run_rpms` for inspection. When more than one inter-peak period is available, a standard deviation is computed from the spread of all periods.

4.6.2 Autocorrelation RPM

When no run has two or more peaks, the autocorrelation period from Stage 3 is used instead. The first run with a valid `autocorr_period_sec` is taken and RPM is computed as $60/T_{\text{autocorr}}$. The `rpm_method` field in the output records which path was used: `"peak_detection"` or `"autocorrelation"`.

4.7 Annotation

Stage 6 reads the source video alongside the keypoints, assessment, and RPM JSON files and writes an annotated copy to `<stem>_final.mp4`.

On selected frames it draws the Body25 skeleton, knee extension arcs, and hip flexion arcs, each labelled with the computed angle. Non-selected frames pass through untouched. Every frame gets a verdict banner, the cadence result, and a knee angle graph.

4.8 Summary

All six stages follow the same pattern: read the previous stage's JSON, do the work, write your own. Any stage can be re-run on its own without repeating what came before. Stage 3 sits at the centre; both Stage 4 and Stage 5 read its output directly rather than re-deriving angles or peaks. Chapter 5 tests the parameter choices made here and measures how well the system as a whole performs against ground truth.

Chapter 5

Evaluation

This chapter tests each pipeline stage against the design in Chapter 3, working from detection accuracy through to the final seat-height verdict, and compares the results with related work.

5.1 Dataset

Ten participants were filmed wearing ordinary clothing, aged between 20 and 35. Cycling ability was not a selection criterion; participants ranged from occasional cyclists to a professional bicycle courier.

For a controlled benchmark, the smartphone was mounted on a tripod and the bicycle held in a static trainer, following the setup used by [24]. Two videos were recorded per participant: one at roughly 60 RPM and one at roughly 90 RPM.

Pass-by footage was gathered in three locations: a pedestrian street, a private car park, and a multi use games area (MUGA). The pedestrian street (Figure 5.1a) gave close-up views but only a short window of frames before the cyclist left the shot. The car park allowed wider angles and longer tracks at greater distance. The MUGA was an open area where participants tended to cycle in loops, giving more varied viewing angles in a single recording.

Cadence ground truth was measured in Kinovea by marking the frame at which each pedal stroke reached the bottom; the count of complete revolutions divided by elapsed time gives the mean RPM. Participants were asked to match metronome audio at 60 and 90 beats per minute (BPM), but this proved difficult, so the actual cadences deviate from those targets. A dedicated cadence sensor would have provided more precise ground truth and is an avenue for future work.



(a) Uncontrolled footage.



(b) Controlled footage.

Figure 5.1: Filming conditions.

5.2 Cyclist and Wheel Detection

The fine-tuned YOLO detector was evaluated on a held-out set of 173 manually annotated frames extracted from three uncontrolled videos that the model had not seen during training.

5.2.1 Detection Accuracy

Table 5.1 shows the results. Overall mAP50 was 0.876. The cyclist class is the easiest to detect (mAP50 = 0.952, precision = 1.00); the two wheel classes scored lower at 0.815 and 0.860. All three beat the COCO validation mAP50 of 0.808 from training (Figure 4.3), which makes sense: the COCO split includes cyclists at varied distances, partial occlusions, and bikes without riders, while the held-out test frames are all side-on riders with the bicycle fully visible.

Class	Precision	Recall	mAP50	mAP50–95
All	0.919	0.789	0.876	0.722
Cyclist	1.000	0.859	0.952	0.815
Front wheel	0.923	0.730	0.815	0.675
Back wheel	0.833	0.776	0.860	0.676

Table 5.1: Per-class detection results.

5.2.2 Class Discrimination

Most errors are wheel misses at oblique angles, which the side-angle filter later discards anyway. 12% of front wheel instances are misclassified as back wheels versus only 6% the other way around. Since direction of travel is determined from the overall pattern across a full window, occasional label swaps do not affect the result.

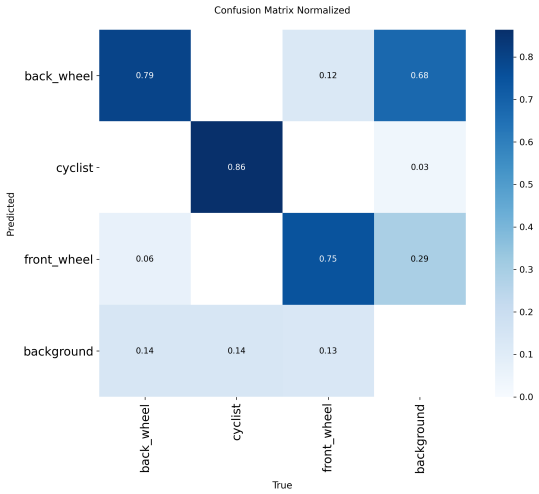


Figure 5.2: Normalised confusion matrix.

5.3 Frame Selection

Selection rate shows how much usable data each video produces and how many frames reach pose estimation, the most expensive stage.

5.3.1 Selection Rate

Selection rate varies by location: pedestrian street videos are lowest at 7.6%, each containing a single pass; car park videos average 12.7%, typically yielding two *burst* windows from a return pass; and MUGA videos are highest at 16.6%, yielding up to four windows as cyclists looped around the space. Two pedestrian-street videos produce windows too short for the cadence stage and are covered in §5.8.

5.3.2 Processing Time Reduction

Table 5.2 summarises the frame counts across the 19 uncontrolled videos.

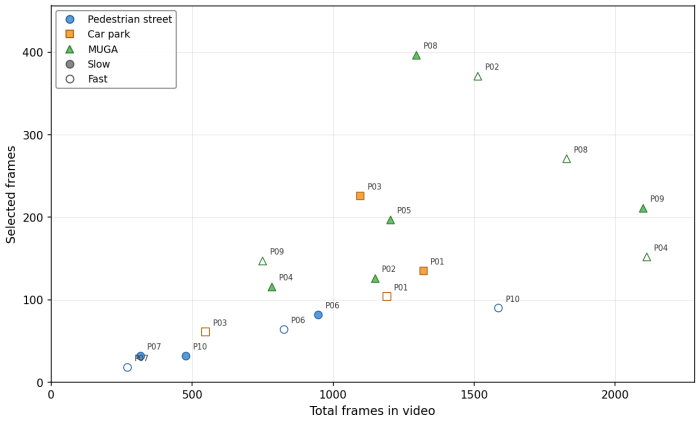


Figure 5.3: Selected frames versus total frames for each uncontrolled video.

Videos	19
Total frames	21,307
Frames sent to OpenPose	2,832
Selection rate	13.3%
OpenPose call reduction	86.7%

Table 5.2: Frame counts before and after the side-angle selection across uncontrolled videos.

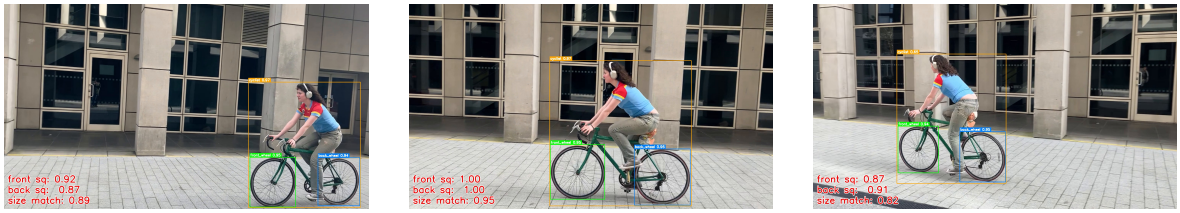
This 7.5× reduction cuts GPU processing time from about 21 minutes to under 3. YOLO takes roughly 33 ms per frame on CPU; OpenPose needs 60.1 ms per frame on GPU.

5.3.3 Frame Selection Thresholds

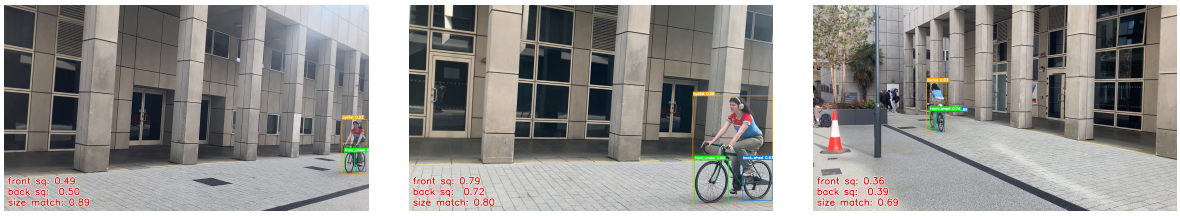
Both thresholds are defined in §4.2. Table 5.3 shows the effect of varying each from its final value. Coverage is the number of videos yielding valid RPM output; SH agree is the fraction of real-world videos whose seat-height verdict matches the trainer-recorded reference for the same subject.

Raising `QUALITY_FRACTION` to 1.0 drops seat-height agreement from 84% to 74% and loses one video entirely; the lower MAE at 1.0 comes from the smaller sample, not better predictions. The value 0.5 is kept.

Lowering `SQUARE_TOL` from 0.15 to 0.08 cuts coverage from 15 to 7 videos. MAE does not decrease consistently, and no threshold below 0.15 meets both coverage $\geq$14 and MAE reduction $\geq$0.5 RPM. The value 0.15 is kept.



(a) Accepted frames.



(b) Rejected frames.

Figure 5.4: Accepted and rejected frames from the side-angle selection stage.

5.3.4 Filming Location Comparison

Selection-window length varies by location (Figure 5.3), and this feeds through into cadence and seat-height accuracy.

On the pedestrian street, each recording gives a single window of fewer than 50 frames on average. That is often too short for two pedal-stroke peaks, so the system falls back to autocorrelation, which has an MAE of 12.4 RPM versus 6.6 RPM for peak detection. Both no-output cases (P07, P10) come from this location, where windows were too short to capture even one full pedal cycle.

Car-park recordings produce one or two windows at greater distance. The extra distance lowers OpenPose keypoint confidence, especially at the ankle, where mean confidence drops from 0.468 to 0.303 (Table 5.4), though the additional windows give more peaks for cadence estimation.

The MUGA produces the most data: up to four windows, a mean selection rate of 16.6%, and no failures.

5.4 Pose Estimation

Pose estimation is evaluated on person detection rate and joint confidence. Since OpenPose accuracy is already well validated in the literature reviewed in Chapter 2, ground-truth joint annotations were not collected for this dataset.

Value	Frames/video	RPM MAE	Coverage	SH agree
QUALITY_FRACTION				
0.5	149.1	10.6	15/19	16/19 (84%)
0.6	149.1	10.6	15/19	16/19 (84%)
0.7	130.8	10.6	15/19	16/19 (84%)
0.8	102.1	9.4	15/19	16/19 (84%)
0.9	102.1	9.4	15/19	16/19 (84%)
1.0	91.7	8.8	14/19	14/19 (74%)
SQUARE_TOL				
0.08	54.6	9.14	7/19	15/19 (79%)
0.10	82.3	8.02	11/19	15/19 (79%)
0.12	120.7	9.72	14/19	16/19 (84%)
0.15	149.1	8.99	15/19	15/19 (79%)

Table 5.3: Threshold selection results for `QUALITY_FRACTION` and `SQUARE_TOL`.

5.4.1 Detection Rate in Selected Frames

OpenPose either returns a full set of keypoints or nothing at all. Detection rate was 99.0% on controlled videos and 85.8% on uncontrolled ones. The drop comes from greater distance, lateral motion blur, and variable outdoor backgrounds.

5.4.2 Joint Confidence

Table 5.4 lists mean keypoint confidence and the proportion of frames above `CONF_MIN` for the six lower-limb joints. Confidence drops across the board in uncontrolled footage. The ankle takes the biggest hit: mean `RAnkle` confidence falls from 0.468 to 0.303 and usable detections from 88.5% to 73.4%. At greater distance the ankle occupies fewer pixels, and motion blur hits the extremities hardest.

As Figure 5.5 shows, the low-confidence joints are consistently the ones facing away from the camera, which is what you would expect from side-on frames.

5.5 Knee Angle Analysis

There is no direct ground truth for knee angles in this dataset, so the knee analysis stage is evaluated through its effect on cadence accuracy, which depends on the peaks it detects.

	Controlled		Uncontrolled	
Joint	Mean conf.	> 0.1	Mean conf.	> 0.1
RHip	0.396	98.1%	0.355	92.5%
RKnee	0.594	94.7%	0.395	79.1%
RAnkle	0.468	88.5%	0.303	73.4%
LHip	0.448	99.5%	0.369	93.7%
LKnee	0.693	98.5%	0.449	85.2%
LAnkle	0.619	97.0%	0.365	77.0%

Table 5.4: OpenPose keypoint confidence and proportion of frames above `CONF_MIN` by filming condition.

5.5.1 Savitzky-Golay Window and Peak Prominence Evaluation

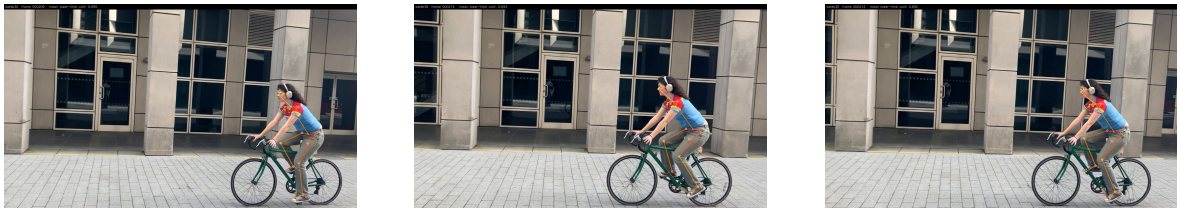
Two parameters affect peak detection: the Savitzky-Golay window length and the peak prominence floor. Table 5.5 shows the effect of varying each independently.

Value	Mean peaks/video	RPM MAE	SH agree
Savitzky-Golay window			
7	2.53	13.4	15/19 (79%)
9	2.32	10.9	16/19 (84%)
11	2.26	9.0	16/19 (84%)
13	2.11	10.1	16/19 (84%)
15	2.11	10.3	16/19 (84%)
Peak prominence floor / °			
10	2.37	9.1	16/19 (84%)
15	2.37	9.1	16/19 (84%)
20	2.26	9.0	16/19 (84%)
25	2.11	8.5	16/19 (84%)
30	2.00	7.5	16/19 (84%)

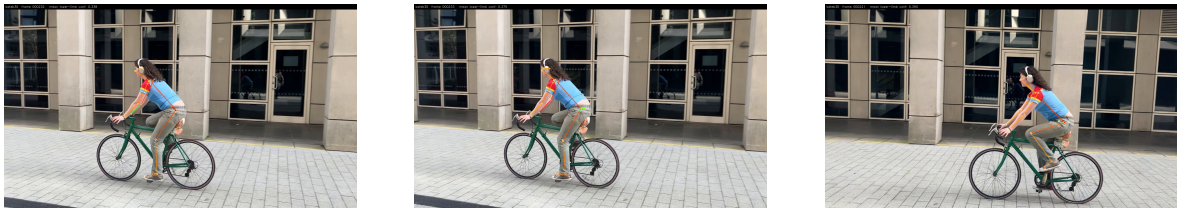
Table 5.5: Savitzky-Golay window and peak prominence floor test results.

Window 11 gives the lowest MAE. Window 7 lets through too much noise, and windows above 11 do not help further.

For prominence, MAE drops from 9.1 to 7.5 RPM as the floor rises from 10° to



(a) Highest confidence.



(b) Lowest confidence.

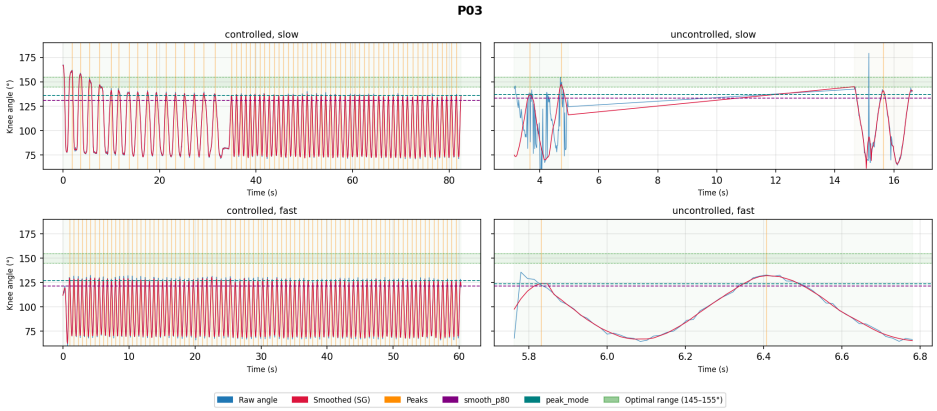
Figure 5.5: OpenPose keypoints on P07's uncontrolled frames ranked by mean lower-limb joint confidence.

30°. 30° looks best on MAE alone, but at that level mean peaks per video falls to 2.00, leaving no margin for noise. 25° keeps mean peaks at 2.11 and MAE at 8.5 RPM, so that value is used.

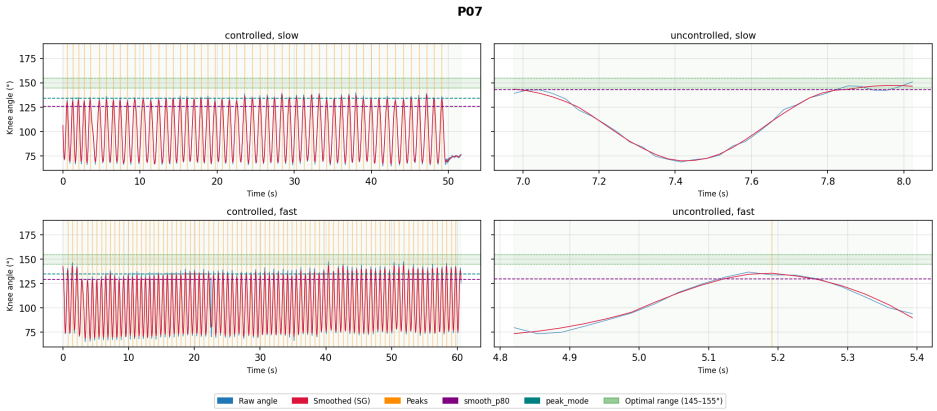
5.5.2 Consistency between Filming Conditions

Every recording of the same participant uses the same bicycle, so the verdict should not change between cadences or filming conditions. Under controlled conditions it does not: all participants get the same verdict at both slow and fast cadence (Figure 5.6), with a mean peak angle delta of 3.80°. The three largest deltas (P02 10.0°, P03 9.0°, P10 6.0°) still fall within the same verdict band. Agreement between controlled and uncontrolled recordings is covered in §5.7.

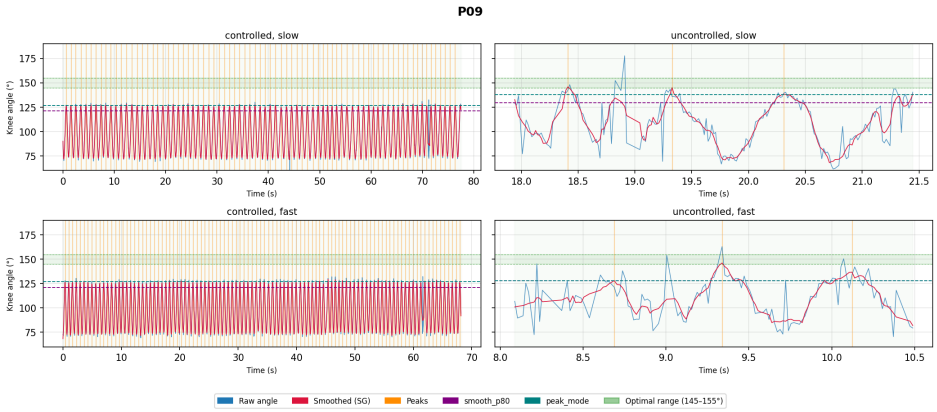
Figure 5.6 shows the smoothed knee angle series for three participants. In controlled recordings, `peak_mode` sits above `smooth_p80` because long trainer sessions yield many stable peaks. In uncontrolled recordings, short windows and noisier signals make individual peak angles more sensitive to distorted frames; `smooth_p80` gives a more conservative and accurate estimate. P09's slow uncontrolled recording illustrates this: `smooth_p80` aligns with the controlled `peak_mode` and produces the correct verdict.



(a) Car park.



(b) Pedestrian street.



(c) MUGA.

Figure 5.6: Raw and smoothed knee angle series for three participants across the three filming locations.

5.6 Cadence

5.6.1 RPM Accuracy

Of the 15 uncontrolled videos that produced output, MAE was 8.5 RPM (root mean squared error (RMSE) 12.4 RPM). Breaking this down: the 10 videos handled by peak detection had MAE of 6.6 RPM; the 5 that fell back to autocorrelation had 12.4 RPM. The autocorrelation error is higher because that path only runs when peak detection cannot find enough peaks to work with. On the 19 controlled videos, MAE was just 0.4 RPM, since the full pedal cycle is visible throughout.

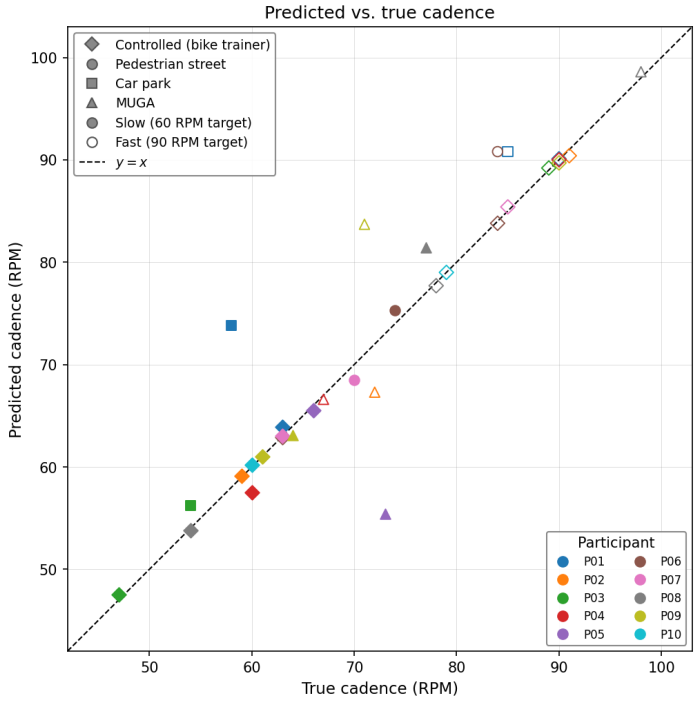


Figure 5.7: Predicted versus true RPM for uncontrolled videos.

5.7 Seat-Height

The goal here is to get the same seat-height verdict from uncontrolled footage as from controlled conditions. No bike-fitter measurement was taken, so the controlled-condition verdict is used as the reference. This is reasonable because OpenPose is checked against the COCO keypoint benchmark [3] and the controlled setup removes most of the imaging variability that makes outdoor footage harder. What is measured, then, is consistency between filming conditions, not correctness against an independent

biomechanical standard. It is also worth noting that static and dynamic knee extension angles differ systematically across individuals [8], so some of the observed delta between trainer and outdoor verdicts comes from genuine postural differences rather than system error.

5.7.1 Verdict Agreement

Table 5.6 shows the 3×3 confusion matrix for the 19 uncontrolled videos. No participant had a `too_high` reference verdict.

	Predicted		
Reference	`too_low`	`optimal`	`too_high`
`too_low`	13	0	1
`optimal`	1	3	1
`too_high`	0	0	0

Table 5.6: Seat-height verdict confusion matrix for 19 uncontrolled videos against controlled reference.

Overall agreement is 16/19 (84%). `too_low` recall is 0.929; `optimal` recall is only 0.600 because two borderline participants tipped into a different category in the uncontrolled recording. Any verdict close to the 145° or 155° threshold should be read with caution, since the range assumes a standard pedalling style and will not fit every rider.

5.7.2 Percentile Selection Sensitivity

The `smooth_pN` step takes the Nth percentile of the smoothed knee angle series. Table 5.7 shows verdict agreement against the controlled reference across p70–p95.

Agreement is flat across p70–p85, with p82 the best single value at 17/19. Above p87 it drops sharply because higher percentiles pick up false peaks from perspective distortion. p80 was chosen over p82 to avoid overfitting to this particular test set.

5.8 Failure Cases

Four uncontrolled videos produced no RPM output (Table 5.8). P02 slow is the most informative case: 126 frames are selected across three windows, but the knee angle trace is shallow and flat-topped, so no peaks exceed the 25° prominence threshold.

Percentile	Agree/19	%	Group 1	Group 2	Group 3
p70	15	79%	5/6	4/4	6/9
p75	15	79%	5/6	4/4	6/9
p80	16	84%	5/6	4/4	7/9
p82	17	89%	6/6	4/4	7/9
p85	16	84%	6/6	4/4	6/9
p87	13	68%	5/6	3/4	5/9
p90	9	47%	3/6	2/4	4/9
p95	7	37%	2/6	2/4	3/9

Table 5.7: Seat-height verdict agreement by percentile.

Video	Cause	Selected frames	Peaks	True RPM
P02 slow	No peaks above threshold	126	0	67
P07 fast	Window < 1 pedal cycle	18	1	85
P10 slow	Window too short	32	0	51
P10 fast	Cadence below range	90	1	39

Table 5.8: Uncontrolled videos with no RPM output.

The two worst RPM errors are P04 slow and P05 slow. P04 slow goes through autocorrelation, which returns double the true cadence. P05 slow uses peak detection, but the window is so short that the two detected peaks sit at an interval that underestimates cadence.

Two of the three disagreeing seat-height verdicts are caused by perspective distortion. P04 fast and P05 slow are assigned `too_high` because the uncontrolled peak angle is inflated to 157.5° and 165.5°, against controlled references of 135° and 151°. The third mismatch, P06 fast, is a borderline case; the uncontrolled estimate of 144.6° falls 0.4° below the 145° threshold.

5.9 Comparison with Related Work

None of the reviewed systems film under the same conditions, so direct comparison only goes so far. On controlled footage the numbers line up with prior work: Bini et al. [4] found 4–8° differences between OpenPose and marker-based capture on a trainer, and the controlled peak-angle deltas here (mean 3.80°) sit within that range. For uncontrolled footage there is no published benchmark to compare against.

Nasution et al. [24] is the closest comparable system; they reported angle errors below 1.8% using a fixed webcam and a stationary trainer. This system gets 84%

verdict agreement under harder outdoor conditions, and on the same controlled footage the verdicts are consistent across cadences for every participant. Table 2.1 in Chapter 2 compares all the reviewed systems.

5.10 Summary

The fine-tuned YOLO detector scored mAP50 of 0.876 on held-out frames, and the side-angle filter cut OpenPose calls by 7.5×. Cadence MAE was 0.4 RPM on controlled footage and 8.5 RPM on uncontrolled footage (6.6 for peak detection, 12.4 for autocorrelation). Seat-height verdict agreement between the two conditions was 16/19 (84%); the three mismatches trace back to perspective distortion or borderline angles. The next chapter discusses what these results mean and where the system falls short.

Chapter 6

Conclusions & Future Work

This chapter draws conclusions from the evaluation results and outlines directions for future work.

6.1 Conclusions

This final year project report set out to determine whether bicycle saddle height can be assessed from a hand-held smartphone recording of a cyclist riding past outdoors. The answer, within limits, is yes.

On controlled trainer footage the system produces consistent verdicts across cadences for every participant, with a cadence MAE of 0.4 RPM and controlled peak-angle deltas (mean 3.80°) that sit within the 4–8° range reported for OpenPose in prior work [4]. On uncontrolled outdoor footage, seat-height verdict agreement with the controlled reference was 16/19 (84%), and cadence MAE was 8.5 RPM. The fine-tuned YOLO detector scored mAP50 of 0.876 on held-out frames, and the side-angle filter reduced the number of frames sent to OpenPose by a factor of 7.5.

The system works best when the filming location gives a long, close, side-on view. MUGA and car-park recordings, which produced multiple windows and higher selection rates, gave more reliable results than pedestrian-street recordings, where a single short pass sometimes captured fewer than one full pedal cycle. The three verdict mismatches trace back to perspective distortion inflating the measured angle or to borderline angles within a degree of the threshold. Perspective distortion is not addressed by the current side-angle filter and accounts for two of the three incorrect verdicts.

Two design decisions are worth noting. The wheel-geometry filter is a cheap proxy for viewing angle: it removes roughly 87% of frames before they reach the pose estimator, saving GPU time without requiring any pose information. The choice between mode-based and percentile-based peak-angle extraction also matters. On controlled footage, where many stable peaks are available, the mode gives a tight estimate. On

outdoor footage, short windows and noisier signals make individual peaks less reliable, and the 80th-percentile method produces more conservative and more accurate verdicts.

The system does not replace a professional bike fit. The 145–155° range assumes a standard pedalling style and will not suit every rider. Static and dynamic knee extension angles differ systematically across individuals [8], so some of the observed delta between trainer and outdoor verdicts reflects genuine postural differences rather than measurement error. The system can, however, give a cyclist a rough idea of whether their saddle height falls in the accepted range, without the need for equipment.

6.2 Future Work

Mobile Application. The current system runs as a set of Python scripts on a desktop machine. Packaging it as an iOS application would make it accessible to the cyclists it is designed for. YOLO-Pose [13] combines object detection and pose estimation in a single forward pass, with faster inference times than OpenPose. The side-angle filter and knee analysis stages are lightweight and would transfer with minimal modification.

Custom Pose Model. OpenPose was chosen because it has been independently tested in cycling biomechanics more than any other markerless estimator, but it was not designed for the outdoor, moving-cyclist scenario this system targets. A pose model trained or fine-tuned on cyclist-specific data could improve keypoint confidence at the ankle, which is the weakest joint in the current system. One option is to combine a single-camera 2D estimator with the 2D-to-3D lifting approach used by Pose2Sim [22]: lifting 2D joint coordinates into 3D would make the angle measurement less sensitive to the viewing-angle distortion that caused two of the three verdict mismatches. Ummu Habeeba et al. [26] took a different approach, combining MediaPipe and Keypoint R-CNN and selecting the higher-confidence joints per frame; a similar fusion strategy could raise accuracy here without requiring a new training dataset.

Expanded Evaluation. The current dataset has 10 participants filmed in three locations. A larger dataset with more variation in body type, bicycle geometry, and lighting would give a clearer picture of where the system breaks down. Cadence ground truth was measured by manual frame counting in Kinovea; a dedicated cadence sensor attached to the crank would provide more precise reference values and allow tighter error bounds. Ground-truth joint angles from a goniometer or marker-based motion capture system would also allow direct measurement of angle error rather than the indirect assessment through verdict agreement used here.

Bibliography

[1] R. Bini, P. Hume, and J. Croft, "Effects of Bicycle Saddle Height on Knee Injury Risk and Cycling Performance," *Sports medicine (Auckland, N.Z.)*, vol. 41, pp. 463–76, Jun. 1, 2011. DOI: 10.2165/11588740-000000000-00000

[2] J. C. Holmes, A. L. Pruitt, and N. J. Whalen, "Lower extremity overuse in bicycling," *Clinics in Sports Medicine*, vol. 13, no. 1, pp. 187–205, Jan. 1994, ISSN: 0278-5919. PMID: 8111852.

[3] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh. "OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields." arXiv: 1812.08008 [cs], Accessed: Oct. 7, 2025. [Online]. Available: http://arxiv.org/abs/1812.08008 pre-published.

[4] R. R. Bini, G. Serrancoli, P. R. P. Santiago, A. Pinto, and F. Moura, "Criterion validity of neural networks to assess lower limb motion during cycling," *Journal of Sports Sciences*, vol. 41, no. 1, pp. 36–44, Jan. 2, 2023, ISSN: 0264-0414. DOI: 10.1080/02640414.2023.2194725 PMID: 36975046. Accessed: Oct. 11, 2025. [Online]. Available: https://doi.org/10.1080/02640414.2023.2194725

[5] W. W. Peveler, B. Shew, S. Johnson, and T. G. Palmer, "A Kinematic Comparison of Alterations to Knee and Ankle Angles from Resting Measures to Active Pedaling During a Graded Exercise Protocol," *The Journal of Strength & Conditioning Research*, vol. 26, no. 11, p. 3004, Nov. 2012, ISSN: 1064-8011. DOI: 10.1519/JSC.0b013e318243fdcb Accessed: Oct. 11, 2025. [Online]. Available: https://journals.lww.com/nsca-jscr/fulltext/2012/11000/a_kinematic_comparison_of_alterations_to_knee_and.13.aspx

[6] E. J. Hamley and V. Thomas, "Physiological and postural factors in the calibration of the bicycle ergometer," *The Journal of Physiology*, vol. 191, no. 2, 55P–56P, 1967, ISSN: 0022-3751.

[7] G. LeMond and K. Gordis, *Greg LeMond's Complete Book of Bicycling*. New York: Perigee Books, 1987, ISBN: 9780399513145.

[8] R. R. Bini and P. Hume, "A comparison of static and dynamic measures of lower limb joint angles in cycling: Application to bicycle fitting," *Human Movement*, vol. 17, no. 1, pp. 36–42, Apr. 4, 2018, ISSN: 1899-1955. DOI: 10.1515/humo-2016-0005 Accessed: Oct. 7, 2025. [Online]. Available: https://hummov.awf.wroc.pl/A-comparison-of-static-and-dynamic-measures-of-lower-limb-joint-angles-in-cycling,183244,0,2.html

[9] R. R. Bini, A. C. Tamborindeguy, and C. B. Mota, "Effects of saddle height, pedaling cadence, and workload on joint kinetics and kinematics during cycling," *Journal of Sport Rehabilitation*, vol. 19, no. 3, pp. 301–314, Aug. 2010, ISSN: 1056-6716. DOI: 10.1123/jsr.19.3.301 PMID: 20811079.

[10] B. Van Hooren, N. Pecasse, K. Meijer, and J. M. N. Essers, "The accuracy of markerless motion capture combined with computer vision techniques for measuring running kinematics," en, *Scand. J. Med. Sci. Sports*, vol. 33, no. 6, pp. 966–978, Jun. 2023.

[11] J. Shotton et al., "Real-Time Human Pose Recognition in Parts from Single Depth Images,"

[12] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann. "BlazePose: On-device Real-time Body Pose tracking." arXiv: 2006.10204 [cs], Accessed: Oct. 4, 2025. [Online]. Available: http://arxiv.org/abs/2006.10204 pre-published.

[13] D. Maji, S. Nagori, M. Mathew, and D. Poddar. "YOLO-Pose: Enhancing YOLO for Multi Person Pose Estimation Using Object Keypoint Similarity Loss." arXiv: 2204.06806 [cs], Accessed: Oct. 4, 2025. [Online]. Available: http://arxiv.org/abs/2204.06806 pre-published.

[14] T.-Y. Lin et al. "Microsoft COCO: Common Objects in Context." arXiv: 1405.0312 [cs], Accessed: Feb. 1, 2026. [Online]. Available: http://arxiv.org/abs/1405.0312 pre-published.

[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. "You Only Look Once: Unified, Real-Time Object Detection." arXiv: 1506.02640 [cs], Accessed: Oct. 7, 2025. [Online]. Available: http://arxiv.org/abs/1506.02640 pre-published.

[16] R. Sapkota et al., "YOLO advances to its genesis: A decadal and comprehensive review of the You Only Look Once (YOLO) series," *Artificial Intelligence Review*, vol. 58, no. 9, p. 274, Jun. 11, 2025, ISSN: 1573-7462. DOI: 10.1007/s10462-025-11253-3 Accessed: Oct. 9, 2025. [Online]. Available: https://doi.org/10.1007/s10462-025-11253-3

[17] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010. DOI: 10.1109/TKDE.2009.191

[18] K. C. Saranya, A. Thangavelu, A. Chidambaram, S. Arumugam, and S. Govindraj, "Cyclist Detection Using Tiny YOLO v2," in *Soft Computing for Problem Solving*, K. N. Das, J. C. Bansal, K. Deep, A. K. Nagar, P. Pathipooranam, and R. C. Naidu, Eds., Singapore: Springer, 2020, pp. 969–979, ISBN: 978-981-15-0184-5. DOI: 10.1007/978-981-15-0184-5_82

[19] F. Crenna, G. B. Rossi, and M. Berardengo, "Filtering Biomechanical Signals in Movement Analysis," *Sensors*, vol. 21, no. 13, p. 4580, Jan. 2021, ISSN: 1424-8220. DOI: 10.3390/s21134580 Accessed: May 4, 2026. [Online]. Available: https://www.mdpi.com/1424-8220/21/13/4580

[20] A. Savitzky and M. J. E. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Analytical Chemistry*, vol. 36, pp. 1627–1639, Jan. 1, 1964. DOI: 10.1021/ac60214a047 Accessed: May 4, 2026. [Online]. Available: https://ui.adsabs.harvard.edu/abs/1964AnaCh..36.1627S

[21] R. R. Bini, G. Serrancoli, P. R. P. Santiago, A. Pinto, and F. Moura, "Validity of Neural Networks to Determine Body Position on the Bicycle," *Research Quarterly for Exercise and Sport*, vol. 94, no. 4, pp. 905–912, Oct. 2, 2023, ISSN: 0270-1367. DOI: 10.1080/02701367.2022.2070103 PMID: 35575754. Accessed: Oct. 11, 2025. [Online]. Available: https://doi.org/10.1080/02701367.2022.2070103

[22] D. Pagnon, M. Domalain, and L. Reveret, "Pose2Sim: An End-to-End Workflow for 3D Markerless Sports Kinematics—Part 2: Accuracy," *Sensors*, vol. 22, no. 7, p. 2712, Jan. 2022, ISSN: 1424-8220. DOI: 10.3390/s22072712 Accessed: Oct. 7, 2025. [Online]. Available: https://www.mdpi.com/1424-8220/22/7/2712

[23] R. Kakavand, R. Ahmadi, A. Parsaei, W. B. Edwards, and A. Komeili. "Comparison of Kinematics and Kinetics Between OpenCap and a Marker-Based Motion Capture System in Cycling." arXiv: 2409.03766 [cs], Accessed: Apr. 4, 2026. [Online]. Available: http://arxiv.org/abs/2409.03766 pre-published.

[24] T. H. Nasution, A. Sitohang, .-. Seniman, and .-. Soeharwinto, "Bike Fitting System Based on Digital Image Processing on Road Bike," *JOIV : International Journal on Informatics Visualization*, vol. 9, no. 2, pp. 699–707, Mar. 31, 2025, ISSN: 2549-9904. DOI: 10.62527/joiv.9.2.2796 Accessed: Oct. 7, 2025. [Online]. Available: http://joiv.org/index.php/joiv/article/view/2796

[25] Y. Zhang and N. Zhao, "Intelligent Bike Fitting System Design Based on OpenPose," in *Proceedings of the 2024 4th International Conference on Big Data, Artificial Intelligence and Risk Management*, ser. ICBAR '24, New York, NY, USA: Association for Computing Machinery, Apr. 27, 2025, pp. 638–643, ISBN: 979-8-4007-0975-3. DOI: 10.1145/3718751.3718853 Accessed: Nov. 23, 2025. [Online]. Available: https://dl.acm.org/doi/10.1145/3718751.3718853

[26] M. A. Ummu Habeeba, P. Jayashree, M. Livitha Poornima, D. Saraswathi, G. Muralimohan, and S. Edward Jero, "FusionPose: A MediaPipe and keypoint R-CNN fused model for enhanced cyclist pose estimation and injury risk prediction," *Advanced Engineering Informatics*, vol. 69, p. 104 014, Jan. 1, 2026, ISSN: 1474-0346. DOI: 10.1016/j.aei.2025.104014 Accessed: Oct. 7, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474034625009073

[27] "OpenPose: OpenPose Doc - Demo," Accessed: May 3, 2026. [Online]. Available: https://cmu-perceptual-computing-lab.github.io/openpose/web/html/doc/md_doc_01_demo.html#maximum-accuracy-configuration

List of Tables

2.1 Comparison of related systems. 8

5.1 Per-class detection results. 25
5.2 Frame counts before and after the side-angle selection across uncontrolled videos. 27
5.3 Threshold selection results for `QUALITY_FRACTION` and `SQUARE_TOL`. . . 29
5.4 OpenPose keypoint confidence and proportion of frames above `CONF_MIN` by filming condition. 30
5.5 Savitzky-Golay window and peak prominence floor test results. 30
5.6 Seat-height verdict confusion matrix for 19 uncontrolled videos against controlled reference. 34
5.7 Seat-height verdict agreement by percentile. 35
5.8 Uncontrolled videos with no RPM output. 35

List of Figures

2.1 Hip-knee-ankle angle described in [9]. 4
2.2 OpenPose Output Formats. 5
2.3 Early YOLO detection model [15]. 6

3.1 System diagram with each step of the designed system and their inputs and outputs. 12

4.1 Intermediate JSON File Schemas. 14
4.2 Qualitative results from COCO validation batch. 15
4.3 YOLO training curves over 50 epochs. 16
4.4 Preprocessing applied to selected frames before OpenPose inference. . . 18

5.1 Filming conditions. 25
5.2 Normalised confusion matrix. 26
5.3 Selected frames versus total frames for each uncontrolled video. 27
5.4 Accepted and rejected frames from the side-angle selection stage. 28
5.5 OpenPose keypoints on P07's uncontrolled frames ranked by mean lower-limb joint confidence. 31
5.6 Raw and smoothed knee angle series for three participants across the three filming locations. 32
5.7 Predicted versus true RPM for uncontrolled videos. 33