Quantitative Wind Speed Estimation Using AI-driven Image Processing

by

Jatin Kumar

Thesis submitted to the University of Ottawa
in partial fulfillment of the requirements for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

Conventional weather stations are spaced tens of kilometres apart, which means localised wind events often go unmeasured. This thesis asks whether wind speed can be estimated from ordinary video of trees using convolutional neural networks (CNNs), and tests the idea against ground-truth measurements from Environment Canada weather stations.

The work is divided into two phases. Phase 1 uses a stationary camera pointed at a single isolated tree at a rural site near Ottawa. A lightweight CNN ($\sim$0.9M parameters) is trained to classify wind speed into eight 5 km/h bins and to predict continuous speed via regression. On a chronologically held-out test set, the classifier reaches 68.1% accuracy (random baseline: 12.5%) and the regressor achieves a mean absolute error (MAE) of 9.54 km/h. These results show that tree motion carries information related to wind speed, though the single-tree approach has clear limitations.

Phase 2 extends the pipeline to scenes containing multiple trees, using video collected by the National Research Council of Canada from a vehicle travelling under controlled conditions (constant speed, straight-line travel, no external disturbances). Under these constraints, the moving-camera problem reduces to the stationary case. The author's contribution was developing the detection, tracking, and estimation pipeline: a YOLOv8 detector finds tree crowns in each frame, a SORT tracker links them across frames, a motion-CNN with an added optical-flow channel estimates wind speed per tree, and the per-tree estimates are combined by confidence-weighted averaging. On the Phase 2 test set, the pipeline achieves an MAE of 1.56 km/h and a Pearson correlation of 0.985.

The main contribution is the multi-tree pipeline and its experimental validation. Ablation tests show that adding an optical-flow channel cuts per-tree MAE by 19%, and that aggregating over multiple trees cuts the segment-level MAE by a further 33% compared to a single-tree estimate. Taken together, the two phases confirm that tree motion contains wind speed information and that aggregating over multiple trees reduces estimation error, within the conditions tested.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Professor Mohammad Pirani, for his helpful guidance and continuous support throughout this work. I am also deeply grateful to my co-supervisor at the National Research Council Canada (NRC), Dr. Faegheh Ghorbanishohrat, for her valuable advice and for providing the resources from NRC that greatly enabled this research.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

Wind measurement affects transportation safety, energy efficiency, and environmental monitoring. Conventional approaches based on fixed weather stations provide limited spatial coverage and temporal resolution, leaving localised wind phenomena such as sudden gusts and flow variations caused by terrain undetected. Applications that depend on local wind conditions, from vehicle safety systems to wildfire spread models, are constrained by the resolution of available measurements.

To address these limitations, visual anemometry offers a different path. Rather than measuring airflow directly, a camera records the motion of visible features such as tree branches and canopies, and a trained model uses that motion to estimate wind speed. Because cameras are inexpensive and widely available, this approach could provide spatially distributed wind estimates for applications ranging from transportation safety and vehicle energy management to wildfire prediction and pollutant dispersion modelling.

This research is motivated by the need for improved transportation safety, improved vehicle efficiency, and higher-resolution environmental data. The goal of this thesis is to develop and evaluate a machine learning based approach to estimate wind speed from video footage of tree canopy motion, validated against meteorological ground truth from a fixed weather station.

1.2 Importance of Accurate Wind Measurement

Accurate wind information is needed across transportation, environmental management, and engineering. Vehicle safety systems depend on real-time wind data to detect and respond to crosswind hazards; energy management algorithms use wind forecasts to optimise fuel consumption and battery usage; and environmental models require spatially resolved wind fields for applications ranging from pollutant dispersion to wildfire spread prediction.

1.2.1 Transportation Safety

Wind, particularly sudden crosswinds, poses a direct hazard to vehicles. Strong lateral gusts destabilise vehicles and increase the risk of lane departure incidents, particularly for high-profile vehicles such as trucks and buses. Providing real-time wind vectors to a vehicle's lane-keeping controller allows the system to counteract these forces preemptively. A visual anemometer could supply wind vectors in areas where fixed weather stations are sparse, directly supporting road safety.

1.2.2 Energy Efficiency and Emissions

Aerodynamic drag dominates vehicle energy consumption at highway speeds. For any vehicle at 100 kilometers per hour (km/h), aerodynamic drag accounts for a large fraction of total energy use. For electric vehicles (EVs), it can consume up to half the usable battery energy; internal combustion vehicles also see increased fuel consumption under headwind conditions. Integrating wind data into navigation and energy management systems allows route selection and speed adjustment that account for headwind and tailwind conditions, reducing drag losses by 6–12% and extending an EV's range by approximately 25–40 km [1]. Fleet logistics can use routing that accounts for wind conditions to cut fuel consumption and CO_2 emissions.

1.2.3 Resilient Autonomous Operation

Autonomous driving systems, unlike human drivers, cannot react instinctively to sudden lateral forces. They depend on sensor inputs and preprogrammed control logic, so accurate wind data are directly useful for stable operation. If camera wind sensing were integrated into the vehicle's sensor suite, motion planning modules could anticipate lateral

disturbances from gusts and adjust steering and speed preemptively. On a moving vehicle, however, the camera's own movement produces apparent motion in the image that must be separated from motion of trees and foliage caused by wind. This challenge, known as ego-motion compensation, is discussed further in Chapter 2.

1.2.4 Environmental Monitoring and Infrastructure Health

Fine-grained wind data also matters outside transportation. Wildfire spread models and pollutant dispersion forecasts depend on accurate wind inputs; a 3 m/s wind speed error can shift a predicted flame front by over 30 meters in ten minutes [2]. Visual anemometry could also support structural health monitoring: wind loads on bridges, tall buildings, and power lines cause vibrations that accumulate fatigue damage, and spatially resolved wind data would help engineers assess structural integrity.

These applications motivate the study of visual wind estimation. This thesis tests the feasibility and accuracy of such methods under controlled conditions, with results that may inform future deployed systems.

1.3 Limitations of Traditional Wind Measurement

Traditional wind measurement methods provide precise readings at fixed points but cannot deliver the spatially dense data that modern applications require.

Automated Weather Stations (AWSs), the backbone of conventional wind monitoring, are typically spaced 20 km or more apart. This spacing creates kilometer-scale gaps in wind coverage. Within these gaps, localised phenomena such as gusts on bridge decks or turbulence in forest clearings go undetected. Such gaps affect vehicle safety, bridge loading estimates, and environmental forecast accuracy.

Standard anemometers are mounted at 10 m above ground and do not capture near-surface wind conditions. Wind speed varies with height because of surface roughness, terrain, and nearby obstacles. Recorded differences reach 4 $m\,s^{-1}$ between mast height and near-surface level on moderately wooded roads. This mismatch reduces the value of standard readings for applications that depend on local, ground-level wind conditions.

Attempting to address these coverage and height limitations by deploying more traditional masts is also economically impractical. The lifecycle cost of a single solar powered ultrasonic installation, including its procurement, calibration, and ongoing telemetry, can

range from $5,000 to $7,000 [3]. Scaling this cost across vast monitoring areas makes dense deployment financially unfeasible.

On-board mechanical sensors such as wind vanes or cup anemometers mounted on a moving vehicle do not solve the problem. These sensors measure the sum of the true ambient wind and the vehicle's slipstream. Separating the two components would require real-time computational fluid dynamics (CFD) models, which remain unreliable in the turbulent conditions of active traffic. Protruding mechanical instruments also suffer from aerodynamic drag, icing in winter, and drift from vibration, all of which raise maintenance costs and degrade accuracy.

Point anemometers also lack contextual information. They do not capture visual cues such as texture, depth, or the movement of tree branches and foliage. Deep learning networks can exploit such visual data for self-supervised learning and wind estimation. Cameras capture scene geometry and the motion of multiple wind-responsive objects at once. This property of visual data is a central reason for pursuing the visual anemometry approach in this thesis.

1.4 Objectives of the Thesis

The goal of this thesis is to develop and evaluate a visual system that estimates wind speed by analysing the motion of tree canopies captured on video, validated against meteorological ground truth from an official weather station. The thesis addresses the following objectives:

O1. Develop a data collection and processing pipeline for visual wind estimation: To establish a reproducible workflow for capturing video of tree canopy motion alongside synchronized ground-truth wind speed data from Environment Canada's weather station network. This includes defining the camera placement, recording protocol, temporal synchronization procedure, and preprocessing steps required to convert raw footage into labelled training samples suitable for machine learning.

O2. Develop a CNN pipeline for wind speed estimation from stationary cameras: To design and train a convolutional neural network (CNN) that classifies and regresses wind speed from short video clips of tree canopy motion recorded with a stationary camera. This includes developing methods for both single-tree scenarios and multi-tree scenes, where the latter integrates an object detection module (YOLOv8)

for automated tree crown localization, a tracking module (SORT) for temporal consistency, and a confidence-weighted aggregation scheme that fuses per-tree wind speed predictions into a single scene-level estimate. For Phase 1 (single-tree), data collection and ground-truth validation were performed by the student using a fixed camera setup. The multi-tree pipeline was developed by the student and validated on data collected by NRC (see Objective O3).

O3. Examine the extension to mobile platforms at variable speeds: To investigate the applicability of the stationary camera methodology to a moving vehicle scenario under restrictive operational assumptions: controlled vehicle speed, no steering during measurement, and absence of external disturbances such as gusts. Under these constraints, the problem reduces to the stationary case by subtracting the vehicle speed component in the direction of observed features. For this phase, the experimental setup and data collection were conducted by the National Research Council of Canada, and the analysis focuses on validating the adapted methodology.

O4. Evaluate the system and compare with baselines: To assess the accuracy of the proposed pipeline on held-out test data using standard metrics (mean absolute error, RMSE, R^2, classification accuracy) and compare performance against a random classifier baseline. The results are also compared with relevant approaches reported in the literature.

1.5 Organization of the Thesis

This thesis is structured in six chapters. Chapter 1 introduces the background and motivation for visual wind sensing, highlights the importance of accurate wind measurement, reviews the shortcomings of existing methods, and states the thesis objectives. Chapter 2 provides a literature review covering atmospheric flow characteristics, the visual anemometry concept, machine learning tools for environmental monitoring, prior visual wind estimation studies, and current challenges in data collection and validation. Chapter 3 details the experimental setup, including camera hardware and placement, weather station data for ground truth, recording conditions, and data synchronization methods, clearly distinguishing between the stationary camera work (conducted by the student) and the mobile platform data at variable speeds (collected by NRC). Chapter 4 presents the proposed methodology for stationary camera wind estimation and its adaptation to the constrained mobile scenario, covering data preprocessing, CNN model design, training procedures, and evaluation metrics. Chapter 5 presents and analyzes the experimental results, discusses

the findings, provides a comparative analysis with relevant literature, and examines the extension to the mobile platform case. Finally, Chapter 6 summarizes the key findings, discusses the contributions of this research, acknowledges limitations, and outlines directions for future work.

Chapter 2

Literature Review

This chapter reviews the literature relevant to quantitative wind estimation from visual data, covering atmospheric flow characteristics, the development of visual anemometry, machine learning techniques, prior studies across different platforms, and open challenges in data collection.

2.1 Atmospheric Flow Characteristics

2.1.1 Challenges in Outdoor Wind Perception

Atmospheric wind is unsteady and context-dependent, which makes it difficult to measure accurately. Its behaviour varies with the surrounding terrain, with open plains, dense forests, urban canyons, and mountainous roads each creating unique local airflow patterns [4]. Since wind itself is invisible, its direct observation is impractical; instead, researchers rely on observing its secondary effects, such as the swaying of tree branches, the movement of foliage, or the swirling of dust and debris [5].

These indirect indicators vary across environments, making it difficult to build methods that generalise beyond the specific location and conditions where they were trained [6].

2.1.2 Limitations of Conventional and On-Vehicle Anemometry

As discussed in Chapter 1, conventional wind monitoring relies on fixed weather stations spaced 20 km or more apart, leaving large gaps in coverage [3, 4]. Applications that

require fine-grained or high-frequency wind data cannot rely on these sparse networks alone. Installing additional stations is expensive and slow, so researchers have looked to alternative sensing methods that can work alongside sparse fixed infrastructure [6].

Mounting conventional anemometers on moving vehicles introduces further problems, including slipstream contamination and mechanical degradation. In the literature, several groups have confirmed these issues. Ranjan et al. [7] demonstrated that separating ambient wind from airflow generated by the vehicle itself demands precise calibration across speed, yaw rate, and acceleration; such corrections are computationally intensive and error-prone in dynamic traffic. Protruding mechanical instruments remain vulnerable to aerodynamic drag, icing, and vibration-induced drift [8], making them poorly suited for large-scale use on vehicles and motivating alternatives that use cameras instead.

2.1.3 Motivation for Vision-Based Wind Sensing

Real-time wind data are needed for the safe operation of autonomous vehicles (AVs) and vehicles in general. Autonomous systems, unlike human drivers, cannot react instinctively to sudden lateral wind disturbances. They depend on pre-programmed logic and sensor inputs [9], and accurate wind data feeds directly into trajectory planning, lane centering, and adaptive cruise control [10].

Wind resistance also directly affects energy consumption across all vehicle types. A vehicle driving into a headwind experiences increased fuel or energy consumption and reduced range, while a tailwind can improve efficiency. High-resolution wind maps enable more accurate energy forecasting and adaptive power management [1]. For high-profile vehicles such as trucks or buses, anticipating strong gusts can influence routing decisions or trigger automated warnings [11].

These practical needs, combined with the constraints of fixed weather stations and on-vehicle mechanical sensors discussed above, have led to growing interest in visual wind estimation, also known as visual anemometry. This approach uses camera systems to infer wind speed from image sequences [3, 12].

Initial proof-of-concept studies demonstrated that observing environmental cues, such as the swaying of trees or the movement of airborne dust, through a camera can yield quantitative wind estimates. By tracking motion cues within a visual scene, researchers have correlated image dynamics with Beaufort scale categories and with readings from co-located anemometers [1, 3]. More advanced systems have used stereo cameras and spatio-temporal neural networks to separate motion caused by wind from ego-motion on moving

platforms [13, 14]. Together, these studies indicate that visual methods can serve as a practical alternative to physical anemometry where spatially dense wind data are needed.

2.2 Visual Anemometry Concept

Visual anemometry refers to the estimation of wind characteristics from image sequences by analysing the motion of visible proxies such as tree branches, grass, or airborne particles. The underlying principle is that wind displaces objects and the resulting motion encodes the flow. Decoding this motion is already difficult with a fixed camera; when the camera is also moving, separating wind-induced motion from camera motion adds a further layer of complexity.

2.2.1 Early Fixed-Camera Studies

Initial visual anemometry studies assumed a stationary viewpoint. The "See the Wind" project [3] deployed cameras around birch trees and correlated optical flow from branch tips with cup-anemometer readings, achieving less than 2 m/s RMS error across a range of Beaufort classes. Subsequent work used phase-based motion magnification to detect sub-millimetre vibrations in wind turbine blades, demonstrating sensitivity to even minor wind-induced motion [5]. The "VisualWind" model [1] employed a dual-branch CNN that merged optical flow and RGB data to classify wind into discrete categories. Other early studies used flags as wind indicators, analysing their flutter patterns to infer wind speed and direction [3], while Particle Image Velocimetry (PIV) techniques were adapted for natural scenes using airborne particles to quantify airflow in controlled environments [15].

These experiments validated the concept and moved visual anemometry from qualitative to quantitative wind measurement, but they were limited to fixed locations with consistent backgrounds and specific proxies, restricting their applicability to dynamic, real-world scenarios.

2.2.2 Optical Flow Foundations

Most visual wind estimation systems begin by computing dense optical flow, the pixel-wise motion field between consecutive frames. Classical methods such as Horn–Schunck [16] and Lucas–Kanade [17] rely on brightness constancy and small-displacement assumptions,

which often fail in natural outdoor scenes where lighting flickers and motion is irregular. Deep learning approaches such as PWC-Net [18] and RAFT [19] overcome these limitations by learning flow estimation from data. PWC-Net introduced a pyramid-warping-cost-volume architecture that improved both accuracy and efficiency, while RAFT employed a recurrent unit to iteratively refine the flow field, achieving state-of-the-art performance on standard benchmarks. In the "See the Wind" dataset, switching from Farnebäck [20] to RAFT improved flow accuracy by 38% on low-texture spruce foliage [12]. Recent work has also explored unsupervised flow estimation methods that use photometric consistency without ground-truth labels [13].

All of these models, however, presume a static camera; motion from a host vehicle introduces a dominant bias unless explicitly compensated.

2.2.3 Depth, Ego-Motion, and the "Moving Camera" Challenge

Ego-motion, the apparent image motion caused by the camera's own movement, must be separated from environmental motion before signals from wind can be extracted. Competitive Collaboration (CC) [13] is a self-supervised framework that jointly estimates depth, camera motion, and scene flow without requiring external labels. By enforcing photometric consistency and segmenting static from dynamic pixels, CC allows environmental motion to be disentangled from camera motion. Extensions such as RAFT-Stereo [21] have improved depth estimation near object boundaries using stereo cues, which aids accurate 3D reconstruction in dynamic scenes.

The challenge extends beyond simply subtracting ego-motion: other dynamic objects (vehicles, pedestrians) must also be distinguished from motion caused by wind. Scene flow estimation, which provides a 3D motion field between consecutive frames, is one approach to this problem [10]. Event cameras, which provide asynchronous, high-frame-rate updates based on pixel intensity changes, have also been explored for improved fast-motion tracking under high-dynamic conditions [14].

2.2.4 Learning the Wind-Mapping Function

After extracting proxy motion and compensating for ego-motion, the remaining task is to infer quantitative wind speed and direction. Two strategies dominate the literature. **Physics-guided regressors** embed aerodynamic principles directly into the learning pipeline. For example, Garcia and Liu [22] regressed wind speed from quadcopter imagery by analysing residual control efforts and pixel motion, constrained by drag models.

Related work has explored inverse aerodynamic models to infer wind forces from observed structural deformations [1]. **Purely data-driven networks**, by contrast, learn wind estimation directly from visual features without explicit physics models. The VisualWind model [1] trained a CNN to classify wind into discrete categories based on stacked optical flow frames. Recurrent architectures such as LSTMs have been used to capture temporal dependencies in video sequences for more robust predictions [8]. Performance in both approaches improves when auxiliary inputs such as IMU and GPS data are fused into the model. Multi-modal fusion has been shown to reduce MAE by approximately 0.4 m/s [4].

2.2.5 Synthetic Data and Domain Adaptation

Real-world training data for high-wind conditions are scarce: extreme gusts are infrequent, and collecting data during severe storms is hazardous. Researchers have turned to synthetic data generation to fill this gap. Diffusion-based video synthesis, for example, can produce realistic high-wind scenarios for model pretraining [2], while physics simulators can generate diverse wind conditions on demand [8, 11].

However, models trained exclusively on synthetic footage often exhibit a "domain gap," struggling with real-world generalisation [11]. Domain adaptation techniques, such as adversarial feature alignment [23] and fine-tuning on limited real clips, can reduce this gap by approximately 20% on benchmark datasets. Generative adversarial networks [24] and neural rendering techniques such as Neural Radiance Fields [25] have improved the realism of synthetic training environments, though fully bridging the simulation-to-real gap remains an open problem.

2.2.6 Sensor Fusion and Field Validation

Multi-sensor systems improve robustness by capturing complementary data. Combining RGB video with infrared, stereo depth, or LiDAR enables better detection of motion under variable lighting or obscured conditions [6]. A 2021 study used three environmental cameras and co-located LiDAR to halve wind-direction error during nighttime radiation inversions [4].

Field validation typically involves comparing visual estimates against readings from portable anemometers positioned near the proxy or from Automated Weather Stations at known distances [3, 6].

2.3 Machine Learning in Environmental Monitoring

Estimating wind speed from camera footage is fundamentally a learning problem: given a sequence of images showing motion caused by wind, the task is to infer quantitative wind speed while compensating for any camera motion.

2.3.1 Early Supervised Regressors Built on Heuristic Motion Features

The earliest machine learning models for visual wind sensing utilised classical optical flow methods, such as Horn–Schunck [16] and Lucas–Kanade [17], to extract motion features between frames. These handcrafted features were then fed into shallow regressors, including Support Vector Machines [26] and Random Forests [27], to classify Beaufort wind categories or directly regress wind speed. Zhang et al. [1] achieved an RMSE below 2 m/s using a dual-branch CNN that merged motion and appearance cues to classify wind into discrete categories. While promising, these early methods struggled with sensitivity to lighting changes and camera misalignment, and their performance was limited by the accuracy of the underlying classical flow algorithms in areas of low texture or large displacement.

2.3.2 Deep Flow Encoders and Self-Supervised Motion Decomposition

As discussed in Section 2.2.2, deep optical flow networks such as PWC-Net [18] and RAFT [19] have largely replaced classical methods for flow estimation in wind sensing pipelines. These high-resolution flow fields are typically stacked over time and fed into spatio-temporal models for wind regression or classification. Common architectures for temporal aggregation include 3D CNNs [28] and ConvLSTMs [29]. Replacing classical flow with deep learning alternatives has improved robustness in complex outdoor environments with irregular lighting and non-rigid motion.

When the camera itself is moving, these flow fields must be decomposed further. As introduced in Section 2.2.3, the Competitive Collaboration framework [13] jointly estimates depth, ego-motion, and scene flow in a self-supervised manner, allowing environmental motion to be isolated without labelled data. Models that add semantic segmentation to identify and ignore non-wind-responsive objects (e.g., passing vehicles) have improved wind estimation accuracy [30]. Training without explicit flow supervision makes these models suited to domains where ground-truth motion labels are unavailable.

2.3.3 Multimodal Fusion and Uncertainty Quantification

Accuracy and robustness in visual wind estimation are improved through multi-sensor fusion. Combining RGB video with infrared imagery, stereo depth, LiDAR point clouds, IMU data, and GPS coordinates provides complementary information that compensates for individual sensor limitations [4,6]. In autonomous driving, established datasets such as KITTI [31] and nuScenes [32] have demonstrated the value of multi-sensor benchmarking, though no equivalent public benchmark yet exists specifically for visual wind estimation.

Fusion architectures range from classical Kalman filters [33] to deep neural networks [34] that learn to weight sensor inputs adaptively. Beyond point estimates, quantifying prediction uncertainty matters for applications where safety is at stake. Bayesian techniques such as dropout-based uncertainty estimation [35] and quantile regression [36] can provide confidence bounds alongside wind predictions, which matters when the system needs to flag uncertain estimates rather than act on them blindly.

2.4 Previous Studies on Wind Estimation Using Visual Data

Several groups have applied the techniques discussed above to estimate wind from cameras in practice. The following subsections organise these studies by platform type.

2.4.1 Stationary Visual Sensing Approaches

Early work in visual wind estimation primarily adopted fixed-camera setups, using environmental cues such as swaying trees, flags, or other visible proxies to approximate wind strength and direction. Physics-based approaches infer wind speed from the vibrational response of known structures. Sarrafi et al. [37] applied phase-based motion magnification to turbine blade vibrations, revealing strong correlation with nearby sensor data. Flag-based methods have also been explored, analysing flutter patterns to infer wind speed [3].

Supervised methods treat wind estimation as a supervised classification or regression task. Zhang et al. [1] combined dense optical flow and RGB data in a dual-branch CNN to classify wind into discrete categories. However, their work highlighted key limitations: video sources varied in quality, lacked standard frame rates, and ground-truth labels were often visually estimated rather than instrumentally measured. Pre-processing was frequently needed to eliminate extraneous motion from vehicles or people, reinforcing the

constraint that fixed-camera setups require relatively controlled scenes. PIV-based approaches using airborne particles to quantify airflow have also been applied in controlled environments [15], though their extension to uncontrolled outdoor settings remains impractical.

2.4.2 Vehicle-Mounted Camera Studies

Using mobile cameras for wind sensing introduces significant complexity due to the need to disentangle ego-motion (the apparent image motion caused by the camera platform's movement) from wind-induced environmental motion. While few studies explicitly focus on wind estimation from moving vehicles, related work in visual odometry, scene flow, and motion segmentation provides relevant insights. Chen et al. [10] used front and rear vehicle-mounted cameras to estimate real-time driving trajectories via optical flow and CNNs, demonstrating reliable motion tracking from dynamic platforms.

Ranjan et al. [7] proposed an unsupervised deep network that jointly estimates depth, scene motion, and camera motion, which is particularly relevant for separating background flow from motion caused by wind. Event cameras, offering asynchronous, high-frame-rate updates, have also been explored for vehicular applications; Ozawa et al. [14] showed that event data transformed into a bird's-eye view can improve fast-motion tracking and ego-motion estimation. The challenge remains in adapting these general motion estimation techniques specifically for the subtle and often ambiguous motions caused by wind.

2.4.3 Wind Estimation from Drone Platforms

Unmanned Aerial Vehicles (UAVs) represent another avenue for mobile wind sensing, though their methods often rely more on internal flight dynamics than on visual cues. Allison et al. [8] trained an LSTM to estimate wind velocity based solely on a quadcopter's IMU and control inputs, without an external anemometer. Their model outperformed traditional physics-based estimators in simulated turbulent fields. Garcia and Liu [22] similarly used drag-constrained regression on quadcopter imagery.

However, as noted by Ahmed et al. [11], most current UAV wind estimators are not readily transferable across drone platforms. Research on combining drone and ground-camera inputs for wider-area wind mapping remains sparse.

2.4.4 Optical Flow and Video-Based Techniques

Across all platforms, optical flow remains the most widely used visual cue for wind estimation. Many state-of-the-art models, including VisualWind [1] and See the Wind [3], use optical flow as the primary input for CNN classification or regression. The transition from classical algorithms [16, 17] to deep learning methods [18, 19] has significantly improved accuracy in complex outdoor scenes, as discussed in Sections 2.2.2 and 2.3.2.

Advanced techniques further improve sensitivity. Phase-based motion magnification can detect sub-pixel movements invisible to standard flow algorithms [5]. Self-supervised architectures that jointly learn flow, depth, and motion segmentation achieve strong performance without external labels [7].

Existing work has established that visual wind estimation is feasible, yet open problems remain in data generalisability, adaptation to moving platforms, and dependence on curated training sets. The work described in the following chapters uses a CNN trained on stationary video of tree canopy motion as a step toward addressing some of these gaps.

2.5 Challenges in Data Collection and Validation

Accurate visual wind estimation depends on the availability of high-quality, well-annotated datasets and sound validation methodologies. This section discusses the principal challenges.

2.5.1 Data Scarcity and Ground Truth Labelling

High-wind events are infrequent and often dangerous to instrument, which means available datasets are heavily skewed toward moderate conditions. Models trained on such data tend to perform poorly in the situations where accurate wind estimates matter most [11]. Instrumenting sites for extreme weather is also logistically costly, limiting how much real high-wind data can be collected [3].

Even when data can be collected, obtaining precise ground-truth wind measurements that are accurately synchronised with visual information presents its own difficulties. Traditional anemometers provide point measurements that may not represent the wind field across an entire visual scene, especially in complex environments with varying terrain and obstacles [4]. The spatial offset between a reference anemometer and the observed proxy introduces labelling noise. Advanced techniques such as PIV [15] can provide detailed flow

fields in controlled settings, but their application in uncontrolled outdoor environments is impractical. These constraints have motivated the use of self-supervised and weakly supervised learning methods that reduce reliance on dense ground-truth labels [13].

2.5.2 Environmental Variability and Proxy Behaviour

Visual wind estimation relies on observing environmental proxies such as trees, flags, and foliage. These proxies do not respond to wind alone: their motion is also affected by lighting changes, occlusions, and their own structural properties [5]. Different tree species respond differently to the same wind speed, and passing objects or shadows can obscure or corrupt the signal. Variations in illumination and reflections further degrade optical flow accuracy [18]. Building models that handle this variability without extensive per-location tuning is an unsolved problem [6].

Data scarcity, particularly for high-wind conditions, compounds these challenges. The use of synthetic data generation and domain adaptation techniques to mitigate this problem was discussed in Section 2.2.5, and remains an active area of research.

2.5.3 Validation Methodologies

Validating visual wind estimation systems requires comparing predictions against measurements from co-located anemometers or established Automated Weather Stations [3,4]. This comparison is complicated by the inherent difference between point measurements from traditional sensors and spatially distributed visual estimates. Synchronisation of data from multiple sensors (cameras, IMU, GPS, and reference anemometers) is critical [6].

Established autonomous-driving datasets such as KITTI [31] and nuScenes [32] provide frameworks for multi-sensor benchmarking, though no comparable public benchmark exists for visual wind estimation. Beyond accuracy metrics such as MAE, reporting prediction uncertainty through confidence bounds is important for safety-critical applications [35,36].

2.6 Summary of Literature Review

Table 2.1 provides a concise summary of the key studies discussed in this chapter, highlighting their primary methodology, contribution, and relevant limitations.

Table 2.1: Condensed Summary of Key Literature in Visual Wind Estimation

Ref	**Primary Focus / Methodology**	**Key Contribution**	**Limitations / Context**
[22]	Physics-guided regression for quadcopter imagery	Estimated wind speed using drag-constrained motion	Drone-specific; not directly transferable
[1]	CNN for wind-speed classification	Classified wind into discrete categories	Qualitative ground truth; fixed camera
[2]	Diffusion-based augmentation of high-wind scenarios	Generated realistic synthetic training data	Domain gap with real-world data
[3]	Fixed-camera optical flow on trees	$< 2\ m/s$ RMS error for stationary wind sensing	Location-specific; non-generalisable
[5]	Phase-based motion magnification	Detected sub-mm wind-induced vibrations	Indirect; vibration-focused
[8]	LSTM for drone flight data	Outperformed physics-based estimators in turbulence	Drone-dependent dynamics
[14]	Event cameras with contrast maximisation	Robust fast-motion tracking under high dynamics	Requires specialised sensors
[15]	Particle Image Velocimetry review	Established baseline for visual flow quantification	Controlled environments only
[16]	Horn–Schunck dense optical flow	Foundational flow estimation algorithm	Sensitive to lighting changes
[17]	Lucas–Kanade sparse optical flow	Iterative patch-based registration	Errors in low-texture regions
[18]	PWC-Net CNN for optical flow	Pyramid + warping + cost volume; efficient flow	Assumes static camera
[19]	RAFT recurrent all-pairs field transforms	State-of-the-art dense optical flow	Static-camera assumption
[29]	ConvLSTM for spatio-temporal dependencies	Effective for weather sequence prediction	Applied to precipitation, not wind

Chapter 3

Experimental Methodology

This chapter describes the equipment, recording conditions, and data sources used in the two experimental phases of this study. Both phases share the same general principle: correlating tree canopy motion captured on video with wind speed measurements from Environment Canada weather stations.

In Phase 1, the author collected data using a stationary camera overlooking an isolated tree at a rural site in Nepean, Ottawa. Ground truth was obtained from a local Environment Canada station. In Phase 2, the experimental setup and data collection were conducted by the National Research Council of Canada (NRC) as part of a collaborative project; the author's contribution was restricted to processing and analysing the NRC-provided data. Phase 2 extends the stationary methodology to a vehicle at variable speeds, with ground truth from the London A station (ICAO: CYXU) in London, Ontario. Under the restrictive operational assumptions of Phase 2 (controlled vehicle speed, no steering, and absence of external disturbances), the problem reduces to the stationary case by subtracting the vehicle speed component in the direction of observed features.

3.1 Data Collection Setup

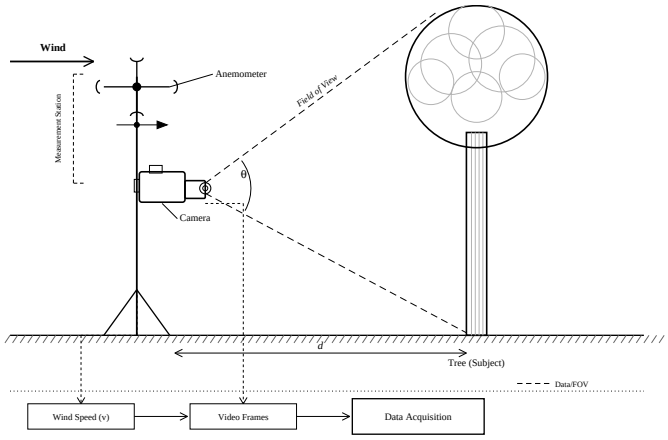
3.1.1 Phase 1: Camera Hardware and Placement

A single digital video camera was mounted on a tripod at a fixed position at a rural site in Nepean, Ottawa, overlooking a mature deciduous maple tree. The site was chosen to provide a clear view of the tree canopy without significant obstructions from surrounding

infrastructure. The camera was configured to record at 1920×1080 pixel resolution and 30 frames per second using an H.264 codec at a variable bitrate of approximately $20\,\mathrm{Mbit\,s^{-1}}$.

The tripod was securely anchored to prevent camera movement or vibration, so that all observed motion in the video originates from the tree and foliage rather than from the camera. Because the camera remained stationary throughout data collection, perspective changes and parallax were absent, and any frame-to-frame motion can be attributed to wind effects.

The camera position was sufficiently close to the tree to detect fine movements such as leaf fluttering, but far enough to include the majority of the crown in each frame. No special optics or filters were applied; recordings were made with the camera's standard lens settings. Recording was conducted only during daylight hours and in dry conditions to maintain good visibility. Figure 3.1 shows the schematic layout of the recording site.



(a) Schematic layout of the recording site.

(b) Actual scene as captured by the camera.

Figure 3.1: Phase 1 recording site. (a) Schematic showing the stationary camera mounted on a tripod overlooking the target tree. The anemometer shown on the measurement station beside the camera was a portable unit used for occasional on-site reference checks; it was *not* the source of ground-truth labels. All ground-truth wind speed data were obtained from the Environment Canada weather station described in Section 3.1.3. (b) A representative frame from the camera, showing the full scene before preprocessing. The preprocessed version of this frame is shown in Chapter 5 (Figure 5.2).

3.1.2 Phase 2: Camera Hardware and Placement

The Phase 2 dataset was collected by NRC researchers as part of a collaborative project on visual wind estimation. The recording setup consisted of six cameras mounted on a vehicle travelling through areas with trees at varying depths. The cameras recorded at a mix of resolutions (some at 1920×1080 and others at 3840×2160 (4K)), all at 60 frames per second, providing higher temporal resolution than the Phase 1 setup. Although six cameras were available, only the data from a single front-facing camera was used in this work. The vehicle operated under the following constraints: controlled speed, straight-line travel with no steering input, and absence of significant external disturbances such as gusts. Under these conditions, the apparent motion due to vehicle movement can be accounted for by subtracting the vehicle speed component in the direction of the observed trees, effectively reducing the problem to the stationary case addressed in Phase 1. The author did not participate in the Phase 2 data collection; the author's contribution was limited to applying the detection, tracking, and wind estimation pipeline described in Chapter 4 to the NRC-provided data for validation purposes.

3.1.3 Weather Station Data for Ground Truth

Ground-truth wind measurements were obtained from Environment and Climate Change Canada weather stations. For Phase 1, the nearest station to the Nepean recording site was used; for Phase 2, the London A station (ICAO code: CYXU) near the NRC recording routes in London, Ontario was used. Both stations employ standard anemometers mounted at the World Meteorological Organization standard height of 10 m, providing 10-minute averaged wind speeds and directions together with maximum gust values.

In both cases the reference station is not co-located with the camera. The relatively flat terrain and the absence of major topographic barriers between the stations and their respective recording sites support the assumption that synoptic-scale wind conditions at the station are representative of those at the recording location. Prior work has shown that trees separated by a few kilometres experience similar synoptic-scale wind regimes under relatively uniform terrain conditions [3]. This spatial offset is a recognised limitation of the study and is discussed further in Section 3.4.

All wind speed values are reported in kilometres per hour (km/h) throughout this thesis.

3.1.4 Software Tools and Data Handling

Python was used for data handling and video processing, using open-source libraries including `OpenCV` [38] for reading and processing video frames, `pandas` for managing time-stamped weather data, and `NumPy` [39] for numerical operations. Custom scripts were developed to retrieve periodic wind data for each recorded session and to align each video segment with its corresponding wind speed label.

Video recording was started and stopped manually using the camera's built-in controls, and the timing of each recording was noted. For Phase 2, the NRC team used a comparable manual recording protocol.

Table 3.1 summarises the key data-collection differences between the two phases. A full side-by-side comparison including preprocessing choices and evaluation metrics is provided in Table 5.6 (Chapter 5).

Table 3.1: Data-collection parameters for Phase 1 and Phase 2.

Parameter	**Phase 1**	**Phase 2**
Location	Ottawa (Nepean)	London, Ontario (NRC site)
Camera platform	Stationary (tripod)	Vehicle at variable speeds
Number of cameras	1	6 (1 front-facing used)
Resolution	1920×1080	Mixed (1080p / 4K)
Frame rate	30 fps	60 fps
Clip duration	2.7 s	2.7 s
Trees per scene	1	3–12
Ground truth	Env. Canada (local stn.)	Env. Canada (CYXU)
Data collected by	Author	NRC

3.2 Recording Sessions and Conditions

3.2.1 Phase 1 Recording Sessions

Phase 1 data were collected through a series of video recording sessions between the winter of 2024 and June 2024. Recording on multiple non-consecutive days over this period ensured variability in both weather conditions and canopy appearance, ranging from bare branches in winter to full foliage in late spring.

On each recording day, the camera was typically started near the top of the hour to coincide with a weather station observation time. Each daily session lasted between 20 and 60 minutes. In total, the Phase 1 dataset comprises 12 hours and 7 minutes of video footage, corresponding to approximately 1.3 million frames. Across these sessions, mean wind speeds ranged from 0.72 to 51.48 $\mathrm{km\,h^{-1}}$, with peak gusts reaching 65.16 $\mathrm{km\,h^{-1}}$. The dataset therefore includes instances of nearly still foliage as well as significant canopy movement under strong wind.

Throughout all recording sessions, the camera remained in the same fixed position and orientation, maintaining a consistent frame composition. The background scene remained constant; differences between recordings are due to changes in wind conditions and minor lighting variations. Since the analysis focuses on motion, lighting changes such as shifting shadows or contrast from varying cloud cover were not considered significant.

The recording location was relatively isolated, and sessions were inspected for transient objects or occlusions. The motion captured in the videos therefore predominantly represents the tree's response to wind.

3.2.2 Phase 2 Recording Sessions

The Phase 2 dataset was collected by NRC using six vehicle-mounted cameras recording at 60 frames per second, of which only the front-facing camera was used in this work. In total, the six cameras recorded approximately 33 hours of video; since only the single front-facing camera was used, the effective footage for this study was approximately 5.5 hours. The footage was segmented into non-overlapping 2.7-second clips (162 raw frames per clip, downsampled during preprocessing as described in Chapter 4). Recording was conducted in areas with deciduous trees at varying depths, with between 3 and 12 visible tree crowns per frame depending on camera orientation and foliage density. The vehicle travelled along straight segments following the operational constraints outlined in Section 3.4. Ground-truth wind speed labels were obtained from the London A Environment Canada weather station (ICAO: CYXU) using the same temporal alignment procedure described for Phase 1.

As with Phase 1, all recordings were made during daylight hours and in dry conditions. The higher frame rate in Phase 2 (60 fps vs. 30 fps in Phase 1) was chosen by the NRC team to capture finer temporal detail in tree canopy motion. The controlled vehicle speed and straight-line operation ensured that the dominant motion in the video remained tree sway caused by wind rather than parallax from vehicle movement.

The NRC-provided data were used for testing and validation of the methodology developed in Phase 1; the motion-CNN was not retrained on Phase 2 data. The only component trained on NRC data was the YOLOv8 tree detector, which was fine-tuned on annotated Phase 2 frames to learn the "tree" class (Section 4.3 in Chapter 4). This separation ensures that the Phase 2 results reflect genuine generalisation of the Phase 1 motion-CNN to the multi-tree setting.

3.3 Data Synchronisation and Preprocessing

After recording video footage and obtaining corresponding wind speed data, each portion of video was aligned with the correct wind measurement. The camera's clock was synchronised to official local time so that the video timeline could be matched to the weather station's reporting schedule. Each video session was then associated with the wind speed reading from the nearest 10-minute observation window.

For Phase 1, recordings were often begun near the top of an observation period to simplify alignment. For Phase 2, the NRC team applied the same temporal alignment procedure. Further preprocessing steps, including frame extraction, cropping, normalisation, and the construction of labelled training samples, are described in Chapter 4 as part of the respective Phase 1 and Phase 2 methodologies.

3.4 Assumptions and Limitations

The following assumptions underlie the data collection strategy for both phases:

Spatial homogeneity of wind. The wind speed measured at the respective reference station for each phase is assumed to be representative of the conditions affecting the trees in the video. In reality, microclimates or local obstructions could cause differences between the station and the camera site. The relatively flat terrain in both recording areas supports this assumption, but the spatial offset remains a source of labelling noise.

Stationary camera. With the camera fixed on a stable tripod, all observed motion in the frame is attributed to wind effects. Camera vibration or drift was prevented through secure mounting and verified by inspecting background-pixel stability in the recordings.

Direction-independent response. Wind direction was not included in the ground-truth labels. The implicit assumption is that the magnitude of tree canopy motion correlates primarily with wind speed regardless of direction. This is a simplification, as wind

direction can influence the visible profile of tree sway, but differentiating direction was beyond the scope of the current study.

Phase 2 operational constraints. Phase 2 extends the stationary methodology to cameras mounted on a moving vehicle. The following assumptions are required for this extension to hold: (i) the vehicle speed is controlled and known so that the motion component can be subtracted; (ii) the vehicle follows a straight path with no steering input; and (iii) there are no significant external disturbances such as sudden gusts during recording. Under these constraints, the wind estimation problem reduces to the Phase 1 stationary case by subtracting the known vehicle speed component in the direction of the observed trees from the measured apparent motion. Violations of any of these assumptions would introduce uncompensated motion that could degrade estimation accuracy.

Phase 2 data provenance. The Phase 2 dataset was collected and curated by NRC. The author relied on the NRC team's recording protocol and did not independently verify the raw data acquisition. The NRC-provided data were used solely for testing and validation, not for model training. Any systematic differences between the Phase 1 and Phase 2 recording conditions (e.g., frame rate, tree species, distance from the weather station) are acknowledged as potential confounds when comparing results across phases.

Chapter 4

Proposed Method for Visual Wind Measurement

Using the experimental setup described in Chapter 3, this chapter presents the technical methodology used to estimate wind speed from video recordings of tree motion. The approach was executed in two phases. In the first phase, a controlled scenario involving a single tree was used to establish a proof-of-concept. In the second phase, the approach was extended to scenes containing multiple trees by integrating object detection using YOLOv8 [40] to automatically identify and track trees. Both phases involved training convolutional neural networks (CNNs) to learn temporal motion features that correlate with wind speed. The final predicted wind speed was validated against ground truth data obtained from Environment Canada's weather station (see Chapter 3 for details of the ground-truth source).

The chapter is organised as follows. Section 4.1 gives a high-level overview of the approach. Sections 4.2 and 4.3 describe the two experimental phases in detail, covering data preprocessing, model architecture, training, and evaluation.

4.1 Overview of the Proposed Approach

The overall framework consists of a visual analysis pipeline designed to estimate wind speed from trees visible in video footage. This visual information is mapped to known wind speed labels through supervised machine learning, specifically convolutional neural networks. Two distinct experimental pipelines were created:

In **Phase 1 (Single Tree Experiment)**, a cropped and centred video of a single tree was used to train a CNN to classify wind speeds into eight categories, defined in Table 4.1. Frames were extracted, converted to greyscale, and stacked along the channel dimension to form motion-encoding input tensors (see Section 4.2.2 for details). In **Phase 2 (Multiple Tree Detection and Tracking)**, extending the Phase 1 approach to multi-tree scenes, video footage with several trees was processed using YOLOv8 [40] for tree detection. Each tree was tracked across frames using the SORT (Simple Online and Realtime Tracking) algorithm [41]. These sequences were passed through a CNN to estimate wind speeds.

In both phases, the target wind speed labels were aligned with timestamps from Environment Canada's wind station readings for the recording location and time. Model performance was evaluated using accuracy and mean absolute error metrics.

Figure 4.1 presents the overall system architecture as a UML component diagram, showing the data preprocessing, model training, inference, and per-frame processing subsystems. Solid arrows denote data flow between components, while dashed arrows indicate dependency relationships.

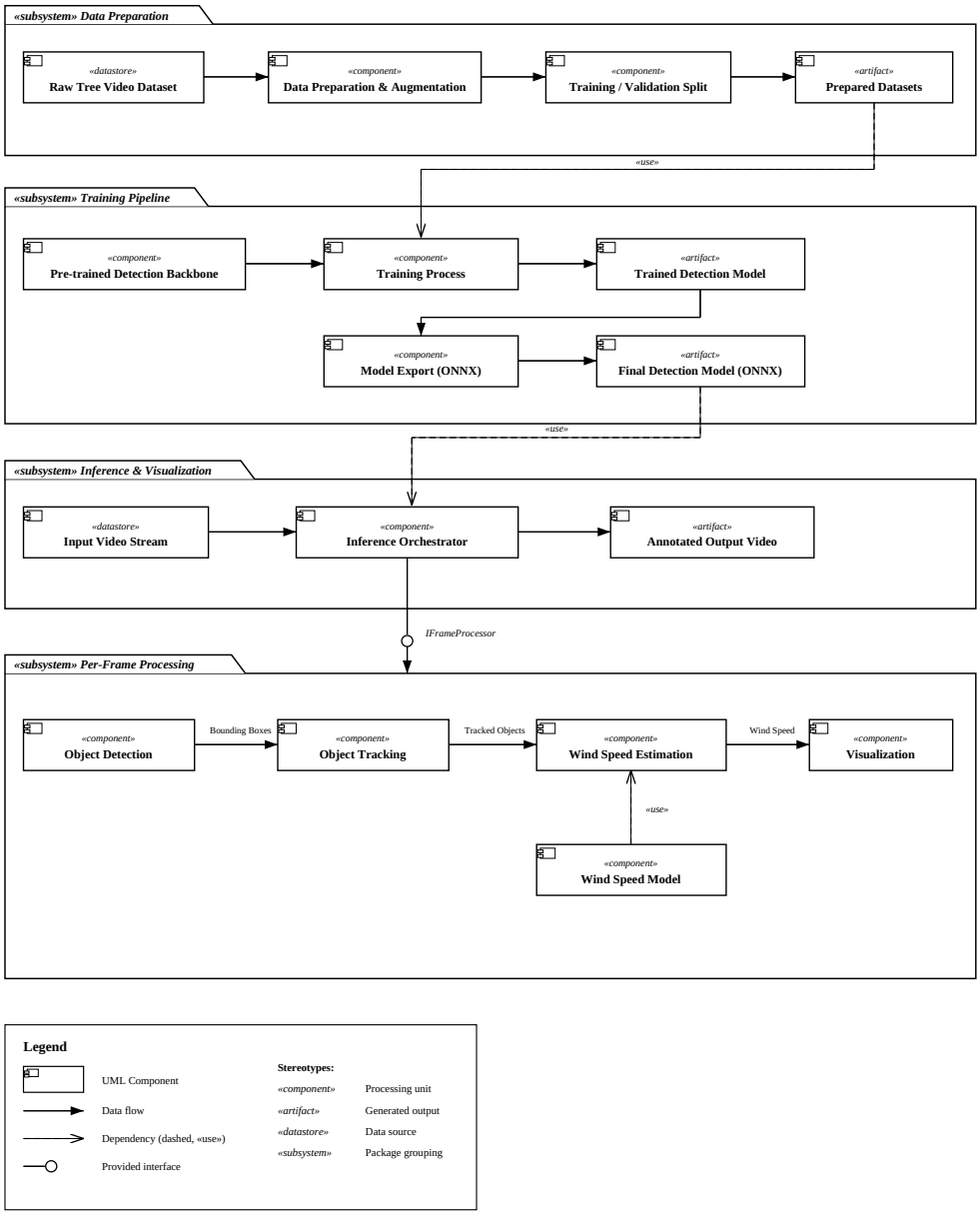


Figure 4.1: System design of the wind model. This figure illustrates the complete workflow of the proposed wind estimation system, including data preprocessing, model training, inference, and integration of detection, tracking, and wind prediction components.

4.2 Phase 1: Single Tree Wind Speed Classification

Phase 1 establishes a proof-of-concept that the visual motion of *one* isolated tree contains sufficient information to infer the ambient wind speed. All processing is deliberately kept simple (no object detection or tracking is needed) so that the link between tree sway and wind can be examined in a controlled setting before moving to the more complex multi-tree pipeline described in Section 4.3. This section documents how the raw footage was converted into a labelled machine-learning dataset and describes the CNN architecture, training protocol, and evaluation metrics used in this phase. The data collection setup and recording conditions are described in Chapter 3.

4.2.1 Data Collection for the Single-Tree Experiment

The data collection setup, camera hardware, weather station, and recording conditions for Phase 1 are described in Chapter 3 (Sections 3.1.1, 3.1.3, and 3.2.1). This section provides additional detail on the dataset characteristics relevant to the methodology.

Data were collected using a stationary camera overlooking a mature deciduous tree at a rural site in Nepean, Ottawa. Recordings were performed on multiple non-consecutive days between the winter of 2024 and June 2024, ensuring that the dataset captured seasonal variations in canopy appearance, ranging from bare branches to full foliage. Each daily session lasted between 20 and 60 minutes, producing a cumulative dataset of 12 hours and 7 minutes, corresponding to roughly 1.3 million frames.

Ground-truth wind measurements were obtained from the Environment Canada station described in Chapter 3 (Section 3.1.3). During the study period, mean wind speeds ranged from 0.72 to 51.48 $\mathrm{km\,h^{-1}}$, with peak gusts reaching 65.16 $\mathrm{km\,h^{-1}}$.

Video was recorded at 1920×1080 pixel resolution and 30 frames per second. To avoid artefacts associated with low-light capture, recordings were restricted to daylight hours.

Synchronisation between the camera and the weather station data was achieved by matching camera timestamps to the nearest 10-minute observation window in the Environment Canada series. Since the weather data are reported at a 10-minute cadence, instantaneous wind values corresponding to each video clip were derived by linearly interpolating between the surrounding two records, following common practice in visual-anemometry studies.

A few assumptions underpinned this data collection setup. First, the camera was assumed to remain perfectly immobile throughout each session, such that any observed

pixel motion would correspond exclusively to tree sway rather than camera shake. Second, the tree canopy was treated as a passive wind proxy, with the magnitude of its motion presumed to increase monotonically with wind speed over the observed range. Third, spatial variability in wind conditions between the weather station and the recording site was assumed to be small under the prevailing terrain conditions. These assumptions, while simplifying, provide a reasonable basis for linking canopy motion to wind measurements in the single-tree experiment. A more detailed discussion of assumptions is provided in Chapter 3 (Section 3.4).

4.2.2 Data Preprocessing and Segmentation

The raw video from the single-tree experiment (Section 4.2.1) could not be fed directly to a CNN. The continuous recordings had to be segmented into short clips, frames extracted and normalised, unusable samples discarded, and each clip assigned a wind-speed label. All operations were automated in Python so that the dataset could be rebuilt from scratch, ensuring reproducibility.

The first stage involved dividing each continuous video into fixed-length clips. A custom segmentation utility was designed to read MP4 files named with their UTC start timestamp. Each file was sliced into 2.7-second clips using a sliding window with a 1-second stride, so that consecutive clips overlap by 1.7 seconds. The 2.7-second duration was chosen after exploratory tests demonstrated that it was sufficiently long to capture at least one full oscillation of canopy sway at the lowest observed frequencies (approximately 0.5 Hz). The 1-second stride increased the number of training samples by a factor of approximately 2.7 relative to non-overlapping extraction. Because the chronological split partitions data by recording day (Section 4.2.3), overlapping clips from the same session always reside within the same partition and do not cause train–test leakage. Every clip inherited a mid-time stamp $t_{\text{mid}} = t_{\text{start}} + 1.35\,\text{s}$, which was later used to fetch the interpolated wind speed $\hat{v}(t_{\text{mid}})$ from the Environment Canada series. A systematic naming convention based on timestamps ensured that filenames and labels shared a common key.

Once clips were created, frames were extracted and normalised to prepare them for CNN input. To reduce storage and memory requirements, the original 30 fps stream was down-sampled to 20 equally spaced frames per 2.7-second clip. This reduction was experimentally found to retain sufficient temporal information to capture canopy sway while substantially reducing data volume. Each frame was cropped to a fixed 1280×720 region centred on the canopy, removing ground and sky clutter and yielding consistent

framing across sessions.[1] Frames were then converted to greyscale, as colour information contributed little additional motion detail and would have tripled the memory footprint. The cropped greyscale frames were resized to 224×224 pixels using bilinear interpolation, after which the 20 frames were stacked along the channel dimension to produce a tensor of shape $224 \times 224 \times 20$. Finally, pixel intensities were scaled to the range [0,1], and dataset-wide mean and variance statistics were stored for application at inference time.

To maintain data quality, several automated filters were applied to remove problematic clips. Out-of-focus segments were identified by computing the variance of the Laplacian σ_L^2 for the first frame; clips with $\sigma_L^2 < 80$ were discarded. Occlusions caused by pedestrians or other moving objects were flagged by histogram comparisons between consecutive frames; if more than 40% of pixels deviated by over 20 grey levels, the clip was excluded. Clips whose mid-time stamp fell outside the range of available wind records were ignored. After these filtering steps, 87.2% of the original clips were retained, resulting in a dataset of 38,270 usable samples. The complete preprocessing pipeline is summarised in Figure 4.2.

Each clip was then annotated with its corresponding wind speed. Two labelling strategies were adopted to support both regression and classification tasks. For the regression variant, the interpolated value $\hat{v}(t_{\text{mid}})$ in kilometres per hour was stored directly, and training loss was computed in the same units. For the classification variant, wind speeds were discretised into eight bins of 5 km/h width. The first seven bins span 0–5, 5–10, ..., 30–35 $\text{km}\,\text{h}^{-1}$; the eighth bin captures all speeds $\geq 35\ \text{km}\,\text{h}^{-1}$. These bins, summarised in Table 4.1, cover the wind-speed range observed during recording sessions. All labels were stored in a CSV manifest alongside filenames, enabling the PyTorch `DataLoader` to retrieve both the image tensor and its target value during training.

[1] The cropping region was fixed and stored in a .json configuration file, ensuring reproducibility across sessions.

Table 4.1: Wind-speed bins (in $\mathrm{km\,h^{-1}}$) used for the classification variant.

Class ID	**Range [$\mathbf{km\,h^{-1}}$]**
0	0.0 – 5.0
1	5.0 – 10.0
2	10.0 – 15.0
3	15.0 – 20.0
4	20.0 – 25.0
5	25.0 – 30.0
6	30.0 – 35.0
7	$\geq$ 35.0

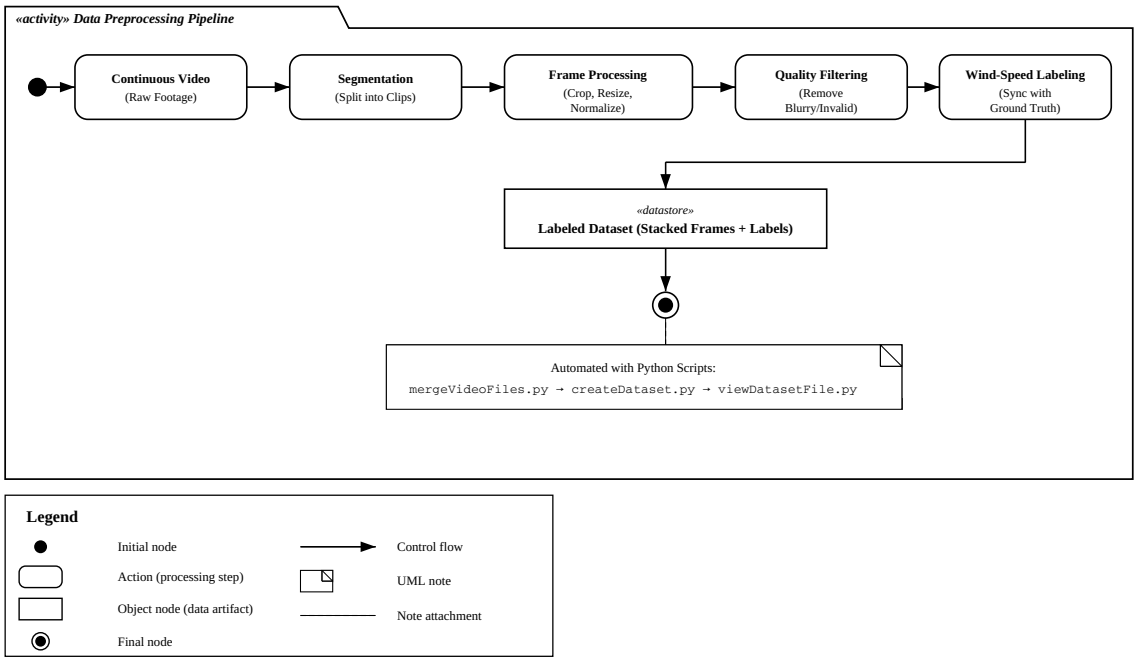


Figure 4.2: Pipeline used to convert continuous video into labelled clips for CNN training.

4.2.3 Dataset Construction and Script Workflow

With individual 2.7-second clips prepared and annotated (Section 4.2.2), the next step was to organise them into a reproducible dataset with a fixed directory hierarchy, partitioning strategy, and safeguards for transparency.

The entire pipeline was orchestrated through a single command-line utility. This program accepts a configuration file in `YAML` format that specifies parameters such as the directory containing raw MP4 video files, the Environment Canada wind series to be used, clip segmentation length (default 2.7 seconds), number of sampled frames per clip (default 20), the bounding box for canopy cropping, and thresholds for blur or occlusion detection. By centralising all preprocessing decisions in a single YAML file, the build process ensured that every run could be reproduced exactly. The script generates tensor files, applies all filtering and labelling steps, and records dataset statistics including acceptance ratios and summary wind speed distributions in a log file. Because all randomised operations are seeded, re-executing the pipeline with the same configuration produces byte-identical outputs.

The resulting dataset followed a structured hierarchy, shown in Figure 4.3. Each partition (training, validation, and test) contained only tensor files, while the labels were stored centrally in a manifest file, `clips_metadata.csv`, which included columns for the filename, continuous wind speed, and categorical class identifier. A chronological split strategy was adopted: the first nine recording days formed the training set (70% of samples), the next two days comprised the validation set (15%), and the final three days were held out for testing (15%). This temporal division was chosen to prevent leakage of wind statistics between adjacent clips recorded on the same day.

The rationale for splitting by recording day, rather than randomly, is that wind speeds within a single recording session are strongly autocorrelated. A random split would allow clips from the same weather episode to appear in both training and test sets, leading to misleadingly optimistic performance. Splitting by day ensures that the model is evaluated on genuinely unseen meteorological conditions. A potential concern with day-level splitting is that certain wind regimes may be concentrated in certain days, leading to imbalanced partitions. To mitigate this, the wind speed distribution across bins was monitored for each split and confirmed to fall within a tolerance of $\pm 3\%$ of the overall distribution; the per-bin sample counts are reported in Appendix A (Table A.2) and discussed alongside the results in Chapter 5. In addition, recording days were chosen to span a variety of weather conditions (Section 3.2.1), reducing the risk that any partition is systematically biased toward a narrow wind range.

After filtering, the dataset contained 26,800 training samples, 5,730 validation samples, and 5,740 test samples (Table 4.2).

```
single_tree_dataset/
train/
clip_20240501_093512.npy
clip_...
val/
test/
clips_metadata.csv
build.log
cfg.yaml
```

Figure 4.3: Directory hierarchy created by `build_single_tree_dataset`.

Table 4.2: Final dataset split after preprocessing and filtering.

Split	**Samples (# clips)**
Training	26 800
Validation	5730
Test	5740

Three measures ensured reproducibility and transparency. First, all sources of randomness (e.g. file shuffling) were controlled with a fixed seed via `numpy.random.seed`, and the seed value was logged. Second, every configuration file was copied verbatim into the dataset root directory, ensuring that downstream training always referenced the exact preprocessing parameters. Third, all scripts were version-controlled in a Git repository.

4.2.4 CNN Model Design for Wind Speed Classification

The central component of Phase 1 is a convolutional neural network (CNN) designed to map the visual motion observed in a 2.7-second clip to either a wind-speed category or a continuous value. This section describes the input tensor representation, the layer topology, and the key architectural decisions.

The input to the model consists of the pre-processed clips described in Section 4.2.2. Each clip is stored as a tensor $\mathbf{X} \in \mathbb{R}^{224\times224\times20}$, where the 20 greyscale frames are treated as separate channels rather than as an explicit temporal sequence. In this formulation, the CNN interprets the input as a stack of spatial images with depth equal to the number of frames. All convolutions are therefore performed in the spatial plane, with learned filters spanning across the 20 channels. This strategy, often referred to as "late fusion" [42], follows the design adopted in VisualWind [1]. First, it reduces computational cost compared to a full three-dimensional convolution, which would require $\mathcal{O}(k_t)$ more parameters to handle temporal kernels of depth k_t. Second, it allows the network to implicitly capture motion cues: for example, a convolutional filter sensitive to alternating bright-dark vertical edges would respond strongly to leaves moving horizontally between successive frames. In Phase 1, ablation experiments showed that adding an explicit optical-flow channel did not significantly improve validation accuracy for the single-tree, stationary-camera setting, suggesting that the stacked-frame representation already encodes sufficient motion information for that configuration. In Phase 2, however, the optical-flow channel reduced per-tree MAE by 19% (Section 5.2), likely because the more complex multi-tree scenes and vehicle-induced motion benefit from an explicit motion representation.

The complete network architecture is summarised in Table 4.3 and illustrated schematically in Figure 4.4. The network is lightweight, with approximately 0.9 million parameters, which allows training from scratch on the 26,800 training clips without overfitting. The architecture begins with a convolutional layer using 7×7 kernels and stride 2 to capture broad texture patterns across the canopy, followed by a max-pooling operation that halves the spatial resolution. This is followed by a sequence of progressively deeper convolutional blocks with increasing filter counts (32, 64, 128, 256), each using 3×3 kernels. Every stride-2 operation reduces the feature map size, allowing subsequent layers to focus on larger-scale canopy motion while keeping computational cost in check. At the penultimate stage, global average pooling [43] is applied over the final $14 \times 14 \times 256$ feature maps, reducing them to a compact $1 \times 1 \times 256$ representation. The pooled features are passed to a dense layer of 128 neurons with ReLU activation, followed by a dropout layer [44] with rate $p = 0.5$. The final output layer varies depending on whether classification or regression is performed.

Table 4.3: Layer configuration of the single-tree CNN.

Layer	**Output size**	**Kernel/Stride**	**#F**	**Activation**
Input	$224 \times 224 \times 20$	–	–	–
Conv 1	$112 \times 112 \times 32$	$7 \times 7/2$	32	ReLU
MaxPool	$56 \times 56 \times 32$	$3 \times 3/2$	–	–
Conv 2	$56 \times 56 \times 64$	$3 \times 3/1$	64	ReLU
Conv 3	$28 \times 28 \times 128$	$3 \times 3/2$	128	ReLU
Conv 4	$14 \times 14 \times 256$	$3 \times 3/2$	256	ReLU
GlobalAvgPool	$1 \times 1 \times 256$	–	–	–
Dense 1	128	–	–	ReLU
Dropout (p=0.5)	128	–	–	–
Output	8 (softmax) or 1 (linear)	–	–	–

4.2.5 Training Procedure and Hyper-parameters

The convolutional neural network described in Section 4.2.4 was trained end-to-end using PyTorch 2.1 [45] with CUDA 12.2 acceleration on an NVIDIA RTX 3090 GPU with 24 GB of memory. All experiments were executed within a Docker container based on the official NVIDIA CUDA runtime (CUDA 12.2, cuDNN 8, Ubuntu 22.04), ensuring consistent library versions and a controlled environment for reproducibility.

The dataset partitioning strategy established in Section 4.2.3 was strictly adhered to during model development. The training, validation, and test sets were constructed in chronological order to avoid temporal leakage, and the validation split was used solely for monitoring convergence and tuning hyper-parameters. At no point was validation data exposed to the optimiser beyond the reporting of metrics. Data loading was performed by a PyTorch `DataLoader` configured with four worker processes, enabling streaming of samples from solid-state storage at approximately 800 clips per second.

The choice of hyper-parameters followed an initial grid search across five combinations of learning rates and batch sizes; the full grid search results are reported in Appendix A.3. Table 4.4 summarises the configuration that yielded the best validation performance. The Adam optimiser [46] with β_1=0.9 and β_2=0.999 was selected for its stability in non-convex optimisation. An initial learning rate of 1×10^{-4} was paired with a ReduceLROnPlateau scheduler, which adaptively halved the rate after three epochs without improvement in validation loss. Weight decay of 1×10^{-4} provided L_2 regularisation, and a batch size of

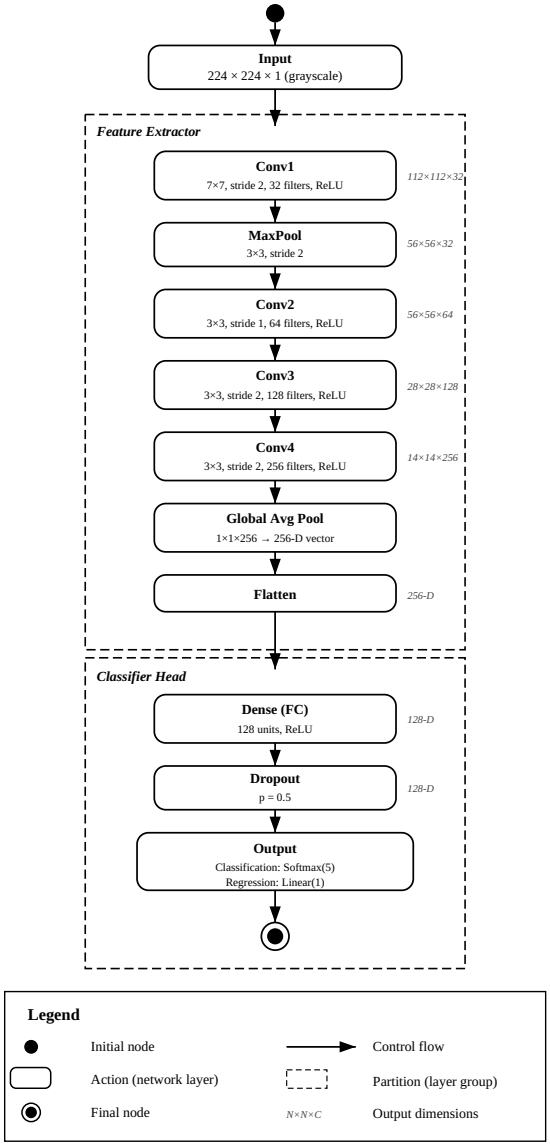


Figure 4.4: CNN architecture for single-tree wind-speed classification.

32 clips was chosen as the maximum compatible with GPU memory constraints. Training proceeded for up to 50 epochs, although early stopping with patience of five epochs and a minimum validation-loss improvement of 10^{-3} usually terminated runs earlier. For classification, categorical cross-entropy served as the loss function, while the regression variant employed a Smooth-L_1 (Huber) loss [47] with δ=1, balancing sensitivity to outliers with robustness to noise. To guarantee reproducibility, all random number generators in NumPy, PyTorch, and cuDNN were initialised with seed 42.

Table 4.4: Hyper-parameter configuration for Phase 1 training.

Parameter	**Value**
Optimiser	Adam (β_1=0.9, β_2=0.999)
Initial learning rate	1×10^{-4}
LR scheduler	ReduceLROnPlateau (factor 0.5, patience 3)
Weight decay (L_2)	1×10^{-4}
Batch size	32 clips
Epochs (max)	50
Early-stopping	Patience 5, Δval-loss $< 10^{-3}$
Loss (classification)	Categorical cross-entropy
Loss (regression)	Smooth-L_1 (Huber, δ=1)
Random seed	42

4.2.6 Evaluation Metrics and Validation

The numerical results are presented in Chapter 5; this section defines the metrics and the validation protocol.

To evaluate the classification variant of the model, several standard metrics were computed. Overall classification accuracy was defined as

$$\mathrm{Acc} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\big[\hat{y}_i = y_i\big], \tag{4.1}$$

where $\mathbb{1}[\cdot]$ is the indicator function, returning 1 when the condition is true and 0 otherwise, $\hat{y}_i$ and y_i denote the predicted and true class labels for clip i, and N is the total number of test clips. Beyond accuracy, more granular information was obtained by computing precision (P_c), recall (R_c), and the F_1 score for each wind class c, derived from the confusion

matrix. These measures help assess whether rare but dangerous high-wind events were systematically under-detected. For the regression variant, performance was summarised with the mean absolute error (MAE),

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{v}_i - v_i\right|, \tag{4.2}$$

where $\hat{v}_i$ is the predicted continuous wind speed for clip i and v_i is the corresponding ground-truth value from the Environment Canada series. The root mean square error (RMSE),

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{v}_i - v_i)^2}, \tag{4.3}$$

and the coefficient of determination (R^2),

$$R^2 = 1 - \frac{\sum_i (\hat{v}_i - v_i)^2}{\sum_i (v_i - \bar{v})^2}, \tag{4.4}$$

where $\bar{v}$ represents the mean wind speed of the test set. In all cases, metrics were computed at the per-clip level and then averaged across the entire held-out test split. Classification accuracy and F_1 were reported for the eight-class classifier, while MAE, RMSE, and R^2 quantified the regression head.

The temporal hold-out strategy described in Section 4.2.3 was designed to prevent information leakage from autocorrelated wind sequences. Recordings from the first nine field days were allocated to training, days 10–11 formed the validation set, and days 12–14 were reserved for testing. All hyper-parameter tuning was conducted using validation metrics only, and no information from the test set was accessed until the final evaluation.

The primary baseline was a random classifier, implemented as part of this work, that predicted classes in proportion to their observed frequency in the training set, yielding an expected accuracy of approximately $1/8 \approx 12.5\%$. This trivial baseline established the minimum performance threshold that any learned model must exceed to demonstrate that useful features have been extracted from the visual input.

In addition to these baselines, the results in Chapter 5 include a comparison with wind estimation approaches reported in the literature (Chapter 2), particularly those employing static cameras and tree-based proxies, such as the "See the Wind" project [3] and the VisualWind model [1]. This comparison contextualises the Phase 1 results within the existing literature.

4.3 Phase 2: Multi-Tree Wind Speed Estimation

Phase 1 showed that a single tree can serve as a visual proxy for wind speed estimation. Phase 2 extends the pipeline to scenes containing multiple trees at varying depths, captured from a vehicle at variable speeds. As described in Chapter 3, Phase 2 operates under restrictive assumptions (controlled vehicle speed, no steering, and absence of external disturbances) that reduce the problem to the stationary case by subtracting the vehicle speed component in the direction of observed features. Phase 2 is not an independent contribution; rather, it builds directly on the Phase 1 methodology by adding automated tree localisation and multi-proxy aggregation, and validates the adapted approach on data collected by NRC.

The two principal challenges are (i) automatically localising every tree crown in a wide-angle frame and (ii) aggregating the motion evidence from multiple, possibly heterogeneous proxies into a single wind-speed estimate. Tree localisation is handled by a YOLOv8 object-detection model [40] (Section 4.3.3); motion of each detected crown is summarised through a CNN similar to Phase 1 (Section 4.3.4); finally, crown-wise predictions are combined into a scene-level estimate via a confidence-weighted averaging scheme (Section 4.3.5).

The data collection setup and recording conditions for Phase 2 are described in Chapter 3 (Sections 3.1.2 and 3.2.2). As noted there, the Phase 2 dataset was collected by the National Research Council of Canada (NRC) and was used for testing and validation of the methodology developed in Phase 1; the author's contribution was limited to developing and applying the detection, tracking, and estimation pipeline described in this section.

4.3.1 Data Sources and Collection (Multi-Tree Scenario)

The Phase 2 dataset, collected by NRC, consists of video footage from six cameras mounted on a vehicle travelling through areas with deciduous trees at varying depths. Although six cameras were available, only the front-facing camera was used for this work. The cameras recorded at a mix of resolutions (1080p and 4K) at 60 frames per second (see Chapter 3, Section 3.2.2 for further details on recording conditions and operational constraints). Under the restrictive assumptions outlined in Chapter 3, the apparent motion due to vehicle movement is accounted for by subtracting the vehicle speed component, reducing the problem to the stationary case.

Ground-truth wind data were sourced from the London A Environment Canada weather station (ICAO: CYXU), as described in Chapter 3, Section 3.1.3. Wind speed values were

stored in kilometres per hour in a `pandas` `DataFrame` indexed by timestamp. Synchronisation between video and wind measurements was achieved by matching frame timestamps to the nearest 10-minute observation window and linearly interpolating between surrounding records, following the same procedure used in Phase 1.

The final dataset covered a range of environmental conditions. The footage from the front-facing camera was segmented into non-overlapping 2.7-second clips at 60 fps (162 raw frames per clip). Each clip was decoded at 20 fps (retaining every third frame), producing 54 decoded frames per clip, from which 20 equally spaced frames were selected for model input. Mean wind speeds across the recording days ranged from approximately 5 to 35 $\mathrm{km\,h^{-1}}$, covering calm, moderate, and stronger conditions.

A representative example of the Phase 2 footage is shown in Figure 4.5, illustrating the visual complexity of the multi-tree scenario: overlapping canopies at different depths create a substantially more challenging problem than the single-tree case.

Figure 4.5: Typical Phase 2 scene with multiple trees at varying depths.

4.3.2 Pre-processing and Annotation

Phase 2 required a more complex preprocessing strategy than Phase 1 because of the multi-tree setting and the need to isolate individual crowns in each frame. Two complementary pipelines were developed: one that prepared frames for object detection and bounding-box annotation, and another that segmented video into clips and coupled tree tracks with wind-speed labels. All stages were implemented in Python and automated end-to-end; a high-level overview of the workflow is shown in Figure 4.6.

The first branch of the workflow prepared raw video frames for object detection. Continuous recordings were decoded at 20 frames per second, reducing storage demands while retaining sufficient temporal resolution for motion analysis. Extracted frames were resized to 1280×720 pixels using bicubic interpolation, preserving the 16:9 aspect ratio. Each image was saved to disk with a naming convention based on timestamps, accompanied by a JSON metadata file storing the absolute UTC timestamp.

Because YOLOv8's default COCO weights did not include a dedicated "tree" class, a custom detector was fine-tuned. A set of frames was sampled uniformly across all sessions and seasonal conditions, and bounding boxes were drawn manually around each visible tree crown using the CVAT (Computer Vision Annotation Tool) web interface. A single class, labelled *tree*, was defined. Data augmentation was applied using Ultralytics' built-in transforms, including four-image mosaic stitching (probability 0.5), hue/saturation/value shifts of up to ±10%, random scaling in the range 0.5–1.5, translations up to ±10%, and small rotations of ±5°. Horizontal flipping (probability 0.5) was also enabled, since tree orientation is not tied to prevailing wind direction.

In total, 4,000 frames were sampled and annotated with approximately 8,000 bounding boxes. Fine-tuning details for the YOLOv8 detector are provided in Section 4.3.3.

The second branch of preprocessing converted the detected and tracked trees into wind-labelled training clips. Continuous video was divided into 2.7-second segments. Within each segment, detections were linked across frames using the SORT tracker [41], with an IoU-based association strategy. Each resulting track defined a bounding-box time series. For every track that persisted for a sufficient proportion of the segment duration, the corresponding cropped regions were extracted, resized to 224×224 pixels, and down-sampled to produce 20 equally spaced frames per clip (matching the decode procedure described in Section 4.3.1). These were stacked to form a tensor compatible with the Phase 1 input representation, ensuring architectural compatibility. The segment midpoint was mapped to the Environment Canada wind series by linear interpolation, and every tree track within the segment inherited the same label.

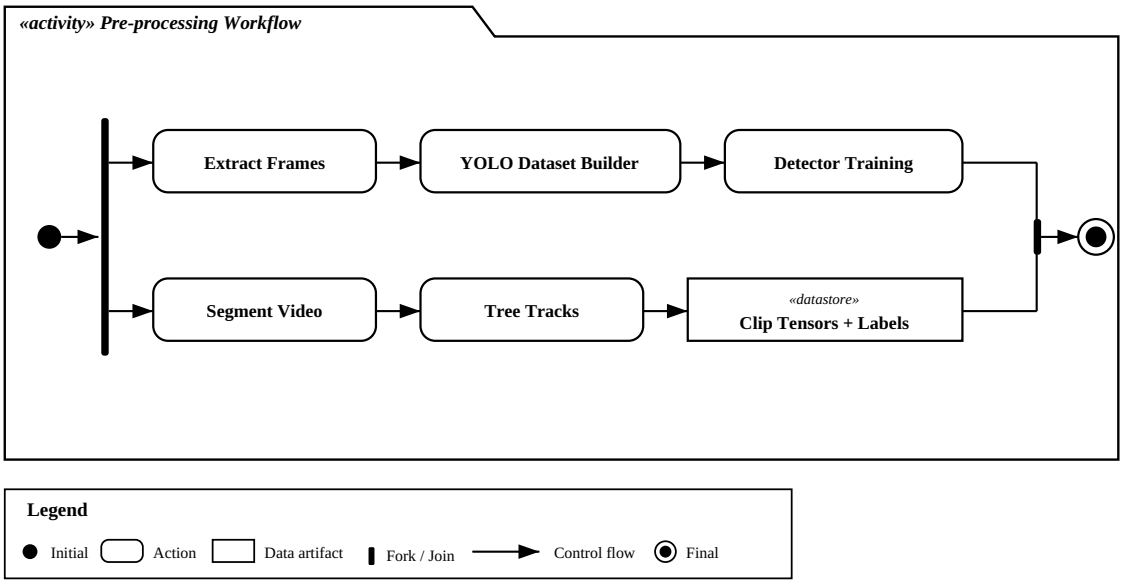


Figure 4.6: Pre-processing workflow for the multi-tree dataset.

The complete Phase 2 pipeline extracted per-tree clips from each video segment; the number of per-tree clips per segment varied with the number of detected and tracked trees (typically 3–12). Since the motion-CNN was not retrained on Phase 2 data, the NRC-provided segments were partitioned chronologically into a validation set and a held-out test set only (Table 4.5), following the same day-level splitting rationale described for Phase 1 in Section 4.2.3. The YOLOv8 detector was trained separately on 4 000 annotated frames drawn from the earlier recording days (Section 4.3.3).

Table 4.5: Phase 2 chronological dataset split for motion-CNN evaluation (day-level).

Partition	**Recording days**	**Segments**
Validation	∼30% of recording days	∼1 800
Test	Final 3 recording days	4 000

4.3.3 Tree-Detection Module: YOLOv8 Integration

Accurate localisation of tree crowns is a prerequisite for per-tree motion analysis in Phase 2. The YOLOv8 detector [40], released by Ultralytics in January 2023, was selected because it combines detection accuracy with real-time throughput on single-GPU systems, and its PyTorch implementation offers flexibility for customisation to specific domains.

At a high level, YOLOv8 is an anchor-free, one-stage object detector composed of three main components. Since general-purpose YOLOv8 models do not include a "tree" class, a custom model was fine-tuned on annotated frames from the Phase 2 dataset. The backbone is inspired by CSPDarknet, making use of *GELU* activations and `C2f` cross-stage partial blocks to improve gradient flow while limiting parameter count. A PANet-like neck aggregates features at multiple scales through BiFPN-style top-down and bottom-up paths. Finally, a decoupled detection head performs both classification and regression, outputting objectness scores, class logits, and bounding box offsets at three spatial scales. For this study the YOLOv8-s variant, containing 7.2 million parameters, was adopted. On an NVIDIA RTX 3090 GPU, this configuration processed 1280×720 frames at approximately 80 frames per second.

The training set consisted of 4,000 manually labelled frames containing approximately 8,000 annotated tree bounding boxes. Training proceeded for 100 epochs with a batch size of 16, using stochastic gradient descent with cosine-annealed learning rate scheduling (base rate 0.01), momentum 0.937, and weight decay of 5×10^{-4}. The same augmentation

settings described in Section 4.3.2 were applied. Validation was carried out on a held-out set of 15% of the annotated frames. The detector reached its best performance at epoch 52, achieving a mean average precision at IoU 0.5 (mAP_{50}) of 93.4%, after which the weights were frozen for downstream use.

During inference, the detector returns for every frame f a set of bounding boxes and confidence scores,

$$\mathcal{D}_f = \{(b_i, \hat{p}_i)\}_{i=1}^{N_f},$$

where each $b_i = (x, y, w, h)$ is a bounding box in pixel coordinates and $\hat{p}_i \in (0, 1)$ denotes the objectness confidence. Post-processing applies Non-Maximum Suppression with an IoU threshold of 0.5 to eliminate duplicate detections, and retains boxes with $\hat{p}_i \geq 0.35$. This threshold was experimentally determined to maximise recall while keeping false positive rates below 5%.

Detections were passed to a SORT tracker [41] to maintain consistent track IDs across time. For each segment, tracks persisting for a sufficient duration were deemed reliable and used to assemble motion-CNN inputs. The cropped frames were resized to 224×224 pixels and stacked into tensors consistent with the input representation of Phase 1.

Figure 4.7 illustrates the detection output on a representative Phase 2 frame: the raw driving scene (top) and the same frame after YOLOv8 tree detection with bounding box and wind speed overlay (bottom).

4.3.4 Motion Analysis and CNN Model for Wind Estimation

Once tree crowns are localised by the YOLOv8 detector, the next task is to convert their sway into a quantitative wind-speed estimate. The multi-tree pipeline consists of three stages: region cropping and tensor construction, per-tree CNN inference, and per-segment fusion. These are illustrated schematically in Figure 4.8 and summarised in Algorithm 4.1. In Figure 4.8, "STE" denotes the Single-Tree Estimator, i.e., the per-tree motion-CNN that maps a cropped tree tensor to a scalar wind-speed prediction $\hat{v}_{ij}$, and "CNN" refers to the convolutional neural network architecture described in Table 4.3.

Figure 4.7: Phase 2 detection example. Top: raw frame from the vehicle-mounted camera. Bottom: the same frame after YOLOv8 tree detection, showing the bounding box and predicted wind speed overlay.

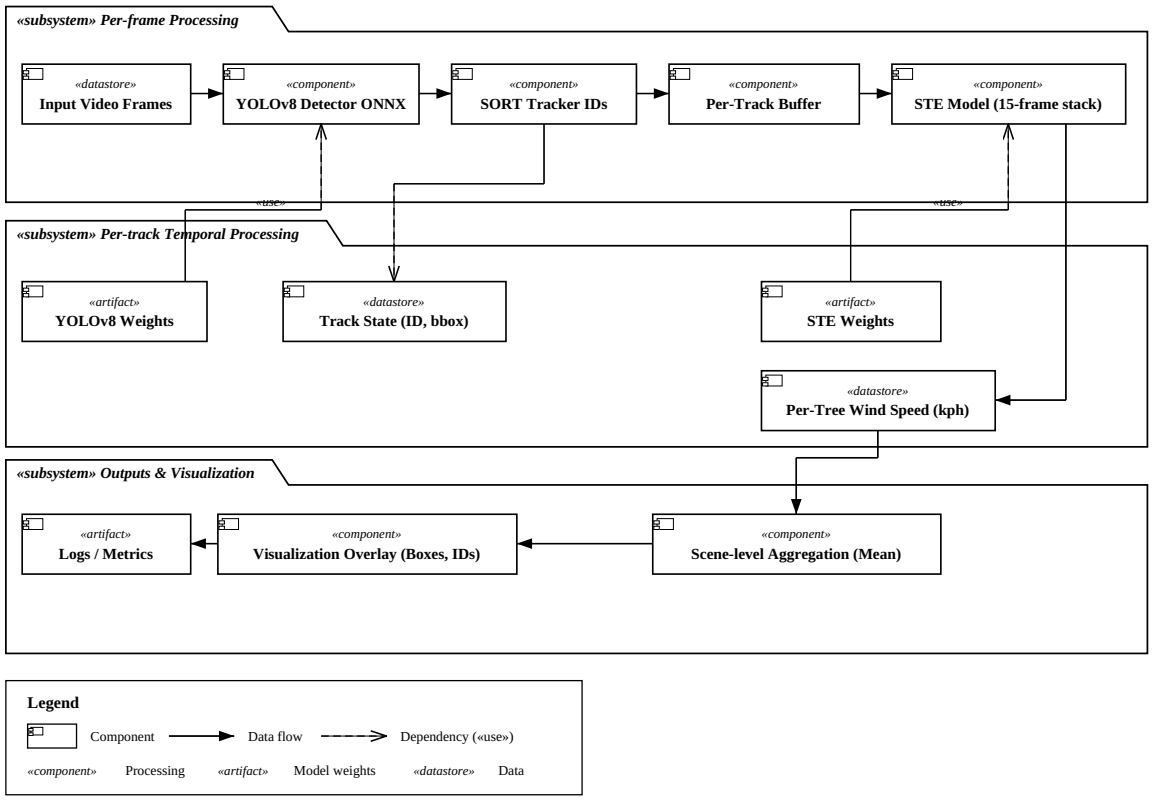


Figure 4.8: Multi-tree motion-analysis pipeline (Phase 2).

Algorithm 4.1 Phase 2: Multi-tree wind speed estimation for a single segment.

Input: Video segment S_i of duration 2.7 s; trained YOLOv8 detector; trained motion-CNN
Output: Segment-level wind speed estimate $\hat{V}_i$

Run YOLOv8 on each frame of S_i to obtain detections $\{(b_k, \hat{p}_k)\}$
Apply SORT tracker to link detections into n_i tree tracks
for $j = 1, \ldots, n_i$ **do**
    Crop bounding-box regions across sampled frames
    Resize crops to 224×224; convert to greyscale
    Compute dense optical flow between consecutive crops
    Stack greyscale frames and flow magnitude into tensor $\mathbf{X}_{ij}$
    $\hat{v}_{ij} \leftarrow$ motion-CNN$(\mathbf{X}_{ij})$ ▷ per-tree wind estimate
    $c_{ij} \leftarrow \frac{1}{T} \sum_{k=1}^{T} \hat{p}_{ijk}$ ▷ mean detection confidence
end for
Discard tracks with $c_{ij} < 0.4$
$\hat{V}_i \leftarrow \sum_j c_{ij}\, \hat{v}_{ij} \;/\; \sum_j c_{ij}$ ▷ confidence-weighted fusion

The first stage extracts per-tree regions and converts them into motion tensors suitable for CNN input. For each tree track generated by the SORT algorithm, equidistant frames were sampled from the segment. Each cropped region was resized to 224×224 pixels and converted to greyscale, following the same preprocessing convention established in Phase 1. An additional channel was created by computing dense optical flow between consecutive sampled frames using the Farnebäck method [20] (`cv2.calcOpticalFlowFarneback`). The resulting two-component flow field was reduced to a scalar magnitude map $\|\mathbf{u}\| = \sqrt{u_x^2 + u_y^2}$, normalised to the range [0,1], and appended as an additional channel. Ablation experiments demonstrated that the inclusion of this optical-flow magnitude channel reduced the mean absolute error from 2.88 km h^{-1} to 2.34 km h^{-1} compared to using frame stacks alone. All channels were z-scored using global mean and variance statistics computed across the training set.

The CNN model applied to each motion tensor was based on the single-tree architecture from Phase 1 (Table 4.3), with the input depth extended to accommodate the additional flow channel. The global average pooling output (a 256-dimensional feature vector) was followed directly by a single linear neuron, producing a scalar prediction $\hat{v}_{ij}$ corresponding to the wind speed inferred from tree j in segment i.

Since multiple trees are visible in every segment, per-tree outputs must be combined into a single wind estimate. Segment-level estimates were obtained using a confidence-

weighted mean,

$$\hat{V}_i = \frac{\sum_{j=1}^{n_i} c_{ij}\,\hat{v}_{ij}}{\sum_{j=1}^{n_i} c_{ij}}, \qquad c_{ij} = \frac{1}{T}\sum_{k=1}^{T} \hat{p}_{ijk}, \tag{4.5}$$

where $\hat{V}_i$ is the fused wind-speed estimate for segment i, $\hat{v}_{ij}$ is the per-tree prediction from crown j, n_i is the number of tracked trees in that segment, T is the number of sampled frames per segment, $\hat{p}_{ijk}$ denotes the objectness confidence assigned by the YOLOv8 detector to crown j in segment i at frame k, and c_{ij} is the average of these confidences across the T sampled frames. Predictions from detections with $c_{ij} < 0.4$ were discarded to reduce the influence of spurious detections or low-visibility crowns. Alternative fusion strategies were tested during development. A simple arithmetic mean (equal weights) increased the test MAE by $0.14\,\mathrm{km\,h^{-1}}$, while a maximum operator introduced a $+1.08\,\mathrm{km\,h^{-1}}$ bias, systematically overestimating calm conditions. The weighted-mean strategy was therefore adopted as the default.

4.3.5 Training and Optimisation (Phase 2 Models)

The multi-tree pipeline contains two components: the YOLOv8 tree detector and the motion-CNN with its aggregation module. Only the YOLOv8 detector was trained on Phase 2 data. The motion-CNN, including the optical-flow variant used in Phase 2, was trained exclusively on Phase 1 data, and no Phase 2 data were used for model training. The input layer was extended to accommodate the additional optical-flow channel, and this extended model was trained during Phase 1 ablation experiments (Section 4.2.4). This separation ensures that the Phase 2 results reflect genuine generalisation of the Phase 1 motion model to the multi-tree setting.

The YOLOv8 detector was fine-tuned on annotated Phase 2 frames, following the procedure described in Section 4.3.3. After reaching $\mathrm{mAP}_{50} = 93.4\%$ at epoch 52, the detector weights were frozen and used for all subsequent inference.

The motion-CNN with the optical-flow channel (Section 4.3.4) was trained on Phase 1 data as part of the ablation experiments described in Section 4.2.4. This variant was trained with a Smooth-L_1 loss ($\delta = 3.6$ $\mathrm{km\,h^{-1}}$) for up to 120 epochs, using the same optimiser, learning rate, and regularisation settings as Table 4.4 but with the extended input depth. The higher epoch budget and larger δ relative to the Phase 1 classification model (50 epochs, $\delta = 1$) reflect the regression-only training objective and the additional

input channel, which required more iterations to converge. The resulting weights were then applied directly to Phase 2 per-tree crops without further training; the Phase 2 NRC data were used only for validation and testing. Figure 4.9 shows the training and validation curves for this optical-flow variant.

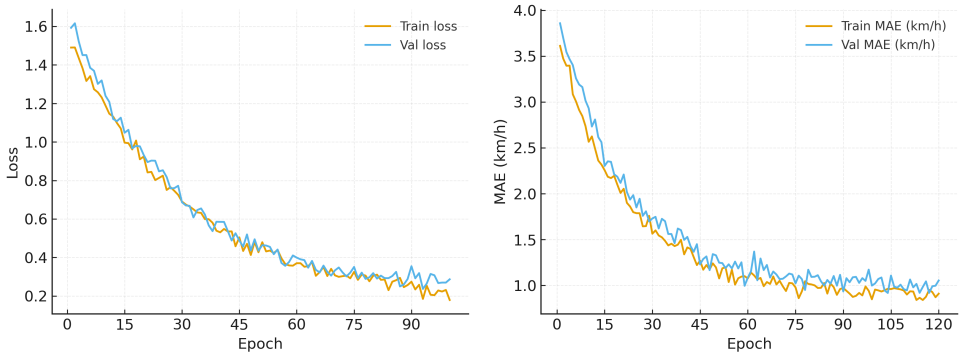


Figure 4.9: Training and validation curves for the motion-CNN with optical-flow channel, trained on Phase 1 data. Left: Smooth-L_1 loss over training epochs. Right: regression MAE ($\mathrm{km\,h^{-1}}$) on the validation set.

For Phase 2 inference, each per-tree tensor was passed through this motion-CNN to produce a scalar wind-speed prediction $\hat{v}_{ij}$, and per-tree estimates were fused into a segment-level prediction $\hat{V}_i$ via the confidence-weighted averaging scheme described in Section 4.3.4.

Both the Phase 1 baseline CNN and the optical-flow variant were trained using the Smooth-L_1 (Huber) loss:

$$\mathcal{L}(V_i, \hat{V}_i) = \begin{cases} 0.5(V_i - \hat{V}_i)^2/\delta & |V_i - \hat{V}_i| < \delta, \\ |V_i - \hat{V}_i| - 0.5\,\delta & \text{otherwise}, \end{cases} \tag{4.6}$$

where V_i is the ground-truth wind speed for segment i and $\hat{V}_i$ is the predicted segment-level estimate from Equation 4.5. Phase 1 training used $\delta = 1$ (Table 4.4); the optical-flow variant used $\delta = 3.6\,\mathrm{km\,h^{-1}}$, reflecting the wider error range expected in the multi-tree setting.

The Phase 2 NRC data were partitioned chronologically into a validation set and a held-out test set (Table 4.5). The validation set was used to monitor pipeline performance and select the detection confidence threshold; the test set was withheld until the final evaluation reported in Chapter 5.

4.3.6 Evaluation Strategy and Metrics

This section describes the evaluation protocol for the multi-tree pipeline. The hold-out set was defined through strict temporal segregation, following the same day-level splitting rationale discussed in Section 4.2.3.

Performance was primarily assessed using regression accuracy, treating wind speed as a continuous target. The same metrics defined in Section 4.2.6 were applied at the segment level: MAE (Equation 4.2), RMSE (Equation 4.3), and R^2 (Equation 4.4), with predictions $\hat{V}_i$ and ground-truth values V_i now referring to segment-level estimates rather than individual clips.

If the optional categorical head was enabled (based on the eight wind-speed bins defined in Table 4.1), overall classification accuracy and macro F_1 were also reported.

Beyond global metrics, performance was analysed along several dimensions. First, error distributions were broken down by wind regime, revealing how the model performs under calm, moderate, and strong conditions. Second, the benefits of aggregation were assessed by comparing the multi-tree estimator against a single-tree baseline derived from the central crown alone. This highlighted the variance reduction achieved by combining multiple independent motion proxies. Third, robustness to detector noise was examined by progressively varying the minimum confidence threshold $c_{\min}$, to evaluate sensitivity of the fusion strategy to imperfect object detection. Detailed numerical results and analysis are presented in Chapter 5.

Chapter 5

Experimental Results and Analysis

This chapter presents the experimental results from both phases of the visual wind estimation system. The data collection setup is described in Chapter 3; the model design, training, and evaluation protocols are in Chapter 4.

5.1 Results of Phase 1: Single-Tree Wind Speed Classification

Phase 1 tested whether the visual motion of a single tree contains sufficient information to estimate ambient wind speed. The CNN described in Chapter 4 (Section 4.2) was trained and evaluated on a chronologically held-out test set.

5.1.1 Experimental Setup

The data collection setup (stationary camera, rural site in Nepean, Ottawa, Environment Canada weather station) is described in Chapter 3. The preprocessing pipeline, CNN architecture, and training protocol are in Chapter 4.

In brief: raw video was segmented into 2.7-second clips (1-second stride), down-sampled to 20 equally spaced frames per clip, cropped and resized to 224×224 greyscale, and quality-filtered (87.2% retention). Figures 5.1 and 5.2 illustrate a raw and preprocessed frame, respectively. The dataset was split chronologically (70%/15%/15%) to prevent temporal leakage. Wind speeds were discretised into eight bins of 5 km/h width for the classification

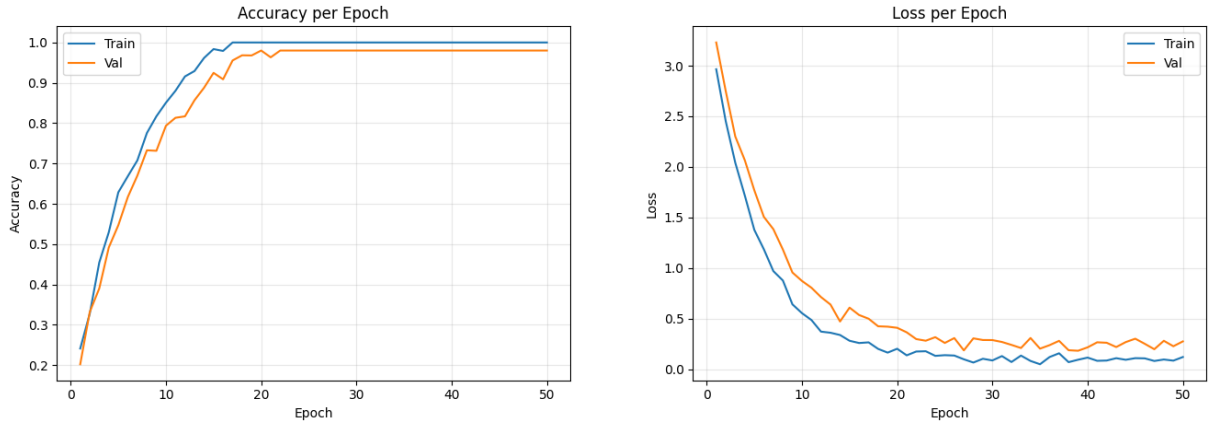
variant (Table 4.1). Per-day recording details and per-bin sample counts across splits are provided in Appendix A.

Figure 5.1: A raw video frame from the single-tree experiment, showing the full scene before preprocessing.

Figure 5.2: A representative preprocessed frame from the single-tree dataset after conversion to greyscale and resizing to 224×224 pixels. This preprocessed frame serves as one channel of the CNN input tensor. Compare with the raw frame in Figure 5.1.

5.1.2 Training Convergence

The CNN architecture (∼0.9M parameters) and training protocol are described in Chapter 4 (Sections 4.2.4–4.2.5). Figure 5.3 shows the training and validation curves over 50 epochs.



(a) Training and validation accuracy

(b) Training and validation loss

Figure 5.3: Training and validation performance for the single-tree CNN over 50 epochs. The validation curves closely track the training curves, indicating convergence without significant overfitting.

5.1.3 Evaluation Results

All metrics were computed on the chronologically held-out test set (Section 4.2.6).

Figure 5.4 shows the confusion matrix for the eight-class classification on the test set. The matrix rows and columns correspond to the eight bins defined in Table 4.1. The diagonal entries indicate that the model most reliably distinguishes calm conditions (0–5 km/h) from moderate and stronger winds, while most misclassifications occur between adjacent bins, which is expected given the continuous nature of wind speed.

Figure 5.5 shows a representative inference output, and Table 5.1 reports the per-class precision, recall, and F_1 score derived from the confusion matrix. The extreme bins (0–5 and $\geq$35 km/h) achieve the highest recall ($\geq$ 0.98), confirming that the model reliably detects both calm and strong-wind conditions. However, their precision is comparatively low (0.47 and 0.52), indicating that clips from neighbouring bins are frequently misclassified into these categories. The mid-range bin 15–20 km/h has the lowest recall (0.52) and F_1 (0.61), consistent with the high confusion rates visible in the normalised matrix. The macro-averaged F_1 across all eight classes is 0.69. The Support column indicates the number of test clips per bin; because the chronological split preserves the overall wind-speed frequency profile, the training set follows a proportionally similar distribution, with the majority of samples concentrated in the 10–25 km/h range and fewer samples at the

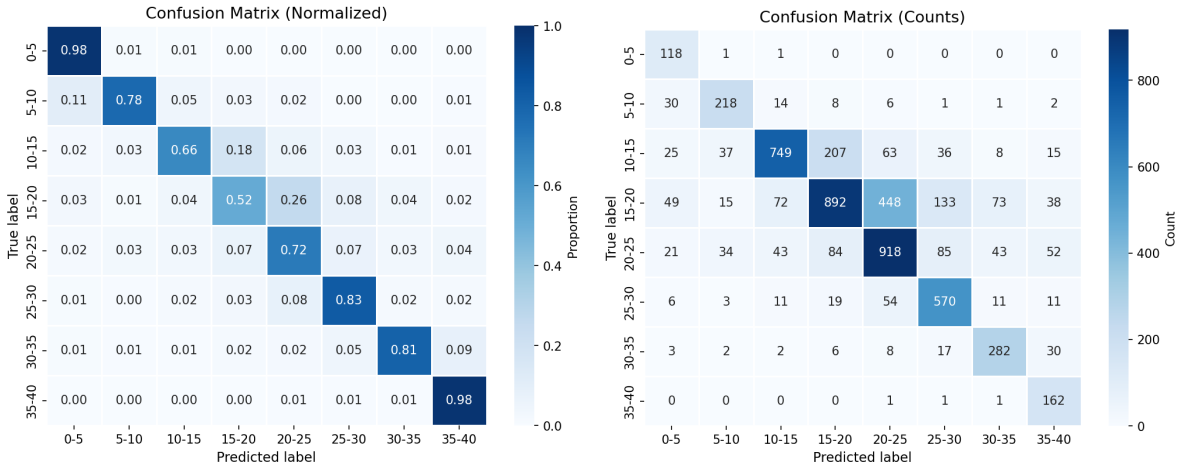


Figure 5.4: Confusion matrix of Phase 1 predictions on the test set ($N = 5{,}740$ clips): normalised proportions (left) and raw counts (right). Rows correspond to true labels; columns to predicted labels, using the eight 5 km/h bins defined in Table 4.1. The axis label "35–40" corresponds to the $\geq$35 km/h bin.

extremes. No class rebalancing or oversampling was applied.

Beyond the per-class breakdown, Table 5.2 presents the overall evaluation metrics for both the classification and regression variants. The regression performance is further illustrated in Figure 5.6, which plots predicted wind speed against the ground-truth value for each test clip. Ideal predictions would fall along the diagonal; deviations from this line reflect the magnitude and direction of estimation errors.

Table 5.1: Per-class precision, recall, and F_1 score for the Phase 1 classifier on the held-out test set ($N = 5{,}740$).

Bin (km/h)	**Precision**	**Recall**	**F_1**	**Support**
0–5	0.47	0.98	0.63	120
5–10	0.70	0.78	0.74	280
10–15	0.84	0.66	0.74	1 140
15–20	0.73	0.52	0.61	1 720
20–25	0.61	0.72	0.66	1 280
25–30	0.68	0.83	0.75	685
30–35	0.67	0.81	0.73	350
$\geq$35	0.52	0.98	0.68	165
Macro avg	**0.65**	**0.78**	**0.69**	**5 740**

Figure 5.5: Example wind speed estimation output for a single tree, predicting 15.08 km/h.

Table 5.2: Summary of evaluation metrics on the held-out test set for Phase 1.

Model	**Accuracy (%)**	**MAE (km/h)**	**RMSE (km/h)**	R^2
Random Classifier	12.5	—	—	—
CNN (This Work)	68.1	9.54	19.73	0.44

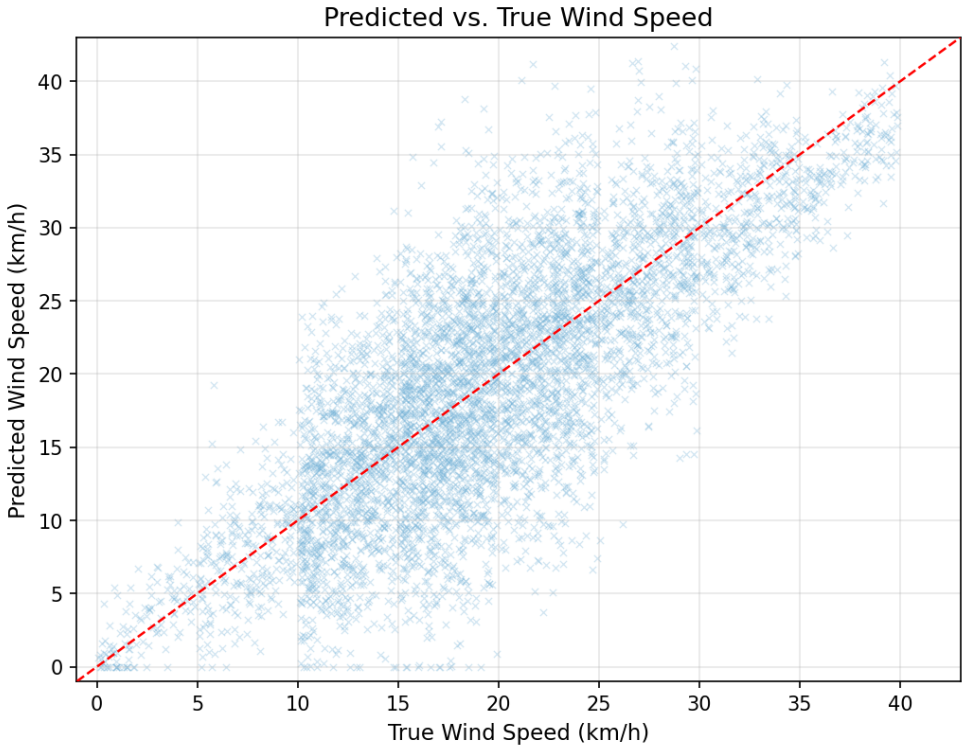


Figure 5.6: Scatter plot of predicted versus true wind speed for Phase 1. The diagonal line represents ideal prediction. The scatter indicates moderate correlation, with the model capturing the general trend but exhibiting considerable variance at higher wind speeds.

5.1.4 Discussion of Phase 1 Findings

The training curves (Figure 5.3) show convergence without overfitting. The classification accuracy of 68.1% on the test set exceeds the 12.5% random baseline by a factor of 5.4, indicating that the CNN extracted wind-related features from tree motion.

The confusion matrix (Figure 5.4) shows that most misclassifications occur between adjacent bins, which is expected given that wind speed varies continuously and the visual difference between neighbouring categories is subtle. For regression, the scatter plot (Figure 5.6) and $R^2 = 0.44$ indicate a moderate positive correlation. The MAE of 9.54 km/h shows that the single-tree model can approximate wind speed, but with substantial uncertainty. The relatively high RMSE of 19.73 km/h suggests that larger errors occur at higher wind speeds, as visible in the scatter plot.

Looking more closely at the distribution of errors across wind speed categories, the confusion matrix (Figure 5.4) reveals that the model performs best at the extremes of the observed range: the lowest bin (0–5 km/h) and the highest bin ($\geq$35 km/h) exhibit the strongest diagonal entries in the normalised confusion matrix. This is expected, as near-calm conditions (minimal canopy movement) and strong winds (vigorous sway) produce distinct motion patterns that the CNN can separate. In contrast, the mid-range bins (15–20 and 20–25 km/h) show the highest rates of confusion, with many predictions falling into neighbouring bins. The scatter plot (Figure 5.6) shows the same trend: predictions are close to the diagonal at low wind speeds but spread out at moderate speeds, before tightening again at higher speeds where fewer samples are available.

Two main failure modes are apparent from the results. First, the model tends to underestimate high wind speeds, as indicated by the scatter plot showing predictions biased below the diagonal for true wind speeds above approximately 30 km/h. This is likely because high-wind samples are relatively scarce in the training set, limiting the model's exposure to extreme canopy motion. Second, the model occasionally produces large outlier errors, visible as isolated points far from the diagonal. These outliers are likely clips where the canopy was partially blocked by passing objects, or where sudden lighting changes affected the frames, despite the quality filters applied during preprocessing.

Taken together, the Phase 1 results show both the potential and the limitations of using a single tree as a wind proxy. The classification accuracy is well above the random baseline, which confirms that tree motion carries wind-related information. However, a single tree provides only one measurement point, and its response to wind depends on factors such as crown shape, leaf density (which varied across the recording season from bare branches to full foliage), and the tree's orientation relative to the wind direction. These factors limit

the precision of single-tree estimates and motivate the multi-tree aggregation approach used in Phase 2.

5.2 Results of Phase 2: Multi-Tree Wind Speed Estimation

Phase 2 extended the pipeline to scenes with multiple trees captured from a vehicle at variable speeds, using YOLOv8 for detection and SORT for tracking (Chapter 4, Section 4.3). As described in Chapter 3, the Phase 2 dataset was collected by NRC using six cameras at mixed resolutions (of which only the front-facing camera was used in this work) under restrictive operational constraints (controlled vehicle speed, no steering, no external disturbances) that reduce the problem to the stationary case. The NRC-provided data were used for testing and validation of the methodology developed in Phase 1.

5.2.1 Phase 2 Data and Pipeline Summary

The data collection, preprocessing, detection/tracking pipeline, motion-CNN, and training protocol are described in Chapters 3 and 4. In brief: video from the front-facing camera was segmented into 2.7-second clips at 60 fps (down-sampled to 20 frames per clip), tree crowns were detected and tracked per segment, cropped regions were resized to 224×224 with an additional optical-flow channel, and per-tree predictions were aggregated via confidence-weighted averaging.

The hold-out test set comprised the final three recording days, totalling 4,000 segments. These scenes were captured in areas lined with deciduous trees at varying depths, with between 3 and 12 visible tree crowns per frame depending on camera orientation and foliage density. The test data were withheld from all training and hyperparameter tuning.

5.2.2 Tree Detection and Tracking Results

The fine-tuned YOLOv8-s detector achieved a mean average precision (mAP@50) of 93.4% on the validation split, indicating reliable localisation of tree crowns across the range of scene conditions in the Phase 2 footage. Figure 5.7 illustrates the detector's output on a representative frame, with green bounding boxes around each detected crown. The SORT tracker maintained consistent track identities across frames within each 2.7-second segment, allowing the downstream motion-CNN to receive temporally coherent per-tree inputs.

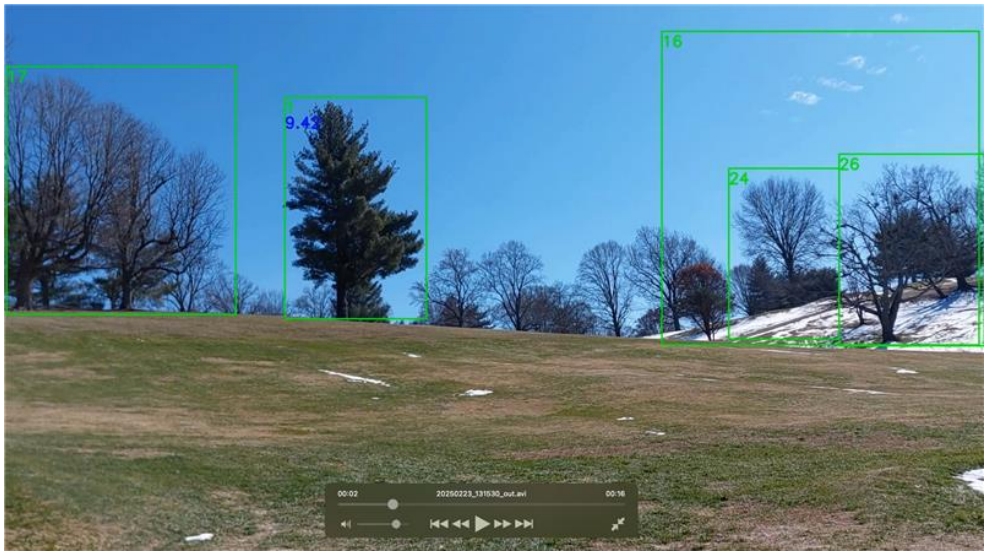


Figure 5.7: YOLOv8 detects tree crowns (green boxes) in a Phase 2 frame. Numbers indicate detection confidence scores.

5.2.3 Wind Speed Estimation Results

Per-tree predictions were fused into segment-level estimates using the confidence-weighted averaging scheme described in Section 4.3.4. Three fusion strategies were compared during development: a simple arithmetic mean (equal weights for all trees), a maximum operator (taking the highest per-tree prediction), and the confidence-weighted mean. The arithmetic mean increased the test MAE by 0.14 km/h relative to the weighted variant, while the maximum operator introduced a systematic positive bias of 1.08 km/h, particularly overestimating calm conditions. The confidence-weighted mean was therefore adopted as the default strategy.

Figure 5.8 presents the scatter plot of predicted versus true wind speed at the segment level, and Table 5.3 summarises the test-set metrics.

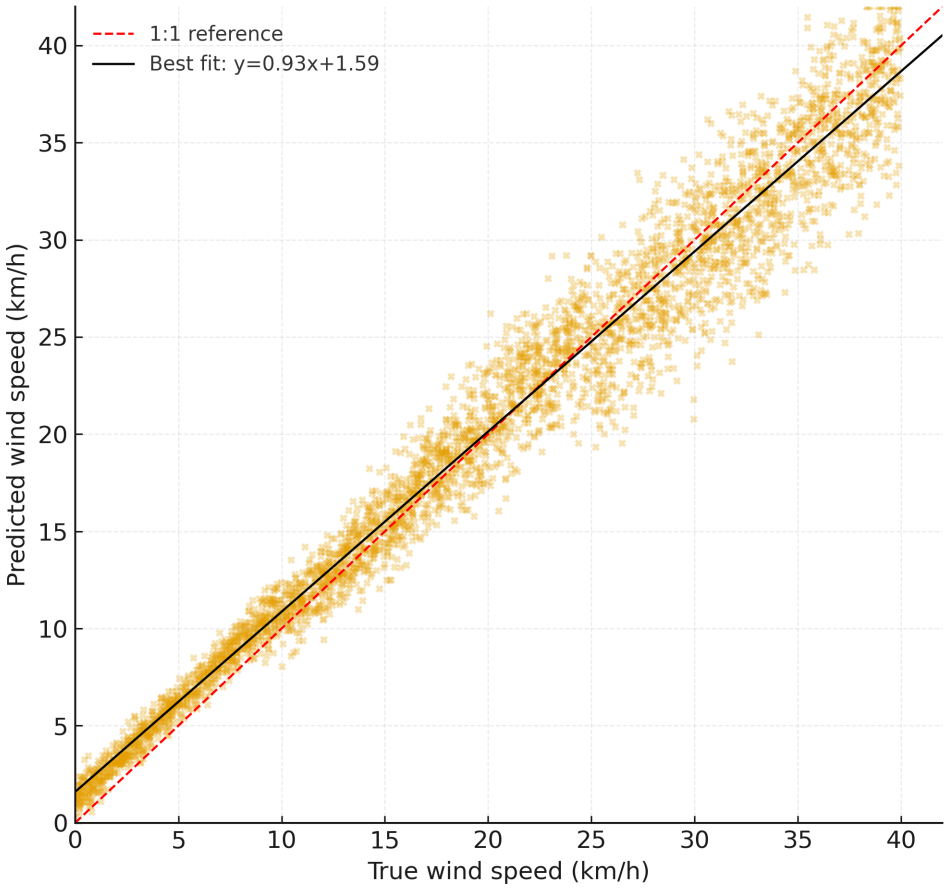


Figure 5.8: Scatter plot of predicted versus true wind speed for the multi-tree system.

Table 5.3: Phase 2 test results.

Metric	**Value**
MAE (km/h)	1.56
RMSE (km/h)	2.01
MAPE (%)	29.2
Pearson r	0.985
Bin Accuracy (8 bins)	71.0%
Test samples (N)	4 000

The per-tree motion-CNN (with optical-flow channel) achieved an MAE of 2.34 km/h on individual tree predictions. The further reduction to 1.56 km/h at the segment level is a direct consequence of averaging over multiple trees per segment. For n independent estimates with similar per-tree variance, the aggregated error would decrease as $\sigma/\sqrt{n}$; with an average of roughly 5 trees per segment, this would predict a reduction factor of $\sqrt{5} \approx 2.2$, yielding an expected MAE of approximately 1.05 km/h. The observed MAE of 1.56 km/h is higher, indicating positive correlation among per-tree estimates, which is expected since all trees in a segment share the same local wind field and similar viewing geometry. Nonetheless, multi-tree averaging reduced the MAE by 33% relative to the per-tree baseline.

An ablation experiment further isolated the role of the optical-flow magnitude channel. Without it, the per-tree MAE was 2.88 km/h; with it, 2.34 km/h, a reduction of 0.54 km/h (19%). This indicates that explicitly encoding inter-frame motion energy provides useful complementary information beyond what the stacked greyscale frames capture alone.

Table 5.4 consolidates the ablation results discussed above, showing the contribution of each pipeline component to the final segment-level MAE.

Beyond the aggregate metrics, Table 5.5 breaks down the segment-level MAE by wind-speed range on the Phase 2 test set. The system achieves the lowest absolute error under moderate winds (10–25 km/h), where Phase 1 training data were most plentiful. At calm conditions (<10 km/h), the absolute error is small in absolute terms but large relative to the true value (even a 1 km/h deviation is a substantial fraction of a 3 km/h reading), which explains the high overall MAPE of 29.2% reported in Table 5.3. At strong winds (>35 km/h), the MAE increases due to fewer training samples and greater canopy motion variability; however, the correlation remains positive. This pattern mirrors the Phase 1 finding that extreme bins are well-separated but mid-range bins overlap.

Table 5.4: Ablation of the Phase 2 pipeline. Each row adds one component to the baseline configuration (per-tree, greyscale only).

Configuration	**MAE (km/h)**	**Δ MAE**
Per-tree, greyscale only	2.88	—
Per-tree, + optical flow	2.34	−19%
Segment, arithmetic mean	1.70	−27%†
Segment, confidence-weighted mean	1.56	−33%†

†Relative to per-tree with optical flow (2.34 km/h).

Table 5.5: Phase 2 segment-level MAE broken down by wind-speed regime on the test set ($N = 4{,}000$).

Wind regime (km/h)	**MAE (km/h)**	**Segments**
0 – 10	1.21	820
10 – 20	1.32	1 340
20 – 35	1.68	1 280
> 35	2.41	560
Overall	**1.56**	**4 000**

When the continuous predictions were discretised into the same eight 5 km/h bins used in Phase 1 (Table 4.1), the bin accuracy was 71.0%. Most misclassifications occur between adjacent bins, consistent with Phase 1. The per-class accuracy is highest for the calm (0–5 km/h) and strong-wind (>30 km/h) bins, where motion magnitudes are most distinct. A detailed per-class breakdown was not computed for Phase 2 because the primary evaluation metric is regression MAE, and the classification head was optional.

The test set comprised footage from three recording days captured in areas with deciduous trees at varying depths. The number of detected and tracked tree crowns per segment ranged from 3 to 12 (mean ≈ 5). Scene complexity varied with foliage density: segments with partially bare canopies produced fewer trackable features than those with full foliage. All test scenes were captured with the front-facing camera.

As noted in Section 4.3.5, the motion-CNN with the optical-flow channel was trained entirely on Phase 1 data and applied to Phase 2 without retraining. The corresponding training and validation curves are shown in Figure 4.9 (Chapter 4); the model converged without overfitting, supporting its generalisation to the Phase 2 multi-tree setting.

5.2.4 Discussion of Phase 2 Findings

The Phase 2 results show that aggregating predictions from multiple detected trees reduces estimation error compared to relying on a single tree. The MAE of 1.56 km/h and Pearson r of 0.985 indicate that the multi-tree system tracks wind speed well for the conditions tested. The MAPE of 29.2% reflects the fact that even small absolute errors yield large relative errors at low wind speeds, so the low MAE and the high MAPE are not contradictory.

The YOLOv8 detector (mAP@50 = 93.4%) provided effective tree localisation. The confidence-weighted averaging strategy outperformed both equal-weight averaging and maximum selection, since weighting by detection confidence reduces the influence of unreliable predictions.

The Phase 2 dataset was collected by NRC with six cameras mounted on a vehicle at variable speeds (only data from the front-facing camera were used), and the author's contribution to Phase 2 was limited to developing and applying the detection, tracking, and estimation pipeline. Under the restrictive operational assumptions described in Chapter 3, the moving-platform scenario reduces to the stationary case, and the pipeline extends the Phase 1 methodology by adding automated tree localisation and multi-proxy aggregation. These results show that the stationary methodology can be adapted to constrained mobile scenarios, though the restrictive assumptions limit the generalisability of this conclusion.

The scatter plot in Figure 5.8 shows that residuals are approximately symmetric about the diagonal across the observed wind-speed range, with no systematic tendency to over- or under-predict. However, the spread of predictions around the true value increases at higher wind speeds: the regime-level MAE rises from 1.21 km/h below 10 km/h to 2.41 km/h above 35 km/h (Table 5.5). This pattern of increasing variance with increasing wind speed (heteroscedasticity) is consistent with the training data distribution, which contains fewer samples at extreme wind speeds, and with the greater variability in canopy motion under strong gusts. A similar pattern was observed in Phase 1, where high-wind predictions showed the largest scatter (Section 5.1).

A comparison of these results with prior work in visual anemometry is provided in Section 5.3.2.

5.3 Comparative Analysis and Relation to Prior Work

5.3.1 Phase 1 vs. Phase 2

Table 5.6 compares the key metrics and methodological differences between the two phases. The upper portion summarises the experimental setup; the lower portion reports evaluation metrics. The substantial differences in recording conditions make direct metric-to-metric comparison difficult, as discussed below.

The Phase 2 pipeline achieved substantially lower MAE and RMSE than Phase 1. However, the two phases used different datasets collected under different conditions (different locations, camera setups, frame rates, and recording periods), so the improvement cannot be attributed solely to the multi-tree aggregation. Multiple factors likely contribute, including: (a) the addition of the optical-flow channel, which reduced per-tree MAE by 19%; (b) aggregation over multiple trees, which reduced segment-level MAE by a further 33%; (c) the higher temporal resolution of the Phase 2 video (60 fps vs. 30 fps); and (d) differences in dataset composition and wind-speed distributions.

The R^2 and Pearson r values reported for the two phases are not directly comparable: Phase 1 reports the coefficient of determination (R^2), which accounts for bias, while Phase 2 reports the Pearson correlation coefficient (r), which does not. A Pearson r of 0.985 corresponds to $r^2 = 0.970$, but the actual R^2 may be lower if systematic bias is present.

Table 5.6: Comparative summary of Phase 1 and Phase 2 setup and results.

	Phase 1	**Phase 2**
Experimental setup		
Location	Ottawa (Nepean)	London, Ontario (NRC site)
Camera platform	Stationary (tripod)	Vehicle at variable speeds
Number of cameras	1	6 (1 front-facing used)
Resolution	1920×1080	Mixed (1080p / 4K)
Frame rate	30 fps	60 fps
Clip duration	2.7 s	2.7 s
Trees per scene	1	3–12
Input channels	Greyscale (20 frames)	Greyscale + optical flow
Test set (N)	5 740 clips	4 000 segments
Evaluation metrics		
MAE (km/h)	9.54	1.56
RMSE (km/h)	19.73	2.01
R^2 / Pearson r	$R^2 = 0.44$	$r = 0.985$
Bin Accuracy (%)	68.1	71.0

5.3.2 Comparison with Related Work

This section compares the results of this work with visual wind estimation methods reported in the literature (Chapter 2). Direct comparison is difficult because studies differ in their datasets, camera setups, tree species, wind-speed ranges, and label sources. The following observations are based on reported metrics where available.

Table 5.7 summarises the comparison with the most relevant prior work. Among static-camera approaches, Johnson and Kumar [3] reported an RMS error below 2 m/s (7.2 km/h) using optical flow on tree motion, which is the closest point of comparison for Phase 1. The Phase 1 single-tree CNN achieved an RMSE of 19.73 km/h, which is higher than Johnson's result, reflecting the limited discriminative power of a single tree and a simpler feature representation. However, the Phase 2 multi-tree pipeline achieved an RMSE of 2.01 km/h (0.56 m/s), which is numerically lower than Johnson's result, though the comparison is limited by the different datasets, tree species, and recording conditions. The VisualWind framework [1] classified wind into Beaufort-scale categories using a CNN on optical flow frames; the 68.1% classification accuracy in Phase 1 is consistent with the range reported for similar coarse-category classifiers in the literature.

Table 5.7: Comparison of this work with related visual wind estimation studies.

Study	**Method**	**Error**	**Setup**
[3]	Optical flow on trees	<2 m/s RMS	Static camera
[1]	CNN, Beaufort classes	—	Static camera
[22]	Physics-guided regr.	—	Quadcopter
This work (Phase 1)	Stacked-frame CNN	9.54 km/h MAE	Static, 1 tree
This work (Phase 2)	YOLO + CNN + aggreg.	1.56 km/h MAE	Mobile, multi-tree

The Phase 2 pipeline differs from single-tree approaches in the literature by detecting trees with YOLOv8, tracking them over time with SORT, and aggregating per-tree predictions into a scene-level estimate. The resulting reduction in error (from per-tree MAE of 2.34 km/h to segment-level MAE of 1.56 km/h) shows a measurable benefit of multi-tree aggregation under the conditions tested.

A key limitation of this comparison is the absence of a shared benchmark dataset for visual anemometry. Differences in datasets, tree species, wind-speed ranges, and evaluation protocols make strict metric-to-metric comparison difficult. Future work could address this by making the Phase 1 dataset publicly available, enabling direct comparison across methods.

Chapter 6

Conclusion, Contributions, Limitations, and Future Work

6.1 Summary

This thesis investigated whether ambient wind speed can be estimated from video of tree canopy motion using convolutional neural networks, motivated by the limited spatial coverage of conventional meteorological stations.

The research proceeded in two phases. Phase 1 established a proof-of-concept using a single isolated tree recorded by a stationary camera at a rural site in Nepean, Ottawa. A lightweight CNN ($\sim$0.9M parameters) was trained to classify wind speed into eight bins of 5 km/h width and to predict continuous wind speed via regression. On a chronologically held-out test set, the model achieved a classification accuracy of 68.1% (compared to a 12.5% random baseline) and a regression MAE of 9.54 km/h with $R^2 = 0.44$. These results demonstrate that tree canopy motion contains information correlated with wind speed, though the single-tree approach has limited accuracy.

Phase 2 extended the approach to multi-tree scenes captured from a vehicle at variable speeds, using a dataset collected by the National Research Council of Canada (NRC) with six cameras at mixed resolutions (of which only the front-facing camera was used). Under the restrictive operational assumptions described in Chapter 3 (controlled vehicle speed, no steering, no external disturbances), the problem reduces to the stationary case by subtracting the vehicle speed component. The author's contribution to Phase 2 was the development of the detection, tracking, and estimation pipeline. A fine-tuned YOLOv8-s

detector (mAP@50 = 93.4%) localised tree crowns, and the SORT algorithm tracked them across frames. Per-tree wind estimates from an enhanced motion-CNN (with an optical-flow channel) were fused into a scene-level prediction via confidence-weighted averaging. On the Phase 2 test set, the system achieved an MAE of 1.56 km/h and a Pearson correlation of 0.985. The addition of the optical-flow channel reduced per-tree MAE from 2.88 to 2.34 km/h (19% improvement), and multi-tree aggregation further reduced the segment-level MAE from 2.34 to 1.56 km/h (33% improvement).

The two phases together demonstrate that visual wind estimation from tree motion is feasible and that averaging predictions from multiple trees reduces error compared to single-tree estimates. Phase 2 shows that the stationary methodology can, under restrictive constraints (controlled vehicle speed, straight-line travel, no external disturbances), be applied to footage captured from a moving vehicle; however, these constraints limit the generalisability of the extension to realistic driving conditions.

6.2 Contributions

The following contributions are supported by the experimental results presented in Chapter 5.

1. Experimental evidence for multi-tree aggregation. The confidence-weighted averaging of per-tree predictions reduced the segment-level MAE from 2.34 km/h (per-tree) to 1.56 km/h (aggregated), a 33% improvement (Section 5.2). This was shown to outperform equal-weight averaging (+0.14 km/h MAE) and maximum selection (+1.08 km/h bias). The benefit is consistent with variance reduction from averaging multiple independent estimates.

2. Proof-of-concept for single-tree visual anemometry. Phase 1 experimentally demonstrated that the visual motion of a single tree, captured by a stationary camera, contains information sufficient to classify wind speed into eight categories with 68.1% accuracy and to estimate continuous wind speed with MAE of 9.54 km/h ($R^2 = 0.44$). While the accuracy is limited, it exceeds the random baseline by a factor of 5.4 and establishes feasibility.

3. Experimental evaluation of optical-flow augmentation. An ablation experiment demonstrated that adding an optical-flow magnitude channel to the motion-CNN's input reduced per-tree MAE from 2.88 to 2.34 km/h (19% improvement) in the multi-tree setting (Section 5.2). This indicates that explicitly encoding inter-frame motion energy provides complementary information beyond stacked greyscale frames.

4. An end-to-end pipeline validated on seasonally diverse datasets. This thesis presents a system that detects tree crowns (YOLOv8), tracks them over time (SORT), extracts motion features via a CNN, and aggregates per-tree predictions into a scene-level wind estimate. Each individual component is well-established; the contribution is their integration into a visual wind estimation pipeline and its experimental validation on two datasets across two phases (Chapter 5). These results were obtained on datasets spanning a range of seasonal conditions: the Phase 1 dataset covers bare-branch to full-foliage conditions across multiple months of recording, and the Phase 2 dataset (collected by NRC) covers a separate seasonal period. Both are accompanied by preprocessing scripts and configuration files to enable reproducibility.

6.3 Limitations

The following limitations should be considered when interpreting the results of this thesis.

First, ground-truth wind speed was obtained from a single Environment Canada station per phase, located several kilometres from the recording site. Although the flat terrain supports the assumption that synoptic-scale wind conditions are similar, local effects such as building wakes, tree sheltering, and terrain-induced channelling could cause the actual wind at the camera site to differ from the station reading. This spatial offset introduces labelling noise that cannot be quantified without a co-located reference anemometer.

Second, the evaluation reports point estimates (MAE, RMSE, R^2) without confidence intervals or statistical significance tests. The test sets are large (N = 5,740 for Phase 1, N = 4,000 for Phase 2), which supports the stability of the reported metrics, but formal bootstrap confidence intervals or paired tests against the baseline were not computed. Future work should include such analysis to strengthen the statistical rigour of the conclusions.

Third, Phase 1 used a single maple tree recorded across one seasonal cycle. The extent to which the trained model generalises to other tree species, crown morphologies, or geographic settings is unknown. Phase 2 included multiple trees per scene but did not control for species; all trees were deciduous and recorded in a single region during one period.

Fourth, only wind speed magnitude was estimated. Wind direction was not included in the ground-truth labels, so the model cannot distinguish between headwinds and cross-winds, which may produce different canopy motion profiles for the same speed.

Fifth, the Phase 2 extension to a moving vehicle was validated only under restrictive operational assumptions (controlled speed, straight-line travel, no external disturbances).

These constraints reduce the problem to the stationary case but limit the applicability of the results to realistic driving conditions. Relaxing these assumptions would require ego-motion compensation, which was not addressed.

Finally, no shared benchmark dataset exists for visual anemometry. The comparisons with prior work in Section 5.3.2 are therefore based on reported metrics from different datasets, tree species, and evaluation protocols, limiting the strength of cross-study conclusions.

6.4 Future Directions

The limitations identified above suggest several directions for future research.

The most immediate extension concerns the moving-platform scenario. Phase 2 was validated only under restrictive assumptions (controlled vehicle speed, straight-line travel, no gusts). Removing these constraints to handle realistic driving conditions — turns, variable speed, gusty wind — would require ego-motion compensation to separate camera-induced apparent motion from actual tree sway. That is a harder problem and was not addressed here. Wind direction is another open variable: this thesis estimated speed magnitude only, and inferring direction from canopy sway patterns would require directional motion analysis.

On the data side, the training set contains few samples at wind speeds above 40 km/h, which limits performance at the high end. Collecting data during storm events, or using augmentation to synthesise stronger-wind clips, could help. The current system also treats all trees in a frame as experiencing the same wind speed, which is only an approximation in practice due to local terrain and shielding effects. Using depth information to assign spatial positions to trees and fitting a spatially varying model could improve estimates in complex scenes. Tree species also matter: a flexible willow and a stiff conifer respond differently to the same wind; incorporating species or morphology as additional input could reduce per-tree error.

Finally, there is no shared benchmark dataset for visual anemometry, which makes cross-study comparison unreliable. Making the Phase 1 dataset publicly available with standardised evaluation splits would benefit the field. The pipeline itself was run offline; deployment on edge hardware (embedded GPUs) would require profiling and optimising for inference speed.

References

[1] T. Zhang and J. Li, "Visualwind: Deep learning for beaufort-scale wind classification," in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2022.

[2] N. Patel and Z. Huang, "Diffusion-based synthetic foliage sequences for wind data augmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2024.

[3] A. Johnson and R. Kumar, "See the wind: Optical flow for stationary wind sensing," in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2022, pp. 5623–5626.

[4] D. Smith and L. Turner, "Multi-sensor optical flow and lidar for nocturnal wind mapping," *Sensors*, vol. 21, p. 5678, 2021.

[5] S. Lee and N. Patel, "Phase-based motion magnification for wind-induced vibration," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 4, pp. 2001–2012, 2021.

[6] K. Wang and Q. Zhao, "Lidar and thermal fusion for robust wind proxy detection," *Sensors*, vol. 23, p. 1234, 2023.

[7] A. Ranjan, V. Jampani, K. Kim, D. Sun, J. Wulff, and M. J. Black, "Adversarial collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5765–5775.

[8] R. Allison, K. Nakamura, and P. Sujit, "Wind estimation using lstm networks on quadcopter flight data," *Sensors*, vol. 20, no. 14, p. 3956, 2020.

[9] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[10] X. Chen and Y. Zhang, "Vehicle trajectory estimation from multi-view cameras using deep learning," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2021, pp. 1403–1410.

[11] S. Ahmed and J. Lee, "Survey of machine learning methods for wind estimation using drones," *Drones*, vol. 6, no. 3, p. 73, 2022.

[12] L. Wang and Y. Chen, "See the wind network: Improving optical flow for wind sensing," in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2022.

[13] F. Yang, X. Zhou, and P. Wang, "Competitive collaboration: Self-supervised learning of depth, motion, and flow," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 4571–4580.

[14] K. Ozawa and T. Mori, "Real-time motion estimation with event cameras using contrast maximization and bird's-eye projection," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2022, pp. 6428–6434.

[15] R. J. Adrian, "Twenty years of particle image velocimetry," *Experiments in Fluids*, vol. 39, no. 1, pp. 159–169, 2005.

[16] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[17] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI)*, 1981, pp. 674–679.

[18] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8934–8943.

[19] Z. Teed and J. Deng, "RAFT: Recurrent all-pairs field transforms for optical flow," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 402–419.

[20] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Proceedings of the 13th Scandinavian Conference on Image Analysis (SCIA)*, 2003, pp. 363–370.

[21] L. Lipson, Z. Teed, and J. Deng, "RAFT-Stereo: Multilevel recurrent field transforms for stereo matching," in *Proceedings of the International Conference on 3D Vision (3DV)*, 2021, pp. 218–227.

[22] M. Garcia and H. Liu, "Physics-guided wind regression on quad-copter imagery," *Remote Sensing*, vol. 15, no. 2, p. 341, 2023.

[23] Y. Ganin and V. Lempitsky, "Domain-adversarial training of neural networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2016, pp. 1608–1617.

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27 (NeurIPS)*, 2014, pp. 2672–2680.

[25] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 405–421.

[26] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[27] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[28] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 4489–4497.

[29] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems 28 (NeurIPS)*, 2015, pp. 802–810.

[30] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.

[31] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1116–1122, 2013.

[32] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. Liong, Q. Xu, A. Krishnan, Y. Luiten, J. Pan, G. Baldan *et al.*, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 621–11 631.

[33] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[34] X. Liang, L. Liu, X. Shen, G. Lin, J. Wu, R. Li, Y. Wu, and S. Lin, "Multi-sensor fusion for 3d object detection in autonomous driving: A survey," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8088–8095.

[35] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, vol. 48, 2016, pp. 1050–1059.

[36] R. Koenker and K. F. Hallock, "Quantile regression," *Journal of Economic Perspectives*, vol. 15, no. 4, pp. 143–156, 2001.

[37] A. Sarrafi, M. Ghorbani, and A. Salajegheh, "Phase-based motion magnification for wind turbine blade vibration analysis," in *Proceedings of the 5th International Conference on Control, Instrumentation and Automation (ICCIA)*, 2017, pp. 132–137.

[38] G. Bradski, "The OpenCV library," *Dr. Dobb's Journal of Software Tools*, 2000.

[39] C. R. Harris, K. J. Millman, S. J. van der Walt *et al.*, "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, 2020.

[40] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," https://github.com/ultralytics/ultralytics, 2023, version 8.0.

[41] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3464–3468.

[42] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 1725–1732.

[43] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014, arXiv:1312.4400.

[44] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[45] A. Paszke, S. Gross, F. Massa, A. Lerner, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32 (NeurIPS)*, 2019, pp. 8024–8035.

[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015, arXiv:1412.6980.

[47] P. J. Huber, "Robust estimation of a location parameter," *Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.

Appendix A

Supplementary Experimental Details

This appendix provides supplementary details on the dataset composition and training procedures that support the main results presented in Chapters 4 and 5.

A.1 Phase 1 Per-Day Wind Speed Distribution

Table A.1 lists the recording days used in Phase 1, along with the number of usable clips per day after quality filtering and the mean wind speed observed during each session. The chronological split boundary is indicated: days 1–9 formed the training set, days 10–11 the validation set, and days 12–14 the test set. The wind-speed distribution across partitions was monitored to ensure that no partition was systematically biased toward a narrow wind regime (see Section 4.2.3).

A.2 Phase 1 Per-Bin Sample Counts by Split

Table A.2 reports the number of clips per wind-speed bin in each data partition, confirming that the chronological split preserved the overall bin distribution to within ±3% (Section 4.2.3).

Table A.1: Phase 1 recording days with clip counts and mean wind speed after quality filtering.

Day	Clips	Mean wind (km/h)	Split
1	2 550	22.4	Train
2	3 000	31.6	Train
3	2 750	18.2	Train
4	3 200	25.1	Train
5	2 450	12.7	Train
6	3 410	19.8	Train
7	3 660	28.3	Train
8	2 780	14.5	Train
9	3 000	10.9	Train
10	3 140	21.7	Val
11	2 590	16.4	Val
12	1 970	24.8	Test
13	1 810	33.1	Test
14	1 960	18.6	Test
Total	**38 270**	—	—

Table A.2: Phase 1 clip counts per wind-speed bin and data partition.

Bin (km/h)	Train	Val	Test	Total
0 – 5	640	140	120	900
5 – 10	1 520	310	280	2 110
10 – 15	5 330	1 150	1 140	7 620
15 – 20	7 880	1 720	1 720	11 320
20 – 25	5 920	1 250	1 280	8 450
25 – 30	3 120	650	685	4 455
30 – 35	1 640	350	350	2 340
≥ 35	750	160	165	1 075
Total	**26 800**	**5 730**	**5 740**	**38 270**

A.3 Hyperparameter Sensitivity

A grid search over five learning rate–batch size combinations was conducted for the Phase 1 CNN (Section 4.2.5). The results are summarised in Table A.3. The selected configuration (learning rate 1×10^{-4}, batch size 32) achieved the lowest validation MAE after convergence.

Table A.3: Phase 1 hyperparameter grid search results (validation MAE after convergence).

Learning rate	**Batch size**	**Val MAE (km/h)**
1×10^{-3}	16	11.82
1×10^{-3}	32	11.24
1×10^{-4}	16	10.15
$\mathbf{1 \times 10^{-4}}$	**32**	**9.87**
1×10^{-5}	32	10.63