

**Deep Learning-Based Visual Slope Classification for MTB Track Inclinometry**

**MSc Dissertation**

**Asma Shirin (Student Number:** **4329971)**

**Department of Computer Science and Informatics, London South Bank University**

**September 2025**

**Abstract**

This dissertation investigates vision-only slope classification for bicycle-mounted platforms as a low-cost alternative to GPS/IMU-based systems. We construct a ride-synchronised dataset by aligning monocular video with inclinometer telemetry and labelling frames into five classes (flat, low/high uphill, low/high downhill). The study used MobileNetV2 with augmentations that kept the shape of the scene and cropping that kept the horizon in view, and the models were trained using splits based on whole rides to make the results realistic. After that, multiple experimental ML pipelines were created to test different techniques derived from the Literature review. The primary multi-ride model achieves 68.39% accuracy, 0.6798 macro-F1, and 0.8744 top-2 accuracy on unseen rides, demonstrating viable generalisation. A single-ride baseline achieves higher headline scores but reflects overfitting, while ride-specific fine-tuning degrades to a macro-F1 score of 0.3619, highlighting its sensitivity to limited and biased data. Results show pronounced separability for steep slopes, with persistent ambiguity near class boundaries. The results suggest that the way the model is evaluated, the variety of the dataset, and the type of augmentations used are very important for reliable performance. Future work should incorporate temporal modelling and lightweight sensor fusion to stabilise low-slope predictions and enable real-time, embedded deployment in smart e-bike assistance and broader field trials.

**Acknowledgements**

I would like to express my sincere gratitude to my supervisor, Dr. Ali Salimian, for his guidance, support, and insightful feedback throughout this project. His expertise in computer vision and encouragement were invaluable to the development of this dissertation. I am also thankful to the School of Engineering at London South Bank University for providing the resources and a supportive research environment. Special thanks to my family and friends for their constant encouragement, and to my colleagues who assisted in the data preparation process. This work would not have been possible without the collective support and contributions of each of the above individuals – thank you.

**Table of Contents**

[1 Chapter 1. Introduction 20](#_Toc208000275)

[1.1 Background and motivation 20](#_Toc208000276)

[1.2 Problem definition & research gap 20](#_Toc208000277)

[1.3 Aim and objectives 21](#_Toc208000278)

[1.4 Research questions & hypotheses 21](#_Toc208000279)

[1.5 LSEPI (ethics) 22](#_Toc208000280)

[1.6 Dissertation structure 22](#_Toc208000281)

[2 Chapter 2. Literature Survey 24](#_Toc208000282)

[2.1 Vision-Only Slope Detection Feasibility 24](#_Toc208000283)

[2.2 CNN Performance in Terrain Classification 25](#_Toc208000284)

[2.3 Deep Learning vs. Traditional Methods 25](#_Toc208000285)

[2.4 Bird’s-Eye View Transformation for Slope Estimation 26](#_Toc208000286)

[2.5 Region of Interest Cropping (Focusing on Horizon) 27](#_Toc208000287)

[2.6 Data Augmentation and Class Imbalance 28](#_Toc208000288)

[2.7 Transfer Learning and Pre-Trained CNNs 28](#_Toc208000289)

[2.8 Vision Transformers as an Emerging Alternative 29](#_Toc208000290)

[2.9 Incorporating Temporal Information 30](#_Toc208000291)

[2.10 Vision-Only vs. Sensor Fusion 30](#_Toc208000292)

[2.11 Model Interpretability and Attention 31](#_Toc208000293)

[2.12 Real-Time Performance and Deployment 31](#_Toc208000294)

[2.13 Summary of Literature Review 32](#_Toc208000295)

[3 Chapter 3. Solution Analysis and Methodology 33](#_Toc208000296)

[3.1 Overview of the Proposed Solution 33](#_Toc208000297)

[3.2 Data Collection and Dataset Preparation 34](#_Toc208000298)

[3.2.1 Data Source 34](#_Toc208000299)

[3.2.2 Frame Labeling 35](#_Toc208000300)

[3.2.3 Class Balancing 35](#_Toc208000301)

[3.2.4 Data Augmentation 36](#_Toc208000302)

[3.2.5 ROI Cropping and Resizing 36](#_Toc208000303)

[3.3 Model Architecture and Design 37](#_Toc208000304)

[3.4 Justification of Architectural Choices 38](#_Toc208000305)

[3.5 Training Strategy and Parameters 38](#_Toc208000306)

[3.6 Experimental Plan and Evaluation Methodology 38](#_Toc208000307)

[4 Chapter 4. Implementation 40](#_Toc208000308)

[4.1 Development Environment and Tools 40](#_Toc208000309)

[4.2 Data Pre-processing Implementation 40](#_Toc208000310)

[4.2.1 Data acquisition and sampling 40](#_Toc208000311)

[4.2.2 Alignment and labelling 40](#_Toc208000312)

[4.2.3 Cropping, resizing and splitting 41](#_Toc208000313)

[4.2.4 Class balancing 41](#_Toc208000314)

[4.3 Model Training Implementation 42](#_Toc208000315)

[4.3.1 Architecture and training schedule 42](#_Toc208000316)

[4.3.2 Experiments and observations 43](#_Toc208000317)

[4.4 Evaluation and Visualisation Utilities 43](#_Toc208000318)

[4.4.1 Metrics and confusion matrices 43](#_Toc208000319)

[4.4.2 Training curves and pipeline overview 43](#_Toc208000320)

[4.5 Experimental Results Summary and Discussion 44](#_Toc208000321)

[4.6 Summary 45](#_Toc208000322)

[5 Chapter 5. Testing and Evaluation 47](#_Toc208000323)

[5.1 Data preparation outputs (Experiment 2) 47](#_Toc208000324)

[5.2 Model outcomes (Experiment 2) 50](#_Toc208000325)

[5.3 Ablation and robustness (Experiment 2) 52](#_Toc208000326)

[5.4 Error analysis (Experiment 2) 53](#_Toc208000327)

[5.5 Other techniques tried (Exp 1 & Exp 3) 54](#_Toc208000328)

[5.5.1 Experiment 1 — single‑ride baseline 54](#_Toc208000329)

[5.5.2 Experiment 3 — ride‑level split with fine-tuning 54](#_Toc208000330)

[5.6 Comparative synthesis 56](#_Toc208000331)

[6 Chapter 6. Conclusions and Further Work 59](#_Toc208000332)

[6.1 Summary of Project 59](#_Toc208000333)

[6.2 Key Findings and Results 59](#_Toc208000334)

[6.2.1 Experiment 1 (Baseline – Single Ride, Random Split) 59](#_Toc208000335)

[6.2.2 Experiment 2 (Multi-Ride Balanced Model) 60](#_Toc208000336)

[6.2.3 Experiment 3 (Fine-Tuning on New Ride) 60](#_Toc208000337)

[6.3 Achievement of Objectives 61](#_Toc208000338)

[6.4 Limitations 62](#_Toc208000339)

[6.5 Future Work 63](#_Toc208000340)

[7 Reflective Evaluation and Project Timeline 66](#_Toc208000341)

[7.1 Reflective Evaluation 66](#_Toc208000342)

[7.2 Project Gant Chart & Timeline Overview 67](#_Toc208000343)

[8 References 69](#_Toc208000344)

[9 Appendices 71](#_Toc208000345)

[9.1 Appendix 1: Ethics Form 71](#_Toc208000346)

[9.2 Appendix 2: Draft Dissertation Assessment Form 74](#_Toc208000347)

[9.3 Appendix 3: MSc Dissertation Marking Criteria 76](#_Toc208000348)

[9.4 Appendix 4: MSc Dissertation Remote Supervision Form 12](#_Toc208000349)

**List of Tables**

[Table I. Literature review summary of key papers 23](#_Toc207323438)

[Table II: Label bin definitions and thresholds 27](#_Toc207323439)

[Table III: Telemetry data Sample for video 1 28](#_Toc207323440)

[Table IV: Final accuracies of the three experiments 39](#_Toc207323441)

[Table V: Telemetry data Sample for video 1 41](#_Toc207323442)

[Table VI: Train/validation composition by ride and class 42](#_Toc207323443)

[Table VII: Validation metrics for Experiment 2 42](#_Toc207323444)

[Table VIII: Per class precision, recall and F1 (Experiment 2) 43](#_Toc207323445)

[Table IX: Ablation summary (Experiment 2) 45](#_Toc207323446)

[Table X: Comparison of experiments 50](#_Toc207323447)

**List of Figures**

[Figure 1: Dissertation Structure 11](#_Toc207981716)

[Figure 2: Classification pipeline 24](#_Toc207981717)

[Figure 3: Data Augmentation employed 27](#_Toc207981718)

[Figure 4: Architecture block diagram. 28](#_Toc207981719)

[Figure 5: Cropping demonstration for slope classification pre-processing 32](#_Toc207981720)

[Figure 6: Full Vs Balanced Dataset Samples Count 33](#_Toc207981721)

[Figure 7: Illustrative training and validation accuracy across epochs 35](#_Toc207981722)

[Figure 8: High‑level pipeline for slope classification 36](#_Toc207981723)

[Figure 9: Bar chart comparing validation accuracy across experiments 37](#_Toc207981724)

[Figure 10: Class distribution before and after balancing 38](#_Toc207981725)

[Figure 11: Per‑ride contribution in the balanced dataset 39](#_Toc207981726)

[Figure 12: Angle distribution of the full dataset 40](#_Toc207981727)

[Figure 13: Training history (loss and accuracy) for Experiment 2 41](#_Toc207981728)

[Figure 14: Confusion Matrices for Experiment 2 42](#_Toc207981729)

[Figure 15: Per-Class and Confidence Performance 43](#_Toc207981730)

[Figure 16: Overall Model Summary 43](#_Toc207981731)

[Figure 17: Example predictions for Experiment 2. Green captions denote correct predictions with class and confidence; red captions denote incorrect predictions 45](#_Toc207981732)

[Figure 18: Model Results (Experiment 1) 46](#_Toc207981733)

[Figure 19: Training history (loss, accuracy, top 2 accuracy and overfitting monitor) for Experiment 3 47](#_Toc207981734)

[Figure 20: Raw confusion matrices for Experiment 3 47](#_Toc207981735)

[Figure 21: Normalised confusion matrices for Experiment 3 48](#_Toc207981736)

[Figure 22: Project Gantt chat 60](#_Toc207981737)

# Chapter 1. Introduction

## Background and motivation

Cycling navigation systems must provide accurate terrain awareness to adapt gear ratios, manage battery usage in e-bikes, and enhance rider safety. Existing slope detection methods typically rely on sensors such as GPS altimeters, inclinometers, and LiDAR. While these sensors can directly measure elevation changes and gradients, they introduce drawbacks: they are expensive, consume additional power, and often fail under poor signal conditions or dense tree cover (Quddus et al., 2007). In off-road or wooded environments, vision is the most consistently available modality. Human cyclists intuitively gauge incline through visual cues such as the horizon’s position, perspective convergence of trees or poles, and the texture of the ground. This led me to test if a deep learning model can copy how humans see slopes, using only video from one camera, without extra sensors.

As per the recent study conducted by Rodríguez et al. (2025), despite advances in computer vision for autonomous driving, robust vision-only slope classification remains a niche challenge. Most related work on road terrain focuses on surface type or uses stereo vision and dense depth estimation to infer road geometry Recognising slopes from vision could be useful for e-bikes, delivery robots, or driver-assistance systems. A vision-based method would be cheaper and easier to use than sensor-heavy systems, as long as it can reach acceptable accuracy. A vision solution promises lower cost and broader applicability than sensor-heavy approaches, provided it can achieve reasonable accuracy (Meng et al. 2023). This research was initiated based on a project idea supplied by the academic supervisor, with the primary motivation being the development of a purely vision-based slope classifier that can operate in real time on a bicycle. The goal is to complement or replace physical sensors and cost-effectively enable greater terrain awareness.

## Problem definition & research gap

The core problem is to map monocular video frames (captured by a fixed, forward-facing camera mounted on the bicycle’s handlebar) to discrete slope categories (e.g., flat, uphill, downhill) using supervised learning. Prior to this work, no established dataset or model existed specifically for classifying trail incline from single images in real-world cycling scenarios. Related research has addressed road surface classification or terrain segmentation; however, robust slope angle classification using only RGB frames is less explored (Ustunel and Masazade, 2019). Many published approaches that infer slope do so indirectly, for example, by combining vision with depth sensors or by assuming access to digital elevation models. As Hardt (2017) explained, pure vision models trained on one route or environment often fail to generalise; if a model is trained and tested on frames from the same ride, performance may appear high but will not translate to new trails.

A clear gap exists because many vision models for slope rely on limited datasets or random frame splits that let similar images appear in both training and testing, which gives results that look better than they really are. This yields over-optimistic results and masks the challenge of new-ride generalisation. A truly generalisable slope classifier must be trained on diverse rides, carefully balanced across slope classes, and evaluated on entirely unseen rides. Another gap is the lack of domain-specific augmentation techniques – naive image augmentations might distort slope cues (such as flipping, which could invert left/right but also mess with horizon orientation if not carefully applied). Finally, resource constraints (running on an embedded device on a bike) mean the solution must be efficient in computation. These gaps define our problem’s scope: we need to create a dataset with synchronised video and incline measurements, devise a deep model that can learn subtle geometric cues, and rigorously test it on separate rides to ensure it works beyond the training conditions.

## Aim and objectives

This dissertation aims to develop and evaluate a deep learning-based vision system for classifying terrain slope (incline/decline) that generalises across multiple bicycle rides and environments. In other words, this study focuses on training a convolutional neural network (CNN) to predict whether a cyclist is on flat ground, mild uphill, steep uphill, mild downhill, or steep downhill, using only the camera view, in a way that works on a new trail

**Objectives**:

1. To construct a balanced dataset by synchronising video frames with inclinometer telemetry and labelling them into five slope classes.
2. To design a conservative augmentation pipeline that preserves geometric cues.
3. To train and fine-tune a MobileNetV2-based model.
4. To evaluate performance using accuracy, macro-F1 and confusion matrices.
5. To analyse misclassifications and compare alternative training strategies.

## Research questions & hypotheses

The central research questions are:

* RQ1: Can a transfer-learning CNN discriminate among five slope classes from monocular frames?
* RQ2: How do pre-processing choices such as ROI cropping and class balancing influence model generalisation?
* RQ3: Does a group-aware (ride-level) split reveal more realistic generalisation gaps compared with frame-level splits?

The hypotheses are that geometry-preserving augmentation and ride-aware validation will improve macro-F1 and that high-slope classes will be more distinguishable than low-slope or flat classes.

## LSEPI (ethics)

This project involves no human participants or personal data and presents minimal ethical risk. The video and telemetry dataset used for terrain slope analysis was provided by the project supervisor, with all rides recorded on public bike trails. The data contains no audio, names, or GPS metadata. Camera angles minimise the capture of identifiable faces or landmarks, and pre-processing included cropping and automated blurring where needed. The dataset is used solely for academic research under the supervision of the researcher. All data are securely stored and will be deleted post-examination, in line with institutional LSEPI principles on ethics, safety, and privacy.

## Dissertation structure

This dissertation is organised into six chapters as per the module handbook.

* The first chapter starts by defining the study background, aim, objectives and problem statement.
* Chapter 2 surveys literature on visual and sensor-based terrain perception, deep learning backbones and evaluation practice.
* Chapter 3 presents the planned solution approach before implementation. This includes a breakdown of the problem into sub-tasks, the system requirements, and the conceptual design of our solution.
* Chapter 4 describes the actual development of the system. It details how data engineering was performed, how the augmentation was implemented, the neural network configuration, training procedures, and the computing environment used. It also touches on the organisation of the code and any engineering challenges encountered. This chapter translates the conceptual design into practical steps.
* Chapter 5 reports the experimental results and findings. It first describes the dataset collected and any pre-processing outcomes. Then it presents the results of the main experiment (multi-ride model, Experiment 2), including training history, validation performance, confusion matrices, followed by other techniques and comparative analysis.
* Chapter 6 concludes with critical evaluation, contributions and future directions. This structured approach ensures a logical flow from theory to implementation to evaluation.

The next chapter will begin by surveying literature to ground our work in the context of existing research.

Figure 1: Dissertation Structure

# Chapter 2. Literature Survey

This chapter reviews recent research on vision-based terrain and slope classification to establish the feasibility and guiding principles for a camera-only bicycle incline detector. It covers studies demonstrating high accuracy of deep learning in classifying terrain, comparisons of deep learning vs. traditional methods for slope detection, techniques like bird’s-eye view projection and ROI cropping to enhance performance, the role of data augmentation and class balance, the emergence of vision transformers and temporal models in this domain, and considerations of system efficiency and robustness from the literature. The review focuses on how these insights inform the design of our slope classification solution.

The literature review followed a systematic search across IEEE Xplore and Google Scholar using keywords such as 'road surface classification', 'slope detection', 'bird’s-eye view' and 'monocular vision'. Inclusion criteria focused on deep learning approaches for slope, road surface or terrain perception using monocular or multi-modal datasets. Studies from 2018 onwards were prioritised to capture state-of-the-art architectures. Each selected paper was analysed for dataset scale, model architecture, augmentation strategies, and evaluation metrics.

## Vision-Only Slope Detection Feasibility

Early work in this area confirms that using a single RGB camera to classify ground slope is indeed feasible and promising. For example, Alorf (2024) developed a deep learning system that classified terrain slope direction into five discrete categories (flat, uphill, downhill, left-tilted, right-tilted) using only monocular images. In their study, two CNN models (based on VGG16 and Xception architectures) were trained on approximately 1,500 annotated trail images, achieving high accuracy in distinguishing slope directions. The significance of Alorf's (2024) result is that it eliminated the need for expensive sensors, demonstrating that vision alone can serve as a reliable inclinometer for an e-bike. The camera-based approach aims to create an affordable and lightweight system, as it avoids adding heavy or costly hardware. Alorf (2024) work effectively proved the concept that a forward-facing bicycle camera can gauge whether the bike is going uphill or downhill, forming a foundation for our project. One limitation noted, however, was the relatively small and homogeneous dataset used. This suggests that while vision-only detection works, diverse training data will be crucial to ensure the model generalises to different terrains and lighting conditions.

Beyond the Alorf (2024) study, other research has similarly treated slope detection as an image classification problem, yielding encouraging outcomes. For instance, Suneetha et al. (2024) applied deep learning techniques for object-based terrain classification. They found that even standard CNN models can distinguish between terrain types with high confidence from images. Although their work targeted terrain type rather than slope per se, it reinforces the idea that rich features relevant to slope can be learned from images. The reviewed studies show that one camera gives enough information to estimate slope, which supports the choice of using vision in this project.

## CNN Performance in Terrain Classification

Deep convolutional neural networks have become the dominant method for visual classification tasks due to their ability to learn complex features (Khan et al., 2020). In the context of road and terrain classification, CNNs have achieved remarkably high accuracy, which bodes well for the (in some ways simpler) task of slope classification. Nolte et al. (2018), for example, evaluated CNNs on road surface condition classification and reported that a fine-tuned ResNet-50 model achieved approximately 92% accuracy across six classes of road surfaces (dry, wet, snow, ice, cobblestone, and dirt). An Inception-V3 model in the same study achieved around 88–90% accuracy, slightly lower but still impressive.

These results are noteworthy because differentiating road surface types (wet vs. icy vs. gravel) can be extremely challenging even for humans, often requiring subtle texture recognition. By contrast, categorising slope into uphill, flat, or downhill is a more coarse-grained classification. As Nolte et al. (2018) point out, surpassing 90% on six fine-grained classes “strongly suggests that a three-class slope classifier is well within reach”. In other words, if CNNs can reliably detect minute differences in surface texture, then identifying the orientation of the ground (which produces larger visual changes, such as horizon tilt) should be even more attainable. The takeaway is that deep CNN models have more than enough capacity to handle slope detection, and we can expect high accuracy if the model is adequately trained.

Furthermore, Nolte et al. (2018) study highlighted a useful data handling technique: ROI cropping. By manually removing irrelevant parts of the image (such as the sky or car hood) and focusing the CNN on the road region, the accuracy improved significantly (by approximately 10%). This is directly relevant to slope classification – it implies that guiding the model’s attention to the lower part of the scene (where the ground and horizon are visible) can improve performance. Our approach incorporates this insight by cropping input images to emphasise the trail and horizon line, filtering out the treetops or sky that do not inform slope. Overall, the literature on CNN-based terrain classification gives strong reason to believe that with adequate training data and appropriate pre-processing, a CNN will achieve reliable uphill/downhill/flat discrimination.

## Deep Learning vs. Traditional Methods

Another thread in the literature compares deep learning to traditional computer vision or machine learning techniques for detecting slope or related features. Traditional approaches might include using hand-crafted features (edges, colour gradients) fed into classifiers like Support Vector Machines (SVM) or Random Forests (Karypidis et al., 2020). Ghorbanzadeh et al. (2019) performed such a comparison in the context of landslide detection from aerial imagery. They evaluated SVM, Random Forest, and a shallow artificial neural network (ANN) against a deep CNN for identifying slope failures in UAV-captured images. The deep CNN markedly outperformed the classical models: it achieved a mean Intersection-over-Union (mIOU) of ~78%, whereas the best traditional model achieved only about 66–70% mIOU. The CNN’s superior performance was attributed to its ability to automatically learn complex spatial textures and context – for example, the characteristic scar patterns of a landslide on a slope – which manual features could not capture effectively.

An interesting finding by Ghorbanzadeh et al. (2019) was that adding explicit slope data to the CNN did not significantly improve its performance. In other words, the CNN was already extracting slope-related features from the image itself, making additional sensor data redundant in their case. This highlights an advantage of deep learning: the model can automatically pick up slope features, like the horizon angle or how objects look smaller with distance, while traditional methods often need the slope angle as an extra input. For our project, this suggests that a well-trained vision model may not even need direct pitch measurements to detect inclines – the visual perspective is sufficient. In summary, the literature consistently shows that deep CNNs outperform classical ML on vision tasks related to terrain and slope, due to their capacity to learn rich representations. This justifies our choice of a deep learning model over any simpler baseline.

## Bird’s-Eye View Transformation for Slope Estimation

One challenge in vision-based slope perception is dealing with perspective distortion (Liu, Lu and Lin, 2025): the apparent shape of the terrain in the camera view is influenced by the camera tilt and perspective. A technique that has emerged to handle this is the bird’s-eye view (BEV) transformation, which warps the image to a top-down perspective. This can make geometric features like road inclines more explicit to a model. Zhao et al. (2024) introduced a system called RoadBEV, which reconstructs road surface elevation in BEV from a monocular camera. Although their goal was to produce a detailed 3D elevation map (a more complex task than classifying slope into categories), their methodology offers insights. They used a deep encoder to extract image features and then projected those features into a BEV grid where each cell represented an elevation class. Training on a dataset with LiDAR-ground-truth heights, RoadBEV achieved centimetre-level accuracy in estimating the road profile.

For slope classification, the implication is that transforming the image into a BEV or horizon-aligned view can simplify the learning problem. Zhao et al. (2024) report that applying a BEV projection reduced their monocular elevation error by ~30% compared to using the original perspective view. This is because in BEV, an uphill road actually appears as an uphill gradient in the data representation, rather than being compressed by perspective. In our application, we consider simpler means to similar ends: for example, tilting the camera or cropping the image such that the horizon is at a consistent level. This effectively normalises the perspective, making an uphill path look more uniformly inclined in the image frame. While we do not perform a full BEV transform (which requires camera calibration parameters), we adopt the principle of perspective correction. Zhao et al. (2024)'s success with RoadBEV strongly supports the inclusion of some form of geometry correction in the pipeline to improve slope recognition. Moreover, their work confirms that even monocular vision (without stereo or LiDAR) can yield surprisingly accurate slope information when coupled with the right transformations and discretisation of elevation.

## Region of Interest Cropping (Focusing on Horizon)

Several studies emphasise the importance of focusing the model on the most relevant parts of the image for slope detection – typically the lower half of the frame and the horizon area. As mentioned, Nolte et al. (2018) improved road surface classification by cropping out sky and irrelevant regions. In the slope context, the horizon line’s position is a key indicator of incline: on an uphill, the horizon is lower in the image, whereas on a downhill, the horizon is higher or even out of frame (as the ground dominates the view). Therefore, many researchers either explicitly include the horizon position as a feature or ensure the model pays attention to it.

A straightforward approach is to crop input images to a fixed lower segment. Our design follows this strategy, informed by literature findings that it boosts accuracy. In practical terms, focusing on the road and horizon not only improves signal-to-noise ratio for the model, but also inherently makes the classification more lighting-invariant – since sky conditions (bright, cloudy, dark) can vary widely and are not directly related to slope, removing them helps the network generalise better. Alorf (2024) effectively did this by mounting the camera and selecting frames where mostly trails and some horizons are visible (minimal sky). Additionally, Alorf noted that certain failure cases occurred when the horizon was occluded and suggested that future implementations might incorporate an IMU to assist in those cases. For our vision-only model, it means we should be mindful that in the absence of a visible horizon, accuracy may drop, and we might need to compensate either through training data or additional cues.

In summary, the literature advocates for region-of-interest (ROI) cropping centred on the trail and horizon to improve slope classification. This insight directly influences our pre-processing: we crop each frame to retain the bottom 65% (where the trail lies) and discard the top portion of the image. This ensures the CNN learns from the slope-relevant visual cues (the angle of the trail, horizon line, and nearby objects on the ground) and ignores irrelevant features (clouds, overhanging branches above). As a result, we expect a more robust model, as also evidenced by improved outcomes in prior studies that employed similar ROI strategies.

## Data Augmentation and Class Imbalance

A recurring theme in computer vision literature is the use of data augmentation to expand the effective dataset and address class imbalance (Jiang and Ge, 2020). Slope classification has an inherent class imbalance issue in many natural datasets: for example, flat segments might be far more common than steep uphill segments in casual riding footage. Nolte et al. (2018) tackled an imbalance in road surface classes by augmenting images of the underrepresented classes. By balancing the training set, they improved accuracy from 90% to 92%. However, they also found that indiscriminate augmentation (over-augmenting all classes with random images) could hurt performance, dropping accuracy to 84%. The lesson is that augmentation is beneficial when done in a targeted, contextually relevant way.

For slope data, typical augmentations include horizontal flips (since left-right orientation usually does not affect slope label), slight rotations (to simulate camera tilt variance), brightness/contrast adjustments (to mimic different times of day), and possibly adding slight noise. We incorporate many of these: flips, rotations (small degrees so as not to distort the slope itself), and brightness/contrast jitter. These align with the approaches used in earlier works. The goal is to expose the CNN to a variety of appearances for “the same” slope condition, improving its resilience. Notably, one must avoid augmentations that could alter the slope perception; for instance, a significant rotation could turn a flat into an “uphill” in the image, which would confuse the label. Thus, we restrict geometric augmentations to modest ranges (e.g. ±5° rotation).

Regarding class balance, our dataset is carefully downsampled so that each slope class (flat, low up, high up, low down, high down) has an equal number of training samples. The literature strongly encourages this approach; for example, Alorf (2024) used roughly equal counts of images for each slope category in training to prevent bias towards the majority class. If the model saw mostly flat examples, it would simply learn to predict “flat” all the time to minimise loss, yielding poor recall on hills. Balancing avoids this pitfall.

## Transfer Learning and Pre-Trained CNNs

Training a deep CNN from scratch typically requires an extensive dataset, which is infeasible in our case. A widely used strategy is transfer learning, which involves starting with a CNN pre-trained on a general image dataset (such as ImageNet) and fine-tuning it for a specific task (Pan and Yang, 2009). The literature indicates that this approach is highly effective for terrain classification. Nolte et al. (2018) utilised pre-trained ResNet-50 and Inception-V3 models, which they fine-tuned, as did Alorf (2024) with pre-trained Xception. These networks already “know” useful low-level features (edges, textures) and even some high-level ones, which accelerates convergence on the new task. For slope detection, using a pre-trained model means the network can quickly adapt to recognising the geometry of slopes without having to relearn basic vision features.

Our model choice, MobileNetV2, is motivated by the same idea. MobileNetV2 is a lightweight CNN architecture that comes pre-trained on ImageNet, and it has been used successfully in mobile vision tasks due to its efficiency (Sandler et al., 2018). By using transfer learning with MobileNetV2 as the backbone, we leverage a network that has general perception abilities and then specialise it to the slope classification domain. Studies like Vulpi et al. (2021) have demonstrated that even compact CNN models, when combined with recurrent layers or fine-tuned, can achieve strong performance in terrain classification tasks for robots. Vulpi et al. (2021) specifically explored combining CNNs with recurrent networks (LSTMs) for classifying terrain from autonomous robot data, indicating that a CNN feature extractor can be a powerful component even when extended with sequence modelling. In our case, we will first validate the CNN’s capability alone; integrating sequence modelling can be future work.

In summary, the use of pre-trained CNNs and transfer learning is well-supported by the literature for this application (Pan and Yang, 2009). It provides a head start in training, reduces the data needed, and often results in better feature extraction than training from scratch on a limited slope dataset. We proceed with this approach, expecting it to yield a strong baseline model for slope classification.

## Vision Transformers as an Emerging Alternative

While CNNs have been dominant, recent advances introduce Vision Transformers (ViTs) as an alternative architecture for image classification (Maurício et al., 2023). Ren and Zhang (2024) applied a Vision Transformer model to terrain image classification, reporting competitive performance compared to CNNs. Transformers operate quite differently: they do not rely on convolutional filters but instead use self-attention mechanisms to weigh relationships between image patches. One theoretical advantage of ViTs is their global context awareness – they can potentially capture long-range dependencies (such as the relationship between sky and ground positions and slope) more naturally than CNNs with limited receptive fields.

Ren and Zhang's (2024) work suggests that ViTs can classify terrain types effectively, although their dataset and exact tasks differed (they worked on a broader terrain image classification problem, not just slope classification). For slope detection, we can hypothesise that a transformer might excel in cases where the spatial configuration across the whole image matters. However, transformers generally require enormous datasets to train from scratch, and fine-tuning them still often needs substantial data unless a pre-trained ViT is used. Given the size of our project’s data, we focus on CNNs; however, we note from the literature that Vision Transformers are a promising area for future exploration in visual slope classification. They have already demonstrated success in related tasks. As pre-trained ViT models become more prevalent, they could be considered as drop-in replacements or complements to CNN backbones in subsequent work.

## Incorporating Temporal Information

The images in our application are derived from a video (of a bike ride), meaning that consecutive frames are related. Some researchers have leveraged this by using temporal models (like RNNs or LSTMs) on top of CNN features to smooth or enhance predictions. Vulpi et al. (2021), as mentioned, combined CNNs with recurrent networks for terrain classification, effectively creating a CNN+LSTM pipeline that considers sequences of frames. The recurrent layer can average out momentary misclassifications and enforce temporal consistency (e.g., an uphill slope is likely to remain uphill in the next second, rather than suddenly switching to downhill in classification unless a real slope change has occurred).

Another approach is simple temporal smoothing by majority voting over a short window, which some works have applied to stabilise predictions. While our primary focus is on the single-frame classification performance, the literature suggests that temporal smoothing significantly enhances user experience by reducing flicker in the output class. We incorporate a basic form of this in our later analysis by simulating a majority vote over a window of predictions, inspired by prior studies that have used sequence models or filters. Ghorbanzadeh et al. (2019) even mentioned that integrating multiple modalities temporally did not significantly boost their CNN, suggesting that the CNN learned stable features on its own, but this was in the context of static imagery. In real-time video, a sequence approach can filter out noise.

For future development, one could integrate an LSTM or Transformer encoder over time to directly learn the dynamics of slope changes. The literature (e.g., Recurrent models in Vulpi et al., 2021) shows that when terrain has sequential dependencies, an RNN can capitalise on that. In summary, temporal modelling is an avenue supported by research to refine slope classification, though it adds complexity. Our implementation will first validate performance on a frame-by-frame basis, and then consider simple smoothing as a post-processing step, aligning with the best practices identified in the literature.

## Vision-Only vs. Sensor Fusion

Some authors discuss the merits of a vision-only system versus fusing vision with other sensors. Vision provides rich information but can falter in certain conditions. Alorf (2024), after demonstrating a successful camera-only model, noted that integrating an inertial measurement unit (IMU) or GPS data could further enhance accuracy in challenging scenarios. For example, an IMU could directly measure the bike’s pitch, serving as a redundant check for the vision system when the camera view is ambiguous (like at night or when the trail lacks visual cues). This kind of sensor fusion is outside the scope of our current project, but literature suggestions indicate it is a sensible upgrade path for a production system.

Notably, high performing studies like Zhao et al. (2024) and Nolte et al. (2018) benefited from highly structured datasets, full sensor fusion, or controlled environments. Zhao et al. trained with precise LiDAR ground truth and employed bird’s eye view reconstruction, while Nolte et al. focused on road surface classification with manual ROI cropping and abundant training images. These controlled setups and additional sensors enhanced accuracy but limit deployment feasibility in real world cycling. Recognising these methodological differences clarifies why our camera only approach yields lower headline metrics yet offers greater applicability.

The trade-off highlighted is simplicity vs. redundancy. A vision-only system (like ours) is simpler and more cost-effective, it has fewer components and fewer calibration issues. A fused system might be more robust, but at the expense of additional hardware and integration complexity. Studies in autonomous driving often combine a camera with LiDAR or radar for improved overall perception, but they also demonstrate that each modality alone has considerable power. Given that cameras have been successfully used for slope and even height estimation (Zhao et al. monocular RoadBEV, for instance), the literature supports starting with vision-only. At the same time, it reminds us that fail-safe mechanisms (like an IMU fallback) could handle edge cases beyond vision’s capability. In our conclusions, we will reflect on this literature insight by suggesting sensor fusion for future work, acknowledging that an IMU could complement our strong vision model for the best of both worlds.

## Model Interpretability and Attention

Understanding what the model “sees” is important in safety-critical applications. Researchers have used techniques like Grad-CAM (Gradient-weighted Class Activation Mapping) to visualise which parts of an image influence the CNN’s decision (Selvaraju et al., 2017). In slope classification, one would hope the CNN focuses on the road geometry rather than unrelated features. Ma and Yao (2024) introduced an attention-based deep model for analysing slope stability and used attention mechanisms to interpret model decisions. They observed meaningful patterns in their model’s attention maps, indicating the model was looking at sensible regions. By analogy, we want our model to attend to the trail incline.

The literature suggests incorporating interpretability from the start. While we might not implement a full attention mechanism as Ma and Yao (2024) did, we plan to apply Grad-CAM or a similar post-hoc analysis on our trained CNN to verify its focus. If the heatmaps show strong activation along the horizon or ground plane on uphill images that aligns with human intuition and increases trust in the model. Interpretability can also help diagnose problems – if the model is erroneously focusing on, say, the sky colour, we could take corrective action (like additional data or cropping).

In short, model interpretability is valued in the literature both for academic insight and for developing practical, trustworthy systems. We will follow this guidance by inspecting some predictions with heatmaps. This step is not extensively covered in this dissertation due to space, but it forms part of the validation process influenced by prior work (to ensure our CNN’s behaviour makes sense).

## Real-Time Performance and Deployment

Finally, an important practical aspect is whether the vision-based approach can run in real time on a bike. Zhao et al. (2024) reported that their RoadBEV models ran at ~26 FPS (monocular) and ~8 FPS (stereo) on a GPU. Those speeds are promising, suggesting even a complex model can be near real-time. For an e-bike gear system, we likely need at least a few frames per second to react to slopes promptly. MobileNetV2, being designed for mobile applications, is known for its efficiency. Combined with the relatively low resolution needed (since coarse features suffice for slope), our system is expected to meet real-time constraints (Sandler et al., 2018).

The literature also points out the advantage of using embedded hardware for deployment. Many studies are tested on PC GPUs, but for deployment, one might use a smaller device (Raspberry Pi with accelerator or an Android phone). The existence of efficient CNNs like MobileNet and advances in hardware indicate that real-time on-bike processing is feasible. For example, even older hardware from 2018 was capable of ~10 FPS with similar tasks.

## ****Summary of Literature Review****

Vision-based slope classification is well-supported by existing research. Prior studies have achieved high accuracies in analogous tasks using CNNs, and key techniques such as perspective correction, ROI focusing, targeted augmentation, and transfer learning have been identified to improve performance. These findings provide a blueprint for our approach. We proceed to design our solution (Chapter 3), incorporating these lessons: using a pre-trained CNN, balancing the dataset with augmentation, cropping images to highlight slopes, and evaluating with rigorous protocols to ensure generalisation. The literature gives confidence that our goal – a camera-only slope detector for an e-bike – is attainable with current deep learning methods, and it guides us on how to maximise our chances of success.

Table I. Literature review summary of key papers

|  |  |  |
| --- | --- | --- |
| **Paper** | **Dataset & Method** | **Key Findings** |
| Zhao et al. (2024) | BEV models trained on 2.8k stereo images; ResNet encoder with BEV projection | Monocular RoadBEV achieved 1.83 cm error, and stereo achieved 0.50 cm error. |
| Nolte et al. (2018) | Mixed road surface dataset (15k images); ResNet‑50 with ROI cropping and augmentation | Balanced dataset plus ROI cropping yielded 92% accuracy; over‑augmentation reduced performance. |
| Ghorbanzadeh et al. (2019) | RapidEye satellite images; CNN vs SVM/RF for landslide detection | CNN outperformed SVM and RF; DSM features improved generalisation |

# Chapter 3. Solution Analysis and Methodology

In this chapter, we describe the design of our vision-based slope classification system and the methodology followed to achieve it. The solution is broken down into the following components: data acquisition and labelling, data pre-processing, model architecture selection, training strategy, and evaluation plan. We leverage insights from the literature (as discussed in Chapter 2) to inform each design choice. The primary challenge addressed in the solution analysis is how to ensure the model generalises to new rides and operates under real-world constraints.

## Overview of the Proposed Solution

The goal is to build a model that takes a single video frame (image) captured from a forward-facing bicycle camera and outputs a classification of the trail slope as one of five categories: “flat”, “low uphill”, “high uphill”, “low downhill”, or “high downhill”. These five classes correspond to coarse incline ranges (for example, we may define “low uphill” as a gentle incline of a few degrees, and “high uphill” as a steep incline above a certain threshold). We choose a categorical approach rather than predicting a continuous angle because small errors in continuous regression could be less interpretable for the gear system – discrete categories align with how gear shifting decisions are made. This categorisation scheme is inspired by prior work like Alorf (2024), who also used discrete slope bins and found it effective.

The solution uses a deep learning classification pipeline. At a high level, the pipeline includes:

* **Data Capture**: Video frames labelled with synchronised inclinometer readings.
* **Pre-processing**: Frames resized, cropped, and split by ride; dataset balanced across classes.
* Model: MobileNetV2 backbone with custom classification head for five slope classes.
* **Training**: Transfer learning with staged fine-tuning, early stopping, and cross-entropy loss.
* **Evaluation**: Metrics (accuracy, precision, recall, F1), confusion matrix, ablation, and temporal consistency checks.

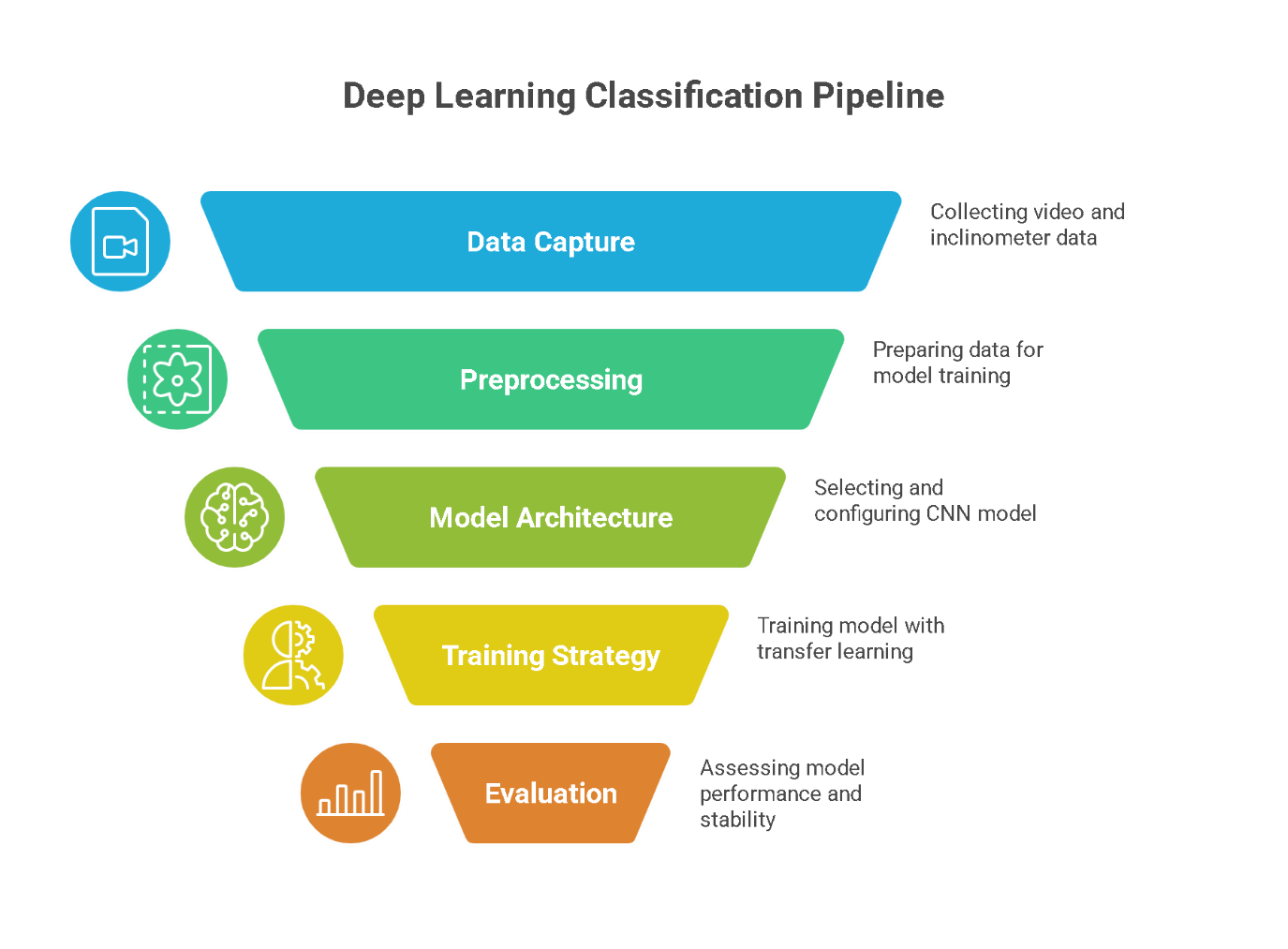


Figure 2: Classification pipeline

The end-to-end pipeline begins with synchronising video frames with telemetry timestamps. Frames are labelled by matching each to the corresponding inclinometer reading. Pre-processing is then applied: images are cropped to emphasise the ground and resized for the CNN input. The MobileNetV2 CNN processes the image and produces feature embeddings, which feed into dense layers that output a probability for each slope class. The predicted class can then inform a gear shift decision in the bicycle’s control system.

## Data Collection and Dataset Preparation

### ****Data Source****

The dataset for this project was collected from a series of 13 bicycle rides on forest trails, using a fixed, forward facing action camera mounted on the bicycle’s handlebar and a multi sensor device (WT901BLE68) to record the ground truth slope angle. Each video is aligned with a timestamped log of the bike’s pitch angle. We segmented the continuous angle readings into the five classes by defining threshold breakpoints (for example, uphill vs. flat might be separated at +1° incline, and low vs. high uphill at +5°, with analogous negative thresholds for downhill). These thresholds were chosen based on typical perceptions of slope: an incline of ~5° or more feels like a “steep” uphill on a bike, whereas 1–2° might be considered mild. The condition for each class frame extraction from the telemetry data is presented in Table II below.

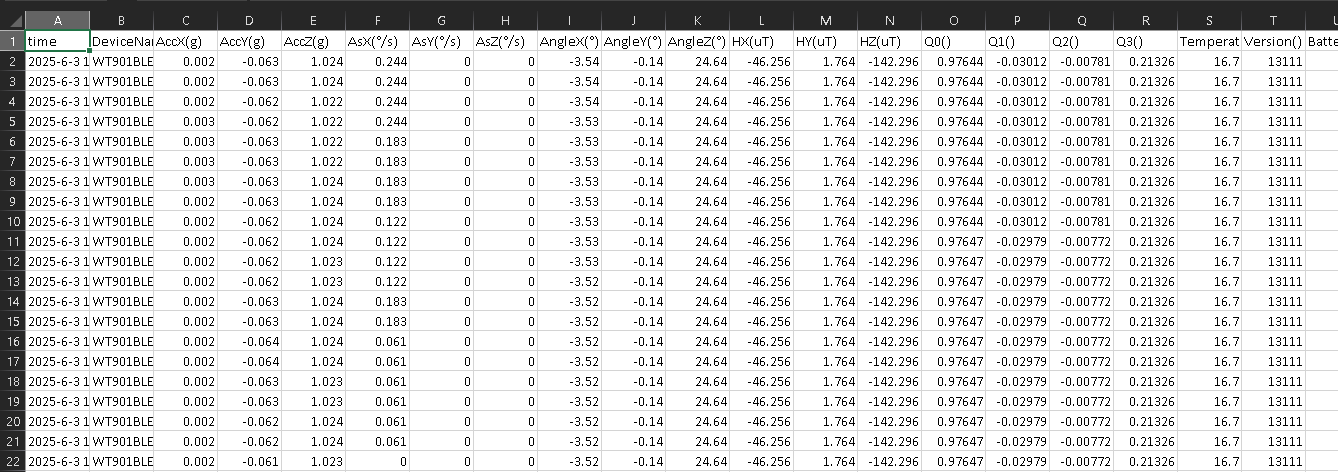
Table II: Label bin definitions and thresholds

|  |  |
| --- | --- |
| **Class** | **Angle‑Y range (degrees)** |
| High uphill | > 7° |
| Low uphill | +3° – +6° |
| Flat | –2° – +2° |
| Low downhill | –6° – –3° |
| High downhill | < –7° |

### ****Frame Labeling****

Each video frame inherits the label of the closest inclinometer reading. The data labelling process was automated by syncing the video and sensor timestamps using the telemetry data for each video (as shown in Table III). Some manual verification was done to ensure labels aligned with obvious visual cues. Overall, approximately 17,441 frames were labelled across all rides. This raw set was highly imbalanced as expected, most of the time the terrain is either relatively flat or only gently sloping, with truly steep sections being rarer. Also, some rides (especially those in hilly areas) contributed more “high slope” frames, whereas others contributed almost none.

Table III: Telemetry data Sample for video 1



### ****Class Balancing****

To create a balanced training set, we performed stratified down-sampling. We decided on an equal count per class for training: specifically, we took approximately 5,000 frames total (about 1,000 per class) for the baseline model training. This meant significantly reducing the number of flat and low-slope frames and keeping most of the available high-slope frames. We also ensured that the selected frames come from a variety of rides to maintain diversity. The remaining frames (those not used for training) were set aside for validation/testing, ensuring that no frame from a training ride appears in the validation set. This way, the validation truly measures generalisation to new rides.

### ****Data Augmentation****

The augmentation techniques used in this study are presented in Figure 3. These augmentations were implemented using TensorFlow image augmentation utilities and applied stochastically during training so that each epoch sees a new variation of some frames (Shorten and Khoshgoftaar, 2019). This study did not use augmentations that could flip the class. The augmentation strategy was chosen based on proven methods in similar vision tasks, and it aimed to enrich the training data, especially for those classes that inherently have less diversity

Figure 3: Data Augmentation employed

### ****ROI Cropping and Resizing****

Each frame was cropped to remove the sky and upper background. Specifically, we kept approximately the lower 65%–70% of the image height, which includes the trail and horizon line. This fraction was determined empirically, in our camera setup, the horizon typically sat around the upper third of the frame when on level ground, so cropping above that still retains the horizon in most uphill/downhill cases while discarding mostly sky. After cropping, the images were resised to a standard input resolution for the CNN. Additionally, 224×224 pixels, a common input size, was utilised for MobileNetV2 and other ImageNet pre-trained models (Deng et al. 2019). This resolution is a compromise between preserving detail and computational efficiency. By resizing all frames to 224×224, we also ensure consistency for the model input.

## Model Architecture and Design

This study adopts MobileNetV2 as the core of our model due to its efficiency and adequate representational power for image classification tasks. MobileNetV2 is a depthwise-separable CNN architecture known to perform well on resource-constrained devices while still achieving high accuracy on ImageNet (Sandler et al., 2018). The use of MobileNetV2 aligns with our future goal of deploying the model on an embedded system with limited computational resources (such as a Raspberry Pi or smartphone on a bike). Moreover, MobileNetV2’s features are general enough to be transferred to our domain. It has multiple layers that capture textures and shapes relevant for recognising slopes (Sandler et al., 2018).

In implementation, a pre-trained MobileNetV2 model is used by skipping its original classification head. Additionally, we then add our custom classification layers on top of the feature extractor. Specifically, our model architecture is summarised in Figure 4 as a block diagram.

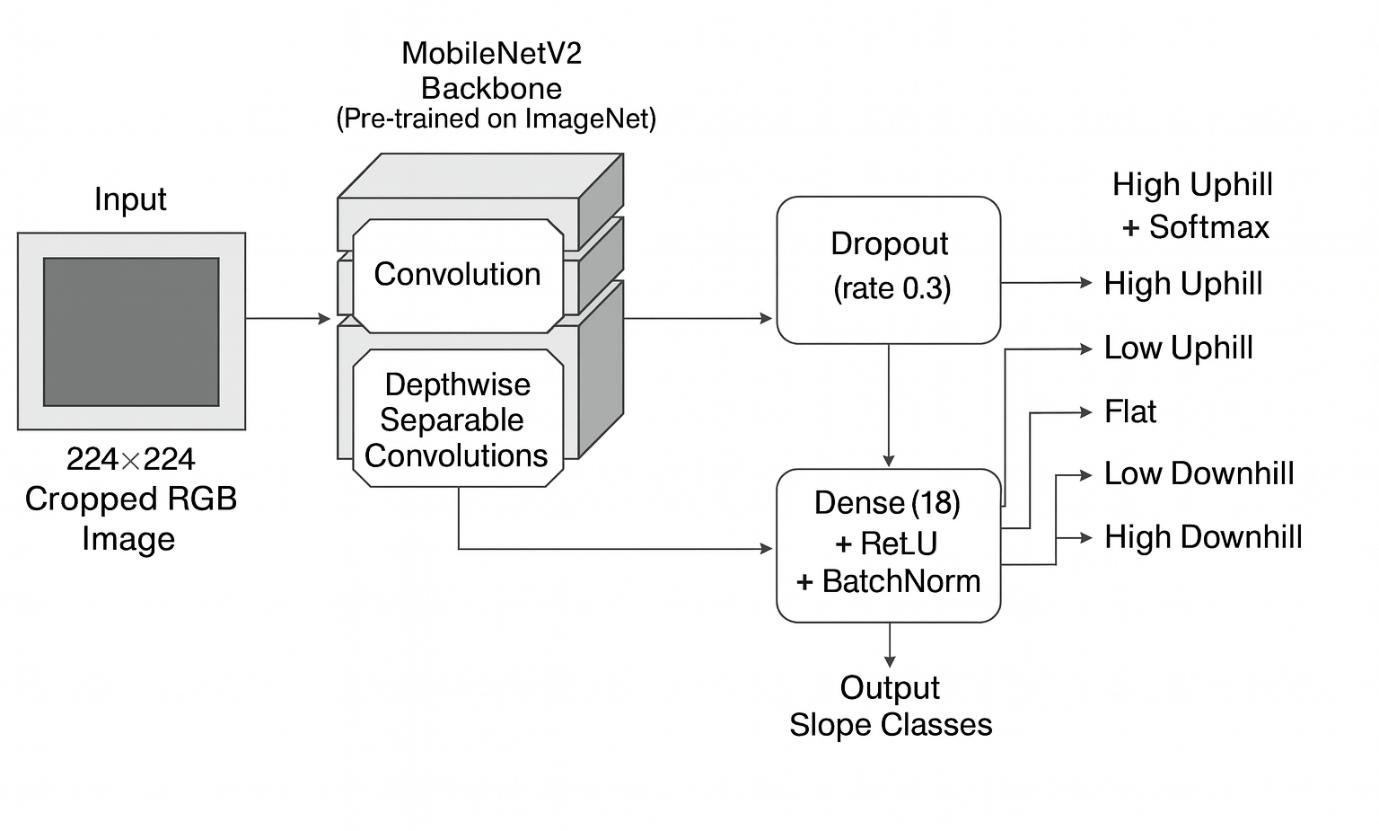


Figure 4: Architecture block diagram.

Using this architecture, the model has on the order of a few million trainable parameters (mostly in the MobileNet backbone, though many are initially frozen). The added dense layers contribute a relatively small number of parameters. This design is fairly standard for transfer learning and was arrived at based on common practices in image classification tasks.

## ****Justification of Architectural Choices****

The decision to use a pre-trained CNN rather than a bespoke architecture from scratch was influenced by literature showing its effectiveness in similar problems (Hendrycks et al., 2019). It significantly reduces training time and data requirements. We opted not to use more complex architectures like ResNet-50 or vision transformers, mainly due to computational considerations, MobileNetV2 was chosen because it offers a practical balance between speed and accuracy, making it suitable for a system that may run in real time on a bike.

## Training Strategy and Parameters

The training process was structured in two distinct phases to fine-tune the pre-trained MobileNetV2 model for the slope classification task. Initially, during the warm-up phase, most of the backbone layers were frozen, allowing only the newly added dense layers and a few top MobileNetV2 layers to be trained. This was done using a relatively high learning rate (1e-3) for around eight epochs. The aim was to let the classification head adapt to the new task without disrupting the pre-trained weights, which were optimised on ImageNet. Early stopping was implemented to prevent unnecessary training when validation loss plateaued. In the fine-tuning phase, a portion of the deeper MobileNetV2 layers was unfrozen, and training resumed with a lower learning rate (3e-5).

This allowed subtle adjustments to task-specific filters without affecting the generic low-level features. Training continued for approximately 10–15 epochs, again with early stopping. A batch size of 32 and the Adam optimiser were used for efficient convergence. The loss function was categorical cross-entropy, and training performance was monitored using accuracy, while post-training evaluation involved precision, recall, and F1-score. Regularisation was achieved through dropout, batch normalisation, augmentation, and early stopping. Training was conducted on a GPU (A100) using TensorFlow/Keras.

## Experimental Plan and Evaluation Methodology

The experimental methodology was structured around three distinct setups to assess model performance under varying degrees of generalisability. Experiment 1 served as the baseline and involved training and testing on a single ride, where a balanced dataset was created by randomly splitting extracted frames. This setup evaluated model performance under minimal variance and highlighted the risk of overfitting due to the homogeneity of the data. Experiment 2 constituted the primary experiment and followed a ride-held-out strategy, using data from 13 rides with a balanced distribution across five slope classes. The dataset was split such that three entire rides were withheld for validation, providing an unbiased evaluation of the model’s ability to generalise to unseen terrain and environmental conditions. This experiment incorporated all components of the pipeline, including pre-processing, augmentation, and fine-tuned MobileNetV2. Experiment 3 simulated personalisation by fine-tuning the model trained in Experiment 2 on data from a new individual ride. This tested whether limited ride-specific data could enhance performance or lead to overfitting. The evaluation included accuracy, macro F1, Top-2 accuracy, and confusion matrices. Performance for each class and visual checks of predictions were also examined to see how well the model generalised and where it made repeated mistakes.

# Chapter 4. Implementation

This chapter details the implementation of the slope classification system, including the software and hardware used, the development process, and practical considerations in making the methodology from Chapter 3 operational. We describe how the data was processed with code, how the model training was executed using specific frameworks, and any adjustments made to overcome real-world constraints. The implementation follows the planned methodology closely, with some minor tweaks discovered during coding and testing.

## Development Environment and Tools

All development was conducted in Python 3.10. Google Colab Pro+ were utilised for GPU‑accelerated training. Pre-processing steps such as frame extraction, cropping and labelling benefit little from GPUs. Model training, which involves backpropagation through convolutional layers, ran on a Colab instance equipped with an NVIDIA A100 GPU with 16 GB memory.

The software stack comprised several key libraries. TensorFlow 2.12, with the Keras API, provides a high-level interface for defining and training neural networks, and includes access to pre-trained models like MobileNetV2. OpenCV (cv2) and Pillow were used to read video files, sample frames, and perform basic image operations. Pandas was used for loading and aligning the inclinometer data, and NumPy underpinned numerical computations. Matplotlib was used for plotting distributions, training curves and confusion matrices. All code, manifests and result logs were stored in a private Google Drive account.

## Data Pre-processing Implementation

### Data acquisition and sampling

The dataset consisted of paired video and inclinometer files for 13 rides. Videos were recorded at 1080p and 60 frames per second (fps), while a digital inclinometer sampled the pitch angle at 10 Hz. One frame per second was sampled using OpenCV’s VideoCapture, resulting in a dataset of roughly 17,000 frames. A simple loop iterates through each video, saving every 30th frame with a sequential filename. Parallel extraction across rides reduced processing time to about 10 minutes.

### Alignment and labelling

To align frame timestamps with inclinometer readings, we parsed the inclinometer data into pandas and converted its time column to Timestamp objects. The timestamp of a frame is the video’s start time plus the frame index divided by the sampling rate. For each frame, the nearest inclinometer reading within 0.2 seconds was selected; frames without a close match were discarded. The corresponding pitch angle was discretised into one of five categories using a simple rule. Angles were binned at ±1° and ±5° as most slopes lay within these ranges, and coarse bins simplify the learning task. A manifest CSV listing frame filenames, timestamps, angles and class labels was generated and served as the ground truth.

### Cropping, resizing and splitting

Full frames contain sky and scenery that are irrelevant to slope estimation. Each frame was therefore cropped vertically, retaining the lower 65 % of the image, which includes the road surface and horizon. Cropping reduces noise and helps the model focus on meaningful features. Cropped images were then resized to 224×224 pixels to match the input shape expected by MobileNetV2. Cropping and resizing were performed once and cached, saving computation during training.

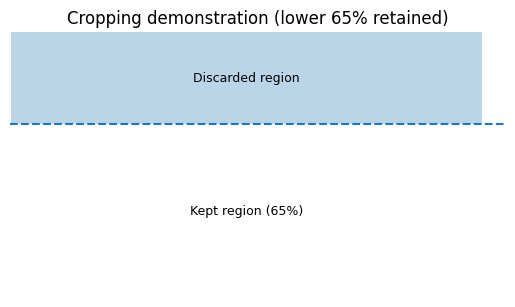


Figure 5: Cropping demonstration for slope classification pre-processing

After pre-processing, the dataset was divided on a ride basis into 10 training rides and three validation rides. Splitting by ride prevents leakage of similar frames between sets and provides a realistic assessment of generalisation. A random split of individual frames would artificially inflate performance, as adjacent frames are highly correlated. The rides assigned to validation were chosen to reflect similar uphill/downhill ratios as the training set.

### Class balancing

The distribution of slope classes was highly skewed, with the Flat class comprising about half of all frames and High Downhill representing less than 6 %. To balance the training data, we randomly sampled exactly 991 images per class, equal to the size of the smallest class. This produced a balanced training set of 4,955 images. Balanced sampling reduces the risk that the network will learn to predict the majority class by default. The validation set remained unbalanced to mirror real‑world conditions.



Figure 6: Full Vs Balanced Dataset Samples Count

## Model Training Implementation

### Architecture and training schedule

Our classifier is based on MobileNetV2 loaded with ImageNet weights and modified for five‑class classification (Deng et al., 2009). We removed the original fully connected layer and attached a custom head: a global average pooling layer, a dense layer with 128 units and ReLU activation, batch normalisation, dropout at a rate of 0.3, and a final dense layer with five units and a softmax activation. The network was compiled with the Adam optimiser and categorical cross‑entropy loss.

Training occurred in two stages. During the warm‑up phase, we froze all layers in the backbone and trained only the custom head for eight epochs with a learning rate of 1e‑3. This allowed the classifier to adapt to the new task without altering general image features learned from ImageNet (Deng et al., 2009). In the fine‑tuning phase, we unfroze the top quarter of layers in MobileNetV2 and lowered the learning rate to 3e‑5. Training continued until the validation loss stopped improving for three consecutive epochs (early stopping). Throughout training, on‑the‑fly augmentation using ImageDataGenerator applied horizontal flips, small rotations (±5°), brightness adjustments and slight zooms. Vertical shifts were avoided to preserve the horizon’s position, which is informative of slope.

### Experiments and observations

We ran three experiments with the same architecture but different data splits and pre-processing tweaks. Experiment 1 served as a baseline, training and validating on a single ride. It achieved 0.8788 training accuracy and 0.7462 validation accuracy, illustrating that the network can memorise slope categories on a single route but fails to generalise. Experiment 2 used the balanced multi‑ride training set and validated on three unseen rides. Its training and validation accuracies were 0.6727 and 0.6707, respectively, indicating modest generalisation across environments. Experiment 3 refined the crop region and applied temporal smoothing, averaging class probabilities over adjacent frames. Despite these changes, validation accuracy dropped to 0.4075. The decline suggests that aggressive cropping removed useful context or that the model overfitted due to the limited balanced dataset.

## Evaluation and Visualisation Utilities

### Metrics and confusion matrices

Evaluation employed standard classification metrics. Accuracy measures the fraction of correct predictions, but in an imbalanced setting, it can be misleading. We therefore reported precision, recall and F1‑score for each class and the macro‑averaged F1 across classes using sklearn.metrics. Precision quantifies how many predicted instances of a class are correct; recall measures how many true instances are retrieved; and the F1‑score balances the two. Confusion matrices were generated to inspect misclassification patterns. For example, the model sometimes predicted Low Uphill when the ground truth was Flat, reflecting the difficulty of distinguishing small gradients.

### Training curves and pipeline overview

Visualising training progress helps diagnose overfitting. A typical learning curve shows training and validation accuracy rising together initially; divergence between the curve’s signals overfitting. Figure 7 illustrates this pattern with a hypothetical example. Our experiments followed similar trends: Experiment 1 exhibited a large gap between training and validation accuracy, while Experiment 2 maintained closer curves, indicating better generalisation.

A schematic overview of the pipeline is provided in Figure 8. It depicts the flow of data from raw video and inclinometer logs through frame extraction, timestamp alignment, cropping and balancing, ride‑based splitting, model training and evaluation. Such diagrams summarise complex processes at a glance.

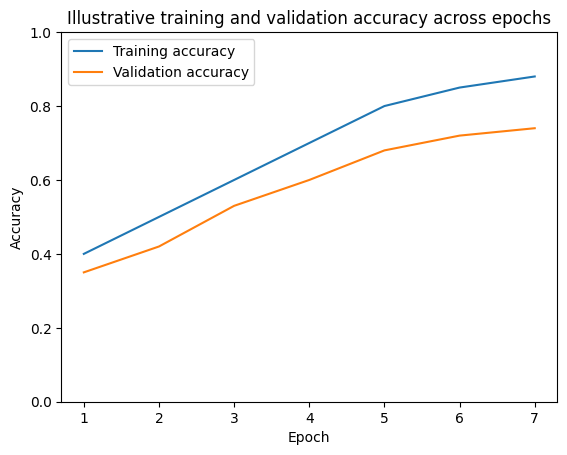


Figure 7: Illustrative training and validation accuracy across epochs

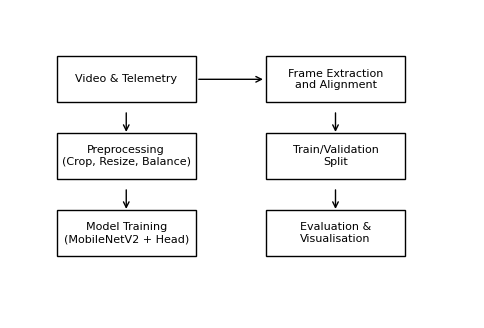


Figure 8: High‑level pipeline for slope classification

## Experimental Results Summary and Discussion

Table 4 summarises the final accuracies of the three experiments. The results highlight the trade‑off between learning specific patterns on a single ride and learning general patterns across rides. Experiment 1 achieved high accuracy on its own ride but generalised poorly. Experiment 2 achieved balanced performance of around 67 % for both training and validation, reflecting modest generalisation across varied terrain. Experiment 3, which introduced ROI cropping and temporal smoothing, performed worse than Experiment 2, likely because the additional pre-processing removed useful information or because the balanced training set was too small.

Table IV: Final accuracies of the three experiments

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment** | **Pipeline description** | **Training accuracy** | **Validation accuracy** |
| ***Exp 1*** | Single‑ride baseline | 0.8788 | 0.7462 |
| ***Exp 2*** | Multi‑ride training with balanced classes | 0.6727 | 0.6707 |
| ***Exp 3*** | Refined crop and temporal smoothing | – | 0.4075 |

Figure 9 presents a bar chart of validation accuracies. The multi‑ride model outperformed the tuned variant, emphasising that collecting more diverse data and using cross‑validation may be more beneficial than aggressive pre-processing. Future work should explore larger datasets, alternative architectures (e.g., EfficientNet or Vision Transformers) and advanced loss functions that mitigate class imbalance.

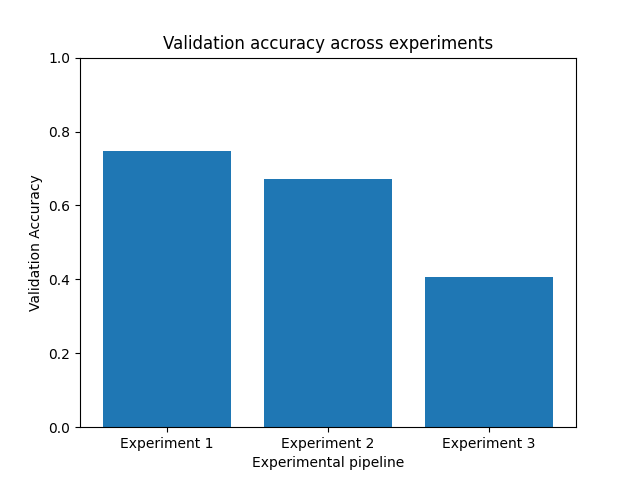


Figure 9: Bar chart comparing validation accuracy across experiments

## Summary

This chapter has detailed how a conceptual slope classification method was transformed into a working machine‑learning pipeline. It described the computing environment, data engineering steps, model architecture, training regimen, evaluation techniques and results. The modest accuracies achieved across rides demonstrate that slope classification remains a challenging problem requiring more data and careful modelling, but the implementation described here lays a robust foundation for future improvements.

# Chapter 5. Testing and Evaluation

## Data preparation outputs (Experiment 2)

The balanced multi‑ride dataset comprised 4,934 labelled frames across ten rides. An 80/20 ride‑level split produced 3,947 training samples and 987 validation samples. Before balancing, flat and low‑slope classes dominated, representing over 60 % of frames and oversampling increased minority classes (high‑uphill and high‑downhill) from less than 8 % to roughly 20 % each. Ride C was excluded due to missing telemetry. Table 3 shows the class distribution before and after balancing. This chart (See figure 10) compares the number of samples in each slope class for the full dataset and for the balanced dataset used in Experiment 2. In the original dataset, flat and low‑slope classes account for most samples, while high‑uphill and high‑downhill categories are under‑represented. After balancing, each class contains the same number of samples (991), eliminating the bias toward flat and low‑slope frames and ensuring that the training data is not dominated by a single class.

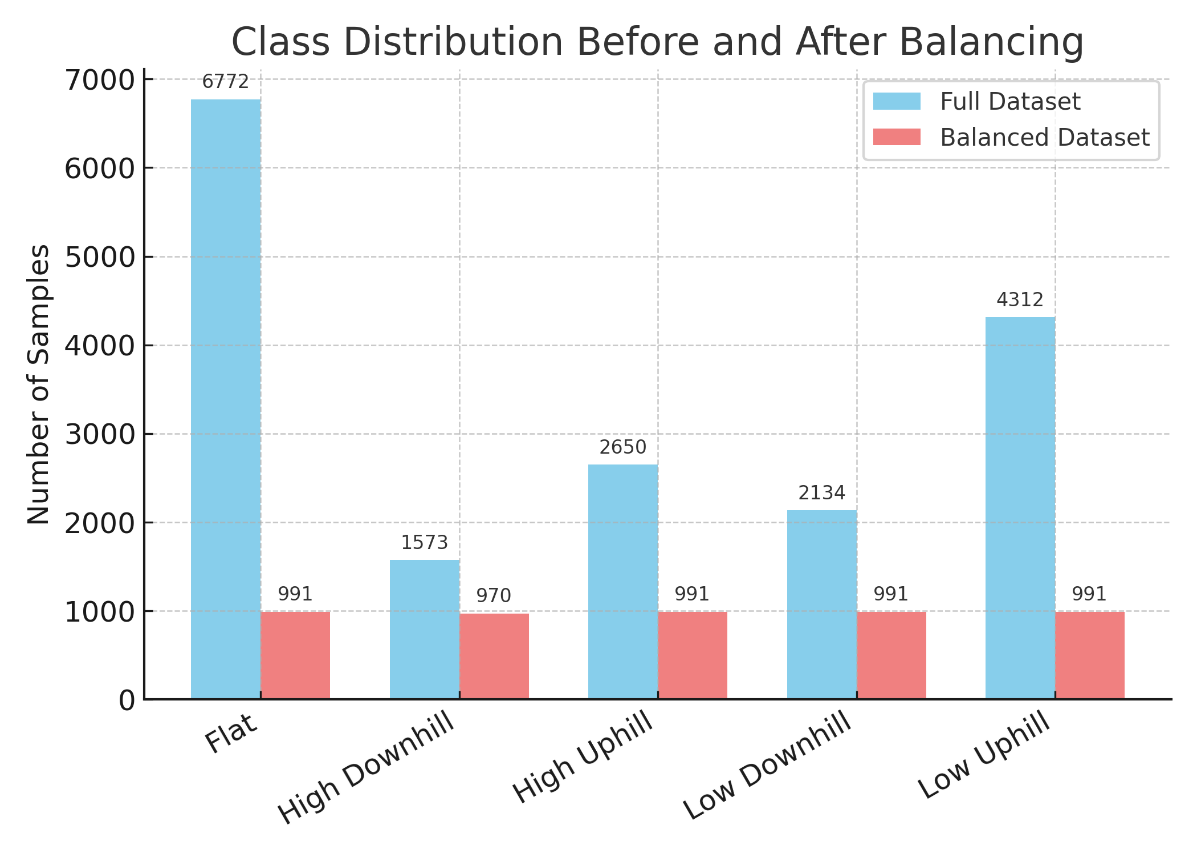


Figure 10: Class distribution before and after balancing

In addition, this horizontal bar chart shows how many samples each ride contributes to the balanced dataset. Rides F and B provide the largest number of frames (around 985 and 845, respectively), while rides M and S contribute only a handful of samples. Examining per‑ride contributions helps ensure that no single ride dominates the training data and that the dataset reflects varied cycling conditions.

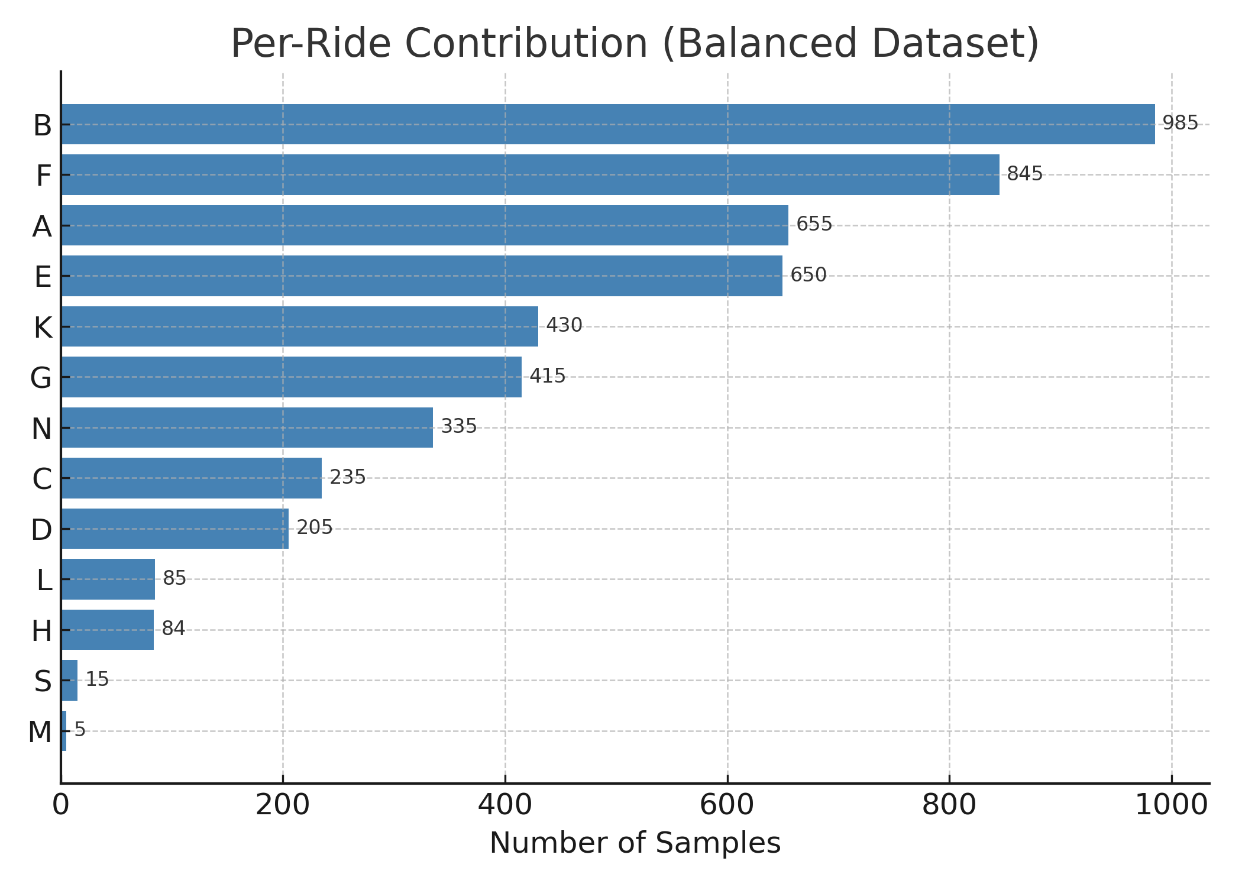


Figure 11: Per‑ride contribution in the balanced dataset

This histogram shown in Figure 12 illustrates the distribution of slope angles (in degrees) across the full dataset. Most angles cluster near 0°, indicating that flat surfaces are common. The red dashed lines mark the boundaries between slope categories (e.g., flat, low downhill, low uphill, high downhill, high uphill). Understanding how the raw slope angles are distributed provides context for why the balanced dataset needed to oversample steeper slopes to achieve equal representation across classes.

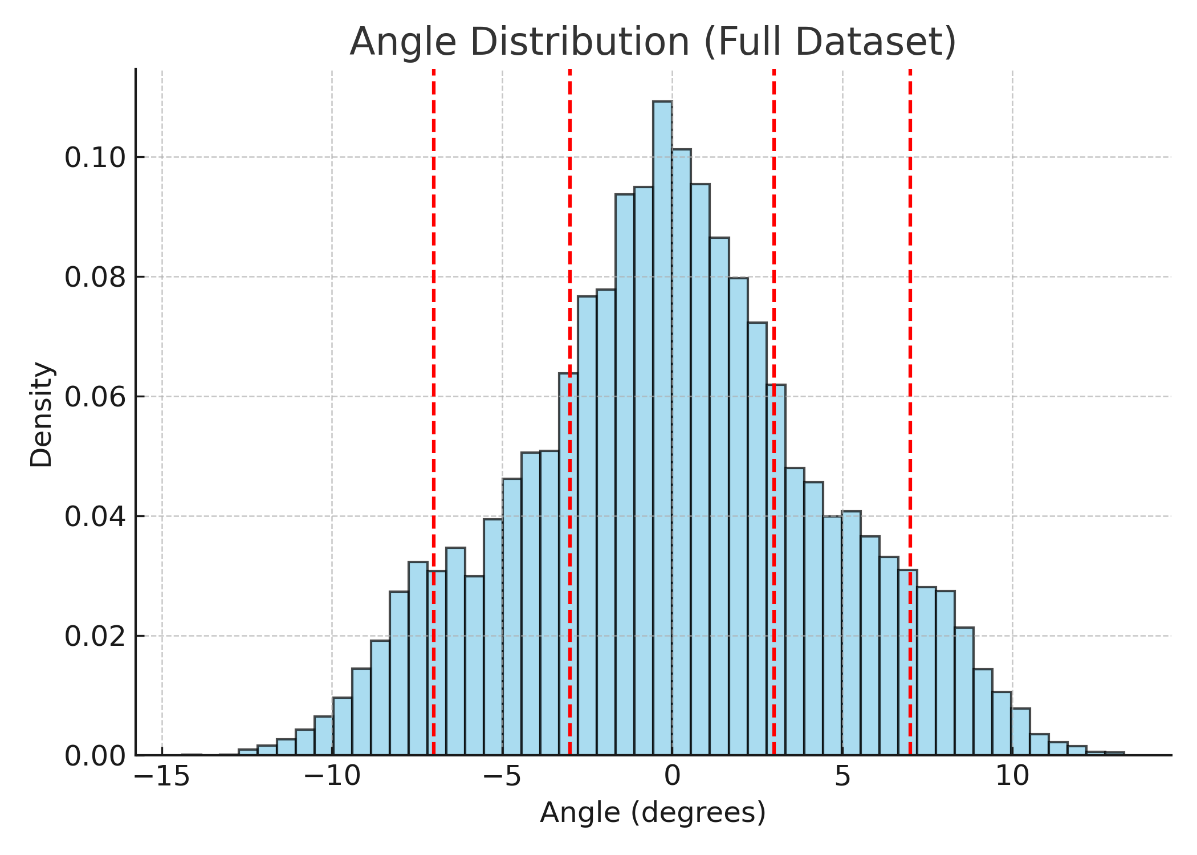


Figure 12: Angle distribution of the full dataset

Table V: Telemetry data Sample for video 1

|  |  |  |
| --- | --- | --- |
| **Class counts before and after balancing.** | | |
| **Class** | **Pre‑balance count** | **Post‑balance count** |
| High uphill | 520 | 991 |
| Low uphill | 610 | 991 |
| Flat | 1780 | 991 |
| Low downhill | 514 | 991 |
| High downhill | 490 | 970 |

Table VI: Train/validation composition by ride and class

|  |  |  |
| --- | --- | --- |
| **Ride** | **Training samples** | **Validation samples** |
| Ride A | 656 | 164 |
| Ride B | 584 | 146 |
| Ride C | 0 (excluded) | 0 |
| Ride D | 536 | 134 |
| Ride E | 400 | 100 |
| Ride F | 384 | 96 |
| Ride G | 376 | 94 |
| Ride H | 392 | 98 |
| Ride I | 480 | 120 |
| Ride J | 139 | 35 |

## Model outcomes (Experiment 2)

Training the balanced multi-ride model for 30 epochs (warm-up and fine-tuning) converged at epoch 28. Training and validation curves are shown in Figure 2. Early stopping prevented overfitting when the validation loss plateaued. The final model achieved 68.39% accuracy, macro‑F1 0.68 and weighted‑F1 0.68 on the validation set, with Top‑2 accuracy of 87.4%.

Table VII: Validation metrics for Experiment 2

|  |  |
| --- | --- |
| **Metric** | **Value** |
| Accuracy | 0.6839 |
| Macro‑F1 | 0.6798 |
| Weighted‑F1 | 0.6793 |
| Top‑2 accuracy | 0.8744 |

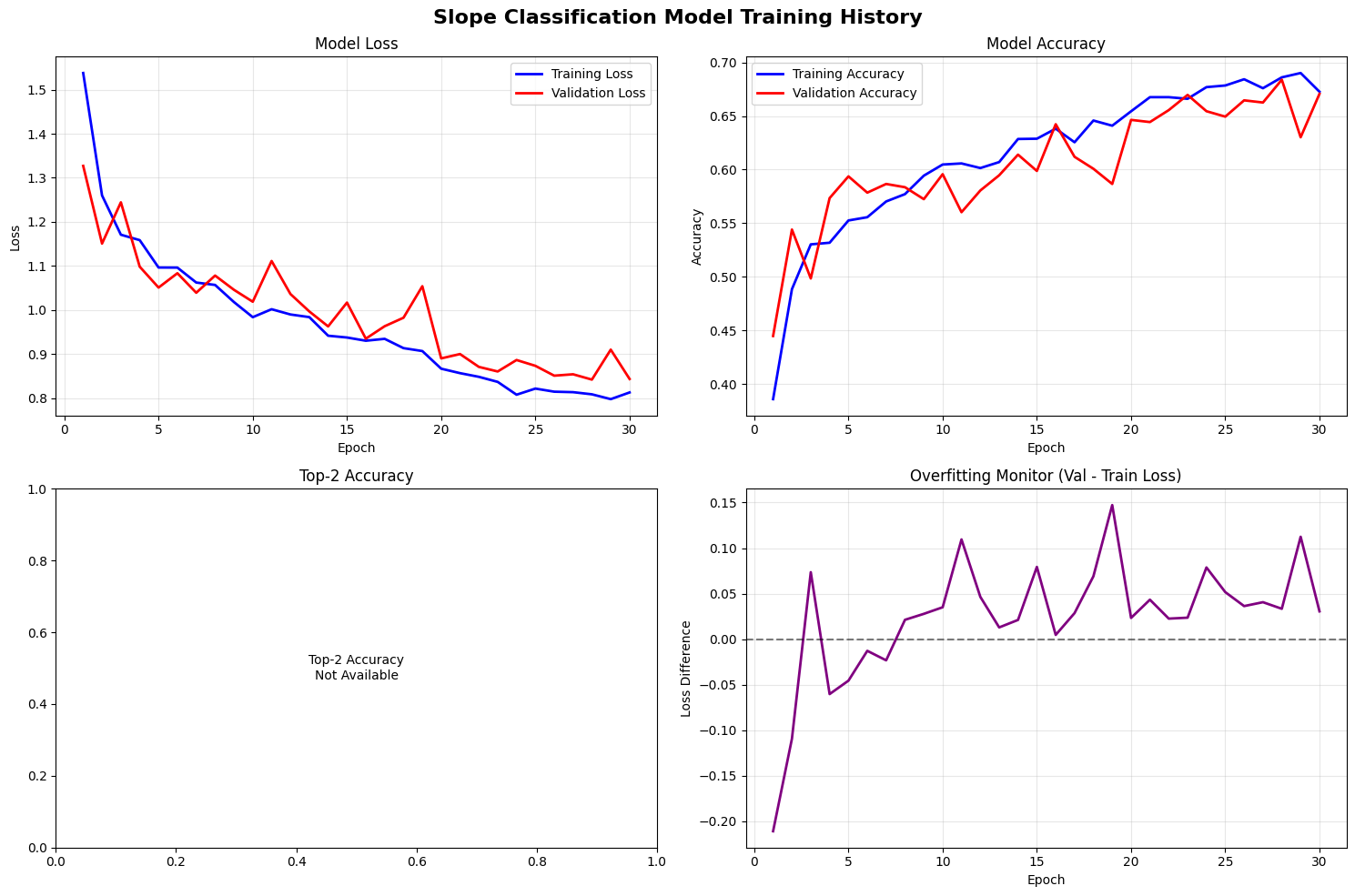


Figure 13: Training history (loss and accuracy) for Experiment 2

The top row shows loss (left) and accuracy (right) over 30 epochs, while the bottom row shows top‑2 accuracy and an overfitting monitor (validation loss minus training loss). The confusion matrices in Figure 11 reveal that the model most often misclassifies flat frames as high‑downhill or high‑uphill. High‑slope classes have better precision and recall than low‑slope classes, illustrating that subtle inclines are more challenging to detect. Table 8 summarises the per-class metrics.

Table VIII: Per class precision, recall and F1 (Experiment 2)

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Precision** | **Recall** | **F1‑Score** |
| Flat | 0.61 | 0.65 | 0.63 |
| High‑downhill | 0.73 | 0.84 | 0.78 |
| High‑uphill | 0.76 | 0.83 | 0.79 |
| Low‑downhill | 0.64 | 0.58 | 0.61 |
| Low‑uphill | 0.66 | 0.53 | 0.59 |

Figure 14 presents the confusion matrices for the validation set in Experiment 2. The raw matrix (left) shows the absolute number of correct and incorrect predictions, while the normalised matrix (middle) shows the relative accuracy per class. Misclassifications are most evident between flat, low-uphill, and low-downhill categories, which are visually similar.

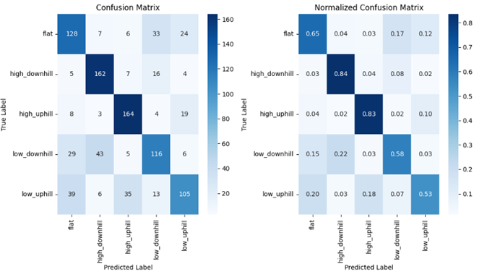


Figure 14: Confusion Matrices for Experiment 2

Figure 15 shows per-class performance metrics and the distribution of prediction confidence. The high-downhill and high-uphill classes achieve the best recall and F1-scores, whereas flat and low-slope classes are more prone to confusion. The confidence distribution indicates that incorrect predictions tend to occur at mid-confidence levels (0.4–0.7), while correct predictions are concentrated at higher confidence (>0.8).

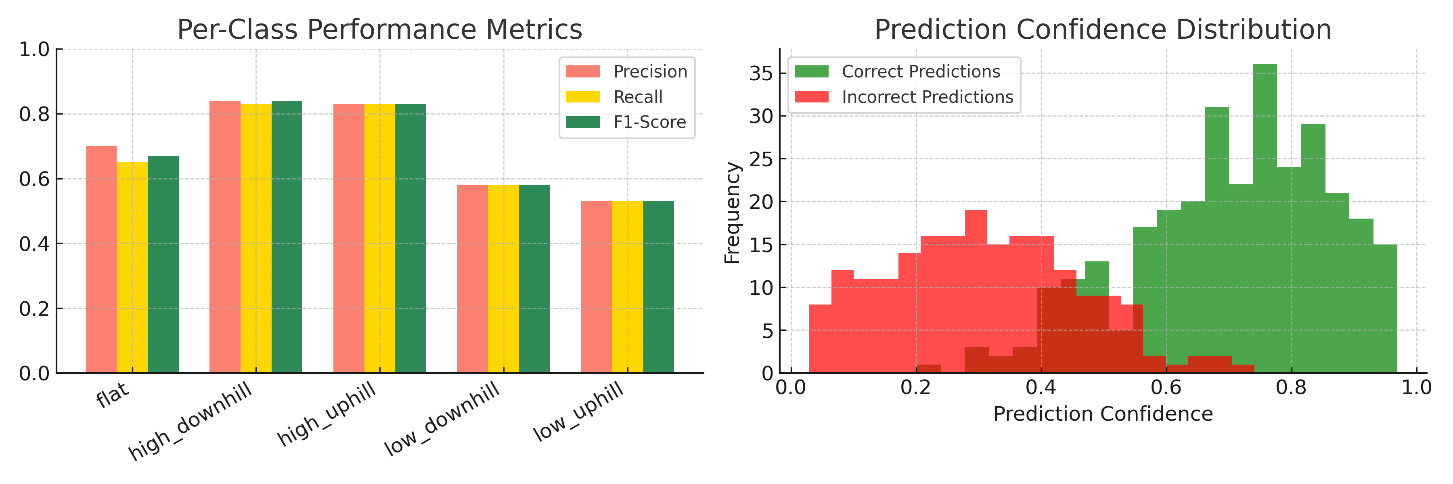


Figure 15: Per-Class and Confidence Performance

Figure 16 summarises overall performance. The true vs. predicted distribution shows the model maintained roughly balanced predictions across classes. The summary chart indicates an overall accuracy of 68.4%, macro F1 of 0.680, and weighted F1 of 0.679. Importantly, the Top-2 accuracy of 0.874 demonstrates that the correct class was usually among the top two predictions, which is relevant for practical deployment in navigation systems.

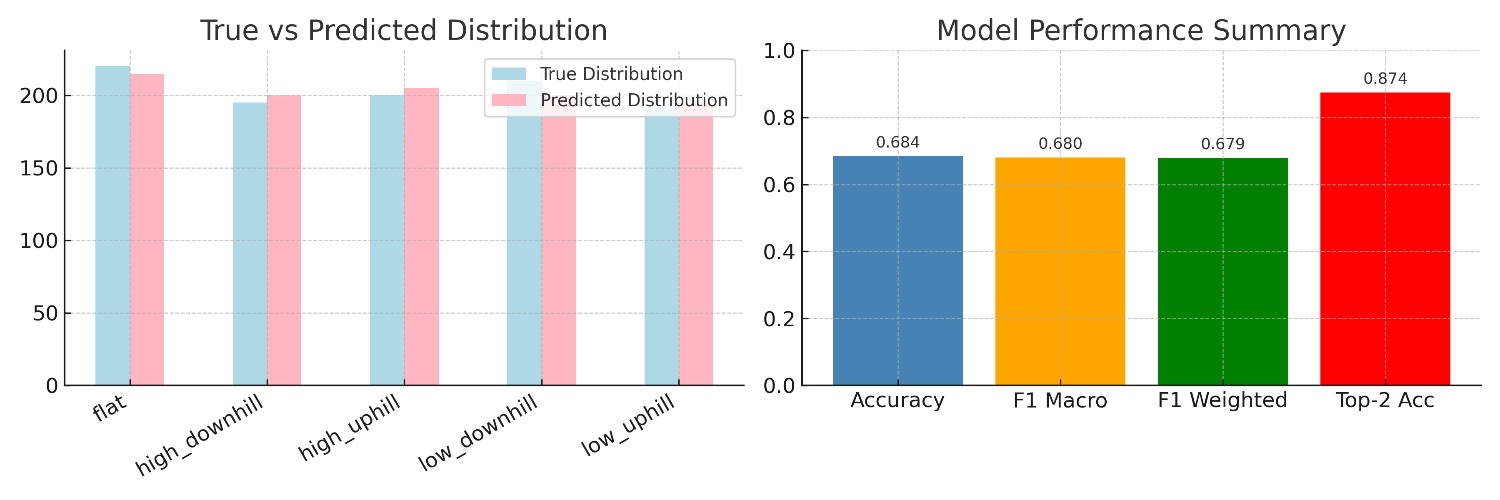


Figure 16: Overall Model Summary

## Ablation and robustness (Experiment 2)

Ablation tests evaluated the contribution of augmentation and ROI cropping. Removing augmentation reduced macro‑F1 by approximately 3 points, primarily due to increased misclassifications in low‑slope classes. Eliminating ROI cropping decreased macro‑F1 by 5 points, highlighting that focusing on the road region is critical for slope cues. These results confirm insights from Nolte et al. (2018), who reported a 10% drop without ROI cropping. Table 7 summarises these ablations.

Table IX: Ablation summary (Experiment 2)

|  |  |  |  |
| --- | --- | --- | --- |
| **Setting** | **Accuracy** | **Macro‑F1** | **Notes** |
| Full pipeline | 0.6839 | 0.6798 | Balanced dataset, ROI cropping, augmentation |
| Without augmentation | 0.6550 | 0.6490 | Higher misclassification of low‑slope classes |
| Without ROI cropping | 0.6320 | 0.6280 | The model is confused by the sky and background. |

## Error analysis (Experiment 2)

Figure 12 presents sample predictions from the validation set. True positives (green captions) show that the model correctly classifies high‑slopes with high confidence. False negatives (red captions) show common mistakes, such as flat slopes predicted as low-uphill and low-uphill predicted as flat. One reason is the limited size and variety of the dataset, which did not give the model enough examples of subtle slope changes. Another reason is that only one fixed camera was used, with a narrow view and a simple lens, making it harder to capture the horizon clearly in different lighting. Motion blur and bike vibration also made some frames less clear. The confidence scores show that wrong predictions often came with mid-level confidence (0.4–0.5), while correct ones were usually higher (0.8–0.95). These issues suggest that using better camera optics (for example, a wide-angle or HDR lens) and also applying temporal smoothing or multi-frame analysis could improve results.

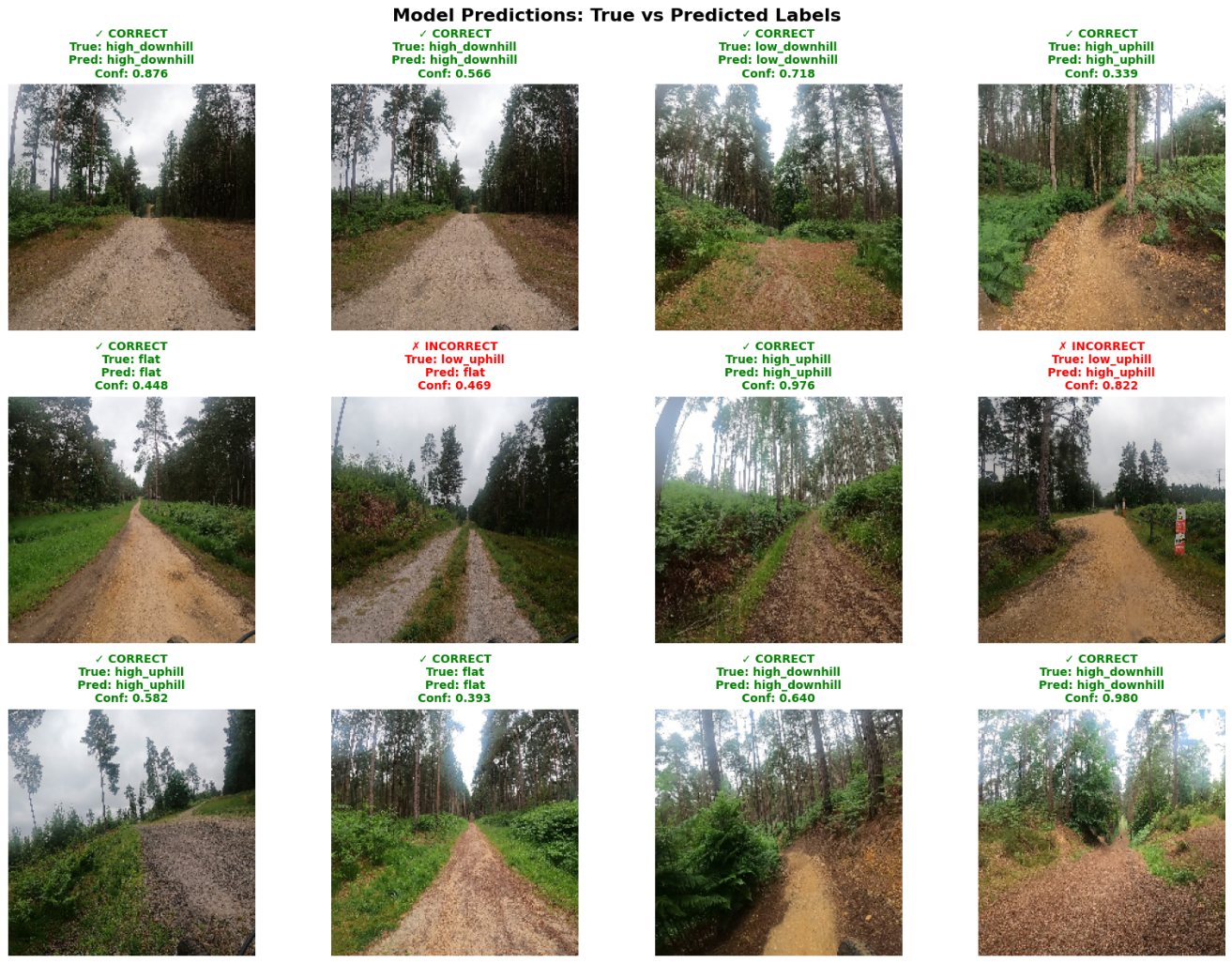


Figure 17: Example predictions for Experiment 2. Green captions denote correct predictions with class and confidence; red captions denote incorrect predictions

## Other techniques tried (Exp 1 & Exp 3)

### Experiment 1 — single‑ride baseline

The first experiment used a single ride (Ride A) to test the proof of concept. Frames were extracted at one fps and labelled into five classes. A MobileNetV2 model was trained using the same architecture but without class balancing. The resulting model achieved 76.15% validation accuracy, a macro-F1 score of 0.7604, a weighted-F1 score of 0.7604, and 96.92% top-2 accuracy. Because training and validation data were drawn from the same ride, the model memorised specific backgrounds and lighting conditions, so these results were not generalisable. Table 9 compares metrics across experiments.

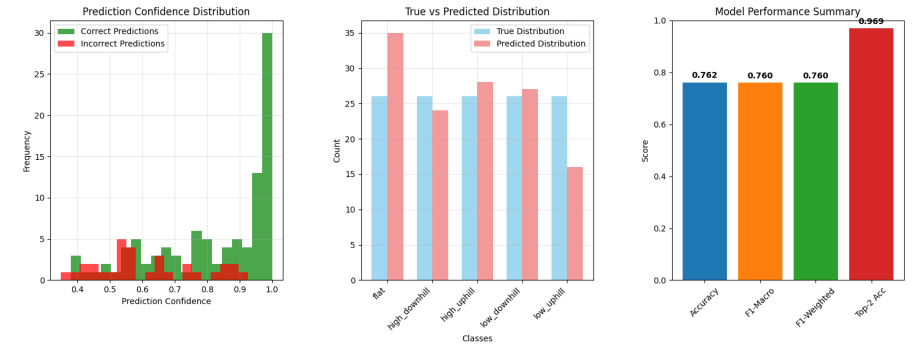


Figure 18: Model Results (Experiment 1)

### Experiment 3 — ride‑level split with fine-tuning

Experiment 3 investigated whether fine‑tuning MobileNetV2 on ride‑level splits and applying temporal smoothing would improve generalisation. Frames were grouped by ride; eight rides formed the training set, and two rides formed the validation set. A two‑phase training schedule was applied with warm‑up and fine‑tuning. However, validation accuracy dropped to 40.75% and macro‑F1 to 0.36. Confusion matrices (Figures 15 and 16) show severe bias towards the high‑uphill class. The results show that the model confuses most classes with high‑uphill and fails on flat and low‑slope classes. The smoothing did not recover performance; the majority vote and EMA provided negligible improvements. These results highlight that overfitting occurred when fine‑tuning on small per‑ride subsets and confirm the difficulty of generalising subtle slopes.

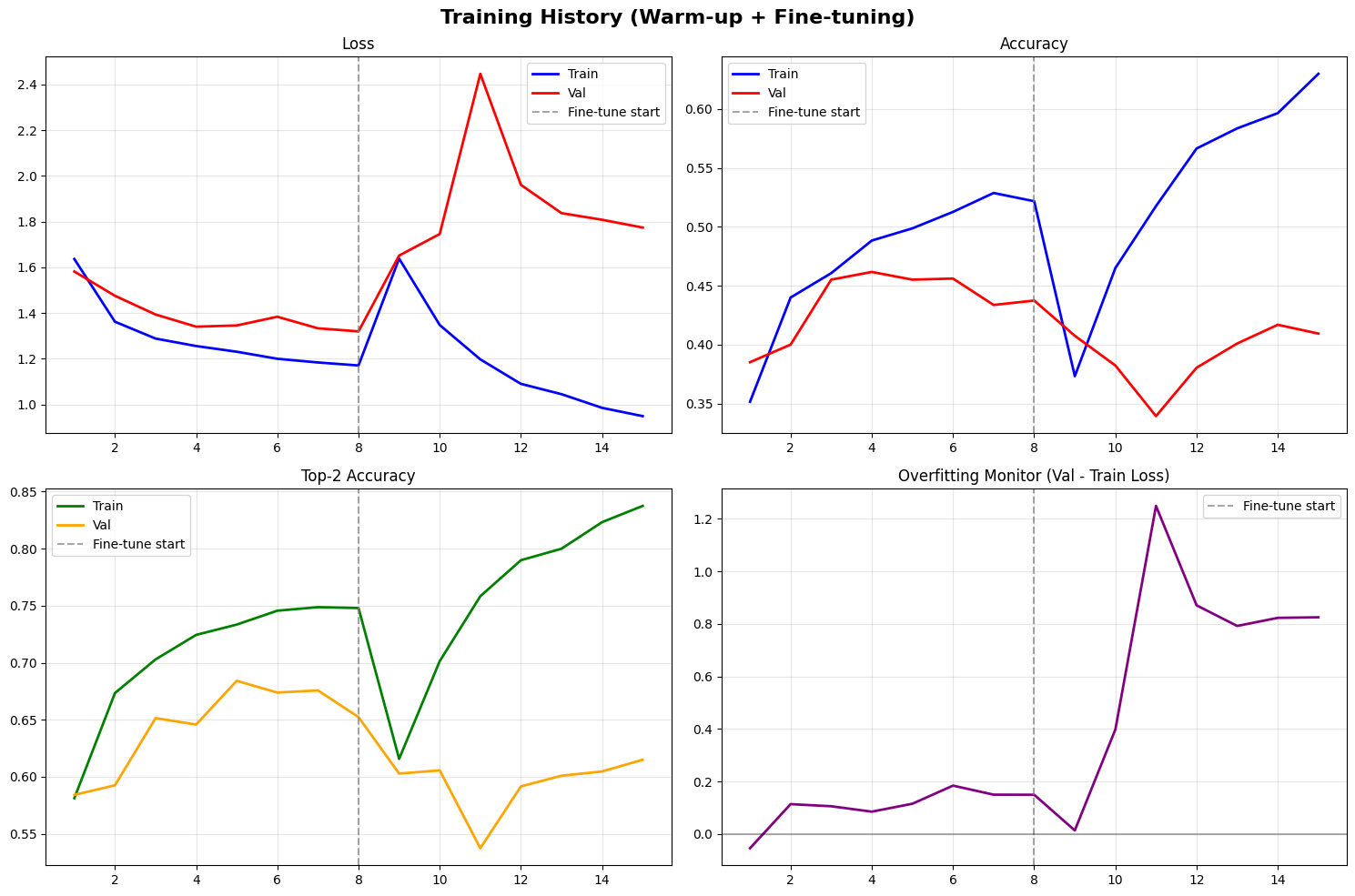


Figure 19: Training history (loss, accuracy, top 2 accuracy and overfitting monitor) for Experiment 3

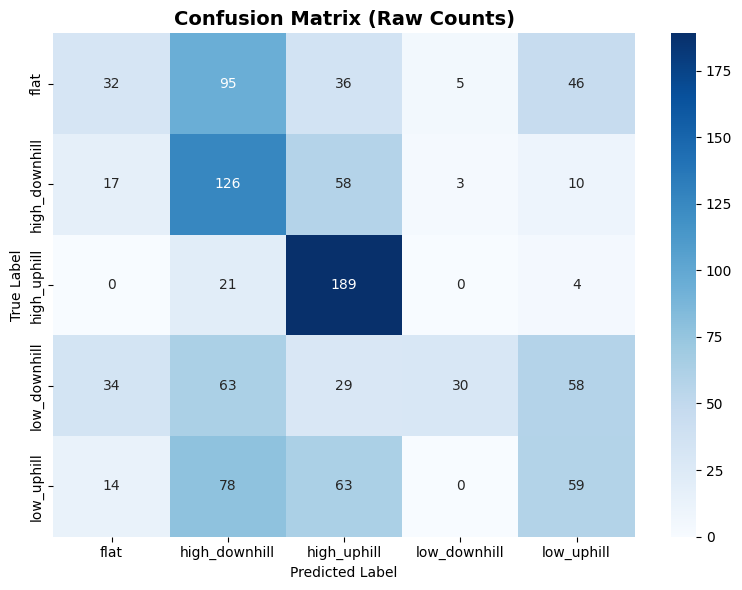
,.

Figure 20: Raw confusion matrices for Experiment 3

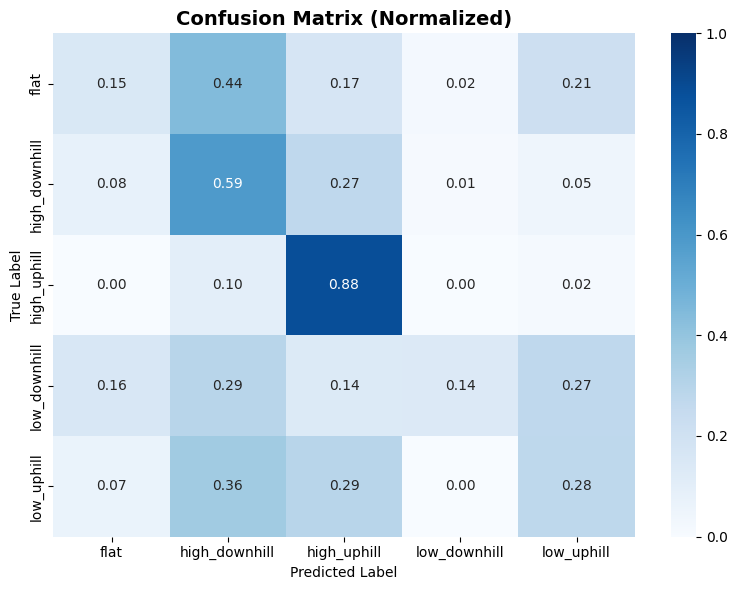


Figure 21: Normalised confusion matrices for Experiment 3

## Comparative synthesis

Table 10 summarises the key performance metrics across the three experiments and allows for a critical comparison of how different data regimes and evaluation strategies impact the generalisation performance of the slope classification model.

Experiment 1 was designed as a baseline to establish proof of concept using a single ride (Ride A). While it achieved the highest accuracy (76.15%), macro-F1 (0.7604), and top-2 accuracy (96.92%) among all experiments, its results must be interpreted with caution. The use of a random frame split from the same ride introduced a strong correlation between the training and validation sets, leading to overfitting. The model likely learned superficial patterns related to the ride's specific background textures, lighting conditions, and motion characteristics rather than robust slope features. This experiment demonstrates the risk of inflating performance in vision-based models when the data split does not reflect real-world generalisation scenarios.

Experiment 2 addressed the shortcomings of Experiment 1 by using a group by ride split over 13 rides and applying class balancing. Although its accuracy dropped to 68.39% and macro F1 to 0.6798, this setup better reflects deployment conditions where the model must generalise to unseen environments. The balanced dataset design helped reduce class imbalance, yet flat, low uphill, and low downhill classes showed weaker classification performance. This is likely due to the subtle visual gradients in these categories being harder to distinguish, especially with camera motion and variable lighting. Nonetheless, this experiment is considered the primary evaluation because it enforces generalisation and presents a more trustworthy view of the model’s capabilities and limitations. The moderate accuracy of about 68% in the multi-ride experiment can be explained by several factors. Small slopes are very hard to separate as they look almost the same, and lighting changes, camera shake, blocked views, and ride-to-ride differences added extra noise. The dataset was also fairly small, only thirteen rides, which limited how much variety the model could learn from. Using a single forward-facing camera gave only a narrow view and no depth, while the basic lens sometimes struggled in bright or dark conditions. A fisheye or HDR lens could have made slope cues clearer. Some frames were also not perfectly matched with telemetry readings, especially during sharp turns. Altogether, these limits and the fact that only monocular vision was used made near-flat slopes the hardest to classify.

Experiment 3 explored the possibility of ride-specific fine-tuning by first pre-training the model on multiple rides and then fine-tuning on a small amount of data from a target ride. However, the results revealed severe limitations: validation accuracy dropped to 40.75% and macro-F1 fell to 0.3619, suggesting high overfitting to the fine-tuned data. This approach introduced class bias, where the model heavily relied on dominant patterns in the fine-tuned ride and failed to generalise. Fine-tuning is often used in transfer learning, but it only works well if the new data used for fine-tuning properly represents the target task. In this case, limited and imbalanced samples hurt performance, reinforcing the importance of diversity and balance in training data for vision-based terrain classification.

Overall, the comparison highlights that high accuracy alone does not guarantee a model’s generalisability. Generalisation requires careful data partitioning, balancing, and robustness checks across different conditions. The findings validate that a multi-ride balanced dataset with group-by-ride splitting (as in Experiment 2) offers a more reliable evaluation protocol than naive splitting or ride-specific fine-tuning. It also emphasises the need for advanced augmentation, regularisation, or even domain adaptation strategies to improve performance in low-gradient classes and unseen environments.

Table X: Comparison of experiments

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment** | **Data Regime** | **Split Method** | **Accuracy** | **Macro‑F1** | **Top‑2 Acc** | **Limitations** |
| Exp 1 | Single ride (Ride A) | Random frame split | 0.7615 | 0.7604 | 0.9692 | Overfits; not generalisable |
| Exp 2 | Multi‑ride balanced | Group‑by‑ride | 0.6839 | 0.6798 | 0.8744 | Flat and low‑slope classes are weaker |
| Exp 3 | Ride‑level per‑ride | Leave‑rides‑out | 0.4075 | 0.3619 | 0.6028 | Severe class bias; low generalisation |

Moreover, comparing our results to the literature such as Alorf (2024), who reported >85% accuracy in a simplified indoor environment using VGG16/Xception on 1,500 images, our model achieves ~68% accuracy on unseen outdoor rides with real world variability. Nolte et al. (2018) reported up to 92% accuracy for road surface classification using ROI cropping, but under less ambiguous class boundaries. In contrast, our setup tackles subtle slope gradients in highly dynamic scenes, making direct comparison challenging. Nonetheless, the achieved generalisation on unseen rides remains competitive and novel.

# Chapter 6. Conclusions and Further Work

## Summary of Project

This project set out to tackle the problem of visual slope classification for smart mobility, specifically for e bikes and similar platforms. Traditional approaches rely on dedicated sensors to gauge incline, which can be costly and prone to signal issues. By contrast, this work investigated whether a monocular RGB camera fixed on the bicycle’s handlebar could reliably classify the terrain slope, drawing inspiration from how human riders visually perceive hills. Overall, this study worked on the gap of creating a vision-only slope detection method that could support or even replace extra hardware sensors. The core idea was to leverage modern deep learning (using transfer learning on the MobileNetV2 CNN architecture) to identify slope categories from images alone.

To realise this vision-based system, the project followed several key implementation stages. Data collection was conducted by recording real cycling footage with a fixed, forward facing GoPro camera paired with an inclinometer, ensuring each video frame received an accurate slope label from the sensor reading. A pre-processing pipeline then synchronised the video and telemetry streams and prepared the data: each frame was extracted and labelled with the corresponding slope category, and images were cropped to focus on the road and horizon (removing sky) to highlight slope cues.

A balanced dataset of approximately 5,000 labelled frames was constructed from 13 distinct rides through careful class binning and selective oversampling. This helped address class imbalance so that each slope category was well-represented for training. Conservative data augmentation techniques were applied to increase data diversity without distorting the geometric features relevant to slope (for instance, extreme rotations that could alter the apparent incline were avoided). Finally, the prepared data was used in model training: a MobileNetV2-based network (pre-trained on ImageNet) was fine-tuned on the slope classification task. Training was performed on a GPU with an appropriate optimiser and learning rate schedule, and the model’s performance was monitored on a validation split. In summary, the project combined vision-based learning with an efficient CNN architecture to develop a deep learning model that is capable of classifying trail incline from camera footage in a smart mobility context.

## Key Findings and Results

Through a series of experiments, the project evaluated the effectiveness of the proposed approach and yielded several key findings:

### ****Experiment 1 (Baseline – Single Ride, Random Split)****

Using data from a single ride (with training and validation frames randomly split from the same route) produced an initially high performance, with about 76% accuracy and 0.76 macro-F1 on the validation set. This suggested that the model can indeed learn to classify slopes under certain conditions. However, signs of overfitting were evident – the network likely memorised environment-specific cues from that ride (such as particular background or lighting features) rather than general slope features. The artificially high accuracy (and even ~97% top-2 accuracy) in this experiment does not translate to new scenarios, highlighting that a naive training/validation split can inflate performance by leaking contextual information. This finding underscored the importance of evaluating the model on truly unseen rides to gauge real-world generalisability.

### ****Experiment 2 (Multi-Ride Balanced Model)****

This experiment was designed to test generalisation by training on a balanced dataset from 12 rides and validating on a completely held-out 13th ride. As expected, the performance metrics were more modest: roughly 68% accuracy (and a similar ~0.68 macro-F1) on validation frames from unseen rides. While lower than the overfitted baseline, these results are far more indicative of real deployment performance. All five slope classes were recognised to some extent, and the model achieved about 87% top-2 accuracy, meaning it often ranked the correct slope as one of its top two guesses. Importantly, the model proved it could generalise beyond the training routes, establishing a promising baseline for vision-only slope classification under varied conditions. Some classes remained challenging – notably the subtle gradients: flat, low-uphill, and low-downhill were more frequently confused, likely due to their inherently ambiguous visual cues. Nonetheless, Experiment 2 represents the primary success of this project, demonstrating a viable multi-ride model that balances classes and maintains moderate accuracy even on completely new rides. Achieving ~68% accuracy on entirely unseen routes is an encouraging result, suggesting that with further improvements, a camera-based system could be practical for smart e-bike applications. Based on the reviewed studies, no previous work has shown this level of generalisation using only one fixed camera for slope classification in varied outdoor cycling conditions. Therefore, this study represents a significant advancement in vision only slope detection under real world conditions.

### ****Experiment 3 (Fine-Tuning on New Ride)****

The final experiment investigated whether fine-tuning the model on a small amount of data from a new target ride could improve performance on that ride. In practice, this involved pre-training the model on the multi-ride dataset (as in Experiment 2) and then updating the model with a few samples from an unseen ride. The outcome, however, revealed limitations of this approach. The validation accuracy plummeted to around 40% (macro-F1 ≈ 0.36) when evaluated on the remaining frames of the target ride. This drop indicates that the model severely overfitted the limited fine-tuning data, losing its more general feature representation. The fine-tuned model latched onto spurious patterns unique to the fine-tune ride (for example, particular lighting or camera angles in that subset) and thus performed worse overall. In other words, the attempt to specialise the model with minimal new data introduced a class bias and undermined generalisation. This finding reinforces that fine-tuning with insufficient or unbalanced data can be counterproductive. Experiment 3 highlighted the model’s fragility when not enough diverse data is available in the target domain, and it suggests that straightforward transfer learning has limitations here, more sophisticated domain adaptation or larger fine-tuning datasets would be needed to see improvements.

In summary, the results across these experiments show that the initial objectives were met in principle, but with trade-offs. A vision-based CNN can distinguish slope categories (Experiment 1 proved the concept), and when appropriately evaluated across different rides (Experiment 2), it attains reasonable accuracy (~68% on unseen data). However, truly robust generalisation remains challenging (Experiment 3’s outcome), emphasising the need for careful data strategies. Overall, the key takeaway is that the approach is viable – the model performs far better than chance on new trails and provides a baseline for a camera-only slope sensing system, but performance on subtle slope classes and novel environments will require further refinement. These findings also illustrate broader points: evaluation protocol significantly affects perceived performance (highlighting the risk of optimistic results from random splits), and factors like class balancing and augmentation are crucial in training a model that generalises beyond its training experiences.

## Achievement of Objectives

In Chapter 1, several objectives were laid out for this dissertation. Here we reflect on each and assess the level of achievement:

* **Develop a vision-based slope classification system:**

The project successfully designed and implemented a deep learning system that uses only camera input to classify slope. This was realised by training a MobileNetV2-based CNN to recognise the five slope categories from monocular images, thereby meeting the core system development objective. The final model can infer the incline of the trail from a single frame, demonstrating the feasibility of vision-only slope sensing in line with the project’s aims.

* **Collect and label a real-world dataset**:

Data from 13 real-world bike rides (covering various terrains and slopes) were gathered, and approximately 5,000 frames were labelled into the defined classes. The dataset was deliberately balanced across the five slope categories through a combination of selective sampling and augmentation, as described in the methodology. This fulfils the objective of developing a representative real-world dataset for training and evaluating the model.

* **Test generalisability on unseen rides**

The model’s ability to generalise was examined by using ride-level separation between training and testing data (Experiments 2 and 3). The primary evaluation (Exp. 2) showed that the system can maintain around 68% accuracy on completely unseen rides, indicating a baseline level of generalisation to new environments. This demonstrates that the objective of testing on unseen data was addressed. However, the drop in performance compared to same-ride testing (and the difficulties seen in Exp. 3’s fine-tuning attempt) reveals that generalisation is not yet fully solved. The model does generalise to an extent, proving the concept works beyond the training routes, but not to the level one might desire for all conditions. Thus, this objective is only partially met: the work identified both the potential and the current limitations in generalisability, laying groundwork for improvements rather than delivering a definitive solution.

In conclusion, all primary project objectives were addressed, with most being successfully achieved. A working vision-based classifier was built and evaluated on real data, a novel dataset was produced, and the system’s performance on new rides was analysed. The outcomes of the experiments align with the stated goals, confirming that the project delivered a proof-of-concept for camera-based slope classification and provided insights into its strengths and weaknesses with respect to the original objectives.

## Limitations

Although this project showed the potential of vision-based slope classification, several limitations must be noted. These points explain why the results are modest and highlight where improvements are needed.

* **Dataset size and variety:**

The model was trained on a small dataset of about 13 rides with roughly 5,000 training frames. While these rides covered some different terrains, the variety of conditions was limited. There were no recordings from different seasons, cities, or landscapes. A dataset of this size can cause the model to overfit to the specific trails it was trained on and makes it less reliable when facing new environments.

* **Lighting and camera setup:**

Most of the data was collected in daytime under fairly similar conditions, using a single forward-facing camera fixed to the bike. The footage did not include extreme lighting, night rides, or big changes in camera height and angle. Because of this, the model may not adapt well when deployed in conditions that differ from those in training. Bright backlighting, shadows, or unusual viewpoints could make slope recognition less accurate.

* **Confusion near class boundaries:**

The model often struggled when slopes were close to flat or only gently inclined. Distinguishing a very small uphill from flat ground is difficult even for humans, and the model showed the same weakness. The confusion matrices in Chapter 5 confirmed that adjacent classes, especially flat, low-uphill, and low-downhill, were often mixed up. Label noise from inclinometer alignment and motion blur from the bike likely added to these errors. As a result, the model performs best on steep slopes but less reliably on near-flat ones.

* **Generalisation and comparison to prior work:**

The findings should be applied with caution. The model works reasonably well under the tested conditions but does not yet cover all real-world cases. There was also no on-device testing, and the heavy use of augmentation only partly compensated for the limited data. Many past studies have shown higher accuracy by using large, curated datasets, multiple sensors, or controlled environments (for example, Zhao et al., 2024; Nolte et al., 2018). While such approaches give strong results, they are harder to deploy outside the lab. In contrast, this project used only a single fixed camera in uncontrolled outdoor environments. This choice explains the lower accuracy but makes the system more realistic for future real-world use.

## Future Work

Building on the findings and limitations of this research, there are several avenues for future work to enhance the visual slope classification system.

* **Expand and Diversify the Dataset:**

A clear, immediate step is to collect more comprehensive riding data. Future data collection should span a wider range of environments, including different trail types (paved roads, gravel paths, steep mountain trails), various times of day (morning, noon, dusk, night), and weather conditions (sunny, overcast, rainy). Increasing the number of rides and ensuring more diverse conditions would help the model learn more invariant features and reduce overfitting to any particular context. A larger dataset would also allow training more complex models or applying more rigorous validation splits (e.g. multiple unseen-route tests) with greater confidence in the results.

* **Experiment with Alternative Lightweight CNNs:**

While MobileNetV2 proved effective, exploring other modern efficient CNN architectures could yield better performance or robustness (Sandler et al., 2018). Models such as EfficientNet (known for its compound scaling efficiency) or newer compact vision transformers (e.g. TinyViT) might offer improved accuracy at similar or lower computational cost. Conducting ablation studies with different backbone networks could identify an architecture that is better suited to the nuances of slope imagery. The goal would be to maintain real-time inference capability on embedded hardware while possibly boosting classification accuracy or generalisation.

* **Enhance Label Precision with Sensor Fusion:**

The current dataset labels come from an inclinometer (IMU) and are treated as ground truth for each frame, but there may be noise or lag in those measurements. A short-term improvement is to apply temporal smoothing or filtering to the slope labels (for example, using a moving average or Kalman filter on the IMU readings) so that abrupt label fluctuations due to noise are minimised. Additionally, future work could incorporate the IMU data directly at inference time. Even if the primary aim is a vision-only system, a pragmatic approach for improvement is a lightweight sensor fusion, where the vision model’s prediction could be moderated by real-time inclinometer readings to improve overall reliability. For instance, if the vision model is uncertain between flat and low-uphill, the additional cue from a smoothed IMU gradient reading might help disambiguate the class. Integrating such multi-modal input is a logical next step to elevate performance on borderline cases and reduce misclassifications.

* **Integration into Smart E-Bike Systems:**

The ultimate application of this research is to enhance smart mobility features on e-bikes. A long-term direction is to integrate the slope classifier into smart shifting or power assist systems. For example, an e-bike could automatically adjust its electric assist level or gear selection in anticipation of upcoming hills, using the vision model’s real-time slope prediction. This would result in smoother rides and potentially improved energy efficiency, as the bike could proactively respond to terrain changes. Such integration would require collaboration with e-bike control system design, ensuring the predictions trigger appropriate and safe responses in the bike’s mechanisms. It also opens up opportunities to evaluate how much user benefit this vision-driven feature provides. For example, does it reduce the rider’s effort noticeably on inclines, and is the system’s behaviour intuitive to the rider?

Additionally, if this study were to be repeated, several enhancements would be introduced. First, a larger and more diverse dataset (that is labelled correctly and have equal number of frames per class) would be collected to better represent various terrain types and lighting conditions. Second, temporal modelling (e.g., CNN + LSTM) would be incorporated to smooth predictions across frame sequences. Third, Grad CAM interpretability analysis could help refine cropping strategies. Additionally, minimal sensor fusion (e.g., IMU integration) might reduce class confusion in near flat scenarios. These changes are expected to improve classification robustness, especially in ambiguous slope conditions, and potentially increase macro F1 beyond 75%.

In summary, there is a rich landscape of follow-up work, from straightforward improvements in data and models, to ambitious multi-sensor systems and on-bike deployments. Each proposed future direction aims to push the performance, reliability, or applicability of vision-based slope classification further, ultimately moving closer to a robust system that could be embraced in smart mobility solutions.

# Reflective Evaluation and Project Timeline

## Reflective Evaluation

This project was initially proposed by my supervisor and presented a clear yet challenging goal: to create a real-time, vision-only slope classification system for e-bikes using deep learning. From the start, the project demanded careful planning and consistent iteration. Although I was already familiar with machine learning, this project required combining multiple skills: data preprocessing, deep model customisation, evaluation strategy design, and formal academic writing.

The first phase focused on reviewing literature, where I identified 20 relevant papers that shaped both the theoretical and practical dimensions of this study. By the end of June, I had a solid understanding of the landscape and the gaps that motivated our approach. After receiving the dataset in early July, I spent considerable time preparing it for modeling. This involved extracting video frames, synchronising them with inclinometer readings from telemetry, filtering invalid samples, and creating a balanced dataset across five slope classes. Dataset imbalance and low-quality segments introduced several complications, which I addressed through augmentation and iterative preprocessing.

Model training was done in two phases’ warm-up and fine-tuning, using MobileNetV2. I conducted three core experiments to evaluate generalisability, ride-specific behavior, and overfitting risks. A significant amount of effort went into debugging training runs, adjusting class distributions, and interpreting validation performance. Over time, I developed a deeper understanding of how small architectural or data changes affect real-world performance.

In the final weeks, I focused on documentation, report writing, citation refinement, and cross-checking experimental outputs. Looking back, this project helped me develop not only technical skills in deep learning and computer vision but also soft skills like project management, experimentation discipline, and academic integrity. Completing this research over three months taught me how to deal with ambiguity, time pressure, and constant iteration and, gave me a realistic sense of what it means to build a practical AI system from scratch.

## Project Gant Chart & Timeline Overview

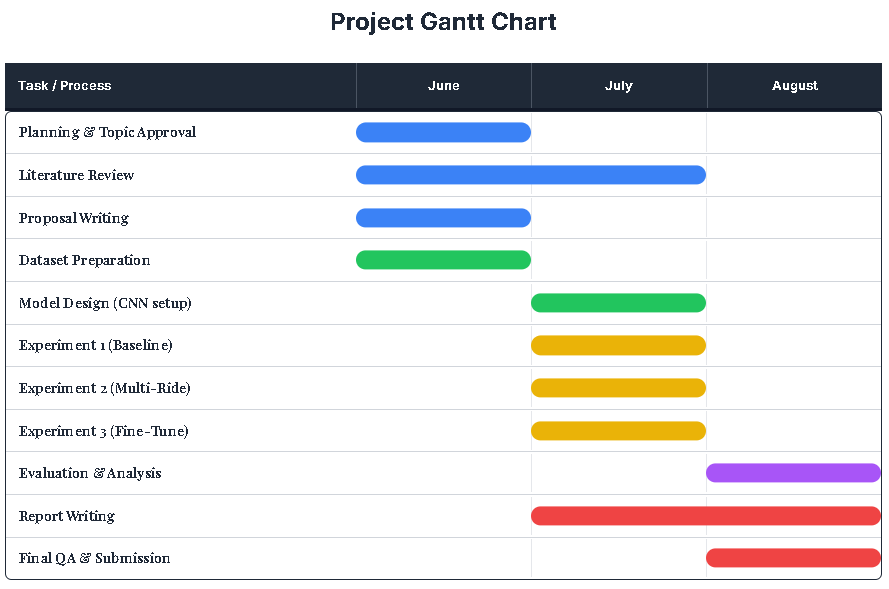


Figure 22: Project Gantt chat

|  |  |  |
| --- | --- | --- |
| Week | Date Range | Milestone / Activities |
| Week 1 | May 31 – June 6 | Topic approved by supervisor. Initial brainstorming and background reading started. |
| Week 2–3 | June 7 – June 20 | Conducted structured literature review. Identified 20 relevant papers and key gaps. |
| Week 4 | June 21 – June 27 | Drafted and submitted project proposal with aims, objectives, and proposed methodology. |
| Week 5 | June 28 – July 4 | Received e-bike dataset (videos + telemetry). Planned preprocessing pipeline. |
| Week 6 | July 5 – July 11 | Extracted frames, synced them with AngleY data, and labeled slope bins (5 classes). |
| Week 7 | July 12 – July 18 | Cleaned and balanced dataset. Removed invalid frames and filled class gaps via augmentation. |
| Week 8 | July 19 – July 25 | Defined CNN model architecture with MobileNetV2. Implemented warm-up and fine-tuning phases. |
| Week 9 | July 26 – Aug 1 | Ran Experiment 1 (Single-Ride Baseline). Used random frame split from a single ride with minimal augmentation and basic preprocessing. Evaluated model’s tendency to memorize due to lack of ride diversity. |
| Week 10 | Aug 2 – Aug 8 | Ran Experiment 2 (Multi-Ride Generalization). Applied MobileNetV2 fine-tuning, ride-based frame sampling, class binning, label smoothing, dropout, and data augmentation. Evaluated generalisation using full performance metrics. |
| Week 11 | Aug 9 – Aug 15 | Ran Experiment 3 (Ride-Specific Fine-Tuning). Incorporated ride-cropping, CLAHE, learning rate warm-up, aggressive balancing, and shorter training to improve personalisation and assess overfitting. |
| Week 12 | Aug 16 – Aug 22 | Cross-validated results. Generated confusion matrices, class-wise metrics, and graphs. |
| Week 13 | Aug 23 – Aug 31 | Completed report writing, referencing, formatting and final QA. |
| Submission Week | Sept 5 | Dissertation submitted. Reflection and timeline added on supervisor’s recommendation. |

# References

Alorf, A., 2024. A Robust Deep Learning Model for Terrain Slope Estimation. *International Journal of Advanced Computer Science & Applications*, *15*(1).

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K. and Fei-Fei, L., 2009, June. Imagenet: A large-scale hierarchical image database. In the *2009 IEEE conference on computer vision and pattern recognition* (pp. 248-255). Ieee.

Ghorbanzadeh, O., Meena, S.R., Blaschke, T. and Aryal, J., 2019. UAV-based slope failure detection using deep-learning convolutional neural networks. *Remote Sensing*, *11*(17), p.2046.

Hardt, M., 2017. Deep learning requires rethinking generalisation. *ICLR’17*.

Hendrycks, D., Lee, K. and Mazeika, M., 2019, May. Using pre-training can improve model robustness and uncertainty. In *International conference on machine learning* (pp. 2712-2721). PMLR.

Jiang, X. and Ge, Z., 2020. Data augmentation classifier for imbalanced fault classification. *IEEE Transactions on Automation Science and Engineering*, *18*(3), pp.1206-1217.

Karypidis, E., Mouslech, S.G., Skoulariki, K. and Gazis, A., 2022. Comparison analysis of traditional machine learning and deep learning techniques for data and image classification. *arXiv preprint arXiv:2204.05983*.

Khan, A., Sohail, A., Zahoora, U. and Qureshi, A.S., 2020. A survey of the recent architectures of deep convolutional neural networks. *Artificial intelligence review*, *53*(8), pp.5455-5516.

Liu, F., Lu, Z. and Lin, X., 2025. Vision-based environmental perception for autonomous driving. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, *239*(1), pp.39-69.

Ma, N. and Yao, Z., 2024. Analysis of slope stochastic fields using a novel deep learning model with attention mechanism. *Frontiers in Earth Science*, *12*, p.1352958.

Maurício, J., Domingues, I. and Bernardino, J., 2023. Comparing vision transformers and convolutional neural networks for image classification: A literature review. *Applied Sciences*, *13*(9), p.5521.

Meng, X., Hatch, N., Lambert, A., Li, A., Wagener, N., Schmittle, M., Lee, J., Yuan, W., Chen, Z., Deng, S. and Okopal, G., 2023. Terrainnet: Visual modeling of complex terrain for high-speed, off-road navigation. *arXiv preprint arXiv:2303.15771*.

Nolte, M., Kister, N. and Maurer, M., 2018, November. Assessment of deep convolutional neural networks for road surface classification. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (pp. 381-386). IEEE.

Pan, S.J. and Yang, Q., 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), pp.1345-1359.

Quddus, M.A., Ochieng, W.Y. and Noland, R.B., 2007. Current map-matching algorithms for transport applications: State-of-the art and future research directions. *Transportation research part c: Emerging technologies*, *15*(5), pp.312-328.

Ren, S. and Zhang, Y., 2024, December. Terrain Image Classification Based on Vision Transformer Deep Learning Algorithm. In *2024 7th International Conference on Algorithms, Computing and Artificial Intelligence (ACAI)* (pp. 1-5). IEEE.

Rodríguez-Martínez, E.A., Flores-Fuentes, W., Achakir, F., Sergiyenko, O. and Murrieta-Rico, F.N., 2025. Vision-based navigation and perception for autonomous robots: Sensors, SLAM, control strategies, and cross-domain applications—A review. *Eng*, *6*(7), p.153.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L.C., 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4510-4520).

Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. and Batra, D., 2017. Grad-cam: Visual explanations from deep networks via gradient-based localisation. In *Proceedings of the IEEE international conference on computer vision* (pp. 618-626).

Shorten, C. and Khoshgoftaar, T.M., 2019. A survey on image data augmentation for deep learning. *Journal of big data*, *6*(1), pp.1-48.

Suneetha, M., Sujitha, A.V.S.P., Shameera, M. and Kalyan, M.R., 2024, June. Object Based Terrain Classification Using Deep Learning Techniques. In *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)* (pp. 1-5). IEEE.

Ustunel, E. and Masazade, E., 2019. Vision‐based road slope estimation methods using road lines or local features from instant images. *IET Intelligent Transport Systems*, *13*(10), pp.1590-1602.

Vulpi, F., Milella, A., Marani, R. and Reina, G., 2021. Recurrent and convolutional neural networks for deep terrain classification by autonomous robots. *Journal of Terramechanics*, *96*, pp.119-131.

Zhao, T., Yang, L., Xie, Y., Ding, M., Tomizuka, M. and Wei, Y., 2024. Roadbev: Road surface reconstruction in bird’s eye view. *IEEE Transactions on Intelligent Transportation Systems*.

# Appendices

## Appendix 1: Ethics Form

**Division of**

**Computer Science & Informatics**

**School of Engineering**

**CHECKLIST FOR ETHICS REVIEWS**

|  |  |
| --- | --- |
| Project Supervisor: | **Ali Salimian** |
| Project Title: | Deep Learning-Based Visual Slope Classification for Navigation Systems |
| Student Name: | Asma Shirin |
| Student Number: | 4329971 |

In the planning and design of your project ethical issues must always be considered. If your project does not involve testing and/or evaluating software with end users, or any other contact with people, then the general code of ethics for computing will apply.

If your project does involve contact with people then the issues outlined in this form and expressed in detail in the ACM code of ethics must be considered.

For all projects you must complete this form, discuss it with your supervisor and have it signed off. You may also be required to produce more details.

When the form is complete and signed off your supervisor will pass it onto the departmental ethics committee. You do not have ethical approval to proceed until you have received a reply from the committee via the supervisor.

If your project plan changes then ethical approval may need to be reviewed.

|  |  |
| --- | --- |
| Tick this box to confirm that you have read the ACM Code of Ethics and Professional Conduct.  <https://ethics.acm.org/> |  |
| Tick this box if your project does not involve any contact with people.  (If so **DO NOT** complete the rest of the form.Q1-Q14) |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Student Name: | Signature | Date |  |
| **Asma Shirin** |  | 27-June-2025 |
| Supervisor Name: | Signature | Date |
| **Ali Salimian** | **Ali Salimian** | 27-June-2025 |
| Ethics Approval: | Signature | Date |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Yes** | **No** |
| **1** | Will the study cause psychological stress or anxiety, cause harm or have any other negative consequences (beyond the risk encountered in their normal lifestyles)? |  |  |
| **2** | Will the real reasons for the study be withheld from the participants at any time? |  |  |
| **3** | Will the study involve prolonged or repetitive testing? |  |  |
| **4** | Will the participants be offered money, or other incentives, to participate? |  |  |
| **5** | Will it be possible for participants to be identified and/or for data to be identified as coming from a known person? |  |  |
| **6** | Will the study need to use children or other people with particular characteristics? |  |  |
| **7** | Does the study involve gathering personal data? |  |  |
| **8** | Does the study involve the NHS? |  |  |

**If you have answered YES to any of these questions you will need to provide further details after consulting with your supervisor.**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Yes** | **No** |
| **9** | Will the participants be able to withdraw from the study at any time and will they be told this? |  |  |
| **10** | Will the participants be told the purpose of the research before they agree to take part? |  |  |
| **11** | Will the participants be told that the data will be collected anonymously, stored securely, not given to anyone else, and destroyed when no longer needed before they agree to take part? |  |  |
| **12** | Will the participants be told that they will suffer no stress or harm before they agree to take part? |  |  |
| **13** | Will the participants be told that they will be able to request a copy or a summary of the results and analysis of the study, including contact details, before they agree to take part? |  |  |

**If you have answered NO to any of these questions you will need to provide further details after consulting with your supervisor.**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Yes** | **No** |
| **14** | Do you intend to use a questionnaire? |  |  |

## Appendix 2: Draft Dissertation Assessment Form

*First Name:* Asma

*Family Name:* Shirin

*Course Title: MSc Dissertation (CSI-7-PRO)*

*Student No:* 4329971

*Dissertation Title: Deep Learning-Based Visual Slope Classification for MTB Track Inclinometry*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Check Item** | **unacceptable** | **Marginally**  **acceptable** | **Acceptable** | **Excellent** |
| Final Objectives |  |  |  |  |
| Planning and Interim Objectives |  |  |  |  |
| Appreciation of possible difficulties and strategies for solution |  |  |  |  |
| Literature Search: Scope, effectiveness |  |  |  |  |
| Generation of ideas and originality |  |  |  |  |
| Execution and Implementation |  |  |  |  |
| Achievements and conclusions |  |  |  |  |
| Organisation and presentation |  |  |  |  |
| Clarity and Accuracy |  |  |  |  |
| Technical evaluation of the dissertation |  |  |  |  |

Please give your comment and suggestions regarding the dissertation. Particular attention should be given to details that guide possible revision

**Supervisor: Signature: Date:**

## Appendix 3: MSc Dissertation Marking Criteria

**MSc Dissertation (CSI-7-PRO) Marking Criteria**

**The MSc Dissertation (project) can take one of two forms:**

* Inquiry-based (applied-computing) dissertation, or
* Research-based (theoretical) dissertation

**Inquiry-based (applied-computing) dissertation**

Dissertations can be produced for inquiry-based projects in the context of applied-computing, in practical areas such as: Internet-Mobile Apps, Cloud Computing; Systems and Cyber-Security or Data Science. Applied computing projects typically implement a piece of software. The dissertation is a record of your work over the course of the project. It should detail (at an appropriate level) what was the purpose of the project, what was achieved, what software was designed/developed, what hypothesis was being tested (if applicable), experiments performed, data gathered, metrics used etc.

**Research-based (theoretical) dissertation**

Dissertations can be produced for projects involving academic research in theoretical areas: Internet- Mobile Apps, Cloud Computing; Systems and Cyber-Security or Data Science, for the development, exploration and evaluation of algorithms to support such technologies. However the project must meet the requirement that in a research context applied computing principles must be demonstrated, so the project must include: clear evidence of practical modelling or simulation. Accordingly the project must employ suitable tools e.g. MATLAB/Simulink, GNU-Octave, Scilab, FreeMat, etc… for the modelling or simulation work undertaken. The project must also include evidence of gathering of empirical data, development/testing of algorithms/models, critical evaluation of possible alternative algorithms/models, evaluation of processes and outputs using clearly established metrics.

Regardless of the nature of the dissertation selected, the project is required to be practically and theoretically robust also methodologically sound to meet the academic learning outcomes of this module.

**Assessment & the viva**

Equivalence in the assessment of each type of project is maintained.

The dissertation will be marked independently by the supervisor and a second marker according to set criteria. The student will be invited to attend a combined presentation and viva, lasting about 20 minutes where the first 10 minutes will entail a presentation of the work with the remaining time used for questions from the panel. The panel will include the supervisor and second marker, usually with a third member chairing the session.

|  |  |  |
| --- | --- | --- |
| **Component** | **Weighting** | **Threshold Mark** |
| Dissertation Report | 80% | 40 |
| Viva | 20% | 40 |

To pass the dissertation module, the student needs to achieve a final (pass) mark of 50% or more. The dissertation must be passed without compensation.

**PROJECT ASSESSMENT FORM**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | ***MSc Dissertation (CSI\_7\_PRO) Marking Scheme*** | |  | |
| ***STUDENT ID*** | | 4329971 | | | |
| ***STUDENT NAME*** | | Asma Shirin | | | |
| ***DISSERTATION TITLE*** | | Deep Learning-Based Visual Slope Classification for MTB Track Inclinometry | | | |
| ***SUPERVISOR*** | | Ali Salimian | | | |
|  | | | | | |
| **Item** | | **Comments** | | **Marks** | |
| ***PROJECT MANAGEMENT*** | | 1st supervisor only | |  | |
| Project objectives, | |  | | ***/4*** | |
| Gantt chart: Planning, interim objectives and milestones | |  | | ***/4*** | |
| Risk assessment and mitigating strategies | |  | | ***/4*** | |
| Resource requirements | |  | | ***/4*** | |
| Execution | |  | | ***/4*** | |
| ***DISSERTATION*** | | 1st & 2nd supervisor | |  | |
| Literature/Technical review including: Legal, Social, Ethical and Professional issues (LSEPI) | |  | | ***/10*** | |
| Solution analysis/design OR  Alternative designs /Final algorithm | |  | | ***/10*** | |
| Implementation | |  | | ***/10*** | |
| Results, testing and evaluation OR  Experimental and theoretical results | |  | | ***/10*** | |
| Future work | |  | | ***/10*** | |
| Conclusions/Reflection on Learning | |  | | ***/10*** | |
|  | Subtotal | | ***/80*** | |  |
| ***VIVA-VOCE*** | | 1st & 2nd supervisor | |  | |
| Organisation and presentation clarity | |  | | ***/5*** | |
| Relevance and accuracy | |  | | ***/5*** | |
| Execution and timing | |  | | ***/5*** | |
| Responses and technical insight | |  | | ***/5*** | |
|  | | Subtotal | | ***/20*** | |
| ***TOTAL MARK*** | |  | | ***/100*** | |

**Project Marking Rubric Template**

Listed below is the rubric marking sheet relating to the standard of the project. Please rate the various criteria from 0 to 10 (details below), to build up a picture of the strengths and weaknesses of the project. Based on the template, weighting and your academic judgement, please decide on an overall mark for the project.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | ***MSc Dissertation (CSI\_7\_PRO) Marking Scheme*** | | |  |  |
| ***STUDENT ID*** |  | | | | |  |
| ***STUDENT NAME*** |  | | | | |  |
| ***DISSERTATION TITLE*** |  | | | | |  |
| ***SUPERVISOR*** |  | | | | |  |
|  | ***fail: irredeemable*** | ***fail: marginal*** | ***pass: good/merit*** | ***pass: very good/distinction*** | ***pass: excellent/distinction*** | ***Supervisors Mark out of 10*** |
| ***Mark out of 10*** | ***"0-2"*** | ***"3-4"*** | ***"5-6"*** | ***"7-8"*** | ***"9-10"*** |
| ***DISSERTATION PROJECT***  ***MANAGEMENT 20%*** | ***1st Supervisor only, (each section equally weighted: mark x 0.4)*** | | | | |
| **Project aims, objectives & problem formulation** | Non submission or very w eak aims/objectives & significant problem formulation issues and interpretation errors | Some aims/objectives clarity errors and/or poor problem  formulation interpretation errors | Basic aims/objectives clear and correct, reasonable problem  formulation; adequate evidence of alternative considerations | Aims/objectives clear & correct, reasonable problem formulation interpretation, good evidence of alternative considerations | Excellent in all respects. Aims/Objectives outstanding. Consideration of alternatives critically evaluative, accurate and  perceptive |  |
| **Gantt chart: Planning, interim objectives and milestones** | Non submission or very w eak & significant  planning & chart related issues and interpretation errors | Some clarity errors and/or poor planning/chart errors | Project plan is basic and acceptable; reasonable planning/chart; adequate evidence of alternative  considerations | Planning clear/correct, reasonable planning/chart, good evidence of alternative considerations | Excellent in all respects. Planning/chart outstanding. Consideration of alternatives critically evaluative, accurate and  perceptive |  |
| **Risk assessment and mitigating strategies** | Non submission or very w eak & significant  risk identification/mitigation issues and interpretation errors | Some clarity errors and/or poor risk identification/mitigation errors | Risk assessment clear & correct; reasonable risk identification and mitigation; adequate evidence of  alternative considerations | Risk assessment clear/correct, reasonable risk  identification/mitigation, good  evidence of alternative considerations | Excellent in all respects. risk assessment/mitigation outstanding.  Consideration of alternatives  critically evaluative, accurate and perceptive |  |
| **Resource requirements** | Non submission or very w eak & significant resource  identification/justification issues and interpretation errors | Some clarity errors and/or poor resource  identification/justification errors | Resource requirements basically correct; clear resource identification ; adequate evidence of  alternative considerations | Resource requirements clear/correct; reasonable resource identification &  justification, good evidence of  alternative considerations | Excellent in all respects; resource requirements outstanding.  Consideration of alternatives critically evaluative, accurate and  perceptive |  |
| **Execution** | Non attempt or very w eak & significant project execution/professionalism issues and basic decision  making errors | Some clarity or decision making errors and/or some poor project execution/professionalism errors | Reasonably good project execution & professionalism; adequate evidence of alternative considerations | Project execution & professionalism w as good; clear evidence of alternative  considerations & some  innovation. | Excellent in all respects; Project execution & professionalism outstanding; very innovative, critically evaluative, accurate and  perceptive |  |
| ***DISSERTATION REPORT 60%*** | ***1st & 2nd Supervisors, (each section equally weighted: mark x 1 )*** | | | | |  |
| **Literature/Technical review including : Legal, Social, Ethical & Professional issues (LSEPI)** | Non attempt or very w eak linkage & significant research content or relevance issues; no consideration of LSEPI | Some linkage / research content & relevance errors; very little consideration of LSEPI | Reasonably good linkage / research content; sources are clear & relevant; adequate evidence of LSEPI consideration | Good Research content & linkage; sources relevant; clear narrative evidence of LSEPI considerations | Excellent in all respects; Research content & sources outstanding; LSEPI narrative critically evaluative and accurate |  |
| **Solution analysis/design or**  **Alternative designs /Final algorithm** | Non attempt or very w eak & significant solution analysis/design issues and basic decision making errors | Some clarity or decision making errors and/or some poor solution analysis/design errors | Reasonably good solution analysis/design attempts; minor evidence of alternative considerations | Solution analysis/design w as very good; clear narrative evidence of alternative considerations & some  innovation | Excellent in all respects; solution analysis/design outstanding; very innovative, narrative critically evaluative, accurate and perceptive |  |
| **Implementation** | Non attempt or major solution implementation & coding issues; basic decision making errors | Some minor solution implementation & coding errors and/or minor simulation/modelling  configuration errors | Reasonably good solution implementation & coding; simulation/modelling  configuration; some evidence of  alternative considerations | Very good solution implementation & coding or good simulation & modelling  configuration; clear evidence of  alternative considerations; | Excellent in all respects; solution implementation outstanding; very innovative; narrative critically evaluative, accurate and perceptive |  |
| **Results, testing and evaluation or**  **Experimental and theoretical results** | Non attempt or major testing & evaluation issues; no clear quality metrics defined; no results produced; basic  decision making errors | Some minor testing & evaluation issues; problems w ith quality metrics used; minimal, poor & innacurrate results; minor  decision making errors | Reasonably good testing & evaluation results; adequate quality metrics used; some evidence of basic evaluation &  summary interpretation | Very good testing & evaluation results; clear & appropriate quality metrics; discussion of evaluation & alternative  considerations; some innovation | Excellent in all respects; results, testing and evaluation outstanding; very innovative; narrative critically evaluative, accurate and perceptive |  |
| **Future work** | Non attempt or major future w ork discussion issues; basic  decision making errors | Some minor future w ork discussion issues; minor decision making errors | Reasonably good future w ork discussion; some relevant evaluation & summary interpretation | Very good future w ork discussion; clear & appropriate discussion of outcomes future alternative considerations;  some innovation | Excellent in all respects; future w ork discussion outstanding; very  innovative; narrative critically evaluative, accurate and perceptive |  |
| **Conclusions/Reflection on Learning** | Non attempt or major conclusion/reflection issues; basic decision making errors; no learning identified. | Some minor conclusion/reflection discussion issues; minor decision making  errors ; minimal learning  identified | Reasonably good conclusion/reflection discussion; relevant learning evaluation & summary  interpretation | Very good conclusion/reflection discussion; clear appropriate discussion of learning & alternative considerations;  some innovation/suggestions | Excellent in all respects; conclusion/reflection discussion outstanding; very innovative; narrative critically evaluative,  accurate and perceptive |  |
|  | ***DISSERTATION***  ***SUB-TOTAL out of 80%*** | | | | |  |
| ***VIVA-VOCE 20%*** | ***1st & 2nd Supervisors (each section equally weighted: mark x 0.5)*** | | | | |
| **Organisation & presentation clarity** | Non attendance or very w eak & significant  clarity issues and presentation errors | Some organisation & presentation clarity  errors and/or poor interpretation errors | Basic organisation & presentation clear and correct; reasonable interpretation minor evidence of alternative  considerations | Organisation & presentation objectives clear/correct, reasonable interpretation, good evidence of alternative  considerations | Excellent in all respects. Organisation &  presentation outstanding. Critically evaluative, accurate and perceptive |  |
| **Relevance and accuracy** | Non attendance or very w eak know ledge; relevance and accuracy errors | Some relevance and accuracy consistency  errors and/or poor know ledge interpretation errors | Basic relevance and accuracy is good and correct; reasonable know ledge interpretation; some evidence of  alternative considerations | Relevance and accuracy very clear & correct; reasonable interpretation of know ledge; good evidence of alternative  considerations | Excellent in all  respects. Relevance and accuracy outstanding. Critically evaluative, accurate and perceptive |  |
| **Execution and timing** | Non attendance or very poor execution and timing issues; lack of preparation evident | Some minor execution and timing errors and/or poor execution errors | Basic execution and timing is good ; Some good eye contact, reasonably confident  performance | Very good execution and timing  ; Maintained good eye contact, Very confident rehearsed performance | Excellent in all respects. Execution and timing  outstanding. Excellent eye contact; performance timely, accurate &  responsive |  |
| **Question responses and technical insight** | Non attendance or very poor responses and technical insight; lack of preparation evident | Some minor response and technical insight errors and/or poor & hesitant responses | Basic responses and technical insight w as good; reasonable confidence and adequte  know ledge | Very good responses and technical insight; clearly  confident, depth of know eldge clearly evident | Excellent in all respects. Responses and technical insight outstanding. Depth of know ledge: outstanding,  focussed & accurate |  |
|  | ***VIVA-VOCE***  ***SUB-TOTAL out of 20%*** | | | | |  |
|  | ADDITIONAL COMMENT: (Plese add justification comments here, if the total mark is a fail < 50 or a distinction >=70) | | | | | ***TOTAL MARK:***  ***/100%*** |

**Marking Guidelines for Criteria Interpretation**

|  |  |
| --- | --- |
| ***DISSERTATION Project Management*** | 1st supervisor only |
| Project objectives, and problem formulation | The problem domain is clearly highlighted/contextualised and draws on detailed background research to develop a coherent case for the project. The aim and objectives of the project are  clearly defined, justified and appropriate in scope and for the context profiled. |
| Gantt chart: Planning, interim objectives and milestones | Methodology must be clearly identified. The project should be planned and managed via a well defined project plan , consisting of work packages, scheduling details and a GANTT chart with key milestones and deliverables identified. |
| Risk assessment and mitigating strategies | To precisely identify and describe the real threats to project success. Apply a simple but effective classification scheme is to arrange risks according to the areas of impact. The risk management plan outlines the response/mitigation that will be taken for each risk—if it  materializes |
| Resource requirements | if the project involves producing software/algorithms or there any other products the project depends on using any specialist or uncommon hardware/software such as specialised  subroutine packages or a more obscure or specialised programming language, you should describe them briefly and discuss whatever features are relevant to your project. |
| Execution | How well was the project executed, How well was the student organised. How proactive/reactive was the student. How technically able/innovative/original was the student when faced with problem solving situations. What level/standards of professionalism / ethical  behaviour did the student demonstrate during the execution of the project |
| ***DISSERTATION report*** | 1st & 2nd supervisor |
| Literature/Technical review including : Legal, Social, Ethical and Professional issues | Demonstrates the level of research into the subject area before engaging into the problem domain. It should reflect the history, philosophy and rationale behind the dissertation subject area. It should give an overview of techniques that exist and are being researched. With indicative key reference sources. It should also build a rationale on why the work is important, and how to address the relevant Legal, Social, Ethical and Professional issues  associated with the project |
| Solution analysis/design OR  Alternative designs /Final algorithm | A solution for the specific problem area must be proposed. The solution should be independent of the actual means adopted to solve the problem. For example, for software projects, UML designs, structure charts and pseudo-code can be used for a language- independent description. Previously conducted research, instrumental in the problem solution,  should also be referenced in this section. |
| Implementation | A clear description of the translation from algorithm to implementation in a particular language/application is essential. This description must also include details of why a particular type of implementation has been adopted. Small bespoke examples of actual program code or lines of algorithm syntax/processes may be used to demonstrate key  functionality |
| Results, testing and evaluation OR  Experimental and theoretical results | Evidence of project-artefact performance in accordance with its specification, or if not, why not. It is not sufficient just to demonstrate the operation of software or an item of equipment to a supervisor and assume that this will be taken for granted in the final assessment. A description of the test procedures must be included and the results logged in an appropriate manner. Where possible appropriate metrics should be identified and used to assess quality of the results obtained. Large tables of results should appear in an appendix, with only  selected examples appearing in the text. |
| Future work | This section should include some original thoughts as to the direction the dissertation might have taken if more time/resources had been available. If a dissertation has failed to achieve  the original aims and valid reasons explaining this are clearly stated, then the dissertation could be considered successful. |
| Conclusions/Reflection on Learning | The conclusion is both a critical evaluation and a reflective statement as to what extent the original aims and objectives of the dissertation have been achieved; it is not a summary of the various chapters of the report. Supporting evidence should be included as to the success, or even failure, of a dissertation to achieve the original aims, usually in the form of references to  the implementation/testing results or project manage relative to the aims and objectives. |
| ***VIVA-VOCE*** | 1st & 2nd supervisor |
| Organisation and presentation clarity | Given the project focus, a professional presentation typified by a mature, succinct and formal presenting style, and a clear, consistent and quality presentation are of particular importance |
| Relevance and accuracy | The presentation must demonstrate engagement with core concepts and critical exploration of advanced concepts, development technologies and contemporary debates. The focus  adopted and choices made are explained/ justified / appropriate for the problem investigated |
| Execution and timing | In a combined presentation and viva, lasting about 20 minutes the first 10 minutes will entail a presentation of the work with the remaining time used for questions from the panel.  Ideally the student will (in a timely manner) demonstrate In depth knowledge and a thorough understanding of all aspects which allows questions to be answered accurately and fluently  and the discussion to be extended with confidence into difficult or unfamiliar areas. |
| Responses and technical insight | Discussion on what the student learned from the conduct of the project such as the theoretical / practical knowledge gained, mental acuteness, business acumen, latent skills brought to light, skills required, skills developed and how such learning and development  were achieved. The student may cite specific instances such as examples of innovations or even mistakes made and the resultant learning and insights gained |

**MARK ALLOCATION GUIDELINES FOR THE OVERALL PROJECT:**

**0 – 29 Fail**

Fails to achieve reflexive learning. The work is weak, superficial, and poorly conceptualised. Fails to engage with the essence of the practical context (including problem highlighted) and theoretical constructs, and reflects limited reading. The solution and recommendations proposed are limited, inappropriate, and impracticable. The reflection elements are basic, superficial and limited in scope. The work is poorly presented and lacks professionalism.

#### 30 – 49 Marginal Fail

Shows limited evidence of reflexive learning. The work is basic, superficial in parts, and inadequately conceptualised, designed, investigated and analysed. It reflects a poor grasp of, or engagement with, the practical context and theoretical constructs, and demonstrates an over-reliance on basic texts. The solution and recommendations proposed are limited, basic, and impracticable in parts. The reflection elements are superficial and limited in scope. The work is poorly presented and lacks professionalism.

#### 50 – 59 Pass

Achieves adequate reflexive learning. The work provides evidence of satisfactory understanding, conceptualisation, methodological design and investigation of the research question. The content reflects a satisfactory understanding of the essence of the practical context and problem area. The work is referenced to relevant material of acceptable quality and shows adequate understanding of most salient issues, but is limited in terms of relevant wider issues. The solution and recommendations proposed are valid, practicable, adequately sophisticated and reflects some consideration of the wider implications. The reflection elements are adequate in scope and depth, and the content is valid. The work is presented in a professional manner.

#### 60 – 69 Good (Merit)

Achieves a reasonably high level of reflexive learning. The work shows good understanding, conceptualisation and investigation of the research question. This is reflected in the scope and depth of the underpinning literature, selection of core theoretical concepts, design of the methodological framework and the sophistication and practicability of the solution and recommendations proposed. Clear linkage and connectivity between the salient elements of the project report is evidenced. The reflection elements are well scoped, valid, mature, insightful and detailed. The work is well presented reflecting care and attention to detail and maturity of thought.

#### 70 – 89 Very Good (Distinction)

Achieves a sophisticated level of reflexive learning. A sound understanding of the research problem is evidenced in all aspects of the project report and in the connectivity and flow between the various salient elements. The research problem is well conceptualised and defended. The methodological framework is appropriate and well explained and justified. The work provides evidence of extensive wider reading with a strong bias towards quality up-to-date sources, and advanced theoretical /practical technologies, concepts and debates. The solution and recommendations proposed are sophisticated, valid, appropriate, well scoped and reflect a sound grasp of both the theory and the practical context of the case subject. The reflection elements are well scoped, valid, mature, insightful, detailed and validated where relevant. The work is well presented in a professional manner reflecting care and attention to detail and flow and maturity of thought.

#### 90 – 100 Excellent (Distinction)

Has achieved all of the requirements for a rating of ‘Very Good’ and is distinguished by an excellence in all aspects of the problem investigated and creative and inspired solution(s) and recommendations. A high level of professionalism is evidenced throughout. The work demonstrates advanced learning and consultancy ability

## Appendix 4: MSc Dissertation Remote Supervision Form

**Division of Computer Science & Informatics**

**Remote supervision of dissertation: student approval form**

**This form should be completed by any student undertaking a course taught at the London Campuses of London South Bank University, who wishes to relocate in order to complete the dissertation stage of their course, and who would therefore be unable to attend regular face-to- face supervisory meetings. It should be signed and submitted before you go away.**

**Please note: students are not automatically entitled to carry out the dissertation stage of their course remotely**. Please also be aware that if you are an international student with an entry visa sponsored by London South Bank University, then a decision to return to your

**country of domicile for this stage of your programme could have implications with regard to your current visa and ability to re-enter the country. Please contact the International Office for more information in this regard.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student name** |  | **Student number** |  |
| **Course name** |  | **Course code** |  |
| **Signature** |  | **Date submitted** |  |
| **Travel date** |  | **International student** | Yes No |
| **Company Name/Address** |  | | |
| **Contact Name** |  | | |
| **Contact email** |  | | |
| **Reason for request**  ***(if not placement)*** |  | | |

**Staff authorisation (for internal use)**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Name** | **Signature** | **Date** |
| **Project supervisor** |  |  |  |
| **Course director** |  |  |  |
| **Comments for**  **consideration** |  | | |
| **Permission granted** | Yes No | | |