Lightweight Learning Based Feature Selection for Real Time Optical Flow Navigation.

Ali Sherif Ali Abosaad

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

GRADUATE PROGRAM IN
EARTH AND SPACE SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

APRIL 2026

Abstract

Accurate state estimation is essential for autonomous quadrotor navigation in GPS-denied environments. Although Visual-Inertial Odometry (VIO) systems perform well, their accuracy depends on the quality and stability of visual features. Classical detectors are efficient but often produce redundant or unstable features, while learning-based methods are more robust but computationally expensive for embedded platforms.

This thesis presents a lightweight fully onboard navigation framework that improves estimation accuracy by enhancing feature quality rather than feature density. A compact adaptive convolutional neural network predicts feature trackability and removes unreliable features before Lucas–Kanade optical flow tracking. An adaptive threshold regulates the retained feature count. Visual velocity estimates, with inertial rotational compensation, are fused with IMU and range measurements using an Extended Kalman Filter.

Experiments on the Quanser QDrone2 show reduced computational cost, improved estimation accuracy, and real-time performance above 60 Hz.

To my professor, my parents, and my fiancée.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Professor Jinjun Shan, for his invaluable guidance, continuous support, and trust throughout my Master's research. I am sincerely thankful for the opportunity he provided to pursue this research and for his mentorship, which has greatly contributed to my academic and professional development.

I would also like to thank my committee member, Professor Regina Lee, for her time, thoughtful feedback, and valuable suggestions, which helped improve the quality and clarity of this thesis.

I am grateful to my lab mates Ahmed Etman, Pukun Lu, Amaldev Haridevan, Hunter Schofield, Yida Zang, SyedReza, Dr. Junjie Kang, Hao Zhang, Hao Wang and special thanks for Dr. Hassan Alkomy for their collaboration, discussions, and technical support. Working alongside such a dedicated and supportive group made this research both productive and enjoyable, and I appreciate the contributions and encouragement of everyone in the lab.

I would like to express my heartfelt appreciation to my parents, my brother and my family for their unconditional love, sacrifices, and constant encouragement.

Finally, I would like to thank my friends for their understanding, and encouragement, which helped me maintain balance during this demanding period.

Most importantly, I am deeply grateful to my fiancée Aya for her patience, love, and unwavering support. Her belief in me and her constant encouragement have meant more than words can express and have been a continual source of motivation throughout this journey.

Table of Contents

List of Tables

List of Figures

Nomenclature

General Notation

$\mathbb{R}$	Set of real numbers
$\mathbb{R}^n$	n-dimensional real vector space
$\|\cdot\|$	Euclidean norm
$(\cdot)^\top$	Vector or matrix transpose
$E[\cdot]$	Expectation operator
Δt	Time interval between consecutive measurements

Coordinate Frames

F_g	Global (inertial) reference frame defined by the OptiTrack system
F_b	Body frame attached to the quadrotor center of mass
F_c	Camera frame attached to the downward-facing monocular camera
$R_{gb} \in SO(3)$	Rotation matrix from body frame to global frame
$R_{bc} \in SO(3)$	Rotation matrix from camera frame to body frame

Camera and Image Quantities

(u_i, v_i)	Pixel coordinates of the i-th feature in the image plane
$\mathbf{p}_i$	Image-plane position of feature i, $\mathbf{p}_i = [u_i, v_i]^\top$
f_x, f_y	Camera focal lengths in pixel units
c_x, c_y	Camera principal point coordinates
$\Delta u_i, \Delta v_i$	Optical flow displacement of feature i in pixels

$I(u,v,t)$	Image intensity at pixel (u,v) and time t
I_u, I_v, I_t	Spatial and temporal image gradients

Optical Flow and Velocity

$\Delta\mathbf{p}_i$	Pixel displacement vector of feature i
v_{opt}	Planar velocity estimated from optical flow
v_{corr}	Velocity corrected for rotational motion
h	Altitude measured by the range sensor
$\omega = [\omega_x, \omega_y, \omega_z]^\top$	Angular velocity measured by the IMU

CNN-Based Feature Selection

$\mathcal{P}_i$	32×32 grayscale image patch around feature i
$s_i \in [0,1]$	CNN-predicted probabilistic reliability score for feature i
y_i	Binary label indicating feature trackability
θ	Learned network parameters
$\sigma(\cdot)$	Sigmoid activation function
τ_{cnn}	Adaptive feature selection threshold

Extended Kalman Filter (EKF)

$\mathbf{x}$	EKF state vector
$\mathbf{p}$	Position vector in global frame
$\mathbf{v}$	Linear velocity vector
$\mathbf{q}$	Unit quaternion representing attitude

$\mathbf{b}_\omega$	Gyroscope bias
$\mathbf{b}_a$	Accelerometer bias
P	State covariance matrix
Q	Process noise covariance matrix
R	Measurement noise covariance matrix
K	Kalman gain

1 Introduction

1.1 Overview and Motivation

Autonomous quadrotors increasingly operate in environments where Global Navigation Satellite System (GNSS) signals are unavailable or unreliable, including indoor spaces, urban canyons, and cluttered environments. In such settings, accurate and reliable onboard state estimation is essential for safe and autonomous flight. Visual–inertial odometry (VIO) has become a cornerstone of navigation in GPS-denied environments, enabling quadrotors to estimate position, velocity, and attitude through the fusion of camera and inertial sensor data [7] [8] [9]. These capabilities have enabled a wide range of real-world applications, including search and rescue, infrastructure inspection, and autonomous flight in cluttered environments [10] [11]. Figure 1 illustrates representative GPS denied environments where quadrotors must rely primarily on onboard sensing.

Despite significant advances in multisensor fusion and state estimation frameworks, the performance of VIO systems remains highly dependent on the quality and stability of visual features extracted from onboard cameras. Classical feature detectors and optical flow techniques, such as Shi–Tomasi, FAST, ORB, and Lucas–Kanade tracking, are widely adopted due to their computational efficiency and suitability for embedded platforms [12] [13] [14] [15]. However, these methods frequently extract redundant or short-lived features, particularly in low-texture regions, dynamic scenes, or under changing illumination conditions. The presence of unstable or low-quality features can inflate computational cost, degrade data association, and amplify drift over time, ultimately limiting long-term navigation accuracy. Figure 2 provides an example of dense classical feature extraction and highlights the presence of redundant and short-lived keypoints.

In parallel, learning-based feature detectors and descriptors have demonstrated superior robustness and repeatability under challenging visual conditions, including motion blur, viewpoint variation, and illumination changes [3] [16] [17]. While these approaches significantly improve feature quality, their high computational demands often make them impractical for real-time onboard deployment on resource-constrained aerial platforms. This creates a fundamental trade-off between robustness and efficiency, where highly reliable feature extraction is achieved at the cost of increased latency, power consumption, and reduced onboard autonomy. Figure 3 illustrates examples of learning-based feature extraction, highlighting their robustness but also their typical computational complexity.

This trade-off motivates the need for a new class of lightweight and adaptive approaches that explicitly target feature reliability while preserving real-time performance. Rather than replacing classical pipelines with heavy deep learning models, this thesis is motivated by the goal of selectively improving feature quality within an efficient, classical vision framework. By integrating lightweight learning-based feature selection with classical optical flow and multisensor fusion, this work aims to enable accurate, efficient, and fully onboard GPS-free quadrotor navigation. The emphasis on feature trackability and long-term stability, rather than feature density, is central to achieving robust state estimation on real-world, resource-limited quadrotor platforms.

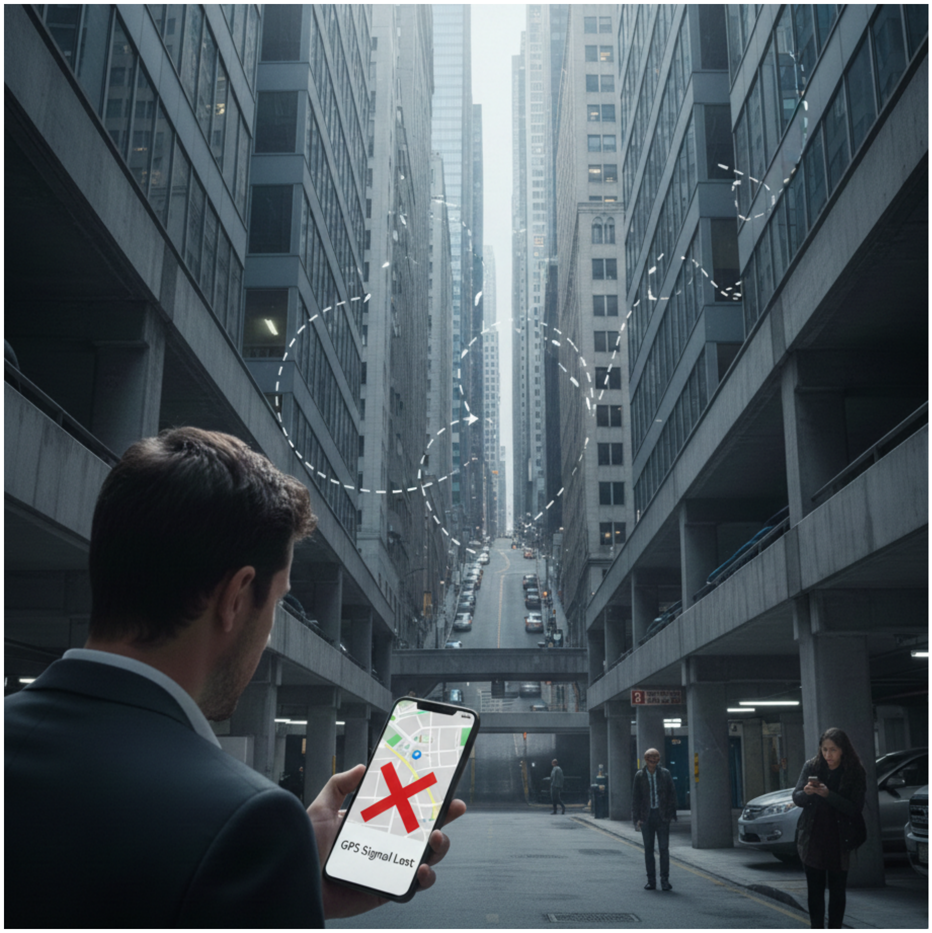

Figure 1: Representative GPS-denied environments for quadrotor navigation, including indoor spaces and urban canyons. [1]

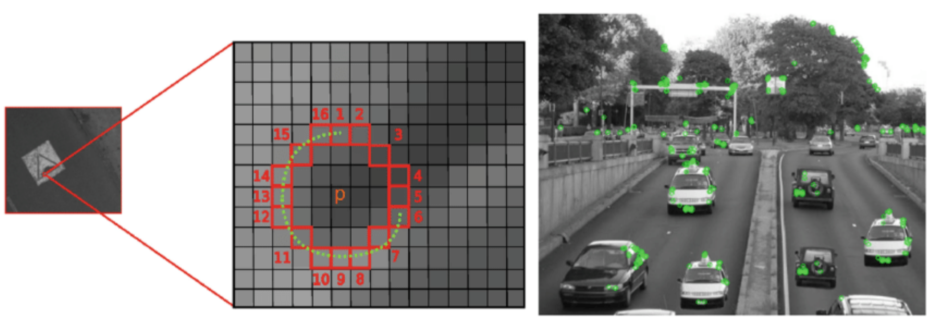

Figure 2: (a) Fast Keypoint detector. (b) ORB feature detection. [2]

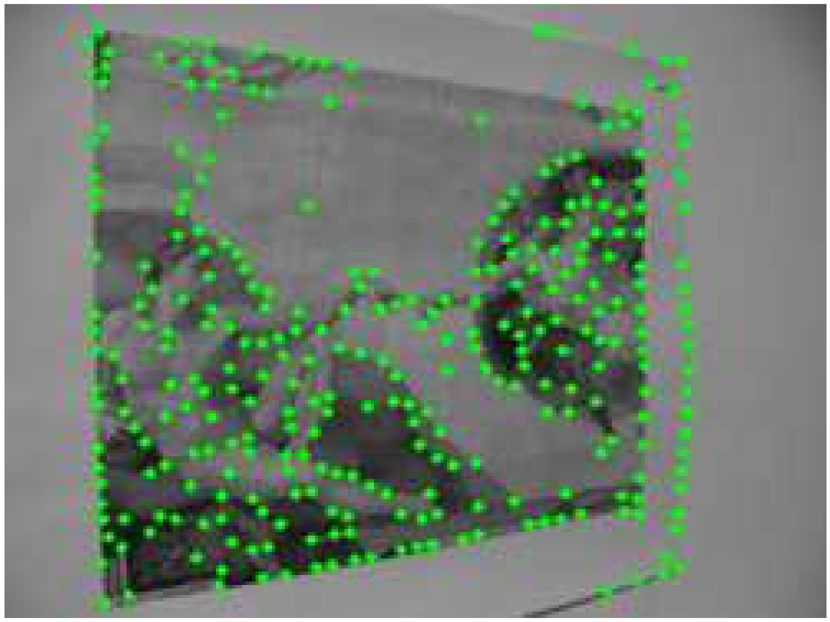

Figure 3: Example of learning-based feature detection SuperPoint, showing improved robustness under challenging visual conditions but with much higher computational time. [3]

1.2 Problem Definition

Despite substantial progress in visual–inertial odometry and multisensor fusion, reliable quadrotor navigation in GPS-denied environments remains fundamentally constrained by the quality of visual features used for state estimation. Existing approaches largely fall into two categories. Classical feature-based pipelines are computationally efficient and suitable for embedded platforms, but they frequently extract redundant, short-lived, or visually unstable features, which degrade data association and amplify drift during long-term operation. In contrast, learning-based feature detectors and descriptors provide improved robustness and repeatability under challenging visual conditions, but their high computational cost and memory requirements make them impractical for real-time deployment on resource-constrained quadrotor platforms.

As a result, there currently exists no widely adopted solution that simultaneously achieves high feature reliability, low computational cost, and real-time onboard performance for small aerial vehicles. This gap is particularly critical for applications that require high-frequency, low-latency state estimation, such as agile flight, autonomous navigation in cluttered environments, and long-duration operation without external infrastructure.

The core problem addressed in this thesis therefore is to design a navigation framework that improves the reliability and long-term stability of visual features while preserving the strict computational and real-time constraints imposed by embedded quadrotor platforms. Specifically, the challenge is to enhance feature quality in a manner that directly benefits downstream state estimation, without resorting to heavy deep learning architectures or sacrificing the efficiency of classical vision pipelines.

1.3 Objectives

The primary objective of this thesis is to develop a lightweight, fully onboard navigation framework for quadrotors operating in GPS-denied environments that improves state estimation accuracy through enhanced visual feature reliability. Rather than increasing feature density or relying on computationally expensive deep learning models, this work aims to selectively improve feature quality in a manner that is compatible with real-time embedded platforms.

Specifically, this thesis seeks to design and integrate a lightweight learning-based feature selection mechanism within a classical optical flow and multisensor fusion pipeline. The objective is to reduce the impact of redundant and short-lived features, improve long-term feature stability, and mitigate drift during extended operation, while preserving high update rates and low computational latency. Through this approach, the thesis aims to demonstrate that targeted, efficient learning can significantly enhance navigation performance without violating the strict resource constraints of small aerial vehicles.

1.4 Research Questions

This thesis is guided by a set of focused research questions that address the central challenge of achieving reliable, real-time quadrotor navigation in GPS-denied environments under strict computational constraints. These research questions are formulated to investigate both the algorithmic and system-level impact of lightweight learning-based feature selection within a classical visual–inertial navigation framework.

RQ1: Can a lightweight learning-based feature selection strategy significantly improve visual feature stability and state estimation accuracy while satisfying real-time embedded computational constraints on a quadrotor platform?

RQ2: How can adaptive regulation of visual features be integrated with classical optical flow to reduce feature redundancy and improve long-term trackability across varying environmental conditions?

RQ3: To what extent does improved visual feature reliability translate into reduced drift and improved velocity and position estimation in a real-world, GPS-denied quadrotor navigation system?

These research questions provide a structured framework for the experimental evaluation and analysis presented in this thesis, enabling quantitative assessment of the proposed methods in terms of accuracy, robustness, and real-time performance.

1.5 Definition of Core Terminology

To ensure clarity and consistency throughout this thesis, key terms used in the formulation, methodology, and analysis are formally defined in this section.

Visual Feature: A visual feature refers to a salient image point detected in the image plane, typically corresponding to corners or textured regions, that can be reliably localized and tracked across consecutive frames. In this work, features are initially detected using the ORB detector.

Feature Trackability: Feature trackability is defined as the ability of a feature to be successfully tracked across multiple consecutive frames using optical flow without significant drift or loss. It reflects the temporal persistence of a feature and its suitability for motion estimation.

Feature Reliability: Feature reliability refers to the likelihood that a feature contributes accurate and stable information for state estimation. In this thesis, reliability is modeled as a probabilistic score predicted by a lightweight convolutional neural network, indicating the expected survival and usefulness of a feature during tracking.

Feature Redundancy: Feature redundancy occurs when multiple features provide overlapping or highly correlated information, typically due to spatial clustering or similar geometric constraints. Redundant features do not significantly increase the information content available to the estimator and may unnecessarily increase computational cost.

Feature Distribution: Feature distribution describes the spatial arrangement of features across the image plane. A well-distributed feature set improves observability and estimation stability by providing diverse geometric constraints, whereas clustered features reduce effective information gain.

Lightweight Learning-Based Module: A lightweight learning-based module refers to a neural network architecture designed to operate under strict computational constraints, characterized by low parameter count, minimal memory usage, and real-time inference capability. In this work, the module performs feature reliability prediction rather than full feature detection.

Adaptive Thresholding: Adaptive thresholding is a mechanism that dynamically adjusts the selection threshold applied to feature reliability scores in order to regulate the number of retained features. This ensures robustness across varying environmental conditions, preventing both feature scarcity and excessive redundancy.

Optical Flow-Based Velocity Estimation: Optical flow-based velocity estimation refers to the computation of planar velocity from pixel displacements of tracked features between consecutive frames, using camera intrinsic parameters and altitude measurements to recover metric motion.

Rotational Motion Compensation: Rotational motion compensation is the process of removing apparent image motion caused by angular velocity of the camera, using inertial measurements. This ensures that the estimated optical flow corresponds to translational motion only.

State Estimation: State estimation refers to the process of estimating the system state, including position, velocity, and orientation, by fusing measurements from multiple sensors within a probabilistic framework. In this thesis, an Extended Kalman Filter (EKF) is used for this purpose.

Real-Time Constraint: The real-time constraint refers to the requirement that all computations within the navigation pipeline must be completed within a bounded time interval to support control and navigation. This imposes a strict trade-off between computational complexity and estimation accuracy.

Information Contribution: Information contribution refers to the extent to which a feature improves the estimation quality by reducing uncertainty in the state estimate. Features with high information contribution are typically well-distributed, long-lived, and geometrically informative.

Computation Time: Computation time refers to the total processing time required to process a single image frame within the navigation pipeline, measured in milliseconds. It reflects the computational efficiency of the method and is a key indicator of suitability for real-time deployment on embedded platforms.

Processing Frequency: Processing frequency refers to the number of frames processed per second, measured in Hertz (Hz). It represents the real-time capability.

Root Mean Square Error (RMSE): Root Mean Square Error is a standard metric used to evaluate estimation accuracy. It is defined as the square root of the average squared difference between estimated values and corresponding ground truth measurements.

Position Estimation Error: Position estimation error refers to the deviation between the estimated position and the ground truth position over a trajectory. In this thesis, it is evaluated using RMSE to quantify overall accuracy.

Velocity Estimation Error: Velocity estimation error represents the difference between the estimated velocity and the ground truth velocity. It is also evaluated using RMSE to assess the accuracy of motion estimation.

Number of Selected Features: The number of selected features refers to the total number of visual features retained after the feature selection process and used for optical flow tracking. This quantity reflects both the computational load and the effectiveness of the feature selection mechanism.

1.6 Contributions

The main contributions of this thesis are summarized as follows:

- **Lightweight learning-based feature selection:** The primary benefit of this module is the mitigation of the classical trade-off between robustness and computational efficiency. By utilizing a compact neural architecture with approximately 136,001 parameters, the proposed approach achieves improved feature reliability comparable to learning-based methods while maintaining the low-latency requirements necessary for real-time quadrotor navigation.

- **Adaptive feature regulation:** The adaptive thresholding mechanism ensures stable operation across varying environmental conditions by dynamically regulating the number of selected features. This prevents feature starvation in low-texture scenes and reduces computational redundancy in highly textured environments, resulting in consistent estimator performance.

- **Hybrid perception pipeline:** The integration of lightweight learning with classical ORB-based detection and Lucas–Kanade tracking preserves computational efficiency while improving measurement quality. This enables high-frequency processing without the overhead associated with dense deep learning pipelines.

- **Improved state estimation performance:** By providing higher-quality visual measurements to the Extended Kalman Filter, the proposed method reduces estimation error and improves the statistical consistency of the state estimation process, particularly under challenging lighting and motion conditions.

1.7 Related Work

1.7.1 Visual–Inertial Odometry and Multisensor Fusion

Visual–Inertial Odometry (VIO) has become a cornerstone of autonomous quadrotor navigation in GNSS-denied environments, enabling reliable state estimation through the fusion of camera and inertial measurements. Early filtering-based approaches established the feasibility of tightly coupled camera–IMU fusion within a computationally efficient framework. The Multi-State Constraint Kalman Filter (MSCKF) introduced by Mourikis and Roumeliotis [18] demonstrated that visual constraints could be incorporated without explicitly augmenting the state with landmark parameters, enabling real-time performance suitable for embedded platforms.

Bloesch et al. [7] further proposed a direct EKF-based formulation for visual–inertial state estimation, emphasizing estimator consistency and robustness under challenging visual conditions.

Optimization-based methods later improved estimation accuracy by jointly minimizing reprojection and inertial residuals over sliding windows. Leutenegger et al. [8] proposed OKVIS, a keyframe-based nonlinear optimization framework integrating visual and inertial measurements. Forster et al. [19] introduced IMU preintegration on manifolds, significantly improving computational efficiency and numerical stability. Qin et al. [20] further developed VINS-Mono, incorporating loop closure and relocalization capabilities for improved long-term performance. More recently, ORB-SLAM3 unified visual, visual–inertial, and multi-map SLAM within a consistent framework [21].

Open-source platforms such as OpenVINS [9] and benchmark datasets such as EuRoC MAV [22] have enabled reproducible evaluation of VIO systems.

Despite these advances, both filtering- and optimization-based approaches remain fundamentally dependent on the quality and stability of visual features provided by the front-end. Poor feature selection degrades data association, reduces constraint quality, and amplifies drift over time. This highlights a critical limitation in existing VIO pipelines: estimator performance is tightly coupled to front-end feature reliability.

1.7.2 Classical Feature Detection and Optical Flow

Classical feature detection and tracking methods remain widely used due to their computational efficiency and compatibility with embedded systems. Shi–Tomasi corner detection [12] selects features based on the eigenvalues of the image gradient covariance matrix, while FAST [13] enables high-speed detection through learned intensity comparisons. ORB [14] combines FAST detection with oriented binary descriptors, providing rotation invariance and real-time performance.

For feature tracking, the Lucas–Kanade (LK) optical flow method [15] remains a standard choice in visual odometry pipelines. The theoretical foundations of optical flow were formalized by Horn and Schunck [23], while the pyramidal implementation [24] enables robustness under larger inter-frame motion.

From a geometric perspective, Hartley and Zisserman [25] emphasized that long-lived, well-distributed features are essential for stable multi-view estimation. However, classical detectors often extract redundant, clustered, or short-lived features, particularly in low-texture or dynamic environments. This creates a mismatch between feature quantity and feature utility, where increasing the number of detected features does not necessarily improve estimation accuracy but does increase computational cost.

Importantly, classical methods do not explicitly account for feature trackability over time. They rely on instantaneous saliency rather than predicting whether a feature will remain stable across frames, which is a key limitation for optical-flow-based navigation.

Beyond feature detectability, the spatial distribution of features plays a critical role in estimation quality. From an information-theoretic perspective, features that are spatially clustered provide redundant measurements, as they are subject to similar noise sources and geometric constraints. This redundancy limits the effective information gain provided to the estimator. In contrast, spatially well-distributed features improve observability across multiple motion directions and contribute to better numerical conditioning of the estimation problem. Therefore, feature selection should not be driven solely by local saliency, but also by spatial diversity and long-term trackability, which are more informative for state estimation.

1.7.3 Learning-Based Feature Detection and Selection

Learning-based feature detectors such as SuperPoint [3], LF-Net [17], and R2D2 [16] have demonstrated improved robustness under illumination variation, viewpoint changes, and motion blur by leveraging convolutional neural networks to learn interest points and descriptors.

More recent work has explored adaptive and reliability-aware feature selection strategies. Zhang et al. [26] proposed adaptive feature selection for visual odometry, highlighting that regulating feature quality is more effective than increasing feature density.

Learning-based methods have also been applied to broader quadrotor navigation problems. DroNet [10] demonstrated end-to-end learning for agile flight, while Penicka et al. [11] investigated learning-based trajectory optimization.

Despite their robustness, deep feature detectors are often computationally expensive and unsuitable for real-time deployment on resource-constrained aerial platforms. This has motivated hybrid approaches that integrate lightweight learning modules into classical pipelines. Rather than replacing handcrafted detectors entirely, these methods aim to selectively improve feature quality while preserving computational efficiency.

An additional challenge in learning-based approaches is generalization across environments. Classical feature detectors such as ORB and FAST rely on fixed geometric and photometric principles, allowing them to operate consistently across diverse scenes without retraining. In contrast, learned feature representations may suffer from domain shift, where performance degrades when deployed in environments that differ from the training data distribution. This introduces a trade-off between robustness and generalization.

The hybrid approach adopted in this thesis mitigates this limitation by preserving the geometric consistency of classical feature detection while restricting learning to a lightweight reliability prediction task. By framing learning as a probabilistic scoring mechanism rather than a full feature replacement, the method reduces the risk of overfitting while maintaining adaptability to varying conditions.

1.7.4 Frequency-Domain Motion Estimation

An alternative class of approaches estimates motion in the frequency domain using phase correlation techniques. The foundational work by Kuglin and Hines [27] estimates image translation through Fourier phase differences, while Foroosh et al. [28] extended this method to achieve subpixel accuracy.

Recent work has applied Fourier-based Phase Correlation (FPC) to UAV velocity estimation in GNSS-denied environments [29]. These approaches operate on global spectral alignment rather than local feature correspondences, offering robustness to illumination changes and repetitive textures.

However, frequency-domain methods rely on global assumptions of translational motion and lack explicit modeling of local feature persistence. They are also sensitive to aliasing and windowing effects, and provide limited geometric constraints for long-term drift mitigation compared to feature-based approaches.

1.7.5 State Estimation and Extended Kalman Filtering

State estimation in multisensor navigation systems commonly relies on the Extended Kalman Filter (EKF), which provides a recursive framework for nonlinear stochastic estimation [30, 31]. In visual–inertial systems, EKF formulations integrate inertial propagation with visual updates, requiring accurate modeling of both process and measurement uncertainties.

In optical-flow-based navigation, image motion contains both translational and rotational components. Rotational motion induces apparent velocities that must be compensated to obtain physically meaningful measurements. This motion-field decomposition is well studied in visual–inertial frameworks such as [7].

The effectiveness of EKF-based fusion is therefore directly influenced by the quality of visual measurements. Unstable or short-lived features degrade the statistical consistency of the estimator and reduce the informativeness of measurement updates.

The EKF framework assumes that measurement noise follows approximately Gaussian distributions. However, unreliable or short-lived visual features introduce outliers that violate this assumption, leading to biased updates and degraded estimation performance. Consequently, the quality of feature selection directly impacts the statistical consistency of the filter.

From this perspective, feature selection can be interpreted as a form of pre-filtering or implicit outlier rejection, where unreliable measurements are removed before entering the estimation pipeline. The approach proposed in this thesis enhances this process by predicting feature reliability prior to tracking, thereby improving measurement quality and preserving the validity of the EKF assumptions.

1.7.6 Problem Formalization

The feature selection problem in visual–inertial navigation can be formulated as the task of identifying an optimal subset of visual features that maximizes estimation performance under computational constraints. Let F_{total} denote the set of all detected features in an image frame. The objective is to select a subset $F_{\text{best}} \subset F_{\text{total}}$ such that the selected features maximize the information contribution to the state estimator while minimizing redundancy and computational cost.

From an estimation perspective, this can be interpreted as maximizing the information gain provided to the estimator while minimizing the innovation error y_k in the EKF update step. Features that are spatially clustered or temporally unstable contribute less useful information and may degrade estimation quality.

The key challenge lies in predicting the future trackability and reliability of features using limited computational resources. High-dimensional learned descriptors can provide such predictions but are often impractical for real-time deployment. This motivates the need for lightweight approaches that approximate feature reliability while preserving real-time performance.

1.7.7 Discussion and Research Gap

The literature reveals a fundamental trade-off between computational efficiency, robustness, and estimation accuracy. Classical methods are efficient but often produce redundant or unreliable features, while learning-based methods improve robustness at the cost of significantly higher computational requirements and potential generalization issues. At the same time, state estimation frameworks remain highly sensitive to the quality, distribution, and persistence of front-end measurements.

In real-time quadrotor navigation, this trade-off is further constrained by latency requirements. Accurate estimation that cannot be computed within strict timing constraints is impractical for control and navigation. This establishes a latency–accuracy Pareto boundary, where both estimation quality and computational efficiency must be jointly optimized.

Despite extensive research in VIO and feature learning, there remains a lack of approaches that explicitly optimize feature reliability under strict real-time and embedded constraints. In particular, existing methods do not adequately address the problem of selecting features based on their expected temporal trackability and contribution to downstream estimation.

This thesis addresses this gap by introducing a lightweight learning-based feature selection module that operates as a pruning mechanism within a classical ORB–LK pipeline. By predicting feature reliability and adaptively regulating feature selection, the proposed approach improves measurement quality without incurring the computational overhead of full learning-based detectors. This enables a balanced trade-off between robustness and efficiency, tailored for real-time onboard quadrotor navigation.

1.8 Thesis Organization

The remainder of this thesis is organized to present the proposed navigation framework in a logical progression from system design to experimental validation and analysis. Chapter 2 presents the overall system architecture and details the proposed methodology, including the quadrotor platform, sensor suite, classical feature detection, lightweight CNN-based feature selection, adaptive feature regulation, optical flow velocity estimation, and multisensor fusion framework. This chapter establishes the technical foundations of the proposed approach and describes how the individual components are integrated into a real-time onboard system.

Chapter 3 describes the experimental setup and evaluation methodology. This includes the hardware platform, onboard computational architecture, sensor calibration, data collection procedures, ground truth acquisition, and the quantitative performance metrics used for evaluation. This chapter ensures the reproducibility and fairness of the experimental results and provides the necessary context for interpreting system performance.

Chapter 4 presents the experimental results and detailed analysis. This chapter evaluates the proposed method in terms of feature stability, computational efficiency, velocity and position estimation accuracy, and drift behavior. Comparisons against baseline classical methods and frequency-domain velocity estimation approaches are provided, along with ablation studies to assess the impact of individual system components. A comprehensive discussion of results, limitations, and practical insights is also included.

Finally, Chapter 5 concludes the thesis by summarizing the main contributions and key findings. This chapter discusses the broader implications of the proposed framework and outlines directions for future work, including extensions to full visual–inertial odometry, integration with additional sensing modalities, and deployment on more computationally constrained or higher-speed aerial platforms.

2 System Architecture

2.1 System Overview

This chapter presents the architecture of the proposed GPS-free quadrotor navigation system. The objective of the system is to estimate planar velocity and full state information in real time using only onboard sensors, while maintaining robustness and computational efficiency suitable for embedded aerial platforms.

The proposed architecture follows a modular, feed-forward pipeline in which the output of each processing block serves as the input to the subsequent stage. Starting from raw sensor measurements, the system performs classical feature detection, learning-based feature selection, optical flow velocity estimation, inertial compensation, and probabilistic sensor fusion to produce a final state estimate.

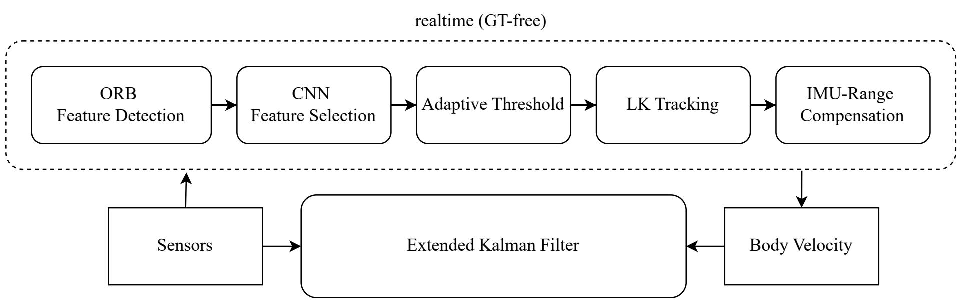

Figure 4: End-to-end system architecture illustrating the data flow from onboard sensors through feature detection, CNN-based feature selection, adaptive threshold, Lucas-Kanade optical flow velocity estimation, inertial compensation, and EKF-based state estimation. [4]

As illustrated in Figure 4 , the system consists of six main stages: classical ORB feature detection, CNN-based feature pruning, adaptive threshold, Lucas–Kanade optical flow tracking, IMU–Range false velocity compensation, and Extended Kalman Filter (EKF) state estimation. Each stage is designed to be computationally efficient and to directly contribute to improving the reliability of downstream state estimation. The modular structure of the pipeline allows each component to be analyzed independently while maintaining a coherent end-to-end architecture.

2.2 Methodological Advantages of the Hybrid Pipeline

The proposed hybrid architecture offers several key advantages over traditional visual-inertial front-end designs by explicitly addressing the limitations of both classical and learning-based approaches.

1) Reduction of Measurement Noise and Outliers: The CNN-based feature selection module predicts the reliability of candidate features and removes those that are likely to exhibit poor trackability. This reduces the number of outliers in the visual measurements and improves data association quality. As a result, the Extended Kalman Filter receives cleaner observations, leading to improved estimation accuracy and enhanced statistical consistency.

2) Deterministic Real-Time Performance: Unlike dense learning-based feature extractors that introduce significant computational overhead, the proposed method operates as a selective filtering stage applied to a classical pipeline. This allows the system to maintain high processing frequencies (exceeding real-time requirements) while avoiding the performance degradation typically associated with deep models.

3) Memory-Efficient Architecture: The CNN module is designed with approximately 136k parameters, enabling the model and intermediate feature representations to reside within fast-access memory. This reduces latency associated with memory transfer and ensures predictable execution time, which is critical for real-time control applications.

4) Balanced Feature Representation: The adaptive thresholding mechanism maintains a consistent number of selected features across varying environmental conditions. This prevents both feature scarcity and redundancy, ensuring that the estimator receives a stable and informative set of measurements.

Overall, the proposed approach improves system performance by optimizing feature quality rather than increasing feature quantity or computational complexity, resulting in a more efficient and robust navigation pipeline.

2.3 Sensor Suite and Data Sources

The proposed navigation framework relies exclusively on onboard sensing. A downward-facing monocular camera provides high-rate grayscale image measurements used for optical flow–based velocity estimation. This configuration is well suited for GPS-denied environments, where ground-plane visual texture is commonly available and stable.

Inertial Measurement Units (IMUs) provide angular velocity and linear acceleration measurements required for motion propagation and compensation of rotationally induced image motion. A downward-facing range sensor supplies altitude information, which is used to scale image-plane motion into metric velocity estimates under the planar motion assumption.

Together, these sensors provide complementary information that enables reliable velocity estimation and state propagation without reliance on external localization infrastructure. The output of this sensing stage forms the input to the feature detection and tracking components described in the following sections.

2.4 Coordinate Frames and Transformations

Accurate state estimation and meaningful comparison with ground truth measurements require a clear and consistent definition of coordinate frames and transformations. In this thesis, three coordinate frames are considered: the camera frame, the body frame, and the inertial (global) frame. All frames are defined as right-handed coordinate systems and follow the right-hand rule. Velocity estimates produced by the vision pipeline are successively transformed through these frames to ensure consistency with the global reference used by the external motion capture system.

2.4.1 Camera and Body Frames

The camera frame, denoted by $\mathcal{F}_c$, is attached to the downward-facing monocular camera used for optical flow–based velocity estimation. Its origin is located at the optical center of the camera, with the x_c and y_c axes aligned with the image plane and the z_c axis aligned with the optical axis. Optical flow measurements yield planar velocity estimates expressed in this frame.

The body frame, denoted by $\mathcal{F}_b$, is rigidly attached to the quadrotor, with its origin located at the vehicle's center of mass. The body-frame axes follow the standard aerospace convention and are aligned with the vehicle structure. Quantities measured by onboard inertial sensors, such as angular velocity and specific force, are naturally expressed in this frame.

Due to the downward-facing mounting configuration of the camera, the camera-frame axes are inverted relative to the body frame in the horizontal plane. Consequently, a positive translational motion of the quadrotor along the body-frame x_b or y_b direction produces an apparent optical flow in the opposite direction in the camera frame. This sign inversion must be explicitly accounted for when transforming velocity estimates obtained from optical flow.

Figure 5 illustrates the relative orientation of the camera and body frames and highlights the axis inversion introduced by the downward-facing camera.

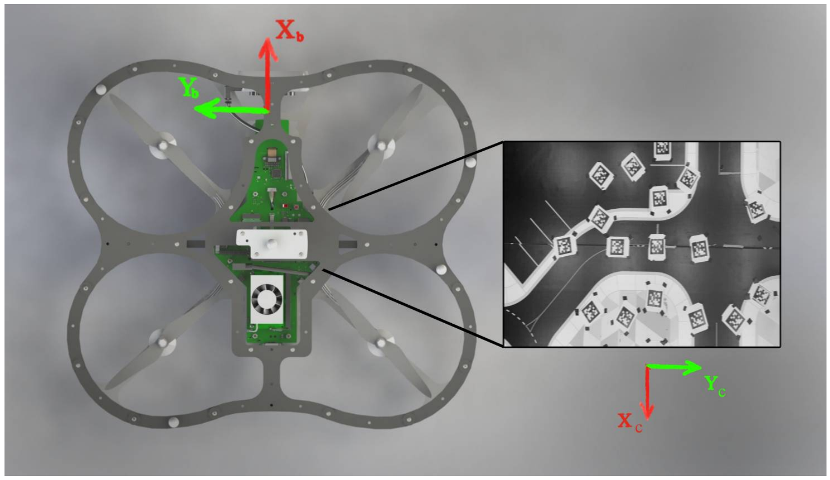

Figure 5: Relative orientation of the camera frame $\mathcal{F}_c$ and the body frame $\mathcal{F}_b$ for the downward-facing camera configuration.

The fixed transformation from camera-frame velocity $\mathbf{v}_c$ to body-frame velocity $\mathbf{v}_b$ is described by a constant rotation matrix $\mathbf{R}_{bc} \in SO(3)$. For the considered mounting configuration, this rotation introduces a sign inversion in the horizontal plane and is given by

$$\mathbf{R}_{bc} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{1}$$

Accordingly, the body-frame velocity is obtained as

$$\mathbf{v}_b = \mathbf{R}_{bc} \mathbf{v}_c. \tag{2}$$

2.4.2 Body and Inertial Frames

The inertial frame, denoted by $\mathcal{F}_g$, represents a fixed global reference frame in which Newton's laws of motion are valid. This frame is used as the primary reference for trajectory representation and performance evaluation throughout the thesis.

The relationship between the body frame and the inertial frame is governed by the quadrotor attitude. The attitude is estimated online by the state estimator and represented internally by a unit quaternion $\mathbf{q}$. For clarity, the corresponding rotation matrix $\mathbf{R}_{gb} \in SO(3)$ can be expressed in terms of roll ϕ, pitch θ, and yaw ψ angles as

$$\mathbf{R}_{gb} = \begin{bmatrix} c_\theta c_\psi & c_\psi s_\phi s_\theta - c_\phi s_\psi & s_\phi s_\psi + c_\phi c_\psi s_\theta \\ c_\theta s_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi & c_\phi s_\theta s_\psi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\phi c_\theta \end{bmatrix}, \tag{3}$$

where $c(\cdot) = \cos(\cdot)$ and $s(\cdot) = \sin(\cdot)$.

Figure 6 illustrates the relative orientation between the body frame and the inertial frame.

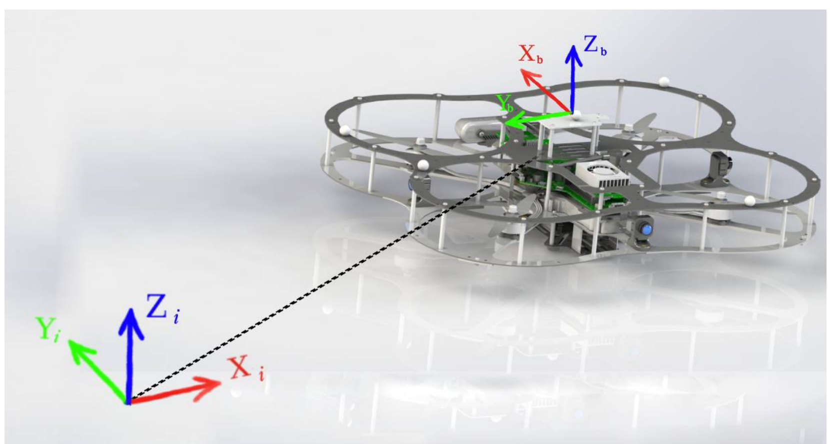

Figure 6: Definition of the body frame $\mathcal{F}_b$ and the inertial (global) frame $\mathcal{F}_g$.

Using this rotation, the transformation from body-frame velocity to inertial-frame velocity is given by

$$\mathbf{v}_g = \mathbf{R}_{gb}(\phi, \theta, \psi)\, \mathbf{v}_b, \tag{4}$$

where $\mathbf{v}_g$ denotes the velocity expressed in the inertial frame.

2.4.3 External Motion Capture Frame

Ground truth measurements are provided by an external OptiTrack motion capture system, which defines its own global coordinate frame. While this frame is inertial, its axis orientation may differ from the inertial frame $\mathcal{F}_g$ adopted in the navigation system.

To enable direct comparison between estimated states and ground truth measurements, the OptiTrack frame is mapped to the inertial frame using a fixed alignment transformation obtained through system calibration. This transformation remains constant throughout all experiments. Figure 7 illustrates the relationship between the OptiTrack frame and the inertial frame used in this thesis.

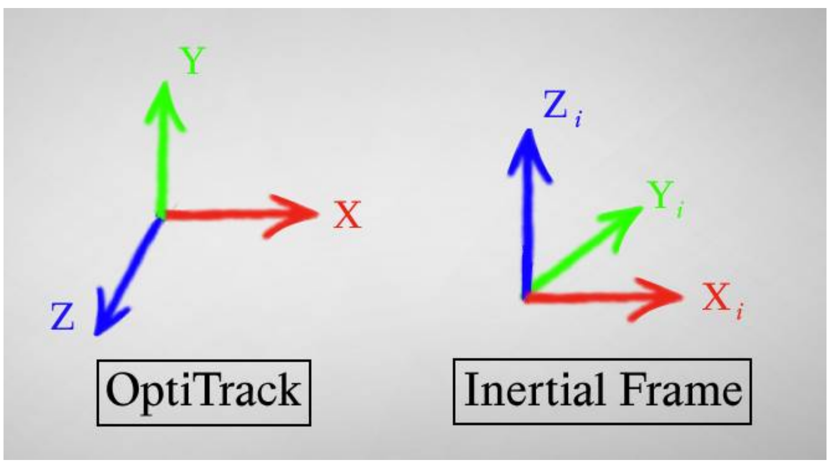

Figure 7: Alignment between the OptiTrack global frame and the inertial frame $\mathcal{F}_g$.

After this alignment, all estimated positions and velocities are expressed in the same inertial coordinate frame as the ground truth data, ensuring consistent evaluation of trajectory accuracy and state estimation performance.

2.4.4 Overall Transformation Chain

Combining the above transformations yields the complete mapping from camera-frame velocity estimates to inertial-frame velocities:

$$\mathbf{v}_g = \mathbf{R}_{gb}(\phi, \theta, \psi)\, \mathbf{R}_{bc}\, \mathbf{v}_c. \tag{5}$$

This transformation chain guarantees that velocities derived from optical flow are correctly oriented and directly comparable to ground truth measurements provided by the external motion capture system.

2.5 Classical Feature Detection (ORB)

The first processing stage in the pipeline performs classical feature detection using the Oriented FAST and Rotated BRIEF (ORB) algorithm [14]. ORB combines FAST corner detection with orientation estimation, offering a favorable balance between computational efficiency and robustness for real-time embedded systems.

ORB identifies salient image locations by detecting pixel intensity patterns that exhibit corner-like characteristics. For each detected keypoint, an orientation is computed using local image moments, enabling rotation-invariant representation. In the proposed system, ORB is employed solely as a fast and reliable candidate feature generator rather than as a full descriptor-based matching framework.

Figure 8: ORB feature candidates detected in a representative frame. Many features are spatially clustered or redundant, motivating learned feature selection.

The output of the ORB stage consists of a set of candidate keypoint locations that serve as inputs to the subsequent feature selection module. While ORB efficiently produces a large number of features, many of these detections are redundant, short-lived, or unsuitable for long-term tracking under optical flow. Consequently, ORB detections are treated as candidates whose reliability must be evaluated before they are used for motion estimation. This design choice preserves the efficiency of classical feature detection while enabling learned pruning in the following stage of the pipeline.

2.6 CNN-Based Feature Selection

The CNN-based feature selection strategy introduced in this section builds upon the lightweight pruning framework presented in [4], and is expanded here with additional details regarding training data generation, network architecture, and integration within the navigation pipeline.

The proposed CNN-based feature selection module operates on candidate features generated by the classical ORB detector and evaluates their suitability for reliable optical flow tracking. Rather than performing feature detection or description, the network functions as a lightweight scoring mechanism that assigns a reliability probability to each candidate feature based on the appearance of the local image.

The following subsections describe the feature selection process in detail, including patch extraction and labeling, network architecture design, training objective formulation, and the integration of the network output into the downstream navigation pipeline. Each component is introduced sequentially to reflect the data flow of the system, from raw image patches to the final set of selected features used for motion estimation.

2.6.1 Patch Extraction and Label Generation

Given a set of candidate feature locations detected by the ORB detector, local image patches are extracted around each keypoint to serve as inputs to the feature selection network. For each detected keypoint at pixel location $\mathbf{p}_i = [u_i, v_i]^\top$ where u_i and v_i represent the horizontal and vertical pixel coordinates of the feature in the image frame, respectively. A fixed size grayscale image patch $\mathcal{P}_i \in \mathbb{R}^{32\times32}$ is extracted and normalized to zero mean and unit variance. The patch size is selected to provide sufficient local spatial context for evaluating feature reliability while maintaining low computational cost suitable for real-time embedded inference.

Training labels are generated automatically using a tracking-driven supervision strategy. Each candidate feature is tracked across consecutive frames using the Lucas–Kanade optical flow algorithm. A feature is considered *surviving* if it remains successfully trackable for at least $L = 5$ consecutive frames without exceeding a predefined reprojection error threshold. Features that satisfy this criterion are labeled as reliable ($y_i = 1$), while not tracked features are labeled as unreliable ($y_i = 0$).

This patch-survival labeling approach directly aligns supervision with the downstream objective of stable optical flow tracking. By avoiding manual annotation or external ground truth, training data can be collected autonomously during normal system operation, ensuring that the learned model captures feature reliability as it manifests in the target navigation scenario.

To generate supervision, each candidate feature is tracked forward in time using the Lucas–Kanade optical flow algorithm across consecutive frames. A feature is considered to have *survived* if it remains successfully trackable throughout all L frames without exceeding a predefined tracking or reprojection error threshold. Features that satisfy this criterion are labeled as trackable ($y_i = 1$), while features that fail to track at any point within the temporal window are labeled as unreliable ($y_i = 0$).

This survival-based labeling strategy directly reflects the objective of the proposed feature selection module: identifying features that are likely to remain stable under optical flow tracking. By relying solely on tracking outcomes, the labeling process does not require manual annotation or external ground truth and can be performed automatically at scale.

2.6.2 Network Architecture

The feature selection network is designed as a compact convolutional neural network to satisfy real-time embedded constraints. The network takes a single grayscale image patch $\mathcal{P}_i$ as input and outputs a scalar score representing the predicted reliability of the corresponding feature.

As shown in Figure 9 the architecture consists of two convolutional layers followed by fully connected layers.

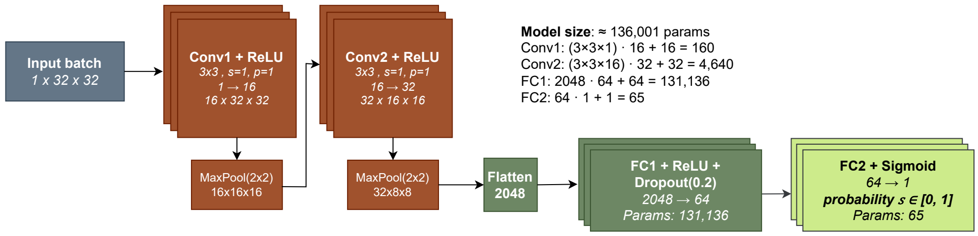

Figure 9: Lightweight CNN architecture used for feature reliability prediction. [4]

- **Conv1:** Convolutional layer with 3×3 kernels, mapping from 1 input channel to 16 output channels, followed by a ReLU activation and a 2×2 max-pooling operation.

- **Conv2:** Convolutional layer with 3×3 kernels, mapping from 16 input channels to 32 output channels, followed by a ReLU activation and a 2×2 max-pooling operation.

- **FC1:** Fully connected layer mapping 2048 input features to 64 output units, followed by a ReLU activation and dropout with probability $p = 0.2$.

- **FC2:** Fully connected layer mapping 64 input features to a single output unit, followed by a sigmoid activation function.

Each convolutional layer applies a set of learned filters to extract local spatial features and is followed by a Rectified Linear Unit (ReLU) activation to introduce nonlinearity. The convolutional layers are responsible for capturing low-level visual cues such as gradients, edges, and texture patterns that are indicative of feature trackability. The total number of trainable parameters in the network is approximately 1.36×10^5; this relatively small parameter count enables high-frequency inference while remaining suitable for real-time deployment on resource-constrained aerial platforms.

The output of the convolutional stage is flattened and passed through fully connected layers, which aggregate the extracted spatial features and produce a single scalar output. Let $f_\theta(\mathcal{P}_i)$ denote the output of the final linear layer of the network for an input patch $\mathcal{P}_i$. A sigmoid activation function is applied at the final layer to map this value to a probability score,

$$s_i = \sigma(f_\theta(\mathcal{P}_i)) \in [0,1], \tag{6}$$

where $\sigma(\cdot)$ denotes the sigmoid function. The resulting score s_i represents the predicted likelihood that the corresponding feature will remain stable and trackable over time, and is subsequently used for feature selection in the downstream navigation pipeline.

This architecture is intentionally minimal and does not attempt to perform feature detection or description. Instead, it operates as a learned scoring function that evaluates the reliability of candidate features generated by the classical detector.

2.6.3 Training Data

Training data for the CNN-based feature selection module is automatically generated using visual sequences from both the EuRoC MAV dataset [22] and real-world quadrotor flight experiments conducted on the target platform. The EuRoC dataset provides high-quality monocular image sequences captured in indoor environments with varying levels of texture, motion dynamics, and illumination conditions, while the real flight data reflect platform-specific visual characteristics and operational conditions encountered during onboard navigation. The combination of these two data sources ensures that the learned model captures feature reliability across both standardized benchmark scenarios and practical deployment environments.

For each image frame in both datasets, candidate feature locations are first detected using the ORB detector. Around each detected keypoint, a fixed-size grayscale image patch $\mathcal{P}_i \in \mathbb{R}^{32\times32}$ is extracted from the first frame of a short temporal window. These patches serve as the input samples to the network, and the same tracking-based labeling and training procedures are applied consistently across both datasets. By leveraging both benchmark and real-world data during training, the feature selection module is encouraged to generalize beyond dataset-specific characteristics, thereby improving robustness and reliability during real-time quadrotor navigation.

Furthermore, because labels are derived from actual tracking performance, the resulting supervision is tightly aligned with the downstream navigation task.

The resulting dataset consists of a balanced set of trackable and unreliable feature patches, enabling effective training of the binary classification network without bias toward either class. This balance encourages the network to learn discriminative visual cues associated with long-term feature stability rather than simple visual saliency.

2.6.4 Objective Function

The feature selection network is trained as a binary classifier that predicts the probability of a feature remaining trackable over time. Given an input patch $\mathcal{P}_i$ and its corresponding survival label $y_i \in \{0,1\}$, the network produces an output score:

$$s_i = \mathcal{F}_\theta(\mathcal{P}_i), \tag{7}$$

where $\mathcal{F}_\theta(\cdot)$ denotes the network parameterized by weights θ.

Training is performed by minimizing the binary cross-entropy loss:

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(s_i) + (1-y_i)\log(1-s_i)\right], \tag{8}$$

where N is the number of training patches.

The network is trained using the Adam optimizer with a learning rate $\eta = 10^{-3}$ and a batch size of 64 for a fixed number of training epochs. Dropout regularization is applied in the fully connected layers to reduce overfitting and encourage generalization across diverse visual conditions. Owing to the low-dimensional nature of the input patches and the compact network architecture, the training process converges rapidly while maintaining stable optimization behavior. These design choices collectively contribute to a reliable feature scoring mechanism that remains effective across varying environments and operating conditions.

This objective encourages the network to assign high confidence to features that exhibit consistent and stable visual characteristics under optical flow tracking. By framing feature selection as a probabilistic prediction problem, the resulting scores provide a principled measure of feature reliability that can be seamlessly integrated into adaptive thresholding and downstream motion estimation. This formulation promotes the retention of features that are more likely to contribute to long-term estimation consistency, thereby improving the robustness of the visual front-end.

2.7 Adaptive Thresholding

To maintain a stable and sufficient number of selected features across varying visual environments, an adaptive thresholding mechanism is applied to the CNN output scores during online inference. Rather than relying on a fixed pruning threshold, the system dynamically adjusts the selection threshold τ_{cnn} based on the number of features retained at each time step. This adaptive strategy enables the visual front-end to respond to changes in scene texture, illumination, and motion dynamics while preserving feature reliability.

At runtime, local image patches are extracted around candidate features detected by the ORB detector and evaluated by the CNN-based feature selection module. Each candidate feature is assigned a reliability score $s_i \in [0,1]$, and only features with scores exceeding the current threshold τ_{cnn} are retained for optical flow tracking. The threshold is updated online to regulate the number of selected features and prevent both feature starvation and excessive redundancy.

Conceptually, the threshold update can be expressed as

$$\tau_{k+1} = \tau_k + \alpha \left(N_k - N_{\text{target}} \right), \tag{9}$$

where N_k denotes the number of retained features at time step k, N_{target} is a desired feature count, and α is a small gain that governs the rate of adjustment. When the number of selected features falls below the target, the threshold is reduced to admit additional candidates; conversely, when the number exceeds the target, the threshold is increased to suppress redundancy.

In practice, the threshold τ_{cnn} is constrained within predefined bounds to ensure stable behavior under extreme visual conditions. Specifically, the threshold is restricted to the interval $[0.5, 0.8]$, preventing overly permissive selection in low-texture scenes and overly aggressive pruning in highly textured environments. The threshold is adjusted incrementally using a small step size, which is chosen to be sufficiently conservative to avoid oscillatory behavior while still allowing the system to adapt in real time.

This adaptive thresholding mechanism requires only the current feature count and operates in constant time, making it well suited for deployment on resource-constrained embedded platforms. By regulating feature selection based on both learned reliability scores and instantaneous scene conditions, the method promotes a consistent supply of high-quality features for downstream optical flow estimation and state estimation. Details of the threshold update procedure are provided in Appendix 1.

2.8 Optical Flow Tracking and Velocity Estimation

Once a refined set of reliable features has been selected through the CNN-based pruning and adaptive thresholding stages, these features are tracked across consecutive image frames to estimate translational motion. Optical flow tracking is performed using the classical Lucas–Kanade (LK) method [15], which estimates local image motion under the assumptions of brightness constancy and small inter-frame displacements.

Let $\mathbf{p}_i^t = [u_i^t, v_i^t]^\top$ denote the image-plane pixel coordinates of the i-th retained feature at time step t. The LK method assumes that the image intensity of a moving point remains constant between two consecutive frames, such that

$$I(u,v,t) = I(u+\Delta u, v+\Delta v, t+\Delta t). \tag{10}$$

Applying a first-order Taylor expansion and neglecting higher-order terms yields the optical flow constraint equation

$$I_u \Delta u + I_v \Delta v + I_t = 0, \tag{11}$$

where I_u and I_v denote the spatial image gradients and I_t denotes the temporal image gradient.

For each feature, the LK method assumes that all pixels within a local window Ω surrounding the feature undergo the same motion. The pixel displacement vector $\Delta \mathbf{p}_i = [\Delta u_i, \Delta v_i]^\top$ is obtained by solving the following least-squares problem:

$$\Delta \mathbf{p}_i = \arg \min_{\Delta u, \Delta v} \sum_{(u,v) \in \Omega} (I_u \Delta u + I_v \Delta v + I_t)^2 . \tag{12}$$

This leads to the closed-form solution

$$\Delta \mathbf{p}_i = \begin{bmatrix} \sum I_u^2 & \sum I_u I_v \\ \sum I_u I_v & \sum I_v^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum I_u I_t \\ -\sum I_v I_t \end{bmatrix}, \tag{13}$$

where the summations are taken over the local window Ω. The updated feature location in the next frame is then given by

$$\mathbf{p}_i^{t+1} = \mathbf{p}_i^t + \Delta\mathbf{p}_i. \tag{14}$$

To obtain a robust estimate of global image motion, the pixel displacements are aggregated across the set of retained features $F_t^\star$. The average pixel displacement is computed as

$$\bar{\Delta u} = \frac{1}{|F_t^\star|} \sum_{i \in F_t^\star} \Delta u_i, \qquad \bar{\Delta v} = \frac{1}{|F_t^\star|} \sum_{i \in F_t^\star} \Delta v_i, \tag{15}$$

where $|F_t^\star|$ denotes the number of tracked features. Dividing by the frame interval Δt yields an average image-plane velocity expressed in pixels per second.

The image-plane velocity is subsequently converted into a metric translational velocity using altitude information provided by the range sensor. Let f_x and f_y denote the focal lengths of the camera in pixel units, and let h represent the altitude of the quadrotor relative to the observed surface. Under the planar motion assumption, the translational velocity components in the body frame are given by

$$v_x = \frac{h}{f_x \Delta t} \bar{\Delta u}, \qquad v_y = \frac{h}{f_y \Delta t} \bar{\Delta v}. \tag{16}$$

This formulation maps pixel displacements obtained from optical flow into metric velocity estimates expressed in meters per second. By averaging over multiple reliable features, the estimation process becomes less sensitive to individual tracking errors and local disturbances. The resulting planar velocity estimate serves as the input to the subsequent inertial compensation and sensor fusion stages described in the following sections.

2.9 IMU–Range False Velocity Compensation

Optical flow–based velocity estimates are affected not only by translational motion but also by rotational motion of the quadrotor [32] [33]. In particular, angular rotations of the camera induce structured image motion that can be misinterpreted as translational velocity if left uncorrected. Among the various sources of error that affect optical flow measurements, rotationally induced motion constitutes the dominant and most systematic contribution, especially during aggressive maneuvers or rapid attitude changes.

Rotational effects are deterministic and can be directly corrected using inertial measurements, exploiting their explicit treatment in other visual inertial navigation systems [20] [8], and in the proposed pipeline.

Let $\omega = [\omega_x, \omega_y, \omega_z]^\top$ denote the angular velocity of the quadrotor measured by the onboard IMU, expressed in the body frame. For a downward-facing camera located at altitude h above the observed surface, rotational motion induces an apparent planar velocity component given by the cross product between the angular velocity and the position vector from the camera to the ground plane [34]:

$$\mathbf{v}_{\text{rot}} = \omega \times \begin{bmatrix} 0 \\ 0 \\ h \end{bmatrix} = \begin{bmatrix} \omega_y h \\ -\omega_x h \\ 0 \end{bmatrix}. \tag{17}$$

Let $\mathbf{v}_{\text{opt}} = [v_x^{\text{opt}}, v_y^{\text{opt}}]^\top$ denote the planar velocity estimate obtained from optical flow as described in Section 2.8. The corrected translational velocity is then computed by subtracting the rotational contribution:

$$\mathbf{v}_{\text{corr}} = \mathbf{v}_{\text{opt}} - \begin{bmatrix} \omega_y h \\ -\omega_x h \end{bmatrix} = \begin{bmatrix} v_x^{\text{opt}} - \omega_y h \\ v_y^{\text{opt}} + \omega_x h \end{bmatrix}. \tag{18}$$

This compensation removes the dominant structured error introduced by camera rotation, yielding a planar velocity estimate that more accurately reflects true translational motion. The corrected velocity $\mathbf{v}_{\mathrm{corr}}$ is subsequently used as an input measurement to the sensor fusion framework described in the following section. By explicitly accounting for rotational effects at the measurement level, the proposed approach significantly improves the consistency and reliability of the visual velocity estimates, particularly during dynamic flight conditions.

2.10 EKF-Based Multisensor State Estimation

To fuse the heterogeneous sensor measurements into a single, consistent state estimate, the proposed system employs an Extended Kalman Filter (EKF). The EKF serves as the backbone estimator of the navigation pipeline, integrating inertial, visual, and range measurements to produce real-time estimates of the quadrotor state in GPS-denied environments.

2.10.1 State Representation

The EKF maintains a 16-dimensional state vector defined as

$$\mathbf{x} = \begin{bmatrix} \mathbf{p}^\top & \mathbf{v}^\top & \mathbf{q}^\top & \mathbf{b}_\omega^\top & \mathbf{b}_a^\top \end{bmatrix}^\top, \tag{19}$$

where $\mathbf{p} \in \mathbb{R}^3$ denotes the position of the quadrotor in the world frame, $\mathbf{v} \in \mathbb{R}^3$ is the linear velocity, $\mathbf{q} \in \mathbb{R}^4$ is the unit quaternion representing attitude, and $\mathbf{b}_\omega \in \mathbb{R}^3$ and $\mathbf{b}_a \in \mathbb{R}^3$ represent the gyroscope and accelerometer biases, respectively.

This state formulation explicitly accounts for sensor biases, which is essential for long-term navigation accuracy when operating without external absolute references.

2.10.2 Prediction Model

The EKF prediction step propagates the state forward in time using inertial measurements from the IMU. Let $\mathbf{u}_{k-1}$ denote the IMU input at time step $k-1$, consisting of measured angular velocity and linear acceleration. The nonlinear state propagation is given by

$$\mathbf{x}_{k|k-1} = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}), \tag{20}$$

where $f(\cdot)$ represents the continuous-time inertial motion model discretized over the sampling interval.

The corresponding covariance propagation is

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^\top + \mathbf{Q}_k, \tag{21}$$

where $\mathbf{P}_{k|k-1}$ is the predicted state covariance, $\mathbf{F}_k$ is the Jacobian of the motion model with respect to the state, and $\mathbf{Q}_k$ is the process noise covariance matrix. The process noise accounts for IMU measurement noise and bias random walks, and is tuned based on sensor noise densities obtained from calibration.

2.10.3 Measurement Models

The EKF update step incorporates measurements from multiple onboard sensors. First, the downward-facing camera provides planar velocity measurements derived from the corrected optical flow:

$$\mathbf{z}_{\text{cam}} = \begin{bmatrix} v_x \\ v_y \end{bmatrix}, \tag{22}$$

which directly constrain the horizontal components of the velocity state.

Second, the downward-facing range sensor provides a measurement related to vertical motion. In the proposed formulation, this measurement is modeled as

$$z_{\text{range}} = v_z, \tag{23}$$

providing an additional constraint on the vertical velocity component.

Finally, the gyroscope measurements from the IMU supply angular velocity observations:

$$\mathbf{z}_{\text{imu}} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}. \tag{24}$$

Each measurement is related to the state through a nonlinear measurement function of the form

$$\mathbf{z}_k = h(\mathbf{x}_{k|k-1}) + \mathbf{n}_k, \tag{25}$$

where $h(\cdot)$ is the measurement model and $\mathbf{n}_k$ denotes measurement noise.

2.10.4 EKF Update Step

Given a measurement $\mathbf{z}_k$, the EKF computes the Kalman gain as

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^\top \left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^\top + \mathbf{R}_k\right)^{-1}, \tag{26}$$

where $\mathbf{H}_k$ is the Jacobian of the measurement function with respect to the state, and $\mathbf{R}_k$ is the measurement noise covariance matrix. The measurement noise covariance is derived from sensor calibration statistics, including optical flow variance and range sensor precision.

The state and covariance are then updated according to

$$\mathbf{x}_k = \mathbf{x}_{k|k-1} + \mathbf{K}_k \left(\mathbf{z}_k - h(\mathbf{x}_{k|k-1}) \right), \tag{27}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}. \tag{28}$$

2.10.5 Discussion

This EKF formulation enables direct fusion of camera-based velocity estimates with inertial and range measurements, producing a consistent real-time estimate of the full quadrotor state.

Importantly, improvements in the visual front-end—achieved through the proposed feature selection neural network—directly influence the quality of the measurements provided to the estimator.

The primary benefit of the proposed feature selection strategy to the EKF is the reduction of measurement outliers, which helps preserve the validity of Gaussian noise assumptions in the update step. By pre-filtering unreliable features prior to measurement fusion, the method improves innovation consistency and reduces the likelihood of biased state updates.

As a result, the EKF not only serves as a state estimator, but also provides a system-level validation of the proposed feature-selection framework by reflecting its impact on navigation accuracy and stability.

2.11 Real-Time Multithreaded Implementation

To ensure real-time performance during onboard operation, the proposed navigation pipeline is implemented using a multithreaded software architecture. The primary design objective of this architecture is to maintain high processing throughput while ensuring deterministic latency, which is critical for stable quadrotor flight and reliable state estimation.

The system employs two main threads operating concurrently. The first is a dedicated capture thread responsible for continuous image acquisition from the downward-facing camera. This thread operates independently of the processing pipeline, buffering incoming frames at the camera rate and ensuring that image acquisition is not delayed by downstream computation. By decoupling frame capture from processing, the system prevents frame drops that could otherwise occur during periods of increased computational load.

The second thread handles all vision and estimation-related processing, including classical feature detection, CNN-based feature pruning, optical flow tracking, and planar velocity estimation. This processing thread consumes the most recent available image frame from the capture buffer, allowing the pipeline to operate on up-to-date visual information without stalling the sensor input. The corrected velocity estimates produced by this thread are then passed to the sensor fusion module for state estimation.

This separation of concerns enables the pipeline to sustain processing frequencies exceeding 120 Hz, even under varying scene complexity and motion dynamics. Importantly, the multithreaded design ensures that variations in processing time do not propagate back to the sensor interface, thereby maintaining consistent timing behavior throughout the system. This architectural choice is essential for achieving robust real-time performance on resource-constrained embedded hardware and allows the proposed feature selection framework to be deployed reliably during autonomous flight.

3 Experimental Framework and Validation Methodology

3.1 Hardware Platform

All experimental evaluations are conducted using the Quanser QDrone2 quadrotor platform, which serves as the baseline aerial system for this thesis. The QDrone2 is a research-grade quadrotor designed for autonomous flight experiments in indoor and GPS-denied environments, and it provides a tightly integrated hardware and software ecosystem suitable for real-time state estimation and control.

The platform is equipped with an onboard embedded computing unit based on an NVIDIA Jetson Xavier System-on-Module (SoM), featuring a high-performance ARM-based CPU and an integrated GPU with native CUDA support. This computational capability enables the execution of computationally intensive algorithms, including visual–inertial odometry, real-time optical flow processing, and lightweight neural network inference, entirely onboard without reliance on external processing or communication.

The QDrone2 features a diverse array of sensing modalities. In this work, the primary sensing components used for state estimation include a downward-facing Omnivision OV989 grayscale camera, capable of operating at frame rates up to 210 Hz, which provides high-rate image measurements for optical flow–based velocity estimation. Onboard inertial measurements are obtained from dual 6-DOF Inertial Measurement Units (IMUs), supplying angular velocity and linear acceleration data required for motion propagation and rotational compensation. A downward-facing time-of-flight (ToF) range sensor provides altitude measurements that are used to scale image-plane motion into metric velocity estimates and to compensate for rotationally induced false velocities.

In addition to the sensors used in this thesis, the platform is also equipped with a front-facing Intel RealSense D435 RGB-D camera, three side-facing Sony IMX219 CSI cameras, and a dedicated optical flow sensor. While these sensors are not utilized in the proposed navigation pipeline, their presence highlights the versatility of the platform and its suitability for a wide range of robotics and autonomous systems research applications.

All components of the proposed navigation pipeline—including classical feature detection, CNN-based feature selection, optical flow tracking, rotational compensation, and EKF-based state estimation—are executed onboard in real time. A photograph of the experimental platform and sensor layout is shown in Figure 10.

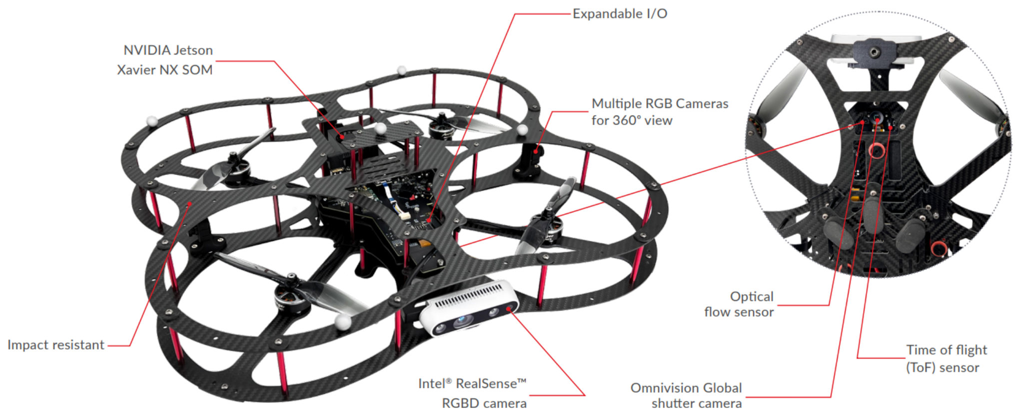

Figure 10: The QDrone2 platform [5].

Another part of the hardware platform is the Ground-Truth information. To quantitatively evaluate state estimation accuracy, ground truth position and velocity measurements are obtained using a motion capture system based on OptiTrack. The system consists of 16 OptiTrack Flex 13 infrared cameras, depicted in Figure 11; the cameras track reflective markers mounted on the quadrotor, providing high-precision six-degree-of-freedom pose measurements at a high update rate.

Figure 11: OptiTrack Flex 13 infrared camera [6].

The OptiTrack measurements are expressed in a global reference frame and are time-synchronized with onboard sensor data. Ground truth velocity is obtained by numerically differentiating the measured position signals using a low-pass filtered finite-difference scheme. These ground truth measurements serve exclusively for evaluation and are not used by the onboard estimator during flight.

The measurement error of the OptiTrack system is negligible compared to the estimation errors evaluated in this work, with manufacturer-reported positional accuracy on the order of sub-millimeters, ensuring reliable ground truth for quantitative performance assessment[35].

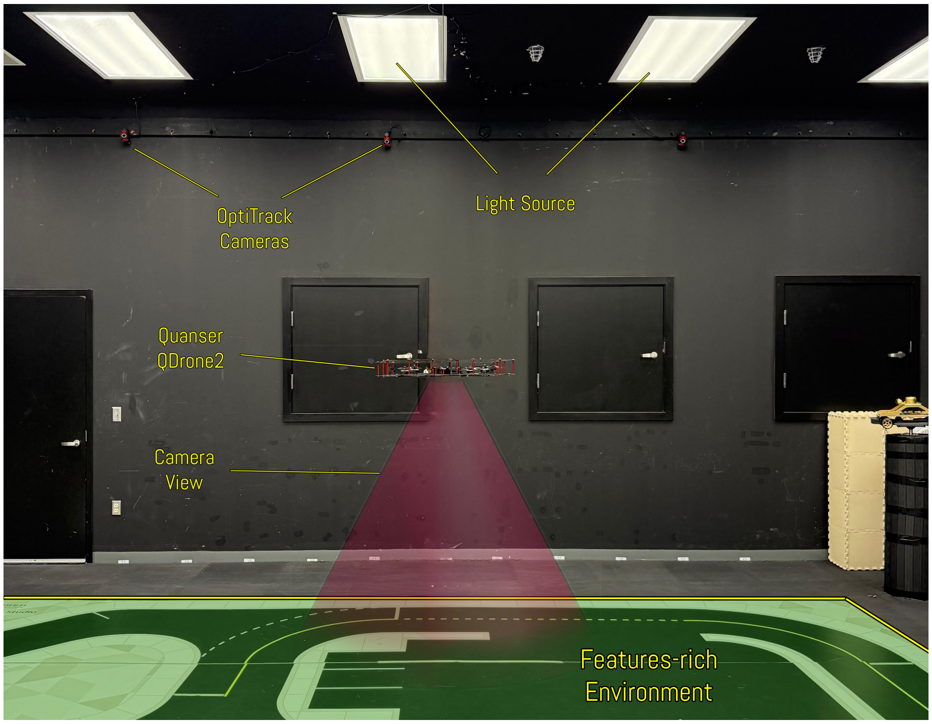

Figure 12: Experimental setup showing the QDrone2 operating over a feature-rich surface with a top-mounted ground truth camera and controlled lighting conditions.

The complete experimental setup shown in Fig. 12 is designed to provide a controlled environment to evaluate the proposed navigation framework under consistent and repeatable conditions. The QDrone2 operates on a feature-rich surface to ensure sufficient visual texture for reliable feature detection and optical flow estimation. A top-mounted ground truth camera system is used to capture reference trajectories, enabling quantitative evaluation of estimation accuracy. The onboard downward-facing camera acquires the frames that will be processed and good features will be extracted for feature tracking and velocity estimation, while a two level light source exists across all experiments. This configuration allows for systematic analysis of the proposed method, isolating the impact of feature selection and ensuring fair comparison across different approaches.

3.2 Sensor Calibration

Accurate sensor calibration is essential for reliable multisensor fusion and long-term state estimation, particularly in GPS-denied environments where estimation errors can accumulate over time. Prior to all experimental flights, a comprehensive calibration procedure is performed for all sensing modalities used by the proposed navigation system.

The intrinsic parameters of the downward-facing monocular camera are calibrated using a standard checkerboard-based calibration procedure. This process estimates the camera focal lengths (f_x, f_y), principal point (c_x, c_y), which are $(\mathit{1050}, \mathit{1050})$ and $(\mathit{640}, \mathit{400})$ respectively, likewise the lens distortion coefficients by minimizing reprojection error over multiple views of a known planar calibration target. The calibrated intrinsics are used to map image-plane optical flow measurements into metric velocity estimates, as described in Section 2.8. Camera calibration is performed using widely adopted computer vision calibration tools, and the resulting parameters are validated by inspecting reprojection residuals.

Extrinsic calibration between the camera, inertial measurement units (IMUs), and range sensor is obtained from manufacturer specifications and verified experimentally. The relative orientations and positions of the sensors are assumed fixed with respect to the quadrotor body frame throughout all experiments. Consistent frame alignment is critical to ensure that visual, inertial, and range measurements are expressed in compatible coordinate frames prior to sensor fusion.

IMU calibration is of particular importance due to the susceptibility of inertial sensors to bias and noise, which can lead to drift if left uncompensated. Gyroscope and accelerometer biases are estimated offline using stationary data collected prior to flight. During this process, the quadrotor is held at rest, allowing the mean bias terms $\mathbf{b}_\omega$ and $\mathbf{b}_a$

to be estimated as

$$\mathbf{b}_\omega = \mathbb{E}[\omega_{\text{meas}}], \qquad \mathbf{b}_a = \mathbb{E}[\mathbf{a}_{\text{meas}}] - \mathbf{g}, \tag{29}$$

where ω_{meas} and $\mathbf{a}_{\text{meas}}$ denote the measured angular velocity and linear acceleration, respectively, and $\mathbf{g}$ is the gravity vector expressed in the body frame.

In addition to offline bias estimation, residual bias drift is explicitly modeled within the EKF state vector as a random walk process. This allows the filter to continuously refine bias estimates online based on sensor measurements and system dynamics, thereby mitigating long-term drift during flight. IMU noise densities and bias random walk parameters are identified from sensor datasheets and stationary data, and are incorporated into the EKF process noise covariance matrix.

The downward-facing range sensor is calibrated to ensure accurate altitude measurements across the expected operating range. Calibration involves validating the sensor output against known distances and compensating for any systematic offset. Accurate altitude estimation is essential for metric scaling of optical flow measurements and for correcting rotationally induced false velocities.

All calibration parameters are fixed prior to experimental evaluation and remain unchanged throughout all experiments to ensure consistency and reproducibility of results.

3.3 Flight Scenarios

To comprehensively evaluate the proposed navigation system under progressively challenging conditions, a set of diverse flight scenarios is designed to stress different aspects of feature tracking, velocity estimation, and multisensor fusion. The selected trajectories vary in geometric complexity, motion dynamics, and visual conditions, enabling a systematic assessment of robustness and accuracy.

The first trajectory consists of a one-dimensional back-and-forth motion along a single horizontal axis. This scenario primarily evaluates the accuracy of planar velocity estimation under simple translational motion, serving as a baseline for subsequent experiments. Although geometrically simple, this trajectory is useful for isolating translational estimation errors and assessing steady-state performance.

The second trajectory extends this motion to two dimensions, where the quadrotor performs back-and-forth motion along both horizontal axes. This trajectory introduces changes in motion direction and requires accurate estimation of velocity components in both axes. It also challenges feature tracking consistency during repeated direction reversals.

The third trajectory follows a circular path in the horizontal plane. Circular motion introduces continuous changes in heading and sustained angular velocity, increasing the coupling between rotational and translational motion. This scenario directly stresses the effectiveness of the rotational false velocity compensation and the robustness of optical flow–based velocity estimation under persistent rotation.

In addition to these trajectories, a cylindrical spiral trajectory is included as the most challenging flight scenario. In this experiment, the quadrotor simultaneously undergoes circular motion in the horizontal plane while varying its altitude, producing coupled translational, rotational, and vertical motion. This trajectory challenges the system's ability to maintain accurate velocity estimation under complex three-dimensional motion and serves as a stress test for the full estimation pipeline.

For each trajectory, experiments are conducted under multiple lighting conditions, including normal indoor illumination and reduced lighting levels. This variation is intended to assess the robustness of feature detection and selection under degraded visual conditions. Furthermore, each trajectory is executed at multiple commanded velocities, allowing evaluation of system performance across different motion intensities. The combination of geometric complexity, lighting variation, and velocity variation provides a comprehensive experimental framework for analyzing the strengths and limitations of the proposed approach.

3.4 Datasets Used

Experimental evaluation of the proposed navigation system is performed exclusively using real-world flight data collected on the Quanser QDrone2 platform. All quantitative results reported in this thesis are obtained from onboard experiments conducted in indoor environments under controlled conditions.

The real flight datasets capture platform-specific dynamics, sensor noise characteristics, and environmental variations encountered during practical deployment. These datasets include synchronized measurements from the downward-facing camera, inertial sensors, range sensor, and OptiTrack motion capture system, enabling direct comparison between estimated and ground truth states. The evaluated flights cover a range of motion profiles, including hover, steady planar motion, and aggressive maneuvers, providing a comprehensive assessment of estimation accuracy and robustness.

Although the EuRoC MAV dataset is utilized during the training phase of the CNN-based feature selection module, as described in section 3.4, it is not used for experimental evaluation in this work. This separation ensures that performance metrics reflect real-world onboard behavior and avoids bias introduced by evaluation on benchmark datasets used during training.

3.5 Evaluation Metrics

The performance of the proposed navigation system is evaluated along three main pillars: accuracy, computational efficiency, and robustness. Together, these metrics provide a comprehensive assessment of the system's suitability for real-time quadrotor navigation in GPS-denied environments.

1) **Accuracy:** Estimation accuracy is quantified using the root mean square error (RMSE) [36] between the estimated and ground truth states. For position estimation, the RMSE is defined as

$$\mathrm{RMSE}_{\mathbf{p}} = \sqrt{\frac{1}{N}\sum_{k=1}^{N} \|\hat{\mathbf{p}}_k - \mathbf{p}_k\|^2}, \tag{30}$$

where $\hat{\mathbf{p}}_k$ and $\mathbf{p}_k$ denote the estimated and ground truth position vectors at time step k, respectively, and N is the total number of samples. Velocity accuracy is evaluated in an analogous manner using planar velocity RMSE. These metrics directly capture the precision of the state estimation pipeline.

2) **Computational Efficiency:** Computational efficiency is evaluated by measuring the real-time performance of the system during flight. This includes the average processing frequency (in Hz) achieved by the full pipeline, as well as the end-to-end computational time between image acquisition and velocity estimation. These measurements assess whether the proposed approach meets the real-time constraints required for onboard quadrotor operation and highlight the computational benefits of the lightweight feature selection strategy.

3) **Robustness:** Robustness is assessed by evaluating system performance across a various conditions. This includes variations in trajectory complexity, commanded velocity, and lighting conditions. Consistent estimation accuracy and stable operation under these diverse scenarios indicate the ability of the proposed method to generalize beyond nominal conditions and to maintain reliable performance in challenging real-world environments.

3.6 Quantification of Lightweight Design

The "lightweight" claim in this thesis is formally defined through quantitative criteria that capture both model-level efficiency and system-level real-time feasibility. Specifically, a method is considered lightweight if it satisfies the following conditions:

1. **Model Compactness:** The total number of trainable parameters remains on the order of 10^5 (i.e., $P < 2 \times 10^5$), ensuring a compact representation that minimizes memory usage and enables fast inference.

2. **Computational Complexity:** The number of floating-point operations (FLOPs) per inference remains in the order of 10^6, significantly lower than deep feature extraction networks which typically operate in the range of 10^9 FLOPs.

3. **Throughput Feasibility:** The full perception pipeline maintains processing frequencies exceeding 30 Hz, corresponding to the minimum real-time requirement for stable quadrotor state estimation and control.

These criteria define the operational boundary for lightweight perception in embedded aerial systems, where computational resources are constrained and estimation must be performed within strict timing limits. The proposed method is evaluated against these criteria in the experimental results presented in Chapter 4.

Lightweight CNN Design Criteria While the architectural structure of the CNN is described in Section 2.6.2, its lightweight nature is characterized here in terms of quantitative efficiency metrics. The network processes grayscale image patches of size 32×32 using a shallow architecture composed of two convolutional layers followed by two fully connected layers.

The total number of trainable parameters is approximately 136,001, which places the model well within the defined compactness criterion. Notably, the first fully connected layer ($2048 \rightarrow 64$) accounts for approximately 96% of the total parameter count, indicating that the computational complexity is dominated by a single low-dimensional projection rather than deep hierarchical feature extraction.

The computational cost of the network is also limited. Due to the small spatial dimensions of the feature maps (32×32 and 16×16), the total number of floating-point operations per inference remains on the order of a few million FLOPs. This is several orders of magnitude lower than conventional deep feature extraction networks.

From a hardware perspective, the compact parameter size (approximately 136k parameters) enables the model weights and intermediate feature maps (up to $32 \times 8 \times 8$) to reside within fast-access cache memory on embedded processors. This reduces reliance on slower external memory access and minimizes inference latency.

Furthermore, the network is designed for a targeted binary classification task—predicting feature reliability rather than dense feature extraction or descriptor learning. This task-specific formulation allows the network to remain shallow while maintaining sufficient representational capacity for robust feature selection under real-time constraints.

3.7 Baseline Methods

To rigorously evaluate the effectiveness of the proposed feature selection framework, comparisons are conducted against representative baseline methods drawn from two dominant classes of visual motion estimation: classical feature-based approaches and learning-based approaches. These baselines are selected to reflect established methods with contrasting computational and robustness characteristics.

The first baseline represents a classical vision-based approach and consists of ORB feature detection combined with Lucas–Kanade optical flow tracking, without learning-based feature pruning or adaptive thresholding. This baseline reflects a widely used configuration for real-time visual velocity estimation on embedded platforms and serves as a reference for evaluating the benefits introduced by the proposed learning-based feature selection and regulation mechanisms.

The second baseline represents a learning-based approach and employs the SuperPoint feature detector, which replaces classical feature extraction with a deep neural network trained to detect robust and repeatable keypoints. SuperPoint has demonstrated strong performance under challenging visual conditions but incurs significantly higher computational cost. This baseline provides a comparison against state-of-the-art learning-based feature extraction methods and highlights the trade-off between robustness and efficiency.

In addition to feature-based approaches, a frequency-domain velocity estimation method based on Fourier Phase Correlation (FPC) is included as a non-feature-based baseline. FPC estimates translational motion directly from image sequences in the frequency domain, avoiding explicit keypoint detection and tracking. This method offers robustness to illumination changes but lacks the ability to leverage long-lived local features, making it particularly suitable for contrasting with feature-based pipelines.

All baseline methods are evaluated under identical experimental conditions using the same sensor data, ground truth measurements, and evaluation metrics. This ensures a fair and consistent comparison, allowing the impact of the proposed feature selection strategy on accuracy, computational efficiency, and robustness to be assessed objectively.

4 Results

This chapter presents the experimental results and analysis of the proposed navigation system. The evaluation focuses on three primary aspects: state estimation accuracy, robustness under varying operating conditions, and computational efficiency. Results are reported using real-world flight experiments conducted on the Quanser QDrone2 platform and are compared against selected baseline methods under identical conditions.

4.1 Lightweight Performance Evaluation

The experimental results demonstrate that the proposed method satisfies the lightweight criteria defined in Section 3.6 while achieving a favorable balance between estimation accuracy and computational efficiency.

The system consistently operates at processing frequencies between 45-62 Hz, significantly exceeding the real-time threshold of 30 Hz required for stable quadrotor operation. In contrast, learning-based approaches such as SuperPoint operate at approximately 5-6 Hz due to their high computational complexity, resulting in nearly an order-of-magnitude reduction in throughput.

From an accuracy perspective, classical methods such as ORB achieve high processing frequencies but exhibit significantly higher estimation error due to the inclusion of redundant and unstable features. The proposed method reduces RMSE by at least 50% compared to ORB while maintaining real-time performance.

These results indicate that the proposed method operates on a favorable latency–accuracy trade-off curve, achieving substantial improvements in estimation accuracy without violating real-time constraints.

Efficiency Interpretation The effectiveness of the proposed approach can be further interpreted in terms of efficiency, defined as the improvement in estimation accuracy per unit of computational cost. By combining selective feature pruning with a lightweight learning-based module, the method improves the quality of visual measurements without introducing significant computational overhead.

The proposed system effectively shifts the Pareto frontier between accuracy and computational efficiency, achieving higher accuracy than classical methods and significantly higher throughput than deep learning-based approaches. This demonstrates that the lightweight characteristic of the method is not solely defined by low computation time, but by its ability to maximize estimation performance within strict resource and timing constraints.

4.2 Overview of Experimental Results

The proposed system is evaluated across multiple flight trajectories with increasing geometric and dynamic complexity. For each trajectory, position and velocity estimates are compared against OptiTrack ground truth measurements. In addition, robustness is assessed by varying lighting conditions and commanded velocities. To provide insight into the underlying performance gains, the behavior of the feature selection and adaptive thresholding mechanisms is analyzed separately. Finally, computational performance is evaluated to verify real-time feasibility.

4.3 Trajectory-Based State Estimation Accuracy

This section evaluates the accuracy of the proposed navigation system across different flight trajectories. For each trajectory, representative position, velocity, and three-dimensional trajectory plots are presented, along with quantitative error metrics. These experiments assess the system's ability to estimate motion accurately under progressively more challenging conditions.

4.3.1 One-Dimensional Back-and-Forth Motion

The first experiment consists of a one-dimensional back-and-forth motion along a single horizontal axis. This trajectory serves as a baseline scenario for evaluating planar velocity and position estimation under simple translational motion. Figure 13 shows the estimated and ground truth position over time, while Figure 14 presents the corresponding velocity estimates.

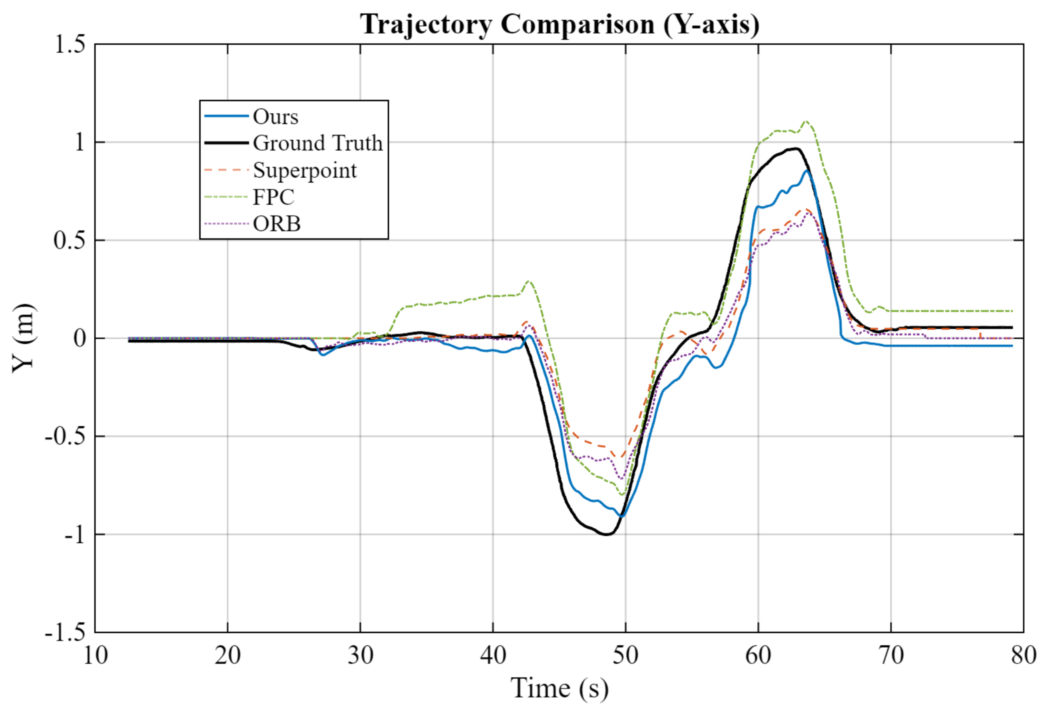

Figure 13: Estimated and ground truth position for one-dimensional back-and-forth motion.

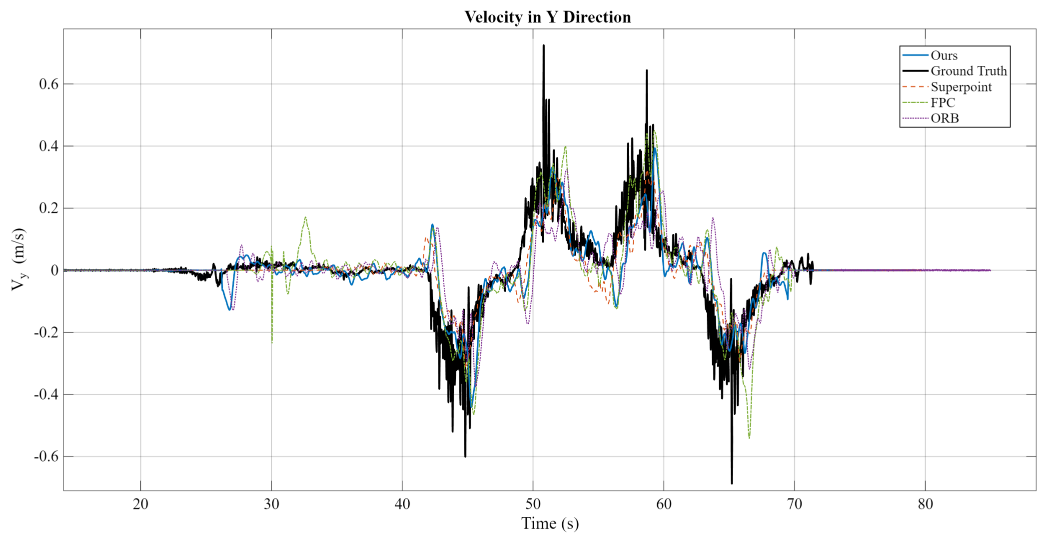

Figure 14: Estimated and ground truth planar velocity for one-dimensional motion.

In addition to the position and velocity plots shown for the one-dimensional back-and-forth motion, Table 7 summarizes the quantitative performance of each method. The table reports the planar position RMSE together with the average per-frame computation time and the resulting processing frequency. This trajectory represents a baseline scenario with simple translational motion, allowing a direct comparison of estimation accuracy and computational efficiency across methods under controlled conditions.

Table 7: Summary of performance for Trajectory 1 (1D back-and-forth). Frequency is computed as $f = 1000/T_{\mathrm{ms}}$.

Method	**RMSE (m)**	**Comp. Time (ms)**	**Freq. (Hz)**
ORB	0.20	10.48	95.4
FPC	0.16	14.49	69.0
SuperPoint	0.15	172.99	5.8
Ours	**0.05**	**16.23**	**61.6**

4.3.2 Two-Dimensional Back-and-Forth Motion

The second experiment extends the motion to two horizontal dimensions, requiring accurate estimation of velocity and position in both axes. This trajectory introduces repeated direction changes and increased sensitivity to feature consistency. Representative position and velocity plots are shown in Figures 15 and 16, respectively.

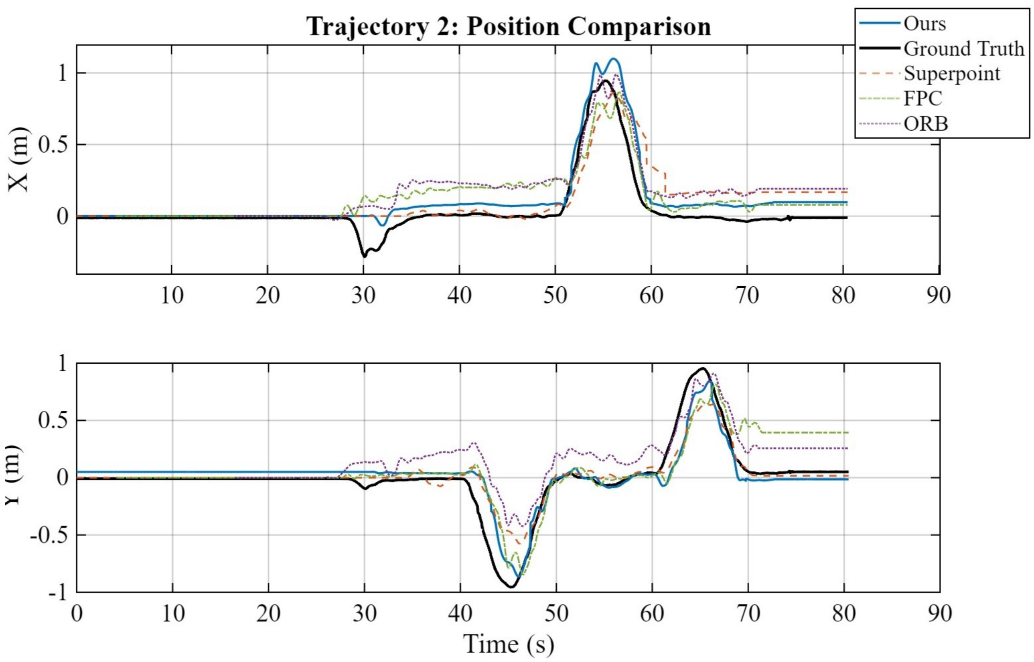

Figure 15: Estimated and ground truth position for two-dimensional back-and-forth motion.

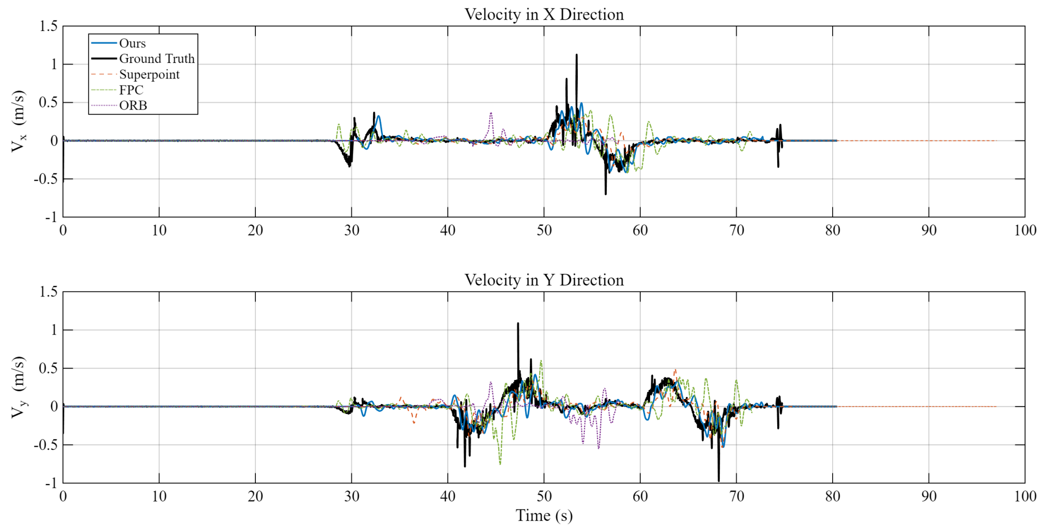

Figure 16: Estimated and ground truth planar velocity for two-dimensional motion.

For the two-dimensional back-and-forth trajectory, Table 8 presents a quantitative comparison of estimation accuracy and computational performance. This experiment introduces increased motion complexity due to simultaneous changes along multiple axes, placing greater demands on feature stability and consistency. The reported metrics complement the trajectory plots by providing a concise numerical comparison across all evaluated methods.

Table 8: Summary of performance for Trajectory 2 (2D back-and-forth). Frequency is computed as $f = 1000/T_{\mathrm{ms}}$.

Method	**RMSE (m)**	**Comp. Time (ms)**	**Freq. (Hz)**
ORB	0.23	16.54	60.5
FPC	0.17	17.76	56.3
SuperPoint	0.11	175.60	5.7
Ours	**0.09**	**19.00**	**52.6**

4.3.3 Circular Trajectory

The circular trajectory introduces sustained rotational motion and continuous changes in heading, placing greater emphasis on rotational false velocity compensation. Figure 17 and Figure 18 illustrate the estimated and ground truth position and velocity, respectively.

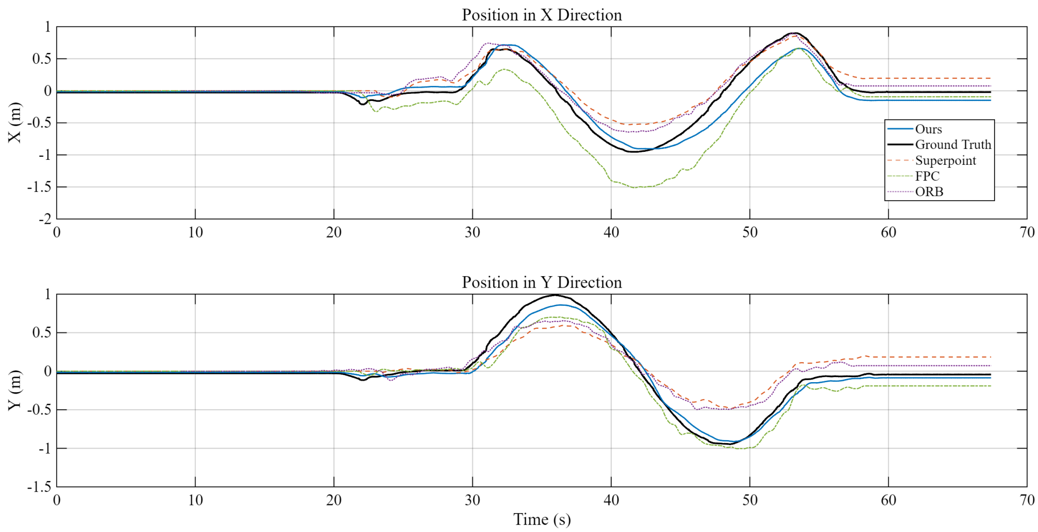

Figure 17: Estimated and ground truth position for circular trajectory.

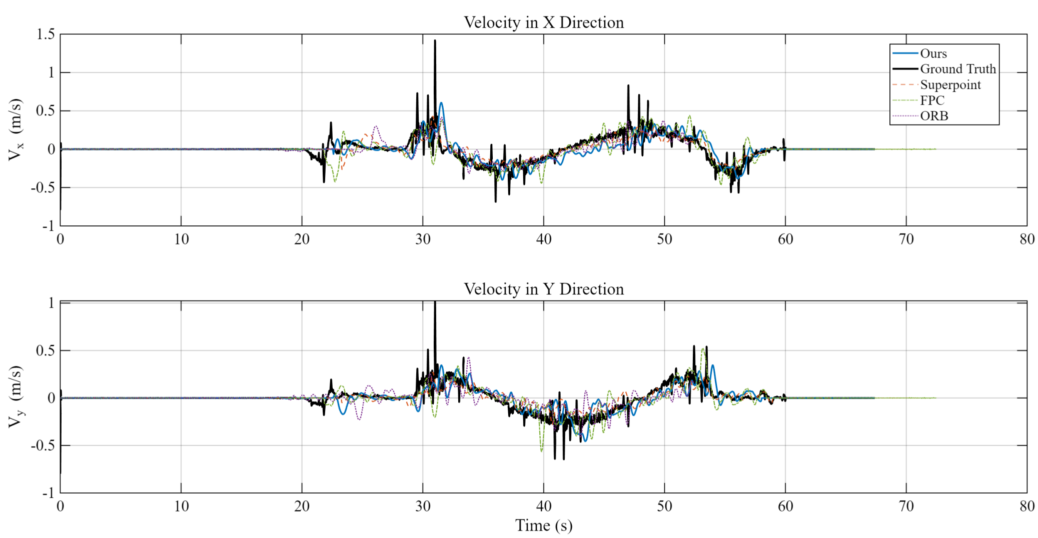

Figure 18: Estimated and ground truth planar velocity for circular trajectory.

Across all evaluated trajectories, the proposed method consistently achieves higher estimation accuracy compared to classical ORB-based tracking and the frequency-domain FPC approach, while maintaining substantially lower computational time than the learning-based SuperPoint method, thereby achieving a favorable balance between accuracy and real-time performance.

Table 9: Summary of performance for Trajectory 3 (circular). Frequency is computed as $f = 1000/T_{\text{ms}}$.

Method	**RMSE (m)**	**Comp. Time (ms)**	**Freq. (Hz)**
ORB	0.11	14.32	69.8
FPC	0.177	16.07	62.2
SuperPoint	0.16	164.30	6.1
Ours	**0.087**	**22.00**	**45.5**

4.3.4 Cylindrical Spiral Trajectory

The cylindrical spiral trajectory represents the most challenging experiment, combining circular motion in the horizontal plane with continuous altitude variation. This trajectory simultaneously stresses translational, rotational, and vertical estimation accuracy. Representative position results are shown in Figures 19–20, and representative velocity results are shown in Figure 21.

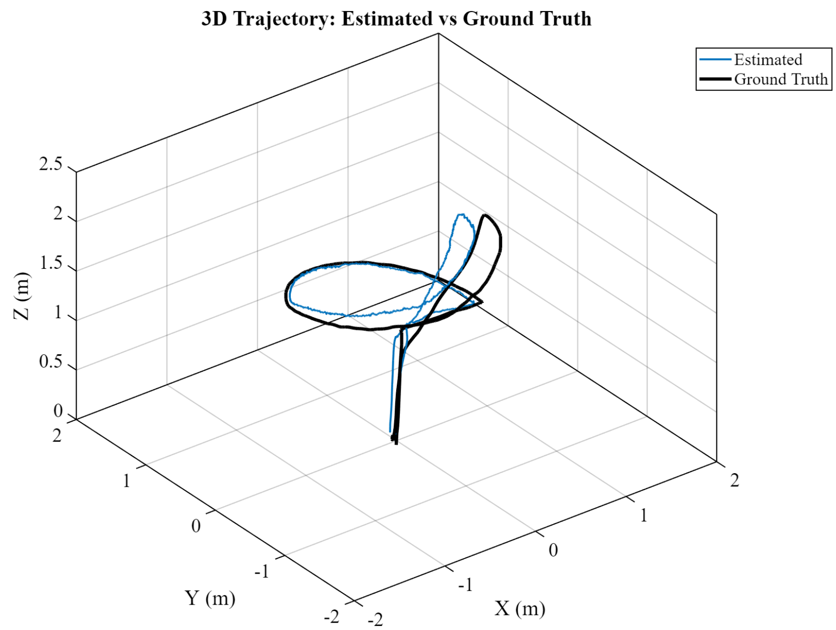

Figure 19: Three-dimensional trajectory for cylindrical spiral motion.

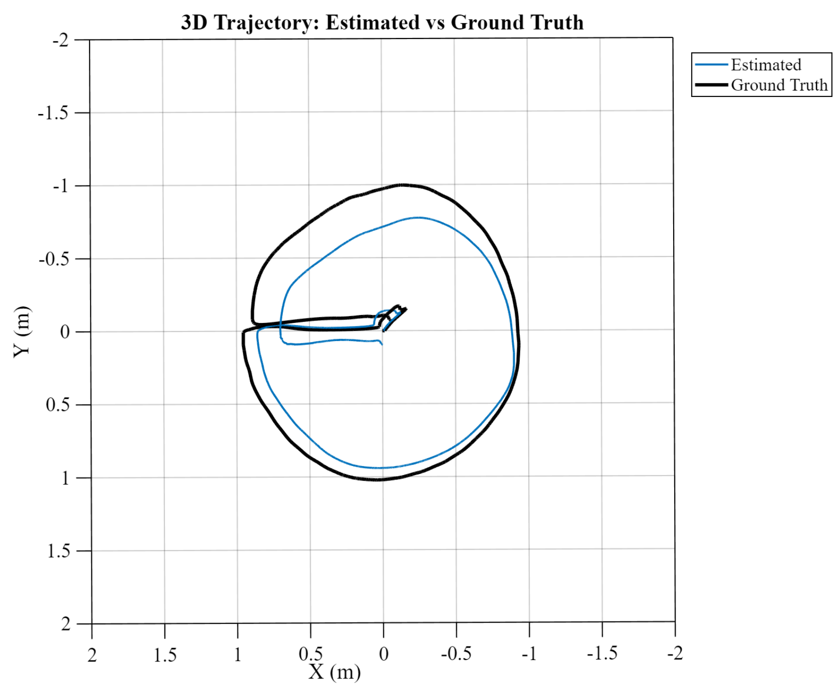

Figure 20: Top-View of clyndrical spiral trajectory.

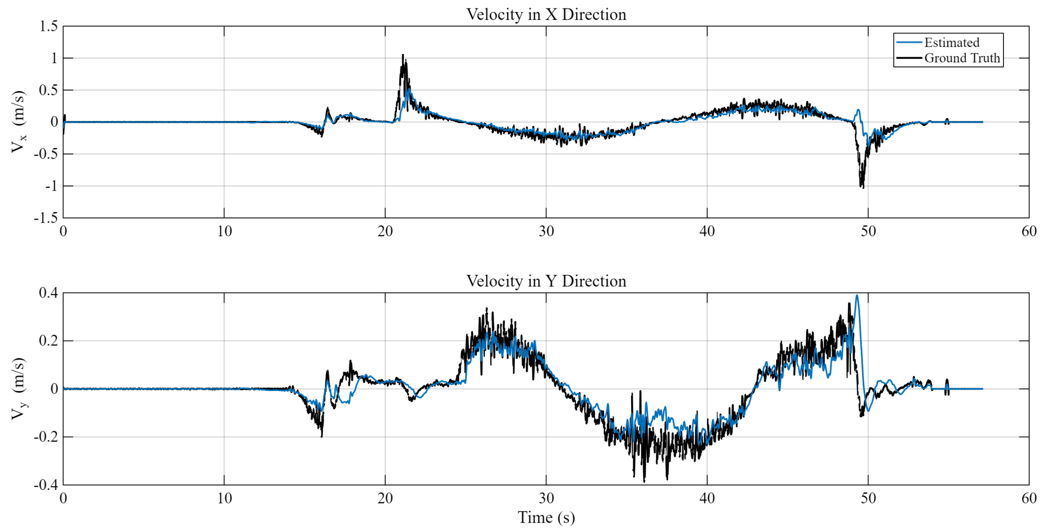

Figure 21: Estimated and ground truth velocity for cylindrical spiral trajectory.

Overall, the cylindrical spiral experiment demonstrates that the proposed system maintains accurate and stable state estimation under simultaneous translational, rotational, and vertical motion. The velocity estimates closely follow the ground truth, while the 3D and top-view trajectory plots confirm consistent position tracking in the horizontal plane. Quantitatively, the achieved RMSE values of 0.07 m in the x direction, 0.109 m in the y direction, and 0.04 m in the vertical axis indicate reliable performance despite the increased trajectory complexity. These results are obtained with an average computation time of 19.32 ms per frame, confirming that the proposed approach preserves real-time capability even in the most challenging experimental scenario.

4.4 Robustness Analysis

This section evaluates the robustness of the proposed system under varying environmental and operational conditions, including changes in lighting and commanded velocity.

4.4.1 Effect of Lighting Conditions

To assess robustness to visual degradation, experiments are conducted under both normal indoor lighting and reduced illumination. Performance is compared across identical trajectories, and representative results are summarized in Figure 22. Despite reduced visual quality, the proposed method maintains stable estimation performance.

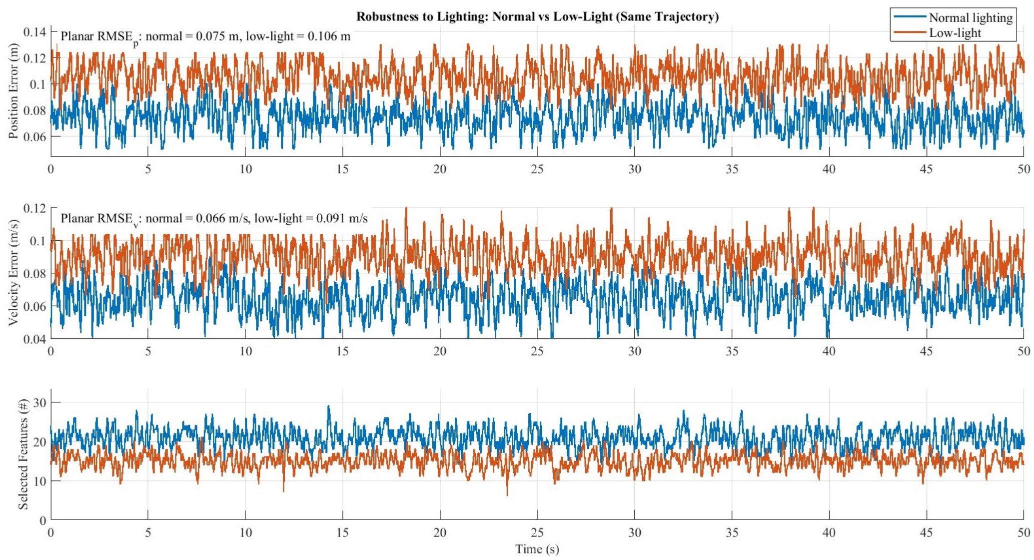

Figure 22: Comparison of estimation performance under normal and low-light conditions.

4.4.2 Effect of Velocity Variations

Each trajectory is executed at multiple commanded velocities (0.3 m/s, 0.5 m/s, and 0.7 m/s) to evaluate sensitivity to motion intensity. Figure 23 illustrates the effect of velocity scaling on planar position estimation accuracy. The results indicate that the proposed system maintains stable performance across a wide range of speeds, with only a marginal increase in error at higher velocities.

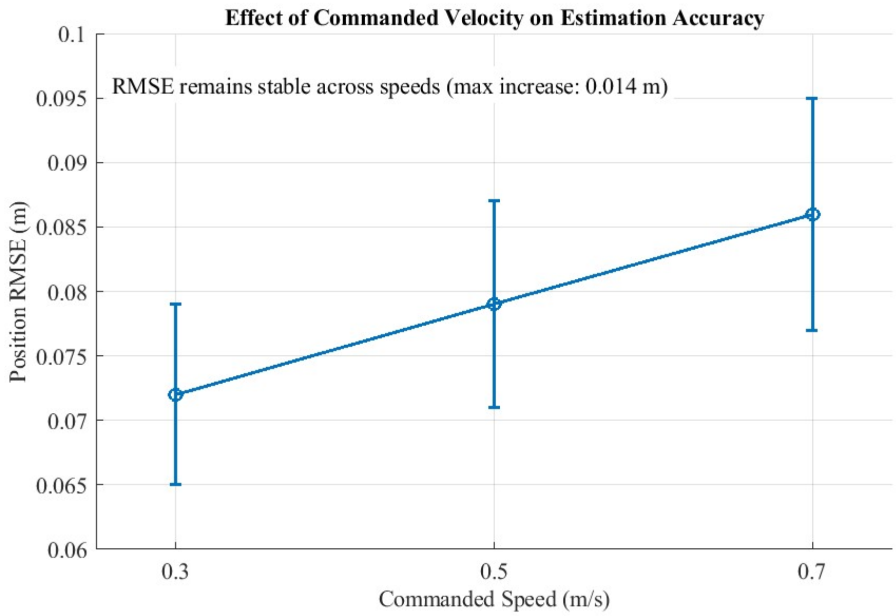

Figure 23: Position RMSE across different commanded velocities.

4.5 Feature Selection and Adaptive Threshold Analysis

While the previous sections demonstrate improved estimation performance, this section analyzes the internal behavior of the proposed feature selection mechanism to provide insight into the source of these improvements.

4.5.1 Qualitative Feature Stability Comparison

This qualitative comparison illustrates the effect of the proposed feature selection mechanism at the image level. Classical ORB detection produces a dense set of features, many of which are clustered in highly textured regions and are prone to short lifetimes. In contrast, the proposed CNN-based pruning retains a smaller subset of features that are more evenly distributed and visually stable, which directly benefits downstream optical flow tracking and state estimation.

Figure 24: Qualitative comparison of feature detection on a representative frame. Right: classical ORB feature detection showing dense and redundant features. Left: CNN-based feature selection retaining fewer but more stable and well-distributed features.

4.5.2 Feature Count and Distribution

Figure 25 illustrates the number of detected and retained features over time. Compared to classical feature extraction, the proposed method maintains a stable number of high-quality features. In addition to Figure 26 shows the distribution of the retained features versus all the features in a complete flight.

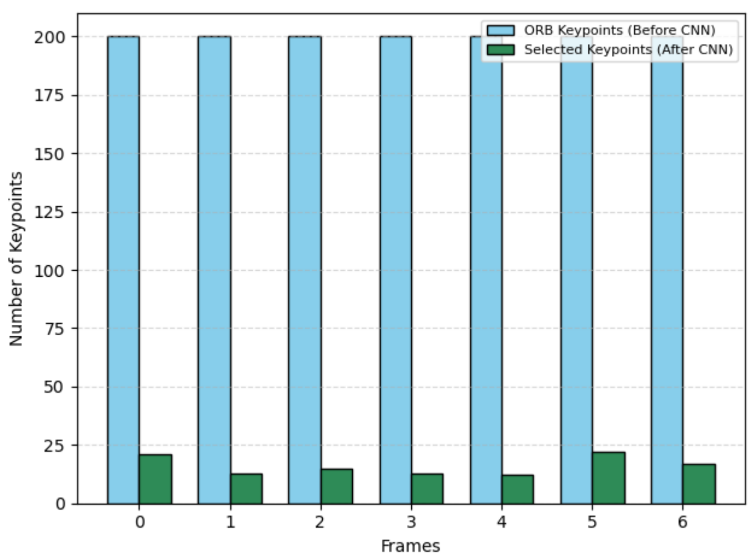

Figure 25: Number of detected and selected features over time.

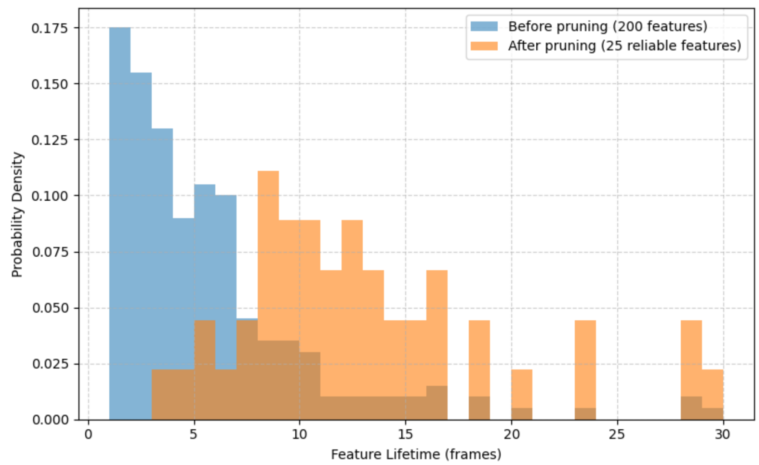

Figure 26: Distribution of the features in a flight.

4.5.3 Adaptive Threshold Behavior

The evolution of the adaptive CNN threshold and corresponding feature count is shown in Figure 27. The threshold adapts smoothly to scene changes, ensuring consistent feature availability.

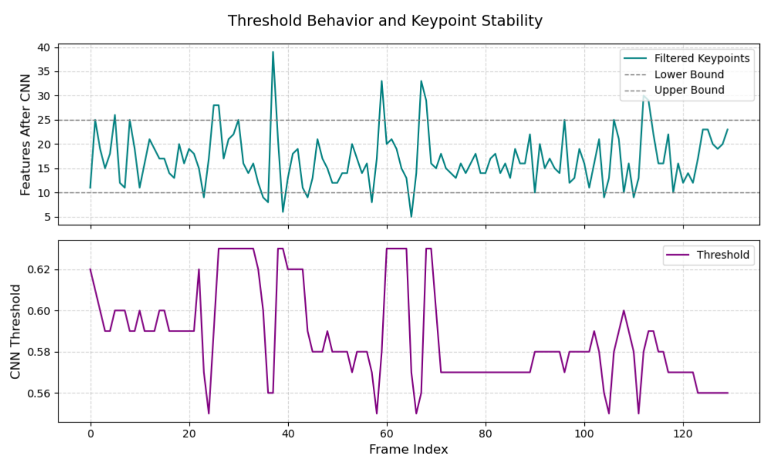

Figure 27: Adaptive threshold evolution and corresponding number of selected features.

4.5.4 Computational Performance

Finally, the computational efficiency of the proposed system is evaluated to assess its suitability for real-time onboard deployment. Figure 28 presents a comparative analysis of the average computation time per frame and the corresponding processing frequency across the evaluated trajectories. The results indicate that the proposed method consistently operates at high processing frequencies, well above the real-time requirements for quadrotor navigation, while maintaining stable and predictable latency. Compared to learning-based baselines, the proposed approach achieves a significantly lower computational cost, and, in contrast to classical methods, it preserves robust estimation performance without sacrificing execution speed. These results confirm that the integration of lightweight learning-based feature selection does not compromise real-time performance and enables efficient onboard execution on resource-constrained aerial platforms.

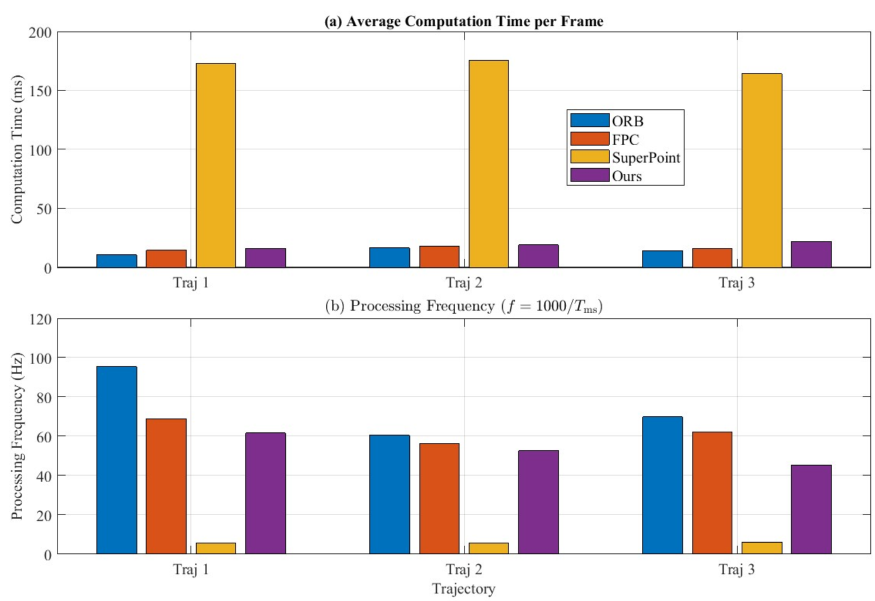

Figure 28: Computation time and processing frequency across different trajectories. The proposed method maintains real-time performance while achieving significantly lower computational cost than learning-based baselines.

4.6 Discussion and Limitations

While the proposed method demonstrates significant improvements in estimation accuracy and computational efficiency, several limitations remain that warrant further investigation.

Dependence on Visual Texture: The proposed framework relies on feature-based optical flow, which inherently requires sufficient texture for reliable feature detection and tracking. In low-texture environments or under poor lighting conditions, the number of detectable features may be significantly reduced, leading to degraded velocity estimation performance.

Generalization of the Learning Module: The CNN-based feature selection module is trained on a specific dataset and may not generalize optimally to unseen environments with different visual characteristics. Variations in lighting, motion blur, or scene structure may affect the reliability predictions, potentially reducing the effectiveness of the feature selection process.

Sensitivity to Dynamic Motion: Although the EKF provides robustness to noise, the system may still experience performance degradation under aggressive maneuvers or rapid changes in motion, where assumptions in the motion and measurement models are violated.

5 Conclusion and Future Work

5.1 Conclusion

This thesis presented a lightweight and adaptive framework for real-time quadrotor navigation in GPS-denied environments, with a specific focus on improving the reliability of visual feature tracking for optical flow–based state estimation. The core motivation of this work was to address a fundamental limitation of classical visual–inertial navigation systems: their strong dependence on the quality and stability of extracted visual features, particularly when operating under challenging conditions such as low texture, varying illumination, and aggressive motion.

To this end, a hybrid classical–learning-based architecture was proposed, combining the computational efficiency of classical feature detection and optical flow with a compact learning-based feature selection module. A lightweight convolutional neural network was introduced to predict feature trackability using patch-level visual information and survival-based supervision. Rather than replacing classical pipelines with heavy deep learning models, the proposed approach selectively improves feature quality by pruning unreliable features while preserving real-time performance on resource-constrained aerial platforms.

An adaptive thresholding mechanism was further developed to regulate the number of retained features dynamically, ensuring stable feature availability across different environments and motion regimes. The resulting filtered optical flow measurements were combined with inertial and range sensing, with explicit compensation for rotationally induced false velocities, and fused within an Extended Kalman Filter framework to produce consistent estimates of the quadrotor state.

The proposed system was implemented fully onboard and evaluated through extensive real-world flight experiments on the Quanser QDrone2 platform. Experimental results demonstrated that the method achieves improved position and velocity estimation accuracy compared to classical optical flow baselines, while maintaining robustness across multiple trajectories, lighting conditions, and flight speeds. In addition, the lightweight design enabled real-time operation at high update rates, outperforming learning-based alternatives that incur significant computational overhead. These results validate the central hypothesis of this thesis: that selectively improving feature reliability, rather than increasing feature density or model complexity, leads to more robust and efficient navigation in GPS-denied environments.

Overall, this work contributes a practical and scalable solution for real-time onboard navigation, bridging the gap between classical computer vision methods and learning-based approaches. The ideas and findings presented in this thesis extend the accompanying ICRA publication by providing deeper theoretical insight, system-level integration details, and comprehensive experimental validation.

5.2 Future Work

While the proposed framework demonstrates strong performance and practical viability, several promising directions remain for future research.

One natural extension of this work is the integration of the proposed feature selection framework into a full visual–inertial odometry or SLAM pipeline. While this thesis focused primarily on planar velocity estimation and state fusion, incorporating the selected features into long-term mapping and loop-closure frameworks could further reduce drift and improve global consistency during extended missions.

Another important direction involves extending the learning-based feature selection to account explicitly for semantic and geometric context. Incorporating additional cues such as surface orientation, depth information from RGB-D sensors, or learned scene semantics could further enhance feature stability in visually ambiguous environments.

Additionally, exploring self-supervised or online adaptation strategies may allow the network to continuously refine its predictions during deployment without offline retraining.

From a systems perspective, future work could investigate tighter coupling between the feature selection module and the state estimator. Rather than treating feature pruning as a preprocessing step, uncertainty-aware integration of feature confidence into the EKF measurement model could provide more principled handling of visual measurement noise and outliers.

Finally, the proposed framework is well suited for extension to more aggressive flight regimes and higher-speed applications such as autonomous drone racing. Evaluating the system under outdoor conditions, extreme illumination changes, and rapid altitude variations would further stress-test its robustness and help generalize the approach beyond controlled indoor environments.

In summary, this thesis lays the groundwork for a new class of lightweight, adaptive, and learning-augmented navigation systems that preserve the efficiency of classical methods while leveraging learning where it is most impactful. The insights gained from this work open several avenues for future exploration toward fully autonomous, reliable, and high-performance aerial navigation in GPS-denied environments.

Bibliography

[1] Google DeepMind, "Gemini (jan 2026 version) [large language model]." `https://gemini.google.com/`, 2026. Accessed: 2026-01-28.

[2] P. Niyishaka and C. Bhagvati, "Digital image forensics technique for copy-move forgery detection using DoG and ORB," in *International Conference on Computer Vision and Graphics (ICCVG)*, pp. 472–483, 2018.

[3] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: Self-supervised interest point detection and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1657–1664, IEEE, 2018.

[4] A. Abosaad and J. Shan, "Lightweight learning-based feature selection for real-time optical flow navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2026. Accepted for publication.

[5] Quanser Inc., "QDrone 2." `https://www.quanser.com/products/qdrone-2/`, 2026. Accessed: 2026-02-05.

[6] NaturalPoint, Inc., "OptiTrack Flex 13 camera." `https://old.optitrack.com/cameras/flex-13/buy.html`, 2026. Accessed: 2026-02-05.

[7] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 298–304, 2015.

[8] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[9] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, “OpenVINS: A research platform for visual-inertial estimation,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, (Paris, France), pp. 2739–2745, 2020.

[10] A. Loquercio, A. I. Maqueda, C. R. del Blanco, and D. Scaramuzza, “DroNet: Learning to fly by driving,” *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1088–1095, 2018.

[11] R. Penicka, Y. Song, M. Wulfmeier, and D. Scaramuzza, “Learning minimum-time flight in cluttered environments,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5719–5726, 2022.

[12] J. Shi and C. Tomasi, “Good features to track,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 593–600, IEEE, 1994.

[13] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, vol. 3951, pp. 430–443, 2006.

[14] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2564–2571, 2011.

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

[16] J. Revaud, P. Weinzaepfel, C. de Souza, M. Humenberger, and G. Puy, "R2D2: Reliable and repeatable detector and descriptor," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 12653–12663, 2019.

[17] Y. Ono, E. Trulls, P. Fua, and K. M. Yi, "LF-Net: Learning local features from images," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 191–201, 2018.

[18] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman Filter for vision-aided inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3565–3572, 2007.

[19] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Proceedings of Robotics: Science and Systems (RSS)*, 2015.

[20] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[21] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multi-map slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[22] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

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

[24] J.-Y. Bouguet, "Pyramidal implementation of the Lucas-Kanade feature tracker: Description of the algorithm," Tech. Rep. 1, Intel Corporation, Santa Clara, CA, USA, 2001. Microprocessor Research Center.

[25] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[26] J. Zhang, S. Singh, and D. Scaramuzza, "Adaptive feature selection for visual odometry in dynamic environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1112–1118, 2021.

[27] C. D. Kuglin and D. C. Hines, "The phase correlation image alignment method," in *Proceedings of the IEEE International Conference on Cybernetics and Society*, pp. 163–165, 1975.

[28] H. Foroosh, J. Zerubia, and M. Berthod, "Extension of phase correlation to subpixel registration," *IEEE Transactions on Image Processing*, vol. 11, no. 3, pp. 188–200, 2002.

[29] H. Deng, D. Li, B. Shen, Z. Zhao, and U. Arif, "Absolute velocity estimation of UAVs based on phase correlation and monocular vision in unknown GNSS-denied environments," *IET Image Processing*, vol. 18, no. 12, pp. 3218–3230, 2024.

[30] A. Gelb, *Applied Optimal Estimation*. Cambridge, MA, USA: MIT Press, 1974.

[31] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York, USA: John Wiley & Sons, 2001.

[32] D. Scaramuzza and F. Fraundorfer, "Visual odometry: Part I—the first 30 years and fundamentals," *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[33] D. Honegger, L. Meier, P. Tanskanen, and M. Pollefeys, "An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1736–1741, 2013.

[34] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. Springer, 2 ed., 2017.

[35] NaturalPoint, Inc., "OptiTrack precision capture quick start guide." `https://docs.optitrack.com/quick-start-guides/quick-start-guide-precision-capture`, 2026. Accessed: 2026-02-05.

[36] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 573–580, IEEE, 2012.

Appendices

A Appendix 1

Algorithm 1 Adaptive CNN Threshold Update

Require: τ_k, N_k, N_{target}, α

Ensure: τ_{k+1}

$\tau_{\min} \leftarrow 0.50, \quad \tau_{\max} \leftarrow 0.80$

$\tau_{k+1} \leftarrow \tau_k + \alpha(N_k - N_{\text{target}})$

if $N_k < N_{\text{target}}$ **then**

    Lower threshold to admit more features

else if $N_k > N_{\text{target}}$ **then**

    Raise threshold to suppress redundancy

else

    No update

end if

$\tau_{k+1} \leftarrow \min(\tau_{\max}, \max(\tau_{\min}, \tau_{k+1}))$

return τ_{k+1}

The adaptive threshold update operates in constant time and requires only the current feature count. The bounded update rule prevents oscillatory behavior while ensuring robust feature availability across varying visual conditions, making it suitable for real-time deployment on embedded platforms.

List of Publications

The work presented in this thesis has resulted in the following publication:

- A. Abosaad, J. Shan, *"Lightweight Learning-Based Feature Selection for Real-Time Optical Flow Navigation,"* accepted for publication in the *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

Parts of this thesis are based on the above publication and have been extended with additional theoretical analysis, system-level integration, and experimental validation.