UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

SAFE LANDING ZONES DETECTION USING AERIAL IMAGERY FOR AUTONOMOUS UAV NAVIGATION

THESIS
PRESENTED
AS A PARTIAL REQUIREMENT
FOR THE DOCTORATE IN INFORMATION SCIENCE AND TECHNOLOGY

BY

SAKINEH ABDOLLAHZADEH

NOVEMBER 2025

This thesis was evaluated by a jury composed of the following persons :

Dr. Soulaimane Berkane ..... Président du jury

Dr. Étienne St-Onge ..... Membre du jury

Dr. Mohamed Dahmane ..... Membre du jury

Dr. Mohand Said Allili ..... Directeur de recherche

Dr. Jean-François Lapointe ..... Codirecteur de recherche

le : 2025-11-13

To my beloved parents, for your love and sacrifices from afar. Being away during my Ph.D. was never easy, but your strength and support always guided me.

To my love, for your unwavering encouragement, belief in me, and warm presence.

And to the brave Iranian women who continue to fight for their rights and freedom.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Prof. Mohand Said Allili, for his invaluable guidance, insightful feedback, and continuous encouragement throughout my research journey. His mentorship played a crucial role in shaping both the technical depth and academic rigor of this work. I also extend my sincere thanks to my co-supervisor, Dr. Jean-François Lapointe, for his support, advice, and thoughtful discussions, which greatly enriched the quality and clarity of this thesis.

Special thanks go to my colleagues and labmates, whose collaboration, shared knowledge, and camaraderie made this experience both productive and enjoyable.

I am deeply grateful to my family for their unwavering love and support, especially during times when distance made their encouragement even more meaningful. Your belief in me has been a constant source of strength.

To my friends, far or close, thank you for being there through every challenge and success, for your understanding, laughter, and companionship that brought balance to my life beyond academia.

And to my love Nemanja, thank you for your steady support, for always believing in me, and for being my anchor through every high and low of this journey.

This thesis was supported in part by collaborative research funding from the National Research Council of Canada's Artificial Intelligence for Logistics Program.

Table of Contents

List of Figures

List of Tables

List of Abbreviations, Acronyms, and Initialisms

BVLOS Beyond Visual Line of Sight

BN Batch Normalization

CBS Convolution + Batch Normalization + SiLU activation

CNNs Convolutional Neural Networks

DPM Deformable Part-Based Models

FCNs Fully Convolutional Networks

FPN Feature Pyramid Network

GNSS Global Navigation Satellite Systems

HOG Histogram of Oriented Gradients

IoU Intersection over Union

LiDAR Light Detection and Ranging

MPA Mean Pixel Accuracy

MRI Magnetic Resonance Imaging

OR-SLZNet Ordinal Regression for Safe Landing Zone Detection Network

PA Pixel accuracy

PAN Feature Pyramid Network

Radar Radio Detection and Ranging

ROI Region of Interest

SIFT Scale Invariant Feature Transform

SiLU Sigmoid Linear Unit activation

SfM Structure from Motion

SLAM Simultaneous Localization and Mapping

SLZ Safe Landing Zone

SPPF Spatial Pyramid Pooling – Fast

SPPNet Spatial Pyramid Pooling Network

SURF Speeded Up Robust Features

SVM Support Vector Machines

UAV Unmanned Aerial Vehicles

YOLO You Only Look Once

Résumé

L'utilisation croissante des véhicules aériens sans pilote (UAV) dans les applications civiles nécessite des méthodes fiables pour la navigation autonome, en particulier pour la détection de zones d'atterrissage sûres (SLZ). Cette thèse explore des méthodologies d'apprentissage profond afin d'améliorer la détection de SLZ à partir d'images capturées par UAV, avec un accent sur la génération de cartes de sécurité denses en temps réel et l'adaptation à des environnements statiques et dynamiques.

Le chapitre 1 présente le rôle croissant des UAV dans les applications modernes et souligne l'importance des atterrissages autonomes sûrs. Il expose les motivations, les défis de recherche et les questions principales qui orientent cette thèse, en mettant en avant l'intégration de la vision par ordinateur dans la navigation des UAV.

Le chapitre 2 fournit une revue complète de l'état de l'art en reconnaissance visuelle. Il couvre à la fois la détection d'objets et la segmentation d'images, en détaillant les méthodes antérieures à l'apprentissage profond (fondées sur des descripteurs classiques) ainsi que les approches modernes basées sur les réseaux de neurones convolutifs (CNN). Une attention particulière est portée aux architectures U-Net, aux fonctions de perte utilisées en segmentation et aux métriques d'évaluation. Le chapitre se termine par une revue de la littérature sur la détection de SLZ, la navigation UAV en environnements dynamiques, et l'utilisation de la régression ordinale par CNN pour l'évaluation de la sécurité.

Le chapitre 3 présente un modèle de régression supervisée utilisant une architecture encodeur-décodeur pour générer des cartes de sécurité denses et continues à partir d'images UAV. Cette méthode permet une évaluation fine de la sécurité du terrain, contrairement aux approches de classification binaire.

Le chapitre 4 introduit OR-SLZNet, un modèle de régression ordinale profonde qui combine des indices photométriques et géométriques tels que la couleur, la profondeur, la planéité et l'inclinaison pour attribuer des niveaux de sécurité à chaque pixel. Le

modèle offre un bon équilibre entre précision prédictive et efficacité computationnelle, ce qui le rend adapté à un déploiement sur des UAV embarqués.

Le chapitre 5 propose un cadre unifié pour la détection de SLZ en environnements dynamiques. Il fusionne la segmentation statique du terrain avec la détection d'objets en temps réel, le suivi multi-objets et la prédiction de trajectoires. Un module de compensation du mouvement par homographie assure l'alignement spatial des images malgré le déplacement de l'UAV, permettant une mise à jour continue des cartes de sécurité.

Les modèles proposés sont évalués sur plusieurs jeux de données publics d'UAV, couvrant des scènes urbaines et naturelles variées, avec des vues nadir et obliques. Les résultats expérimentaux confirment leur efficacité à générer des cartes de sécurité fiables, même dans des conditions difficiles. Toutes les approches ont été conçues pour respecter les contraintes de traitement en temps réel, ce qui les rend parfaitement adaptées à un déploiement sur des plateformes UAV aux ressources limitées.

Abstract

The increasing use of Unmanned Aerial Vehicles (UAVs) in civilian applications necessitates reliable methods for autonomous navigation, particularly for SLZ detection. This thesis investigates deep learning methodologies to enhance SLZ detection using UAV-captured imagery, with a focus on generating dense safety maps in real-time and adapting to both static and dynamic environments.

Chapter 1 introduces the role of UAVs in modern applications and highlights the importance of safe autonomous landing. It outlines the motivation, research challenges, and questions that guide the thesis, with emphasis on the integration of computer vision into UAV navigation.

Chapter 2 provides a comprehensive review of the state of the art in visual recognition. It covers both object detection and image segmentation, detailing pre-deep learning approaches (e.g., traditional feature-based methods) and deep learning-based solutions, including modern convolutional neural networks (CNNs). Special attention is given to U-Net architectures, segmentation loss functions, and performance metrics. The chapter concludes with a literature review of SLZ detection methods, UAV navigation in dynamic environments, and the use of CNN-based ordinal regression for safety assessment.

Chapter 3 introduces a supervised deep regression model based on an encoder-decoder architecture that generates dense, continuous safety maps from UAV imagery. This method allows for fine-grained evaluation of terrain safety compared to binary classification schemes.

Chapter 4 presents OR-SLZNet, a deep ordinal regression model that combines photometric and geometric cues such as color, depth, flatness, and inclination to assign safety levels to each pixel. The model balances predictive accuracy and computational efficiency, enabling deployment on lightweight UAV platforms.

Chapter 5 proposes a unified framework for SLZ detection in dynamic environments. It fuses static terrain segmentation with real-time object detection, multi-object track-

ing, and trajectory prediction. A homography-based motion compensation module ensures consistent spatial alignment despite UAV movement, supporting continuous safety map updates.

The proposed models are evaluated on multiple public UAV datasets covering varied urban and natural scenes with both nadir and oblique views. Experimental results confirm their effectiveness in generating reliable safety maps, even under challenging conditions. All approaches are designed to meet real-time constraints, making them well-suited for deployment on resource-limited UAV platforms.

Chapter 1

Introduction

1.1 UAV Adoption and Applications

Unmanned Aerial Vehicles (UAVs), or drones, have shifted from niche tools to widely adopted civil platforms. Relative to crewed aviation, they offer rapid field deployment in constrained or hazardous settings and increasingly automate aerial data acquisition. With high-resolution imaging and complementary sensors, UAVs provide on-demand measurements that feed operational workflows across sectors [192, 239, 113, 163, 8].

Several factors have accelerated adoption. First, cost effectiveness: compared to manned aircraft, UAVs reduce design, manufacturing, and operating expenses, broadening access to aerial capabilities [113]. Second, accessibility: many systems offer vertical take-off and landing, removing the need for runways and facilitating use in constrained or remote sites. Third, safety: removing onboard crews lowers the risk of casualties in malfunction scenarios, which is crucial for dangerous or complex missions [170]. Fourth, versatility: platforms span sizes and configurations suited to tight spaces and specialized tasks [150]. Fifth, data: integrated cameras and sensors deliver rich measurements for analysis across domains such as agriculture, environment, and infrastructure [150]. Finally, evolving regulations and airspace integration frameworks are enabling safer operations and wider commercial uptake [170].

Across civil sectors, UAVs now enable precision agriculture (crop-health monitoring, soil assessment, variable-rate input application), mapping and surveying (photogrammetry, topographic reconstruction, construction progress monitoring), and infrastructure inspection (bridges, power lines, pipelines, rail corridors, and buildings). They support environmental monitoring and conservation through pollution tracking, habitat and

wildlife observation, and forest-health assessments; they enhance emergency response with rapid scene awareness for search and rescue, disaster assessment, and wildfire monitoring; and they provide time-critical logistics in hard-to-reach communities, notably for medical supplies [92, 191, 20, 34, 93, 150, 170].

1.2 Autonomous Safe Landing Zone Detection for UAVs

Regardless of the application, every UAV mission must ultimately end with a safe landing that protects people, infrastructure, and the drone itself. The landing phase is often the most safety-critical portion of the flight, as it typically occurs in cluttered, constrained, or partially unknown environments and may coincide with degraded sensing, communication, or power margins. Ensuring that the landing decision can be made autonomously and reliably—rather than relying solely on a remote pilot's situational awareness—is therefore a central requirement for scalable, repeatable UAV operations.

A key barrier to fully realizing the potential of UAVs is the design of reliable autonomous navigation systems. Beyond reducing operator workload, autonomy must support safe maneuvering around obstacles, robust operation under uncertainty, and resilience to degraded or unavailable Global Navigation Satellite Systems (GNSS) or communication links. These capabilities are essential for beyond-visual-line-of-sight (BVLOS) operations—flights where the remote pilot cannot maintain direct visual contact—because the aircraft must sense and avoid, replan, and execute contingencies without continuous human input, enabling safe, compliant integration into civilian airspace [150, 89, 107].

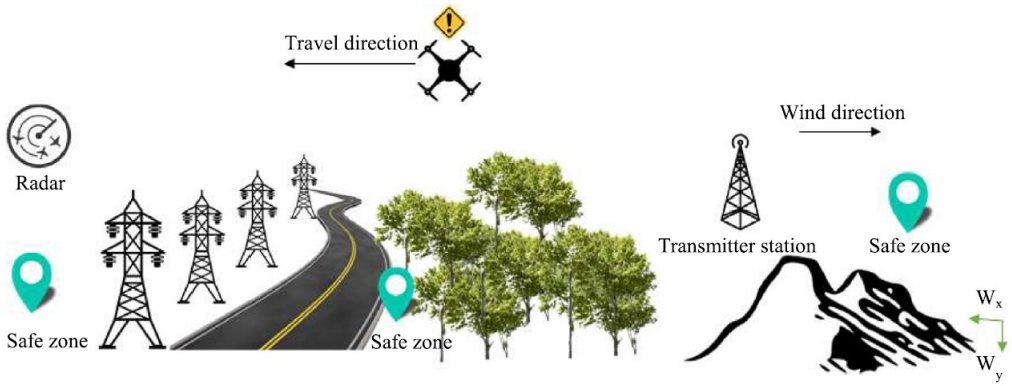
Achieving this level of autonomy requires a comprehensive perception stack that fuses complementary onboard sensing to provide situational awareness and inform decision making. Four core functionalities are particularly critical. Internal state awareness monitors platform health via real-time telemetry (e.g., power, temperature, vibration, fault indicators), enabling self-diagnosis and safe fallback behaviors [2]. Environmental perception characterizes the external scene—terrain, structures, and weather—using multimodal sensors such as RGB and depth cameras, LiDAR, radar, and thermal imaging [46, 17], supporting semantic understanding and terrain assessment. Target and landmark recognition detects, classifies, and tracks salient objects (e.g., people, vehicles, markers), a capability that has been significantly strengthened by deep learning-based detectors [175, 205]. Finally, obstacle detection and classification identifies and differen-

tiates static and dynamic hazards, leveraging sensor fusion and learning-based segmentation to enable robust collision avoidance under diverse environmental conditions [54].

Safe Landing Zone (SLZ) detection operates at the intersection of these modules. It integrates semantic understanding with geometric assessment to evaluate candidate landing sites in terms of both static terrain properties (surface type, inclination, roughness, structural integrity) and dynamic constraints, including moving agents, transient obstacles, and short-term changes in the environment. Vision-based SLZ approaches are particularly attractive due to their low cost, high spatial resolution, and compatibility with embedded computing platforms. Recent methods train deep networks on annotated aerial datasets to generate dense, pixel-wise safety maps that score or rank feasible landing areas in real time [101, 168, 71]. Such maps can be continuously updated along the flight path, enabling context-aware selection of nominal, alternate, and emergency landing sites.

Reliable SLZ detection is therefore a foundational capability for civil UAV operations rather than an optional add-on. First, it enhances safety by reducing the risk of collision with people, infrastructure, vehicles, or natural obstacles during both nominal and contingency landings [92, 129]. Second, it increases mission robustness: when weather deteriorates, the planned site becomes unusable, or GNSS/communications are lost, the UAV can autonomously identify suitable alternatives within its reachable footprint (see Figure 1.1). Third, it improves operational efficiency by limiting time spent hovering or searching for viable sites, conserving limited battery or fuel and extending effective range and task throughput. Finally, by enabling consistent landing decisions across heterogeneous environments—urban, rural, industrial, coastal, or mountainous—SLZ detection supports scalable deployment of autonomous UAV services in realistic, safety-critical conditions [23, 100, 89].

Figure 1.1: Examples of emergencies that occur when a UAV is in flight [92].



1.2.1 Key Factors in SLZ Selection

Selecting a SLZ is pivotal for flight safety and mission success. The assessment spans two complementary domains, site conditions and obstacle environment that must be judged jointly to determine suitability [151, 1, 56].

Site conditions: The landing surface should provide sufficient usable area relative to the vehicle's footprint (wingspan or rotor disc), with an additional safety margin to accommodate touchdown dispersion, rollout, and minor crosswind corrections. Local slope is ideally near level to minimize tipping or post-contact sliding, as excessive inclination can destabilize the airframe. Surface roughness and texture also matter: large debris, holes, vegetation, or protrusions threaten landing gear integrity and can induce shocks that compromise state estimation. When possible, the site orientation should allow approaches into the prevailing wind to aid deceleration, flare, and lateral control.

Obstacle environment. A clear approach/egress corridor is required to maintain safe clearance during descent, flare, and potential go-around. Tall obstacles in the near field (e.g., buildings, trees, poles, terrain ridges) reduce vertical margins, while insufficient horizontal standoff limits options for missed approaches or emergency maneuvers. Ensuring a clear glide path with adequate lateral buffers mitigates gust-induced drift and reduces the chance of sensor occlusion.

Additional operational factors. The ground should possess adequate bearing capacity to prevent gear sinkage or overturning an especially important consideration for heavier platforms and saturated or compliant surfaces. Illumination and visibility must

be sufficient for detection and alignment; in dusk, night, or degraded visual environments, either the SLZ should be illuminated or the UAV equipped with suitable lighting or thermal sensing. Dynamic hazards (people, vehicles, animals), overhead utilities, water bodies, and sloped rooftops introduce additional risk and should be excluded from candidate sites.

A disciplined evaluation of these cues lowers the likelihood of damage or mission loss. Contemporary SLZ detection pipelines encode such geometric, appearance, and contextual features into learning-based models to deliver real-time suitability maps and risk scores [85, 241].

1.2.2 Types of Landing Zones

Understanding how landing zones are characterized and classified is essential for designing robust guidance, navigation, and control pipelines. At a high level, Figure 1.2 summarizes common categories across operating environments (indoor vs. outdoor) and platform dynamics (static vs. dynamic), highlighting how each combination imposes distinct sensing and planning requirements [191].

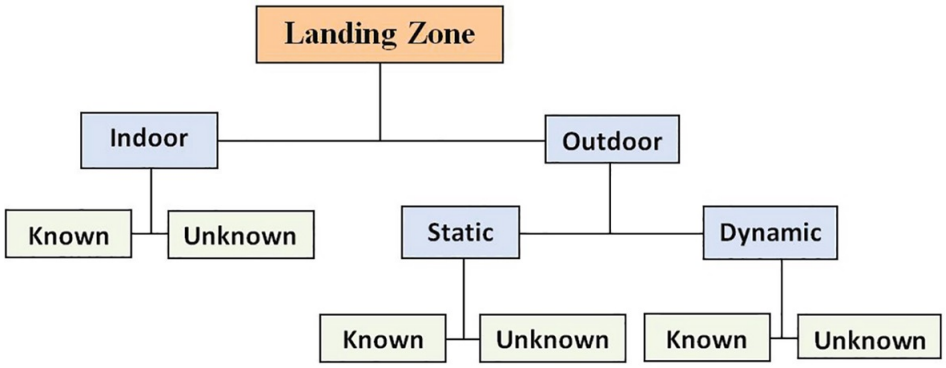


Figure 1.2: Different types of landing zones [191].

Indoor environments are typically structured and space-constrained, with GNSS often unavailable or unreliable. Candidate landing areas tend to be planar and relatively static (e.g., factory floors, warehouse aisles, inspection halls), yet clutter, tight corridors, and moving agents still complicate approach and touchdown. SLZ detection indoors generally relies on vision (e.g., cameras detecting fiducials, high-contrast patterns, or texture-poor flat patches) complemented, when needed, by LiDAR or depth sensing to

recover geometry and free space. Onboard perception and compute must operate in real time to localize markers, estimate surface suitability, and maintain safe corridors for approach and aborts [191].

Outdoor environments encompass delivery, inspection, security, agriculture, and humanitarian missions, and are inherently more variable. Weather, illumination, terrain, and dynamic obstacles (vehicles, pedestrians, animals) require fusing appearance cues with geometry and wind-aware planning. In practice, pipelines integrate obstacle detection, terrain and roughness estimation, and short-horizon trajectory generation with wind and drift compensation so that approach, flare, and go-around remain feasible under changing conditions [191].

Platform dynamics introduce a complementary axis of complexity. Static landing zones are fixed, flat areas such as helipads, rooftops, runways, or open fields. When these are known static sites, they are often pre-surveyed or visually marked (e.g., geometric shapes, high-contrast paint) so that onboard vision can lock onto expected patterns and refine pose for precision touchdown. By contrast, unknown static sites lack prior labels or fiducials, requiring the vehicle to search and rank candidate patches online using flatness, slope, roughness, and clearance cues; in more complex scenes, LiDAR or thermal/infrared can assist in disambiguating safe surfaces [191]. Dynamic landing zones are moving platforms (e.g., trucks, ships, buses). Known dynamic platforms may carry distinctive markers or be cataloged a priori, enabling target recognition and motion-aware tracking; landing then couples visual servoing with predictive control that accounts for the platform's velocity and heading. Unknown dynamic platforms are unmarked and present the most challenging case: the UAV must detect, track, and continually reassess feasibility in the presence of platform motion, occlusions, and environmental disturbance, while preserving safe abort options [191].

Complementing this environment and dynamics view, Figure 1.3 (from [226]) proposes a vision-centric taxonomy that groups autonomous landing scenes as static, dynamic, or complex, with subclasses defined by the detection target (e.g., marked pad vs. unmarked terrain vs. moving vehicle). This perspective emphasizes the perception bottlenecks, including marker detection, terrain suitability estimation, obstacle clearance, and motion-compensated tracking, which are especially salient in GPS-denied or emergency landings where decisions must be taken with minimal delay and strong safety guarantees [226].

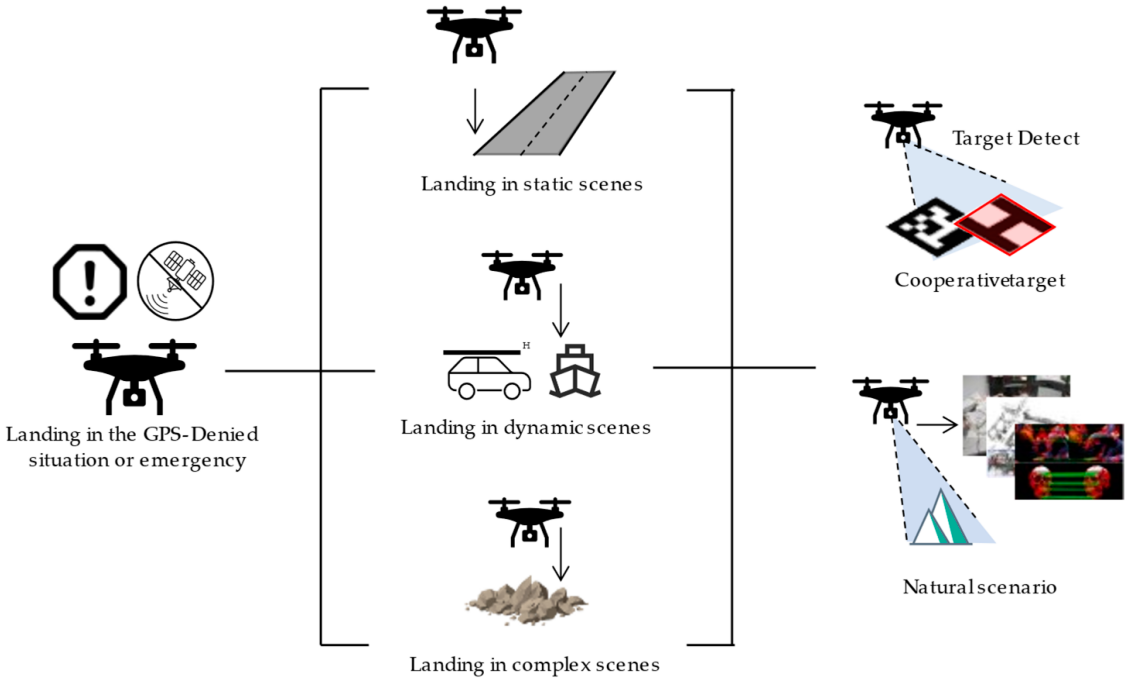


Figure 1.3: Vision-based classification of autonomous UAV landing scenarios in GPS-denied or emergency conditions. The framework distinguishes between static, dynamic, and complex scenes, with target detection strategies categorized into cooperative and natural scenarios [226].

1.3 Why Vision-Based SLZ Mapping Matters

SLZ mapping aims to determine where an unmanned aerial vehicle (UAV) can land safely, and with what level of risk. A useful SLZ map does more than separate landable from non-landable areas: it provides a graded assessment that accounts for obstacles, surface quality, dynamic agents, and uncertainty. For small and medium UAVs, passive vision is particularly well suited to this task. Cameras are light, inexpensive, and energy-efficient, while modern learning-based methods can convert image streams into landing decisions directly onboard the platform [160, 20].

A first reason for using vision lies in semantic understanding. Landing safety depends on what is present on the ground, not only on whether an area appears empty. Vision-based models can distinguish between surfaces such as short grass, concrete, water, shrubs, vehicles, or human crowds, and detect thin or small structures that are critical for safety, such as poles, fences, or cables [232, 241]. Two regions with similar geometry may have very different risk profiles; access to appearance and context allows the SLZ

map to reflect these differences and to provide interpretable reasons for labeling a region as safe or unsafe.

Vision also contributes geometric information without requiring heavy active sensors. Using monocular or stereo depth cues, visual odometry, and structure-from-motion, the UAV can estimate local slope, roughness, height discontinuities, and distances to obstacles [115, 35, 86]. Optical flow further supports the detection of looming obstacles and helps assess relative motion during approach [145]. Although individual estimates may be noisy, combining observations over several frames and viewpoints generally yields geometric information that is sufficient for assessing touchdown stability and clearance.

In addition, vision enables SLZ mapping to account for dynamic elements. Object detection and tracking can identify and follow pedestrians, vehicles, or other moving agents in the scene [144]. This allows the system to down-rank regions that are currently free but likely to be occupied shortly, which is essential during final approach when the situation can change within a few seconds. When required, simple activity cues can also improve awareness in shared or constrained environments [108].

From a systems perspective, vision integrates well with real-time, onboard processing. Current embedded accelerators can run compact neural networks at sufficient frame rates to update the SLZ map throughout descent. This enables a closed-loop interaction with guidance and control: as altitude decreases and visibility improves, the system refines candidate regions, rejects zones that become occluded or unsafe, and authorizes landing only when the estimated safety level exceeds a predefined threshold. Because the outputs are spatial and semantic, they can also be visualized as safety maps and hazard overlays, which facilitates human supervision, debugging, and compliance with explainability requirements [242, 160].

Other sensing modalities remain important but play a complementary role. LiDAR provides accurate 3D measurements but often at higher cost, mass, and power consumption, which can be limiting for small UAVs or large fleets [129]. Radar is robust in adverse weather and over long ranges but offers limited resolution for fine-grained landing decisions [195]. Thermal cameras are valuable at night or for search-and-rescue but do not provide the detailed textures and colors that many vision models exploit for general SLZ assessment [10]. In contrast, passive vision cameras are comparatively low-cost and lightweight while still offering high spatial resolution, making them particularly attractive as the primary sensing modality for SLZ perception. In hybrid configurations, cameras typically form the core perception source, while inertial sensors, altimeters,

GNSS (when available), and selective active ranging are fused to improve robustness in challenging conditions [104, 48].

Finally, vision-based SLZ mapping benefits from a practical data and deployment ecosystem. As illustrated in Figure 1.4, our AI-embedded vision model architecture enables real-time, onboard SLZ perception and decision support. Training data can be collected directly from UAV operations, augmented synthetically, or generated in simulation, making it possible to expose models to diverse environments, seasons, and rare hazards. Domain adaptation techniques then help maintain performance as operating conditions change. Together, the semantic richness, geometric cues, dynamic awareness, favorable size–weight–power characteristics, and data efficiency of camera-based systems explain why vision is increasingly adopted as the primary modality for SLZ mapping, and why it serves as the foundation for the risk-aware landing framework developed in this thesis [129, 242, 160].

1.4 Key Challenges

Developing a reliable and efficient system for SLZ detection in UAV operations involves addressing several tightly coupled challenges that arise from both the complexity of real-world environments and the constraints of onboard platforms. First, SLZ perception must operate in dynamic and uncertain scenes. A region that appears safe at one instant may rapidly become hazardous due to the motion of vehicles, pedestrians, animals, or other UAVs. Consequently, the system must continuously update its assessment of candidate landing areas by coupling terrain analysis with real-time detection, tracking, and short-horizon motion reasoning, while running at low latency on resource-constrained embedded hardware [71, 101]. Ensuring that such models remain both responsive and accurate under operational constraints is a central difficulty.

A second challenge is the extraction of semantically meaningful safety cues from visual data. Raw images must be converted into estimates of surface type, roughness, structural integrity, slope, and contextual usage, while also identifying hazardous elements such as cables, poles, vehicles, or crowds [168, 191]. This often requires the fusion of complementary cues (e.g., RGB appearance, depth proxies, flatness, or inclination maps) under varying illumination, viewpoints, and weather conditions. Designing models that can robustly learn and generalize these relationships, rather than overfitting to specific environments or textures, remains non-trivial.

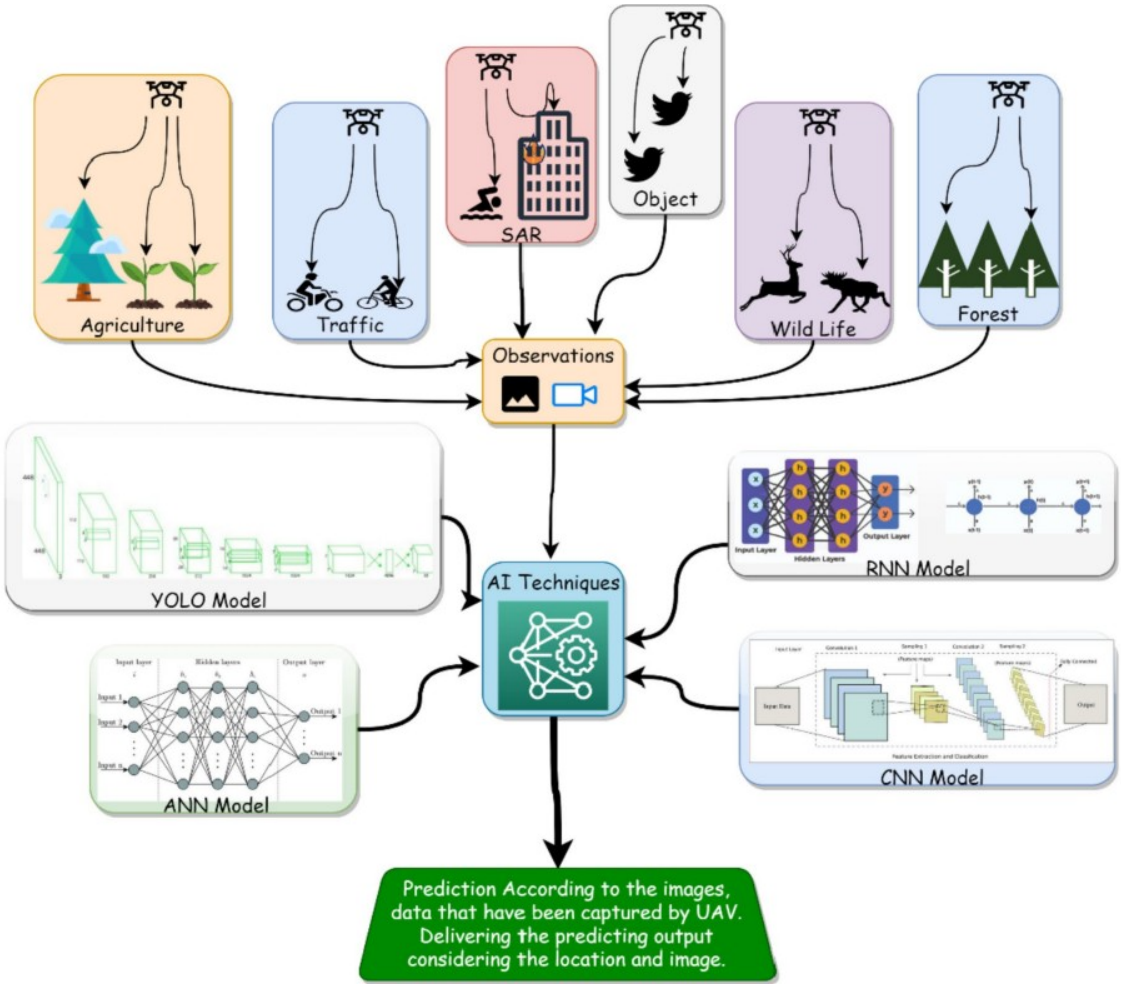


Figure 1.4: AI-embedded vision model architecture for real-time, onboard SLZ perception and decision support [163].

A third difficulty lies in jointly modeling static and dynamic cues within a single SLZ framework. Many existing approaches treat terrain classification, obstacle detection, and motion analysis as separate components, which complicates temporal consistency and may lead to conflicting decisions [226, 71]. In practice, however, landing safety depends on the interaction between long-term static properties (e.g., surface geometry and material) and transient events (e.g., a pedestrian crossing a previously safe area). An effective SLZ system must therefore integrate high-resolution semantic maps with time-sensitive motion information in a coherent, spatially aligned, and temporally stable representation.

These perception requirements are further constrained by the limited computational, memory, and energy budgets of UAV platforms. High-resolution aerial imagery and multi-modal inputs are costly to process in real time, especially when employing deep neural networks with millions of parameters [101, 191]. Achieving an acceptable trade-off between accuracy, complexity, and latency demands compact architectures, efficient feature sharing, and task-aware optimization strategies that preserve safety-critical performance while respecting onboard limitations.

Generalization across diverse operating conditions represents another major challenge. In real deployments, UAVs encounter heterogeneous environments, including urban areas, roads, rooftops, industrial sites, natural fields, coastal regions, and unstructured terrain, observed from varying altitudes and viewing angles, and populated by different types of objects and activities [191, 168]. Appearance shifts due to geography, season, illumination, and sensor characteristics can significantly degrade the performance of models trained on limited datasets. Robust SLZ detection therefore requires representations and training strategies that maintain reliable performance across domains without exhaustive environment-specific annotation.

Finally, safe decision-making must be maintained under uncertainty and degraded sensing. Adverse weather (haze, fog, rain, snow), low sun angles, shadows, glare, motion blur, occlusions, and sensor noise can reduce the reliability of both semantic and geometric estimates [71, 226]. In such conditions, the system must avoid overconfident predictions, propagate uncertainty into the SLZ map, and, when necessary, delay or refuse landing until sufficient evidence is available. Developing models that explicitly handle incomplete or ambiguous information, and that can adjust their behavior according to confidence levels, is essential to guarantee operational safety.

In light of these challenges, this thesis proposes a vision-based SLZ mapping framework that integrates semantic segmentation, deep regression, ordinal safety ranking, and dynamic obstacle forecasting into a unified and computationally efficient architecture. The proposed approach is designed for real-time onboard deployment, evaluated across diverse datasets and viewpoints, and analyzed under both nominal and degraded conditions to demonstrate its robustness and applicability to real-world UAV landing scenarios.

1.5 Research Problem

The growing use of Unmanned Aerial Vehicles (UAVs) in civilian, industrial, and emergency-response applications demands reliable mechanisms for identifying SLZs under nominal, degraded, and time-critical conditions. In many practical scenarios such as GNSS outages, communication loss, low battery, system faults, or sudden environmental changes the UAV must autonomously select a landing site using only onboard sensing and computation. Passive vision sensors are particularly attractive for this task due to their favorable size–weight–power characteristics and their ability to support both semantic scene understanding and geometric inference. However, exploiting vision to produce landing decisions that are sufficiently accurate, interpretable, and robust for real-world deployment remains a challenging open problem.

Existing SLZ detection approaches present several critical limitations. A large body of prior work relies on binary safe/unsafe classification or on coarse, rule-based heuristics, which fails to reflect the graded nature of landing risk and provides limited flexibility for decision-making in constrained or partially suitable environments. Many methods focus on static or simplified scenes, neglecting dynamic obstacles such as pedestrians, vehicles, or other UAVs and ignoring how short-term motion patterns influence future occupancy of candidate landing areas. Other solutions depend on costly or heavy active sensors (e.g., LiDAR) or on computation that exceeds the capabilities of embedded platforms, hindering deployment on small and medium UAVs. Furthermore, existing evaluations are often dataset- or scenario-specific, with limited evidence of generalization across diverse viewpoints, terrains, and environmental conditions.

From an operational perspective, safe landing decisions must integrate multiple sources of information into a coherent representation: semantic attributes of the terrain (e.g., roads, grass, rooftops, water), geometric properties (e.g., flatness, slope, roughness), dynamic agents and their predicted motion, and the ego-motion of the UAV itself. In most prior work, these components are treated as disjoint modules whose outputs are combined through ad hoc rules, leading to inconsistencies, latency issues, and a lack of principled handling of uncertainty. There is a need for a unified, image-aligned SLZ mapping framework that continuously updates landing suitability in real time, explicitly models safety as an ordered quantity, and remains compatible with the computational and energy constraints of onboard hardware.

The central research problem addressed in this thesis is therefore to design, implement, and evaluate a vision-based SLZ mapping framework that: (i) produces dense, fine-grained and interpretable safety maps instead of binary decisions; (ii) leverages both photometric and geometric cues to capture the structure of landing risk; (iii) extends naturally to multi-level ordinal safety scales suitable for risk-aware planning; (iv) incorporates dynamic obstacles and UAV ego-motion to maintain a temporally consistent assessment of landing feasibility; and (v) satisfies real-time and resource constraints for deployment on practical UAV platforms. The subsequent chapters develop and validate this framework through continuous deep regression, ordinal regression networks for visual safety mapping, and an integrated pipeline for SLZ identification in dynamic environments.

1.6 Research Questions

To address the aforementioned challenges, this thesis investigates the following research questions:

1. **How can deep learning-based vision models be designed to accurately predict SLZs for UAVs in both urban and natural scenes?**
   This explores the potential of semantic segmentation and object detection techniques in generating dense safety maps.

2. **What is the impact of integrating both photometric and geometric cues such as color, depth, flatness, and inclination—on the accuracy and reliability of SLZ prediction?**
   This investigates whether multimodal input improves scene understanding for landing zone assessment.

3. **How can dynamic obstacles and UAV motion be incorporated into the SLZ detection process to ensure safe landings in real-time scenarios?**
   This question focuses on integrating object detection, tracking, and motion compensation through homography.

4. **What design strategies can ensure that the proposed SLZ detection framework remains computationally efficient and suitable for real-time**

deployment on resource-constrained UAV platforms?
This addresses the trade-off between model complexity and inference performance.

5. **How does the proposed approach perform across diverse environments such as construction zones, suburban roads, or natural terrain—when compared to existing state-of-the-art methods?**
This evaluates generalization and robustness of the proposed system in varying operational conditions.

1.7 Thesis Organization

The remainder of this document is organized as follows.

- **Chapter 2** reviews the state of the art in visual recognition and SLZ detection for UAVs. It covers object detection, semantic segmentation, relevant deep learning architectures, loss functions, and evaluation metrics, and the literature review.
- **Chapter 3** presents the deep regression-based method for continuous SLZ safety mapping from UAV imagery, detailing the network architecture, training strategy, and experimental evaluation on static scenarios.
- **Chapter 4** introduces the OR-SLZNet ordinal regression framework for multi-level safety assessment, describes the integration of photometric and geometric cues, and compares its performance against baseline classification and regression models.
- **Chapter 5** extends the framework to dynamic environments by integrating terrain segmentation, object detection, multi-object tracking, trajectory prediction, and homography-based motion compensation into a unified SLZ pipeline, with experiments on realistic UAV video sequences.
- **Chapter 6** summarizes the main findings of the thesis, discusses limitations, and outlines perspectives for future research on robust, certifiable SLZ assessment and its integration into broader autonomous navigation systems.

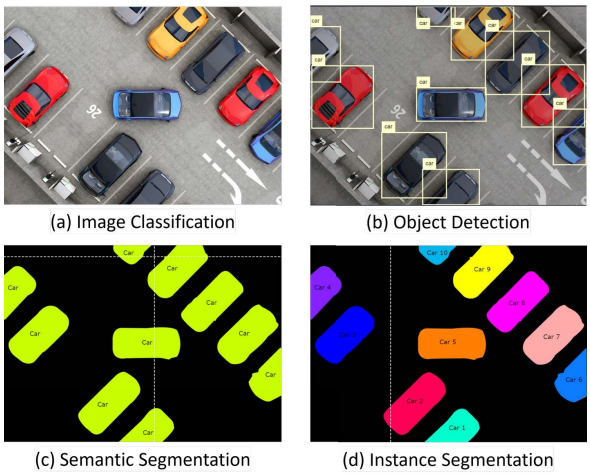
Chapter 2

State of the Art

2.1 Visual Recognition Problems in Computer Vision

Computer vision is a multidisciplinary field that enables computers to analyze digital images or videos at a high level, similar to how humans perceive visual information. It relies on techniques from computer science, mathematics, and artificial intelligence to interpret visual data, extract meaningful information, and support autonomous or assisted decision-making. Today, computer vision underpins a broad range of applications, including autonomous driving, medical imaging, augmented reality, surveillance, robotics, and industrial automation. It can recognize objects in photos, interpret complex scenes, and even anticipate future events based on visual cues. As hardware capabilities and machine learning algorithms continue to advance, the boundaries of what is feasible with computer vision are constantly being extended, creating new opportunities for innovation and discovery [215, 198].

Figure 2.1: Four Main Visual Recognition Tasks in Computer Vision



Within this field, four main visual recognition tasks serve as core building blocks for perception systems (Figure 2.1).

Image Classification. Image classification is the task of assigning a single label or category to an entire image (Figure 2.1(a)). It underpins numerous applications, including content-based image retrieval, industrial quality control, and computer-aided medical diagnosis. The advent of deep learning, and in particular convolutional neural networks (CNNs), has dramatically improved classification accuracy, enabling models to recognize a wide variety of objects and scenes with high reliability [198]. In many computer vision pipelines, image classification also serves as a foundational problem upon which more complex recognition tasks are built.

Object Detection. Object detection extends image classification by simultaneously recognizing and localizing multiple objects within an image using bounding boxes (Figure 2.1(b)). This capability is essential for tasks such as autonomous driving, UAV navigation, surveillance systems, and augmented reality, where knowing *what* is present is not sufficient without knowing *where* it is. Compared to pure classification, object detection must address additional challenges, including scale variation, occlusion, overlapping objects, and cluttered or dynamic backgrounds, which require robust and efficient algorithms [215].

Semantic Segmentation. Semantic segmentation provides a denser and more detailed understanding of the scene by assigning a semantic label to every pixel in the image

(Figure 2.1(c)). This pixel-level categorization is crucial for applications that require precise scene understanding, such as autonomous navigation, land-use mapping, medical image analysis, and image editing. However, semantic segmentation is computationally demanding and typically relies on large annotated datasets and high-capacity models, which may limit deployment on resource-constrained platforms [102].

Instance Segmentation. Instance segmentation goes one step further by distinguishing not only between semantic categories but also between individual object instances within the same category (Figure 2.1(d)). Each object is segmented with pixel-level precision, making this task well suited to applications such as robotic manipulation, fine-grained scene analysis, crowd understanding, and safety-critical decision-making where accurate object boundaries and separations are essential. Despite these advantages, instance segmentation remains highly challenging due to overlapping objects, complex shapes, cluttered scenes, and large intra-class variability [102].

2.2 Object Detection

Object detection is a core building block of modern computer vision systems, bridging the gap between image-level recognition and dense scene understanding. Instead of only predicting which categories are present, detectors output a set of localized object hypotheses, each with a category label and spatial support (typically a bounding box, and in some cases a coarse mask). This joint reasoning over category and location underlies many applications introduced earlier—such as autonomous driving, UAV navigation, surveillance, medical imaging, and human–computer interaction—where decisions depend on both what is present and where it is [224, 248, 65].

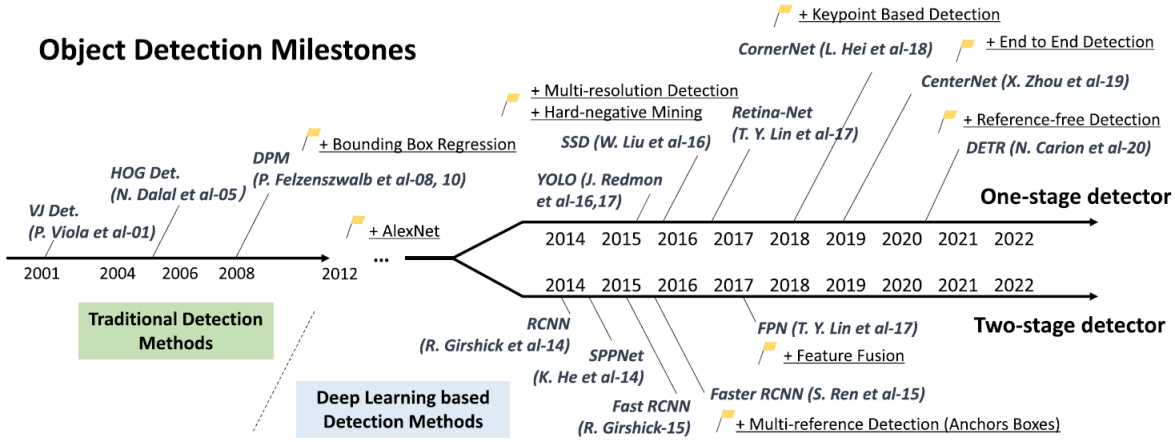
Modern detection systems combine several design choices that jointly determine accuracy, robustness, and efficiency [65]. One concerns the representation of objects: bounding boxes remain the dominant choice because of their simplicity and computational efficiency [248], whereas pixel-wise masks, as used in semantic and instance segmentation, provide finer localization for tasks that require precise shape information (e.g., in medical imaging or high-precision inspection). A second concerns the detection paradigm. Two-stage (region-based) detectors first generate candidate regions and then classify and refine them, typically achieving strong accuracy at higher computational cost [224, 121]. One-stage (region-free) detectors perform classification and localization in a single pass over dense sampling points or anchor grids, simplifying the pipeline and

enabling real-time inference on embedded or resource-constrained platforms [65, 224]. In practice, two-stage methods remain competitive when maximum accuracy is paramount, while one-stage methods dominate latency-sensitive scenarios.

Object detection supports a broad range of domains, from face and pedestrian detection in consumer devices and intelligent transportation, to text detection in natural scenes, remote sensing analysis, and medical imaging. For autonomous vehicles and mobile robots in particular, detection provides the foundational perception layer required to identify obstacles, traffic participants, and relevant infrastructure, supporting robust scene understanding and safe navigation in dynamic environments [248, 65]. Progress in this field has been accelerated by large-scale benchmarks such as PASCAL VOC, MS COCO, and specialized datasets for faces and pedestrians, which offer standardized evaluation protocols for fair comparison of algorithms [224, 248].

As illustrated in Figure 2.2, the evolution of object detection over the past two decades can be broadly divided into two periods [248]. The pre-deep learning era (before 2014) was dominated by hand-crafted features (e.g., HOG, SIFT) and classical classifiers (e.g., SVMs, boosting), as detailed in Appendix A. The deep learning-based era (after 2014) was initiated by region-based convolutional frameworks and subsequently advanced through end-to-end trainable two-stage and one-stage detectors. This transition led to substantial gains in accuracy, robustness, and scalability, establishing deep neural networks as the de facto standard for modern object detection.

Figure 2.2: Road map of object detection [248].



2.2.1 Deep Learning-Based Object Detection Methods

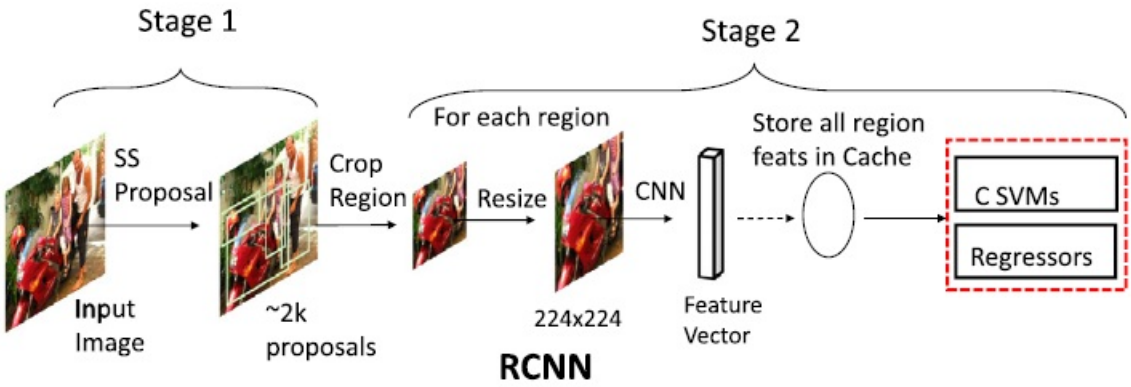
With the advent of deep learning, object detection underwent a fundamental transformation. Unlike traditional pipelines that relied on handcrafted features and independently designed stages, deep learning-based approaches learn hierarchical, task-specific feature representations directly from data. Convolutional Neural Networks (CNNs) enable end-to-end or near end-to-end training, integrating feature extraction, localization, and classification within unified architectures. As a result, deep learning-based detectors have achieved substantial improvements in accuracy, robustness, and scalability compared to traditional methods, and now constitute the dominant paradigm in modern object detection [224, 248].

Two-stage detectors

A central class of such approaches is formed by **two-stage detectors**, which operate in two sequential steps: region proposal generation followed by region-wise classification and bounding-box refinement. Although generally more computationally demanding than one-stage designs, two-stage detectors typically deliver higher localization precision and state-of-the-art performance on challenging benchmarks.

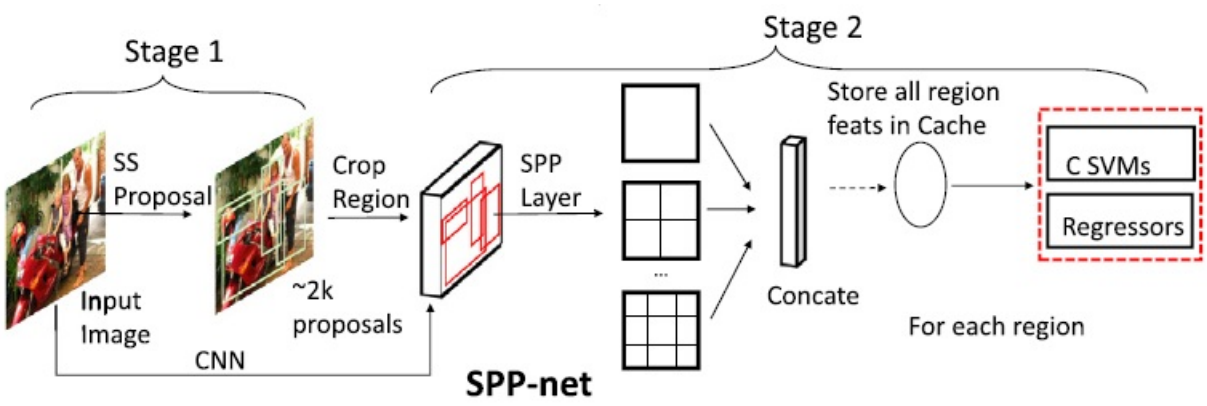
R-CNN (Regions with CNN Features) [68] was one of the pioneering deep learning-based object detectors and marked the transition from handcrafted to learned features. Its design combines (i) bottom-up region proposals obtained using Selective Search; (ii) high-capacity CNNs to extract deep feature representations from each proposed region; and (iii) class-specific linear SVMs and bounding-box regressors for final classification and localization. The model leverages supervised pretraining on large-scale image classification datasets followed by task-specific fine-tuning, which was crucial when detection datasets were relatively small. R-CNN significantly improved detection accuracy over traditional methods, but introduced several notable drawbacks [121]. First, training is complex and multi-stage: the CNN, SVMs, and bounding-box regressors are trained separately, making the pipeline cumbersome and difficult to optimize jointly. Second, feature extraction is computationally and storage intensive: deep features must be computed and cached for thousands of proposals per image, particularly expensive with networks such as VGG-16. Third, inference is slow, as each region proposal is passed independently through the CNN at test time, preventing real-time deployment. These limitations motivated more efficient architectures (Figure 2.3).

Figure 2.3: R-CNN framework for two-stage object detection [224].



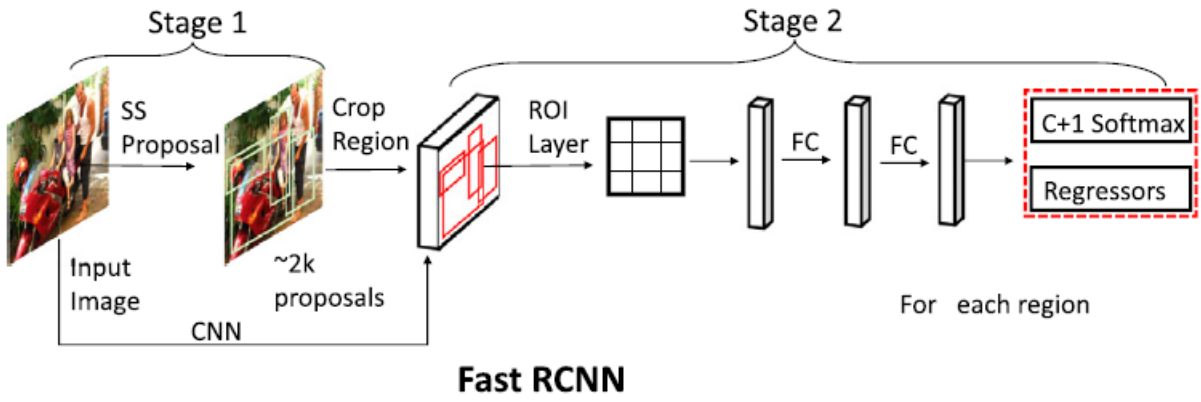
SPPNet (Spatial Pyramid Pooling Network) [76] was proposed to accelerate R-CNN and improve robustness to scale and aspect ratio. Instead of forwarding each proposal independently through the CNN, SPPNet computes a single convolutional feature map for the entire image. A Spatial Pyramid Pooling (SPP) layer is then applied on top of the last convolutional layer to generate fixed-length feature vectors for proposals of arbitrary size. Concretely, the SPP layer overlays multiple spatial grids (e.g., 1×1, 2×2, 3×3, 6×6) on the proposal's feature map and performs pooling (typically max pooling) in each bin; the pooled outputs are concatenated into a descriptor of length $K\sum_{\ell} n_{\ell}^2$ for a feature map with K channels and grid sizes $n_{\ell}\times n_{\ell}$. This preserves coarse spatial layout and multi-scale context while eliminating the need to warp proposals to a fixed size, substantially reducing redundant computation and improving detection speed [224]. However, SPPNet still relies on a multi-stage training scheme, and in its original form does not backpropagate gradients through all convolutional layers before the SPP layer, effectively freezing early layers and limiting full end-to-end optimization [121]. Its framework within the two-stage pipeline is shown in Figure 2.4.

Figure 2.4: SPPNet framework within a two-stage detector [224].



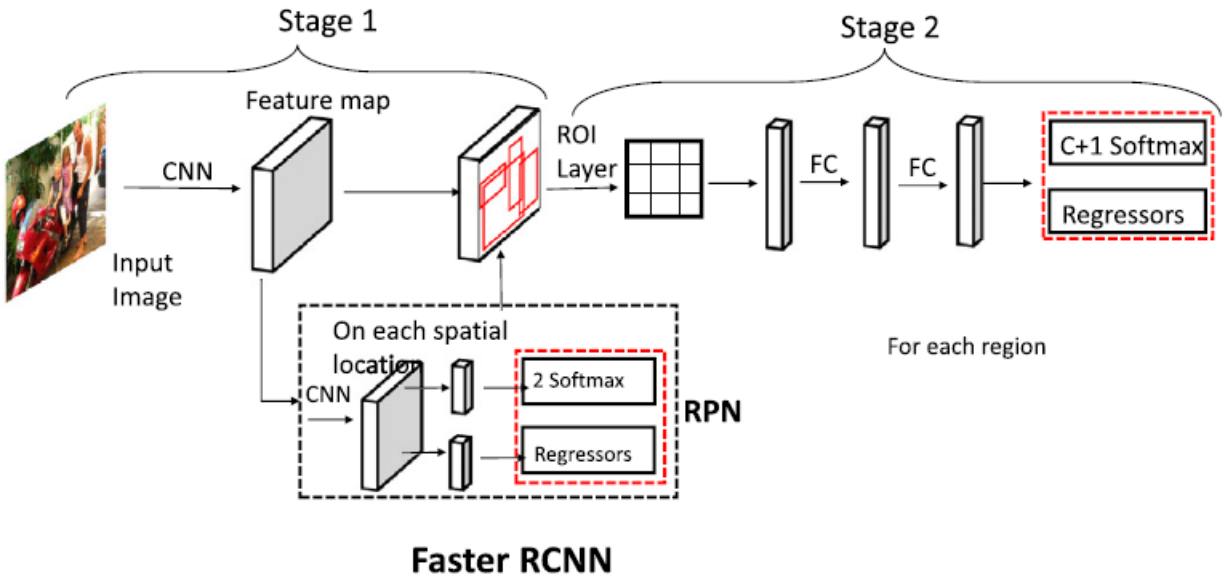
Fast R-CNN [67] addresses key drawbacks of both R-CNN and SPPNet by enabling efficient, single-stage training while sharing computation across all region proposals. The image is first processed by a CNN to generate a shared convolutional feature map. Candidate regions (e.g., from Selective Search) are then projected onto this feature map, and a Region of Interest (RoI) Pooling layer converts each projected region into a fixed-size feature map by partitioning it into a grid and applying max pooling within each cell. The pooled RoI features are passed through fully connected layers and then branch into two output heads: a softmax classifier over object categories (including background) and a class-specific bounding-box regression module. This design allows backpropagation through both the RoI Pooling and convolutional layers, enabling end-to-end training (excluding proposal generation) and dramatically reducing redundant computation, since the convolutional backbone is executed only once per image [224, 121]. Fast R-CNN achieves higher detection accuracy, significantly faster training, and much faster inference than R-CNN and SPPNet (Figure 2.5).

Figure 2.5: Fast R-CNN framework for two-stage object detection [224].



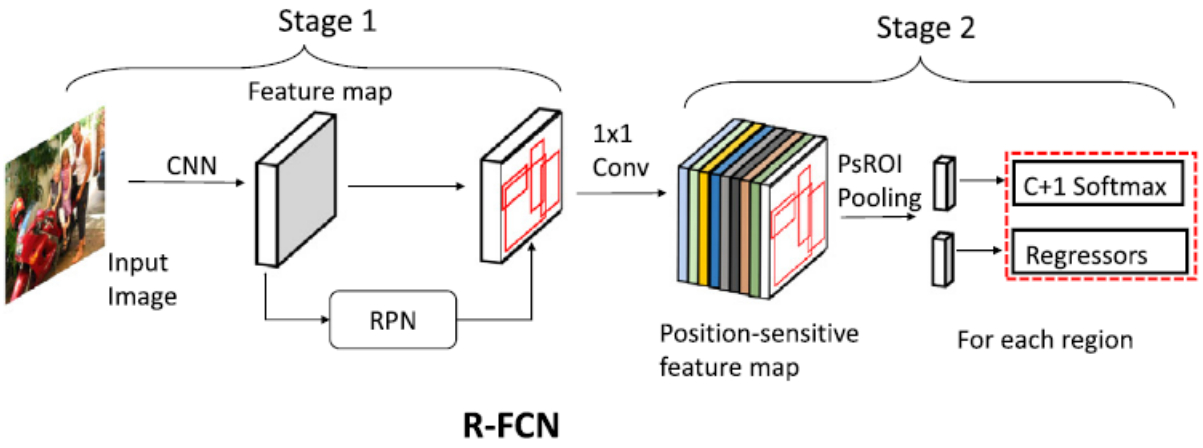
Faster R-CNN [178] represents a major step toward fully integrated, learnable detection pipelines by introducing the Region Proposal Network (RPN). Instead of relying on external proposal mechanisms, the RPN is a fully convolutional module that operates on the shared backbone feature map and, at each spatial location, predicts objectness scores and bounding-box offsets for a set of predefined anchors with different scales and aspect ratios. High-scoring proposals are selected, refined, and then passed through RoI Pooling (or RoI Align in later variants) and subsequent layers for classification and final localization. By sharing convolutional features between the RPN and the detection head, Faster R-CNN eliminates the computational bottleneck of external proposals and further improves accuracy [224]. Nonetheless, each RoI is still processed by region-wise fully connected layers, so computational cost remains relatively high when a large number of proposals are used, which can hinder strict real-time performance. The overall architecture is summarized in Figure 2.6.

Figure 2.6: Faster R-CNN framework with integrated Region Proposal Network [224].



R-FCN (Region-based Fully Convolutional Network) [50] is proposed to further reduce the computational overhead associated with per-RoI fully connected layers in Faster R-CNN. R-FCN is designed as a nearly fully convolutional architecture in which almost all computations are shared across the entire image. The key innovation is the introduction of position-sensitive score maps, where each class is associated with a bank of maps encoding responses for specific relative positions (e.g., top-left, center, bottom-right) within an object. For each RoI, Position-Sensitive RoI (PSROI) Pooling partitions the RoI into a grid and pools from the corresponding position-sensitive channels, aggregating spatially aligned responses into class scores. This mechanism preserves the necessary translation variance for detection (unlike standard fully convolutional classification networks, which are largely translation invariant) while avoiding heavy region-wise fully connected sub-networks [224, 121]. As a result, R-FCN achieves accuracy competitive with Faster R-CNN but with significantly improved efficiency, making it well-suited for large-scale detection scenarios (Figure 2.7).

Figure 2.7: R-FCN framework with position-sensitive score maps [50].



One-Stage (Unified) Detectors

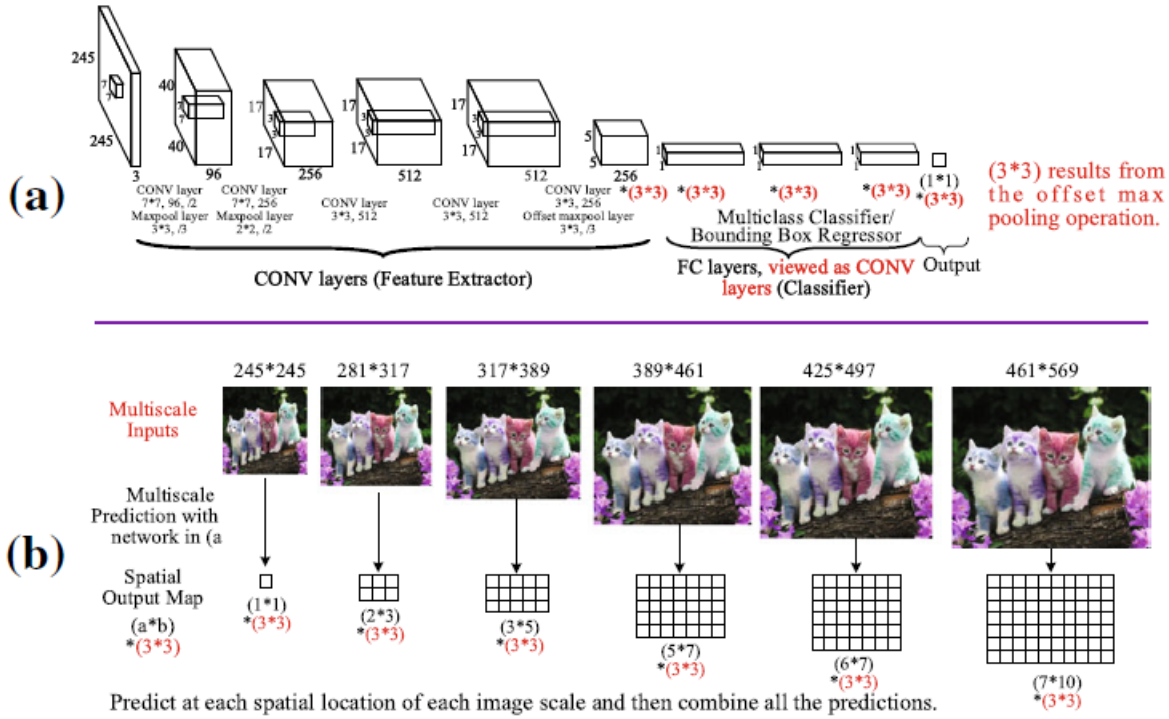
One-stage detectors bypass the explicit region proposal stage and directly perform classification and bounding-box (or mask) regression in a single network pass over dense sampling points or anchor boxes. By unifying detection into a single-stage formulation, they substantially reduce computational overhead and are well-suited for real-time applications, especially on embedded or resource-constrained platforms.

OverFeat [190] is an early single-stage detector that won ILSVRC-2013. The original classifier is cast into a fully convolutional network by interpreting the fully connected layers as 1×1 convolutions. This lets a single forward pass over an image of arbitrary size produce dense class-score maps. As shown in Figure 2.8, panel (b), the image is evaluated at multiple enlarged scales; for each scale the FC-as-CONV classifier outputs an $a \times b$ grid of predictions, where a and b are the grid's height and width (i.e., the number of sliding-window positions along rows and columns).

To densify sampling without recomputing features, OverFeat applies offset max pooling after the last CONV layer. Pooling is repeated at every stride offset, producing 3×3 interleaved outputs per location—the red "3×3" in Figure 2.8(a). Consequently, each scale yields $a \times b \times 9$ candidate views that vote for object presence, improving robustness while remaining efficient.

Bounding boxes are obtained by a regression head applied only at locations whose class score exceeds a threshold. Reusing the shared CONV features, the regressor predicts four continuous values (x, y, w, h) relative to that location/scale's receptive field; only the regressor's FC layers are evaluated, so no feature recomputation is needed. Finally, predictions from all locations, offsets, and scales are combined by a greedy merge procedure—functionally similar to non-maximum suppression but with score-weighted box averaging—to produce the final detections. Despite its efficiency, OverFeat's accuracy lagged behind R-CNN methods [121], reflecting early training challenges for FCNs and the absence of later proposal/anchor mechanisms [224].

Figure 2.8: Illustration of the OverFeat detection framework [121]



You Only Look Once (YOLO) is a widely adopted one-stage object detection family that has significantly advanced modern computer vision. Unlike traditional two-stage frameworks such as Faster R-CNN, which decouple region proposal and classification, YOLO formulates detection as a single regression problem and processes the entire image in one forward pass [175]. This unified design enables real-time inference with high

throughput and low latency, making it well suited to time-critical applications such as autonomous driving, security surveillance, and robotics.

Since its initial release by Redmon et al. in 2015, YOLO has undergone continuous refinement across multiple versions, improving accuracy, speed, and generalization. Recent variants are increasingly adapted to diverse imaging modalities including multispectral, thermal, and hyperspectral data, broadening their use in remote sensing, environmental monitoring, and defense systems. Figure 2.9 illustrates the main milestones in the evolution of the YOLO family from v1 to v11 [65].

Figure 2.9: Milestones in YOLO evolution (v1 to v11) [65].

YOLO Version	Year	Backbone	Key Contributions	Framework
YOLOv1 [175]	2015	Custom CNN	Unified single-stage detector	Darknet
YOLOv2/9000 [176]	2016	Darknet-19	Anchors, multi-scale, 9000 classes	Darknet
YOLOv3 [177]	2018	Darknet-53	FPN-style multi-scale, residuals	Darknet
YOLOv4 [21]	2020	CSPDarknet-53	SPP, PAN, rich augmentations	Darknet
YOLOv5 [87]	2020	CSP-based	SPPF, PAN/FPN, strong augmentations, variants	PyTorch
YOLOv6 [109]	2022	EfficientRep	RepVGG, TAL, deploy-focused design	PyTorch
YOLOv7 [217]	2022	E-ELAN	Re-parameterized convs, efficient scaling	PyTorch
YOLOv8 [212]	2023	C2f-based	Decoupled head, multi-task support	PyTorch
YOLOv9 [218]	2024	GELAN	PGI, improved mAP/FLOPs trade-off	PyTorch
YOLOv10 [216]	2024	CSP variants	NMS-free training/inference	PyTorch
YOLOv11 [98]	2025	Enhanced CSP/C2PSA	Attention, multi-task, edge-ready design	PyTorch

Table 2.1: Overview of YOLO model evolution.

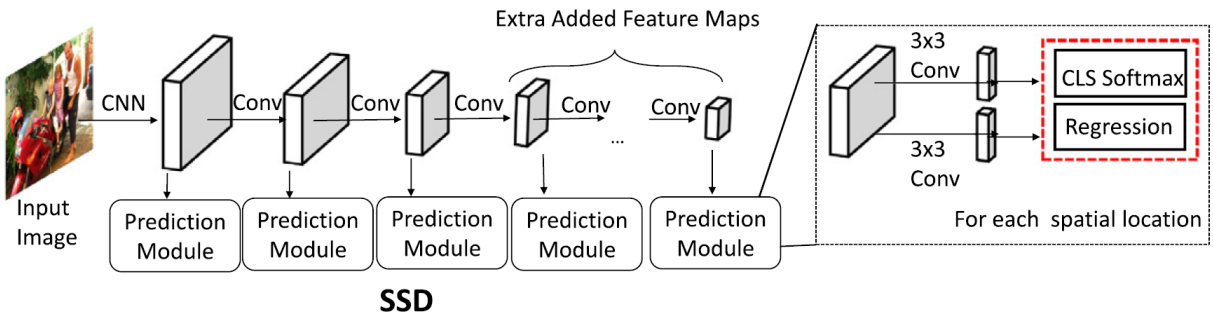
The evolution from YOLOv1 to YOLOv11, detailed in Appendix B and summarized in Table 2.1, highlights steady gains in speed, accuracy, and efficiency. Successive versions introduce innovations such as multi-scale feature extraction, more expressive backbones, improved feature aggregation, attention mechanisms, and deployment-aware re-parameterization. As a result, the YOLO family has become a reference one-stage detector for real-time applications, including autonomous navigation, multispectral analysis, industrial automation, and healthcare diagnostics. The most recent releases, such as YOLOv10 and YOLOv11, further optimize computational cost while extending the framework to tasks like instance segmentation, pose estimation, and generic image classification, reinforcing its role as a versatile backbone for modern computer vision. For the object detection module in Chapter 5, we employ YOLOv11 as our primary detector [98]. YOLOv11 offers a favorable trade-off between accuracy, inference speed, and

deployment efficiency, making it well suited to the real-time constraints of our UAV-based SLZ detection framework.

SSD (Single Shot MultiBox Detector) [123]: In 2016, Liu et al. introduced the Single Shot MultiBox Detector (SSD) to improve object detection speed while maintaining high accuracy. Unlike YOLO [175], SSD utilizes a set of anchor boxes with multiple scales and aspect ratios within each grid cell, discretizing the output space of bounding boxes more effectively. A key innovation in SSD is its multi-scale feature map approach, where object detection occurs at different layers of the network. Shallow feature maps capture local information for precise object positioning, while deeper feature maps focus on classification by leveraging segmentation information, shown in Figure 2.10. This hierarchical detection strategy allows SSD to perform well across varying object sizes.

SSD combines YOLO's regression-based detection with Faster R-CNN's anchor mechanism, leading to significant improvements in speed and accuracy. Built on the VGG-16 backbone with additional convolutional layers, SSD processes 300×300 resolution images efficiently. It achieves a mean average precision (mAP) of 74.3% at 59 FPS, surpassing YOLOv1's [175] 63.4% mAP at 45 FPS and Faster R-CNN's [178] 7 FPS performance despite its high accuracy. However, SSD has limitations, such as the need for manually set prior box parameters and suboptimal performance on small object detection compared to Faster R-CNN [178]. Nevertheless, SSD remains a robust one-stage detector, striking a balance between detection speed and accuracy, making it well-suited for real-time applications [224, 9].

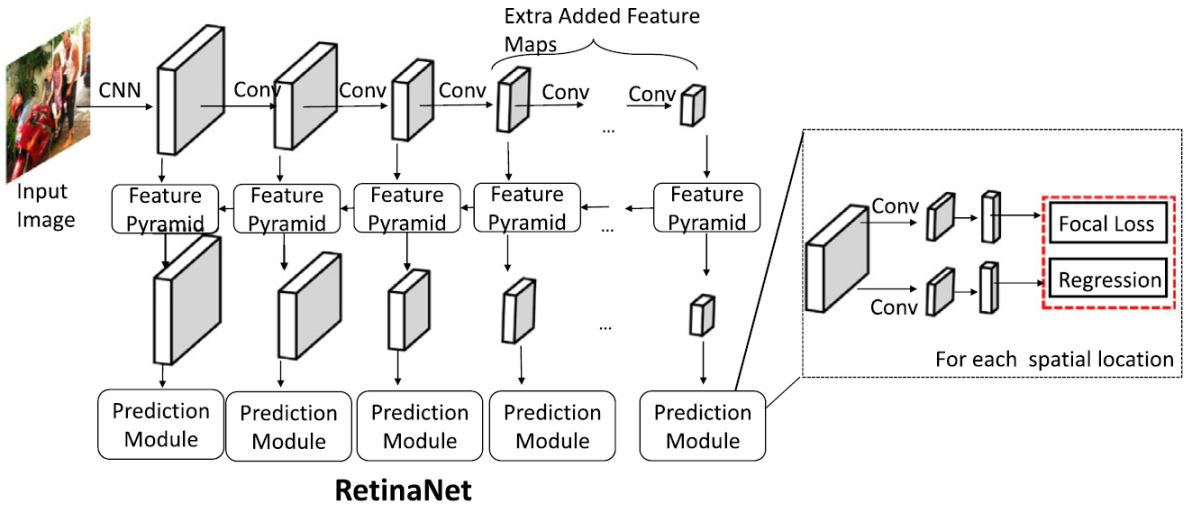
Figure 2.10: SSD Architecture [224]



RetinaNet [117]: proposed by Lin et al. in 2017, addressed the long-standing issue of class imbalance in one-stage object detectors, which had traditionally lagged behind two-stage detectors in accuracy. The key innovation in RetinaNet was the introduction of Focal Loss, a modified cross-entropy loss function that reduces the impact of easy negative samples while emphasizing hard, misclassified examples. This approach significantly improved training stability and enabled RetinaNet to achieve accuracy comparable to Faster R-CNN while maintaining the efficiency of one-stage detection.

As shown in Figure 2.11, RetinaNet utilizes a ResNet backbone for feature extraction and integrates a Feature Pyramid Network (FPN) to enhance multi-scale object detection. FPN improves the model's ability to detect objects of various sizes by leveraging feature representations at different levels of abstraction. The model achieved a COCO mAP@0.5 of 59.1%, demonstrating its effectiveness in real-world detection scenarios. While Focal Loss alleviates the imbalance between foreground and background classes, it remains susceptible to noise, requiring careful sample labeling. Despite this, RetinaNet established itself as a highly efficient and accurate one-stage object detector, bridging the gap between traditional single-stage and two-stage methods [224, 9].

Figure 2.11: RetinaNet Architecture [224]

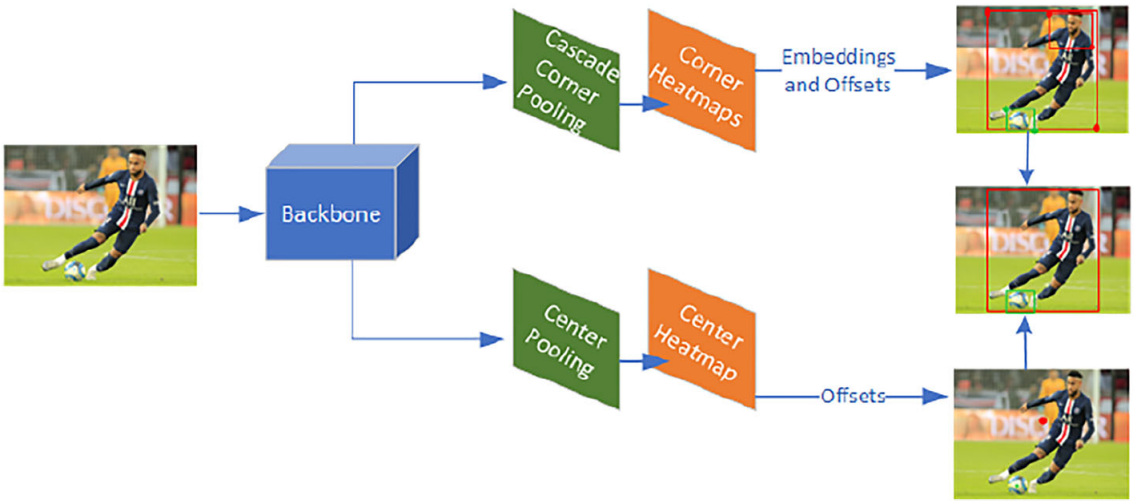


CenterNet [60]: released in April 2019, a new anchor-free detection method proposed on the basis of CornerNet [106]. CenterNet is a keypoint-based object detection framework that eliminates the need for complex post-processing steps such as non-maximum

suppression (NMS) and group-based keypoint assignment. Unlike previous approaches like CornerNet [106], which detect objects using pairs of corner keypoints, CenterNet represents each object by a single center point and directly regresses its attributes, including size, orientation, location, and pose. This simplified detection paradigm allows CenterNet to achieve efficient, fully end-to-end object detection.

A key improvement over CornerNet is the use of Cascade Corner Pooling, which enhances the ability to capture internal object features by sequentially extracting maximum boundary values, the architecture of CenterNet is shown in Figure 2.12. CenterNet demonstrated superior performance on the MS COCO dataset, achieving a COCO mAP@0.5 of 61.1%, surpassing many existing one-stage and even some two-stage detectors. Additionally, CenterNet's flexible architecture extends beyond object detection to tasks such as 3D object detection, human pose estimation, depth estimation, and optical flow learning. However, a notable limitation of CenterNet arises when two objects have overlapping center points after downsampling, leading to detection errors where the model identifies them as a single object. Despite this, CenterNet remains a powerful and efficient detection framework, setting a new standard for keypoint-based object detection[9].

Figure 2.12: CenterNet Architecture [9]

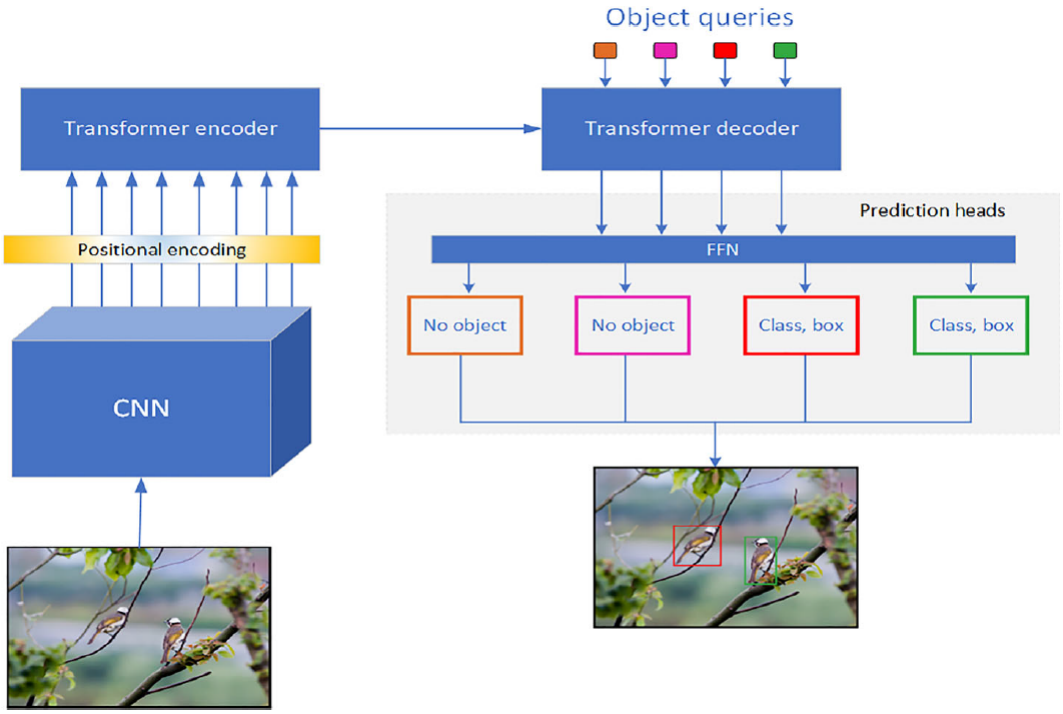


Transformer-Based Detectors

Recently, transformer architectures have been adopted for object detection, demonstrating improved global context modeling and reducing reliance on anchor-based methods. **DEtection TRansformer (DETR) [33]** introduced by Carion et al. in 2020, was a groundbreaking application of Transformer architecture in object detection. Unlike traditional CNN-based methods, DETR formulates object detection as a set prediction problem, eliminating the need for anchor boxes and non-maximum suppression (NMS). The model (Figure 2.13) consists of three main components: a CNN backbone for feature extraction, a Transformer encoder-decoder for global feature learning, and a feedforward network for predicting bounding boxes and class labels. The self-attention mechanism in DETR enables it to model complex relationships between objects across an image, making it particularly effective for detecting overlapping and cluttered objects.

Despite its simplicity and competitive performance, DETR has notable limitations, including slow convergence and suboptimal detection of small objects. To address these challenges, Zhu et al. [246] later introduced Deformable DETR, which improved convergence speed and enhanced small object detection capabilities. DETR achieves state-of-the-art performance on the MS COCO dataset, with a COCO mAP@0.5 of 71.9%, making it a promising approach for end-to-end object detection. While it is computationally intensive compared to traditional detectors like Faster R-CNN, DETR's fully attention-based architecture simplifies the object detection pipeline, paving the way for future Transformer-based models in computer vision [208, 9, 248].

Figure 2.13: The process of DETR and its structure [9]

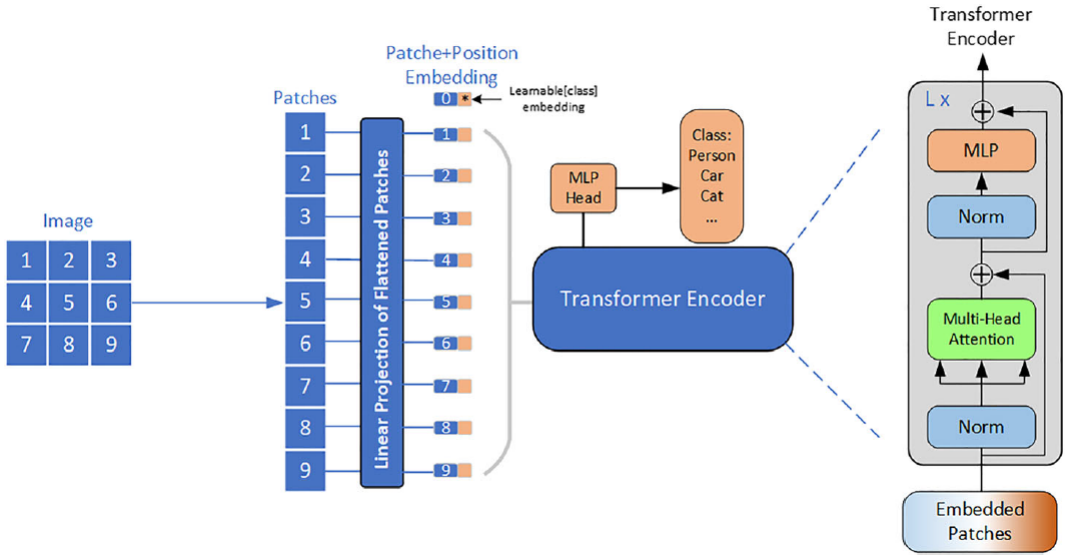


Vision Transformer (ViT) [58]: applies the standard Transformer architecture to image classification tasks with minimal modifications. As shown in Figure 2.14, unlike traditional convolutional neural networks (CNNs), ViT processes images as sequences of fixed-size patches rather than pixel grids. An input image is divided into $N = HW/P^2$ patches, where H and W represent the image dimensions, and P^2 is the resolution of each patch. These patches are flattened and embedded before being passed through the Transformer encoder, similar to word embeddings in natural language processing (NLP). A classification token is appended to the patch sequence, allowing ViT to perform classification based on global context.

One challenge in ViT is the loss of spatial information during the transformation of patches into linear embeddings. To address this, positional embeddings are added to retain location information. While this design leverages the scalability and efficiency of NLP Transformers, ViT requires large-scale datasets for effective training and demands significant computational resources. Additionally, ViT struggles with encoding precise positional information, limiting its performance on certain vision tasks. Despite these

challenges, ViT has demonstrated strong performance on classification benchmarks and has influenced the development of Transformer-based models in computer vision [9].

Figure 2.14: Vision Transformer structure [9]

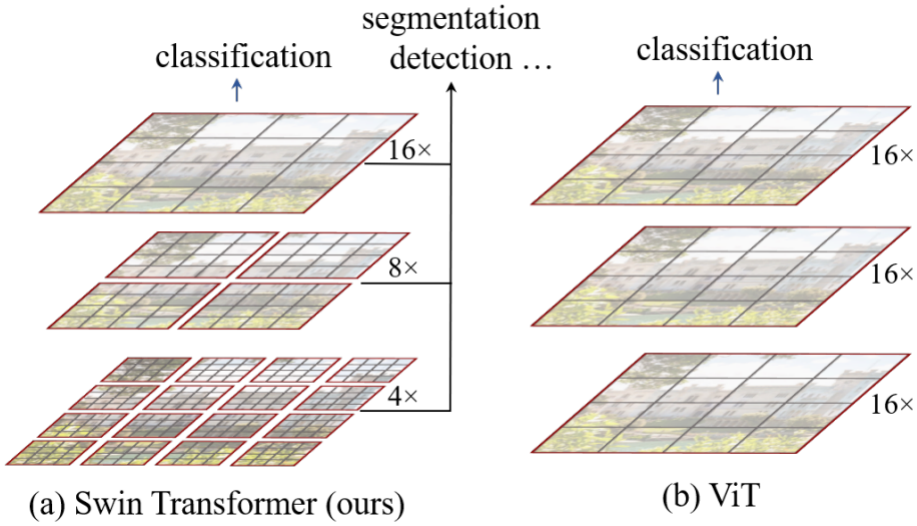


Swin Transformer (Swin ViT) [124]: Swin Transformer was introduced to address key limitations of the Vision Transformer (ViT) and improve its applicability to various computer vision tasks. Unlike ViT, which processes small resized images, Swin Transformer directly takes original images as input, preserving spatial information. A key innovation in Swin Transformer is its hierarchical network structure, similar to convolutional neural networks (CNNs), which progressively expands the receptive field while reducing resolution and increasing the number of channels. This hierarchical design allows Swin Transformer to perform classification, object detection, and instance segmentation effectively, achieving state-of-the-art (SOTA) results.

Main difference between Swin Transformer and ViT is depicted in Figure 2.15, Swin Transformer also improves upon ViT by adapting local attention mechanisms instead of global self-attention, making it more computationally efficient while maintaining strong feature representation. Its structure is inspired by Feature Pyramid Networks (FPN) and U-Net, enabling it to detect and segment objects more effectively. Due to its efficiency and accuracy, Swin Transformer has become a universal backbone architecture for Transformer-based vision tasks. However, it has notable challenges, including high

computational costs, significant GPU memory requirements, and the need for precise fine-tuning in certain applications. Despite these limitations, Swin Transformer successfully bridges the gap between CNNs and Transformers, making Transformer-based architectures more practical for dense prediction tasks in computer vision [9].

Figure 2.15: Main difference between Swin Transformer and ViT [124]



2.2.2 Object Detection Metrics

Evaluating object detection models requires assessing both what is detected (classification) and where it is detected (localization). Unlike semantic segmentation, where outputs are dense pixel-wise predictions, object detection produces a variable number of bounding boxes with class labels and confidence scores. Metrics must therefore (i) handle varying numbers of predictions per image, (ii) penalize duplicate detections, (iii) incorporate geometric overlap between predicted and ground-truth boxes, and (iv) support comparison across classes and object scales. In addition to accuracy, modern works often report computational metrics such as inference speed and model size, especially for real-time and embedded applications [224, 248, 65].

Intersection over Union (IoU). Intersection over Union is the fundamental measure of bounding box overlap. For a predicted box B_p and ground-truth box B_g, it is defined

as

$$\mathrm{IoU}(B_p, B_g) = \frac{|B_p \cap B_g|}{|B_p \cup B_g|}.$$

A predicted box is typically counted as a true positive (TP) if its IoU with an unmatched ground-truth box is greater than or equal to a specified threshold (e.g., 0.5) and the predicted class is correct; otherwise, it is a false positive (FP). By varying the IoU threshold, one can impose stricter or looser localization requirements [61].

Precision, Recall, and F1. Given TP, FP, and false negatives (FN), precision and recall are defined as

$$\text{Precision} = \frac{TP}{TP + FP}, \qquad \text{Recall} = \frac{TP}{TP + FN}.$$

Precision reflects the reliability of detections (low FP), while recall reflects how completely objects are found (low FN). The F1 score,

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}},$$

provides a single summary at a chosen confidence threshold; however, official benchmarks usually rely on metrics derived from the full precision–recall curve rather than a single operating point.

Average Precision (AP) and mean Average Precision (mAP). Average Precision (AP) summarizes the precision–recall (PR) relationship for a given class. Predictions are sorted by confidence, then precision and recall are computed as the confidence threshold is swept from high to low. AP is defined as the area under the PR curve [61]:

$$\mathrm{AP}_c = \int_0^1 \text{Precision}_c(r)\, dr,$$

computed numerically in practice. The mean Average Precision over C classes is

$$\text{mAP} = \frac{1}{C} \sum_{c=1}^{C} \mathrm{AP}_c.$$

The PASCAL VOC protocol reports AP at a fixed IoU threshold of 0.5, denoted AP_{50} or mAP@0.5, and this convention remains widely used [61].

COCO-Style AP Metrics. The MS COCO benchmark introduced a more stringent and informative evaluation scheme that has become standard for modern detectors [118]. The primary COCO metric is

$$\mathrm{AP}_{50:95},$$

defined as the mean of AP over ten IoU thresholds from 0.5 to 0.95 in steps of 0.05 (commonly referred to as mAP@0.5 : 0.95). In addition, COCO reports:

- $\mathbf{AP}_{50}$: AP at IoU = 0.5 (looser localization).
- $\mathbf{AP}_{75}$: AP at IoU = 0.75 (stricter localization).
- $\mathbf{AP}_S$, $\mathbf{AP}_M$, $\mathbf{AP}_L$: AP for small, medium, and large objects.

These metrics jointly evaluate classification, localization accuracy, and robustness across object scales. Many contemporary implementations and toolkits (including YOLO-based frameworks) adopt this protocol and routinely report mAP@0.5 and mAP@0.5 : 0.95 alongside precision and recall, ensuring consistency with VOC/COCO standards.

Average Recall (AR). Average Recall focuses on a detector's ability to cover ground-truth objects across varying thresholds and constraints on the number of predictions. COCO reports AR under different maximum detection limits (e.g., AR@1, AR@10, AR@100) and for different object sizes [118]. AR is particularly informative for evaluating proposal methods and high-recall settings, complementing AP, which emphasizes the precision–recall trade-off.

Duplicate Detections and Matching Protocol. To ensure consistent evaluation, each ground-truth object may be matched to at most one predicted box. Predictions are considered in descending order of confidence; the first prediction that attains IoU above the threshold with an unmatched ground-truth box is marked as TP, while subsequent overlapping predictions are FPs. This matching protocol penalizes duplicate detections of the same object. In practice, Non-Maximum Suppression (NMS) or its variants are applied prior to evaluation to reduce redundancy. All metrics described above are computed under this standardized matching scheme [61, 118].

Efficiency Metrics. For deployment-oriented scenarios (e.g., UAV platforms, embedded systems, real-time surveillance), accuracy metrics are complemented with:

- inference speed (frames per second, latency per image),
- model size (number of parameters, storage),

– computational cost (e.g., FLOPs).

While these are not accuracy measures, reporting them together with AP/mAP is essential for judging the practicality of object detection models in real-world applications [224, 248].

2.3 Image Segmentation

After introducing the fundamental concepts of visual recognition in Section 2.1, we now turn our attention to segmentation techniques. This section contrasts three widely used paradigms: semantic segmentation, instance segmentation, and panoptic segmentation. We then outline the core methodologies underlying many advanced approaches, starting from pre-deep learning (classical) methods, summarized in Appendix C, and moving on to deep learning-based techniques in Section 2.3.1.

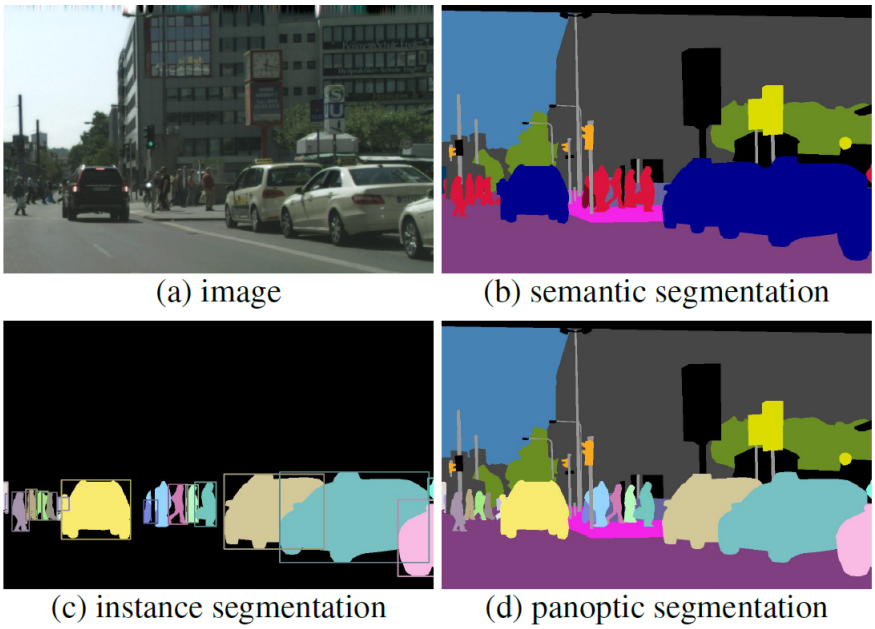
Throughout this thesis, segmentation plays a central role in all three contributions to the SLZ problem. The model architectures proposed in Chapters 4 and 3 are both derived from the U-Net encoder–decoder 2.3.2 family of segmentation networks. Moreover, a key component of the integrated framework presented in Chapter 5 relies on semantic segmentation to extract static objects in the scene and to delineate potential SLZs.

Semantic Segmentation vs. Instance Segmentation: Semantic segmentation assigns a class label to every pixel in an image, grouping all pixels that belong to the same semantic category (e.g., all cars, all roads, all buildings). In contrast, instance segmentation goes one step further by distinguishing between different objects of the same class and assigning a unique identifier to each instance. In other words, semantic segmentation (Figure 2.16(b)) answers the question of what is present (e.g., all pixels corresponding to car), whereas instance segmentation (Figure 2.16(c)) answers both what and which one (e.g., pixels for Car 1, Car 2, and so on).

Panoptic Segmentation: Panoptic segmentation unifies semantic and instance segmentation within a single, coherent representation. Each pixel is assigned a semantic class label and, when applicable, an instance identifier, thereby capturing both category-level and object-level information. Practically, this means that the pixel encoding includes two components: one for semantic classification and one for instance indexing. This joint formulation provides a more complete understanding of the scene than using semantic and instance segmentation separately. As illustrated in Figure 2.16(d), panop-

tic segmentation not only differentiates individual cars but also labels non-object regions such as the sky, road, or vegetation with appropriate semantic classes [102].

Figure 2.16: Different Image Segmentation Approaches [102]



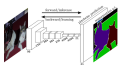
2.3.1 Deep Learning Image Segmentation Methods

Deep learning has fundamentally transformed image segmentation by replacing hand-crafted features and heuristic pipelines with end-to-end trainable architectures that learn rich, hierarchical representations from data. Modern models can jointly capture low-level appearance, high-level semantics, and long-range context, and are adaptable across natural, medical, aerial, and industrial imaging domains. Below, we summarize the main families of deep learning-based segmentation methods.

Fully Convolutional Networks (FCNs). Fully Convolutional Networks (FCNs) [127] were among the first deep learning architectures proposed specifically for semantic segmentation. They convert classification CNNs into dense predictors by replacing fully connected layers with convolutional layers, enabling input images of arbitrary size and producing correspondingly sized segmentation maps. FCNs exploit skip connections to fuse coarse, high-level semantic features from deeper layers with fine, appearance-rich features from shallower layers, improving boundary localization and segment consis-

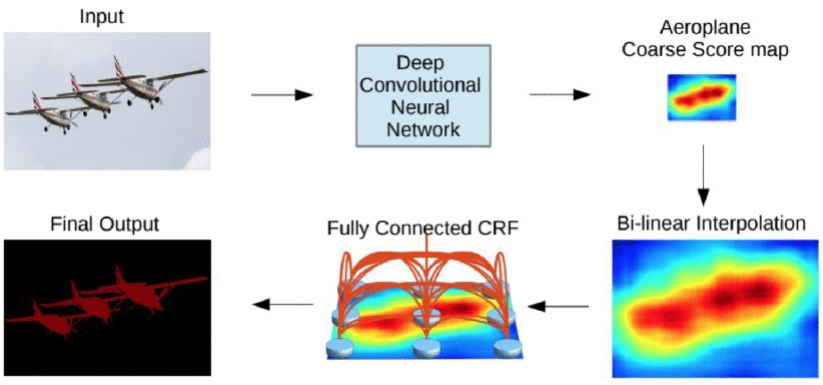
tency (Figure 2.17). They achieved state-of-the-art performance on early benchmarks and established the paradigm of fully convolutional dense prediction. However, FCNs have limitations, including relatively coarse outputs without strong multi-scale context modeling, challenges in modeling global dependencies, and difficulties when extended directly to 3D volumetric data [147]. Subsequent architectures build upon and refine these ideas.

Figure 2.17: Fully Convolutional Network for image segmentation [147].



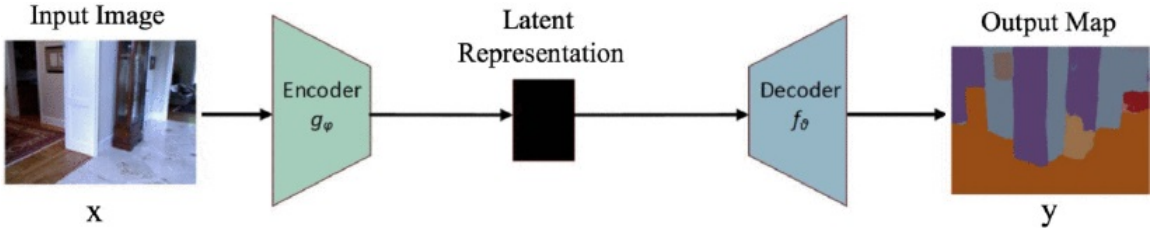
Convolutional Models with Graphical Models. To compensate for the limited spatial and contextual modeling of early FCNs, convolutional networks have been combined with probabilistic graphical models such as Conditional Random Fields (CRFs) and Markov Random Fields (MRFs). In these hybrid CNN+CRF/MRF frameworks, the CNN typically produces unary potentials (per-pixel class scores), while the graphical model encodes pairwise or higher-order relationships between neighboring pixels or regions, encouraging label smoothness and alignment with image edges [147]. A CNN+CRF model illustrated in Figure 2.18. of Dense CRFs have been particularly effective for sharpening object boundaries, and end-to-end or iterative refinement schemes have integrated CRF-like operations into deep networks. These approaches improve boundary accuracy and region consistency, though they can increase complexity and computational cost.

Figure 2.18: Illustration of a CNN+CRF model [147].



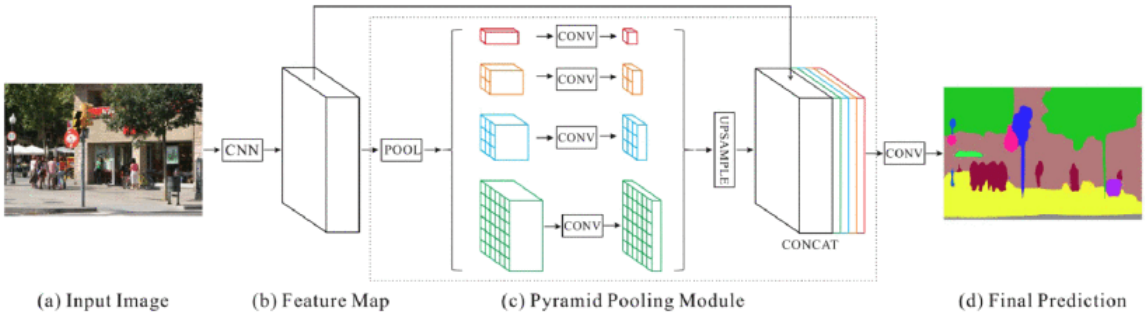
Encoder–Decoder Based Models. Encoder–decoder architectures form a dominant family of segmentation models, an example of this architectures showed in Figure 2.19. The encoder progressively reduces spatial resolution while extracting high-level semantic features; the decoder then upsamples these features to recover dense predictions. Early variants employed deconvolution or transposed convolutions, while SegNet [14] introduced decoders that reuse encoder pooling indices for more accurate and memory-efficient upsampling. HRNet [233] maintains high-resolution representations throughout the network, repeatedly fusing multi-scale branches to preserve fine details. In medical imaging, U-Net [182] has become a standard: it uses a symmetric contracting and expanding path with skip connections that directly concatenate encoder feature maps to decoder layers, enabling precise localization with strong contextual encoding. Extensions such as V-Net [28] generalize this paradigm to 3D data and address class imbalance. Despite potential information loss during aggressive downsampling, encoder–decoder models offer a flexible and powerful framework for a wide range of 2D and 3D segmentation tasks [147].

Figure 2.19: Example of an encoder–decoder architecture [147].



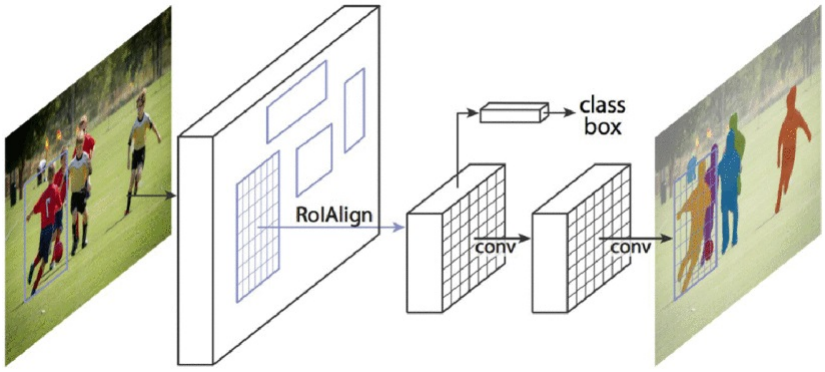
Multiscale and Pyramid Network-Based Models. To capture objects and context at different spatial scales, multiscale and pyramid-based architectures have been proposed. As shown in Figure 2.20, Feature Pyramid Networks (FPN) [116] exploit the inherent hierarchical structure of CNNs by combining low-resolution, semantically strong features with high-resolution, spatially precise features through top-down and lateral connections, yielding rich multi-scale feature maps for detection and segmentation. Pyramid Scene Parsing Network (PSPNet) [243] aggregates context via spatial pyramid pooling over multiple receptive field sizes and fuses this information with local features to better disambiguate similar regions [147]. Such designs significantly improve robustness to scale variation and complex scene layouts.

Figure 2.20: The PSPNet architecture [147].



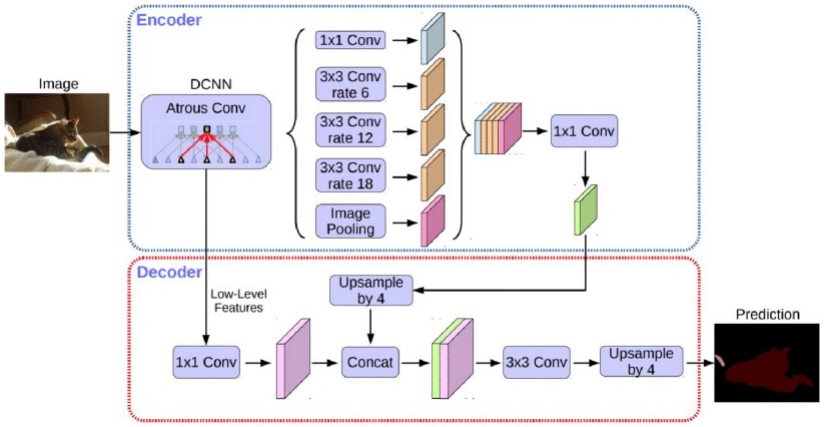
R-CNN Based Models for Instance Segmentation. Instance segmentation extends semantic segmentation by distinguishing individual object instances. R-CNN-based methods, particularly Mask R-CNN [75], are highly influential in this area. Mask R-CNN builds on Faster R-CNN [179] by adding a parallel branch that predicts a binary mask for each detected instance, using RoIAlign for precise spatial alignment. PANet [122] enhances information flow with bottom-up path augmentation and adaptive feature pooling, improving mask quality. Other approaches, such as MaskLab [41] and TensorMask [45], explore richer feature combinations and dense 4D mask representations within the R-CNN framework (Figure 2.21). These models provide strong, flexible baselines for instance-level segmentation, though at relatively high computational cost [147].

Figure 2.21: Mask R-CNN architecture for instance segmentation [147].



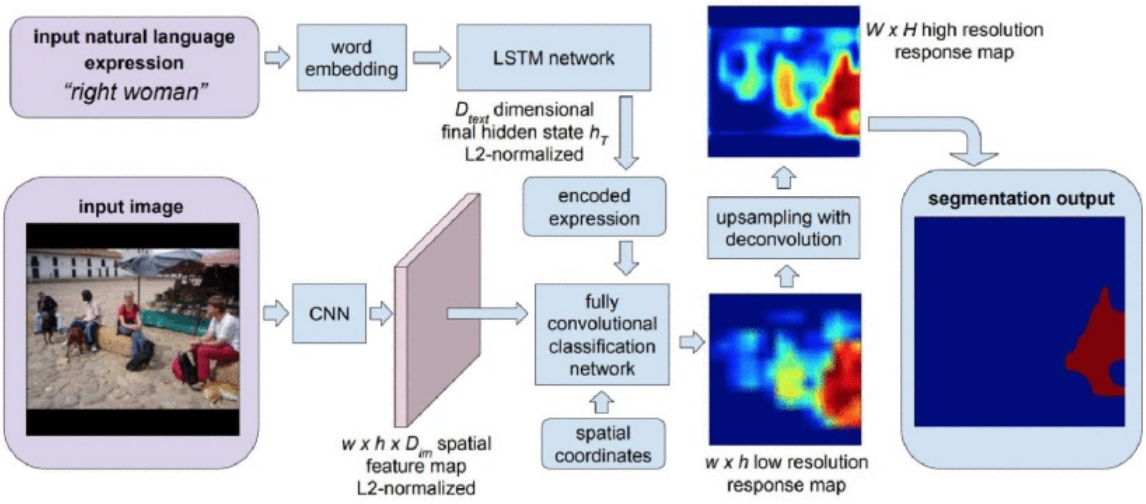
Dilated Convolutional Models. Dilated (atrous) convolutions enlarge the receptive field of convolutional filters without increasing the number of parameters or reducing resolution, making them well-suited for dense prediction. The DeepLab family illustrates their effectiveness. DeepLabv2 [42] combines dilated convolutions with Atrous Spatial Pyramid Pooling (ASPP) to capture multi-scale context and employs dense CRFs for boundary refinement. DeepLabv3+ [44] extends this with an encoder–decoder structure and depthwise separable convolutions for efficient, high-quality segmentation (Figure 2.22). More broadly, dilated convolutions are used to balance context and detail in many modern architectures, contributing to strong performance in both accuracy-focused and near real-time settings [147].

Figure 2.22: DeepLabv3+ architecture [147].



RNN-Based Models. Recurrent Neural Networks (RNNs) have also been explored for segmentation to explicitly model long-range dependencies and structural relationships. For example, ReSeg [214] employs ReNet layers that sweep across the image horizontally and vertically, aggregating contextual information, while graph-LSTM architectures [114] operate on superpixels to capture higher-level structure. Other models combine CNN encoders with LSTMs (Figure 2.23) driven by natural language expressions to segment objects referred to in text [80]. Although RNN-based methods highlight the benefits of sequential and structured modeling, their limited parallelism and higher computational cost have made them less prevalent than CNN- and Transformer-based designs [147].

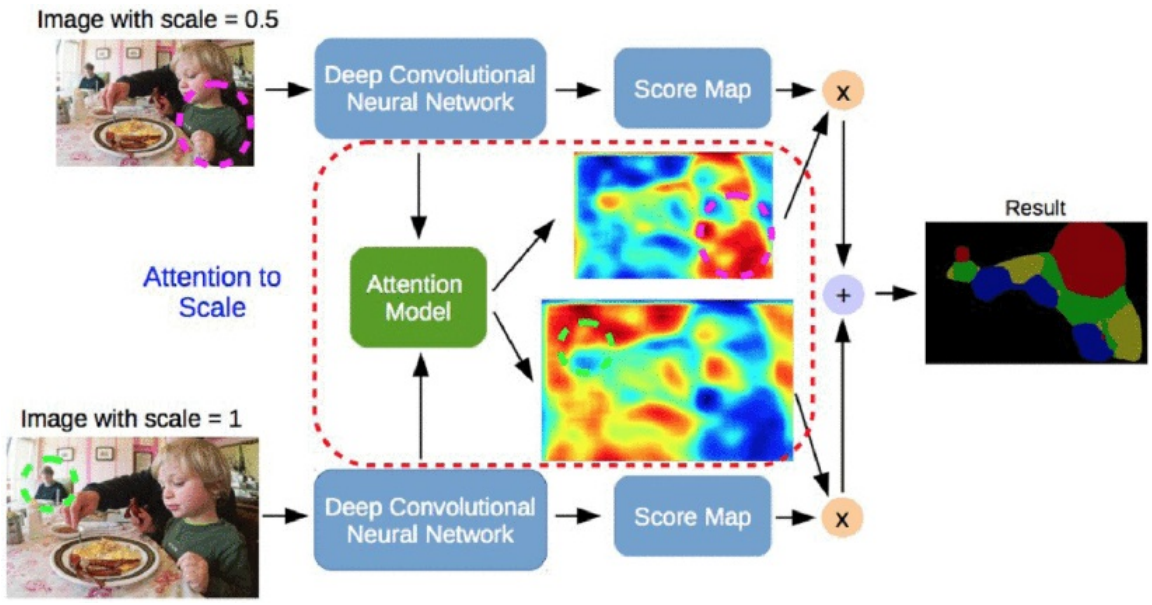
Figure 2.23: Semantic segmentation from natural language expressions using a CNN+LSTM model [147].



Attention-Based Models. Attention mechanisms enhance segmentation networks by allowing them to weight spatial locations, channels, and scales according to task relevance. Scale-aware attention modules can selectively emphasize features corresponding to important objects at different resolutions, outperforming simple average or max pooling [43]. Reverse Attention Networks [81] further improve performance by explicitly modeling "where not to attend," refining boundaries and suppressing false positives. Attention has since become a core component in many state-of-the-art architectures, enabling better focus on salient structures and more effective integration of global con-

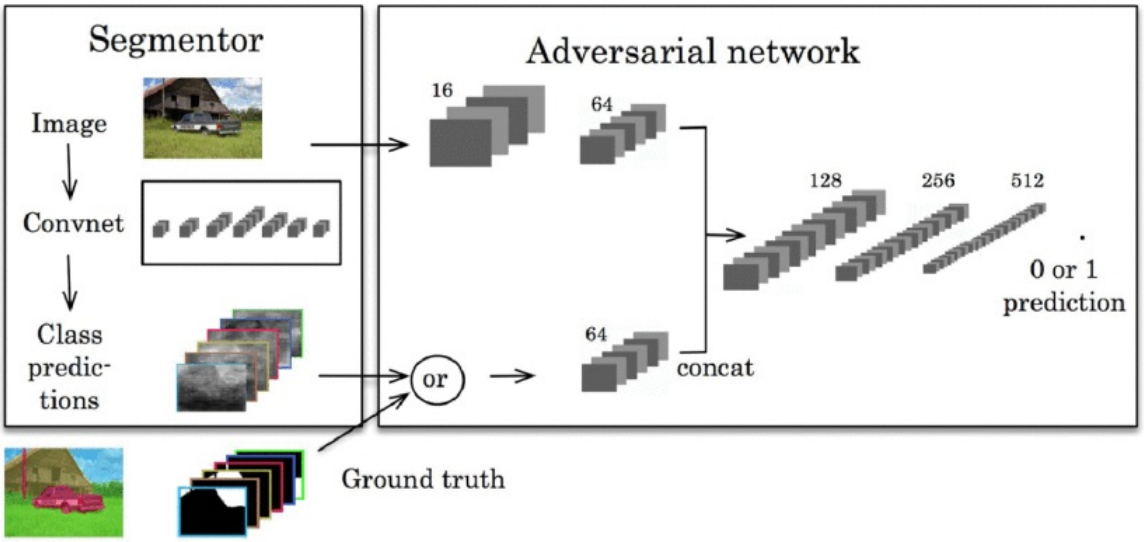
text [147]. An example of an attention-based semantic segmentation model is given in Figure 2.24.

Figure 2.24: Example of an attention-based semantic segmentation model [147].



Generative Adversarial Networks (GANs) for Segmentation. Generative Adversarial Networks have been employed to improve segmentation quality by encouraging outputs that resemble realistic label maps. In adversarial training frameworks Figure 2.25, a segmentation network (generator) predicts masks, while a discriminator network distinguishes between ground-truth and predicted segmentations [133]. This setup imposes higher-level structural regularization that complements pixel-wise losses. In medical imaging and other domains, multiscale adversarial losses have been shown to sharpen boundaries and reduce artifacts [227]. Although more complex to train, GAN-based approaches demonstrate the potential of adversarial learning to refine and stabilize segmentation results [147].

Figure 2.25: GAN-based semantic segmentation framework [147].



In summary, deep learning-based image segmentation has evolved from early FCNs to a diverse ecosystem of encoder–decoder networks, pyramid and dilated-convolution models, instance-segmentation frameworks, attention and Transformer-based designs, and adversarial training schemes. These architectures leverage end-to-end learning, multi-scale context, and increasingly powerful global reasoning mechanisms to achieve substantially higher accuracy and robustness than classical methods across a wide range of visual domains.

2.3.2 U-Net

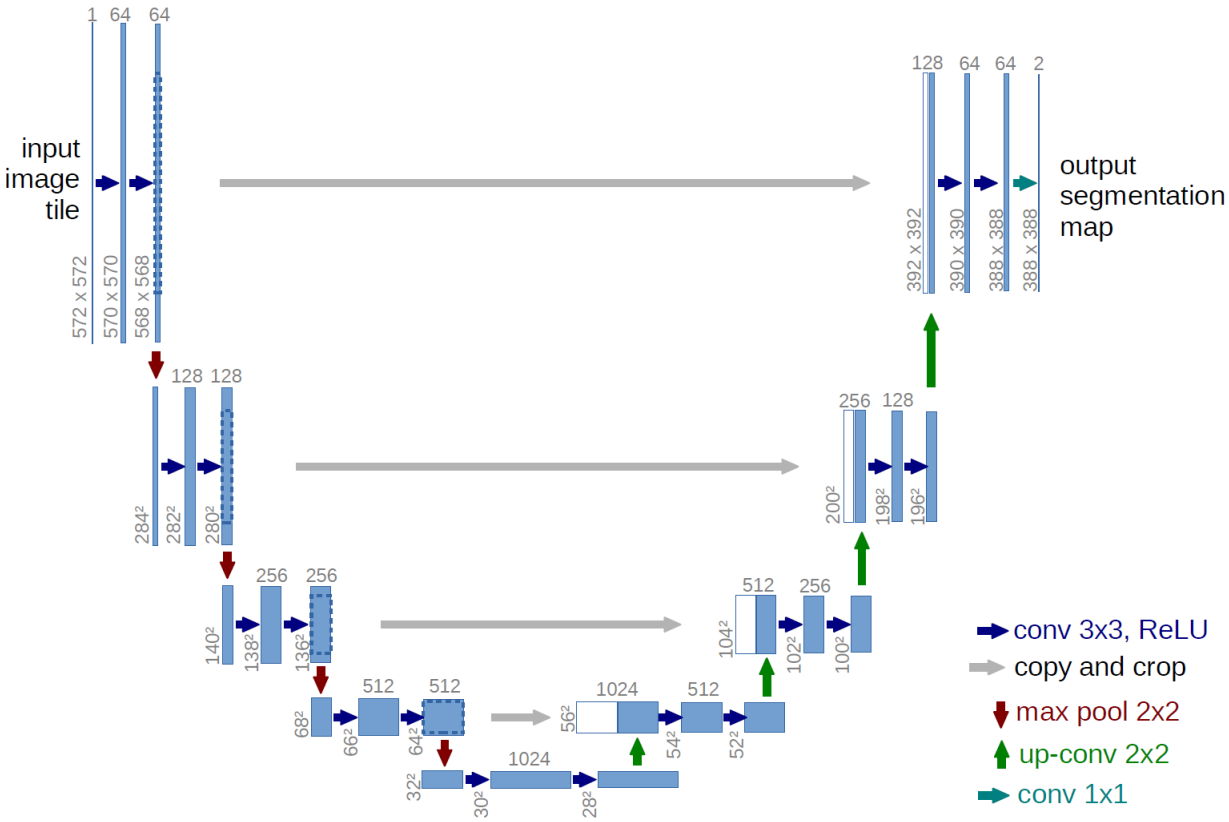
U-Net [182] is a widely used deep learning architecture for semantic segmentation, originally proposed for biomedical microscopy images and subsequently adopted in many other domains, including aerial and UAV imagery. It can be viewed as a modified Fully Convolutional Network (FCN) specifically designed to address challenges such as limited annotated data, the need for precise localization, and the preservation of fine structural details. Its characteristic U-shaped, symmetric architecture with extensive skip connections enables the network to effectively combine contextual information with high-resolution spatial cues [182, 147].

The U-Net architecture, illustrated in Figure 2.26, is composed of two main paths. The contracting path (encoder) is responsible for capturing context. It consists of re-

peated applications of 3×3 convolutions followed by rectified linear unit (ReLU) activations, and 2×2 max pooling operations for downsampling. As spatial resolution decreases, the number of feature channels increases, allowing the network to learn increasingly abstract and semantically rich representations. Opposite to this, the expanding path (decoder) focuses on precise localization and reconstruction of the segmentation map. It uses up-convolutions (transpose convolutions or other upsampling operations) to progressively restore spatial resolution. At each decoding stage, feature maps from the corresponding encoder layer are concatenated with the upsampled features via skip connections. These skip connections are crucial: they reintroduce fine-grained details lost during downsampling and provide the decoder with both low-level boundary information and high-level contextual cues. A final 1×1 convolution layer maps the resulting features to class scores, producing a dense pixel-wise segmentation.

This combination of deep context encoding and fine-detail recovery gives U-Net several practical advantages. First, it is highly effective in settings with limited annotated data: the architecture's strong inductive bias and skip connections allow robust learning from relatively small training sets, which is particularly valuable in medical imaging and specialized UAV applications [182, 147]. Second, U-Net is computationally efficient enough to support real-time or near real-time inference on modern hardware, making it suitable for time-critical tasks such as on-site disaster assessment, vegetation and land-cover mapping, or traffic monitoring from UAV imagery [219]. Third, the precise localization enabled by its skip connections leads to accurate, sharp object boundaries and reliable pixel-level predictions, which are essential whenever small structures, thin objects, or detailed regions must be segmented. These properties have established U-Net and its numerous variants as foundational architectures for modern image and volume segmentation across a broad range of domains.

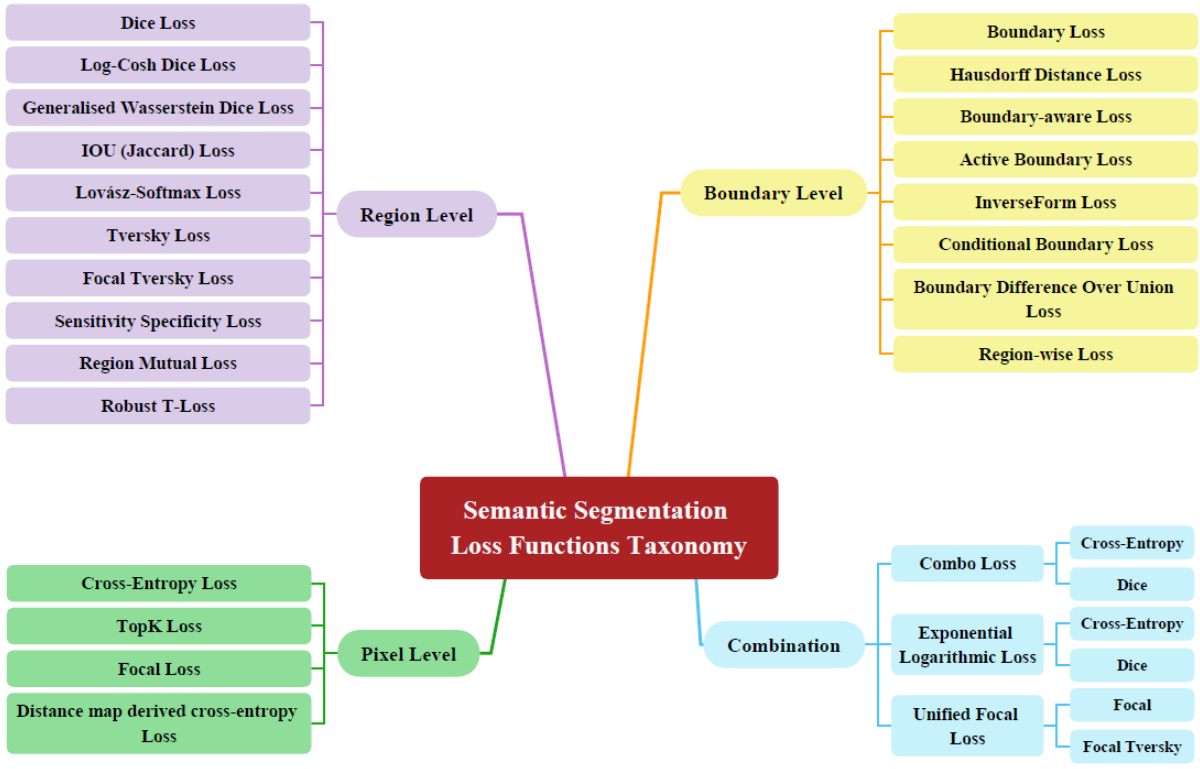
Figure 2.26: The U-Net architecture [147].



2.3.3 Loss Functions

Careful design of loss functions is central to training accurate and robust segmentation networks. Different losses emphasize different aspects of prediction quality—per-pixel correctness, region-level overlap, boundary precision, or robustness to severe class imbalance—and thus directly affect both convergence during training and performance in real-world deployments. As summarized in Figure 2.27, semantic segmentation losses are often grouped into four categories: (i) region-level, (ii) boundary-level, (iii) pixel-level, and (iv) combination losses [12]. In this work, we focus on several widely used and conceptually important pixel- and region-level losses that are closely related to our study.

Figure 2.27: Semantic segmentation loss function taxonomy [12].



Cross-Entropy and Weighted Cross-Entropy. Cross-entropy (CE) loss measures the discrepancy between the predicted class probabilities and the ground-truth labels at each pixel [103]. Let $t_n^c \in \{0,1\}$ denote the one-hot ground truth for pixel n and class c, and $\hat{y}_n^c \in [0,1]$ the corresponding softmax probability. The standard multi-class cross-entropy is

$$L_{CE} = -\sum_{n=1}^{N}\sum_{c=1}^{C} t_n^c \log \hat{y}_n^c.$$

CE encourages the network to assign high probability to the correct class at each pixel and is widely used due to its probabilistic interpretation and stable optimization properties. However, when classes are highly imbalanced (e.g., small target structures against a dominant background), CE tends to be biased toward frequent classes. Weighted

Cross-Entropy (WCE) addresses this by assigning a weight w_c to each class:

$$L_{WCE} = -\sum_{n=1}^{N}\sum_{c=1}^{C} w_c\, t_n^c \log \hat{y}_n^c,$$

where larger w_c values are used for underrepresented classes. When all $w_c = 1$, WCE reduces to standard CE. In practice, CE/WCE often serve as the baseline pixel-level loss in segmentation networks [12, 219].

Dice Loss. The Dice coefficient measures the overlap between a predicted segmentation Y and the ground-truth mask T:

$$\text{Dice} = \frac{2|Y \cap T|}{|Y| + |T|}.$$

It can be interpreted as the harmonic mean of precision and recall and is especially useful when the positive class occupies only a small portion of the image, as is common in medical imaging and fine-structure segmentation. Dice loss is defined as $L_{\text{Dice}} = 1 - \text{Dice}$, with a differentiable extension for soft predictions. For multi-class segmentation, a standard formulation is [146, 12, 219]

$$L_{\text{Dice}} = 1 - \frac{1}{C}\sum_{c=1}^{C} \frac{2\sum_{n=1}^{N} t_n^c y_n^c}{\sum_{n=1}^{N} t_n^c + \sum_{n=1}^{N} y_n^c},$$

where y_n^c denotes the predicted (soft) score for class c and pixel n. Dice loss directly optimizes region-level overlap, making it well suited for highly imbalanced segmentation tasks.

IoU (Jaccard) Loss. Intersection over Union (IoU), or the Jaccard index,

$$\text{IoU} = \frac{|Y \cap T|}{|Y \cup T|},$$

is another widely used set-similarity measure. Like Dice, it quantifies overlap between prediction and ground truth but penalizes discrepancies differently, often more harshly when the overlap is small. IoU loss is typically defined as $L_{\text{IoU}} = 1 - \text{IoU}$, with a soft

multi-class extension [171, 12]:

$$L_{\text{IoU}} = 1 - \frac{1}{C}\sum_{c=1}^{C}\frac{\sum_{n=1}^{N} t_n^c y_n^c}{\sum_{n=1}^{N} t_n^c + \sum_{n=1}^{N} y_n^c - \sum_{n=1}^{N} t_n^c y_n^c}.$$

Because IoU-based metrics are often used as evaluation criteria, IoU loss provides a training objective that is closely aligned with the final performance measure.

Tversky Loss. The Tversky index generalizes Dice and IoU by allowing asymmetric weighting of false positives and false negatives, which is particularly important in applications where missing a target (FN) is more severe than over-segmentation (FP), or vice versa. A soft Tversky index for class c is given by [186]

$$\text{TI}_c = \frac{\sum_{n=1}^{N} t_n^c y_n^c}{\sum_{n=1}^{N} t_n^c y_n^c + \alpha \sum_{n=1}^{N} t_n^c (1 - y_n^c) + \beta \sum_{n=1}^{N} (1 - t_n^c) y_n^c},$$

where α and β control the penalties for false negatives and false positives, respectively. Setting $\alpha = \beta = 0.5$ recovers the Dice coefficient; $\alpha = \beta = 1$ is related to IoU. The Tversky loss is defined as

$$L_T = 1 - \frac{1}{C}\sum_{c=1}^{C} \text{TI}_c,$$

providing a flexible objective for heavily imbalanced segmentation tasks [12, 219].

Combo Loss. Combo loss [204] explicitly combines the strengths of region-based and pixel-wise losses to better handle class imbalance and stabilize training. It is defined as

$$L_{\text{combo}} = \alpha L_{WCE} + (1 - \alpha) L_{\text{Dice}},$$

where $\alpha \in [0, 1]$ controls the relative contribution of Weighted Cross-Entropy and Dice loss. The WCE component encourages correct pixel-wise classification, especially for minority classes through class weights, while the Dice component emphasizes good overlap for the segmented structures. By tuning α and the WCE weights, Combo loss can be adapted to different datasets and clinical or operational requirements [12].

2.3.4 Metrics for Segmentation Models

A comprehensive evaluation of segmentation models should consider not only accuracy but also inference speed, memory footprint, robustness, and visual plausibility of the

predicted masks. In practice, however, most works emphasize quantitative accuracy metrics. Below we summarize several commonly used measures for assessing segmentation quality [147]. While these metrics enable fair comparisons across methods, they should be complemented with qualitative inspection, since many applications ultimately depend on human interpretation of the results.

Pixel Accuracy (PA). Pixel Accuracy measures the proportion of correctly classified pixels over all pixels. Let p_{ij} denote the number of pixels of true class i predicted as class j for $K+1$ classes (including background). Then

$$\text{PA} = \frac{\sum_{i=0}^{K} p_{ii}}{\sum_{i=0}^{K} \sum_{j=0}^{K} p_{ij}}.$$

Although PA is intuitive, it can be misleading when class distributions are highly imbalanced, as large background regions may dominate the score even if foreground segmentation is poor.

Mean Pixel Accuracy (MPA). Mean Pixel Accuracy addresses class imbalance by computing the accuracy for each class separately and averaging:

$$\text{MPA} = \frac{1}{K+1} \sum_{i=0}^{K} \frac{p_{ii}}{\sum_{j=0}^{K} p_{ij}}.$$

This metric assigns equal importance to each class, including small or rare categories, and thus provides a more balanced assessment of model performance [147].

Intersection over Union (IoU). Intersection over Union (IoU), or the Jaccard index, is one of the primary metrics used in semantic segmentation challenges. For a given class, IoU is defined as

$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|},$$

where A and B are the predicted and ground-truth regions, respectively. Mean IoU (mIoU) is obtained by averaging class-wise IoUs. IoU directly measures the quality of overlap and penalizes both over- and under-segmentation, making it a robust and widely adopted metric [147].

Dice Score. The Dice score is closely related to IoU and is especially popular in medical image segmentation. For binary segmentation with foreground as the positive class, the Dice coefficient is equivalent to the F1 score. It emphasizes correct overlap

and is sensitive to errors in small structures, which makes it particularly useful when the target regions occupy only a small fraction of the image [147, 219].

Taken together, PA, MPA, IoU, and Dice provide complementary perspectives on segmentation performance. High-quality segmentation models should achieve strong quantitative scores across these measures while also producing visually coherent and interpretable results that meet the requirements of their target applications.

2.4 Literature review

Several surveys have been proposed to present the theory, methods, and challenges related to vision-based UAV navigation and SLZ detection.Yuncheng Lu et al. [132] explored vision-based methods for UAV navigation, emphasizing localization, obstacle avoidance, and path planning. Loureiro et al. [129] presented existing methods using multiple sensors (LiDAR, Vision) that address the problem of emergency landing site detection. Kakaletsis et al. [89] presented vision-based techniques for UAV safe navigation, particularly for safe landing. They also proposed a typical vision-based pipeline for carrying out safe landing. Alama et al. [191] presented a taxonomy for vision-based UAV landing methods, which is built using several criteria related to whether the landing site is indoor or outdoor, static or dynamic, or known or unknown. For known landing sites, Xin et al. [226] presented the challenges for designing and detecting markers for vision-based autonomous landing. Given the achievements brought by using deep learning techniques for vision recognition, we look at related works to our method through three categories: classical techniques, deep learning techniques, and CNN-based ordinal regression.

2.4.1 Classical techniques for SLZ detection

These methods include 3D reconstruction and or 2D image analysis for detecting SLZs. Vision-based 3D reconstruction generally uses stereo, tomography, and structure from motion, which can provide terrain flatness and orientation. Bosh et al. [23] used homography estimation to detect planar surfaces for safe landing. However, the method works well only when a large planar surface exists in the scene. Chatzikalymnios et al. [35, 36] used stereo and information from the inertial measurement unit (IMU) for 3D terrain reconstruction, on which terrain flatness, inclination, and steepness are estimated to

identify potential SLZs. Likewise, Liu et al. [125] proposed a 3D reconstruction method using structure from motion, which is used for autonomous safe landing. Detecting SLZs has also been investigated using methods combining machine learning and 2D image processing techniques. For example, Gabor filters [92] and the histogram of oriented gradients (HOG) [72] have been used to extract useful representations to train SVM for separating safe from unsafe regions.

2.4.2 Deep Learning techniques for SLZ detection

Semantic segmentation has been widely applied to identify SLZs for UAVs by classifying image pixels into thematic categories such as vegetation, pavement, or obstacles. Early works [101, 168, 56] utilized lightweight models to efficiently segment static terrain while others, such as [1], introduce continuous safety mapping through deep regression. Datta et al. [52] enhanced segmentation to better differentiate roads, trees, and water bodies. Guerin et al. [71] proposed an emergency landing framework using the Multi-Scale-Dilation (MSDnet) network on the UAVid dataset [136], integrating regional damage grading and Bayesian neural networks for runtime safety prediction. Other works include SafeUAV-Net [139] and PatchmatchNet-A [125], which leverage depth segmentation and dense reconstruction, respectively. Models such as KDP-Net [241], Wu et al.'s DeepLabv3+ with ShuffleNetv2 backbone [223], and CNN-Transformer hybrids [131] were introduced to address real-time constraints and dynamic scenes. Morales-Navarro et al. [152] combined superpixel segmentation with depth-based DNN classification to improve 3D SLZ detection in urban aerial imagery. Additionally, transfer learning was explored in [59] to adapt pretrained models for SLZ scene recognition.

Monocular and stereo depth estimation plays a critical role in analyzing terrain slope and roughness for safe UAV landings. Chen et al. [37] applied a graph-based method over depth maps extracted from monocular vision to evaluate ground safety. Several studies leveraged supervised or self-supervised depth prediction [138, 139] to improve understanding of terrain topography. These approaches enable UAVs to reason about elevation changes, thereby avoiding hazardous regions. Lim et al. [115] further advanced the field by integrating depth and semantic cues from LiDAR and cameras to support robust and real-time SLZ detection.

To improve scene understanding and robustness in unstructured environments, several works employed sensor fusion. Liu et al. [120] projected 3D LiDAR data into camera

views to refine semantic labeling. Zou et al. [247] proposed a multi-stage SLZ detection system combining point cloud and image data for terrain analysis and neural recognition. Lim et al. [115] demonstrated an efficient real-time system using both LiDAR and camera data to analyze slope and surface categories. These multimodal methods allow UAVs to operate effectively in visually ambiguous or cluttered scenes.

Handling emergencies requires evaluating not only terrain suitability but also the associated risks. Guerin et al. [71] introduced damage grading and uncertainty modeling to guide UAV emergency landing. Loera et al. [126] proposed a risk-aware planning method based on terrain hazards. Bong et al. [22] utilized dynamic segmentation and open-vocabulary models to generalize SLZ detection in unfamiliar environments. In contrast to binary segmentation, Abdollahzadeh et al. [1] introduced a continuous safety score regression model, allowing fine-grained interpretation of terrain safety.

Detecting humans and dense crowds is essential for ensuring UAVs avoid populated zones. Safadinho et al. [184] and Shao et al. [193] introduced deep vision-based models for real-time human detection. Gonzalez et al. [70] employed CNNs and density maps to identify crowded regions, while works like [210, 164] used lightweight CNNs for fast deployment. Advanced YOLO models [188, 13] and Bayesian fusion across multiple UAVs [88, 89] further improved accuracy. Surveillance-based approaches [69, 140] monitored crowd dynamics, supported by datasets like UAV-Human [111] and UAV-CROWD simulator [172], which aid in training and evaluating UAVs for people-aware SLZ detection.

Various frameworks have been proposed to enhance UAV autonomy and safety. Yang et al. [232] introduced a semantic SLAM system for indoor UAV landing. Badiya et al. [16] proposed a robust landing pipeline combining YOLOv5, DeepSORT, and PID control to handle obstacle-laden environments. Springer et al. [199] developed an appearance-based autonomous landing approach using synthetic data and a U-Net segmentation model, focusing on unstructured environments. Dataset creation and simulation tools, such as those from Peñarroya et al. [165], have been critical in benchmarking performance under synthetic and real-world conditions. These developments underscore the importance of deep learning and data-driven models in improving UAV navigation, especially in complex or unknown terrains.

2.4.3 UAV Navigation in dynamic environments

UAVs require robust object tracking and trajectory prediction techniques for efficient navigation and guidance. This section reviews advancements in these areas, integrating findings from various studies.

Object detection and tracking are fundamental for UAV applications like surveillance, reconnaissance, and search-and-rescue missions. Techniques such as YOLO-based frameworks accelerated with TensorRT [137] have enabled efficient real-time detection and landing recognition. Template-driven Siamese networks [201] have addressed challenges like occlusion and appearance changes, achieving competitive results on UAV benchmarks. Optimized neural network-based tracking algorithms and deep learning models combining detection and tracking [159, 95] have further enhanced real-time multi-object tracking capabilities.

Strategies for real-time ground target tracking in dynamic environments [43] have been demonstrated, along with advancements in multi-object tracking using architectures like GM-YOLO and Transformer-based methods [234, 235]. Memory maps that integrate metadata with video object detection [99] have boosted tracking accuracy in both short- and long-term scenarios. The integration of geo-location data with neural network-based recognition [220] has also contributed to robust target tracking.

Trajectory prediction remains critical for UAV navigation, with surveys highlighting methods that leverage machine learning and reinforcement learning approaches [196]. Multi-trajectory model predictive control (mt-MPC) has been proposed for safe navigation in unknown environments [183]. Frameworks for UAV-assisted traffic speed prediction [244] have incorporated deep learning for spatiotemporal data processing, while particle filters and image segmentation techniques [236] have enhanced vehicle trajectory prediction from UAV imagery.

Integrated systems that combine detection, tracking, and navigation provide holistic solutions for UAV operations. These include pipelines leveraging YOLO-based detection with Re-ID datasets for advanced tracking and collision avoidance [11], dynamic visual SLAM systems for efficient mapping in 3D environments [222], and vision-guided adaptive tracking methods addressing emergency landings [53]. Prediction-based path planning frameworks have also been developed for dynamic crowd surveillance [47].

Additional advancements include methods for the detection of moving object using image registration [24], refined multi-object tracking frameworks utilizing deep reinforcement learning [173], and car detection models emphasizing road segmentation for

accuracy [78]. Efforts to improve multi-object tracking have introduced IoU matching-based methods for fast tracking [169], as well as deep learning frameworks for 6-DOF path planning and obstacle-free navigation [221].

Despite these advancements, challenges such as computational limitations, occlusion, and real-time performance persist. Emerging technologies like machine learning, 5G, and cloud computing are proposed as potential solutions [4]. Future research should focus on developing robust algorithms that handle dynamic environments, improve accuracy, and ensure efficient real-time operations.

2.4.4 CNN-based ordinal regression

Ordinal regression has a long story in statistics and machine learning [73, 105]. Early works proposed extensions of generalized linear models to predict ordinal response variables [209]. The most widely used model is the so-called proportional odds model, which uses the logistic function to represent the cumulative distribution of ordinal responses [142]. Most of machine learning models for ordinal regression reformulated the problem as multi-task binary classification [110]. Early methods in this regard include the ones using the perceptron [194], support vector machines [49, 110] and random forests [84]. This concept has been recently investigated using CNNs to estimate persons' age from images [156]. Basically, an ordinal regression model with K ranks can be formulated using a neural network with $K-1$ binary outputs, with the kth output predicting whether the target exceeds the rank r_k. Another approach used Siamese architecture to compute ranks from pair-wise comparisons between input images [167]. Similarly, [64] proposed depth estimation using CNN ordinal regression, where depth values are divided into non-uniform intervals using a spacing-increasing discretization strategy. To obtain consistent ordinal responses from the CNN network, Cao et al. [32] proposed the COnsistent Rank Logits (CORAL) method to ensure rank-monotonicity and consistent ordinal scores in the model outputs. The model however is rigid in the sense that all ranks are predicted from the same activation function on which intervals are defined by thresholds.

2.4.5 Limitations of Existing Vision-Based SLZ Methods

Existing vision-based SLZ methods exhibit several important limitations. First, many approaches assume access to prior maps, pre-surveyed landing sites, or visual markers

(e.g., fiducials on helipads or rooftops). While effective in controlled or known environments, these assumptions break down in unknown, rapidly evolving, or emergency scenarios where no infrastructure is available and the UAV must autonomously discover safe alternatives. Second, safety is often modeled as a binary label (safe/unsafe), which ignores the inherently ordered nature of landing risk and the need for graded safety margins near hazards. In such formulations, misclassifying a highly unsafe region as marginally unsafe is treated no differently than misclassifying it as fully safe, even though the operational consequences are dramatically different.

Third, many methods rely solely on RGB imagery, which captures rich appearance information but lacks explicit metric geometry such as slope, roughness, and height. This makes it difficult to distinguish, for example, between flat asphalt, sloped roofs, and cluttered vegetation, and limits robustness under challenging illumination or texture conditions. Fourth, evaluation practices are often misaligned with operational risk. Standard segmentation metrics such as mIoU or F1-score treat all pixel errors equally and do not account for error severity or ordinal structure; confusing "very unsafe" with "unsafe" is penalized the same way as confusing "very unsafe" with "very safe," despite the vastly different safety implications.

A further limitation is that many SLZ pipelines retain a static view of what is fundamentally a dynamic problem. People, vehicles, and other moving agents can invalidate a candidate landing zone seconds after it is identified, yet numerous methods operate frame-by-frame, without explicit temporal reasoning, tracking, or short-horizon prediction. Finally, the perception-to-decision pipeline is typically fragmented: segmentation, detection, and landing decision-making are treated as separate modules, often optimized in isolation. This modular design leaves little room for end-to-end, risk-aware reasoning that directly encodes asymmetric costs (e.g., overestimating versus underestimating safety) and mission-level constraints.

2.5 Motivation and Contributions

The limitations outlined in Section 2.4.5 show that existing vision-based SLZ methods are not yet fully aligned with the requirements of real-world UAV operations. In particular, there is a need for SLZ assessment that is graded rather than binary, risk-aware rather than purely geometric, and dynamic and integrated rather than static and fragmented. This thesis responds to these needs through three main contributions, which

together form a unified vision-based framework for SLZ assessment tailored to real-time deployment on resource-constrained UAV platforms.

SLZ Detection for UAVs Using Deep Regression. The first contribution formulates SLZ detection as a continuous safety mapping problem. Building on a semantic segmentation backbone, the proposed method maps terrain classes and contextual cues into continuous safety scores defined over the image plane. This representation enables fine-grained ranking of candidate landing sites, supports adaptive decision thresholds (e.g., in emergency scenarios), and interfaces smoothly with downstream guidance and control compared to hard, binary masks. It directly addresses the limitations related to binary safety modeling and reliance on pre-defined landing markers or maps discussed in Section 2.4.5. This contribution is detailed in Chapter 3.

Deep Ordinal Regression for SLZ Detection Using Photometric and Geometric Information The second contribution introduces OR-SLZNet, an ordinal regression framework for multi-level SLZ safety assessment in UAV imagery. Recognizing that landing suitability is inherently ordered rather than categorical, OR-SLZNet assigns each pixel an ordinal safety level by jointly exploiting photometric cues (color, texture) and geometric cues (e.g., flatness, slope, depth proxies) within an encoder–decoder architecture. Rank-aware losses and risk-aligned metrics explicitly encode the ordered structure of safety levels and the asymmetric cost of different misclassifications, improving robustness to annotation uncertainty and yielding more interpretable safety maps. This contribution targets the limitations related to unordered classification, RGB-only sensing, and misaligned evaluation metrics highlighted in Section 2.4.5. The model achieves real-time inference and shows strong generalization across multiple real and synthetic UAV datasets. These advances are presented in Chapter 4.

A Framework for SLZ Mapping for UAVs in Dynamic Environments. The third contribution extends SLZ assessment to dynamic environments through an integrated perception and decision framework. The proposed pipeline combines semantic segmentation-based terrain understanding with real-time object detection, multi-object tracking, short-horizon trajectory prediction, homography-based ego-motion compensation, and uncertainty estimation. By predicting the future occupancy of candidate regions and continuously updating the safety map over time-to-land, the framework main-

tains a temporally consistent estimate of safe landing areas in cluttered, time-varying scenes such as roads with traffic, construction zones, or pedestrian corridors. This contribution addresses the static, frame-by-frame assumptions and fragmented pipelines identified in Section 2.4.5, enabling end-to-end, risk-aware reasoning from pixel-level perception to landing decisions. The dynamic SLZ framework is documented in Chapter 5.

The above contributions are supported and disseminated through the following publications by the author:

- *A Vision-Based Framework for Safe Landing Zone Mapping of UAVs in Dynamic Environments*, IEEE Open Journal of the Computer Society, under revision, May 2025.
- *Visual Safety Mapping for UAV Landings Using Ordinal Regression Networks*, IEEE Transactions on Artificial Intelligence, accepted, November 2025.
- *Safe Landing Zones Detection for UAVs Using Deep Regression*, IEEE, 2022.[1]

[1] Available at: `https://ieeexplore.ieee.org/document/9867062`.

Chapter 3

SLZ Detection for UAVs Using Deep Regression

In this chapter, we concentrate primarily on extracting 2D data from UAV-acquired images to generate a safety map using deep learning techniques. To achieve this, We propose a deep learning-based approach that directly generates a safety map from UAV-acquired images. This map goes beyond simple safe/unsafe classification, providing a pixel-wise continuous safety score. This score indicates the landing suitability at every location within the image, enabling informed decision-making for autonomous UAV landing.

We utilize a supervised regression model built on a semantic segmentation framework. This framework trains on a labeled dataset where each image segment is assigned a corresponding landing safety score: Low-risk, Medium-risk, and High-risk. Additionally, to account for potential collisions with obstacles like walls and vehicles, we incorporate a safety margin around these structures. This margin smoothly transitions the safety score from high-risk near the obstacle to safer zones further away.

Our experiments encompass images captured from both nadir (vertical) and oblique viewpoints in diverse environments, including urban and natural landscapes. In contrast to existing methods like [92], which categorize the terrain as simply safe or unsafe, our approach offers a more nuanced solution. The continuous safety map empowers the UAV to adapt its landing strategy based on the detailed risk assessment provided for every image pixel.

The obtained results demonstrate the promising capabilities of the proposed framework for real-world UAV landing safety evaluation.

3.1 Methodology

This section details our methodology for generating a UAV safety map using two-dimensional information extracted from RGB images captured by a UAV-mounted camera. This safety map serves as the foundation for extracting SLZs suitable for autonomous UAV landing under normal or emergency conditions, relying solely on real-time environmental perception. The primary objective is to minimize potential damage risks to the UAV and surrounding elements during landing.
Conventional methods focus on preventing UAV landings on specific object types (e.g., humans, buildings) by employing object detection techniques and circumventing them using rectangular bounding boxes that designate unsafe zones. Our approach deviates from this strategy by proposing a segmentation-inspired method that produces a comprehensive safety map encompassing all object types.

3.1.1 Deep Regression for Safety Score Prediction:

We formulate SLZ detection as a deep regression problem, aiming to obtain continuous safety scores for each location within the image. Deep regression techniques, prevalent in computer vision, are adept at predicting continuous outputs from images, such as head pose, age estimation, or depth perception [105, 32]. The emergence of deep learning has empowered deep regression using Convolutional Neural Networks (CNNs) to map input features to outputs via intricate non-linear transformations. In essence, these CNN architectures comprise several convolutional layers typically followed by fully-connected regression layers that incorporate linear or sigmoid activation functions [105]. Notably, these models have surpassed the state-of-the-art in numerous traditional computer vision tasks, including image classification and semantic segmentation tasks [182].

3.1.2 Encoder-Decoder Architecture for Safety Map Generation:

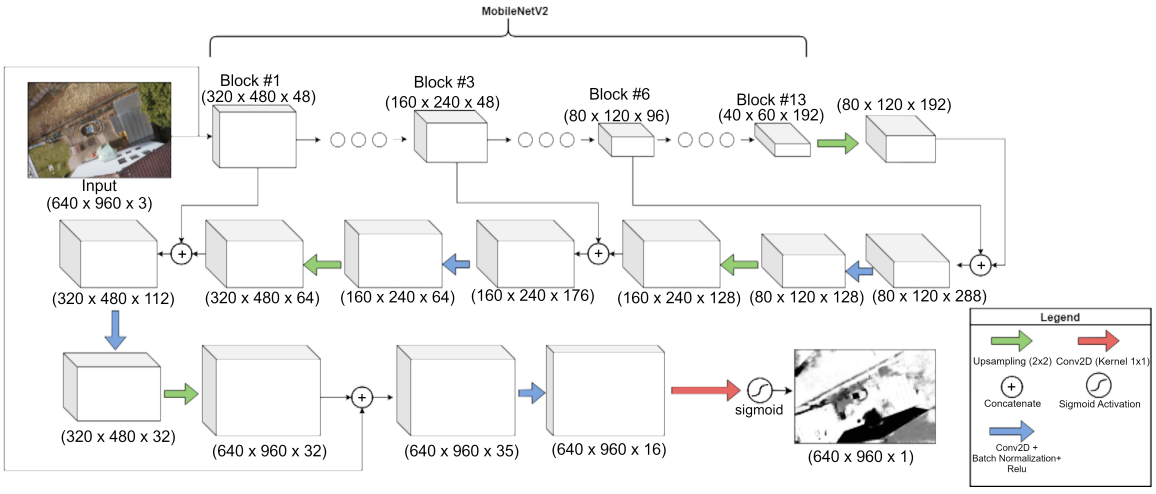
Our deep regression model leverages an encoder-decoder architecture(Figure 2.19), which has achieved remarkable success in semantic segmentation tasks exemplified by the U-Net(Section 2.3.2) model and its variants [182]. However, unlike segmentation techniques that assign categorical labels for thematic classes [158, 25], our model generates contin-

uous safety values across three primary classifications: low risk, medium risk, and high risk. These classifications reflect the landing suitability of each location. Additionally, we incorporate a smooth safety transition (margin) between vertical and flat structures to account for potential obstacles, resulting in continuous safety values between the three main risk levels. The ultimate goal is to generate a safety heatmap that facilitates the extraction of the safest landing zones through straightforward thresholding techniques [26].

3.1.3 Network Architecture Details:

Figure 3.1 illustrates our SLZ detection architecture. Similar to segmentation models, the core structure is comprised of consecutive encoding and decoding sub-networks. The encoder sub-network progressively compresses the input image into a latent-space representation via an encoding function ($z = f(x)$) . Conversely, the decoder sub-network upsamples the latent representation ($y = g(z)$) to recover the output, effectively capturing the essential semantic information from the input that is crucial for predicting the safety scores (y). The decoder's final output is a dense safety map that retains the same spatial resolution as the input image.

Figure 3.1: Architecture of the modified U-Net with a MobileNetV2 encoding phase network.

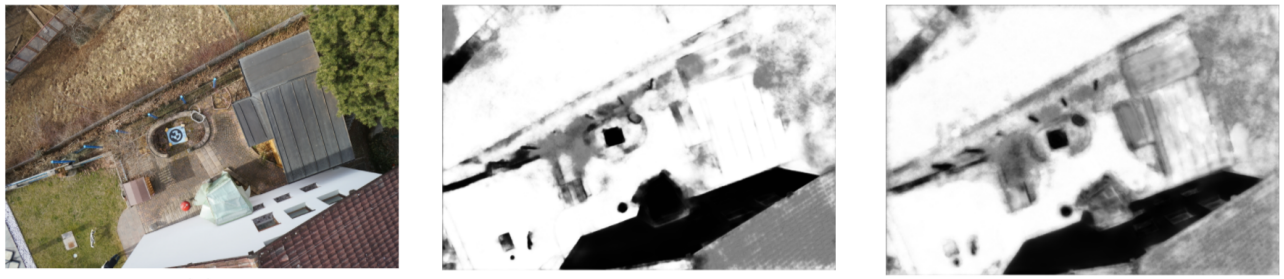


To accommodate the computational constraints of a UAV with limited processing power, our framework adopts a network based on the MobileNetV2 back-end [187]. This pre-trained model leverages the ImageNet dataset [55] for improved performance.

During the decoding phase, the encoder's output is upsampled and passed through convolutional layers with a 3×3 pixel kernel size. This process reintroduces encoded information from specific blocks (13, 6, 3, and 1) and the original input image through concatenation. In conventional semantic segmentation architectures with K classes [182, 128], a softmax activation layer is employed to generate a label for each pixel from the K-dimensional output vector. Our architecture utilizes a sigmoid activation function in the final layer, producing a prediction value y ranging from 0 to 1 for each pixel, representing the predicted safety score.

For visualization purposes, the output is transformed into a grayscale image by scaling the safety score map's dynamic range from $[0, 1]$ to $[0, 255]$, as shown in Figure 3.2.

Figure 3.2: (a) Urban Area sample from the dataset (b) Prediction map from the input (c) Prediction map including security border with Low-risks represented with lighter pixels and higher risks represented with darker pixels.



3.2 Experimental Results

3.2.1 Dataset Collection:

This section details the experimental evaluation of our methodology for generating UAV safety maps. To improve the model's generalizability, We investigated two common drone view scenarios: the vertical (nadir) view and the oblique view.

To train our model effectively, we established ground truth data derived from the RGB images. Each location within an image is assigned a corresponding safety value. For enhanced visualization, these safety values are encoded within the grayscale range (0 to 255). To maintain simplicity, we categorized safety into three primary grades:

Unsafe: 0 (Black) - Represents areas with high collision risk.**Moderately Safe:** 127 (Gray)- Areas with some potential hazards. **Safe:** 255 (White)- Low-risk areas suitable for Landing.

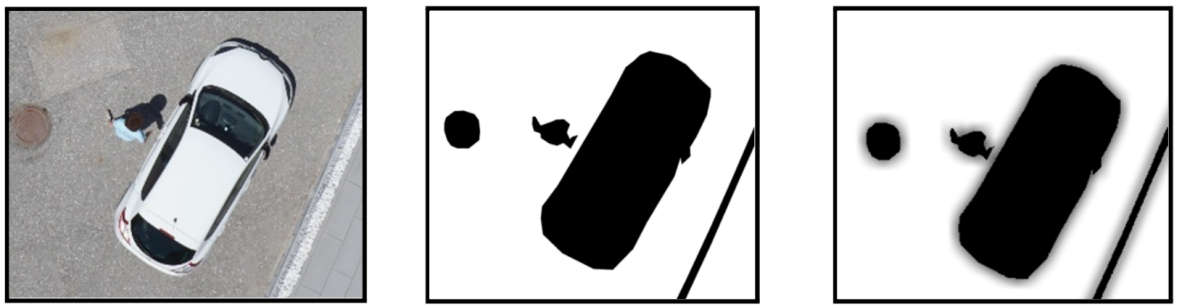
ICG Dataset

The ICG Semantic Drone Dataset [83] concentrates on semantic understanding of urban environments to enhance the safety of autonomous drone flights and landing maneuvers. The imagery showcases over 20 houses captured from a nadir (bird's-eye) perspective at an altitude ranging from 5 to 30 meters above ground level. The dataset offers a publicly available set containing 400 high-resolution images (6000x4000 pixels) with 24 thematic classes for semantic understanding. Additionally, the dataset includes bounding box annotations to aid in person detection tasks, promoting the development of safer autonomous drone flight and landing procedures. We generated the safety ground truth by converting the dataset's thematic classes into safety scores:

- **Safe:** Paved areas, gravel, dirt, grass, and ar-markers.
- **Moderately Safe**: Rocks, vegetation, and roofs with an incline.
- **Unsafe:** Cars, trees, walls, windows, doors, fences, fence poles, dead trees, obstacles, people, unlabeled regions, water, pools, dogs, bicycles, and conflicting objects.

The example of conversion of original segmentation mask of ICG dataset into gray-scale three level safety map has illustrated in Figure 3.3 (b).

Figure 3.3: (a) Original segmentation map from [83] (b) Grayscale conversion produced from risk levels (c) Grayscale conversion produced from risk levels with security border



MidAir dataset

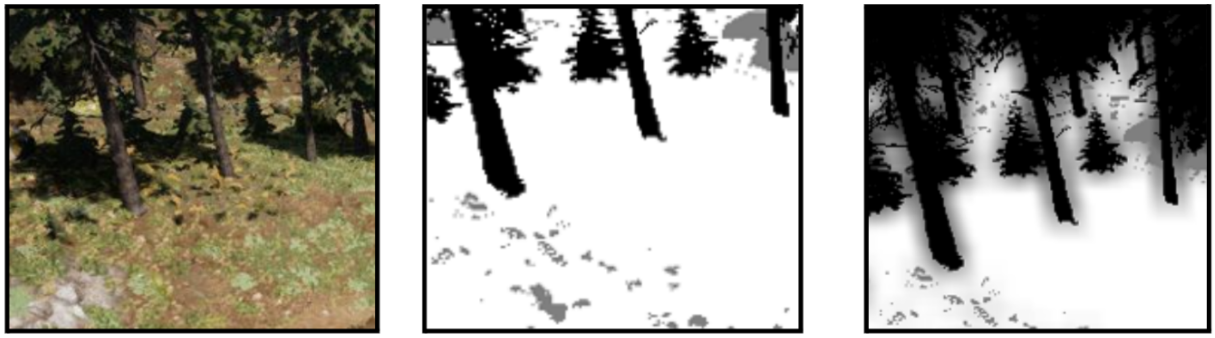
The MidAir dataset [63], meticulously crafted by the Montefiore Institute, serves as an invaluable resource for researchers developing algorithms for low-altitude drone flights. This dataset offers extensive information, including RGB images of size 1024×1024 pixels, captured by multi-modal sensors. Additionally, it provides surface normal orientation, depth, object semantics, and stereo disparity.

The dataset has 420,000 training frames, divided into 54 distinct trajectories, each encompassing diverse weather and seasonal conditions. The dataset also offers three different environment maps, further increasing its versatility and applicability. Since the dataset is voluminous, we selected a subset of 3855 images and segmentation maps, sampled from various weather conditions (i.e., sunny, foggy, etc.), then splitted into two parts: 75% for training and 25% for testing. Here's a breakdown of the conversion process for ICG dataset:

- **Safe:** Dirt ground, ground vegetation, and road.
- **Moderately Safe**: Rocky ground, boulders, and train tracks.
- **Unsafe:** Man made construction, Road Sign, Other man-made stuff, Water plane, empty, Animals.

The example of conversion of original segmentation mask of MidAir dataset into grayscale three level safety map has illustrated in Figure 3.4 (b)

Figure 3.4: (a) Original segmentation map from [63] (b) Grayscale conversion produced from risk levels (c) Grayscale conversion produced from risk levels with security border



To enhance the distinction between safe and non-safe areas, a safety margin has been implemented at all transitions between upright and flat structures. This margin creates a smoother gradient between safety levels, improving navigation and decision-making for autonomous systems.

The safety margin is generated by applying a Gaussian filter to the original ground truth data, which initially contains three distinct safety levels. This filtering technique effectively blurs the boundaries between the levels, creating a smoother transition zone. Importantly, the unsafe (black) and medium safe (gray) zones within the ground truth remain unaltered. This ensures a clear distinction between critical danger zones and areas with some level of safety.

The resulting safety margin is visualized in Figure 3.3 (c) and Figure 3.4 (c) for illustrative purposes.

3.2.2 Model Training Details:

The Keras API within TensorFlow 2.7 was used to build the model on an NVIDIA GeForce RTX 2070 GPU equipped with 6 GB of memory. The segmentation maps, represented as grayscale images with three categories (low, medium, and high) assigned distinct values (255, 127, and 0) (refer to Figure 3.2), were normalized by division by 255 to yield safety scores corresponding to each class. A custom generator was then employed to feed the processed data into the network. The training process encompassed 50 epochs

with a batch size of one image due to image size constraints. The model has been trained with the Adam optimizer, with a default learning rate of 5×10^{-4}, decreasing at a rate of 0.1 after a stagnation of the validation loss for four consecutive epochs, with a minimum value of 1×10^{-7}.

3.2.3 Model Evaluation

We trained the models for 50 epochs, allowing them to predict safety maps within a specific range relevant to the target region. The obtained scores, initially ranging from 0 to 1, were then converted for visual representation:

Grayscale: Scores were mapped to a grayscale intensity between 0 (black) and 255 (white), with higher values indicating greater safety.

Heatmap: Using the Hue, Lightness, and Saturation (HLS) color space, the scores were transformed into a heatmap. Green represents safe areas, yellow indicates slightly safe zones, and red depicts unsafe regions.

Visualizations for urban and natural scene predictions are presented in Figures 3.5 and 3.6, respectively.

Since the safety maps are continuous scores (0-1), standard segmentation metrics like Intersection over Union (IoU) or Dice coefficient are not suitable for evaluation. While ground truth masks might use labels like "Low," "Medium," and "High" risk, the predictions themselves hold continuous values. Therefore, we employed Mean Absolute Error (MAE) and Mean Squared Error (MSE) as the primary quantitative metrics. These metrics reflect the average difference between the predicted and ground truth safety values.

Our qualitative evaluation in Figures 3.5 and 3.6 visually confirms the effectiveness of our method. While the ICG dataset yielded positive results, it occasionally struggled to differentiate between grass and trees. This resulted in some instances where trees received low-risk scores on certain maps, while grass received high-risk scores on others. Conversely, the MidAir dataset generally produced clear visual outputs, accurately

depicting safe areas (primarily roads and grass) versus unsafe zones (mostly trees and rocks). As expected, the model encountered difficulties in darker and foggier scenes, which are inherently unsuitable for flying drones (refer to Figure 3.6).

Quantitative evaluation is presented in Table 4.8. We assessed performance using two metrics for both datasets, analyzing predictions with and without a safety margin between vertical and flat sutures. For the ICG dataset, the Mean Absolute Error (MAE) was 0.157 without a margin and improved to 0.127 when incorporating a margin. The MidAir dataset displayed similar trends, with MAE values of 0.169 and 0.129, respectively.

In terms of Mean Squared Error (MSE), the MidAir dataset – capturing a natural scene from a first-person perspective – achieved an average of 0.069 without a margin and 0.049 with a margin. The ICG dataset's MSE values were 0.122 and 0.056, again demonstrating improvement with the inclusion of a safety margin. These results collectively indicate that our model generates consistently accurate predictions. Notably, incorporating safety margins consistently enhanced performance across both training and test data, as shown in Table 4.8.

To further evaluate our approach, we compared it to a pure segmentation method. This involved training both datasets with a categorical cross-entropy loss function to produce segmentation outputs. Segmentation results are detailed in Table 3.2. Interestingly, we observed minimal differences in MAE and MSE between segmentation and regression for the ICG dataset. However, the MidAir dataset exhibited significantly lower MAE and MSE values when using regression compared to segmentation. This finding reinforces the advantage of employing regression for the specific task of SLZ detection.

Figure 3.5: Predictions on the test samples from the ICG dataset. (a) Input image (b) Ground truth (3 categories) (c) Safety score map map (grayscale) going from lighter (low landing risks) to darker (high landing risks) (d) Safety heat map (HLS colorspace) going from green (low landing risks) to red (high landing risks)

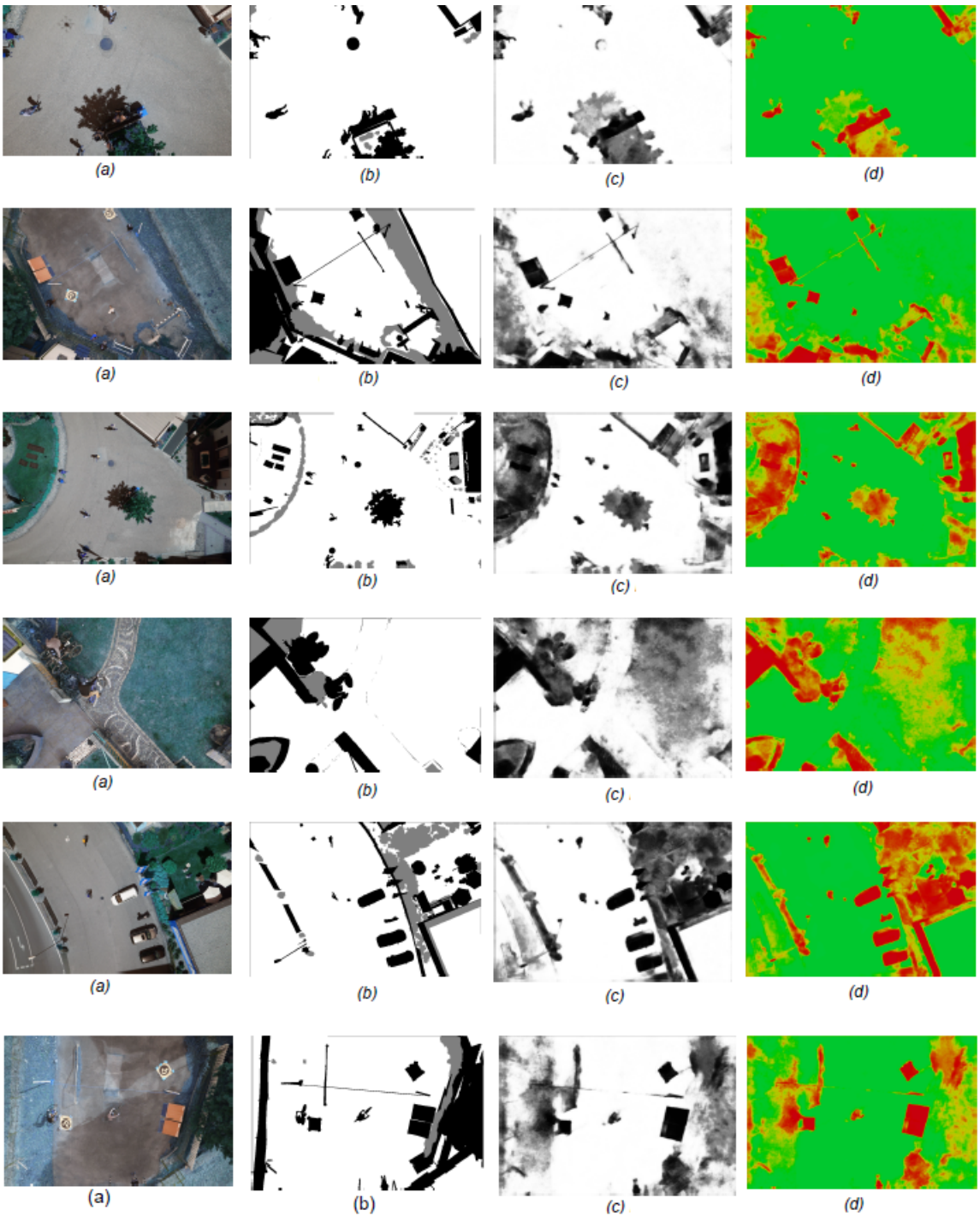


Figure 3.6: Predictions on the test samples from the MidAir dataset. (a) Input image (b) Ground truth (3 categories) (c) safety score map (grayscale) going from lighter (low landing risks) to darker (high landing risks) (d) Safety score heat map (HLS colorspace) going from green (low landing risks) to red (high landing risks)

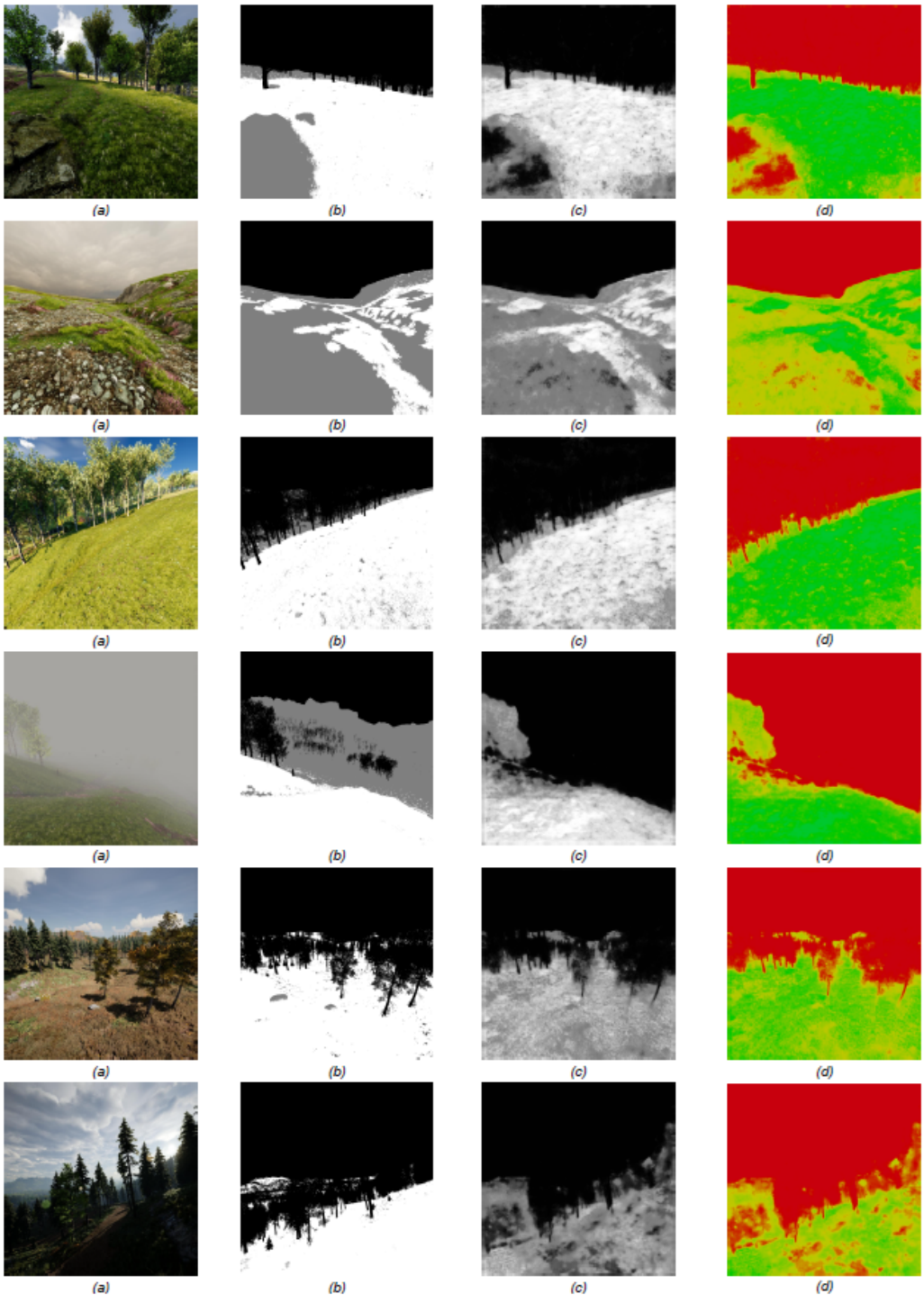


Table 3.1: Experimental results (Regression)

Dataset	**Loss**	**Margin**	**MAE**		**MSE**	
			Training	Testing	Training	Testing
ICG	MAE	No	**0.104**	**0.153**	0.073	0.112
ICG	MAE	Yes	0.120	0.206	0.062	0.109
ICG	MSE	No	0.177	0.193	0.077	0.104
ICG	MSE	Yes	0.171	0.240	**0.061**	**0.098**
MidAir	MAE	No	0.029	0.104	0.018	0.054
MidAir	MAE	Yes	**0.016**	**0.087**	**0.005**	0.045
MidAir	MSE	No	0.029	0.145	0.016	0.078
MidAir	MSE	Yes	0.020	0.124	**0.005**	**0.041**

Table 3.2: Experimental results (Segmentation)

Dataset	**Loss Function**	**Margin**	**MAE**		**MSE**	
			Training	Testing	Training	Testing
ICG	Categorical Crossentropy	No	0.113	0.152	0.054	0.108
MidAir	Categorical Crossentropy	No	0.065	0.196	0.031	0.1139

3.3 Conclusion

In this chapter, we proposed a method for SLZ prediction for UAVs based on supervised deep regression. The model is built on a segmentation backbone and is able to predict dense safety maps by taking into account the color/texture distribution of images as well as the proximity to upright and flat structures. We implemented the model on two different datasets containing images in both vertical (nadir) and oblique views as well as urban and natural scenes. Our results showed the huge potential of our method for accurately identifying SLZ. While the tests have been conducted entirely on existing aerial images datasets, further tests using real drones will enable to better evaluate the proposed approach. In addition, combination of 2D and 3D information will give precious cues about surface orientation and the presence of obstacles that cal potentially increase the accuracy of SLZ detection.

Chapter 4

Deep Ordinal Regression for SLZ Detection Using Photometric and Geometric Information

4.1 Introduction

Ensuring the reliability and safety of UAV operations is a major challenge for large-scale deployment, particularly in compliance with legislated safety rules and regulations [89, 107]. This necessitates the implementation of robust mechanisms for obstacle avoidance [240] and autonomous operation in environments lacking GPS or communication signals [150]. A key requirement in such scenarios is the ability to identify SLZs during emergency situations triggered by component failures, communication loss, or adverse weather conditions, all of which can lead to crashes and potential harm to people, property, or other vehicles [89]. SLZ detection also supports human pilots in executing planned missions more safely by enabling landing trajectory optimization. To achieve this, UAVs must avoid hazardous areas such as those containing dense vegetation, water bodies, buildings, or dynamic obstacles—and prioritize flat, obstacle-free regions devoid of people or animals. Integrating automated decision-making processes for detecting such areas is essential for achieving reliable and autonomous UAV navigation.

Past methods for SLZ detection used either non-vision-based, vision-based, or combined approaches to analyze the scene surrounding the UAV [191]. Non-vision-based approaches use mainly Light Detection and Ranging (LiDAR) to estimate digital eleva-

tion models (DEMs) from which a 3D representation of the terrain is generated [129]. This is achieved by computing the reflection time of laser beams projected from the UAV to the target area. Although LiDAR can provide highly accurate 3D terrain maps with point clouds [228], it is a very costly sensor compared to cameras. Vision-based approaches use an onboard camera, which can provide continuous images and videos about the UAV's immediate environment. Compared to LiDAR, cameras have the advantage of being inexpensive, lighter, and a passive means for terrain sensing. In addition, efficient computer vision techniques can be used to extract relevant cues to identify SLZs [20, 89].

Most vision-based SLZ detection methods relied on prior knowledge about the landing sites, which can be a 3D elevation map [191], or designation of static landing locations using easily-identifiable markers or patterns [226]. However, this knowledge is not always available, especially in emergency landing situations which can occur at arbitrary locations. Additionally, in GPS-denied environments, the UAV should be endowed with computational intelligence that can visually explore the UAV surroundings to quickly identify potential SLZs [90]. The progress of Convolutional Neural Networks (CNNs) in the last decade has significantly enhanced computer vision for different scene understanding tasks, including image classification, object detection and tracking [112, 206], and semantic segmentation [203]. This, in turn, has spurred research for vision-based UAV navigation, more particularly for SLZ detection [89, 226].

For example, methods have been proposed for detecting crowded areas during UAV landing [70, 89, 164, 210]. These methods proposed architectures that perform binary (crowd/non-crowd) segmentation of the scene; thus mapping the scene into two classes: safe and high-risk landing zones. Other methods perform multi-label segmentation of the scene, by assigning safety labels to image regions (e.g., safe, moderately safe, and unsafe) [72, 139]. The class labels correspond generally to semantic concepts such as crowds, vehicles, and vegetation, which give a rough indication about damage risk for the UAV or third parties. These methods suppose the safety classes are independent to each other, and, consequently, they are treated equally during training and validation. However, safety levels have a strong ordinal relationship, which cannot be captured by multi-class segmentation. For example, collision with a human should have a much higher cost than trying to land a UAV on a rough rocky ground. Also, a security perimeter should be guaranteed for the surrounding of very unsafe objects (e.g., moving objects) to avoid

collisions. Consequently, the predicted safety maps should be be spatially consistent and carry the most relevant information to minimise the risk of damage.

This chapter presents the development of a deep-learning model designed for vision-based detection of SLZs using an ordinal regression strategy. The proposed model, referred to as **OR-SLZNet** (Ordinal Regression for SLZ Detection Network), aims to generate dense safety maps from visual inputs in both urban and natural environments. These safety maps can be seamlessly integrated into autonomous UAV navigation pipelines for safer landings.

Conventional semantic segmentation techniques often treat class labels as independent categories. However, when assessing safety, such an assumption is limiting. Safety levels inherently possess an ordinal structure, ranging from highly unsafe to highly safe conditions. The OR-SLZNet model addresses this limitation by formulating the SLZ detection problem as a multi-task ordinal regression, enabling the model to learn and represent the ordered nature of safety labels more effectively.

To implement this concept, the safety spectrum is divided into five ordered levels: very unsafe, unsafe, moderately safe, safe, and Very Safe. Unsafe regions typically correspond to areas that pose a risk to UAVs or surrounding individuals and property (e.g., high vegetation, water bodies, pedestrians, cyclists, or infrastructure such as power lines). In contrast, safe regions are characterized by flat, unoccupied surfaces free from water and obstacles (e.g., lawns, pavements), minimizing the potential for damage upon landing.

The OR-SLZNet model adopts an encoder-decoder architecture inspired by U-Net [182], enhanced with task-specific adaptations to handle ordinal outputs. Each level of safety is predicted using a set of binary sub-tasks, allowing the model to incrementally assess the safety level of each pixel. The model also incorporates:

- A **dual encoder** to extract multi-modal features from color (RGB), depth, and terrain flatness inputs.
- **Attention mechanisms** to prioritize semantically relevant features for accurate sub-task predictions.
- **Multi-scale convolutional blocks** to enrich contextual understanding across spatial resolutions.

The main technical contributions of this work are outlined below [185]:

- A novel deep-learning model, **OR-SLZNet**, is introduced for ordinal safety prediction in UAV imagery. It employs multiple binary classifiers corresponding to each safety level, allowing for more consistent and interpretable safety ranking. The model integrates attention gates and multi-scale convolutions to better capture local context and critical obstacle details.

- To accurately assess terrain safety, the model leverages **multi-modal inputs**, including RGB images for semantic cues, depth maps for distance awareness, and flatness indices for geometric reasoning. These complementary inputs enhance the model's ability to identify viable SLZs that are both safe and practical for landing.

- The training process is supported by safety annotations derived from several public datasets, including **AeroScapes** [155], **ICG** [83], **MidAir** [63], **UAVid** [136], and **Valid** [38]. These datasets span a diverse range of aerial scenes, encompassing both synthetic and real-world imagery, vertical and oblique viewing angles, and a mixture of natural and built environments. Safety levels are annotated by aggregating thematic classes according to criteria commonly used in SLZ evaluation such as flatness, slope, and absence of dynamic or structural hazards.

Experimental evaluation on these datasets demonstrates that OR-SLZNet outperforms traditional semantic segmentation models in the context of SLZ detection, providing more accurate and consistent safety maps.

The rest of this chapter is organized as follows: Section 4.2 presents the proposed methodology. Section 4.3 presents experimental results validating our work. We end the chapter with a conclusion and future work perspectives.

4.2 Methodology

The OR-SLZNet aims to predict dense SLZ maps from the aerial images acquired by an UAV. The model is built on an architecture trained in an end-to-end fashion to extract tailored features, enabling the prediction of an ordinal safety value for each image location. Real-world scenes are often complex, and RGB images alone may not provide sufficient information to accurately assess landing safety. To overcome this limitation, RGB channels are complemented by depth data, surface flatness, and surface inclination features. Depth provides an estimate of the relative distance of objects to the

UAV; whereas flatness and inclination are useful to represent the local surface geometry. Herein, we describe the process for extracting these features.

4.2.1 Input cues for SLZ detection

Depth map

Depth is an important information for UAV navigation. It can give a good estimate of the relative object distance to the camera, which can help identify obstacles and terrain characteristics [174]. Therefore, depth is used as one of the input channels to predict our safety maps. However, depth can be difficult to estimate from single images since no motion or stereo information is available. Recent advancements in computer vision and deep learning techniques have enabled the development of methods for depth estimation on monocular images [148], [229] and [230]. These methods rely on cues such as context, texture variation, shading, and defocus. In our model, we use the DeepAnything model [230], a two-stage framework first predicting depth up to an unknown scale, and then using 3D point cloud encoders to recover a smoother depth map.

Flatness map

Flatness is a characteristic that indicates whether an area is flat enough and clear of obstacles (e.g., bushes, stones, etc.). It is generated using the idea presented in [35, 149], which is based on using depth gradient values to estimate a region flatness. We use the gradient magnitude of the input grayscale depth image to identify depth discontinuities. Let $d(x, y)$ be the calculated gradient magnitude on the depth map. Flatness is then given by :

$$F(x, y) = 1 - \exp(-G_\sigma * d(x, y)) \tag{4.1}$$

where G_σ is a Gaussian convolution kernel of size $m \times m$. For the datasets, we found that $m = 13$ gives the best results. Finally, the negative of the resulting filtered image is used to represent the flatness map, which represents the degree at which the surface is void of bumpy elements that can cause damage for the UAV.

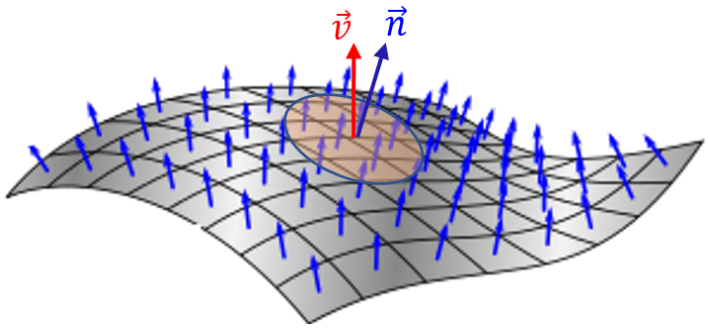


Figure 4.1: Illustration of the inclination computation using local surface normal vectors estimated from depth.

Inclination-map

Simply having a flat surface is not enough to ensure a safe landing. Another important cue that should be considered is surface inclination or slope. This can be estimated using the angle between the local surface normal and the vertical axis (see Figure 4.1 for illustration). Surface normals can be determined when 3D digital surface map is available. Otherwise, a 3D representation of the scene can be estimated, which can be challenging to obtain in monocular vision. In our study, we utilized the repository from [15] to extract surface normals from RGB images for all datasets, except for MidAir [63], which already provided surface normals as part of the dataset. Then, we compute the average normal for each location using the location neighborhood. Finally, the inclination is computed using the scalar product between the normalized versions of the averaged surface normal $\vec{n}$ and the upward vector $\vec{v}$:

$$I(x, y) = 1 - \vec{v} \cdot \vec{n}(x, y) \tag{4.2}$$

Note that the upward vector $\vec{v}$, signifying the zenith direction of the 3D world coordinate system, corresponds to $(x, y, z) = (0, 0, 1)$. It can be determined from the UAV navigation parameters or estimated from some vertical objects (e.g., buildings, trees, etc.). In the above formula, $I = 0$ indicates a horizontal surface, whereas $I = 1$ indicates a vertical surface. The map in Figure 4.1 illustrates the inclination computation using local normal vectors.

4.2.2 OR-SLZNet Model Architecture

We propose a deep ordinal regression network to deal with the SLZ detection problem. The model, as illustrated in Figs 4.2 has a multi-modal input including six channels (RGB, depth, flatness, and inclination). The model is based on an encoder-decoder Unet architecture designed to predict ordinal safety outputs for each image location. The architecture is composed of three-level feature extraction modules: low-level (LL), mid-level (ML), and high-level (HL) feature extraction modules.

The LL module is composed of two parallel sub-Unets made of sub-encoders and sub-decoders specialized in parsing photometric and geometric features, respectively. Each sub-encoder consists of four convolution blocks and ends up with a bottleneck of 256 features. These are then passed to the sub-decoder which is composed of three up-sampling blocks (the forth block is transparent in the architecture is used in the pre-training of each sub-Unet). The outputs of the two sub-decoders are concatenated to constitute intermediary features.

The ML module generates contextual information by using dilated convolutions with sizes (d_1, d_2, d_3, d_4). This technique captures context from larger regions and improves model performance for segmenting objects of various sizes. It is especially beneficial for UAV imaging, where object sizes vary due to changes in UAV altitude and camera view angle. Dilated Convolution enables the model to recognize objects with different sizes [39].

Finally, the HL module refines the intermediary features to solve each of the four sub-tasks. Each sub-task is a pipeline that starts by generating two feature maps using convolution with 3×3 and 7×7 kernel sizes, then passed through the Spatial-Channel Attention Block (SCAB) (described in Section 4.2.4). This is followed by two convolution blocks with 3×3 and 1×1 kernel sizes, respectively.

4.2.3 Safely prediction using extended binary classification

Inspired from [110], a series of $K - 1$ binary classification sub-tasks are used to output K ordinal safety levels per location. The classification sub-tasks share the same feature representation learned from the low-level feature extraction module. However, each sub-task is characterized by its own high-level feature extraction pipeline. Each pipeline also contains an attention gate that sifts through the decoded features to select the most relevant (spatial/channel) information for each rank prediction.

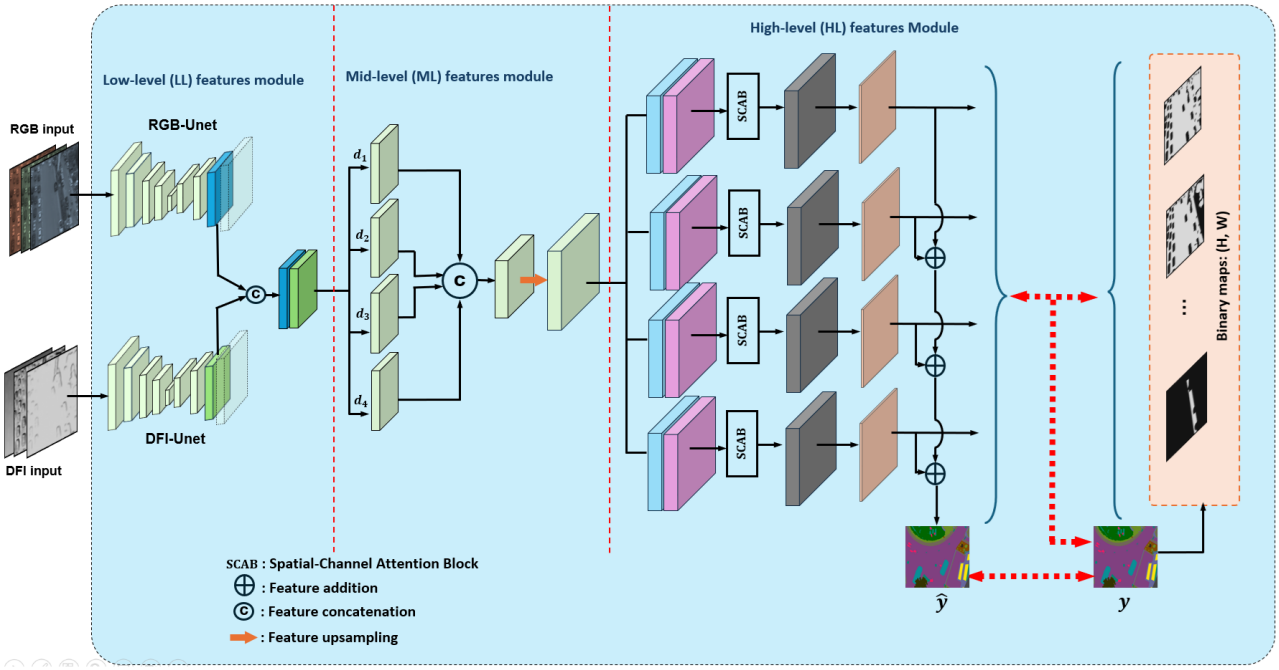


Figure 4.2: The detailed architecture of the OR-SLZNet model.

Suppose that we have a space of K safety ranks $\mathcal{S} = \{s_1, ..., s_K\}$, with ranks ordered such that $s_1 \prec s_2 \prec \cdots \prec s_K$ (in our case the relation $s_i \prec s_j$ means rank s_j is safer than rank s_i). We want to predict for every pixel its optimal rank by transforming the problem into a series of $K-1$ binary classification sub-tasks $T_1, T_2, ..., T_{K-1}$. Each binary task T_k predicts whether the target variable is less than, or greater equal, to a specific safety rank s_k.

In our implementation, the safety outcome space is discretized into $K = 5$ levels, as follows: $\{$*Very Unsafe*, *Unsafe*, *Moderately Safe*, *Safe*, *Very Safe*$\}$. Given n training examples $D = \{(X^{(i)}, Y^{(i)})\}_{i=1}^{n}$ composed of images with their safety masks, the goal is to build a model representing the mapping $f : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X} \subseteq R^{n \times m \times d}$ is the space of input images and $\mathcal{Y} \subseteq N^{n \times m}$ is the space of masks whose element values are taken from the set $\mathcal{S}$.

Given a ground-truth safety rank Y_p for a pixel p, the optimal mapping f should minimize the cost of rank prediction $\hat{Y}_p$. A matrix $\mathcal{C}$ can be used to encode the pairwise cost $\mathcal{C}_{Y_p, \hat{Y}_p}$ of mis-assigning rank $\hat{Y}_p$ when the true pixel rank is Y_p. A popular cost function can be defined by the absolute error $\mathcal{C}_{Y_p, \hat{Y}_p} = |Y_p - \hat{Y}_p|$ [110]. When the cost of rank mis-assignment is nit symmetric, one can use an asymmetric loss function that

can put more penalty for mis-assigning lower to higher ranks, for example. This can be formulated as follows:

$$\mathcal{C}^{a}_{Y_p,\hat{Y}_p} = \alpha \left[\min(0, \hat{Y}_p - Y_p)\right]^2 + \beta \left[\max(0, \hat{Y}_p - Y_p)\right]^2 \tag{4.3}$$

where α and β are weights determining the penalty direction. When $\alpha > \beta$ (resp. $\alpha < \beta$), more penalty is given for mislabeling lower ranks with higher ranks (resp. mislabeling higher ranks with lower ranks). When $\alpha = \beta$, the formula boils down to a symmetric loss function.

While the input images are shared among the different binary classification sub-tasks, the coding of the ground truth will be different for each task. That is, each pixel p is associated an ordinal response vector $\mathbf{y}_p = (y_p^1, y_p^2, ..., y_p^{K-1})^T$, where y_p^k is the actual response value for pixel p with regard to the k-th sub-task, such that:

$$y_p^k = 1(Y_p \geq s_k) \tag{4.4}$$

where $1(\cdot)$ is an indicator funciton. Moreover, to ensure consistency between the different classifiers and the ranking, we assume that if $y_p^k > 0$, then $y_p^j > 0, \forall j < k$ [110]. In other words, a pixel is assigned a given safety rank if it has passed all lower safety ranks. Let $Y^{(i,k)}$ be the mask encoding all the pixel outputs for the k-th sub-task. Then, the training of all sub-tasks is done in parallel using the dataset $D_k = \{(X^{(i)}, Y^{(i,k)})\}_{i=1}^n$ for the k-th sub-task. One can also associate a weight for each data point $w^{(i,k)}$ representing the importance of the i-th sample to k-th classifier; for simplicity, we set $w^{(i,k)} = 1, \forall(i,k)$.

Let $h_k(X^{(i)}) \in \{0, 1\}$ be the prediction of the k-th sub-task. Then, the predicted full rank of every pixel in the image $\hat{Y}^{(i)}$ is given by the following function:

$$\hat{Y}^{(i)} = 1 + \sum_{k=1}^{K-1} h_k(X^{(i)}) \tag{4.5}$$

The ordinal regression model composed of several task modules is trained in an End-to-End fashion. This enables to automatically learn good features for defining each safety rank. Moreover, sharing low-level and mid-level representations between sub-tasks can enforce consistency between their predictions.

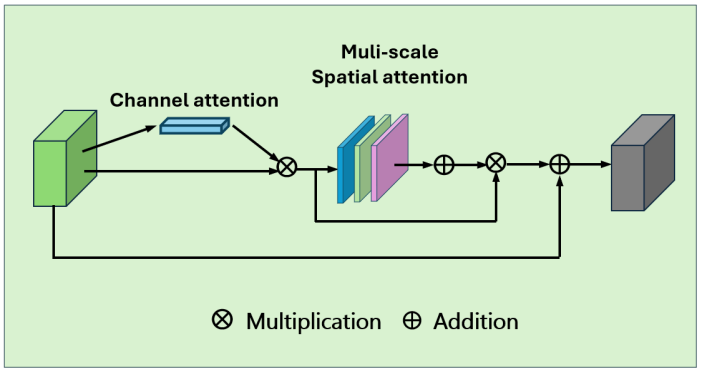


Figure 4.3: Spatial-channel Attention Block (SCAB) Architecture

4.2.4 Attention for SLZ detection

Attention is a powerful technique used to improve computer vision tasks such as image classification and segmentation by selectively focusing on the most informative regions and channels for better prediction. Over the past few years, several attention mechanisms have been developed [74]. For segmentation networks, attention is often included either in the encoder to extract richer encoded information [225], or in the the decoder to reconstruct a latent representation that fits a specified criteria given by the loss function.

For the SLZNet model, we integrated an attention mechanism at the top decoding blocks to reconstruct a representation to solve each sub-task. This is achieved through the Spatial-channel Attention Block (SCAB) layer which is composed of channel attention followed by spatial attention. The channel attention weights are computed through global average pooling, followed by sigmoid activation. These weights are multiplied by the input features. The spatial attention weights are computed through one standard and two dilated convolutions to enlarge the receptive field. The produced weights are averaged and pout through a sigmoid function and again multiplied by the input features. Finally, the original features are added as a residual to the output of the spatial attention. The SCAB module is particularly useful for putting focus on small critical objects such as humans and small vehicles that are difficult to recognize at high altitudes.

4.2.5 Loss function

To train the SLZNet model, we use a loss function that handles $K-1$ separate outputs corresponding to each task, in addition to the asymmetric regularization to penalize severe mislabeling of non-safe zones. The complete loss function over all pixels of image

$I^{(i)}$ is given by:

$$\mathcal{L}_{tot} = \sum_{k=1}^{K-1} \mathcal{L}_k(Y^{(i,k)}, \hat{Y}^{(i,k)}) + \mathcal{L}_g(Y^{(i)}, \hat{Y}^{(i)}) \tag{4.6}$$

where $\mathcal{L}_k$ is the loss function associated with the k-th sub-task, whereas $\mathcal{L}_g$ is the global rank loss. The sub-task loss $\mathcal{L}_k$ is composed of dice and weighted binary cross entropy losses, whiuch is given by:

$$\mathcal{L}_k = DICE(Y^{(i,k)}, \hat{Y}^{(i,k)}) + wBCE(Y^{(i,k)}, \hat{Y}^{(i,k)}) \tag{4.7}$$

The $DICE$ between two sets A and B is equal to:

$$DICE(A, B)) = 2|A \cap B|/(|A| + |B|)$$

The $wBCE$ is designed to enforce the rank consistency between sub-tasks and is given for a pixel p and sub-task k as follows:

$$wBCE = -w_1^k y_p^k \log(y_p^k) + -w_0^k (1 - y_p^k) \log(1 - y_p^k) \tag{4.8}$$

where w_1^k and w_0^k are the enforcing consistency weights. For example, if a rank of a pixel p is $Y_p = 3$, sub-tasks T_1 and T_2 should predict $y_p^1 = y_p^2 = 1$, whereas sub-tasks T_3 and T_4 should predict $y_p^1 = y_p^2 = 0$. Consequently, we assign $w_1^1 \gg w_0^1$, $w_1^2 \gg w_0^2$, $w_1^3 \ll w_0^3$ and $w_1^4 \ll w_0^4$.

Finally, the global loss $\mathcal{L}_g$ for each pixel p is given by Eq. (4.3). To obtain a prediction of the full rank of a pixel $\hat{Y}_p$, we use Eq. (4.5) where the prediction of each sub-task h_p^k is obtained by thresholding the sub-task output $\hat{y}_p^k$ using a shifted Sigmoid function $h_p^k = [1 + \exp(a(\delta - \hat{y}_p^k)]^{-1}$. The best values for the constants a and δ are 100 and 0.5, respectively.

4.2.6 SEG-SLZNet

SEG-SLZNet with RGBDFI (late-fusion). The model ingests a 6-channel tensor formed by concatenating RGB (3-ch) and DFI (3-ch). It splits this into two streams, each passed through its own UNet with an ImageNet-pretrained, frozen encoder and a trainable decoder (so pretrained encoders stay stable while decoders adapt to SLZ semantics). From

each branch, we take the final high-resolution decoder feature map, concatenate them along channels to fuse appearance (RGB) with geometry/illumination cues (DFI), then apply a lightweight SegmentationHead to output per-pixel logits for the 5 safety classes (softmax for inference). Training uses cross-entropy on class indices. This setup keeps 3-channel encoders (so pretrained weights are valid) while leveraging complementary RGB/DFI information for more robust safety mapping.

4.3 Experimental Results

4.3.1 Dataset annotation

To demonstrate the effectiveness of safety prediction, we conducted several experiments examining different aspects of the proposed model. Since no existing dataset addresses the safety prediction problem like our approach, we generated safety annotation for several datasets proposed for semantic segmentation. These include AeroScapes [155], ICG [83], MidAir [63], UAVid [136] and Valid [38]. These datasets include a variety of acquisition scenarios (e.g., different camera angles, different altitudes) and scene types (e.g., real and synthetic data, urban and natural environments).

To create a five-level safety classification for each RGB image across all datasets, we generated safety ranks for the semantic classes within the datasets. For visualization purposes, safety ranks were represented using specific RGB colors: Red for *Very Unsafe (VU, rank 0)*, Brown for *Unsafe (U, rank 1)*, Orange for *Moderately Safe (MS, rank 2)*, Cyan for *Safe (S, rank 3)*, and Green for *Very Safe (VS, rank 4)*. The detailed division of semantic classes into safety levels for different datasets is provided in Table 4.1. It is important to note that due to the diversity of classes across datasets, similar classes may have been assigned to different safety categories. Additionally, all images, along with their corresponding maps and masks, were resized to 512×512 pixels for consistency. The images selected for each dataset were then split into three subsets: 80% for training, 10% for validation, and the remaining 10% for testing. A detailed description for each dataset follows (see Figure 4.4 for illustrative examples):

AeroScapes Dataset [155]

Comprises 3,269 high-resolution real-world images of size 1280 × 720 pixels, captured by a drone flying at altitudes (from 5 to 50 meters) on a variety of urban and natural

Table 4.1: Safety Level Classification Overview.

Dataset	Very Safe	Safe	Moderately Safe	Unsafe	Very Unsafe
Aerospace	paved-area, grass, sport field	road	construction	vegetation, obstacle, car	person, bike, drone, boat, animal, water, sky, background
ICG	paved-area, grass, AR-marker	dirt, gravel	roof	rocks, vegetation, car, tree, wall, fence-pole, bald-tree, obstacle, window, door, fence	person, unlabeled, bicycle, conflicting, water, pool, dog
MidAir	ground vegetation	road, dirt ground	rocky ground, train track, boulders	man-made construction, road sign, trees, other human-made stuff	water plane, empty, animals
UAVid	low vegetation	road	building	static car, moving car, tree	humans, background clutter
Valid	pavement	land	roof, building, bridge,	small vehicle, large vehicle, chair, stones, lamp, fence, garbage bin, sign, ship, pier-rubble, tree, traffic light, other plant, ice, bus stop, tunnel, harbor, other low obstacle	plane, power line, other high obstacle, animal, person, water, pool, background

scenes. The dataset background class posed a challenge due to its complexity, making it difficult to annotate. To address it, we replaced his class with four additional subclasses: grass, water, paved area, and sports field. This refinement allowed for a more precise classification. Finally, we discarded images without any viable landing zones (e.g., scenes entirely over water), making a dataset of 2,725 images, divided as follows: 2,168 for training, 279 for validation, and 278 for testing.

ICG Dataset [83]

is a collection of 400 real-world images, of resolution 6000×4000 pixels, captured by a drone operating at altitudes spanning from 5 to 30 meters. It focuses on a semantic understanding of urban scenes to increase autonomous drone flight safety and landing procedures. The images depict more than 20 houses from a nadir (bird's eye) view. The segmentation ground truth consists of 24 thematic classes. For the task of person detection, the dataset also contains bounding box annotations of the training and test set. We used all 400 images from this dataset for various experiments, allocating 320 for training, 40 for validation, and 40 for testing.

MidAir Dataset [63]

The Montefiore Institute Dataset of Aerial Images and Records (MidAir) is a synthetic dataset specifically designed to simulate low-altitude drone flights. This dataset offers extensive information, including RGB images of size 1024×1024 pixels, captured by multi-modal sensors. Additionally, it provides surface normal orientation, depth, object semantics, and stereo disparity.

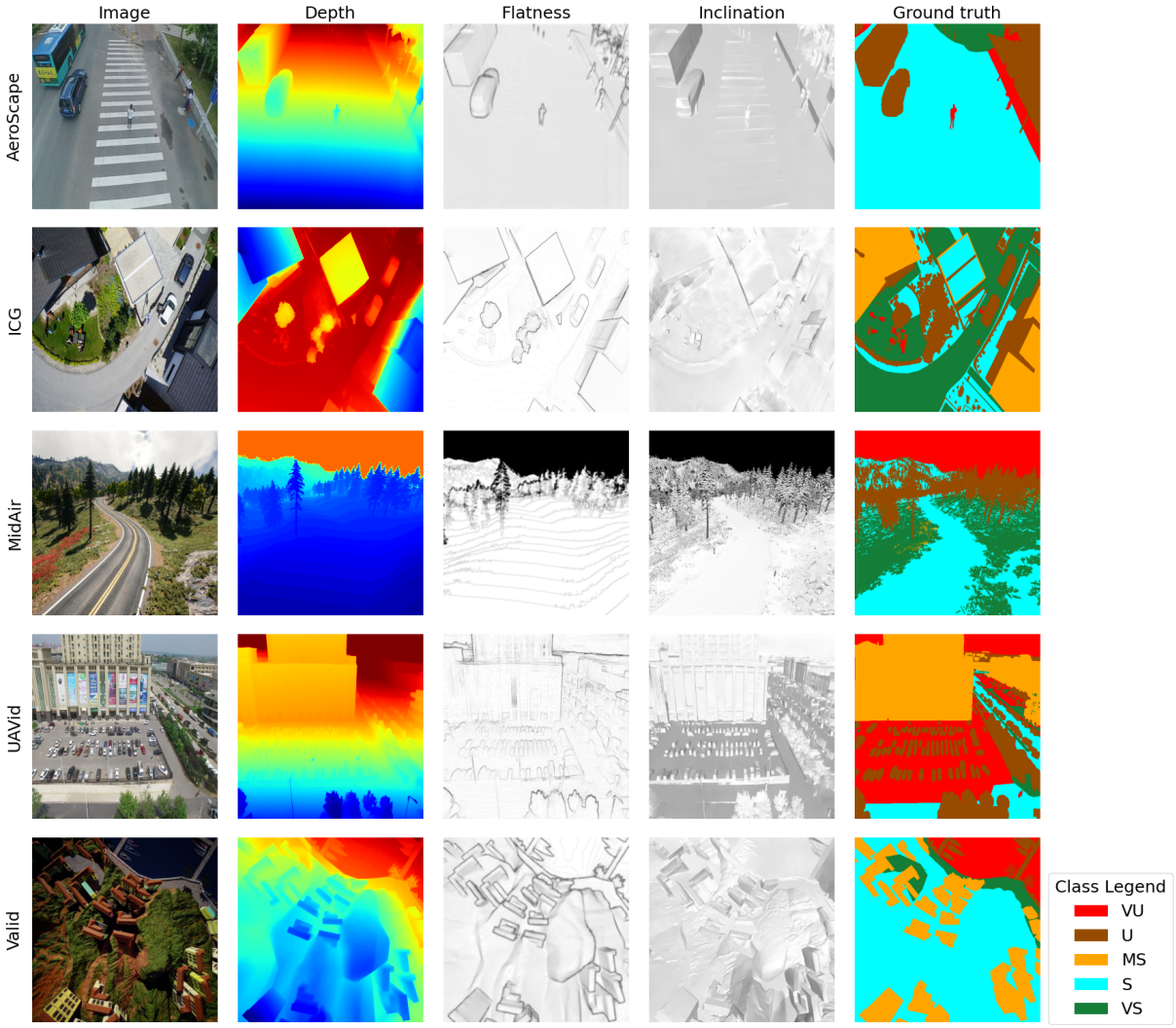


Figure 4.4: (a) Original image AeroScapes [155], ICG [83], MidAir [63], UAVid [136] (b) depth map (c) flatness map (d) inclination map (e) five-level safety ground truth.

The dataset has 420,000 training frames, divided into 54 distinct trajectories, each encompassing diverse weather and seasonal conditions. The dataset also offers three different environment maps, further increasing its versatility and applicability. For our model training and evaluation purposes, We selected a representative sample of 2,202 RGB images (with 1,761 for training, 221 for validation, and 220 for testing), each accompanied by its corresponding ground-truth, depth, and surface normal maps.

UAVid Dataset [136]

Consists of high-resolution 4K real-world images captured from videos recorded by drones in urban environments. The images portray streets from an oblique viewpoint and are specifically designed for semantic segmentation tasks. The dataset encompasses eight different object classes that need to be identified. For our purposes, we modified the original dataset split. We selected 820 images and randomly divided them into 653 for training, 86 for validation, and 81 for testing. We used the original ground truth masks to generate our five-level safety maps.

VALID Dataset [38]

Short for Virtual Aerial Image Dataset, provides a collection of synthetic images specifically created for segmentation tasks involving 30 distinct categories. The dataset comprises 6,690 images with a resolution of 1024×1024 pixels., generated in six virtual environments (Airport, Downtown, Mountain, Neighborhood, Night, and Seaside). Each environment includes images captured at three different altitudes (20, 50, and 100 meters), except for Night, which only comprises images at 20 meters. Additionally, the dataset encompasses various lighting conditions. Our work combined images from various environments and altitudes to create a comprehensive dataset for a single training phase. We randomly divided this dataset into 5,352 images for training, 669 for validation, and 669 for testing.

4.3.2 Model training and evaluation

In this section, we evaluate the performance of OR-SLZNet using the five distinct datasets previously introduced. All experiments were conducted on an NVIDIA station with a 48GB RTX 6000 Ada GPU, and a system with 128GB memory. Each model was trained for up to 200 epochs, allowing for thorough convergence and performance evaluation.

We evaluate the model performance using key metrics such as mean Intersection over Union (mIoU), Accuracy (Acc), Dice, and Mean Square Error (MSE) across training, validation, and test splits. While these metrics provide a general sense of the model effectiveness, they do not fully capture the critical safety aspect of our problem. In particular, mis-classifying Very Unsafe (rank 0) as Safe or Very Safe (ranks 3 or 4) is far more dangerous than the reverse, given the higher safety risks involved in such errors.

To better reflect these concerns, we use Asymmetric Mean Square Error (AMSE) in Eq. (4.3), which applies a higher penalty when the model predicts a higher safety rank than the actual rank. Specifically, we set $\alpha = 1$ and $\beta = 3$ to emphasize the cost of overestimating safety. We calculate AMSE for individual sub-tasks and as a total score across all classes. This ensures that even if the model's performance on conventional metrics is not superior, it focuses on minimizing AMSE, particularly for the Very Unsafe rank. Ensuring accurate detection of this rank is vital for making reliable UAV landing decisions, where any failure to avoid unsafe zones could lead to severe consequences. Note that if we discard the ordinal nature of the safety scores, the five classes can still be derived from a standard semantic segmentation model.

To demonstrate the performance and advantages of our proposed OR-SLZNet model, we developed a variant of it by replacing the ordinal regression with a segmentation head. This model integrates the photometric (RGB) and a geometric (DFI) features as in the OR-SLZNet model. That is, the RGB and DFI features go through their respective encoders, then decoded and concatenated in a similar way to OR-SLZNet. Finally, the produced features pass through a dual-layer segmentation head to produce the final output. This model is trained using cross-entropy (CE) loss.

The following sections outline our experiments, beginning with an ablation study that demonstrates the importance of different inputs and components for both SEG-SLZNet and OR-SLZNet. This analysis emphasizes the role of key components of our model. Next, we explore the impact of using an asymmetric loss function to prioritize correct predictions for critical safety classes, ensuring better performance in high-risk mis-classifications. Finally, to enable a more realistic scenario, we added a safety margin, labelled unsafe, around the very unsafe parts since this part can constitute some risk for a flying UAV. This adds a challenge for safety estimation by enabling to assess how well our model handles subtle safety boundaries compared to standard segmentation methods. Finally, we tested OR-SLZNet in hazy conditions to evaluate its robustness to adverse weather conditions.

Ablation study

As discussed in Sec. 4.2.2, OR-SLZNet integrates both photometric (RGB) and geometric (DFI) features to leverage rich visual information alongside spatial and structural cues. In this ablation study, we first demonstrate the effectiveness of the feature fusion for both SEG-SLZNet and OR-SLZNet. Then, we compare the performance of OR-

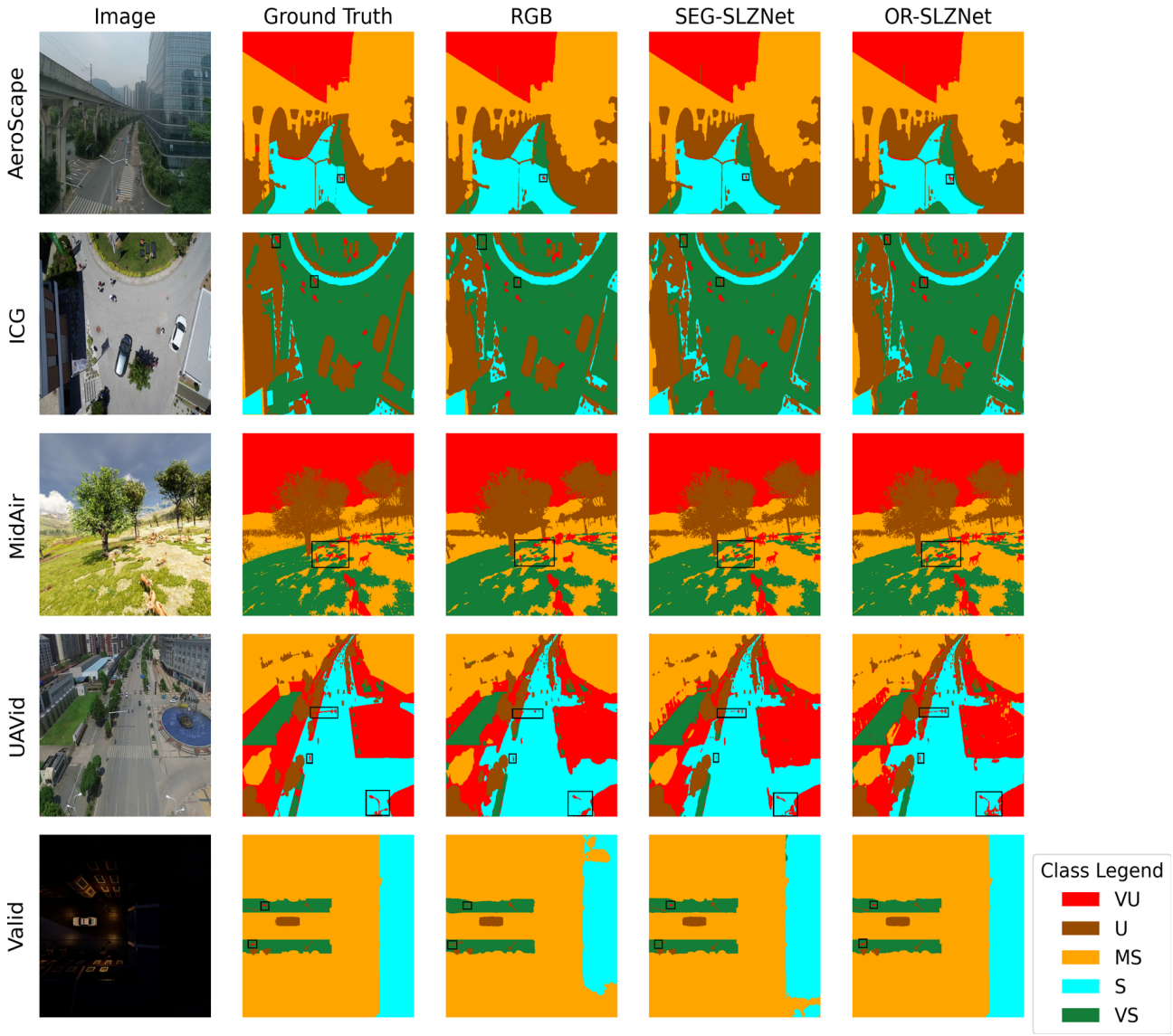


Figure 4.5: Illustrative Ablation Study Model Prediction Comparison

SLZNet and SEG-SLZNet using four overall evaluation metrics, as well as AMSE. For this experiment, we used only a symmetric version of our loss function (DICE+BCE), whereas for the SEG-SLZNet we used the cross-entropy (CE) loss. The comparative results across all five datasets are presented in Tables 4.2 to 4.3. The bold numbers in the tables indicate the direct comparison between OR-SLZ and Seg-SLZ: for each metric, the value corresponding to the better-performing method is shown in bold.

Note that except for MidAir where depth and surface normals were provided, the other datasets required to estimate these cues. Thus, MidAir provides an ideal baseline

Table 4.2: Ablation study on input features analyzing the performance of: (a) SEG-SLZNet and (b) the OR-SLZNet models.

Dataset	Loss	Input	Train				Validation				Test			
			mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE
AeroScape	CE	DFI	0.963	0.992	0.981	0.082	0.855	0.966	0.916	0.300	0.798	0.953	0.883	0.352
		RGB	0.969	0.994	0.984	0.073	0.955	0.991	0.977	0.106	0.953	0.990	0.976	0.118
		Fusion	0.973	0.994	0.986	0.063	0.957	0.991	0.978	0.101	0.956	0.991	0.977	0.112
ICG	CE	DFI	0.953	0.990	0.976	0.129	0.807	0.957	0.892	0.540	0.815	0.959	0.897	0.493
		RGB	0.949	0.989	0.974	0.152	0.866	0.971	0.926	0.386	0.868	0.972	0.929	0.396
		Fusion	0.959	0.992	0.979	0.116	**0.903**	**0.980**	**0.949**	0.258	0.900	0.979	0.947	0.266
MidAir	CE	DFI	0.901	0.979	0.948	0.147	0.886	0.975	0.939	0.177	0.871	0.971	0.928	0.213
		RGB	0.900	0.979	0.947	0.149	0.885	0.975	0.937	0.185	0.876	0.973	0.932	0.205
		Fusion	0.924	0.984	0.960	0.108	0.908	0.980	0.951	0.137	0.896	0.978	0.944	0.165
UAVid	CE	DFI	0.913	0.982	0.954	0.269	0.793	0.953	0.882	0.857	0.798	0.954	0.886	0.813
		RGB	0.930	0.985	0.964	0.209	0.854	0.968	0.921	0.518	0.854	0.969	0.921	0.49
		Fusion	0.954	0.991	0.977	0.131	0.860	0.970	0.924	0.501	0.861	0.970	0.925	0.508
Valid	CE	DFI	0.950	0.990	0.974	0.103	0.887	0.974	0.936	0.250	0.886	0.974	0.936	0.253
		RGB	0.949	0.989	0.974	0.110	0.933	0.986	0.965	0.149	0.928	0.985	0.962	0.158
		Fusion	0.967	0.993	0.983	0.069	0.962	0.992	0.981	0.081	0.96	0.992	0.979	0.083

(a)

Dataset	Loss	Input	Train				Validation				Test			
			mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE
AeroScape	DBCE	DFI	0.968	0.993	0.984	0.061	0.923	0.984	0.960	0.133	0.924	0.984	0.960	0.144
		RGB	0.972	0.994	0.986	0.057	0.957	0.991	0.977	0.100	0.954	0.991	0.976	0.112
		Fusion	**0.981**	**0.996**	**0.991**	**0.036**	0.957	0.991	0.978	**0.091**	0.954	0.991	0.977	**0.101**
ICG	DBCE	DFI	0.927	0.985	0.962	0.162	0.790	0.953	0.883	0.516	0.795	0.954	0.885	0.518
		RGB	0.959	0.992	0.979	0.100	0.862	0.970	0.926	0.365	0.867	0.971	0.929	0.364
		Fusion	**0.960**	0.992	**0.980**	**0.103**	0.901	0.979	0.948	**0.255**	**0.901**	0.979	**0.948**	**0.250**
MidAir	DBCE	DFI	0.908	0.981	0.951	0.126	0.888	0.976	0.941	0.163	0.878	0.974	0.935	0.187
		RGB	0.905	0.980	0.950	0.127	0.889	0.976	0.941	0.159	0.876	0.973	0.933	0.189
		Fusion	**0.929**	**0.985**	**0.963**	**0.091**	0.908	**0.981**	**0.952**	**0.125**	**0.897**	0.978	**0.946**	**0.153**
UAVid	DBCE	DFI	0.912	0.982	0.954	0.211	0.787	0.952	0.881	0.756	0.791	0.953	0.883	0.759
		RGB	0.921	0.983	0.959	0.197	0.855	0.969	0.922	0.470	0.854	0.969	0.921	0.490
		Fusion	**0.964**	**0.993**	**0.982**	**0.075**	**0.867**	**0.972**	**0.929**	**0.441**	**0.862**	**0.970**	**0.926**	**0.470**
Valid	DBCE	DFI	0.951	0.990	0.975	0.091	0.938	0.987	0.968	0.110	0.931	0.986	0.964	0.118
		RGB	0.951	0.990	0.975	0.097	0.947	0.989	0.973	0.107	0.942	0.988	0.970	0.113
		Fusion	**0.969**	**0.994**	**0.984**	**0.057**	**0.964**	**0.993**	**0.982**	**0.068**	**0.961**	**0.992**	**0.98**	**0.073**

(b)

Table 4.3: Comparison Between SEG-SLZNet and OR-SLZNet Using AMSE Metric Across Safety Classes for Different Datasets

Dataset	Model	Train						Validation						Test					
		VU	U	MS	S	Vs	Total	VU	U	MS	S	Vs	Total	VU	U	MS	S	Vs	Total
AeroScape	**SEG-SLZNet**	0.616	0.097	0.055	0.032	0.080	0.131	0.969	**0.184**	0.160	0.050	0.197	0.247	1.448	0.158	0.231	**0.050**	0.180	0.291
	OR-SLZNet	**0.297**	**0.082**	**0.048**	**0.023**	**0.033**	**0.075**	**0.805**	0.186	**0.115**	**0.046**	**0.132**	**0.210**	**1.283**	**0.137**	**0.143**	0.051	**0.122**	**0.247**
ICG	**SEG-SLZNet**	0.595	0.562	0.018	**0.211**	0.087	0.210	1.723	**1.405**	**0.079**	**0.534**	0.165	**0.493**	1.411	1.583	**0.053**	**0.485**	0.160	0.538
	OR-SLZNet	**0.442**	**0.510**	**0.018**	0.229	**0.075**	**0.188**	**1.598**	1.559	0.095	0.542	**0.146**	0.512	**1.335**	**1.547**	0.056	0.499	**0.141**	**0.518**
MidAir	**SEG-SLZNet**	0.104	0.284	0.794	0.371	0.089	0.263	0.116	0.312	0.805	0.427	0.094	0.277	0.140	0.301	1.003	0.364	0.103	0.309
	OR-SLZNet	**0.074**	**0.244**	**0.525**	**0.268**	**0.081**	**0.193**	**0.095**	**0.287**	**0.713**	**0.409**	0.094	**0.252**	**0.107**	**0.278**	**0.901**	**0.326**	**0.097**	**0.277**
UAVid	**SEG-SLZNet**	0.747	0.225	0.031	0.131	0.339	0.271	2.652	0.932	0.096	0.379	1.233	1.014	2.490	1.012	0.089	0.339	1.530	0.977
	OR-SLZNet	**0.452**	**0.213**	**0.027**	**0.127**	**0.317**	**0.208**	**2.193**	**0.841**	**0.080**	**0.378**	**1.292**	**0.900**	**2.166**	**0.907**	**0.071**	**0.347**	**1.523**	**0.885**
Valid	**SEG-SLZNet**	0.322	0.410	0.020	0.072	0.109	0.118	0.455	0.535	0.031	0.082	0.143	0.153	0.443	0.549	0.036	0.089	0.145	0.161
	OR-SLZNet	**0.210**	**0.391**	**0.020**	**0.024**	**0.069**	**0.100**	**0.290**	**0.467**	0.031	**0.032**	**0.077**	**0.129**	**0.316**	**0.483**	0.036	**0.084**	**0.125**	**0.139**

for testing the combination of photometric and geometric inputs. As shown in Tables 4.2.(a) and 4.2.(b), both SEG-SLZNet and OR-SLZNet achieved comparable performance for the DFI and RGB features. In contrast, for other datasets, the RGB results were generally superior. Although other datasets lack ideal geometric maps, the results still demonstrate that for both SEG-SLZNet and OR-SLZNet, fusing low-level photometric and geometric features improves performance. While the improvement is not drastic

in some cases, such as with AeroScape, adding geometric information consistently leads to better results.

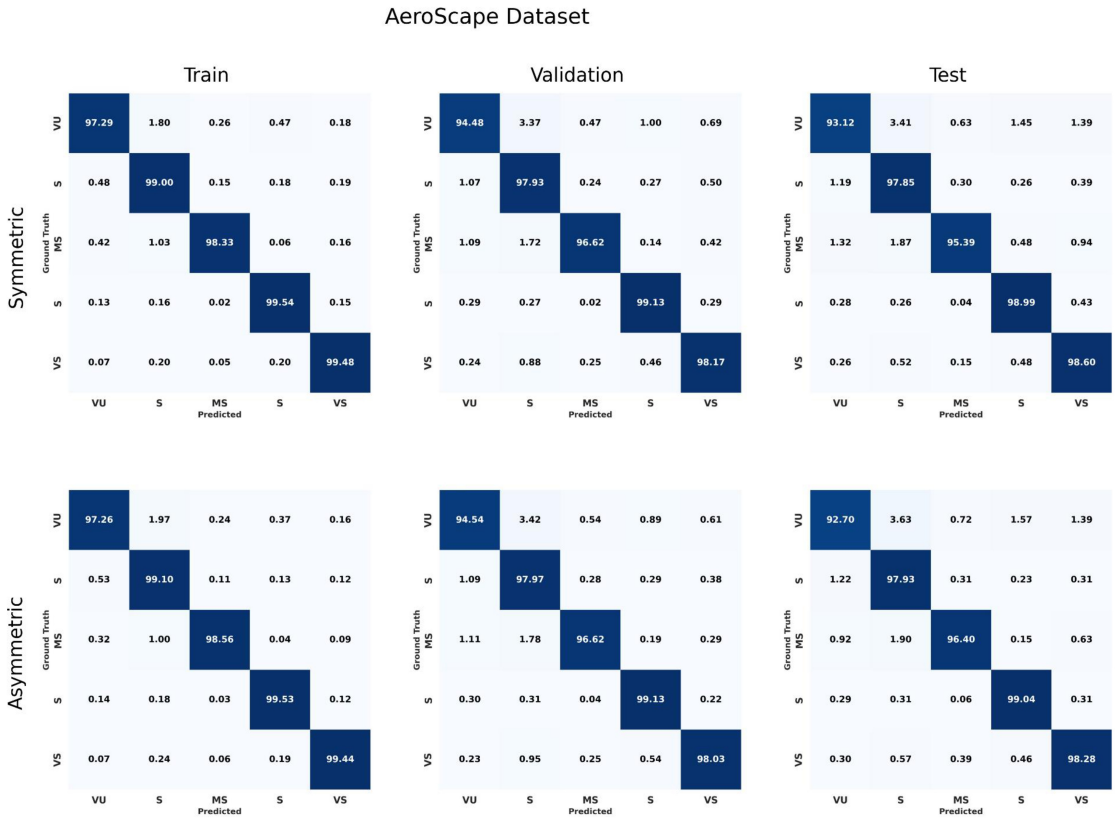
A key part of our ablation study is the comparison between the fused SEG-SLZNet and OR-SLZNet models. Across almost all datasets, our proposed OR-SLZNet consistently outperforms SEG-SLZNet in terms of mIoU, Accuracy, and Dice metrics. Notably, in the AeroScape dataset, the performance of both models is comparable, while in the ICG dataset, SEG-SLZNet shows a slight advantage in the validation split. However, across all datasets, for the MSE metric, OR-SLZNet consistently performs better. Since the MSE metric accounts for the difference between the predicted rank and the actual rank, this demonstrates that the ordinal regression approach of OR-SLZNet provides a more accurate ranking of safety levels.

Table 4.3 shows AMSE metric per rank and for overall ranks for SEG-SLZNet and OR-SLZNet. In all the datasets, across all train, validation, and test splits, the AMSE of the very unsafe class for OR-SLZNet is lower than for SEG-SLZNet. This suggests that OR-SLZNet is more effective in distinguishing between safety levels, especially in critical very unsafe regions usually characterized with a high risk for hitting obstacles. In Tables 4.2 to 4.3, instances where OR-SLZNet outperforms SEG-SLZNet are highlighted in bold. The visualizations of the predictions from SEG-SLZNet and OR-SLZNet are shown in Figure 4.5.

Asymmetric Loss Function

In Section 4.2.5, we introduced our innovative asymmetric loss function, specifically designed to enforce rank consistency among classifiers. This method serves as a powerful mechanism for guiding the model toward more accurate predictions by imposing greater penalties for mis-classifications involving critical classes. The asymmetric loss function emphasizes safety by penalizing errors where more hazardous very unsafe parts are incorrectly classified as safer categories. This prioritization pushes the model to exhibit more reliable and precise performance in high-stakes situations. By harnessing this loss function, our OR-SLZNet model shows a distinct advantage over the Seg-SLZNet model, particularly in its capacity to fine-tune predictions to privilege higher accuracy scores for very unsafe levels. In contrast, standard segmentation models lack this tailored enforcement mechanism, which complicates the prioritization of safety-critical outcomes.

Tables 4.4 and 4.5 present the results of our asymmetric loss function in relation to overall performance metrics and AMSE. The results indicate that, while the metrics



Asymmetric loss weights: [25., 5., 5., 5.]

Figure 4.6: Confusion matrices for the AeroScape dataset across training, validation, and test splits, comparing the performance of symmetric and asymmetric loss functions.

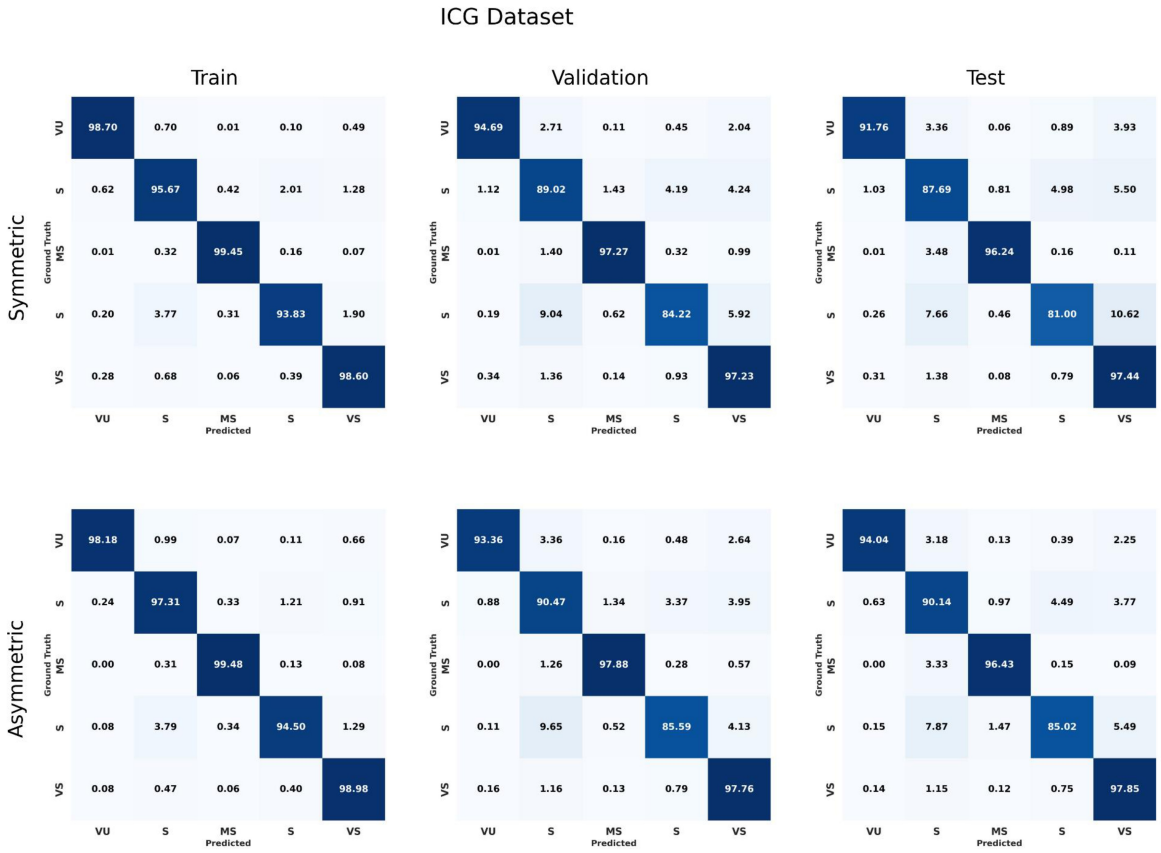


Figure 4.7: Confusion matrices for the ICG dataset across training, validation, and test splits, comparing the performance of symmetric and asymmetric loss functions.

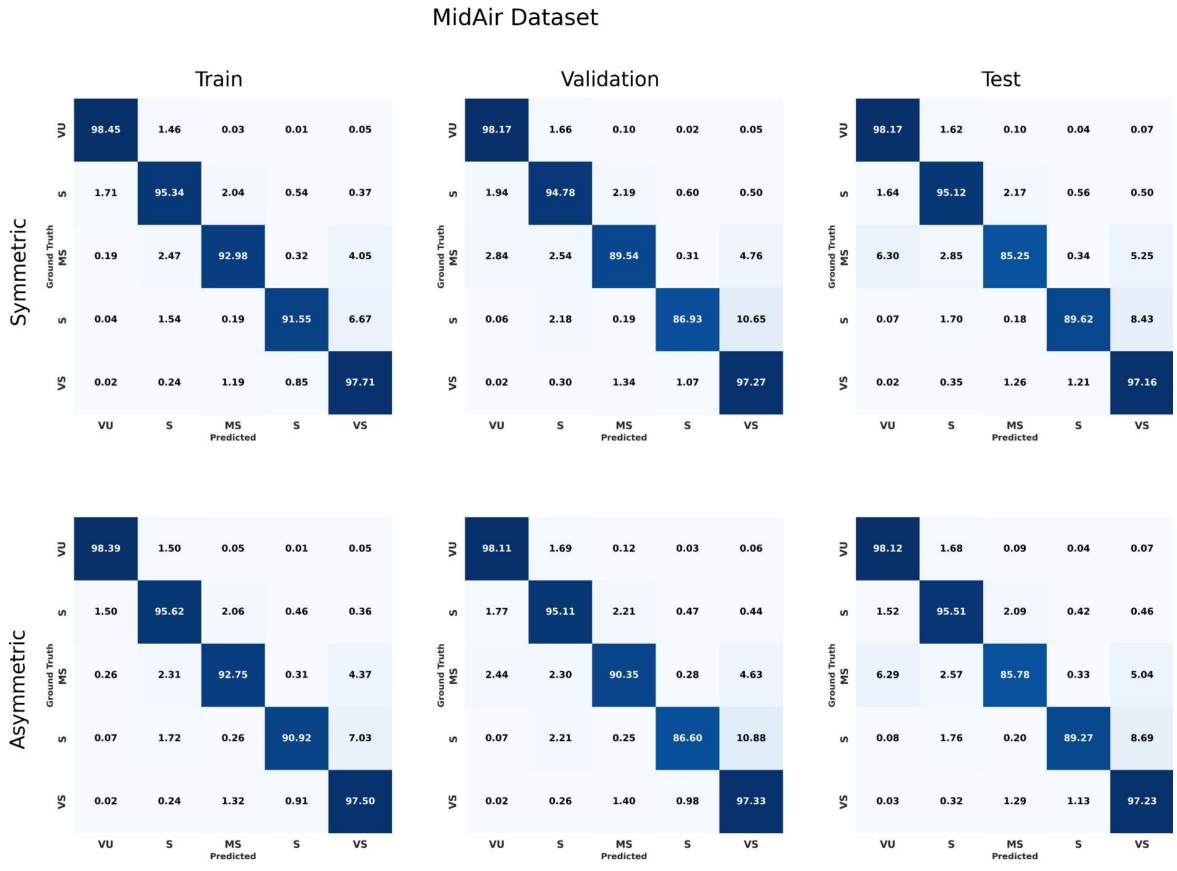


Figure 4.8: Confusion matrices for the MidAir dataset across training, validation, and test splits, comparing the performance of symmetric and asymmetric loss functions.

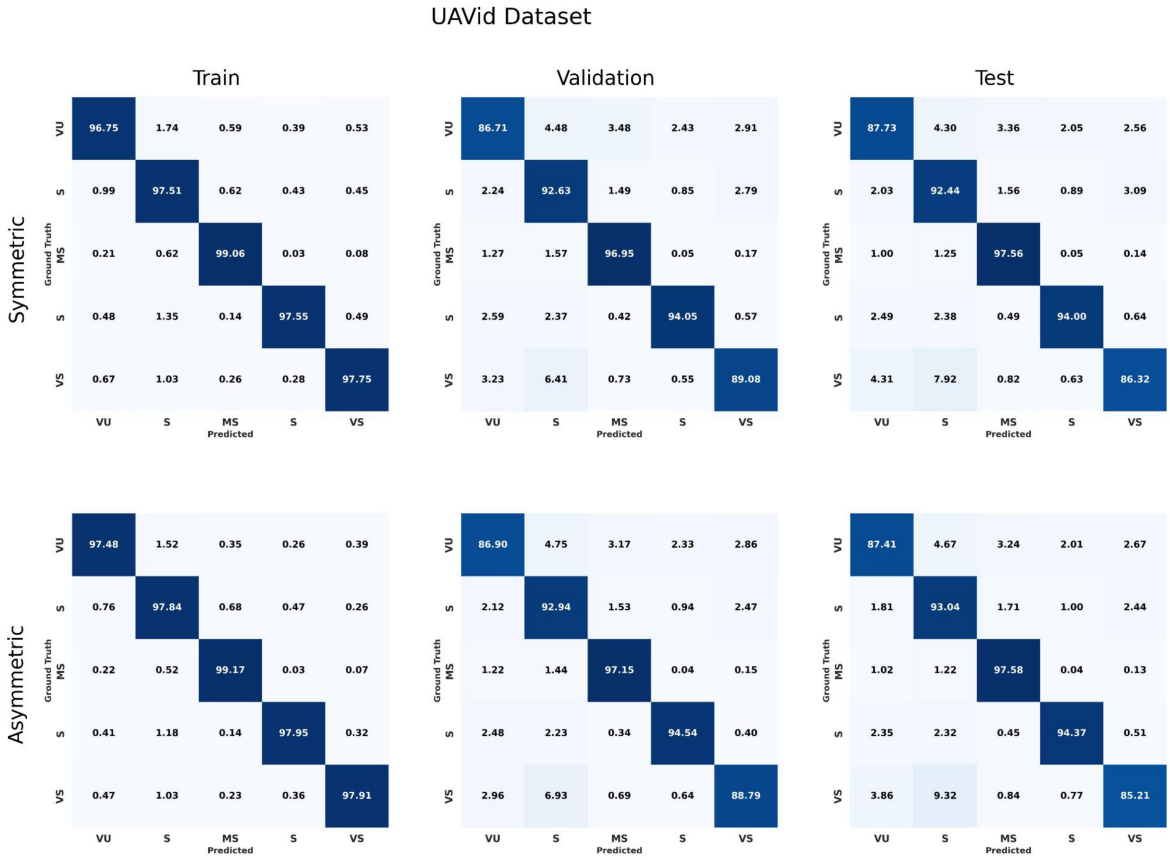


Figure 4.9: Confusion matrices for the UAVid dataset across training, validation, and test splits, comparing the performance of symmetric and asymmetric loss functions.

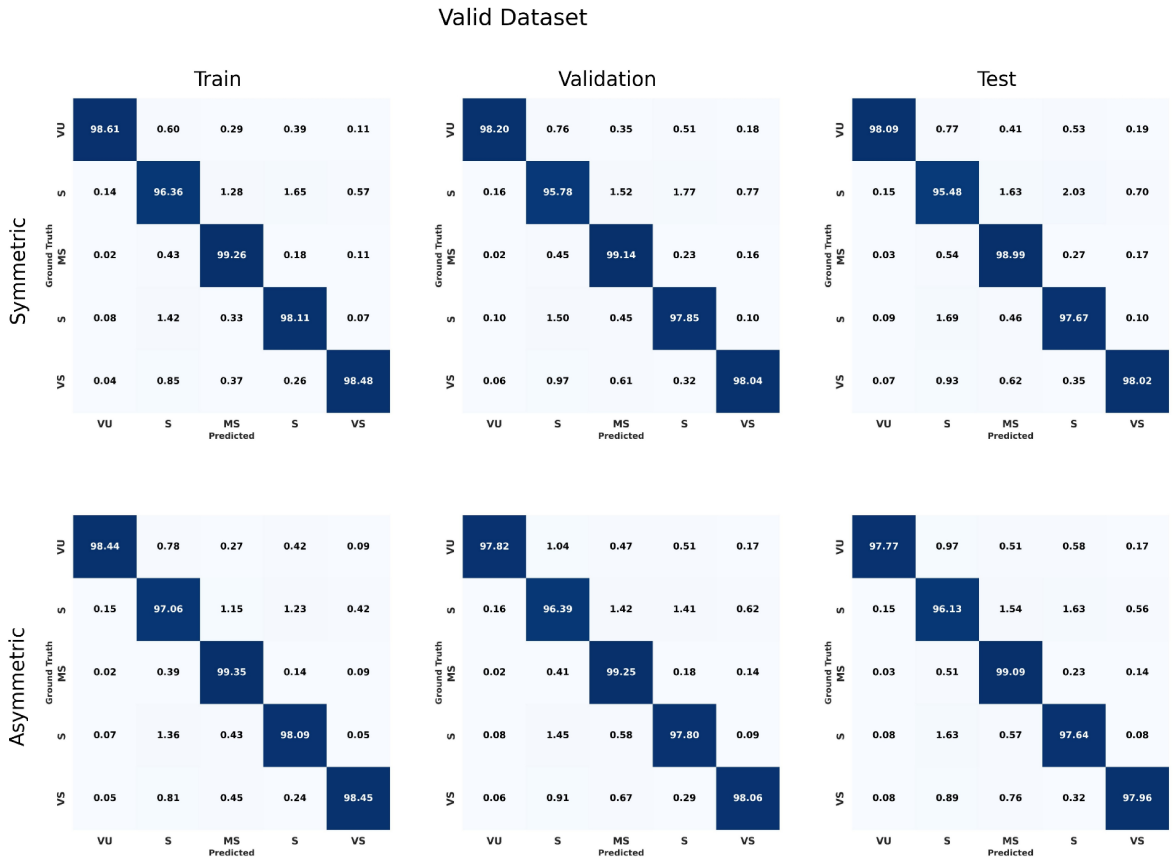


Figure 4.10: Confusion matrices for the Valid dataset across training, validation, and test splits, comparing the performance of symmetric and asymmetric loss functions.

Table 4.4: Evaluation of OR-SLZNet Performance Using the Consistent Asymmetric Loss Function.

Dataset	Weight	Train				Validation				Test			
		mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE
AeroScape	**[25., 5., 5., 5.]**	0.982	0.996	0.991	**0.033**	0.957	0.991	0.978	**0.094**	**0.956**	0.991	0.977	0.101
ICG	**[50., 5., 5., 5.]**	**0.964**	**0.993**	**0.982**	**0.088**	**0.903**	**0.980**	**0.949**	**0.250**	0.900	0.979	0.947	**0.253**
MidAir	**[10., 25., 5., 10.]**	0.926	0.985	0.962	0.097	**0.91**	0.981	**0.953**	0.126	**0.899**	**0.979**	**0.947**	**0.150**
UAVid	**[25., 10., 10., 10.]**	0.964	0.993	0.982	0.077	0.857	0.969	0.923	0.482	0.862	0.970	0.926	0.470
Valid	**[10., 25., 5., 10.]**	**0.971**	0.994	**0.985**	**0.052**	**0.966**	0.993	**0.983**	**0.063**	**0.963**	**0.992**	**0.981**	**0.067**

Table 4.5: Evaluation of Segmentation Model and Inference on different examples

Dataset	Weight	Train						Validation						Test					
		VU	U	MS	S	Vs	Total	VU	U	MS	S	Vs	Total	VU	U	MS	S	Vs	Total
AeroScape	**[25., 5., 5., 5.]**	**0.688**	**0.156**	**0.104**	**0.047**	**0.141**	**0.186**	**0.712**	**0.179**	**0.101**	**0.044**	0.136	**0.195**	**1.194**	0.159	**0.199**	0.050	**0.100**	**0.243**
ICG	**[50., 5., 5., 5.]**	**0.383**	**0.405**	**0.016**	**0.200**	**0.061**	**0.154**	**1.515**	**1.520**	**0.090**	**0.525**	**0.143**	**0.498**	**1.298**	1.592	**0.048**	0.508	**0.138**	0.525
MidAir	**[10., 25., 5., 10.]**	0.079	**0.229**	0.567	0.288	0.087	0.203	0.099	**0.258**	**0.685**	0.423	0.093	**0.244**	**0.104**	**0.253**	**0.870**	0.343	**0.096**	**0.267**
UAVid	**[25., 10., 10., 10.]**	**0.347**	**0.153**	**0.023**	**0.096**	**0.181**	**0.149**	**2.191**	**0.829**	0.080	**0.333**	1.117	**0.947**	**2.159**	**0.841**	**0.070**	**0.328**	**1.495**	0.880
Valid	**[10., 25., 5., 10.]**	0.212	0.296	0.020	0.024	**0.067**	0.102	0.303	**0.381**	**0.027**	**0.074**	**0.121**	**0.115**	**0.312**	**0.391**	**0.031**	**0.080**	0.126	**0.124**

mIoU, accuracy, Dice, and MSE remain relatively stable in both validation and test stages, the AMSE for the very unsafe level is consistently reduced when using the asymmetric loss. The enhancements range from 19% to over 30% across different datasets, demonstrating the ability of asymmetric loss function to effectively mitigate classification errors in very unsafe areas. In addition, Figures 4.6, 4.7,4.8,4.9 and 4.10 present the confusion matrices for different datasets across training, validation, and test splits, comparing the performance of symmetric and asymmetric loss functions, with a particular focus on improving the identification of Very Safe regions.

Finally, note that forcing the model to classify asymmetrically can raise important considerations regarding weight selection and the trade-offs involved in sacrificing general performance for improved classification of specific classes. The selection of weights can be influenced by imbalanced sampling and the defined safety concerns, particularly for very unsafe classes, such as humans, small vehicles, and wildlife, which are typically rare in UAV-collected datasets compared to more prevalent categories like grass and paved areas. As a result, it is essential to prioritize these critical regions for accurate classification. However, in certain scenarios such as identifying the safest landing spot, where small objects or markers may be present classifying the Very Safe class becomes equally crucial. Fortunately, our loss function allows for flexible weight selection, enabling the model to intensify its prediction focus according to safety priorities.

Safety Margin

To enhance the safety representation of our data, we introduced a safety margin surrounding the very unsafe parts (Rank 0) in the ground truth of the ICG and UAVid datasets. This approach reflects a realistic scenario where the vicinity of very unsafe

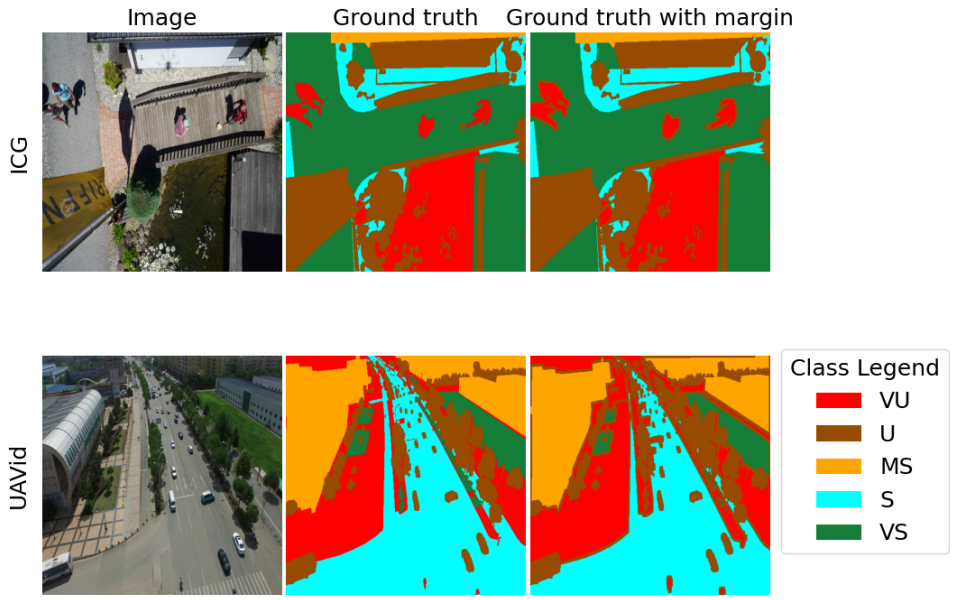


Figure 4.11: Example of Safety Margin Around Very Unsafe Class

areas pose a collision risk, especially for moving objects. Given that our masks consist of discrete ranks, we designated this margin as the unsafe class (Rank 1). For the ICG dataset, we conducted experiments using masks with safety margins of 50 and 70 pixels, whereas for the UAVid dataset, we used only a 50-pixel margin. Figure 4.11 presents some examples on ICG and UAVid along with the original masks and the masks integrating the safety margin. In Tables 4.6 and 4.7 the results for margin inclusion on SEG-SLZNet and OR-SLZNet are provided.

In evaluating the impact of margin, a distinct trade-off emerges between the overall model performance and the prediction of the very unsafe (VU) rank. We can remark that OR-SLZNet consistently outperforms SEG-SLZNet, both with and without margins (as shown in the fusion ablation study) for predicting the VU rank, manifested by lower AMSE values. Specifically, as we can observe from Tables 4.6 and 4.7, for both the 50-pixel and 70-pixel margins, OR-SLZNet achieves lower AMSE than SEG-SLZNet, even in its fusion configuration without margins (refer to Table 4.3). This indicates that margin inclusion effectively captures more nuanced unsafe regions.

For the overall performance, mIoU, accuracy, Dice, and MSE, both models generally perform better without margin, where the complexity and noise are reduced. This suggests that while the OR-SLZNet yields superior overall results in terms of MSE, accuracy, and Dice, the inclusion of margins enhances the models capacity to detect the

Table 4.6: Comparison of SEG-SLZNet and OR-SLZNet with Margin Inclusion

Dataset	Model	Loss	Margin	Train				Validation				Test			
				mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE
ICG	**SEG-SLZNet**	**CE**	**50**	0.955	0.991	0.977	0.107	0.902	0.979	0.949	0.237	0.900	0.979	0.947	0.239
	OR-SLZNet	**DBCE**	**50**	**0.957**	0.991	**0.978**	**0.096**	0.900	0.979	0.947	0.241	0.899	0.979	0.947	0.245
	SEG-SLZNet	**CE**	**70**	0.957	0.991	0.978	0.105	0.900	0.979	0.947	0.250	0.899	0.979	0.947	0.251
	OR-SLZNet	**DBCE**	**70**	0.955	0.991	0.977	**0.101**	0.896	0.978	0.945	0.250	0.898	0.978	0.946	**0.248**
UAVid	**SEG-SLZNet**	**CE**	**50**	0.947	0.989	0.973	0.077	0.821	0.961	0.901	0.386	0.825	0.962	0.904	0.398
	OR-SLZNet	**DBCE**	**50**	**0.959**	**0.992**	**0.979**	**0.05**	**0.830**	**0.963**	**0.905**	**0.347**	**0.829**	**0.963**	**0.906**	**0.385**

Table 4.7: Comparison of SEG-SLZNet and OR-SLZNet Using AMSE Metric with Margin Inclusion Across All Safety Categories

Dataset	Model	Loss	Margin	Train						Validation						Test					
				VU	U	MS	S	Vs	Total	VU	U	MS	S	Vs	Total	VU	U	MS	S	Vs	Total
ICG	**SEG-SLZNet**	**CE**	**50**	0.091	0.661	0.016	0.159	0.079	0.201	0.490	1.434	0.057	0.421	0.159	0.455	0.411	1.477	0.052	0.381	0.155	0.476
	OR-SLZNet	**DBCE**	**50**	**0.050**	**0.619**	0.016	0.181	**0.061**	**0.183**	**0.323**	1.687	0.106	0.441	**0.135**	0.497	**0.308**	1.731	**0.046**	0.411	**0.136**	0.521
	SEG-SLZNet	**CE**	**70**	0.065	0.522	0.012	0.174	0.089	0.180	0.387	1.367	0.060	0.451	0.186	0.460	0.304	1.486	0.048	0.395	0.174	0.490
	OR-SLZNet	**DBCE**	**70**	**0.043**	**0.482**	0.019	0.186	**0.057**	**0.154**	**0.259**	1.715	0.083	0.493	**0.143**	0.513	**0.269**	1.691	0.050	0.472	**0.141**	0.524
UAVid	**SEG-SLZNet**	**CE**	**50**	0.111	0.209	0.018	0.113	0.268	0.14	1.052	0.923	0.063	1.369	0.37	0.743	0.917	0.932	0.056	0.383	1.8	0.73
	OR-SLZNet	**DBCE**	**50**	**0.086**	**0.154**	**0.014**	**0.061**	**0.145**	**0.097**	**0.927**	**0.782**	**0.062**	**0.326**	1.337	**0.646**	**0.737**	**0.927**	**0.059**	0.305	**1.670**	**0.669**

very unsafe rank. In this regard, OR-SLZNet consistently outperformed SEG-SLZNet in both margin and non-margin settings, with margins further narrowing the performance gap between the two models.

As we can observe from Table 4.7, the ICG dataset particularly benefits from margin inclusion, with OR-SLZNet demonstrating improved VU AMSE values (e.g., 0.308 with a 50-pixel margin and 0.269 with a 70-pixel margin compared to 1.448 without margins) while maintaining stable overall performance. Conversely, UAVid faces more challenges due to the added complexity, with global metrics such as accuracy, Dice, and MSE showing some decline. Although the 50-pixel margin offers a slight improvement in VU AMSE for UAVid, the fusion method (without margins) delivers better overall results in this dataset. This indicates that while margins enhance performance in ICG, balancing precision for unsafe regions, they have a more detrimental impact on global performance in UAVid. Notably, the VU class in UAVid largely consists of background classes, while ICG presents a more challenging dataset where adding margins around VU areas proves to be more effective.

Haze Effect

To evaluate OR-SLZNet in adverse weather conditions, we need to introduce weather variations since, except for MidAir and Valid, all other datasets were captured in clear weather conditions. To test the behaviour of our model in hazy conditions, we augmented a proportion of the images in ICG and AeroScape with haze which is generated through

the atmospheric scattering model (ASM) [141]:

$$I(x, y) = J(x, y)T(x, t) + A(1 - T(x, y)), \tag{4.9}$$

which gives the hazed image I from the clear version of it J, where A is the luminance and T is the transmittance approximated by the expression $T = \exp(-\beta D)$, with D is the depth map of the image and β is the scattering coefficient. This transmission map controls how much light is scattered based on depth, mimicking the real-world phenomenon where distant objects appear progressively hazier due to increased scattering. In our implementation, we set $\beta = 2.$ and $A = [1, 1, 1]$ which introduced a significant haze in images (see Figure 4.12 for illustration). We randomly replaced 25% of images with their hazed version in each of the training, validation, and test sets for both datasets.

Table 4.8: Evaluation of OR-SLZNet Using AMSE Metric with Margin Inclusion Across All Safety Categories

Dataset	Model	Loss	% Haze	Train						Validation						Test					
				VU	U	MS	S	Vs	Total	VU	U	MS	S	Vs	Total	VU	U	MS	S	Vs	Total
AeroScape	SEG-SLZNet	CE	25	0.625	0.173	0.144	0.029	0.061	0.152	0.907	0.253	0.253	0.044	0.135	0.249	1.321	0.224	0.269	0.048	0.129	0.285
	OR-SLZNet	DBCE	25	0.349	0.074	0.044	0.024	0.049	0.083	0.757	0.192	0.133	0.047	0.145	0.206	1.243	0.153	0.171	0.047	0.121	0.241
ICG	SEG-SLZNet	CE	25	0.428	0.547	0.010	0.134	0.103	0.201	1.382	1.612	0.080	0.445	0.205	0.538	1.214	1.840	0.057	0.411	0.185	0.595
	OR-SLZNet	DBCE	25	0.222	0.553	0.016	0.181	0.308	0.149	0.852	1.591	0.203	0.621	0.223	0.560	0.794	1.489	0.072	0.632	0.217	0.544

Table 4.9: Comparison of SEG-SLZNet and OR-SLZNet with Haze Effect

Dataset	Model	Loss	% Haze	Train				Validation				Test			
				mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE	mIoU	Acc	Dice	MSE
AeroScape	SEG-SLZNet	CE	25	0.971	0.994	0.985	0.069	0.951	0.990	0.975	0.116	0.951	0.990	0.975	0.129
	OR-SLZNet	DBCE	25	0.979	0.996	0.989	0.042	0.954	0.991	0.977	0.102	0.955	0.991	0.977	0.107
ICG	SEG-SLZNet	CE	25	0.960	0.992	0.979	0.115	0.893	0.977	0.943	0.282	0.893	0.977	0.943	0.290
	OR-SLZNet	DBCE	25	0.944	0.989	0.971	0.155	0.884	0.975	0.938	0.304	0.887	0.976	0.940	0.296

As expected, the overall performance decreased when haze images replaced original images in both datasets. Table 4.9 shows that the performance drop was more pronounced in ICG dataset for both SEG-SLZNet and OR-SLZNet, compared to AeroScape Dataset. The impact of haze inclusion in the input images for AeroScape was less significant than in ICG, yet OR-SLZNet consistently outperformed SEG-SLZNet in the ICG dataset. However, the trade-off between overall performance and specifically for very unsafe rank remains evident under the haze effect. In this regard, OR-SLZNet demonstrates a better ASME for the VU class compared to SEG-SLZNet. This indicates that even in haze-affected scenes, where RGB information may be less informative, our proposed OR-SLZNet, which utilizes more than just color features, proves to be significantly more effective in detecting VU instances. The visualizations of the predictions from SEG-SLZNet and OR-SLZNet are shown in Figure 4.12.

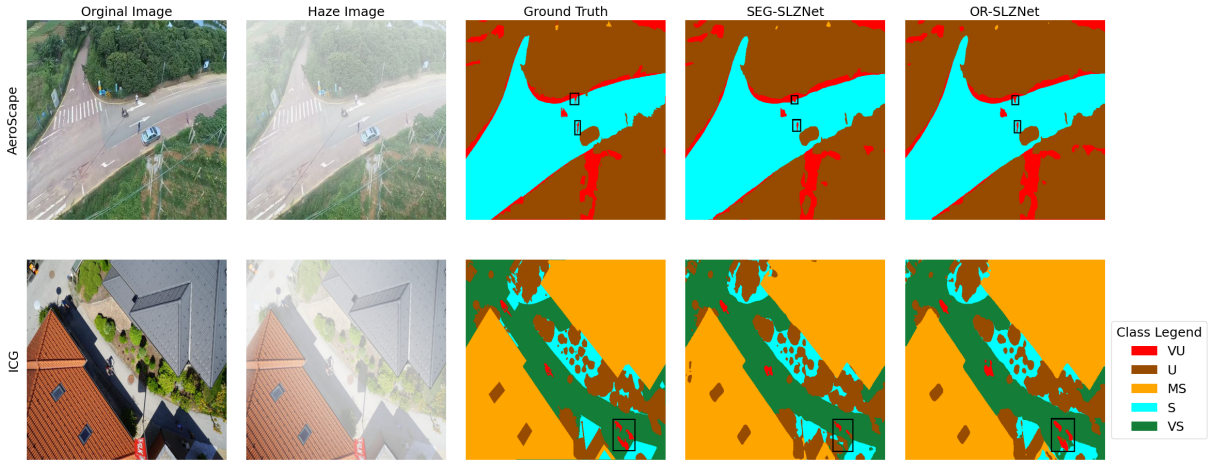


Figure 4.12: Haze prediction for SEG-SLZNet and OR-SLZNet

4.4 Discussion

OR-SLZNet relies solely on visual inputs, making it suitable for GPS-denied or communication-limited environments. Though designed for static images, it supports frame-by-frame inference on UAV video feeds, enabling real-time SLZ detection. Real-world tests confirm its effectiveness in such settings. However, temporal inconsistencies may arise from motion blur or scene changes. To address this, future work will integrate SLAM or visual-inertial odometry for temporal stability [189], and incorporate object tracking to monitor moving hazards (e.g., humans, animals, vehicles), ensuring continuous situational awareness without GPS. Also, while OR-SLZNet is effective against moderate noise, severe degradation due to adverse weather conditions remains challenging. Future work will explore uncertainty-aware prediction, and more advanced photometric and geometric augmentations (e.g., using generative models) to boost model reliability.

Energy efficiency is critical for deployment on resource-constrained UAVs [143]. OR-SLZNet is lightweight, with a low inference time ($\tilde{0}$.02s/frame), making it suitable for real-time use on embedded systems. This reduces processing load and extends flight time, especially during long missions or emergency landings. OR-To enhance this capability, future work will explore dynamic inference scheduling (e.g., activating the model only at low altitudes or during anomalies), as well as compression techniques like pruning, quantization, and knowledge distillation [119]. Adaptive resolution processing based on flight speed, altitude, or scene complexity will also be considered to further optimize computation.

A promising direction is multi-agent UAV systems, where cooperative SLZ detection improves coverage, energy efficiency, and fault tolerance. By sharing intermediate outputs, such as partial safety maps, obstacle detections, or uncertainty scores, drones can reduce redundancy and make better collective decisions [7, 207]. Task specialization is a key advantage: lightweight drones can perform quick, low-resolution scans, while more capable ones handle detailed SLZ mapping, segmentation, or dynamic object detection. This compute-aware allocation can optimize drone roles based on hardware and battery levels. In disaster response, for example, high-end drones can track moving hazards while others can focus on terrain mapping or localization [162].

Finally, inter-drone communication can enable enable SLZ map fusion, ensuring spatial consistency and resolving conflicts from occlusions or sensor noise. It can also support dynamic reconfiguration, allowing drones to adapt to failures or changing conditions by reallocating tasks. In complex environments (e.g., urban or forested areas), more drones can be assigned for detailed analysis. Additionally, integrating multi-agent reinforcement learning (MARL) offers a promising approach for coordinated landing decisions, enabling UAVs to learn optimal policies through interactions. This allows adaptive, context-aware site selection while accounting for obstacles, energy limits, and mission priorities, enhancing system resilience and autonomy in real-world scenarios.

4.5 Conclusions

In this chapter, we proposed a novel model for SLZ detection in UAV applications using deep ordinal regression networks. The model generates accurate, dense safety maps by leveraging multimodal information, including color, depth, and local terrain geometry. We validated its performance on diverse datasets with both vertical (nadir) and oblique views, covering urban and natural environments. Results highlight the model strong potential for improving UAV navigation automation and safety. Comparative evaluations against state-of-the-art segmentation methods further confirmed its effectiveness. Although validated on 2D aerial data, the predicted SLZ maps are readily applicable for UAV piloting assistance and 3D trajectory planning. Moreover, a lightweight version of the model can support efficient onboard deployment, enabling real-time decision-making in single or multi-UAV operations.

Chapter 5

A Framework for SLZ Mapping for UAVs in Dynamic Environments

5.1 Introduction

As UAV deployment increases in real-world environments marked by continuous movement and unpredictability, the concept of dynamic SLZs has become essential. Unlike static SLZs that rely on fixed terrain analysis, dynamic SLZs must accommodate evolving scene contexts where previously safe areas may become hazardous in seconds. Urban intersections, construction sites, and post-disaster zones are typical examples where transient obstacles such as pedestrians, vehicles, and debris—frequently obstruct potential landing surfaces. In such cases, especially during emergencies like equipment failures or signal loss, UAVs must rely on autonomous systems capable of perceiving environmental changes and making time-sensitive decisions to ensure safe landings [18, 53, 197].

Identifying dynamic SLZs thus presents a complex and multi-dimensional challenge [222]. Beyond detecting flat, obstacle-free regions suitable for landing [40], systems must also assess the location, behavior, and predicted trajectories of moving objects, including humans, vehicles, bicycles, and animals. Static analysis alone is insufficient—areas initially deemed safe can quickly become risky due to the motion of surrounding agents [191]. A robust SLZ detection framework must therefore integrate static scene segmentation with real-time object tracking and motion forecasting to support adaptive, context-aware decisions [91, 153, 231]. Furthermore, accounting for uncertainties in motion prediction is crucial; by modeling these uncertainties explicitly, the system can

better evaluate landing risks and avoid unsafe outcomes under variable lighting and obstacle dynamics [180, 231].

Traditional methods for SLZ identification often rely on static segmentation techniques [1, 168, 101], predefined safe zones, or specific visual landing markers. While these approaches can be effective in controlled or less dynamic environments, they struggle to adapt to the complexities of real-world, ever-changing scenes. In particular, methods that classify thematic surface types such as grass, pavement, or open fields—as inherently safe often overlook the fluctuating nature of the environment and the hazards posed by dynamic objects [180]. Roads and cycling paths, which are generally flat and wide, may appear suitable for landing based on static analysis, yet they are frequently occupied by moving vehicles, bicycles, or pedestrians, significantly complicating the decision-making process. Moreover, many existing approaches repeatedly classify such thematic regions as reliable SLZs without integrating contextual awareness of transient obstacles [1, 56, 153]. This oversimplification introduces substantial safety risks, as areas deemed safe under static assumptions can quickly become unsuitable due to unpredictable elements like traffic congestion, construction activity, or crowd movement. For example, roads and cycling lanes may initially seem ideal due to their geometry, but their constant occupation by dynamic agents renders them highly unreliable—and potentially dangerous—for autonomous UAV landings [180].

To enable safe and adaptive UAV landings in real-world conditions, it is essential to move beyond static terrain classification and adopt a comprehensive framework that integrates both static and dynamic scene understanding. This integration allows for real-time hazard assessment and facilitates context-aware decision-making. By employing semantic segmentation, UAVs can identify structurally appropriate landing zones by classifying thematic terrain types such as roads, paths, grass, and paved areas. However, static analysis alone is insufficient, as it does not account for the presence or motion of dynamic agents. Therefore, accurately forecasting the motion of such objects is crucial to ensuring that landing zones remain unoccupied and viable at the time of descent.

To address the challenges of safe UAV landing in dynamic environments, we propose a comprehensive vision-based framework that unifies multi-scale semantic segmentation, real-time dynamic object tracking, short-term trajectory forecasting, and homography-based motion compensation into a single, closed-loop pipeline. Unlike traditional methods that rely on static terrain analysis or predefined safe zones, our approach incorporates uncertainty-aware reasoning to evaluate scene dynamics and adapt landing decisions ac-

cordingly. By integrating predicted object occupancy into the decision process, the UAV can proactively identify and update SLZs that are free of transient obstacles, even in complex and rapidly changing scenarios. We can summarize our key contributions as follows:

- We introduce a comprehensive framework that dynamically identifies and maps safe landing zones for UAVs using vision-based sensing. Our framework integrates object detection, semantic segmentation, and predictive modeling to assess future object occupancy to distinguish between suitable landing areas and regions with high risk of collision. By leveraging visual data, our approach enables UAVs to navigate and land safely in complex, dynamic environments without reliance on external localization systems.

- To account for UAV motion, we introduce an optimized homography computation method that leverages multi-scale image analysis and a recursive update mechanism. This approach ensures more accurate geometric transformations, reducing distortions caused by UAV dynamics. Additionally, to mitigate trajectory prediction errors, we integrate an uncertainty-aware forecasting module that estimates both the expected position and confidence bounds of moving objects. By explicitly modeling uncertainty, the system can proactively adjust safety envelopes and exclude areas of high risk from potential landing zones. Furthermore, we employ context-aware segmentation to provide semantic cues to enhance scene understanding.

- Our approach has been rigorously tested across a diverse set of real-world scenarios, including urban roads with vehicular flow, suburban roads featuring mixed pedestrian and bicycle traffic, and parks. In each setting, we evaluated key performance metrics for each module of the pipeline, starting from semantic segmentation, to object detection, tracking and trajectory prediction. Finally, we evaluate the quality of the safe landing mapping accounting for static and moving obstacles. Quantitative and qualitative results confirmed that the system performs well in different scenarios, which underscores the practical viability of our framework for predicting SLZs in highly dynamic environments.

The remainder of this chapter of thesis is structured as follows: Section 5.2 details the proposed methodology. Section 5.3 presents experimental results and analysis. Finally,

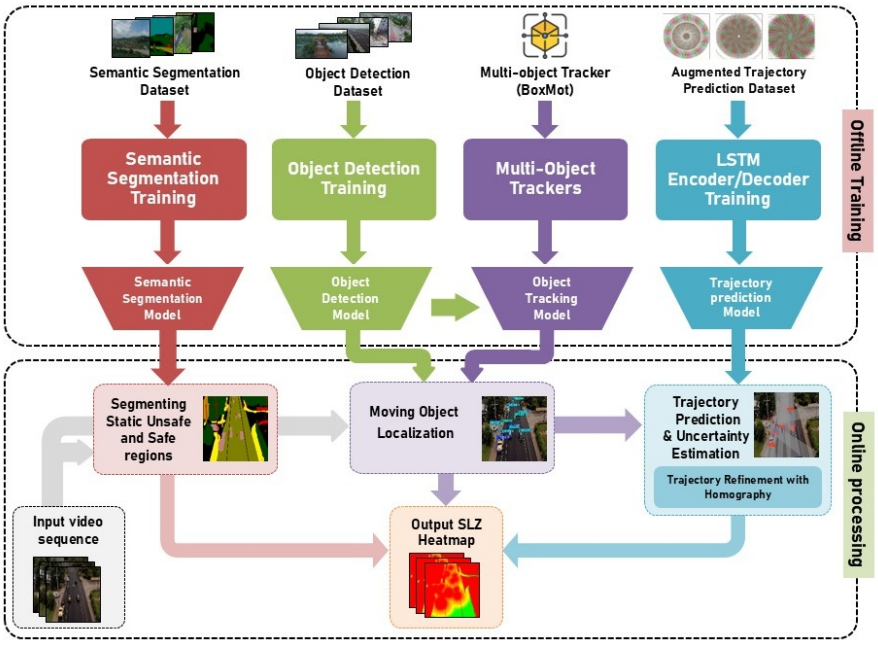


Figure 5.1: Representation of the different modules constituting our pipeline for safe landing zones prediction in dynamic environments.

the thesis concludes with Section 5.4, which discusses key findings and outlines directions for future work.

5.2 Methodology

To address the critical challenge of ensuring UAV safety in dynamic environments, we introduce a unified vision-based framework that enables continuous safe landing zone (SLZ) mapping under motion and environmental uncertainty. Our system integrates real-time semantic segmentation, homography-based motion compensation, and trajectory prediction to detect obstacles, track their motion, and estimate predictive uncertainty. This allows SLZs to be dynamically updated in response to scene changes, object motion, and camera shifts.

Unlike prior work limited to static scene analysis, our framework produces time-aware safety maps that better reflect evolving risk in complex environments. The full pipeline is illustrated in Figure 5.1.

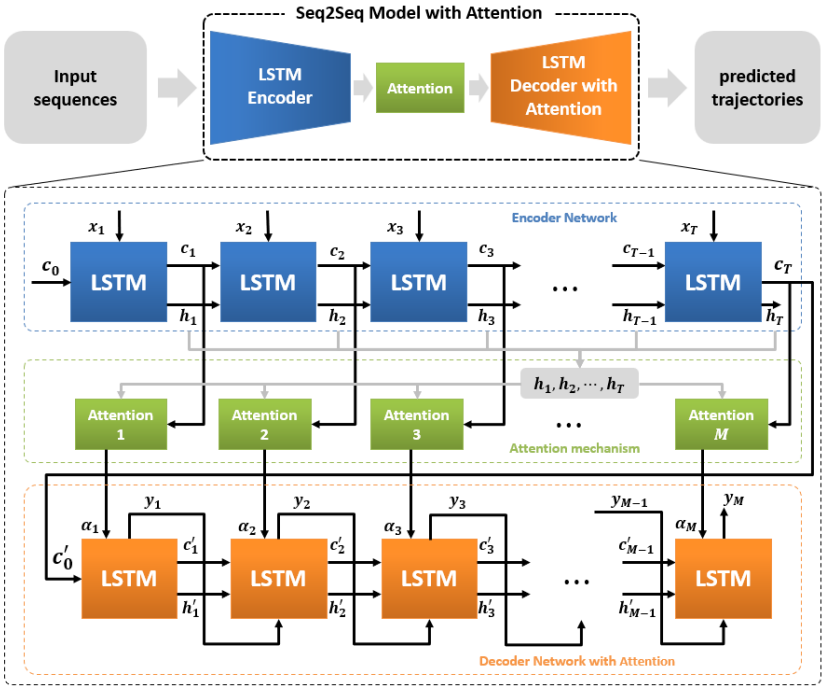


Figure 5.2: Encoder-Decoder architecture for trajectory prediction using attention LSTMs.

5.2.1 Multiple Online Object Detection and Tracking

To prevent collisions, a UAV must continuously monitor its immediate surroundings, detecting and analyzing objects that encroach upon its operational space and pose potential risks. For instance, if a pedestrian or vehicle enters the designated landing zone, the UAV should not only identify their presence but also track their movement patterns to anticipate possible collisions. By leveraging real-time object detection and trajectory prediction, the UAV can dynamically adjust its landing strategy, either by waiting for the obstruction to clear or proactively selecting an alternative safe landing area to ensure safe and efficient operations.

To ensure accurate and real-time detection of potential obstacles across diverse environments, we employ the state-of-the-art YOLOv11 object detector, known for its high accuracy and low-latency inference performance [97]. To enhance detection of small objects, we integrate Slicing Aided Hyper Inference (SAHI) [3], which partitions images into overlapping slices before detection to improve resolution on small targets. The detector is trained in an offline setting using a rich and diverse dataset composed of samples from the Stanford Drone Dataset (SDD) [245], VisDrone [181], and a curated collec-

tion of aerial imagery gathered via the RobotFlow platform. This training set includes various environments and object types (e.g., humans, vehicles, bicycles, and animals) under different lighting and altitude conditions, thereby enhancing the robustness and generalization of the model across urban, suburban, and rural scenes.

For object tracking, we adopt BoxMOT [29], a modular multi-object tracking (MOT) framework that combines detection with object re-identification (ReID) to maintain persistent object identities across frames. By incorporating both appearance embeddings and motion cues, BoxMOT ensures stable tracking even in the presence of occlusions or rapid viewpoint changes. This allows the UAV to maintain an accurate understanding of dynamic object trajectories within its field of view, which is critical for collision avoidance.

Finally, we employ a predictive module to estimate the future positions of moving objects from their trajectory histories. This encoder–decoder network, trained on the motion patterns of multiple object types, enables the UAV to proactively adapt its flight plan, rerouting around high-risk zones, adjusting speed to maintain safe separation, or selecting alternative landing sites with lower collision probability, thereby significantly reducing collision risk.

5.2.2 Moving Trajectory Prediction

Encoder-decoder trajectory prediction

To predict object motion trajectories, we propose a lightweight sequence-to-sequence (Seq2Seq) model based on Long Short-Term Memory (LSTM) networks with attention, illustrated in Figure 5.2. Unlike Transformer-based models, which are computationally intensive, LSTM architectures offer efficiency with fewer parameters while maintaining high predictive accuracy [238]. The integrated attention mechanism enhances the model's ability to focus on relevant past trajectory points, improving its handling of complex, dynamic motion patterns. Dropout and multiple LSTM layers further improve generalization and robustness, making the approach well-suited for real-time motion forecasting.

The encoder is a stacked LSTM network that processes the input sequence $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ and transforms it into a fixed-length context vector, as follows:

$$(\mathbf{h}_t, \mathbf{c}_t) = \text{LSTM}(\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}), \tag{5.1}$$

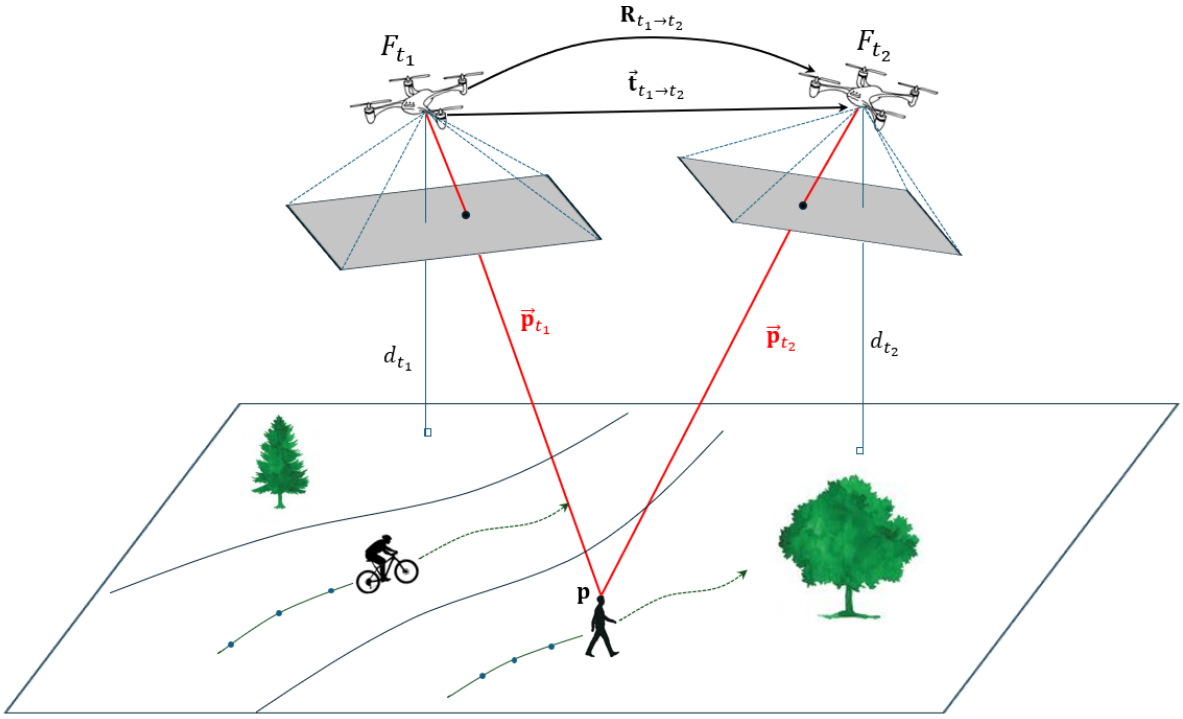


Figure 5.3: Illustration of inter-frame homography estimation.

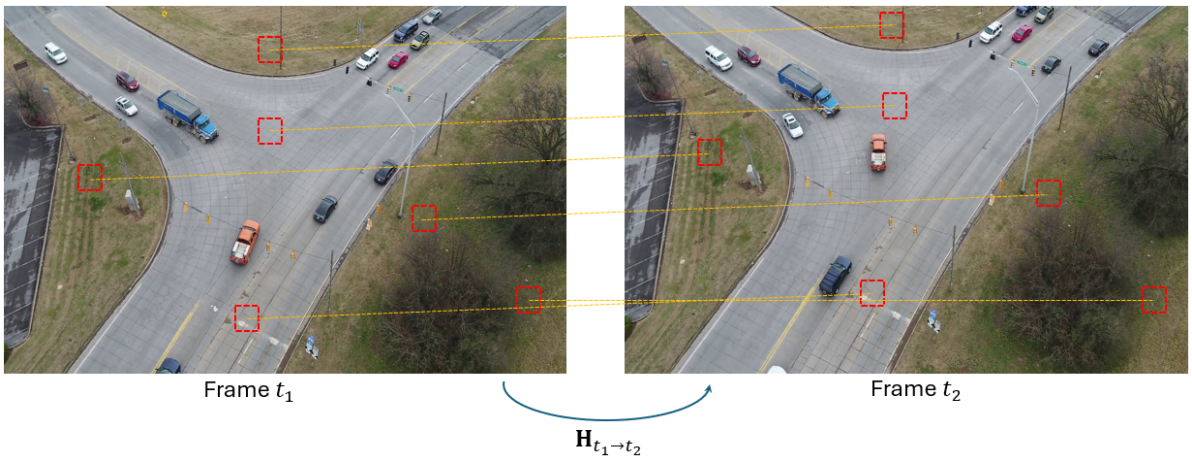


Figure 5.4: Illustration of point matching using patch-based correspondence.

where $\mathbf{h}_t$ and $\mathbf{c}_t$ represent the hidden and cell states at timestep t. In our case, we use an LSTM with two layers with dropout applied between them to prevent overfitting.

The decoder is an LSTM network augmented with an attention mechanism. At each timestep, it receives the previous output y_{t-1} and the context vector c_t, and updates its hidden and cell states as follows:

$$(\mathbf{h}'_t, \mathbf{c}'_t) = \text{LSTM}([y_{t-1}; c_t], \mathbf{h}'_{t-1}, \mathbf{c}'_{t-1}), \tag{5.2}$$

producing the next predicted output y_t. During training, *teacher forcing* is applied with a fixed ratio, using the ground truth value y_{t-1} instead of the predicted output to stabilize learning.

To overcome the limitations of a fixed-length context vector, an attention mechanism computes a weighted sum of the encoder outputs, allowing the decoder to focus on the most relevant parts of the input sequence. The attention weights α_t are computed as:

$$\alpha_t = \text{softmax}\left(\mathbf{v}^T \tanh(\mathbf{W}[\mathbf{h}_t; \mathbf{e}_i])\right), \tag{5.3}$$

where $\mathbf{h}_t$ is the decoder's current hidden state, $\mathbf{e}_i$ are the encoder outputs, and $\mathbf{W}$, $\mathbf{v}$ are learnable parameters. The full model integrates the encoder, attention module, and decoder. It is trained to minimize the mean squared error (MSE) between predicted and ground truth trajectories.

When an object first enters the scene, there is insufficient motion history to predict its trajectory. To address this, we extrapolate the missing past points using average velocity estimation, ensuring temporal continuity within frame boundaries. The resulting trajectories are then passed to an LSTM model for future point prediction. To mitigate noise-induced fluctuations, predicted trajectories are smoothed when necessary.

Inter-frame homography estimation

Let $\mathbf{R}_{t_1 \to t_2}$ and $\mathbf{t}_{t_1 \to t_2}$ denote the rotation matrix and translation vector induced by the UAV motion between two time instants t_1 and $t_2 = t_1 + \Delta t$. Let $\mathbf{p}_{t_1} = (x_{t_1}, y_{t_1}, z_{t_1})$ and $\mathbf{p}_{t_2} = (x_{t_2}, y_{t_2}, z_{t_2})$ represent the coordinates of a 3D point $\mathbf{p}$ in the scene, expressed relative to the UAV camera frame at times t_1 and t_2, respectively. Then, the following geometric transformation holds:

$$\mathbf{p}_{t_2} = \mathbf{R}_{t_1 \to t_2}\, \mathbf{p}_{t_1} + \mathbf{t}_{t_1 \to t_2} \tag{5.4}$$

Let d_{t_1} be the altitude of the UAV at instant t_1 (i.e., distance from the UAV to the planar scene), and $\mathbf{n}$ be the normal vector of the scene. Then, we have : $d_{t_1} = \mathbf{n}^T \mathbf{p}_{t_1}$. Using Eq. (5.4), we obtain the following relation:

$$\mathbf{p}_{t_2} = \left[\mathbf{R}_{t_1 \to t_2} + \frac{\mathbf{t}_{t_1 \to t_2}}{d_{t_1}} \mathbf{n}^T \right] \mathbf{p}_{t_1} \tag{5.5}$$

Let $\mathbf{r}_{t_1} = (\frac{x_{t_1}}{z_{t_1}}, \frac{y_{t_1}}{z_{t_1}}, 1)$ and $\mathbf{r}_{t_2} = (\frac{x_{t_2}}{z_{t_2}}, \frac{y_{t_2}}{z_{t_2}}, 1)$ be the homogeneous coordinates of $\mathbf{p}_{t_1}$ and $\mathbf{p}_{t_2}$ projected on the UAV image at instants t_1 and t_2, respectively. Then, we can prove that:

$$\mathbf{r}_{t_2} = \gamma \left[\mathbf{R}_{t_1 \to t_2} + \frac{\mathbf{t}_{t_1 \to t_2}}{d_{t_1}} \mathbf{n}^T \right] \mathbf{r}_{t_1} = \gamma \, \mathbf{H}_{t_1 \to t_2} \, \mathbf{r}_{t_1}, \tag{5.6}$$

where $\gamma = \frac{z_{t_1}}{z_{t_2}}$, and $\mathbf{H}_{t_1 \to t_2} = \left[\mathbf{R}_{t_1 \to t_2} + \frac{\mathbf{t}_{t_1 \to t_2}}{d_{t_1}} \mathbf{n}^T \right]$ is the homography matrix between frame t_1 and t_2.

The estimation of the homography matrix, $\mathbf{H}_{t_1 \to t_2}$, aims to compute the projective transformation that geometrically relates two consecutive frames, t_1 and t_2. This transformation plays a crucial role in compensating for perspective distortions, enabling accurate image alignment and improved registration [135]. However, a reliable estimation of $\mathbf{H}_{t_1 \to t_2}$ depends on establishing precise correspondences between feature points in both images [203]. When both the UAV and scene objects are moving, homography estimation becomes unreliable, as keypoints may lie on dynamic objects rather than the static background, leading to incorrect transformations. This challenge intensifies in dense or dynamic environments, where keypoints are more likely to lie on moving objects, causing the estimated homography to deviate from the true geometric transformation between frames.

To reduce the influence of dynamic objects, we adopt a patch-based correspondence strategy (Fig. 5.4). Specifically, small localized patches (e.g., 50×50 pixels) are randomly sampled from frame t_1, avoiding regions overlapping with moving objects. Their corresponding matches in frame t_2 are then found using normalized cross-correlation. Only high-confidence matches are retained to ensure robust keypoints for estimating the homography matrix $\mathbf{H}_{t_1 \to t_2}$. To accelerate this process, we apply two optimization strategies. First, we perform multi-resolution matching, generating $L = 3$ scales per frame and beginning the search at the coarsest scale ($\ell = 3$). Matches found at lower resolutions guide the search in higher ones, significantly reducing the required window size (typically ± 10 pixels). Second, we leverage the previously estimated homography

$\mathbf{H}_{t_0 \to t_1}$ to predict likely match regions, further narrowing the search space and improving efficiency.

Once a reliable set of keypoints is established, the homography matrix is computed using the Direct Linear Transform (DLT) algorithm, combined with RANSAC for outlier rejection [203]. This robust estimation process ensures that keypoints influenced by moving objects do not distort the final homography matrix, providing an accurate representation of camera motion even in dynamic and challenging environments.

5.2.3 Dynamic SLZ Mapping and Visualization

The final step of our pipeline overlays outputs from all modules to visualize SLZs, excluding dynamically or structurally hazardous regions. SLZs typically correspond to flat, obstacle-free surfaces such as roads, grass, pavement, and sport fields. This visualization aids rapid pilot decision-making.

To build the safety map, we first map the segmentation output, consisting in environment categories such as road, grass, and buildings, into a binary mask selecting regions corresponding to structurally safe classes. Then, for detected and tracked moving objects, we extract the center coordinates and bounding box dimensions. Detected objects are tracked using YOLOv11 [97] and BoxMot [29]. The past trajectory points for each track are refined using a homography transformation (5.2.2) to compensate for UAV motion. The refined past trajectories are then fed into our trajectory prediction model, depicted in Figure 5.2.

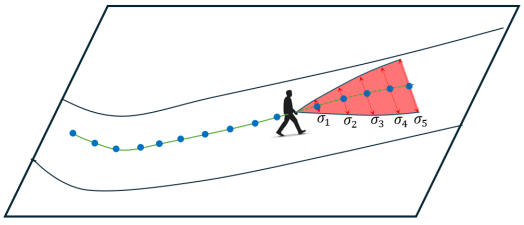


Figure 5.5: A simple visualization of the occupancy buffer for moving objects.

Finally, a grayscale safety map is generated over the refined free zone, assigning pixel values from 255 (very safe) to 0 (unsafe) based on proximity to moving objects. Around each predicted trajectory, we build a dynamic buffer zone that reflects the likely occupancy region of the object. This buffer resembles an isosceles triangle that expands over time. The width of this buffer at each point p_i along the trajectory is proportional

to standard deviation of the prediction error at that position σ_i, effectively capturing uncertainty in future motion at that position (see Figure 5.5). Within this buffer, pixel safety scores are inversely weighted by their distance to the predicted path, accounting for higher risk in regions of uncertain motion. Outside the buffer zone, pixel values are instead computed based on distance to the object center, ensuring that both immediate and anticipated risks are represented in the final safety map. This process yields a per-pixel, risk-aware grayscale representation of the scene, emphasizing safer zones for potential landing.

5.2.4 Model Setting and Implementation Details

The model is implemented in PyTorch, with both the encoder and decoder composed of two LSTM layers, each configured with a hidden state size. A dropout rate of 30% is applied between LSTM layers to prevent overfitting. Training is performed with a batch size of 32, and optimization is done using the Adam optimizer. The learning rate is selected through tuning, and a teacher forcing ratio of 0.1 is used to balance between ground truth inputs and model-generated predictions during training.

For object detection, we trained YOLOv11 [97] for 100 epochs using our custom-labeled dataset described in Section 5.3.1. The training was performed using the AdamW optimizer with an initial learning rate of 1e-4. To improve generalization, we applied data augmentation, including horizontal and vertical flipping, brightness and contrast variation, and random cropping. The network was fine-tuned using transfer learning from weights pre-trained on COCO, allowing faster convergence and improved performance in aerial scenes.

For segmentation,we used a U-Net from the Pytorch Segmentation Models Library with an EfficientNet-B2 encoder pretrained on ImageNet. To further enhance performance, we integrated dilated convolutions to capture multi-scale contextual information and squeeze-and-attention [79] mechanism to improve feature selection in cluttered or complex aerial scenes. These architectural improvements were specifically designed to address challenges commonly encountered in UAV-based semantic segmentation, such as detecting small-scale objects and handling class variability. The training was conducted for 150 epochs using a combination of class-balanced cross-entropy and Dice losses, optimized with Adam and a learning rate equal to 1e-4. All input images were resized

to (512 × 512) pixels, and standard augmentation techniques (e.g., flipping, rotation, cropping) were used to improve generalization.

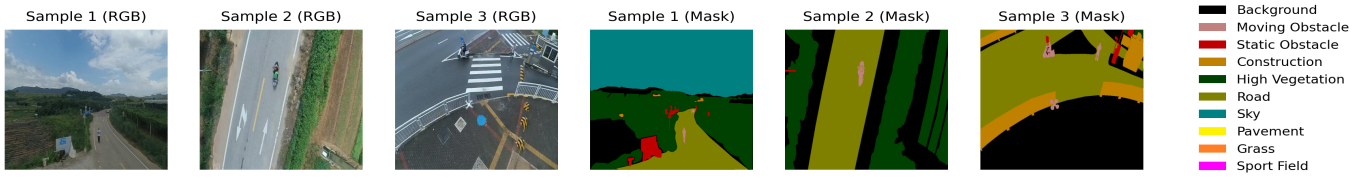


Figure 5.6: Sample RGB images with their corresponding segmentation masks. Distinct colors in the masks indicate different semantic classes.

5.3 Experimental Results

We conducted extensive experiments to validate the proposed framework, evaluating each system component under diverse scenarios with dynamic environments, varying object densities, and terrain types. Comparative analyses with state-of-the-art methods were also performed to highlight the strengths and limitations of our approach.

Predicting SLZs in dynamic environments requires understanding both the thematic content of the scene (e.g., terrain classes) and track moving objects (e.g., vehicles, pedestrians). To train the components of our framework effectively, we used the following datasets:

5.3.1 Datasets

Thematic segmentation dataset

To train a robust segmentation model, we compiled a diverse dataset from multiple sources, including the AeroScape [155], ICG [83] and UAVID [136]. The final dataset includes 2,630 images with these classes: *moving obstacle*, *static obstacle*, *construction*, *high vegetation*, *road*, *sky*, *pavement*, *grass*, and *sports field*. The images are split to 2,369 for training and 260 for validation. Figure 5.6 shows representative samples and their corresponding masks.

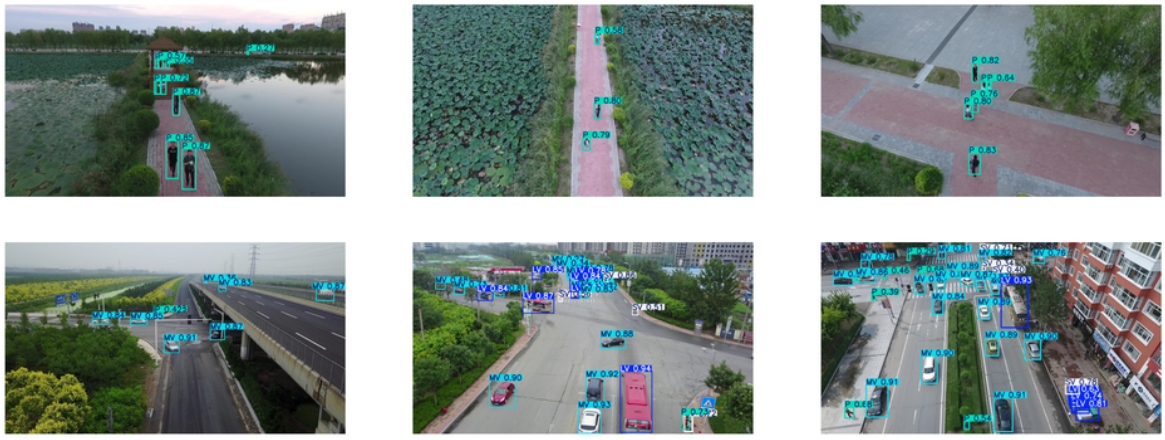


Figure 5.7: Sample images from the object detection training dataset, along with detection results produced by YOLOv11

Scenario 1 Scenario 2 Scenario 3 Scenario 4

Figure 5.8: Sample frames from videos used to evaluate safety landing zone prediction.

Object detection dataset

For object detection, we used a refined version of the publicly available *Urban Zone Aerial Object Detection Dataset* [66], with four classes: *Person (P)*, *Small Vehicle (SV)*, *Medium Vehicle (MV)*, and *Large Vehicle (LV)*. For ore diversity, we added images images from the Stanford Drone and the VisDrone datasets. The final dataset includes 4,022 images for training and 485 for validation. Figure 5.7 shows sample detections with bounding boxes.

Trajectory prediction dataset

Tracking moving objects is essential to anticipate hazards and avoid collisions. For trajectory prediction, we gathered UAV videos showing vehicles, pedestrians, and animals from various angles and resolutions. Each object's trajectory was labeled, and 100 rotational variations were generated per track to boost diversity. Sequences were formatted

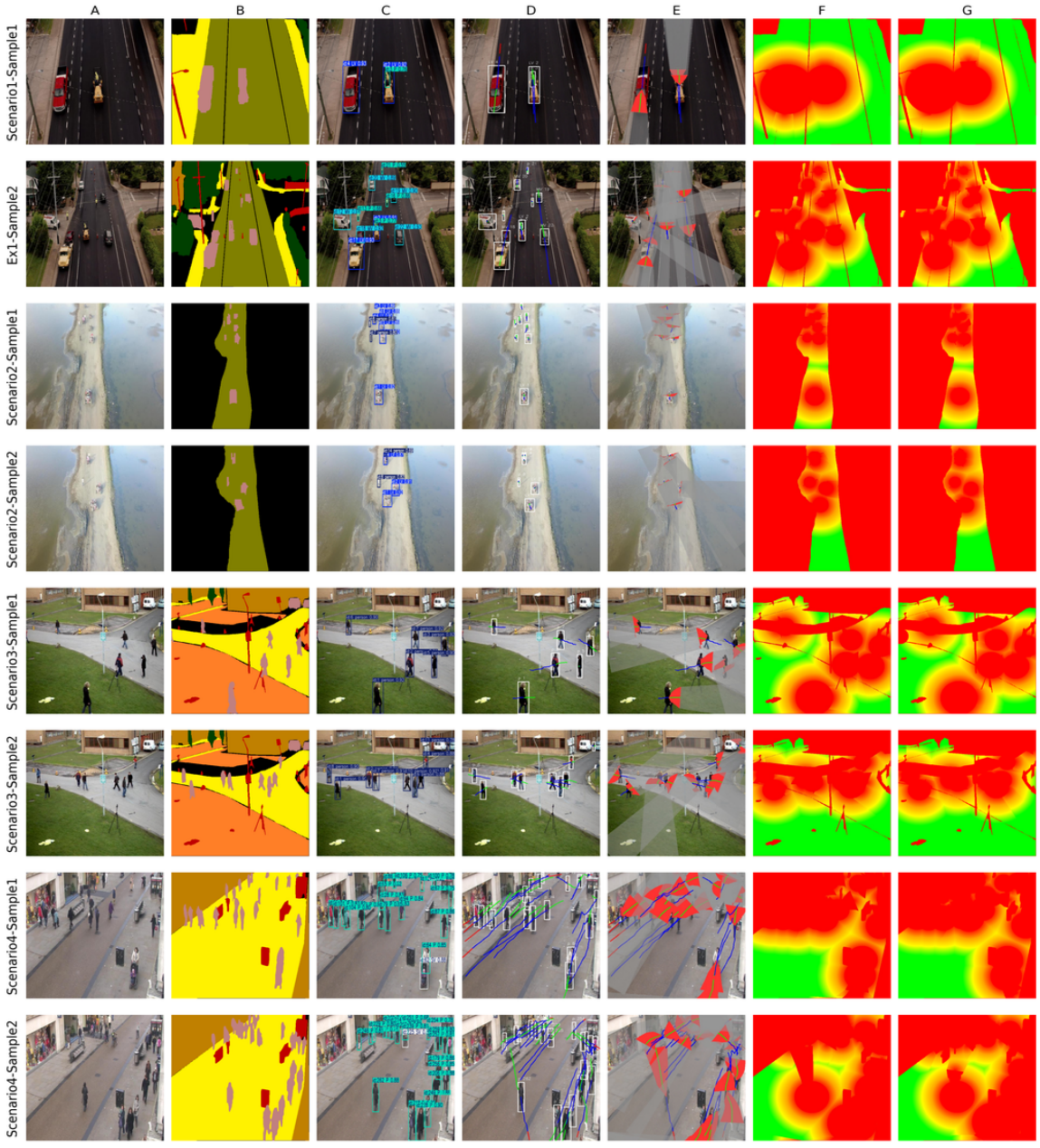


Figure 5.9: Visualization of our framework's heatmap generation across different components. (A) Original RGB image; (B) segmentation prediction; (C) detection and tracking results (D) trajectory prediction: red lines indicate generated missing past points, blue lines represent the observed past trajectory, and the green lines show future trajectory predictions; (E) uncertainty visualization: red regions denote the first level of uncertainty along the predicted trajectory and grey regions indicate the second level of uncertainty based on object movement direction; (F) final SLZ heatmap, ranging from red (very unsafe) to green (very safe); and (G) ground truth heatmap.

with 15 input and 8 output time steps. The final dataset includes 660,870 training and 73,430 validation samples, enabling robust prediction across diverse scenarios.

Test deployment dataset

To quantitatively evaluate the quality and robustness of our SLZ maps, we selected four video sequences that represent different scenarios of dynamic environments:

- *Scenario* 1 (**S1**), 205 frames, altitude $\simeq$ 10m, angle $\simeq 30°$: depicts a street under construction with dynamic elements such as workers, trucks, and cars, along with static obstacles like orange cones. Pavement areas and unobstructed parts can be viable for landing.
- *Scenario 2* (**S2**), 299 frames, altitude $\simeq$ 30m, angle $\simeq 40°$: depicts a marine environment with an unpaved road, featuring moving trucks and pedestrians. In this scenario, the unpaved road is the only viable landing area, making dynamic scene understanding critical for accurate SLZ identification.
- *Scenario 3* (**S3**), 245 frames, altitude $\simeq$ 10m, angle $\simeq 25°$: depicts a street with pedestrians. Although grass and pavement may appear safe based on static scene analysis, pedestrians frequently traverse these areas.
- *Scenario 4* (**S4**), 160 frames, altitude $\simeq$ 15m, angle $\simeq 30°$: depicts an urban alley with pedestrians, cyclists and strollers. Pavement is the only available landing zone when not occupied by pedestrians.

All sequences were recorded at 30 frames per second and manually annotated frame by frame for segmentation, object detection, and tracking (with object IDs), to generate accurate ground truth maps of safe and unsafe landing zones. Figure 5.8 presents a representative frame from each sequence [1].

5.3.2 Ablation Study

Segmentation Evaluation

To assess the generalization ability of our segmentation model, we conducted experiments under two training data configurations:

[1]Link to download data: redhttps://drive.google.com/drive/folders/1AOQ811FBPP31dMbr37-uGGCyBx0M236F?usp=drive_linklink to download datasets

– *Configuration* I: The model was trained on a composite dataset including images from AeroScape, ICG, and UAVid, offering a diverse set of semantic categories and environmental conditions.

– *Configuration* II : This configuration extends the first by incorporating additional images collected from the web that are thematically aligned with our deployment scenarios (e.g., urban , parking and construction zones).

Table 5.1: Segmentation performance comparison across two data training configurations.

Split /Scenario	**Configuration I**			**Configuration II**		
	Acc.	**mIoU**	**Dice**	**Acc.**	**mIoU**	**Dice**
Train	0.981	0.987	0.973	0.981	0.971	0.962
Validation	0.978	0.910	0.944	0.962	0.909	0.931
S1	0.929	0.679	0.517	0.971	0.768	0.868
S2	0.945	0.602	0.751	0.971	0.904	0.935
S3	0.961	0.701	0.824	0.980	0.866	0.928
S4	0.955	0.665	0.798	0.971	0.847	0.917
Avg	0.947	0.662	0.723	0.973	0.846	0.912

Our model evaluation was performed on the four deployment scenarios that were kept strictly unseen during training. The results in Table 5.1 show a clear improvement in segmentation performance when the training dataset is enriched with web-sourced images thematically aligned with the deployment environments. While both configurations achieve high training and validation accuracy, Configuration II consistently outperforms Configuration I on unseen test scenarios, achieving an average mIoU of 84.6% and Dice 91.2% compared to 66.2% and 72.3%, respectively. Configuration I, trained only on general aerial datasets, struggles to generalize well across unseen scene structures, whereas Configuration II better captures semantic variability thanks to its exposure to curated samples representing images close to our testing scenarios. These results confirm that domain shift can result in drop of segmentation performance. However, data augmentation enables to enhance the performance and improve results. This substantial gain highlights the importance of diversifying data.

Detection evaluation

We targeted multiple object categories relevant to aerial scene understanding, such as pedestrians and vehicles. Table 5.2 summarizes the detection performance across training, validation, and diverse test scenarios. On the training set, the model achieved

high box-level Precision (93.2%), Recall (79.6%), and strong detection accuracy with a mean average precision mAP@50 of 87.9%. Validation results show similarly strong generalization with mAP@50 of 85.1%.

Across the four test scenarios, the model consistently performed well. In Scenarios 1, 2 and 3, the model achieved an excellent detection accuracy with precision and recall above 95%, and mAP@50 values reaching up to 99.5%. Scenario 3 stands out with the highest mAP@50–95 of 90.6%, highlighting precise multi-scale localization even under occlusions. Although Scenario 4 presents a more complex scene with a higher object count, the model still performs reliably, achieving mAP@50 of 93.2% and mAP@50–95 of 60.9%. These results demonstrate our model robustness and accuracy in detecting a wide range of object types in complex aerial environments, enabling reliable support for tasks such as trajectory prediction and SLZ assessment.

Table 5.2: Detection evaluation in different scenarios.

Dataset	**# Samples**	**Box(P)**	**R**	**mAP50**	**mAP50-95**
Train	24,787	0.932	0.796	0.879	0.596
Validation	5,580	0.912	0.775	0.851	0.549
S1	2,001	0.985	0.976	0.985	0.863
S2	2,327	0.975	0.951	0.983	0.771
S3	1,767	0.992	0.993	0.995	0.906
S4	3,262	0.881	0.896	0.932	0.609

Multi-Object tracking evaluation

To assess multi-object tracking in diverse environments, we used BoxMot [29], a modular framework combining object detection and re-identification (ReID). A custom-trained YOLOv11 served as the base detector, and objects were tracked across frames using visual embeddings to maintain identity despite occlusions or abrupt motion. BoxMot processes each frame in 15–23 ms, enabling real-time tracking.

Tracking performance was evaluated using three metrics [134]: 1) *Higher Order Tracking Accuracy* (HOTA): evaluates detection and association accuracies, offering a balanced assessment of object localization and identity tracking, 2) *Multiple Object Tracking Accuracy* (Mota): aggregates errors from false positives, false negatives, and identity switches into a single score, reflecting the overall tracking quality, 3) *Identification precision-recall* (IDF1): measures how well the tracker maintains identity consistency over time. It is the F1-score of detections that are correctly identified.

We implemented a custom evaluation pipeline to compute these metrics. Object crops were extracted from video frames, and visual embeddings were generated using a pre-trained ResNet-50 model. Cosine similarity between embeddings is then used to match identities between ground truth and predictions, constrained by an IoU threshold of 0.3 to ensure spatial overlap. Matched identities across frames were used to calculate the metrics.

As shown in Table 5.3, S1 (construction) had strong IDF1 (96.96%) but lower MOTA (80.21%), while S2 (marine) had balanced performance (HOTA = 94.43%). S3 (pedestrians) achieved the best results overall. S4, with similar-looking pedestrians and occlusions, was most challenging (HOTA = 82.39%, MOTA = 65.99%, IDF1 = 87.45%). These results confirm that our pipeline maintains strong identity tracking (IDF1), with detection accuracy varying by scene complexity. HOTA remains a reliable metric for overall performance across SLZ scenarios.

Table 5.3: Evaluating multi-object tracking across different scenarios.

Scenarios	**HOTA**	**MOTA**	**IDF1**
S1	90.54%	80.21%	96.96%
S2	94.43%	89.17%	94.44%
S3	97.48%	94.91%	99.40%
S4	82.9%	65.99%	87.45%

Trajectory prediction results

Table 5.4: Evaluating seq2seq trajectory prediction.

Dataset Split	**Num Samples**	**MSE**	**MAE**
Train	660,870	0.0004	0.0047
Validation	73,430	0.0004	0.0048
S1	1,435	0.0015	0.0073
S2	2,107	0.0001	0.0021
S3	1,525	0.0002	0.0047
S4	1,457	0.0006	0.0065

We evaluated our trajectory predictor across four scenarios representing varying motion complexity, from straight paths to curved trajectories in dense environments. For each test sample, we computed the Mean Square Error (MSE) and Mean Absolute Error (MAE), averaged over the entire set. Results are summarized in Table 5.4.

The model performs best in S2 and S3 (straight motion), with minimal error. S4, involving moderate curvature, showed slightly higher but acceptable errors. S1, with complex trajectories around moving objects, had the highest MSE and MAE. Overall, the low error values confirm the model's ability to generalize across diverse motion patterns. Qualitative overlays of predicted versus ground-truth paths (see Figure 5.9) further reveal that even in the most congested scenario, the model captures both the coarse directionality.

5.3.3 Safety Map Evaluation

To assess similarity between predicted and ground truth safety maps, we use Weighted Mean Squared Error (WMSE), which penalizes overestimations (predicting a pixel as safer than it is) more heavily than underestimations. For each pixel, the squared error between predicted and true safety values (scaled 0–1) is calculated, with a higher weight if the predicted value exceeds the ground truth. WMSE is then averaged over all pixels and images. A WMSE of 0 indicates perfect alignment; higher values reflect greater deviations.

Table 5.5 shows results across the four scenarios. S2 performed best (MSE = 6×10^{-4}, WMSE = 8×10^{-4}), indicating high accuracy with minimal overestimation. S3 also had low error (MSE = 15×10^{-4}, WMSE = 23×10^{-4}), while S1 showed moderate discrepancies (WMSE = 168×10^{-4}). S4 had the highest errors (MSE = 202×10^{-4}, WMSE = 344×10^{-4}). This suggests this scene poses more difficulty, likely due to occlusions or visually ambiguous regions that make safety estimation more challenging. Figure 5.9 presents one qualitative examples per scenario, comparing predicted and ground truth heatmaps step by step.

Table 5.5: Evaluation of Safety map: Inference Across Diverse Examples

Scenarios	**MSE**	**WMSE**
S1	0.0101	0.0168
S2	0.0006	0.0008
S3	0.0015	0.0023
S4	0.0202	0.0344

Finally, for qualitative evaluation, we applied our model to three additional unseen landing scenarios (**S5**: altitude $\simeq$ 15m, angle $\simeq 15°$, **S6**: altitude $\simeq$ 20m, angle $\simeq 0°$, **S7**: altitude $\simeq$ 15m, angle $\simeq 0°$). Each scenario depicts a distinct environment in which

the UAV initiates a descent from a bird's-eye view. As illustrated in Figure 5.10, the model consistently predicts safe landing zones across diverse scenes. It effectively avoids unsafe regions and exhibits strong adaptability to varying scene structures, even under challenging conditions such as occlusions or difficult lighting. These qualitative results further support the model's capacity for generalization and its practical suitability for real-world UAV deployment.

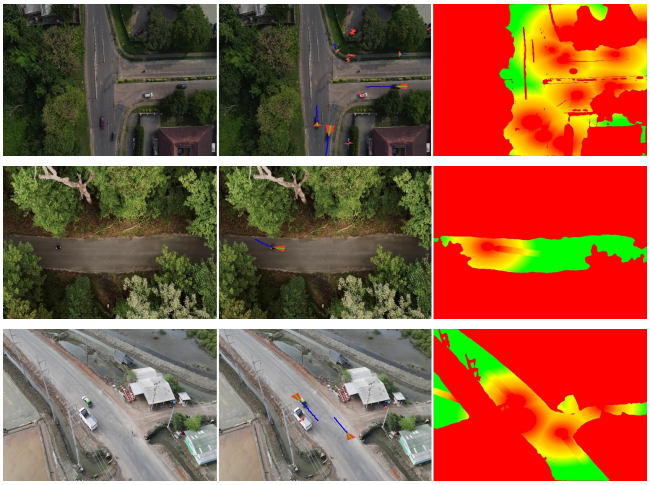


Figure 5.10: Samples from unseen landing scenes (S5, S6, S7) depicting: (left) RGB frame, (middle) trajectory prediction, and (right) predicted safety map.

5.3.4 Discussion

The proposed SLZ framework demonstrates strong performance in dynamic environments by combining static terrain segmentation with real-time object tracking and trajectory prediction. Tests across varied scenes confirm that this integration reduces false-positive SLZ detections. A key strength of our approach lies in its ability to accurately project moving object trajectories under UAV motion, thanks to optimized homography computation and multi-scale scene alignment. By continuously updating the transformation, the system maintains spatial coherence even during flight, ensuring reliable landing decisions. Unlike methods that rely solely on static segmentation or basic tracking, our unified pipeline fuses static terrain analysis with real-time dynamic tracking, improving both safety and timing for collision avoidance—as confirmed by our ablation results.

Despite the strengths of our system, several limitations remain. The lightweight trajectory models may struggle in cases of erratic motion, such as abrupt turns, group

dynamics, or non-linear paths. Moreover, the reliance on overhead RGB imagery reduces robustness in occluded environments, such as dense vegetation or urban canyons, where critical visual cues may be absent. Future work should investigate the integration of IMU data and additional sensing modalities (e.g., infrared, LiDAR) to improve resilience under such conditions.

Although the system is optimized for high-end embedded GPUs, deployment on resource-constrained platforms may require model compression or pruning strategies to maintain real-time performance. Semantic segmentation accuracy could also be improved by employing multiple specialized models tailored to different scene types, selectively activated based on contextual cues, as proposed in recent work [6]. Beyond binary SLZ decisions, segmentation outputs could enable more fine-grained safety assessments by incorporating multiple risk levels, identifying scene-specific hazards (e.g., dynamic agents, uneven terrain), and capturing temporal dynamics such as crowd formation or obstacle movement.

5.4 Conclusion

We presented a comprehensive vision-based framework for Safe Landing Zone (SLZ) identification that addresses both static terrain analysis and dynamic obstacle prediction in real-world UAV operations. By combining semantic segmentation of the landing surface with real-time tracking and short-term forecasting of moving objects, and compensating for UAV motion through an optimized homography pipeline, our system ensures that SLZ maps remain accurate and up-to-date even in highly dynamic scenes. Experiments across four diverse scenarios, including construction sites, suburban roads, and public parks, demonstrated strong SLZ detection performance. Beyond empirical results, the framework offers practical advantages: its modular design allows independent upgrading of components (e.g., segmentation, tracking, homography), and its closed-loop architecture enables continuous scene reassessment, adapting to abrupt UAV maneuvers or sudden object appearance.

Chapter 6

Conclusion

This thesis presented a comprehensive study on vision-based SLZ detection for UAVs, addressing both static and dynamic environments using advanced deep learning techniques. The motivation stemmed from the growing reliance on UAVs in civilian applications and the corresponding need for reliable autonomous landing strategies under complex scene conditions.

We began by reviewing the foundations of computer vision techniques applicable to UAV navigation, covering object detection and image segmentation from both classical and deep learning perspectives. A detailed literature review revealed the limitations of binary classification and rigid mapping techniques in SLZ detection, particularly in real-time and safety-critical contexts.

In Chapter 3, we proposed a deep regression model to generate continuous safety maps from UAV imagery. By assigning safety scores at the pixel level, this model provided a flexible and context-aware alternative to conventional safe/unsafe classifications. The regression-based approach demonstrated promising results on diverse datasets, establishing its potential for real-world deployment.

Chapter 4 introduced OR-SLZNet, a novel deep ordinal regression model that integrates multimodal scene information including photometric and geometric cues such as color, flatness, and depth. This model extended the binary classification paradigm by mapping terrain safety across multiple ordered levels. Notably, the architecture maintained a low inference time, making it suitable for embedded UAV systems and real-time

processing. Experimental validation on several benchmark datasets highlighted its accuracy, robustness, and adaptability across different environments and viewpoints.

To handle the complexities of dynamic scenes, Chapter 5 proposed a unified SLZ mapping framework that combines semantic segmentation, object detection and tracking, trajectory prediction, and homography-based motion compensation. This closed-loop pipeline continuously updated safety maps in response to transient obstacles and UAV motion, ensuring reliable landing decisions in real-time scenarios. Tests on real-world-like dynamic scenes, including roads, construction zones, and public areas, demonstrated strong performance and adaptability.

In summary, this thesis contributes the following:

- A deep regression-based approach for dense SLZ prediction using UAV imagery.
- OR-SLZNet, a real-time ordinal regression model leveraging both photometric and geometric scene cues.
- A dynamic SLZ mapping framework integrating vision-based tracking, motion forecasting, and UAV motion compensation.

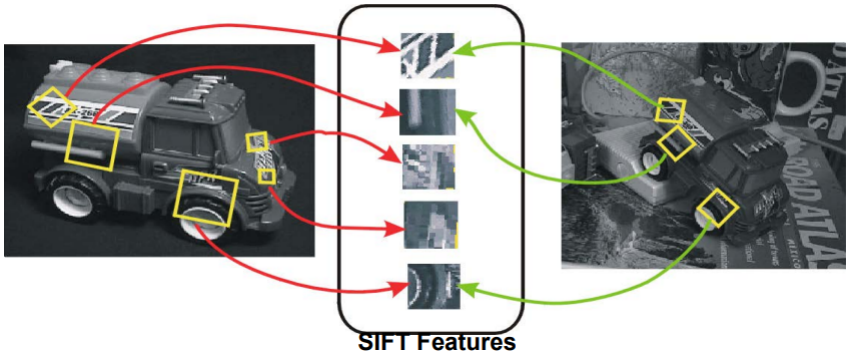
Together, these contributions offer a scalable and efficient vision-based solution for safe UAV landings in varied operational contexts. Future work will explore integration with onboard UAV hardware, extension to 3D terrain modeling, and deployment in real-world field tests. Advancements in lightweight model architectures and multi-sensor fusion could further enhance the reliability and autonomy of UAV landing systems.

Appendix A

Pre-Deep Learning Object Detection Methods

Before the advent of deep learning, object detection relied on handcrafted features and classical machine learning algorithms. Typical pipelines were decomposed into three stages: (i) proposal generation, (ii) feature extraction, and (iii) region classification [224, 248]. These approaches laid the groundwork for modern detectors but were limited in their ability to capture complex visual variability and to scale to large, unconstrained datasets.

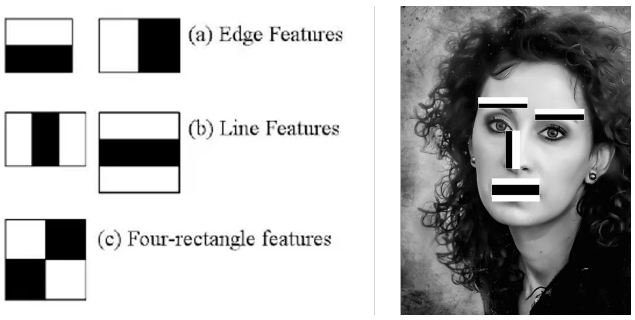
Figure A.1: SIFT (Scale-Invariant Feature Transform) [130].



A common early strategy was the multi-scale sliding window approach [224]. The image was scanned exhaustively at different locations, scales, and aspect ratios over an image pyramid, and at each window position features were extracted and passed to a

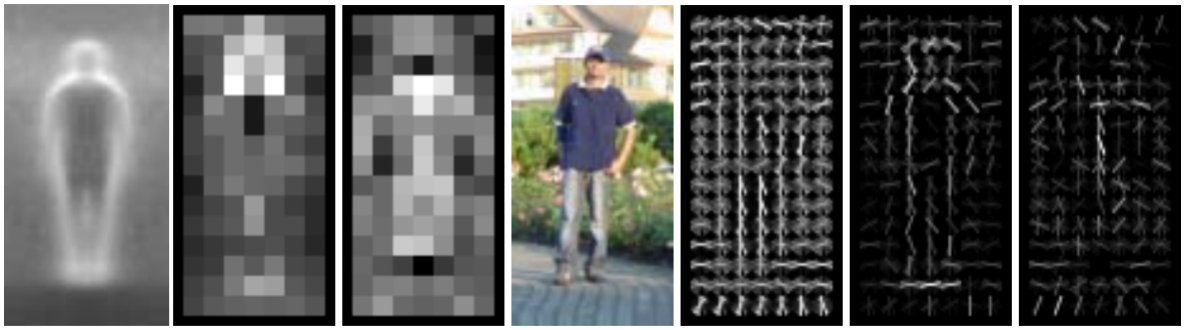
classifier to decide whether an object of interest was present [248]. While conceptually simple, this exhaustive search generated an enormous number of candidate windows, resulting in redundant proposals, high computational cost, and sensitivity to design choices, which restricted its suitability for real-time or large-scale applications.

Figure A.2: Haar-like features used in Viola–Jones detection [213].



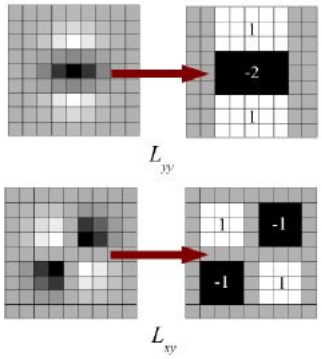
Handcrafted feature descriptors were central to improving robustness within this framework. SIFT (Scale-Invariant Feature Transform) [130] detects stable keypoints across scales using difference-of-Gaussian extrema and encodes local neighborhoods with gradient orientation histograms, providing robustness to changes in scale, rotation, and moderate illumination variations (Figure A.1). Haar-like features [213] capture edge and texture patterns through intensity differences between adjacent rectangular regions and were used in the Viola–Jones framework, where an AdaBoost-based cascade of classifiers enables efficient real-time face detection (Figure A.2). HOG (Histogram of Oriented Gradients) [51] describes local object shape by aggregating gradient orientation histograms over spatial cells with normalization, proving particularly effective for human detection due to its sensitivity to contours and robustness to illumination and pose changes (Figure A.3). SURF (Speeded-Up Robust Features) [19] was introduced as a more computationally efficient alternative to SIFT, using integral images and box filters to approximate Hessian-based detectors and a compact descriptor, thereby accelerating detection and matching (Figure A.4). These descriptors became standard components of pre-deep-learning object detection systems and remain influential in classical feature-based recognition [224].

Figure A.3: Histogram of Oriented Gradients for human detection [51].



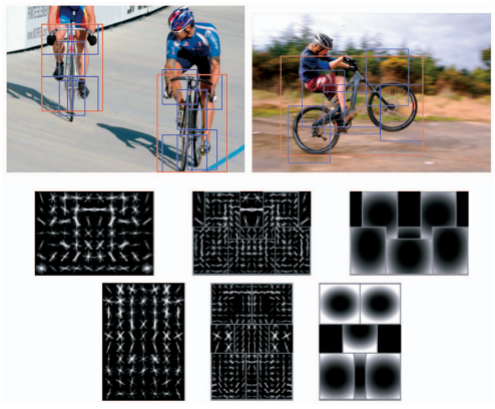
Once candidate regions and features were obtained, classification was typically performed using traditional discriminative models. Support Vector Machines (SVMs) were widely adopted for their strong performance on moderate-sized datasets and their ability to handle high-dimensional feature spaces [224]. By maximizing the margin between classes, linear SVMs provided efficient and relatively scalable solutions, whereas kernel SVMs (e.g., RBF) offered greater flexibility at the cost of increased computational complexity and more demanding hyperparameter tuning. Boosting-based methods, such as AdaBoost, further contributed to early successes [213]. By combining multiple weak learners into a strong classifier and focusing iteratively on misclassified samples, boosting improved detection accuracy. In cascade architectures, simple classifiers rapidly rejected easy negatives in early stages, reserving more complex computations for promising candidates, which enabled practical real-time detection in constrained scenarios.

Figure A.4: SURF (Speeded-Up Robust Features) [19].



A major breakthrough in traditional object detection came with Deformable Part-Based Models (DPM) [62]. In DPM(Figure A.5), objects are represented as a set of parts arranged in a deformable star-structured configuration, with each part modeled using HOG features and spatial relationships encoded via learned deformation costs. Training is performed using a latent SVM framework with hard negative mining, allowing the model to capture intra-class variability, articulations, and partial occlusions more effectively than rigid templates. DPM achieved state-of-the-art performance on benchmarks such as PASCAL VOC from 2007 to 2009 [224], and is often regarded as the culmination of the pre-deep-learning era. However, its computational demands and limited scalability became increasingly apparent as datasets grew in size and complexity.

Figure A.5: Deformable Part-Based Models (DPM)) [62].



In parallel, several other traditional strategies were explored. Template matching detected objects by correlating image patches with predefined templates [30], offering simplicity but lacking robustness to changes in scale, rotation, and viewpoint. Edge-based detection methods leveraged contour information, using operators such as the Canny edge detector to delineate object boundaries [31]; while effective under controlled conditions, they were sensitive to noise and illumination changes. Color-based detection employed color histograms and probabilistic models to isolate objects with distinctive chromatic properties [202], performing well in specialized scenarios (e.g., skin or logo detection) but degrading in cluttered or visually ambiguous scenes.

Despite their historical impact, these traditional pipelines exhibit several inherent limitations. Exhaustive sliding-window search leads to a large number of redundant proposals and high computational costs [248]. Handcrafted features, though carefully engineered, have limited capacity to encode high-level semantic information and complex appearance variations in unconstrained environments [224]. The separation of proposal generation, feature extraction, and classification into independent stages complicates global optimization and makes the systems brittle and task-specific [248]. Most importantly, these methods cannot learn rich hierarchical representations directly from large datasets, restricting their adaptability and ultimate performance. These limitations motivated the shift toward deep learning-based detectors such as R-CNN [68] and YOLO [175], where convolutional neural networks unify feature learning, localization, and classification within end-to-end trainable architectures, leading to substantial gains in both accuracy and efficiency [224, 248].

Appendix B

YOLO family

YOLOv1 (2015) [175]: introduced by Joseph Redmon in 2015, revolutionized object detection by treating it as a single-pass regression problem rather than relying on region-based approaches like R-CNN [68]. Unlike previous models that performed detection in multiple stages, YOLOv1 processes the entire image at once, making it significantly faster and computationally efficient. It divides the input image into an $S \times S$ grid, where each grid cell predicts bounding box locations, confidence scores, and class probabilities. This unified approach eliminates the need for separate region proposal steps, enabling real-time object detection at 45 frames per second (FPS). However, YOLOv1 sacrifices accuracy for speed, achieving a mean average precision (mAP) of 64.4% on the VOC 2007 dataset, compared to Fast R-CNN's 70% mAP at 0.5 FPS. Despite its limitations, YOLOv1 marked a major breakthrough by introducing a single-stage detection framework that influenced the development of more advanced YOLO versions [9, 65].
YOLOv2 (2016) [176]: introduced by Redmon et al., enhanced the original YOLO architecture by incorporating several key improvements. The backbone network was redesigned as Darknet-19, which consists of 19 convolutional layers and 5 max pooling layers. Unlike YOLOv1 [175], YOLOv2 eliminated dropout and introduced batch normalization after each convolutional layer, leading to more stable training and improved accuracy. It also employed a new training strategy known as union training, which allowed simultaneous training for detection and classification tasks.

One notable extension of YOLOv2 was YOLO9000, a model trained on both the COCO and ImageNet datasets, enabling it to detect around 9000 object categories. Despite these advancements, YOLO9000 retained the same core architecture as YOLOv2. The model achieved a mean average precision (mAP) of 76.8% on the VOC 2007 dataset

at 67 frames per second (FPS), outperforming traditional object detection models such as R-CNN, ResNet, and SSD. Additional improvements in YOLOv2 included the introduction of anchor boxes to enhance recall and a high-resolution classifier for better classification performance, making it a significant advancement over YOLOv1 [9, 65].

YOLOv3 (2018) [177]: In 2018, Redmon and Farhadi introduced YOLOv3, the final version released by Redmon before stepping away from the project. The primary improvement in YOLOv3 over YOLOv2 [176] was the introduction of a deeper feature extractor, Darknet-53, which replaced Darknet-19. Darknet-53 is a more powerful backbone network composed of 53 convolutional layers and incorporates residual connections, improving both feature extraction and network depth.

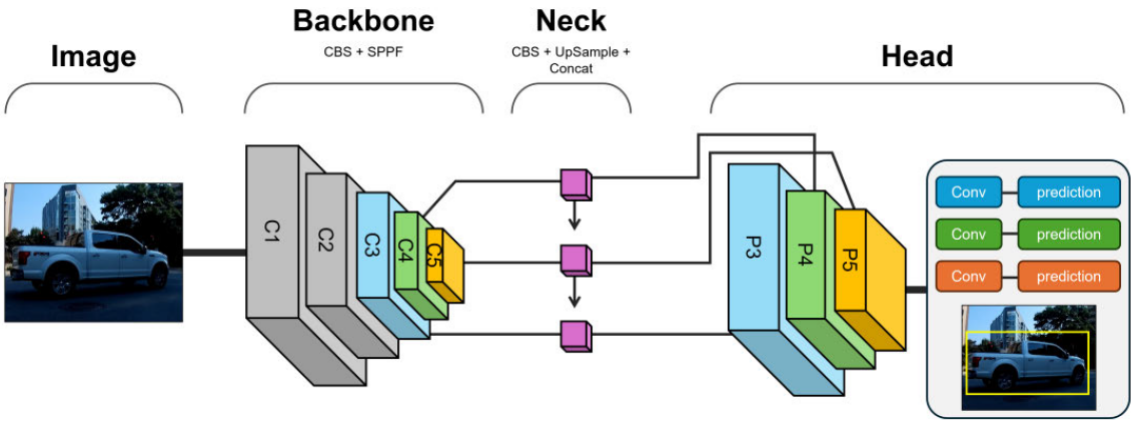
YOLOv3 also introduced multi-scale predictions by detecting objects at three different scales, inspired by the Feature Pyramid Network (FPN) architecture. This enhancement improved the model's ability to detect objects of varying sizes more accurately. Similar to YOLOv2 [176], YOLOv3 employed anchor boxes but expanded their usage by introducing three anchor boxes per scale, resulting in a total of nine anchor boxes across all scales. These modifications led to a significant improvement in detection accuracy, with YOLOv3 achieving a 13.9% higher mean average precision (mAP) on the COCO dataset compared to YOLOv2 [176]. The combination of a deeper backbone, residual connections, and multi-scale detection made YOLOv3 a major advancement in the YOLO series, enhancing both speed and accuracy in object detection [9, 65].

YOLOv4 (2020) [21]: introduced by Bochkovskiy et al. in 2020, continued the evolution of the YOLO framework by incorporating state-of-the-art optimization techniques to improve detection efficiency and accuracy. While it did not introduce new theoretical innovations, YOLOv4 made significant enhancements in data processing, network training, activation functions, and loss functions. The model utilized CSPDarknet-53 as its backbone and introduced spatial pyramid pooling (SPP) and the path aggregation network (PAN) in the neck, enhancing feature extraction. A major contribution of YOLOv4 was the implementation of the mosaic data augmentation technique, which combined four training images into one, improving the detection of small objects. These optimizations allowed YOLOv4 to achieve a balance between inference speed and precision while remaining trainable on consumer-grade GPUs [82, 9, 65].

YOLOv5 (2020) [87]: developed by Glenn Jocher and also released in 2020, marked a paradigm shift in YOLO development by transitioning away from the Darknet framework to PyTorch. Architecturally (Figure B.1), the backbone stacks CBS blocks (Con-

volution + Batch Normalization + SiLU, the Sigmoid Linear Unit) with C3 modules built on CSP bottlenecks (Cross-Stage Partial); the C5 stage ends with SPPF (Spatial Pyramid Pooling–Fast) for lower-latency inference. The neck follows a PAN/FPN design (Path Aggregation Network / Feature Pyramid Network), performing CBS, upsampling, and concatenation to fuse features into pyramid levels P3, P4, and P5 (output strides 8, 16, 32). The head applies small convolutional predictors at each level to output bounding-box offsets, objectness, and class probabilities per anchor. Training uses strong data augmentation—mosaic, MixUp, random affine, and HSV color jitter—to improve generalization. While YOLOv4 [21] and YOLOv5 both advanced one-stage detection, YOLOv5's PyTorch implementation and SPPF layer broadened adoption and sped up deployment [82, 9, 65].

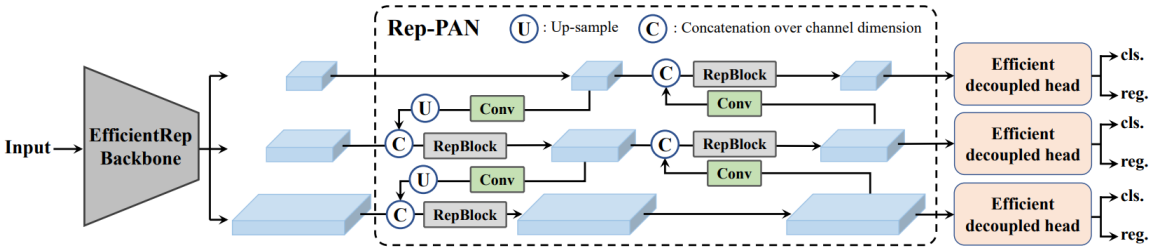
Figure B.1: YOLOv5 Architecture [65]



YOLOv6 (2022) [109]: released by Meituan in 2022, improves efficiency and accuracy with several architecture and training changes. Architecturally (Figure B.2), its EfficientRep backbone is built on RepVGG (re-parameterized VGG): multi-branch convolutions used at training are re-folded into a single 3×3 conv at inference for faster runtime. The neck is a PAN variant using RepBlocks or CSPStackRep blocks for stronger multi-scale fusion. Training refinements include Task Alignment Learning (TAL) for label assignment, VariFocal Loss for classification, and SIoU (Scylla-IoU) or GIoU (Generalized IoU) for box regression. A self-distillation scheme and a quantization pipeline with RepOptimizer plus channel-wise distillation further boost generalization and speed without

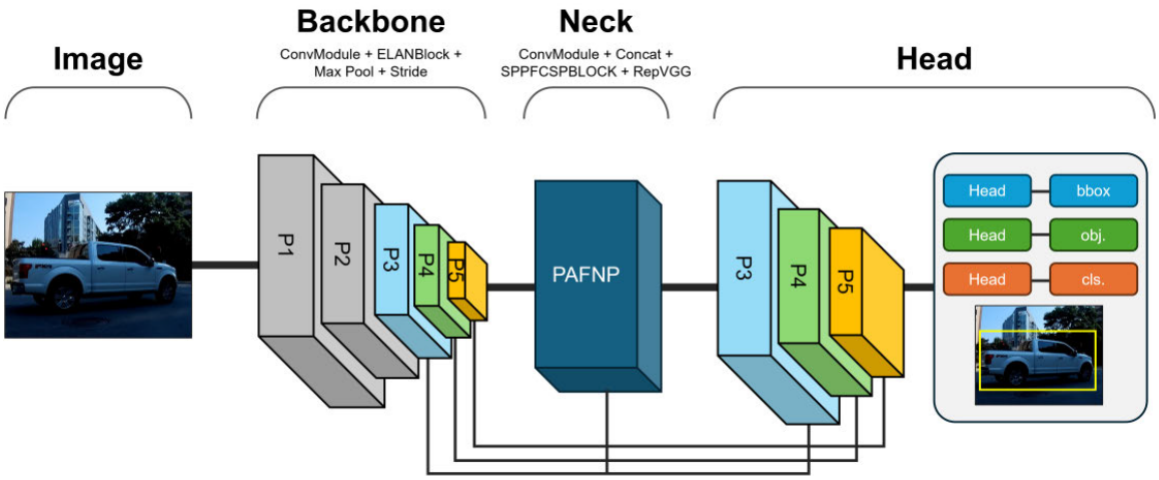
sacrificing accuracy. The Bidirectional Concatenation (BiC) module strengthens localization, and Anchor-Aided Training (AAT) supports both anchor-based and anchor-free heads. YOLOv6 ships in eight model sizes; the largest reports 57.2% AP on COCO at about 75 FPS on an NVIDIA Tesla T4, making it faster and more accurate than prior YOLO releases [82].

Figure B.2: YOLOv6 Architecture [109]



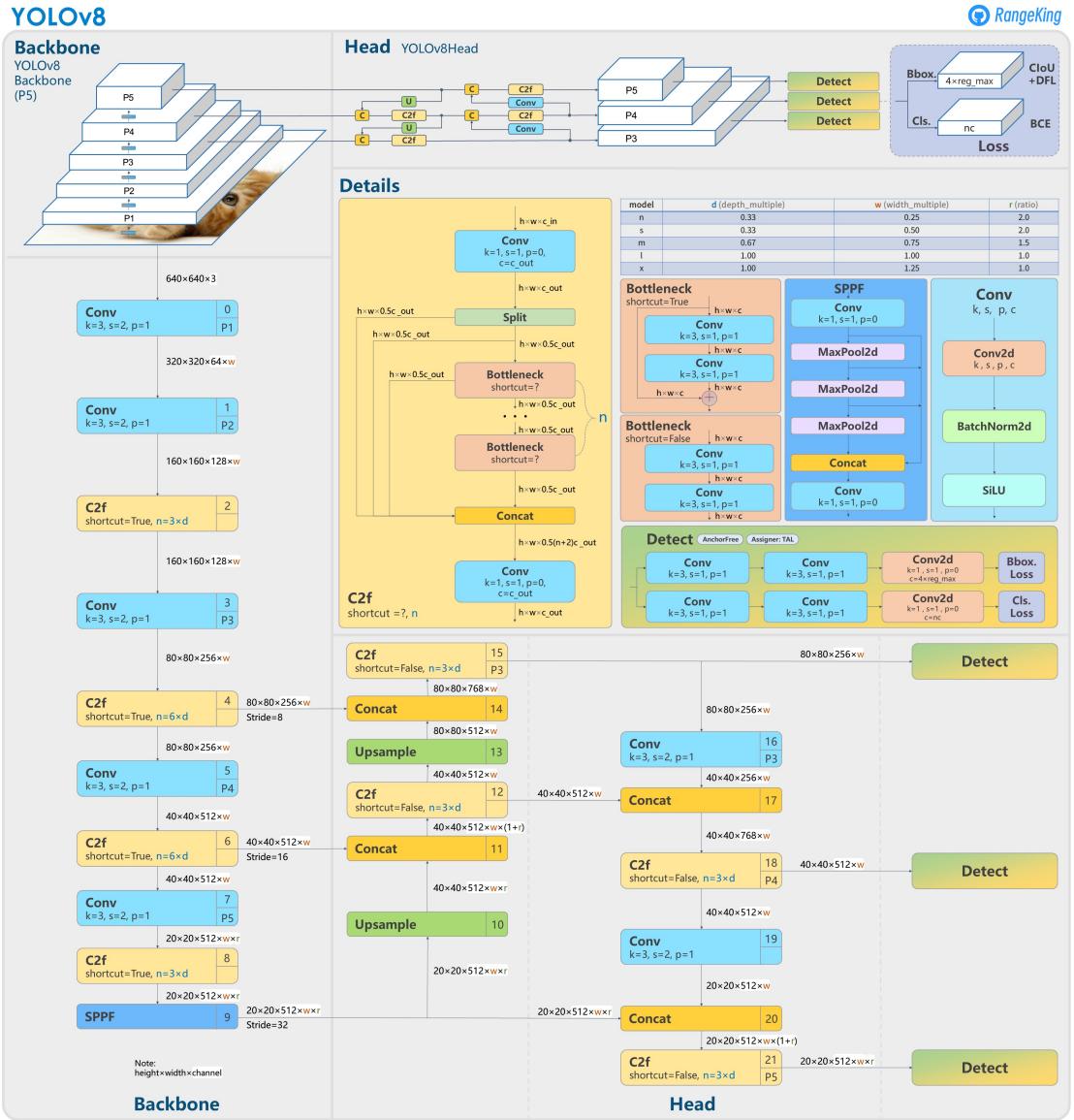
YOLOv7 (2022) [217]: released in 2022, introduced several advancements in object detection, outperforming many existing models with speeds ranging from 5 FPS to 160 FPS. Unlike its predecessors, YOLOv7 was trained on the MS COCO dataset without pre-trained backbones, demonstrating its strong standalone learning capabilities. A major innovation was the introduction of the Extended Efficient Layer Aggregation Network (E-ELAN), which improved learning efficiency by optimizing gradient paths and feature aggregation. YOLOv7 also introduced a novel model scaling approach for concatenation-based architectures, ensuring proportional depth and width scaling to maintain structural integrity. Additionally, it incorporated planned re-parameterized convolutions (RepConvN), which removed identity connections to address residual issues found in previous networks like ResNet and DenseNet. The model further enhanced label assignment by implementing a dual-head approach, using coarse assignment for auxiliary heads and fine assignment for lead heads, improving training efficiency. Batch normalization was integrated directly into the convolutional layers during inference, reducing computational overhead. While YOLOv7 demonstrated improved real-time detection efficiency, studies have shown that YOLOv5 [87] still outperforms it in precision and mean average precision (mAP), though YOLOv7 has a slightly higher recall. Despite this, YOLOv7 remains a powerful choice for real-time applications, offering a balance between speed and accuracy without increasing inference costs [82, 65].

Figure B.3: YOLOv7 Architecture [65]



YOLOv8 (2023) [212]: introduced by Ultralytics in January 2023, represents a significant evolution in the YOLO series, incorporating various architectural enhancements and expanded capabilities. It retains a backbone similar to YOLOv5 but introduces the C2f module (cross-stage partial bottleneck with two convolutions), which improves feature extraction and detection accuracy. YOLOv8 supports multiple scaled versions, ranging from nano (YOLOv8n) to extra-large (YOLOv8x), catering to different computational and application needs. A key advancement in YOLOv8 is its expanded support for semantic segmentation through YOLOv8-Seg, which employs a CSPDarknet53 backbone and a C2f module, diverging from traditional YOLO neck architectures. Benchmarked on the MS COCO dataset, YOLOv8x achieved an average precision (AP) of 53.9% at an image size of 640 pixels, outperforming YOLOv5. Additionally, YOLOv8 demonstrated a remarkable inference speed of 280 FPS on an NVIDIA A100 with TensorRT, emphasizing its real-time efficiency. With its improved architecture, loss functions, and segmentation capabilities, YOLOv8 stands as a versatile and powerful tool for object detection and semantic segmentation applications [82, 65].

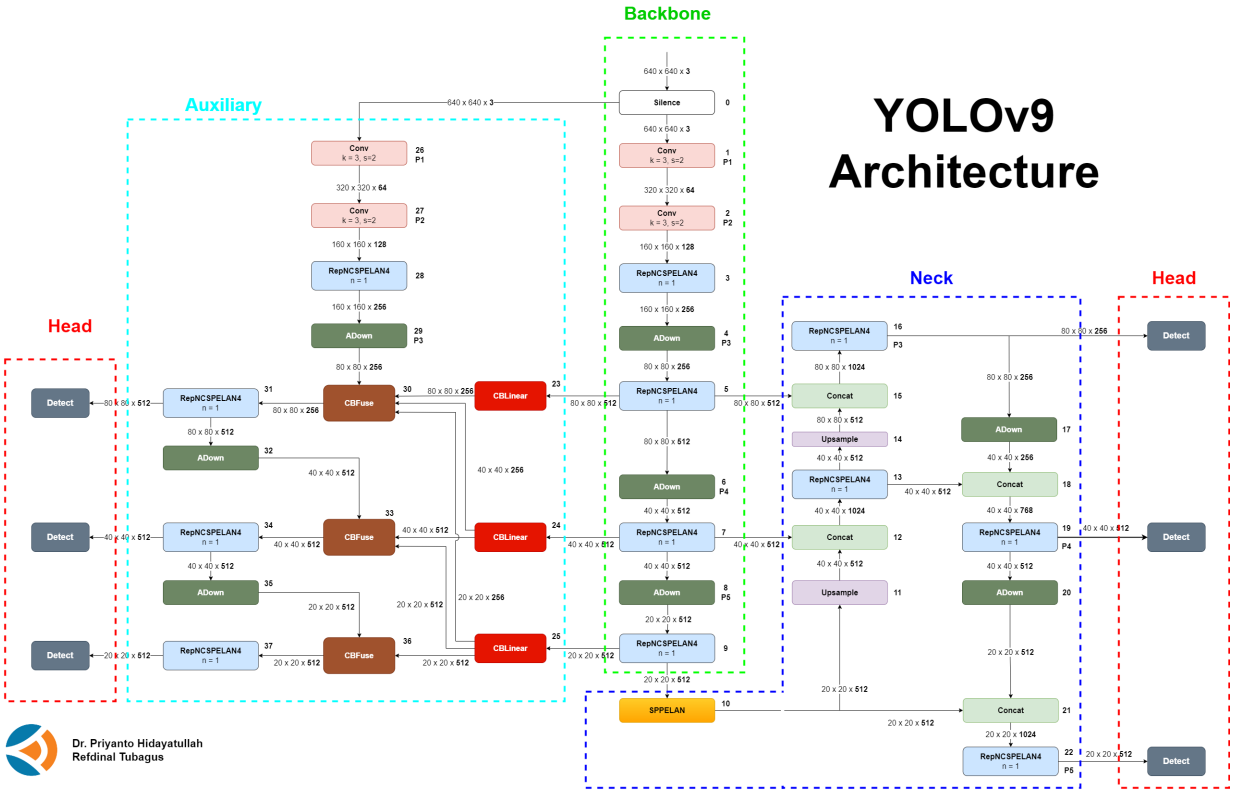
Figure B.4: YOLOv8 Architecture [211]



YOLOv9 (2024) [218]: released in 2024 by Wang et al., introduced groundbreaking innovations to enhance gradient flow, reduce error accumulation, and improve model convergence. A key advancement was the introduction of the Generalized Efficient Layer Aggregation Network (GELAN), which built upon CSPNet and ELAN to provide greater flexibility in integrating computational blocks such as convolutional layers and attention mechanisms. Additionally, YOLOv9 addressed vanishing gradients with

the Programmable Gradient Information (PGI) module, which improved backpropagation across multiple prediction branches, enhancing detection accuracy in complex scenarios. The model also incorporated reversible functions to minimize data loss during transmission through the neural network, ensuring better information preservation and reconstruction. Compared to YOLOv8-X, YOLOv9-E achieved a 1.7% improvement in mean average precision (mAP) while reducing the number of parameters by 16% and computation by 27%, making it more efficient and computationally less intensive. These optimizations positioned YOLOv9 as a versatile and high-performance model for real-time object detection tasks [5, 65].

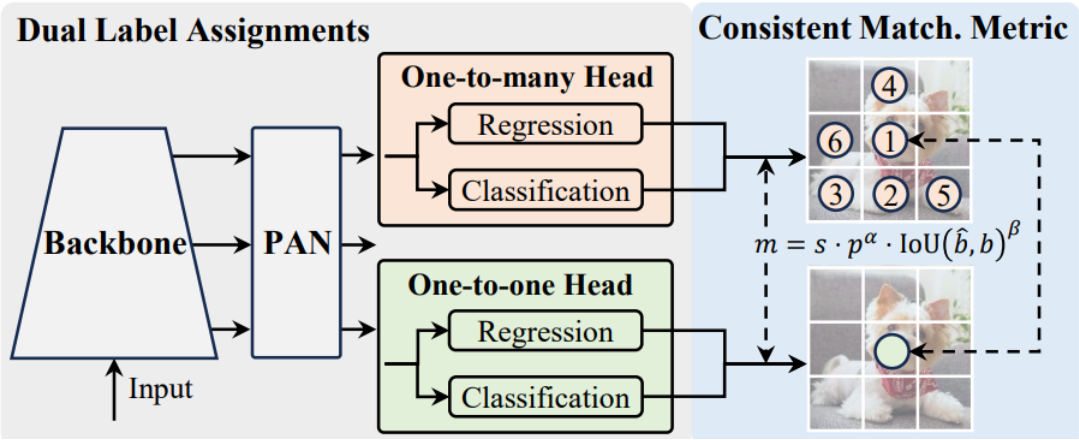
Figure B.5: YOLOv9 Architecture [77]



YOLOv10 (2024) [216]: released in May 2024 by Wang et al. from Tsinghua University, introduced significant advancements in real-time object detection by improving feature aggregation and computational efficiency. A key innovation was the introduction of the C3k2 block, which enhanced feature extraction while reducing computa-

tional overhead, making the model highly suitable for deployment on edge devices. Another major improvement was the elimination of Non-Maximum Suppression (NMS) for post-processing, which reduced inference latency and improved deployment efficiency in real-world applications. YOLOv10 demonstrated superior performance in detecting small and occluded objects, making it particularly effective for tasks like facemask detection and autonomous vehicle applications. Compared to YOLOv9, YOLOv10-M achieved the same mAP (51.1%) while reducing parameters by 23%, and it outperformed YOLOv8 variants by 1.2–1.4% mAP while requiring 28–57% fewer parameters. With a benchmarked mAP50 of 0.944, YOLOv10 set a new standard for balancing detection precision and computational efficiency, further cementing its role in real-time object detection [5, 65].

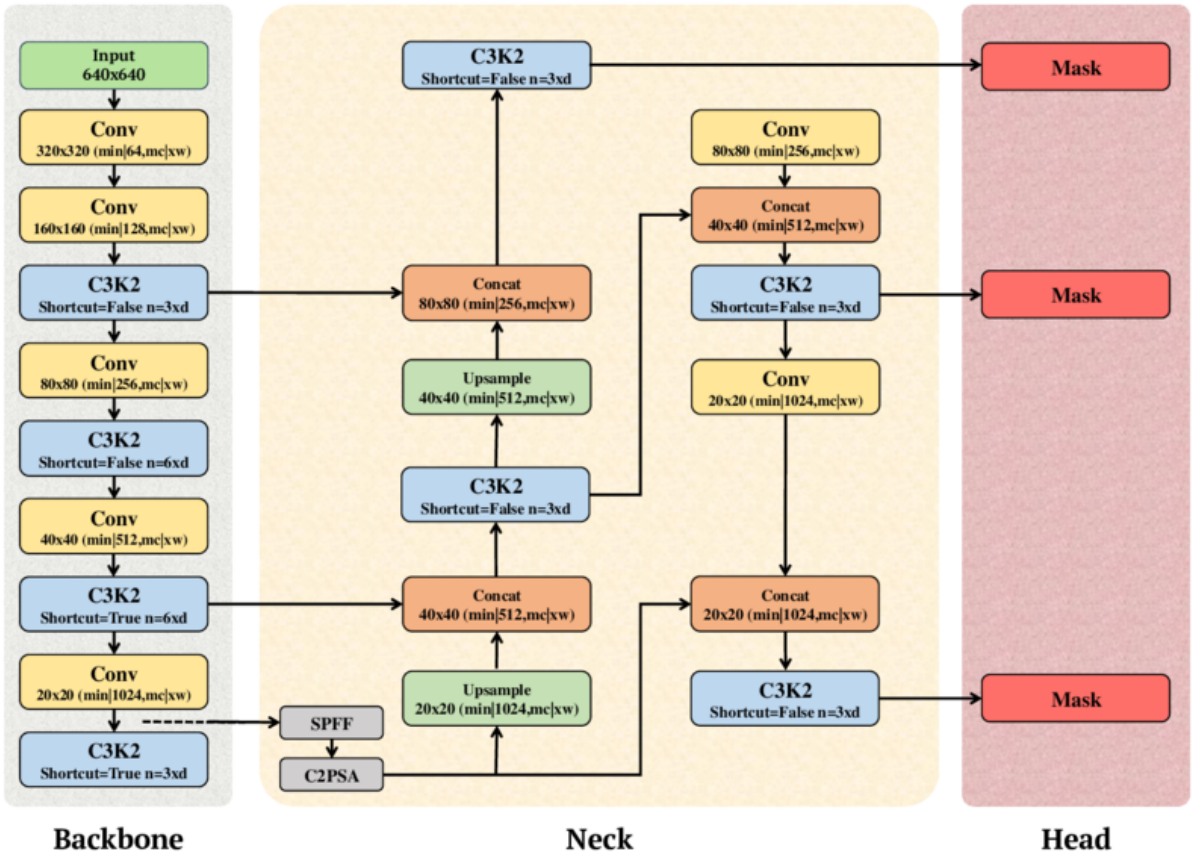
Figure B.6: YOLOv10 Architecture [216]



YOLOv11 (2025) [98]: introduced significant advancements in real-time object detection by enhancing spatial awareness, efficiency, and multi-task capabilities. A key innovation was the introduction of C2PSA (Cross-Stage Partial with Spatial Attention) blocks, which improved the model's ability to focus on critical regions within an image, particularly benefiting applications in healthcare, autonomous systems, and environmental monitoring. The model featured a restructured backbone with smaller kernel sizes and optimized layers, increasing processing speed without sacrificing accuracy. Additionally, the integration of Spatial Pyramid Pooling-Fast (SPPF) enabled faster feature aggregation, making YOLOv11 the most efficient and accurate YOLO model to date, achieving a benchmarked mAP50 of 0.958.

Beyond object detection, YOLOv11 extended its capabilities to instance segmentation, image classification, pose estimation, and oriented object detection, making it a versatile solution for various computer vision tasks. The model introduced multiple scaled versions, from nano to extra-large, ensuring deployment flexibility across resource-constrained edge devices and high-performance computing environments. The refined attention mechanisms, particularly the C2PSA component, enhanced the model's ability to detect small or partially occluded objects in complex scenes. These architectural innovations positioned YOLOv11 as a leading real-time object detection framework, setting a new benchmark in efficiency, scalability, and detection accuracy [98, 5].

Figure B.7: YOLOv11 Architecture [96]



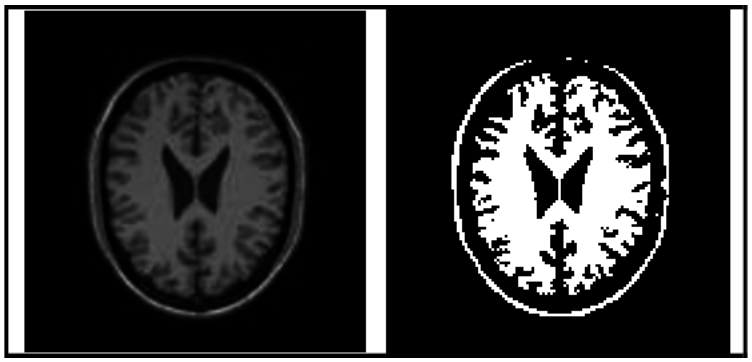
Appendix C

Pre-Deep Learning Image Segmentation Methods

Before the emergence of deep learning, image segmentation was addressed using classical, algorithm-driven techniques designed to partition an image into meaningful regions based on intensity, color, texture, or spatial continuity. These methods rely on explicit mathematical models or optimization principles rather than learned feature representations. Although they have notable limitations in complex real-world scenarios, they established many of the core ideas that modern segmentation networks build upon.

Thresholding. Thresholding is one of the simplest segmentation techniques, converting a grayscale image into a binary mask by assigning pixels above a chosen threshold to the foreground and those below to the background [161]. Global thresholding assumes a bimodal histogram and applies a single threshold to the entire image, while adaptive or local thresholding adjusts the threshold within neighborhoods to handle illumination changes. Despite its simplicity and efficiency, thresholding is highly sensitive to noise, lighting variation, and overlapping intensity distributions between classes.

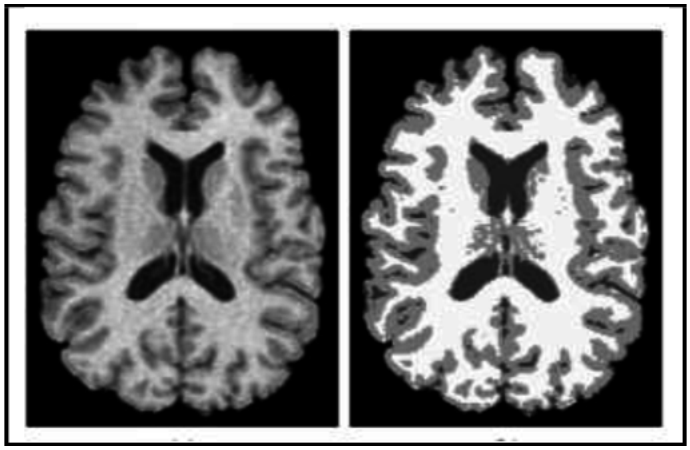
Figure C.1: Segmentation by thresholding technique [237].



Region Growing. Region growing methods start from one or more seed pixels and iteratively expand these seeds by aggregating neighboring pixels that exhibit similar properties such as intensity, color, or texture [157]. The process yields connected, homogeneous regions and can incorporate spatial constraints naturally. However, its performance heavily depends on the choice of seeds and similarity criteria, and it can be sensitive to noise and intensity inhomogeneities.

K-means Clustering. Clustering-based segmentation treats pixels as points in a feature space (e.g., intensity, color channels, spatial coordinates) and partitions them into a predefined number of clusters. K-means assigns each pixel to the closest cluster center, with each resulting cluster interpreted as a segment [57]. While conceptually simple and widely used, K-means requires specifying the number of clusters in advance and assumes roughly spherical, well-separated clusters, which may not hold in complex images.

Figure C.2: Segmentation of brain MRI: original image (left) and 3-means segmentation (right) [237].



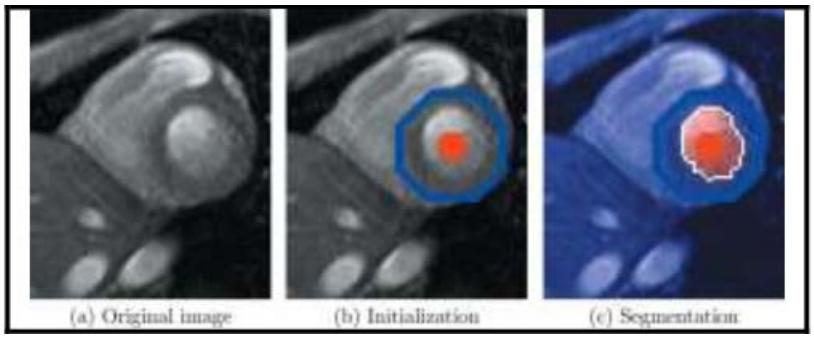
Watershed Transform. The watershed method interprets the gradient magnitude image as a topographic surface, where high gradients correspond to ridges and low gradients to valleys [154]. Simulated flooding from local minima fills catchment basins until watershed lines (ridges) emerge, delineating segment boundaries. Watershed segmentation can produce precise object boundaries but often suffers from severe over-segmentation, requiring careful preprocessing and marker-based control.

Active Contours (Snakes). Active contours, or snakes, are deformable curves that evolve under internal smoothness constraints and external image forces derived from features such as edges or region statistics [94]. Starting from an initial contour, the model iteratively moves toward object boundaries to minimize an energy functional. Extensions such as geodesic active contours and level sets improve robustness to topology changes and complex shapes. These methods can yield accurate boundaries but are sensitive to initialization and local minima.

Figure C.3: Segmentation of brain MRI: original image (left), reference contours (middle), active contour result (right) [237].

Graph Cuts. Graph-cut-based methods formulate segmentation as an energy minimization problem on a graph where pixels (or superpixels) are nodes and edges encode similarity or boundary penalties [27]. By defining data terms (fit to foreground/background models) and smoothness terms (penalizing label discontinuities), segmentation is obtained via a minimum s-t cut that optimally partitions the graph. Graph cuts provide globally optimal or near-optimal solutions for certain energy forms and have been highly influential, but require appropriate energy design and often some user interaction or prior models.

Figure C.4: Segmentation of cardiac MRI by graph cuts: original image (left), initialization (middle), final segmentation (right) [237].



Conditional Random Fields (CRFs). CRFs extend pixel-wise classification by modeling contextual dependencies between neighboring pixels or regions in a probabilistic framework [166]. Unary potentials capture class likelihoods (e.g., from intensity or hand-

crafted features), while pairwise or higher-order terms encourage spatial coherence and alignment with edges. In pre-deep-learning settings, CRFs were often used to refine noisy segmentations and incorporate prior knowledge, though inference could become computationally demanding for large images.

Sparsity-Based Methods. Sparsity-based segmentation approaches assume that image patches or regions can be represented as sparse linear combinations of basis elements (atoms) from one or multiple dictionaries [200]. Different structures or materials correspond to different sparse codes, enabling separation of components such as textures, edges, or anatomical structures. While powerful in certain applications, these methods require carefully constructed dictionaries and can be computationally expensive.

Overall, classical segmentation methods provided important conceptual and algorithmic foundations, including region homogeneity, edge-based partitioning, energy minimization, and probabilistic modeling. However, their reliance on hand-crafted features, heuristic parameters, and limited global context restricts their performance in complex, heterogeneous scenes. These limitations have driven the transition toward deep learning-based segmentation models, which learn rich, hierarchical representations and achieve significantly higher accuracy and robustness across diverse applications.

Bibliography

[1] Abdollahzadeh, S., Proulx, P.-L., Allili, M. S., and Lapointe, J.-F. Safe landing zones detection for uavs using deep regression. In *2022 19th Conference on Robots and Vision (CRV)* (2022), pp. 213–218.

[2] Ahmed, F., and Jenihhin, M. Holistic ijtag-based external and internal fault monitoring in uavs. In *2023 IEEE 24th Latin American Test Symposium (LATS)* (2023), IEEE, pp. 1–6.

[3] Akyon, F. C., Altinuc, S. O., and Temizel, A. Slicing aided hyper inference and fine-tuning for small object detection. In *2022 IEEE international conference on image processing (ICIP)* (2022), IEEE, pp. 966–970.

[4] Alhafnawi, M., Bany Salameh, H. A., Masadeh, A., Al-Obiedollah, H., Ayyash, M., El-Khazali, R., and Elgala, H. A survey of indoor and outdoor uav-based target tracking systems: Current status, challenges, technologies, and future directions. *IEEE Access 11* (2023), 68324–68339.

[5] Ali, M. L., and Zhang, Z. The yolo framework: A comprehensive review of evolution, applications, and benchmarks in object detection. *Computers 13*, 12 (2024).

[6] Allaoui, M. L., and Allili, M. S. Mixlvmm: A mixture of lightweight vision mamba model for enhancing skin lesion segmentation across high tone variability. *IEEE Access 13* (2025), 121234–121249.

[7] Ameur, A. I., et al. Efficient vehicular data sharing using aerial p2p backbone. *IEEE Trans. on Intelligent Vehicles* (2024), 1–14.

[8] Amirkhani, D., Allili, M. S., and Lapointe, J. Cracksight: An efficient crack segmentation model in varying acquisition ranges and complex backgrounds. *IEEE Trans Autom. Sci. Eng. 22* (2025), 19197–19214.

[9] Arkin, E., Yadikar, N., Xu, X., Aysa, A., and Ubul, K. A survey: object detection methods from cnn to transformer. *Multimedia Tools and Applications 82*, 14 (2023), 21353–21383.

[10] Arnegaard, O. T., Leira, F. S., Helgesen, H. H., Kemna, S., and Johansen, T. A. Detection of objects on the ocean surface from a uav with visual and thermal cameras: A machine learning approach. In *2021 International Conference on Unmanned Aircraft Systems (ICUAS)* (2021), IEEE, pp. 81–90.

[11] Arsenos, A., Petrongonas, E., Filippopoulos, O., Skliros, C., Kollias, D., and Kollias, S. Nefeli: A deep-learning detection and tracking pipeline for enhancing autonomy in advanced air mobility. *Aerospace Science and Technology 155* (2024), 109613.

[12] Azad, R., Heidary, M., Yilmaz, K., Hüttemann, M., Karimijafarbigloo, S., Wu, Y., Schmeink, A., and Merhof, D. Loss functions in the era of semantic segmentation: A survey and outlook. *arXiv preprint arXiv:2312.05391* (2023).

[13] Bachir, N., and Memon, Q. A. Benchmarking yolov5 models for improved human detection in search and rescue missions. *Journal of Electronic Science and Technology 22*, 1 (2024), 100243.

[14] Badrinarayanan, V., Kendall, A., and Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence 39*, 12 (2017), 2481–2495.

[15] Bae, G., and Davison, A. J. Rethinking inductive biases for surface normal estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 9535–9545.

[16] Baidya, R., and Jeong, H. Simulation and real-life implementation of uav autonomous landing system based on object recognition and tracking for safe landing in uncertain environments. *Frontiers in Robotics and AI 11* (2024), 1450266.

[17] Balestrieri, E., Daponte, P., De Vito, L., Picariello, F., and Tudosa, I. Sensors and measurements for uav safety: An overview. *Sensors 21*, 24 (2021), 8253.

[18] Bartolomei, L., Kompis, Y., Teixeira, L., and Chli, M. Autonomous emergency landing for multicopters using deep reinforcement learning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2022), IEEE, pp. 3392–3399.

[19] Bay, H., Tuytelaars, T., and Van Gool, L. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9* (2006), Springer, pp. 404–417.

[20] Belmonte, L. M., Morales, R., and Fernández-Caballero, A. Computer vision in autonomous unmanned aerial vehicles—a systematic mapping study. *Applied Sciences 9*, 15 (2019), 3196.

[21] Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934* (2020).

[22] Bong, H. M., Zhang, R., de Azambuja, R., and Beltrame, G. Dynamic open vocabulary enhanced safe-landing with intelligence (dovesei). *arXiv preprint arXiv:2308.11471* (2023).

[23] Bosch, S., Lacroix, S., and Caballero, F. Autonomous detection of safe landing areas for an uav from monocular images. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2006), IEEE, pp. 5522–5527.

[24] Bouhlel, F., Mliki, H., and Hammami, M. Mod-ir: moving objects detection from uav-captured video sequences based on image registration. *Multimedia Tools and Applications 83*, 16 (2024), 46779–46798.

[25] Boulmerka, A., and Allili, M. S. Foreground segmentation in videos combining general gaussian mixture modeling and spatial information. *IEEE Trans. Circuits Systems for Video Technology 28*, 6 (2018), 1330–1345.

[26] Boulmerka, A., Allili, M. S., and Ait-Aoudia, S. A generalized multiclass histogram thresholding approach based on mixture modelling. *Pattern Recognition 47*, 3 (2014), 1330–1348.

[27] Boykov, Y., Veksler, O., and Zabih, R. Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence 23*, 11 (2001), 1222–1239.

[28] Brosch, T., Tang, L. Y., Yoo, Y., Li, D. K., Traboulsee, A., and Tam, R. Deep 3d convolutional encoder networks with shortcuts for multiscale feature integration applied to multiple sclerosis lesion segmentation. *IEEE transactions on medical imaging 35*, 5 (2016), 1229–1239.

[29] Broström, M. Boxmot: Pluggable sota tracking modules for segmentation, object detection and pose estimation models. `https://github.com/mikel-brostrom/boxmot`, 2025. Accessed: 2025-05-09.

[30] Brunelli, R., and Poggio, T. Face recognition: Features versus templates. *IEEE transactions on pattern analysis and machine intelligence 15*, 10 (2002), 1042–1052.

[31] Canny, J. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, 6 (1986), 679–698.

[32] Cao, W., Mirjalili, V., and Raschka, S. Rank consistent ordinal regression for neural networks with application to age estimation. *Pattern Recognition Letters 140* (Dec 2020), 325–331.

[33] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. End-to-end object detection with transformers. In *European conference on computer vision* (2020), Springer, pp. 213–229.

[34] Cazzato, D., Cimarelli, C., Sanchez-Lopez, J. L., Voos, H., and Leo, M. A survey of computer vision methods for 2d object detection from unmanned aerial vehicles. *Journal of Imaging 6*, 8 (2020).

[35] Chatzikalymnios, E., and Moustakas, K. Autonomous vision-based landing of uav's on unstructured terrains. In *2021 IEEE International Conference on Autonomous Systems (ICAS)* (2021), IEEE, pp. 1–5.

[36] Chatzikalymnios, E., and Moustakas, K. Landing site detection for autonomous rotor wing uavs using visual and structural information. *Journal of Intelligent & Robotic Systems 104*, 2 (2022), 27.

[37] Chen, J., Du, W., Lin, J., Borhan, U. M., Lin, Y., Du, B., Li, X., and Li, J. Emergency uav landing on unknown field using depth-enhanced graph structure. *IEEE Transactions on Automation Science and Engineering* (2024), 1–12.

[38] Chen, L., Liu, F., Zhao, Y., Wang, W., Yuan, X., and Zhu, J. Valid: A comprehensive virtual aerial image dataset. In *2020 IEEE international conference on robotics and automation (ICRA)* (2020), IEEE, pp. 2009–2016.

[39] Chen, L., Papandreou, G., Schroff, F., and Adam, H. Rethinking atrous convolution for semantic image segmentation. *CoRR abs/1706.05587* (2017).

[40] Chen, L., Yuan, X., Xiao, Y., Zhang, Y., and Zhu, J. Robust autonomous landing of uav in non-cooperative environments based on dynamic time camera-lidar fusion. *arXiv preprint arXiv:2011.13761* (2020).

[41] Chen, L.-C., Hermans, A., Papandreou, G., Schroff, F., Wang, P., and Adam, H. Masklab: Instance segmentation by refining object detection with semantic and direction features. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 4013–4022.

[42] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence 40*, 4 (2017), 834–848.

[43] Chen, L.-C., Yang, Y., Wang, J., Xu, W., and Yuille, A. L. Attention to scale: Scale-aware semantic image segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 3640–3649.

[44] Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 801–818.

[45] Chen, X., Girshick, R., He, K., and Dollár, P. Tensormask: A foundation for dense object segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision* (2019), pp. 2061–2069.

[46] Chen, X., Ma, H., Wan, J., Li, B., and Xia, T. Multi-view 3d object detection network for autonomous driving. In *IEEE CVPR* (2017), vol. 1, p. 3.

[47] Chopra, O., Nagrare, S. R., Jana, S., and Ghose, D. Drone path planning and dynamic arena-target allocation for crowd surveillance. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering 238*, 11 (2024), 1082–1102.

[48] Chriki, A., Touati, H., Snoussi, H., and Kamoun, F. Uav-based surveillance system: an anomaly detection approach. In *2020 IEEE Symposium on Computers and Communications (ISCC)* (2020), pp. 1–6.

[49] Chu, W., and Keerthi, S. S. Support vector ordinal regression. *Neural computation 19*, 3 (2007), 792–815.

[50] Dai, J., Li, Y., He, K., and Sun, J. R-fcn: Object detection via region-based fully convolutional networks. *Advances in neural information processing systems 29* (2016).

[51] Dalal, N., and Triggs, B. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)* (2005), vol. 1, Ieee, pp. 886–893.

[52] Datta, S., and Durairaj, S. Review of deep learning algorithms for urban remote sensing using unmanned aerial vehicles (uavs). *Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science) 17*, 2 (2024), 66–77.

[53] Debele, Y., Shi, H.-Y., Wondosen, A., Warku, H., Ku, T.-W., and Kang, B.-S. Vision-guided tracking and emergency landing for uavs on moving targets. *Drones 8*, 5 (2024), 182.

[54] Debnath, D., Vanegas, F., Sandino, J., Hawary, A. F., and Gonzalez, F. A review of uav path-planning algorithms and obstacle avoidance methods for remote sensing applications. *Remote Sensing 16*, 21 (2024), 4019.

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

[56] Dhami, H. S., Ignatyev, D., and Tsourdos, A. Semantic segmentation based mapping systems for the safe and precise landing of flying vehicles. *IFAC-PapersOnLine 55*, 22 (2022), 310–315.

[57] Dhanachandra, N., Manglem, K., and Chanu, Y. J. Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science 54* (2015), 764–771.

[58] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[59] Du, H., Wang, W., Wang, X., and Wang, Y. Autonomous landing scene recognition based on transfer learning for drones. *Journal of systems engineering and electronics 34*, 1 (2023), 28–35.

[60] Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., and Tian, Q. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision* (2019), pp. 6569–6578.

[61] Everingham, M., Eslami, S. M. A., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL visual object classes challenge: A retrospective. *International Journal of Computer Vision 111*, 1 (2015), 98–136.

[62] Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence 32*, 9 (2009), 1627–1645.

[63] Fonder, M., and Van Droogenbroeck, M. Mid-air: A multi-modal dataset for extremely low altitude drone flights. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops* (2019), pp. 0–0.

[64] Fu, H., Gong, M., Wang, C., Batmanghelich, K., and Tao, D. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the*

IEEE conference on computer vision and pattern recognition (2018), pp. 2002–2011.

[65] Gallagher, J. E., and Oughton, E. J. Surveying you only look once (yolo) multispectral object detection advancements, applications and challenges. *IEEE Access* (2025).

[66] Ganderla, S. Urban zone aerial object detection dataset. Kaggle. Accessed: 2025-10-08.

[67] Girshick, R. Fast r-cnn. *arXiv preprint arXiv:1504.08083* (2015).

[68] Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2014), pp. 580–587.

[69] Gonzalez-Trejo, J., and Mercado-Ravell, D. Dense crowds detection and surveillance with drones using density maps. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)* (2020), IEEE, pp. 1460–1467.

[70] González-Trejo, J., Mercado-Ravell, D., Becerra, I., and Murrieta-Cid, R. On the visual-based safe landing of UAVs in populated areas: a crucial aspect for urban deployment. *IEEE Robotics and Automation Letters 6*, 4 (2021), 7901–7908.

[71] Guérin, J., Delmas, K., and Guiochet, J. Certifying emergency landing for safe urban uav. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)* (2021), IEEE, pp. 55–62.

[72] Guo, X., Denman, S., Fookes, C., and Sridharan, S. A robust uav landing site detection system using mid-level discriminative patches. In *2016 23rd International Conference on Pattern Recognition (ICPR)* (2016), IEEE, pp. 1659–1664.

[73] Gutiérrez, P. A., Perez-Ortiz, M., Sanchez-Monedero, J., Fernandez-Navarro, F., and Hervas-Martinez, C. Ordinal regression methods: survey and experimental study. *IEEE Transactions on Knowledge and Data Engineering 28*, 1 (2015), 127–146.

[74] Hassanin, M., Anwar, S., Radwan, I., Khan, F. S., and Mian, A. Visual attention methods in deep learning: An in-depth survey. *Information Fusion 108* (2024), 102417.

[75] He, K., Gkioxari, G., Dollár, P., and Girshick, R. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 2961–2969.

[76] He, K., Zhang, X., Ren, S., and Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence 37*, 9 (2015), 1904–1916.

[77] Hidayatullah, P., and Tubagus, R. Yolov9 architecture explained.

[78] Hoanh, N., and Pham, T. V. A multi-task framework for car detection from high-resolution uav imagery focusing on road regions. *IEEE Transactions on Intelligent Transportation Systems* (2024).

[79] Hu, J., Shen, L., and Sun, G. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 7132–7141.

[80] Hu, R., Rohrbach, M., and Darrell, T. Segmentation from natural language expressions. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14* (2016), Springer, pp. 108–124.

[81] Huang, Q., Xia, C., Wu, C., Li, S., Wang, Y., Song, Y., and Kuo, C.-C. J. Semantic segmentation with reverse attention. *arXiv preprint arXiv:1707.06426* (2017).

[82] Hussain, M. Yolov1 to v8: Unveiling each variant–a comprehensive review of yolo. *IEEE access 12* (2024), 42816–42833.

[83] Institute of Computer Graphics and Vision (ICG). Accessed Jan. 31, 2021.

[84] Janitza, S., Tutz, G., and Boulesteix, A.-L. Random forest for ordinal responses: prediction and variable selection. *Computational Statistics & Data Analysis 96* (2016), 57–73.

[85] Jiang, B., Chen, Z., Tan, J., Qu, R., Li, C., and Li, Y. A real-time semantic segmentation method based on stdc-ct for recognizing uav emergency landing zones. *Sensors 23*, 14 (2023), 6514.

[86] Jiang, S., Jiang, W., Guo, B., Li, L., and Wang, L. Learned local features for structure from motion of uav images: A comparative evaluation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 14* (2021), 10583–10597.

[87] Jocher, G., Stoken, A., Borovec, J., Changyu, L., Hogan, A., Diaconu, L., Poznanski, J., Yu, L., Rai, P., Ferriday, R., et al. ultralytics/yolov5: v3. 0. *Zenodo* (2020).

[88] Kakaletsis, E., Mademlis, I., Nikolaidis, N., and Pitas, I. Bayesian fusion of multiview human crowd detections for autonomous uav fleet safety. In *2020 28th European Signal Processing Conference (EUSIPCO)* (2021), IEEE, pp. 2473–2477.

[89] Kakaletsis, E., Mademlis, I., Nikolaidis, N., and Pitas, I. Multiview vision-based human crowd localization for uav fleet flight safety. *Signal Processing: Image Communication 99* (2021), 116484.

[90] Kakaletsis, E., and Nikolaidis, N. Potential uav landing sites detection through digital elevation models analysis. *arXiv preprint arXiv:2107.06921* (2021).

[91] Kakaletsis, E., Symeonidis, C., Tzelepi, M., Mademlis, I., Tefas, A., Nikolaidis, N., and Pitas, I. Computer vision for autonomous uav flight safety: An overview and a vision-based safe landing pipeline example. *Acm Computing Surveys (Csur) 54*, 9 (2021), 1–37.

[92] Kaljahi, M. A., Shivakumara, P., Idris, M. Y. I., Anisi, M. H., Lu, T., Blumenstein, M., and Noor, N. M. An automatic zone detection system for safe landing of uavs. *Expert systems with applications 122* (2019), 319–333.

[93] Kanellakis, C., and Nikolakopoulos, G. Survey on computer vision for uavs: Current developments and trends. *Journal of Intelligent & Robotic Systems 87* (2017), 141–168.

[94] Kass, M., Witkin, A., and Terzopoulos, D. Snakes: Active contour models. *International journal of computer vision 1*, 4 (1988), 321–331.

[95] Keawboontan, T., and Thammawichai, M. Toward real-time uav multi-target tracking using joint detection and tracking. *IEEE Access 11* (2023), 65238–65254.

[96] Khan, A. T., and Jensen, S. M. Leaf-net: A unified framework for leaf extraction and analysis in multi-crop phenotyping using yolov11. *Agriculture 15*, 2 (2025), 196.

[97] Khanam, R., and Hussain, M. Yolov11: An overview of the key architectural enhancements. arxiv 2024. *arXiv preprint arXiv:2410.17725*.

[98] Khanam, R., and Hussain, M. Yolov11: An overview of the key architectural enhancements. *arXiv preprint arXiv:2410.17725* (2024).

[99] Kiefer, B., Quan, Y., and Zell, A. Memory maps for video object detection and tracking on uavs. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2023), IEEE, pp. 3040–3047.

[100] Kim, C., Lee, E. M., Choi, J., Jeon, J., Kim, S., and Myung, H. Roland: Robust landing of uav on moving platform using object detection and uwb based extended kalman filter. In *2021 21st International Conference on Control, Automation and Systems (ICCAS)* (2021), pp. 249–254.

[101] Kinahan, J., and Smeaton, A. F. Image segmentation to identify safe landing zones for unmanned aerial vehicles. *Irish Conference on Artificial Intelligence and Cognitive Science* (2021), 235–247.

[102] Kirillov, A., He, K., Girshick, R., Rother, C., and Dollar, P. Panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019).

[103] Kline, D. M., and Berardi, V. L. Revisiting squared-error and cross-entropy functions for training neural network classifiers. *Neural Computing & Applications 14* (2005), 310–318.

[104] Kuang, Q., Wu, J., Pan, J., and Zhou, B. Real-time uav path planning for autonomous urban scene reconstruction. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), IEEE, pp. 1156–1162.

[105] Lathuilière, S., Mesejo, P., Alameda-Pineda, X., and Horaud, R. A comprehensive analysis of deep regression. *IEEE Trans. on Pattern Analysis and Machine Intelligence 42*, 9 (2020), 2065–2081.

[106] Law, H., and Deng, J. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 734–750.

[107] Lee, D., Hess, D. J., and Heldeweg, M. A. Safety and privacy regulations for unmanned aerial vehicles: A multiple comparative analysis. *Technology in Society 71* (2022), 102079.

[108] Lee, M.-F. R., Chen, Y.-C., and Tsai, C.-Y. Deep learning-based human body posture recognition and tracking for unmanned aerial vehicles. *Processes 10*, 11 (2022), 2295.

[109] Li, C., et al. Yolov6: A single-stage object detection framework for industrial applications. *arXiv preprint arXiv:2209.02976* (2022).

[110] Li, L., and Lin, H.-T. Ordinal regression by extended binary classification. *Advances in neural information processing systems 19* (2006).

[111] Li, T., Liu, J., Zhang, W., Ni, Y., Wang, W., and Li, Z. Uav-human: A large benchmark for human behavior understanding with unmanned aerial vehicles. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2021), pp. 16266–16275.

[112] Li, X., Li, X., Li, Z., Xiong, X., Khyam, M. O., and Sun, C. Robust vehicle detection in high-resolution aerial images with imbalanced data. *IEEE Transactions on Artificial Intelligence 2*, 3 (2021), 238–250.

[113] Li, Y., Liu, M., and Jiang, D. Application of unmanned aerial vehicles in logistics: A literature review. *Sustainability 14*, 21 (2022), 14473.

[114] Liang, X., Shen, X., Feng, J., Lin, L., and Yan, S. Semantic object parsing with graph lstm. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14* (2016), Springer, pp. 125–143.

[115] Lim, J., Kim, M., Yoo, H., and Lee, J. Autonomous multirotor uav search and landing on safe spots based on combined semantic and depth information from an onboard camera and lidar. *IEEE/ASME Transactions on Mechatronics* (2024).

[116] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 2117–2125.

[117] Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 2980–2988.

[118] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)* (2014), Springer, pp. 740–755.

[119] Liu, D., et al. A survey of model compression techniques: past, present, and future. *Frontiers in Robotics and AI 12* (2025).

[120] Liu, F., Shan, J., Xiong, B., and Fang, Z. A real-time and multi-sensor-based landing area recognition system for uavs. *Drones 6*, 5 (2022), 118.

[121] Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., and Pietikäinen, M. Deep learning for generic object detection: A survey. *International journal of computer vision 128* (2020), 261–318.

[122] Liu, S., Qi, L., Qin, H., Shi, J., and Jia, J. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 8759–8768.

[123] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14* (2016), Springer, pp. 21–37.

[124] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In

Proceedings of the IEEE/CVF international conference on computer vision (2021), pp. 10012–10022.

[125] Liu, Z., Wang, C., Chen, K., and Meng, W. Vision-based autonomous landing on unprepared field with rugged surface. *IEEE Sensors Journal 22*, 18 (2022), 17914–17923.

[126] Loera-Ponce, J. A., Mercado-Ravell, D. A., Becerra-Durán, I., and Valentin-Coronado, L. M. Risk assessment for autonomous landing in urban environments using semantic segmentation. *arXiv preprint arXiv:2410.12988* (2024).

[127] Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 3431–3440.

[128] Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 3431–3440.

[129] Loureiro, G., Dias, A., and Martins, A. Survey of approaches for emergency landing spot detection with unmanned aerial vehicles. *Proceedings of the Robots in Human Life—CLAWAR* (2020), 129–136.

[130] Lowe, D. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision* (1999), vol. 2, pp. 1150–1157 vol.2.

[131] Lu, W., Zhang, Z., and Nguyen, M. A lightweight cnn–transformer network with laplacian loss for low-altitude uav imagery semantic segmentation. *IEEE Transactions on Geoscience and Remote Sensing 62* (2024), 1–20.

[132] Lu, Y., Xue, Z., Xia, G.-S., and Zhang, L. A survey on vision-based uav navigation. *Geo-spatial information science 21*, 1 (2018), 21–32.

[133] Luc, P., Couprie, C., Chintala, S., and Verbeek, J. Semantic segmentation using adversarial networks. *arXiv preprint arXiv:1611.08408* (2016).

[134] Luiten, J., Osep, A., Dendorfer, P., Torr, P., Geiger, A., Leal-Taixé, L., and Leibe, B. Hota: A higher order metric for evaluating multi-object tracking. *International journal of computer vision 129*, 2 (2021), 548–578.

[135] Luo, Y., Wang, X., Liao, Y., Fu, Q., Shu, C., Wu, Y., and He, Y. A review of homography estimation: advances and challenges. *Electronics 12*, 24 (2023), 4977.

[136] Lyu, Y., Vosselman, G., Xia, G.-S., Yilmaz, A., and Yang, M. Y. Uavid: A semantic segmentation dataset for uav imagery. *ISPRS Journal of Photogrammetry and Remote Sensing 165* (2020), 108 – 119.

[137] Ma, M.-Y., Shen, S.-E., and Huang, Y.-C. Enhancing uav visual landing recognition with yolo's object detection by onboard edge computing. *Sensors 23*, 21 (2023), 8999.

[138] Madhuanand, L., Nex, F., and Yang, M. Y. Self-supervised monocular depth estimation from oblique uav videos. *ISPRS journal of photogrammetry and remote sensing 176* (2021), 1–14.

[139] Marcu, A., Costea, D., Licaret, V., Pirvu, M., Leordeanu, M., and Slusanschi, E. Safeuav: Learning to estimate depth and safe landing areas for uavs from synthetic data. In *European Conference on Computer Vision (ECCV) UAVision Workshop* (2018).

[140] Masmoudi, N., Jaafar, W., Cherif, S., Abderrazak, J. B., and Yanikomeroglu, H. Uav-based crowd surveillance in post covid-19 era. *Ieee Access 9* (2021), 162276–162290.

[141] McCartney, E. *Optics of the atmosphere: scattering by molecules and particles*. John Wiley and Sons, 1976.

[142] McCullagh, P. Regression models for ordinal data. *Journal of the Royal Statistical Society: Series B (Methodological) 42*, 2 (1980), 109–127.

[143] Messaoudi, K., Baz, A., Sami Oubbati, O., Rachedi, A., Bendouma, T., and Atiquzzaman, M. Ugv charging stations for uav-assisted aoi-aware data collection. *IEEE Transactions on Cognitive Communications and Networking 10*, 6 (2024), 2325–2343.

[144] Micheal, A. A., Vani, K., Sanjeevi, S., and Lin, C.-H. Object detection and tracking with uav data using deep learning. *Journal of the Indian Society of Remote Sensing 49* (2021), 463–469.

[145] Miller, A., Miller, B., Popov, A., and Stepanyan, K. Uav landing based on the optical flow videonavigation. *Sensors 19*, 6 (2019).

[146] Milletari, F., Navab, N., and Ahmadi, S.-A. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)* (2016), Ieee, pp. 565–571.

[147] Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., and Terzopoulos, D. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence 44*, 7 (2022), 3523–3542.

[148] Ming, Y., Meng, X., Fan, C., and Yu, H. Deep learning for monocular depth estimation: A review. *Neurocomputing 438* (2021), 14–33.

[149] Mittal, M., Valada, A., and Burgard, W. Vision-based autonomous landing in catastrophe-struck environments. *arXiv preprint arXiv:1809.05700* (2018).

[150] Mohsan, S. A. H., Othman, N. Q. H., Li, Y., Alsharif, M. H., and Khan, M. A. Unmanned aerial vehicles (uavs): practical aspects, applications, open challenges, security issues, and future trends. *Intelligent Service Robotics* (2023), 1–29.

[151] Morales-Navarro, N., Osuna-Coutiño, J. d. J., Pérez-Patricio, M., Camas-Anzueto, J., Velázquez-González, J. R., Aguilar-González, A., Ocaña-Valenzuela, E. A., and Ibarra-de-la Garza, J.-B. Three-dimensional landing zone segmentation in urbanized aerial images from depth information using a deep neural network–superpixel approach. *Sensors 25*, 8 (2025), 2517.

[152] Morales-Navarro, N., Osuna-Coutiño, J. d. J., Pérez-Patricio, M., Camas-Anzueto, J., Velázquez-González, J. R., Aguilar-González,

A., Ocaña-Valenzuela, E. A., and Ibarra-de-la Garza, J.-B. Three-dimensional landing zone segmentation in urbanized aerial images from depth information using a deep neural network–superpixel approach. *Sensors 25*, 8 (2025), 2517.

[153] Morales-Navarro, N., Osuna-Coutiño, J. d. J., Pérez-Patricio, M., Camas-Anzueto, J., Velázquez-González, J. R., Aguilar-González, A., Ocaña-Valenzuela, E. A., and Ibarra-de-la Garza, J.-B. Three-dimensional landing zone segmentation in urbanized aerial images from depth information using a deep neural network–superpixel approach. *Sensors 25*, 8 (2025), 2517.

[154] Najman, L., and Schmitt, M. Watershed of a continuous function. *Signal processing 38*, 1 (1994), 99–112.

[155] Nigam, I., Huang, C., and Ramanan, D. Ensemble knowledge transfer for semantic segmentation. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018), IEEE, pp. 1499–1508.

[156] Niu, Z., Zhou, M., Wang, L., Gao, X., and Hua, G. Ordinal regression with multiple output cnn for age estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 4920–4928.

[157] Nock, R., and Nielsen, F. Statistical region merging. *IEEE Transactions on pattern analysis and machine intelligence 26*, 11 (2004), 1452–1458.

[158] Nouboukpo, A., and Allili, M. S. Spatially-coherent segmentation using hierarchical gaussian mixture reduction based on Cauchy-Schwarz divergence. *Int'l Conf. on Image Analysis and Recognition* (2019), 388–396.

[159] Nousi, P., Mademlis, I., Karakostas, I., Tefas, A., and Pitas, I. Embedded uav real-time visual object detection and tracking. In *2019 IEEE International Conference on Real-time Computing and Robotics (RCAR)* (2019), IEEE, pp. 708–713.

[160] Ortega, L. D., Loyaga, E. S., Cruz, P. J., Lema, H. P., Abad, J., and Valencia, E. A. Low-cost computer-vision-based embedded systems for uavs. *Robotics 12*, 6 (2023), 145.

[161] Otsu, N., et al. A threshold selection method from gray-level histograms. *Automatica 11*, 285-296 (1975), 23–27.

[162] Oubbati, O. S., Badis, H., Rachedi, A., Lakas, A., and Lorenz, P. Multi-uav assisted network coverage optimization for rescue operations using reinforcement learning. In *IEEE 20th Consumer Communications Networking Conference* (2023), pp. 1003–1008.

[163] Pal, O. K., Shovon, M., Mridha, M., and Shin, J. In-depth review of ai-enabled unmanned aerial vehicles: trends, vision, and challenges. *Discover Artificial Intelligence 4*, 1 (2024), 1–24.

[164] Papaioannidis, C., Mademlis, I., and Pitas, I. Autonomous uav safety by visual human crowd detection using multi-task deep neural networks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)* (2021), IEEE, pp. 11074–11080.

[165] Peñarroya, P., Centuori, S., Sanjurjo, M., and Hermosín, P. A lidar-less approach to autonomous hazard detection and avoidance systems based on semantic segmentation. *Celestial Mechanics and Dynamical Astronomy 135*, 3 (2023), 34.

[166] Plath, N., Toussaint, M., and Nakajima, S. Multi-class image segmentation using conditional random fields and global classification. In *Proceedings of the 26th annual international conference on machine learning* (2009), pp. 817–824.

[167] Polania, L. F., Fung, G. M., and Wang, D. Ordinal regression using noisy pairwise comparisons for body mass index range estimation. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2019), IEEE, pp. 782–790.

[168] Putranto, H. Y., Irfansyah, A. N., and Attamimi, M. Identification of safe landing areas with semantic segmentation and contour detection for delivery uav. In *2022 9th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)* (2022), IEEE, pp. 254–257.

[169] Qian, Y., Wang, Z., Gao, Y., Zhang, W., and Wang, H. A multi-object tracking method in moving uav based on iou matching. *IEEE Access* (2024).

[170] Rabiu, L., Ahmad, A., and Gohari, A. Advancements of unmanned aerial vehicle technology in the realm of applied sciences and engineering: A review. *Journal of Advanced Research in Applied Sciences and Engineering Technology 40*, 2 (2024), 74–95.

[171] Rahman, M. A., and Wang, Y. Optimizing intersection-over-union in deep neural networks for image segmentation. In *International symposium on visual computing* (2016), Springer, pp. 234–244.

[172] Rahmun, M., Deb, T., Bijoy, S. A., and Raha, M. H. Uav-crowd: Violent and non-violent crowd activity simulator from the perspective of uav. *arXiv preprint arXiv:2208.06702* (2022).

[173] Rallabandi, S., Madhan, V., Telaprolu, A., and Nazeer, M. Improved stnnet: A benchmark from detection, tracking and counting crowds using drones. In *2024 IEEE 9th International Conference for Convergence in Technology (I2CT)* (2024), IEEE, pp. 1–6.

[174] Ram Prasad, P., Pankaj Kumar, S., Fabio, N., Carmen, B., and Sambit, B. Monocular vision-aided depth measurement from rgb images for autonomous uav navigation. *ACM Trans. on Multimedia Computing, Communications and Applications 20(2)* (2023), Article 37.

[175] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 779–788.

[176] Redmon, J., and Farhadi, A. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 7263–7271.

[177] Redmon, J., and Farhadi, A. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018).

[178] Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence 39*, 6 (2016), 1137–1149.

[179] Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence 39*, 6 (2017), 1137–1149.

[180] Rezaee, M. R., Hamid, N. A. W. A., Hussin, M., and Zukarnain, Z. A. Comprehensive review of drones collision avoidance schemes: Challenges and open issues. *IEEE Transactions on Intelligent Transportation Systems* (2024).

[181] Robicquet, A., Sadeghian, A., Alahi, A., and Savarese, S. Learning social etiquette: Human trajectory understanding in crowded scenes. In *European conference on computer vision* (2016), Springer, pp. 549–565.

[182] Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18* (2015), Springer, pp. 234–241.

[183] Saccani, D., Cecchin, L., and Fagiano, L. Multitrajectory model predictive control for safe uav navigation in an unknown environment. *IEEE Transactions on Control Systems Technology 31*, 5 (2022), 1982–1997.

[184] Safadinho, D., Ramos, J., Ribeiro, R., Filipe, V., Barroso, J., and Pereira, A. Uav landing using computer vision techniques for human detection. *Sensors 20*, 3 (2020), 613.

[185] Sakineh Abdollahzadeh, Mohand Saïd Allili, A. B. J.-F. L. Cvisual safety mapping for uav landings using ordinal regression networks. *IEEE Transactions on Artificial Intelligence* (2026).

[186] Salehi, S. S. M., Erdogmus, D., and Gholipour, A. Tversky loss function for image segmentation using 3d fully convolutional deep networks. In *International workshop on machine learning in medical imaging* (2017), Springer, pp. 379–387.

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

[188] Sary, I., Armin, E., and Andromeda, S. Performance comparison of yolov5 and yolov8 architectures in human detection using aerial images. ultima computing: Jurnal sistem komputer, 15 (1), 8-13, 2023.

[189] Sepahvand, S., et al. A novel fuzzy image-based uav landing using rgbd data and visual slam. *Drones 8*, 10 (2024).

[190] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229* (2013).

[191] Shah Alam, M., and Oluoch, J. A survey of safe landing zone detection techniques for autonomous unmanned aerial vehicles (uavs). *Expert Systems with Applications 179* (2021), 115091.

[192] Shakhatreh, H., Sawalmeh, A., Al-Fuqaha, A., Dou, Z., Almaita, E., Khalil, I., Othman, N., Khreishah, A., and Guizani, M. Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *IEEE Access 7* (2019), 48572–48634.

[193] Shao, Z., Cheng, G., Ma, J., Wang, Z., Wang, J., and Li, D. Real-time and accurate uav pedestrian detection for social distancing monitoring in covid-19 pandemic. *IEEE Transactions on Multimedia 24* (2022), 2069–2083.

[194] Shen, L., and Joshi, A. K. Ranking and reranking with perceptron. *Machine Learning 60* (2005), 73–96.

[195] Shin, Y. H., Lee, S., and Seo, J. Autonomous safe landing-area determination for rotorcraft uavs using multiple ir-uwb radars. *Aerospace Science and Technology 69* (2017), 617–624.

[196] Shukla, P., Shukla, S., and Singh, A. K. Trajectory-prediction techniques for unmanned aerial vehicles (uavs): A comprehensive survey. *IEEE Communications Surveys & Tutorials* (2024).

[197] Singh, J., Adwani, N., Kandath, H., and Krishna, K. M. Rhfsafeuav: Real-time heuristic framework for safe landing of uavs in dynamic scenarios. In *2023 International Conference on Unmanned Aircraft Systems (ICUAS)* (2023), IEEE, pp. 863–870.

[198] Sinha, R., Pandey, R., and Pattnaik, R. Deep learning for computer vision tasks: A review. arxiv 2018. *arXiv preprint arXiv:1804.03928*.

[199] Springer, J., Guðmundsson, G. Þ., and Kyas, M. Toward appearance-based autonomous landing site identification for multirotor drones in unstructured environments. In *International Conference on Multimedia Modeling* (2025), Springer, pp. 198–211.

[200] Starck, J.-L., Elad, M., and Donoho, D. L. Image decomposition via the combination of sparse representations and a variational approach. *IEEE transactions on image processing 14*, 10 (2005), 1570–1582.

[201] Sun, L., Yang, Z., Zhang, J., Fu, Z., and He, Z. Visual object tracking for unmanned aerial vehicles based on the template-driven siamese network. *Remote Sensing 14*, 7 (2022), 1584.

[202] Swain, M. J., and Ballard, D. H. Color indexing. *International journal of computer vision 7*, 1 (1991), 11–32.

[203] Szeliski, R. *Computer vision: algorithms and applications*. Springer Nature, 2022.

[204] Taghanaki, S. A., Zheng, Y., Zhou, S. K., Georgescu, B., Sharma, P., Xu, D., Comaniciu, D., and Hamarneh, G. Combo loss: Handling input and output imbalance in multi-organ segmentation. *Computerized Medical Imaging and Graphics 75* (2019), 24–33.

[205] Tang, G., Ni, J., Zhao, Y., Gu, Y., and Cao, W. A survey of object detection for uavs based on deep learning. *Remote Sensing 16*, 1 (2023), 149.

[206] Telegraph, K., and Kyrkou, C. Spatiotemporal object detection for improved aerial vehicle detection in traffic monitoring. *IEEE Transactions on Artificial Intelligence* (2024), 1–13.

[207] Tong, P., Yang, X., Yang, Y., Liu, W., and Wu, P. Multi-uav collaborative absolute vision positioning and navigation: A survey and discussion. *Drones 7*, 4 (2023).

[208] Trigka, M., and Dritsas, E. A comprehensive survey of machine learning techniques and models for object detection. *Sensors 25*, 1 (2025), 214.

[209] Tutz, G. Ordinal regression: A review and a taxonomy of models. *Wiley Interdisciplinary Reviews: Computational Statistics 14*, 2 (2022), e1545.

[210] Tzelepi, M., and Tefas, A. Human crowd detection for drone flight safety using convolutional neural networks. In *2017 25th European Signal Processing Conference (EUSIPCO)* (2017), pp. 743–747.

[211] Ultralytics. Yolov8 anchor-free bounding box prediction - issue 189, 2023. Accessed: March 5, 2025.

[212] Varghese, R., and Sambath, M. Yolov8: A novel object detection algorithm with enhanced performance and robustness. In *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)* (2024), IEEE, pp. 1–6.

[213] Viola, P., and Jones, M. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001* (2001), vol. 1, Ieee, pp. I–I.

[214] Visin, F., Ciccone, M., Romero, A., Kastner, K., Cho, K., Bengio, Y., Matteucci, M., and Courville, A. Reseg: A recurrent neural network-based model for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (2016), pp. 41–48.

[215] Voulodimos, A., Doulamis, N., Doulamis, A., and Protopapadakis, E. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience 2018* (2018).

[216] Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., et al. Yolov10: Real-time end-to-end object detection. *Advances in Neural Information Processing Systems 37* (2025), 107984–108011.

[217] Wang, C.-Y., Bochkovskiy, A., and Liao, H.-Y. M. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2023), pp. 7464–7475.

[218] Wang, C.-Y., Yeh, I.-H., and Mark Liao, H.-Y. Yolov9: Learning what you want to learn using programmable gradient information. In *European conference on computer vision* (2024), Springer, pp. 1–21.

[219] Wang, R., Lei, T., Cui, R., Zhang, B., Meng, H., and Nandi, A. K. Medical image segmentation using deep learning: A survey. *IET Image Processing 16*, 5 (2022), 1243–1267.

[220] Wang, S., Jiang, F., Zhang, B., Ma, R., and Hao, Q. Development of uav-based target tracking and recognition systems. *IEEE Transactions on Intelligent Transportation Systems 21*, 8 (2019), 3409–3422.

[221] Wang, Y., Wang, H., Liu, Y., Wu, J., and Lun, Y. 6-DOF UAV Path planning and tracking control for obstacle avoidance: a deep learning-based integrated approach. *Aerospace Science and Technology 151* (2024), 109320.

[222] Wu, D. Object detection and tracking for drones: A system design using dynamic visual slam. *Applied and Computational Engineering 81* (2024), 71–82.

[223] Wu, J., Zhang, Z., and Huang, W. Semantic map construction of uav autonomous landing in unknown environment. In *2024 36th Chinese Control and Decision Conference (CCDC)* (2024), pp. 5018–5025.

[224] Wu, X., Sahoo, D., and Hoi, S. C. Recent advances in deep learning for object detection. *Neurocomputing 396* (2020), 39–64.

[225] Xie, Y., Yang, B., Guan, Q., Zhang, J., Wu, Q., and Xia, Y. Attention mechanisms in medical image segmentation: A survey. *https://doi.org/10.48550/arXiv.2305.17937* (2023).

[226] Xin, L., Tang, Z., Gai, W., and Liu, H. Vision-based autonomous landing for the uav: A review. *Aerospace 9*, 11 (2022), 634.

[227] Xue, Y., Xu, T., Zhang, H., Long, L. R., and Huang, X. Segan: Adversarial network with multi-scale l 1 loss for medical image segmentation. *Neuroinformatics 16* (2018), 383–392.

[228] Yan, L., Qi, J., Wang, M., Wu, C., and Xin, J. A safe landing site selection method of uavs based on lidar point clouds. In *2020 39th Chinese Control Conference (CCC)* (2020), IEEE, pp. 6497–6502.

[229] Yang, L., Kang, B., Huang, Z., Xu, X., Feng, J., and Zhao, H. Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR* (2024).

[230] Yang, L., Kang, B., Huang, Z., Zhao, Z., Xu, X., Feng, J., and Zhao, H. Depth anything v2. *arXiv:2406.09414* (2024).

[231] Yang, L., and Wang, L. An optimization-based selection approach of landing sites for swarm unmanned aerial vehicles in unknown environments. *Expert Systems with Applications 204* (2022), 117582.

[232] Yang, L., Ye, J., Zhang, Y., Wang, L., and Qiu, C. A semantic slam-based method for navigation and landing of uavs in indoor environments. *Knowledge-Based Systems 293* (2024), 111693.

[233] Yuan, Y., Chen, X., and Wang, J. Object-contextual representations for semantic segmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16* (2020), Springer, pp. 173–190.

[234] Yuan, Y., Wu, Y., Zhao, L., Chen, H., and Zhang, Y. Multiple object detection and tracking from drone videos based on gm-yolo and multi-tracker. *Image and Vision Computing 143* (2024), 104951.

[235] Yuan, Y., Wu, Y., Zhao, L., Pang, Y., and Liu, Y. Uav-ttracker: Multi-object tracking of uav video by transformer and new matching pattern. In *2024 IEEE 4th International Conference on Power, Electronics and Computer Applications (ICPECA)* (2024), IEEE, pp. 546–551.

[236] Yusuf, M. O., Hanzla, M., Rahman, H., Saqid, T., Al Mudawi, N., Almujally, N. A., and Algarni, A. Enhancing vehicle detection and tracking in uav imagery: a pixel labeling and particle filter approach. *IEEE Access* (2024).

[237] Zanaty, E., and Ghoniemy, S. Medical image segmentation techniques: an overview. *International Journal of informatics and medical data processing 1*, 1 (2016), 16–37.

[238] Zeineldeen, M., Zeyer, A., Schlüter, R., and Ney, H. Chunked attention-based encoder-decoder model for streaming speech recognition. In *ICASSP 2024-*

2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2024), IEEE, pp. 11331–11335.

[239] Zetout, A., and Allili, M. S. Csdnet: Context-aware segmentation of disaster aerial imagery using detection-guided features and lightweight transformers. *Remote Sensing 17*, 14 (2025).

[240] Zhang, Y., Yan, C., Xiao, J., and Feroskhan, M. Npe-drl: Enhancing perception constrained obstacle avoidance with non-expert policy guided reinforcement learning. *IEEE Transactions on Artificial Intelligence* (2024), 1–15.

[241] Zhang, Z., Zhang, Y., Xiang, S., and Wei, L. Kdp-net: An efficient semantic segmentation network for emergency landing of unmanned aerial vehicles. *Drones 8*, 2 (2024), 46.

[242] Zhang, Z., and Zhu, L. A review on unmanned aerial vehicle remote sensing: Platforms, sensors, data processing methods, and applications. *Drones 7*, 6 (2023).

[243] Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 2881–2890.

[244] Zheng, Y., Luo, J., Qiao, Y., and Gao, H. Uav-assisted traffic speed prediction via gray relational analysis and deep learning. *Drones 7*, 6 (2023), 372.

[245] Zhu, P., Wen, L., Du, D., Bian, X., Fan, H., Hu, Q., and Ling, H. Detection and tracking meet drones challenge. *IEEE transactions on pattern analysis and machine intelligence 44*, 11 (2021), 7380–7399.

[246] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., and Dai, J. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159* (2020).

[247] Zou, C., Sun, Y., and Kong, L. Unmanned aerial vehicle landing on rugged terrain by on-board lidar–camera positioning system. *Applied Sciences 14*, 14 (2024).

[248] Zou, Z., Chen, K., Shi, Z., Guo, Y., and Ye, J. Object detection in 20 years: A survey. *Proceedings of the IEEE 111*, 3 (2023), 257–276.