Effective Image Enhancement and Fast Object Detection for Improved UAV Applications

Yikun Tian

In the fulfilment of the requirement for the degree of

Master of Philosophy

Centre for Signal and Image Processing

Department of Electronic and Electrical Engineering
University of Strathclyde, Glasgow

Supervised by

Dr. Hong Yue

10th September, 2023

Declaration of Authorship

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

The copyright of this thesis belongs to the author under the terms of the United Kingdom Copyright Acts as qualified by University of Strathclyde Regulation 3.50. Due acknowledgement must always be made of the use of any material contained in, or derived from, this thesis.

田祎琨

Yikun Tian

Sept. 10, 2023

Acknowledgements

Despite of the hard time during the pandemic, I almost cannot believe my master's journey comes to an end, eventually, from the University of Strathclyde. Looking back, there were so many challenges and difficulties, yet I have successfully passed through and achieve this wonderful target. I have to greatly thank all the people I met, who have helped me a lot in different aspects. Without their support, I cannot complete this task.

First of all, I would like to thank my First supervisor, Dr. Hong Yue, for her constant support and valuable guidance, especially to learn how to do research. Her rich experience has benefitted me to progress effectively and quickly in many skills and learning activities. To me, she is always knowledgeable and considerate, and allowing me to explore freely yet with constructive feedback to improve. I greatly thank for such valuable learning opportunities from her, which has helped the success of my studies.

Second, I would like to greatly thank my mentor, Professor Jinchang Ren, who was supposed to be my primary supervisor yet the situation changed due to his reallocation to a different university. However, he has continued to guide my study with the full support, in various ways, where his extraordinary knowledge and experience in the computer vision and machine learning field has guided my study in the right direction. I was always deeply impressed and touched by his care and rigorous academic attitude.

Thanks to all the colleagues and friends I met during the study, these include but not limited to Dr. Guoliang Xie, Mr Siyuan Chen, Dr. Yijun Yan, Mr Bin Zhao et al. Thanks for your patience to help and support me, and also the happy time we have spent together! These memorable moments have driven me to overcome the difficulties in the study and the loneliness when abroad, especially during the lockdown in pandemic.

Finally, I would like to thank particularly my parents for their continuous and heartfelt love, support and trust on me, for which I swear to pay back using my whole life to look after them. Nothing is more valuable than a sweet family together. Thanks again for all those who have helped me in my life!

Abstract

As an emerging field, unmanned aerial vehicles (UAVs) feature from interdisciplinary techniques in science, engineering and industrial sectors. The massive applications span from remote sensing, precision agriculture, marine inspection, coast guarding, environmental monitoring, natural resources monitoring, e.g. forest, land and river, and disaster assessment, to smart city, intelligent transportation and logistics and delivery.

With the fast growing demands from a wide range of application sectors, there is always a bottleneck how to improve the efficiency and efficacy of UAV in operation. Often, smart decision making is needed from the captured footages in a real-time manner, yet this is severely affected by the poor image quality, ineffective object detection and recognition models, and lack of robust and light models for supporting the edge computing and real deployment.

In this thesis, several innovative works have been focused and developed to tackle some of the above issues. First of all, considering the quality requirements of the UAV images, various approaches and models have been proposed, yet they focus on different aspects and produce inconsistent results. As such, the work in this thesis has been categorised into denoising and dehazing focused, followed by comprehensive evaluation in terms of both qualitative and quantitative assessment. These will provide valuable insights and useful guidance to help the end user and research community.

For fast and effective object detection and recognition, deep learning based models, especially the YOLO series, are popularly used. However, taking the YOLOv7 as the baseline, the performance is very much affected by a few factors, such as the low quality of the UAV images and the high-level of demanding of resources, leading to unsatisfactory performance in accuracy and processing speed. As a result, three major improvements, namely transformer, CIoULoss and the GhostBottleneck module, are introduced in this work to improve feature extraction, decision making in detection and recognition, and running efficiency. Comprehensive experiments on both publicly available and self-collected datasets have validated the efficiency and efficacy of the proposed algorithm.

In addition, to facilitate the real deployment such as edge computing scenarios, embedded implementation of the key algorithm modules is introduced. These include the creative implementation on the Xavier NX platform, in comparison to the standard workstation settings with the NVIDIA GPUs. As a result, it has demonstrated promising results with improved performance in reduced resources consumption of the CPU/GPU usage and enhanced frame rate of real-time processing to benefit the real-time deployment with the uncompromised edge computing.

Through these innovative investigation and development, a better understanding has been established on key challenges associated with UAV and Simultaneous Localisation and Mapping (SLAM) based applications, and possible solutions are presented.

Keywords: Unmanned aerial vehicles (UAV); Simultaneous Localisation and Mapping (SLAM); denoising; dehazing; object detection; object recognition; deep learning; YOLOv7; transformer; GhostBottleneck; scene matching; embedded implementation; Xavier NX; edge computing.

Contents

List of Figures

List of Tables

Acronyms

AI	Artificial Intelligence
AHE	Adaptive Histogram Equalisation
ANN	Artificial Neural Networks
AOD-Net	All-in-One Dehazing Network
AP	Average Precision
BM3D	Block Matching 3D
BN	Batch Normalization
CA	Channel Attention
CBDNet	Convolutional Blind Denoising Network
CLAHE	Contrast-Limited Adaptive Histogram Equalisation
CNN	Convolutional Neural Network
DCP	Dark Channel Prior
DCPDN	densely connected pyramid dehazing network
DL	Deep learning
DnCNN	Denoising Convolutional Neural Network
DPM	Deformable Part-based Model
E-ELAN	Efficient Layer Aggregation Network
FFA-Net	Feature fusion attention network
FFDNet	Fast and Flexible Denoising Network
FN	False negatives
FP	False positives
FPN	Feature Pyramid Network
FPS	Frames per second
GAN	Generative Adversarial Networks
GCANet	Gated context aggregation network
GPS	Global Positioning System
GridDehazeNet	Attention-based multi-scale defogging network
HardGAN	Haze-aware representation distillation GAN
HOG	Histogram of Oriented Gradients
INS	Inertial Navigation System
IMU	Inertial Measurement Units
LN	Layer Normalization

MEMS	Micro-Electro-Mechanical Systems
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NAFNet	Nonlinear Activation Free Network
NLM	Non-local Means
NMS	Non-maximum suppression
PSNR	Peak Signal-to-Noise Ratio
R-CNN/RCNN	Regions with CNN, or Region Based CNN
RGB	Red, Green and Blue
RMSE	Root Mean Squared Error
ROI	Region of Interest
RPN	Region Proposal Network
SCA	Spatial-Channel Attention
SEIF	Sparse Extended Information Filter
SHA	Single Haze-aware Attenuation
SiLU	Sigmoid Weighted Linear Unit
SLAM	Simultaneous Localisation and Mapping
SMNS	Scene Matching Navigation
SSD	Single shot multibox detector
SSIM	Structural Similarity (Index of Image)
SVO	Semi-direct monocular visual odometry
SVM	Support Vector Machine
TN	True negatives
TNRD	Nonlinear Reaction Diffusion
TOPS	trillion operations per second
TP	True Positives
UAV	Unmanned aerial vehicles
VO	Visual Odometry
VIO	Visual-Inertial Odometry
VDN	Variational Denoising Network
WNNM	Weighted Nuclear Norm Minimization
YOLO	You Only Look Once

Chapter 1. Introduction

1.1 Background and Motivation

An unmanned aerial vehicles (UAV), or a drone, is an unmanned aircraft that operates without a human pilot on board. UAVs can be powered by engines, featuring reusable, capable of sustained flight using aerodynamic lift, and capable of carrying a payload. UAVs can complete designated tasks either through remote control or autonomous planning, as depicted in Fig. 1.1, which illustrates typical applications. Compared to manned aircrafts, UAVs offer advantages such as compact size, lower cost, and the ability to perform tasks in dangerous and complex environments. However, as the use cases for drones continue to expand, they increasingly operate in complex and even unknown environments. Therefore, intelligent environmental perception is of paramount importance for the development of drone applications, representing a significant issue and challenge in the field of drone application development [1, 2].

Currently, the commonly used environmental perception technologies primarily include the Global Positioning System (GPS), inertial systems, and matching navigation systems [3][4]. Among these, the GPS relies on multi-satellite positioning technology to calculate its own position by processing received satellite signals. However, it is susceptible to electromagnetic interference when communicating with satellites and its positioning accuracy can degrade significantly due to signal obstruction caused by terrain features, buildings, and other factors [5]. Inertial navigation systems use self-attitude information as a navigation data source and determine the drone's position through cumulative data from its self-attitude. However, they suffer from static drift, which accumulates errors over time, resulting in inaccurate pose estimation and even errors [6]. Matching navigation involves aligning measured environmental information with a prior map to obtain the drone's position. However, establishing a complete and effective database is costly and challenging, particularly with changes in flight altitude and environmental conditions [7].

In contrast to the above methods, Simultaneous Localization and Mapping (SLAM) technology, which combines localisation and map building simultaneously, allows drones to perceive and model their surrounding environment during flight. It also estimates their motion and position using the generated environment model, providing better perception capabilities in unknown environments [8]. As a result, SLAM is currently the mainstream technology for intelligent environmental perception for UAVs [9].

(a) Agricultural and forestry protection (b) Geographic surveying and mapping

(c) Emergency rescue (d) Power infrastructure inspection

Figure 1.1. Typical applications of drones

Depending on the sensors used, SLAM technology can be divided into laser SLAM [10] and visual SLAM [11]. Laser SLAM employs laser radar as the sensor, resulting in highly accurate maps with mature and reliable technology. However, laser radar is expensive and has limited detection range (typically around 200 meters). Visual SLAM, on the other hand, uses cameras as the sensing sensors, which is far more cost-effective, to capture images of the surrounding environment, extracting abundant texture features and other information [3][9]. It matches features or other information between adjacent frames, estimates the motion between image frames based on the obtained matches, and finally optimizes and adjusts the built scene map. Visual SLAM offers advantages such as simplicity, ease of installation, low cost, and unlimited detection range. Therefore, it is widely used on intelligent drones with payload constraints, as it can use onboard cameras to construct maps and achieve positioning without the need for additional sensors, while providing rich, human-visible information [9].

However, existing visual SLAM methods are often susceptible to environmental factors such as lighting conditions and texture features, which can lead to interference and defects in the constructed environmental map. Furthermore, the computed camera trajectory frequently deviates from the actual values, resulting in poor accuracy. Additionally, inaccurate drone trajectory estimation can reduce the applicability and robustness of the algorithm. These issues hinder the technology's effectiveness in drone applications [12]. Therefore, for drone applications, research into visual SLAM techniques that offer better applicability and robust performance is of paramount importance [13].

1.2 Research Objectives and Scope of the Work

In visual SLAM based UAV navigation, one of the major challenges is the analysis of the UAV images, i.e. how to accurately extract the useful featured information and measurement for supporting the mapping and estimation in an efficient manner. However, due to the relatively poor quality of the UAV images caused by the harsh working environment, poor image contrast, low spatial resolution and severe noise, accurate and efficient mapping from images becomes very challenging. As such, how to enhance the image quality and improve the efficacy and efficiency of object detection and recognition are the selected technical bottlenecks to be tackled in this thesis.

Accordingly, the major research objectives of the thesis are defined as follows.

1) To provide a comprehensive survey of the field and identify the state of the arts in relevant topics;
2) To investigate typical models in enhancement of UAV images degraded by different factors and provide useful insights how they perform to satisfy the needs of SLAM based UAV applications;
3) To investigate and develop useful models and solutions for improving the efficiency and efficacy of existing object detection and recognition approaches and facilitate the edge computing based real-time deployment.

With these identified key tasks and objectives, SLAM based deployment will become more feasible and satisfy the needs from various application scenarios.

1.3 Research Methodology

According to the defined key research objectives, the corresponding research methodologies adopted are highlighted as follows.

1) Comprehensive survey to understand the research problem and state of the art, where comparative study is carried out to analyse relevant approaches and models in both qualitative and quantitative ways;
2) According to various degradations of the UAV images, mixed-models are applied accordingly to address the associated challenges in terms of denoising and dehazing, respectively. This can well tackle the properties of degradation for efficacy and efficiency;
3) For improving the efficiency and efficacy of object detection from UAV images, intensive experiments with different models will be implemented. Meanwhile, the effect of hardware acceleration will be validated in a case study, to further verify the effectiveness of the improved implementation of the deep learning model.
4) To further refine the models using the feedback received from peer-reviewing and also publish the results for increased value and impact.

1.4 Contributions

Following the defined research aims and objectives, the major contributions of this thesis are summarised as follows.

1) First of all, a comprehensive survey of related topics and challenges is summarised in Chapter 2, which will provide useful insights for the community to understand the technical background and associated difficulties. Relevant work was published in the 6th Int. Conf. on Machine Vision and Information Technology (CMVIT) in 2022 and received the Best Poster Award;
2) Considering the quality degradation of the UAV images, various approaches and models have been proposed, yet they focus on different aspects and produce inconsistent results. As such, relevant work has been categorised into

denoising and dehazing focused, followed by comprehensive evaluation in terms of both qualitative and quantitative assessment. Useful findings are derived to show how conventional approaches can still be useful when comparing with the deep learning models in denoising, yet various deep learning models demonstrate superior performance in different datasets or under different metrics that is worth further investigation. These will provide valuable insights and useful guidance to help the end user and research community. Relevant work was published in the 13th Brain-Inspired Cognitive Systems (BICS) Conference in Aug. 2023.

3) For fast and effective object detection and recognition, deep learning based models, especially the YOLO series, are popularly used. However, taking the YOLOv7 as the baseline, the performance is very much affected by a few factors, e.g. quality of the UAV images and high-level demanding of computational resources, leading to unsatisfactory poor performance in terms of the accuracy and processing speed. As a result, three major improvements, namely transformer, CIoULoss and the GhostBottleneck module, are introduced for improved feature extraction, decision making in detection and recognition, and running efficiency. Comprehensive experiments on both publicly available and self-collected datasets have fully validated the improved efficiency and efficacy of the proposed approach. Relevant work is under preparation as a journal submission to the MDPI Journal of Mathematics.

4) In addition, to facilitate the real-time deployment in edge computing scenarios, embedded implementation of the key algorithm modules is introduced. This includes the creative implementation on the Xavier NX platform, in comparison to the standard workstation settings with the NVIDIA GPUs. As a result, it has demonstrated promising results with improved performance in terms of reduced resources consumption of the CPU/GPU usage and enhanced frame rate of processing to benefit the real-time deployment at the edge side. This work will be included in the journal draft as mentioned above.

Herein the efficiency is to measure the computational speed, indicating the computational complexity of the specific algorithm. The efficacy, or effectiveness, indicates how effective the model can be applied to complete the detection or classification tasks. Specifically, the accuracy measures the percentage of correct detection for object detection or classification.

1.5 Thesis organisation

The remainder of the thesis is organised as follows:

Chapter 2: Literature Review. This chapter starts with an introduction to UAVs, providing insights into both fixed-wing and multi-rotor UAVs, emphasizing their shared requirements for autonomy. We then delve into the importance and significance of aerial image preprocessing, leading to discussions on the current state of research in image enhancement, image edge detection, and their relevance to image feature extraction. The chapter proceeds to explore the application of computer vision in UAVs, highlighting the challenges and necessity of visual navigation research as well as a comprehensive overview of SLAM applications in UAVs.

In Chapter 3, enhancement of UAV images is focused, where denoising and dehazing are addressed separately, using both publicly available and self-collected datasets. By categorising related techniques to conventional approaches in the spatial domain and transform domains and deep learning models, relevant models and approaches are reviewed in detail, followed by the quantitative evaluation and visual comparison. Useful findings are identified to demonstrate the superior performance from some of the conventional approaches as well as deep learning models. These will provide valuable insights to help the end users to understand relevant techniques and make the proper decision to meet their needs.

In Chapter 4, an improved YOLOv7 model is proposed for efficient and effective object detection and recognition from UAV images, again tested in both publicly available and self-collected datasets. First, the development of deep learning based object detection and recognition is reviewed. There are a few reasons to select the Yolo for object detection, which include i) real-time detection, ii) high accuracy, iii) a unified framework for detection and classification, iv) robustness and generalisation, and v) an end to end model for efficiency. Although YOLOv7 has achieved promising results in various applications, it shows limitations in UAV applications. Three major improvements, namely the transformer module, CIoU loss function and GhostBottleneck module, are introduced as additional features to improve the feature extraction, computational efficiency and robustness of the YOLOv7. Quantitative results in terms of the recall, precision and average precision are used to validate the efficacy of the proposed work. In addition, embedded implementation on the NVIDIA Xavier NX is detailed, along with the quantitative results to show the reduced resources consumption and improved frame rate of real-time processing.

Finally, Chapter 5 summarises the contributions and conclusions of the thesis, followed by the suggested future work to further improve the development of this emerging field.

1.6 Author's Publications

1.**Y. Tian**, H. Yue, B. Yang and J. Ren, "Unmanned Aerial Vehicle Visual Simultaneous Localization and Mapping: A Survey," Journal of Physics: Conference Series, vol. 2278, 6th Int. Conf. on Machine Vision and Information Technology (CMVIT), Feb. 2022, 10.1088/1742-6596/2278/1/012006 (**Best Poster Award)**.

2.**Y. Tian**, H. Yue, J. Ren, "Image Enhancement for UAV Visual SLAM Applications: Analysis and Evaluation", In Proc. the 13th International Conference on Brain-Inspired Cognitive Systems (BICS'23), Aug. 5-6, Kuala Lumpur, Malaysia, 2023.

3. **Y. Tian**, H. Yue, and J. Ren, "An improved YOLOv7 with embedded implementation for fast and effective object detection and recognition for UAV applications", Under preparation (to be submitted to IEEE Journal of Selected Topics in Applied Earth Observation and Remote Sensing (JSTARS).

Chapter 2. Literature Review

In this chapter, detailed background knowledge and related work is presented. These include the survey of the UAV techniques in Section 2.1, the summary of the workflow for aerial image processing in Section 2.2, discussions of the computer vision techniques in Section 2.3, and the visual SLAM analysis in Section 2.4. Finally, a short summary is provided in Section 2.5 to conclude the chapter.

2.1 Unmanned Aerial Vehicles (UAVs)

An Unmanned Aerial Vehicle (UAV), also known as a drone, is a flying vehicle operated without a human pilot on board. The entire flight process can be automatically controlled by radio remote control equipment and onboard computer programs. Drones were first conceived in 1914 when the United Kingdom, in response to wartime needs, proposed the use of radio remote control to guide small unmanned aircraft for bomb-dropping missions [14]. Over the course of more than a century, drones have seen multifaceted development, initially starting as target drones for military training and evolving into reconnaissance aircraft used for intelligence gathering.

Entering the 21st century, drones have experienced widespread adoption in civilian applications, playing vital roles in disaster response and relief efforts, environmental exploration, weather forecasting, agriculture and forestry protection, express delivery, aerial photography, power line inspections, and more. They have greatly enhanced convenience in both work and daily life [1][9].

Key characteristics of drones include ease of take-off and landing, adaptability to various mission requirements, and the ability to capture timely image data [1][2]. Drones exhibit remarkable endurance, even in harsh weather conditions, allowing them to effectively operate in hazardous areas for extended periods. They can transmit real-time target images, enabling the prompt acquisition of various intelligence information and situational awareness.

Drones are equipped with a variety of devices that enable them to perform diverse tasks in various scenarios [1-3]. Before 2008, the application of drones was primarily limited to the military sector. Over the past decade, with the advancement of

Micro-Electro-Mechanical Systems (MEMS) technology, drones have gradually become integrated into daily life, appearing in various forms tailored to different fields and tasks, such as photography, cinematography, power line inspections, and resource exploration. Drone technology has made significant progress in terms of structure, functionality, and navigation [13][15].

According to research data published by Drone Industry Insights, the global market for drones itself excluding any related applications was estimated to be approximately $22.5 billion in 2020 [16]. It is expected to exceed $48 billion by 2026, indicating substantial market growth potential. Drones will continue to evolve towards multi-task payloads, miniaturization of mission payloads, autonomous intelligent flight, and standardized drone management. Based on their structural design, drones can mainly be classified into rotary-wing drones and fixed-wing drones.

2.1.1 Fixed-Wing Unmanned Aircraft

Fixed-wing unmanned aircraft originated as early as 1917 [14]. In fact, the United States began utilizing fixed-wing drones for military reconnaissance as early as 1960, where they played a significant role. Fixed-wing unmanned aircraft consist of five primary components: wings, fuselage, tail, landing gear, and propulsion system. Compared to other drone structures, they offer advantages such as a strong payload capacity and extended flight times, making them widely employed in military and meteorological applications [2-3] [9] [12].

Based on the design of their wings, fixed-wing UAVs can harness the lift generated by airflow, reducing the power demand to counteract gravity, thus extending the drone's endurance. Analogous to the speed advantage of fixed-wing airplanes over helicopters, fixed-wing UAVs have a speed advantage compared to rotary-wing drones. These characteristics make them exceptionally suitable for long-distance travel. The extended range, increased flight time, and speed of fixed-wing drones make them an ideal choice for enthusiasts and surveillance professionals. Compared to other options in the market, fixed-wing drones offer higher performance but also require more complex operation, necessitating pilots with more extensive remote-control experience.

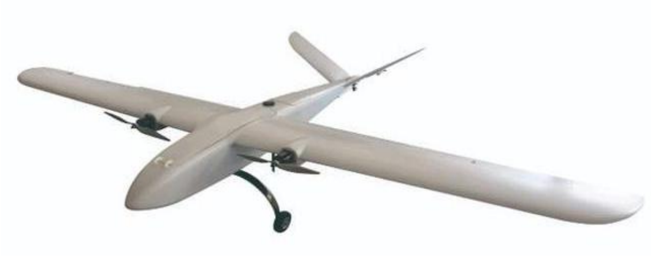


Figure 2.1. Schematic diagram of a fixed-wing UAV (online source)

2.1.2 Multi-Rotor Unmanned Aircraft

Rotorcraft UAVs draw their flight power from rotors and possess vertical take-off and landing capabilities, as well as omnidirectional flight capabilities [1-3][14]. They are highly maneuverable, compact, and adaptable to urban environments, but controlling them poses a significant challenge. Currently, in the field of rotorcraft unmanned aircraft, quadcopters are considered a successful model due to their symmetrical structure.

A quadcopter UAVs feature four rotors symmetrically distributed, allowing them to perform various maneuvers such as roll, pitch, and yaw by adjusting the rotor speeds, with relatively low control complexity due to the symmetrical design [14][17]. Presently, there are matured products with applications in multiple civilian sectors. For instance, in agriculture, drones can monitor crop growth, and in areas where ground-based crop protection equipment may struggle to operate, quadcopter drones can serve as alternatives for agricultural tasks.

Multi-rotor UAVs are characterized by their compact size and excellent maneuverability, making them a suitable choice for aerial photography. They can effortlessly hover and take off vertically, thereby enhancing their versatility. However, quadcopters, hexacopters, and octocopters also have their drawbacks. The addition of more rotors can make the drone more challenging to learn and control. Additionally, these moving components consume extra electrical power, leading to faster battery depletion. Since most multi-rotor drones typically have less than an hour of flight time, they are better suited for short-range applications such as photography and videography.

Figure 2.2: Illustration of a multi-rotor UAV (source: https://blog.csdn.net/modeling3D/article/details/115229274)

2.2 Aerial Image Processing

2.2.1 Importance and Significance

The primary purpose of image pre-processing is to eliminate irrelevant information in images, restore useful real data, simplify data to the maximum extent, and enhance the detectability of relevant information, thereby improving the usability and reliability of extracted features and further benefit image segmentation, matching, and recognition. Distortion correction, image enhancement, dehazing/noise reduction, and brightness adjustment, among other techniques, all fall within the scope of image pre-processing for drone imagery and require appropriate processing based on actual conditions.

Compared to indoor drones or mobile robots, drones in outdoor environments face a significant challenge in achieving safe and reliable visual navigation and positioning. Outdoor flight environments exhibit significant variations and are constrained by factors such as the drone's high altitude, small size, and lightweight, resulting in poor wind resistance and limitations in digital camera focal lengths [1][9]. Consequently, images taken in outdoor settings are prone to distortion, warping, tilting, zooming, and are susceptible to environmental variables like lighting and smoke. Moreover, the captured images are often small in size but have large data volumes, directly impacting image quality. Low-quality images can hinder the accuracy of visual positioning. Therefore, image pre-processing is essential before entering the image matching phase in visual SLAM [13][15][18].

Finding suitable methods to effectively remove rain, snow, haze, and other image-degrading factors while enhancing image details can improve the accuracy of feature recognition, laying a solid foundation for the subsequent image registration phase.

2.2.2 Image enhancement

With the continuous deterioration of the global environment, haze conditions are becoming increasingly severe. This not only poses health hazards to humans but also presents a significant threat to various fields such as aviation and transportation. Due to the presence of a large number of fine particles suspended in the haze, it not only reduces human visibility but also severely degrades the quality of images captured by outdoor imaging equipment, affecting areas such as road monitoring, aerial remote sensing, and military observation. These images often suffer from low contrast, low brightness, and poor clarity [19].

In the context of drone aerial photography, degraded images not only pose challenges for image pre-processing but, if not processed effectively, can also impede subsequent tasks such as target recognition and visual navigation. Therefore, it is crucial to perform effective dehazing and enhancement on images and videos captured in hazy weather conditions [20].

Image dehazing algorithms are image analysis and processing methods designed to meet specific scene requirements, highlight image details, and enhance image quality [21]. Based on their principles, dehazing algorithms can be categorized into three types: physics-based dehazing algorithms, image enhancement-based dehazing algorithms, and deep learning-based dehazing algorithms. The following Fig. 2.3 illustrates typical algorithms from each of these three categories.

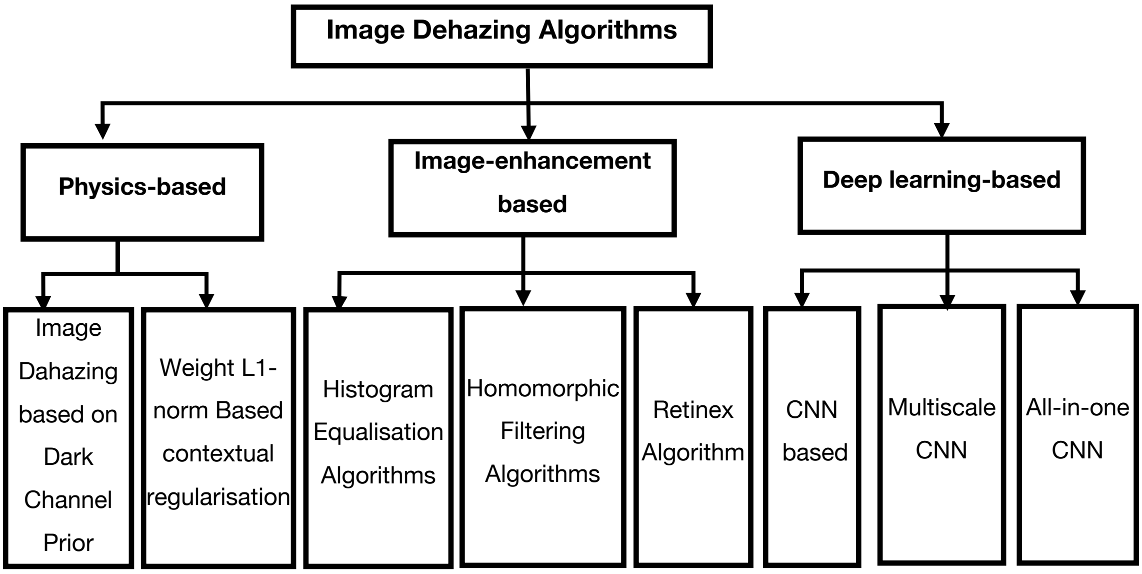


Image Dehazing Algorithms
Physics-based
Image-enhancement based
Deep learning-based
Image Dahazing based on Dark Channel Prior
Weight L1-norm Based contextual regularisation
Histogram Equalisation Algorithms
Homomorphic Filtering Algorithms
Retinex Algorithm
CNN based
Multiscale CNN
All-in-one CNN

Figure 2.3. Categorisation of typical image dehazing algorithms

(1) Physics-based algorithms are based on the attenuation of light propagation in hazy conditions. These algorithms construct a degradation model to restore clear images by measuring or estimating parameters such as atmospheric transmittance and scattering intensity. However, the restored images obtained by these algorithms often suffer from halo and artifacts [22].

(2) Image enhancement-based algorithms focus on processing regions of interest in hazy images, such as enhancing contrast to improve texture and clarity. These algorithms are relatively simple and fast but may yield poor results on images with limited colour information. Two typical algorithms in this category are the Homomorphic Filtering Algorithm [23] and the Retinex Algorithm [24].

The Homomorphic Filtering Algorithm [23] aims to correct the grayscale range of images, addressing uneven illumination in images, thereby enhancing details in dark areas without losing details in bright areas. It attempts to minimize the decrease in image quality caused by insufficient illumination by enhancing the high-frequency details of the image. The basic principle behind image homomorphic filtering is the reflection imaging principle during image generation.

As an image enhancement method, the Retinex Algorithm [24] relies on three basic assumptions [25]: (1) The real world is colourless. (2) Each colour domain is composed of the three primary colours, red, green, and blue, with fixed wavelengths. (3) Combinations of the three primary colours constitute the colour of each unit area. The foundation of the Retinex algorithm is that an object's different colours are determined by its ability to reflect light of different wavelengths. While illumination may exhibit non-uniformity, it does not affect the ultimate appearance of colours. The Retinex algorithm can achieve a balance between compression, edge enhancement, and colour constancy within a certain dynamic range, demonstrating good adaptive enhancement capabilities. The Retinex algorithm can further be categorized into single-scale Retinex algorithms, multi-scale weighted average Retinex algorithms, and multi-scale Retinex algorithms with colour restoration.

(3) Deep learning-based methods use simulated hazy image datasets to train neural network models for image restoration. Typically, various deep learning models based on different variations of the Convolutional Neural networks (CNNs) are used, such as DeHazeNet [26] and densely connected pyramid dehazing network (DCPDN) [27]. As these algorithms often rely heavily on the training data and the training strategy, which may have limited the generic adaptability to hazy images in real scenarios.

2.3 Computer Vision in UAV Applications

Computer vision is a scientific discipline that has emerged in recent years. It was first proposed by David Marr in the 1980s [28]. Marr believed that the theoretical approach to computer vision should start with images and extract the content of the real physical scene. Computer vision technology is a bio-inspired technology that, compared to traditional methods, can not only collect data from the surrounding environment but also simulate the human brain's processing of collected data. It operates at high speed and has a large storage capacity. While computer vision technology demonstrates powerful information processing capabilities in the application of artificial intelligence, it still differs in nature from human emotional analysis.

Driven by the development of society, UAVs have become a hot research topic in the field of artificial intelligence and have been widely used in various industries [13][15][18]. In the 1990s, research primarily focused on using computer vision technology for surface roughness detection of materials. At the same time, computer vision technology began to penetrate various fields, such as automotive driving, where it was used for real-time monitoring of vehicle speed and surroundings. At this time, the combination of computer vision and drones was virtually unknown. It wasn't until the early 21st century that the integration of computer vision technology with drone technology began to gain momentum. Computer vision could provide drones with rich visual information for relative positioning and orientation adjustments. The combination of computer vision technology and drones has been used in research on autonomous landing, attitude measurement, navigation, and obstacle avoidance for drones. To date, the integration of computer vision technology and drones remains a focal point of attention in the field of artificial intelligence, with significant development potential.

In recent years, with the improvement of computer capabilities and the rapid development of computer vision technology, drones have expanded from their initial military applications to other industries and have found widespread use [1]. Computer vision technology serves as the "eyes" of drones, providing them with a wealth of visual information for obtaining relative positions. Computer vision techniques analyse images captured by imaging sensors, extracting semantic content and information contained in the images for understanding and interpretation [2].

The integration of computer vision technology with drones initially aimed at aerial reconnaissance and geological surveying but has now been widely applied in various fields, including industrial inspection, geological mapping, disaster information acquisition, emergency supply delivery, agricultural and forestry protection, and unmanned combat, among others [3][5][6]. These application scenarios impose higher requirements on drone image processing, autonomous navigation, target recognition, and precision striking capabilities.

2.3.1 Object Detection and Tracking

Object detection and tracking have been prominent areas of interest in the field of computer vision in decades [3][29][30]. They involve the detection, recognition, and tracking of objects in image sequences containing moving targets, along with behaviour understanding and description [31]. Research in object analysis encompasses disciplines such as pattern recognition, image processing, computer vision, and AI. Combining UAVs with image sensors and motion-based object detection and tracking technologies allows for real-time tracking of targets locked by image sensors. This enables the acquisition of dynamic target information and accurate assessment of target behaviour, thus providing valuable real-time information for a wide range of applications [32].

With the advancement of convolutional networks, researchers have proposed numerous deep learning-based object detection methods, which can be broadly categorized into two types: Two-stage object detection algorithms based on candidate regions and One-stage object detection methods based on regression. In Two-stage detection (represented by region-based CNN, or the R-CNN [29] series), input images first go through a candidate box generation network and then through a classification network to classify the content of candidate boxes. Compared to One-stage methods, Two-stage methods offer higher accuracy but require more computation, resulting in slower processing. One-stage detection is an end-to-end object detection algorithm, such as the YOLO [30] series and single shot multibox detector (SSD) [33] series, which predicts object categories and their positions in a single step [15].

Proposed by Ross B. Girshick at the University of California, Berkeley in 2014 [29], R-CNN has surpassed the contemporary end-to-end method, the OverFeat algorithm proposed by Yann Lecun [34], in terms of performance. Its algorithmic structure has since become a classic framework for subsequent two-stage object detection.

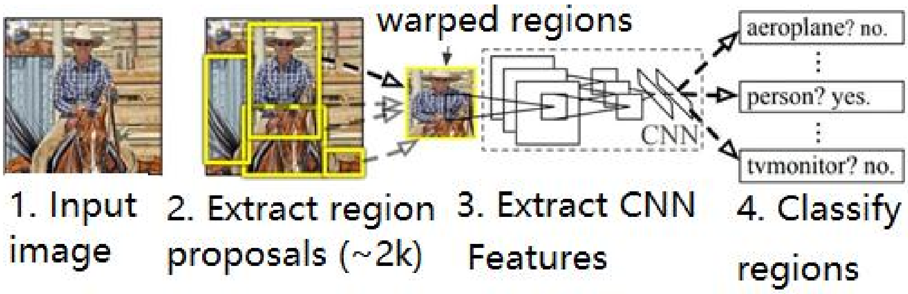


Figure 2.4. Diagram to illustrate the step-wise implementation of the R-CNN [29].

As shown in the Fig. 2.4, the specific steps of R-CNN can be summarized into the following four key stages [29].

1) Input the image, followed by certain pre-processing such as normalisation and denoising if applicable;
2) Region proposal generation: R-CNN utilizes the selective search method to generate approximately 2,000 candidate regions (region proposals) for the input image. Initially, the image is divided into several sub-regions, and then the similarity between these sub-regions is calculated, including factors such as colour and texture. Finally, these regions are merged to obtain approximately these candidate regions.
3) Region based feature extraction: Each candidate region is resized to a fixed size (224×224) and then fed into a CNN model (AlexNet convolutional network) to obtain a feature vector. AlexNet consists of 5 convolutional layers and 2 fully connected layers, resulting in a 4,096-dimensional feature vector for each small region. Since there are multiple small regions, this results in a matrix of size 4,096×2,000.
4) Classification: The feature vectors are input into a multi-class support vector machine (SVM) classifier to predict the probability of each class for objects within the candidate regions. A separate SVM classifier is trained for each class, where the probability of one class is associated to a particular class is inferred from the feature vector. The SVM classifier's weight matrix has a dimension of 4096×N, where N is the number of classes. Objects that are classified into a specific class have bounding boxes drawn around them, while those that cannot be classified are discarded. Finally, a few of the most likely bounding boxes are selected. To improve the localization accuracy, a bounding box is also trained with the regression model to refine the precise positions of the boxes.

The advantages of R-CNN include the ability to apply high-capacity CNN to bottom-up candidate regions for object localization and segmentation. Moreover, in cases where labelled training data is insufficient, it initially performs supervised pre-training on auxiliary tasks and then fine-tunes the model for the specific domain, resulting in significant performance improvements [18].

While R-CNN exhibits good recognition accuracy, it is computationally intensive. Currently, the most widely used series for object detection is YOLO [30]. YOLO redefines object detection as a regression problem, applying a single CNN to the entire image, dividing it into a grid of ROIs, and predicting class probabilities and bounding boxes for each grid cell. Since it treats detection as a regression problem, it eliminates the need for complex pipelines. YOLO is exceptionally fast, being 1000 times faster than "R-CNN" and 100 times faster than "Fast R-CNN" [35]. The diagram of the architecture of the YOLOv5 is illustrated in Fig. 2.5.

As seen in Fig. 2.5, the network architecture of YOLOv5 can be divided into four main parts as detailed below.

- **Input**: This includes Mosaic data augmentation, image resizing, and calculation of the adaptive anchor box. Mosaic data augmentation combines four images to enrich the background of the images. Image size processing adapts the original images of different aspect ratios with minimal added black borders to a standard size. Adaptive anchor box calculation updates anchor box values based on the differences between predicted and real boxes via iterative parameter updates.
- **Backbone**: This consists of BottleneckCSP and Focus modules. The BottleneckCSP module is to enhance the learning performance of the entire CNN while significantly reducing computational complexity. The Focus module slices the input channels, expanding them by a factor of four and generating downsampled feature maps through convolution, reducing computational load and improving speed.
- **Neck**: It combines the conventional FPN (Feature Pyramid Network) layers with a bottom-up feature pyramid, fusing semantic and positional features and integrating the backbone layers with detection layers to capture richer feature information.
- **Head**: The head outputs a vector containing class probabilities, object scores, and the position of the object's bounding box. The detection network comprises three detection layers, each responsible for detecting objects of different sizes. Each detection layer produces corresponding vectors, ultimately generating predicted bounding boxes and class labels for objects in the original image.

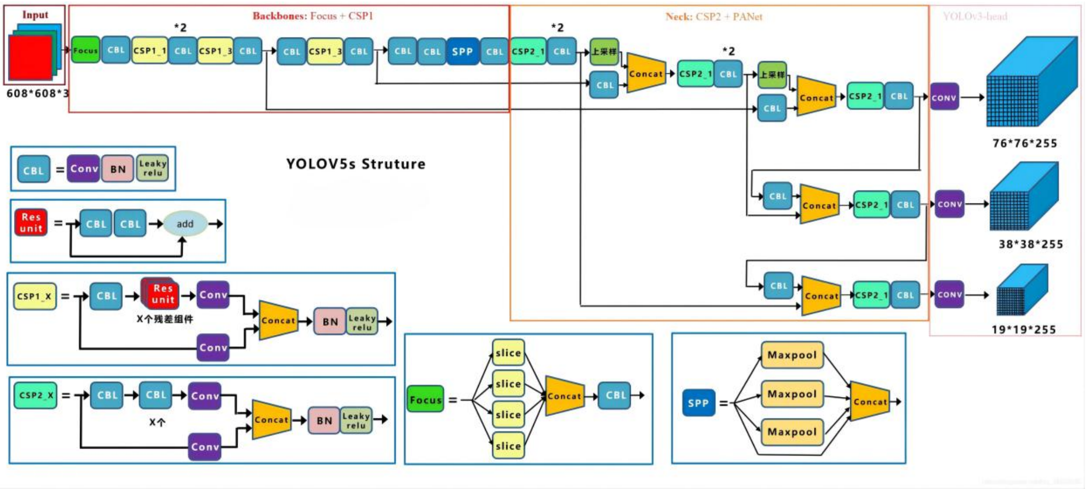


Figure 2.5. Diagram of the YOLOv5 architecture [36]

Object tracking can be divided into single-object tracking and multi-object tracking. The goal of tracking is to build a motion model that predicts the possible state of the object in the next frame, providing prior knowledge for estimating the object's state and finding the optimal object position in the current frame. Traditional object tracking relies on feature extraction methods combined with filtering-based search algorithms. Algorithms like MeanShift are based on probability density distribution and iteratively converge towards local peaks of the probability density distribution for tracking [37][38].

Particle Filter is based on particle distribution statistics [31], approximating probability density functions by finding a set of randomly sampled particles propagating in the state space. It uses the sample mean instead of integral calculations to obtain the minimum variance estimate of the system state. Kalman Filter [39][40], on the other hand, models and predicts the trajectory of the object's motion rather than modelling the object's intrinsic features. It estimates the object's potential location in the next frame.

Another category of the tracking algorithms is DL based, such as the YOLO series [30], SSD series [33], two-stage series [41], and more.

2.3.2 Obstacle Avoidance and Path Planning

With the increasing widespread applications of drones, safety concerns have become more prominent. When drones fly in complex urban environments with uncertain flying conditions, collision accidents are not uncommon.

This necessitates that drones possess autonomous obstacle avoidance and path planning capabilities, allowing them to promptly, accurately, and safely navigate around obstacles to reach designated target locations efficiently [42]. The two key components to achieving autonomous drone obstacle avoidance are obstacle detection and path planning. Obstacle detection refers to the drone's ability to accurately detect obstacles in its surroundings using its sensing devices. Path planning involves the drone using obstacle information and its flight conditions to chart an optimal path to its intended destination.

During the autonomous flight of a drone, encountering obstacles involves a three-phase process: first, the drone perceives the obstacle; second, it navigates around the obstacle; and third, it searches for a new flight path. In other words, the drone progresses from detecting an obstacle to autonomously navigating around it and then to autonomously planning a flight route.

In the first phase, the drone senses the presence of an obstacle, quickly identifies it, and hovers in place, awaiting further instructions. In the second phase, the drone accurately perceives the contours of the obstacle using sensors and then autonomously maneuvers around it. In the third phase, the drone, based on the environmental data it has gathered, utilizes algorithms to autonomously plan a flight route, thereby achieving the goal of autonomous obstacle avoidance [33][43].

Effective obstacle detection relies on accurate environmental perception. Compared to active detection methods such as ultrasonic, laser, and infrared, visual detection only requires passive reception of image information, making it more secure and less susceptible to signal and radiation interference [44]. It holds great potential in complex electromagnetic and geographic environments. Existing visual obstacle avoidance methods are mostly based on image segmentation, depth extraction, optical flow algorithms, among others [1-3]. Image segmentation methods offer high computational efficiency but lower detection accuracy. Depth extraction methods require constructing 3D maps and may not meet real-time obstacle avoidance requirements. Optical flow methods can balance real-time performance and accuracy, making them a preferable choice for visual obstacle avoidance.

Path planning is crucial in ensuring that drones successfully complete assigned tasks [45]. Geometry-based algorithms include artificial potential field methods, topological methods, etc. Biologically inspired and optimization-based algorithms include the A* algorithm [46] etc.

The basic idea of the potential artificial gravitational field (AGF) method for path planning is to abstract the motion of a robot in its surrounding environment as a type of movement within an AGF [47]. In this context, the goal point exerts "attraction" on the mobile robot, while obstacles generate "repulsion" forces. Ultimately, the robot's motion is controlled by calculating the resultant force. Paths generated by the AGF method are generally smooth and safe, however, similar to most of other optimisation methods, it still suffers from the problem of local minima.

This algorithm works best when obstacles are regular in shape; otherwise, it can lead to significant computational complexity or even become infeasible [44]. However, from another perspective, the artificial potential field method offers an elegant and concise mathematical description, making it still quite attractive. Its limitations become evident when obstacles are located near the goal, as the robot may never reach the destination.

In many previous studies, the goal and obstacles were positioned far apart [48]. As the robot approached the goal, the repulsive force from obstacles diminished significantly, sometimes even to the point of neglect, allowing the robot to be solely influenced by the attractive force and reach the goal directly. However, in many practical environments, at least one obstacle is often located very close to the goal [12][13][15]18]. In such cases, as the mobile robot approaches the goal, it also moves closer to the obstacle. Using the previously defined attraction and repulsion field functions, the repulsive force can become much greater than the attractive force. This makes the goal point no longer the global minimum of the entire potential field, rendering it impossible for the robot to reach the goal. This situation leads to the problem of local optima, making the design of the "attraction field" a crucial aspect of this method.

The A* algorithm is a heuristic search algorithm [46], where heuristic search involves establishing heuristic search rules during the process of search to calculate the distance relationship between the current location and the target one. This prioritizes the search direction towards the location of the target, ultimately leading to improved search efficiency.

In the context of drone path planning, it is not only essential to navigate swiftly and accurately around obstacles but also to prioritize the optimality, rationality, and completeness of the planned paths in future approaches [49]. There is potential to apply DL in this field as well, such as deep re-enforced learning [50].

2.3.3 Visual Navigation

Navigation systems provide critical support for the execution of flight missions by UAVs. The "2007-2032 Unmanned Aircraft Systems Roadmap" published by the U.S. Department of Defense [51] highlights the future requirements for UAVs, including autonomous flight capabilities, efficient comprehensive reconnaissance and sensing, and precision targeting. To achieve UAVs' autonomous flight, efficient reconnaissance, and precision targeting, accurate environmental perception, efficient image pre-processing, high-precision positioning, and navigation capabilities are indispensable.

Currently, UAV navigation systems are primarily represented by GPS and Inertial Navigation System (INS) [52]. GPS is a passive navigation system that is susceptible to environmental factors and external interference, limiting the autonomy of UAVs. INS accumulates errors over time, which can significantly impact navigation task execution. With the maturation of image processing and computer vision technologies, vision-based navigation methods have gradually been introduced into UAV systems [4][5]. Visual navigation, which uses environmental information as clues, leverages image sensors to perceive the surrounding environment and provides positioning and navigation for mobile carriers through specific calculations. It offers advantages such as real-time performance, minimal interference, rich information content, and good compatibility. In various environments, both indoors and outdoors, it can independently or as an auxiliary system provide navigation information for UAVs, making it a hot research topic in the UAV field.

From a technical perspective, visual navigation can generally be divided into three categories [53].

1) Matching Navigation: This approach involves a complete prior map in the computer, where real-time data is matched with the prior data to determine the current position.
2) Simultaneous Localization and Mapping (SLAM): SLAM does not rely on prior databases. It combines visual landmarks with its own motion to establish relative environmental information, thereby determining its relative position. Typical frameworks include ORB-SLAM2 [54], SVO [55], and others.
3) Active Search: This approach, which also doesn't rely on prior databases, separates visual perception from self-motion. Visual perception is used solely for target or feature recognition, while path information is obtained from self-attitude sensors. Within the defined path, it enables the search for target features.

Scene Matching Navigation (SMNS) [56] is characterized by its simple equipment structure, passive nature, and high positioning accuracy. It uses image sensors to match the area images near the UAV's flight or target area with stored reference images to obtain aircraft position data. As an auxiliary navigation method, when combined with inertial navigation systems, SMNS can form a highly autonomous navigation system with a high-precision, enabling precision targeting for UAVs [57]. Matching navigation can achieve absolute positioning of the aircraft. It compares the real-time scene map captured by the aircraft with the map data stored in the computer's memory. After a successful match, it retrieves the geographical location from the stored map database and provides real-time correction to the aircraft's inertial system for its current position.

2.4 Current Status in Visual SLAM Research

SLAM, initially proposed in the field of robotics by Chatila, Smith, and others [58][59], refers to the process in which a robot, starting from an unknown location in an unknown environment, continually observes environmental features during its motion to determine its own position and orientation. It then incrementally constructs a map of the surrounding environment based on its own position, achieving the simultaneous goals of localization and map construction. Due to its significant academic and practical value, SLAM has long been considered a key technology for achieving fully autonomous mobile robots. With this foundation, path planning can be performed, obstacles can be detected and avoided in real-time, ensuring safe operation [60].

Modern and popular visual SLAM systems can generally be divided into front-end and back-end components, as illustrated in the diagram below. The front-end is responsible for data association, equivalent to Visual Odometry (VO) [61]. It studies the transformation relationship between frames and primarily accomplishes real-time pose tracking. It processes input images, computes pose changes, and also detects and handles loop closures. When Inertial Measurements Units (IMU) information is available, it can also participate in fusion calculations (similar to Visual-Inertial Odometry, VIO). The back-end primarily optimizes the output results from the front-end. It utilizes filtering theories or optimisation techniques to perform tree or graph optimisation, ultimately obtaining the optimal pose estimation and mapping [62].

Due to limited payload capacity, there are two mainstream methods for position sensing in UAVs. The first approach aims to compensate for the relatively low accuracy of IMUs by combining them with visual SLAM systems to estimate the UAV's 3D spatial pose [63]. For instance, researchers e.g. Celik K have proposed monocular visual SLAM methods based on distance measurements, with a single camera mounted on UAVs [64]. In GPS-denied environments, these systems rely on monocular cameras for autonomous UAV navigation. Researchers at the Massachusetts Institute of Technology (MIT), such as Bethke and colleagues [65], have developed a vision-based multi-agent cooperative system that can detect both the position and the 3D velocity of the objects. This system offers excellent real-time tracking of targets. Valenti and others [66] have addressed the issues of pure visual rotational drift and potential loss of tracking during rapid movement by fusing RGB-D odometry and IMU data.

To leverage the high manoeuvrability and speed of UAVs and overcome the slow image acquisition and transmission rate of monocular cameras, sensors are combined with higher transmission rates, such as IMUs and camera sensors, to achieve better position and attitude estimation results [63]. This approach enables absolute scale estimation for monocular vision and fuses visual and inertial information for positioning. For instance, researchers at the University of Maryland, e.g. Conroy and colleagues [67], improved the localization of UAVs using cameras and ultrasonic sensors, enhancing optical flow algorithms. Researchers at the Queensland University of Technology in Australia, led by Milford Michael J [68], conducted in-depth research on algorithms such as visual odometry and visual expectation using monocular cameras. They combined these with the Rat SLAM algorithm to achieve UAV localization and navigation. Researchers at the Swiss Federal Institute of Technology in Zurich, led by Forster and colleagues [69], studied a micro-UAV equipped with a vertically oriented camera and an embedded minicomputer. They obtained positioning information using a combination of direct and feature-based semi-direct monocular visual odometry methods (SVO) methods. However, the accuracy of this approach is relatively low.

Both of the above-mentioned approaches utilize IMUs and monocular cameras for SLAM research. They require multiple sensors, which increases the complexity of the system and significantly raises the cost compared to purely camera-based visual SLAM methods [70][71].

2.5 Summary

This chapter provides an overview of the current status and applications of UAVs. It highlights the importance and significance of pre-processing aerial images, briefly introducing the importance and methods of image distortion correction, enhancement, and edge detection. The chapter then focuses on the application of computer vision techniques in UAV technology, covering scenarios such as target detection and tracking, visual navigation, autonomous obstacle avoidance, and path planning.

Furthermore, the chapter presents an in-depth summary of the current state of research in visual SLAM. It discusses both the front-end and back-end components of visual SLAM and their applications in UAVs. This groundwork sets the stage for the subsequent research, with identified knowledge gaps in image enhancement and object detection, especially how to denoising and dehazing the UAV captured images as well as how to achieve effective and efficient object detection accordingly. These will be addressed in detail in the following chapters.

Chapter 3. Image Enhancement: Analysis and Evaluations

3.1 Introduction

When capturing images with the UAV and aerial platforms, the imaging quality can be significantly affected by complex environmental conditions. Factors such as device shake and dust, both inside and outside the equipment, inevitably introduce various forms of noise into the resulting images. Such noise can interfere with the perception and understanding of the presented information by subsequent detection and recognition algorithms. In addition, environmental issues such as fogs may also cause significant degradation of the image quality, hence it has resulted in a new topic of image dehazing. Considering the inconsistent illumination and limited lighting, images under poor lighting conditions may also introduce new challenges.

With numerous models and approaches being proposed to tackle these issues, it has come to a question how effective they are when working with the real data. In this chapter, the aforementioned question will be answered by comprehensive evaluation and assessment of different approaches as detailed below.

3.2 Datasets and Evaluation Criteria

3.2.1. Datasets Description

For image denoising, the CBSD68 dataset [72] and the SIDD dataset [73] were used. The CBSD68 dataset consists of 68 colour images of varying sizes. The SIDD dataset includes approximately 30,000 noisy images captured under different lighting conditions using five representative smartphones, along with corresponding "noise-free" ground truth images. In addition, an own dataset including 2,035 virtual simulation scene images was also used for testing the effect of denoising. Some sample images are given in Fig. 3.1. Note that these images are not necessarily all captured by UAVs.

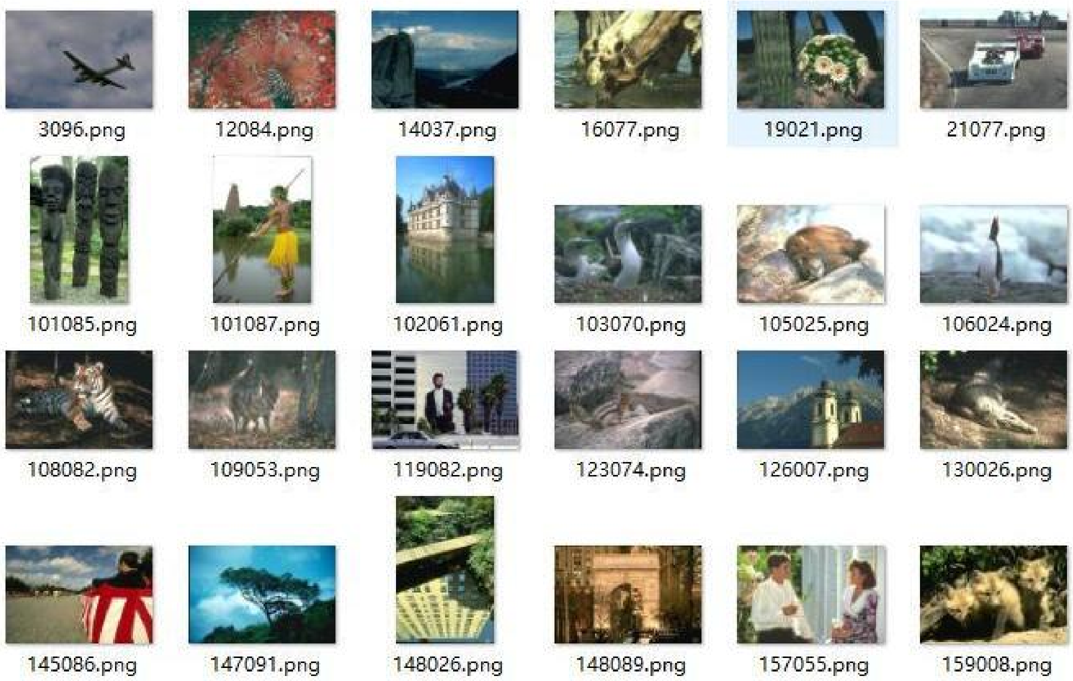


Figure 3.1 Sample images from the CBSD68 dataset for denoising testing.

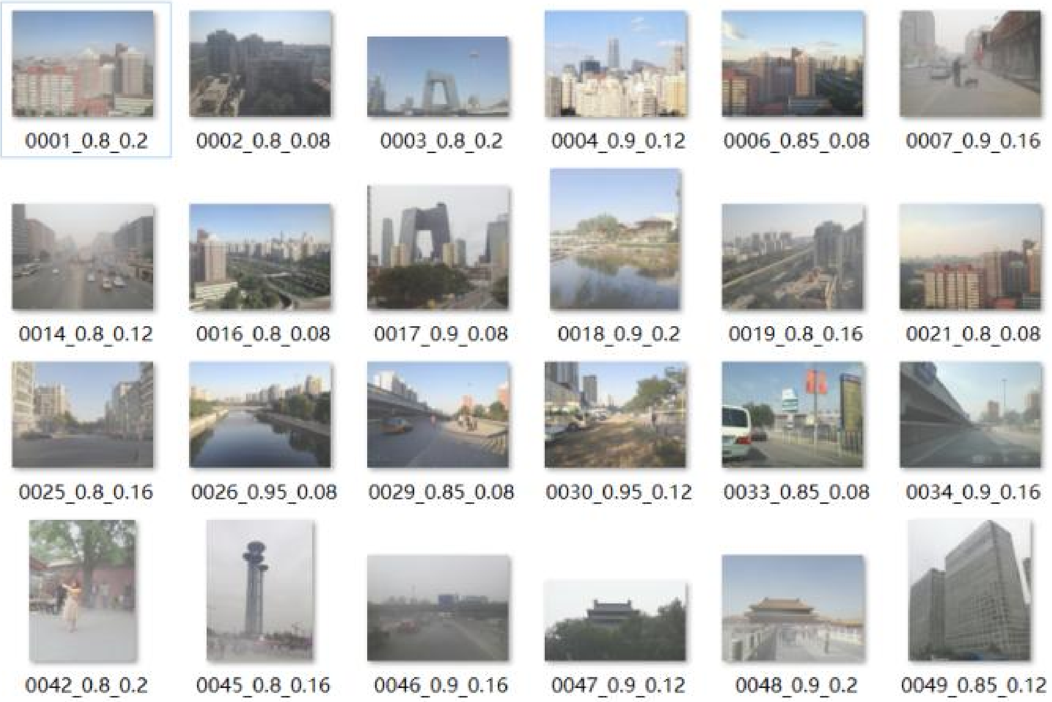
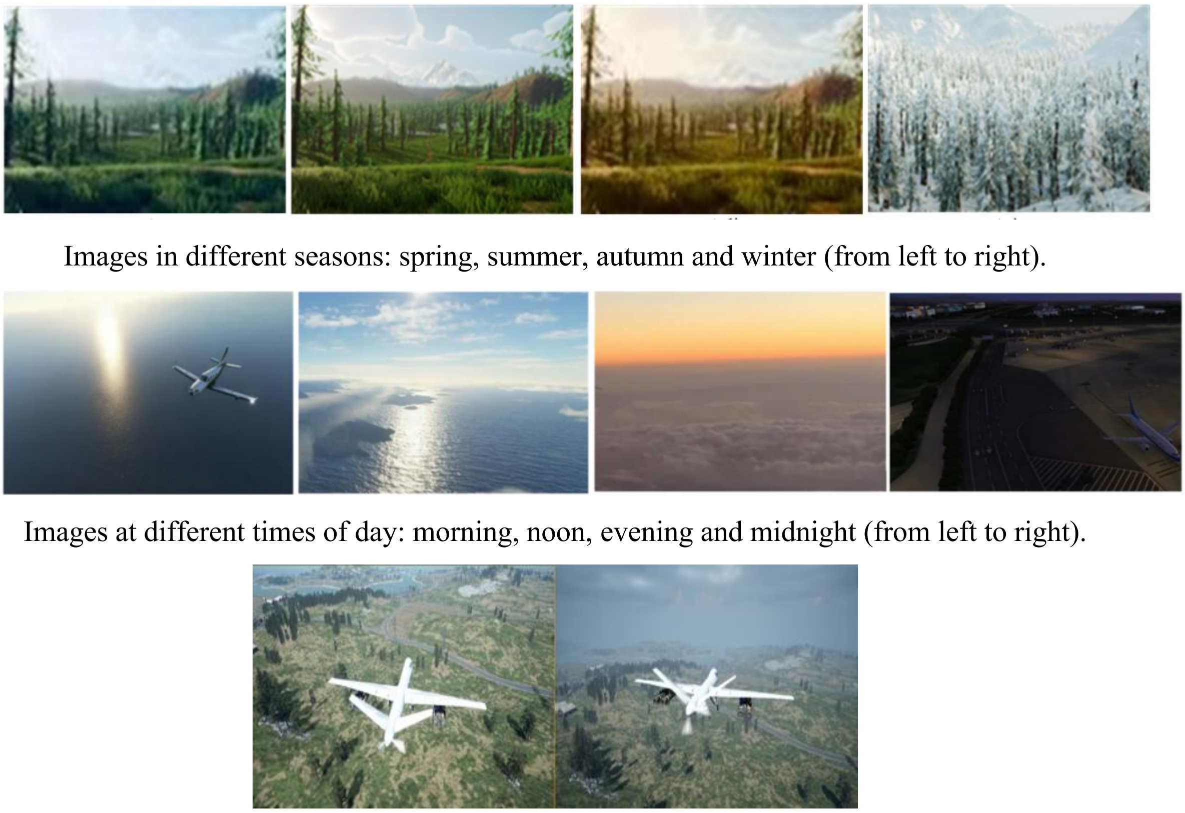


Figure 3.2. Samples images from the SOTS-outdoor dataset for dehazing assessment.

For image dehazing assessment, the SOTS-outdoor [74] public dataset is used. The SOTS dataset is a synthetic dataset consisting of 1,000 test images, divided into indoor and outdoor categories, each containing 500 images, see some sample images in Fig. 3.2. The outdoor images are used to test the dehazing effects of the algorithms.

In addition, a self-collected dataset is also used, which contains a total of 2,035 pictures of virtually simulated scenes, as shown in Fig. 3.3. The simulated virtual scenes include natural environments in different seasons, extreme weather (e.g. rainy, snowing, foggy) and different times of day (e.g. morning, noon, evening and midnight). The large variations have enriched the contents within the dataset for testing the robustness of the related algorithms in terms of their performance in denoising and defogging etc.



Images in different seasons: spring, summer, autumn and winter (from left to right).

Images at different times of day: morning, noon, evening and midnight (from left to right).

Images in different weather conditions, e.g. raining (left) and foggy (right).

Figure 3.3. Sample images from the self-collected dataset of virtually simulated scenes in different seasons (top), at different times of day (middle) and different weather conditions (bottom).

3.2.2. Evaluation Metrics

(1) **Peak Signal-to-Noise Ratio** (PSNR)

Given a hazy image I and a haze-free image K both of size M*N, the Mean Squared Error (MSE) is defined as:

$$MSE = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}[I(i,j) - K(i,j)]^2 \tag{3.1}$$

where $I(i,j)$ and $K(i,j)$ represent the grayscale values of pixels at location (i,j) in the hazy and haze-free images, respectively.

The Peak Signal-to-Noise Ratio (PSNR) is then defined as:

$$PSNR = 10log_{10}(\frac{I\_max^2}{MSE}) = 20log_{10}(\frac{I\_max}{\sqrt{MSE}}) \tag{3.2}$$

where I_max denotes the maximum possible pixel value in the image, or 255 for an eight-bit image.

Eq. (3.2) is commonly used for grayscale images. For colour images with three channels (RGB), the MSE is calculated separately for each channel, and the resulting MSE values are used to compute the PSNR for each channel. The final PSNR for the colour image is obtained by taking the average of the PSNR values across all channels.

PSNR is one of the most widely used objective metrics for assessing image quality. It measures the difference between two images, such as a compressed image and its original, to evaluate the quality of the compressed image. However, many experimental results have shown that the PSNR score may not always correlate perfectly with the visual quality perceived by the human eye. In some cases, images with higher PSNR values may appear worse to human observers compared to images with lower PSNR values. Typically, a PSNR value between 30 and 50 indicates a higher degree of similarity between two input images, a higher value suggesting a closer match.

(2) **Structural Similarity** (SSIM)

This metric is used to measure the structural similarity between two images. Structural Similarity (SSIM) [75] compares the structure, luminance, and contrast of two images, denoted as X and Y. Given two images, and the structural similarity between the two images can be computed using the following formula:

$$SSIM(X,Y) = \frac{(2\mu_X\mu_Y + c_1)(2\sigma_{XY} + c_2)}{(\mu_X^2 + \mu_Y^2 + c_1)(\sigma_X^2 + \sigma_Y^2 + c_2)} \tag{3.3}$$

In Eq. (3.3), μ_X represents the average value (mean) of X, μ_Y is the average value (mean) of *Y*, σ_X^2 is the variance of *X*, σ_Y^2 is the variance of *Y*, and σ_{XY} is the co-variance over a window between the two images. The constants c_1 and c_2 are used to maintain stability and are typically set to small values like 0.01 and 0.03, respectively. The range of the Structural Similarity (SSIM) metric is from -1 to 1, with larger values indicating less distortion. When two images are identical, the SSIM value is 1.

3.3 Image Denoising Models and Evaluation

For denoising of aerial images, researchers have proposed a plethora of image denoising methods, primarily categorized into traditional image denoising and deep learning-based image denoising methods, as detailed below.

3.3.1 Conventional Denoising Methods

Conventional denoising methods can be further divided into spatial domain denoising methods and transform domain methods. Spatial domain methods primarily utilize filters for denoising. They process the neighbourhood of each pixel in the image using a filter, iterating through the entire image. Spatial domain denoising methods can be classified based on the linearity of the filters into linear filtering methods and non-linear filtering methods [79].

In linear filtering methods, the most common one is the mean filter. For a pixel contaminated by noise, the mean filter, and its extension such as Gaussian filtering, calculates the average value or weighted average of all the pixels in its neighbourhood and assigns this value to the contaminated pixel. Non-linear filtering methods typically include the median filter and bilateral filter. The median filter, or even with weighted version, initially sorts the pixels around a particular pixel, resulting in an ordered data sequence. Then, it assigns the median value from this sequence to the pixel, effectively removing low and high-frequency components in noisy images. Thus, it is commonly used for eliminating salt-and-pepper noise. However, it has the drawback of potentially causing image discontinuities.

The bilateral filter [80] considers both the grayscale similarity and spatial position relationships between pixels. It assigns higher weight values to pixels that are both close to the center pixel and have similar grayscale values, while giving lower weight values to pixels that are farther away or have dissimilar grayscale values. The advantage of the bilateral filter lies in its ability to preserve more edge information, but it requires further improvement in protecting image texture and detail information.

Local filters can effectively remove noise when the noise level is relatively low but are less effective at higher noise levels. To address this issue, the Non-Local Means (NLM) [81] denoising algorithm leverages the self-similarity and redundancy in the image's structure for denoising. Danbov et al. introduced the Block Matching 3D (BM3D) denoising algorithm [82], which involves finding a series of similar image blocks and grouping them to obtain multiple three-dimensional blocks. Filtering is then performed in three-dimensional space, followed by using a three-dimensional inverse transform to produce the denoised result. Compared to NLM, this algorithm achieves a higher PSNR but comes with a higher complexity.

For transform domain filtering, these rely on the transformation of the image to a different domain before applying the filtering. Typical algorithms include the Fourier transform and wavelet transform, where it is assumed that the noise will be more easily characterised and distinguished in the transformed domains. For example, Fourier transformation converts data from the time domain to the frequency domain, where noise in the frequency domain often appears in high-frequency regions. Noise removal can be achieved through low pass filtering in the frequency domain. However, this process also eliminates the texture and detail information in the image.

In addition to Fourier transformation, wavelet transformation has also been employed for image denoising. Denoising methods utilizing wavelet transformation process noise removal based on the differences between image features and noise after undergoing wavelet transformation. The advantage of wavelet-based denoising is that it can simultaneously preserve both frequency and spatial information in the image. However, its drawback lies in its weaker directionality, as it can only extract limited directional information.

3.3.2 Deep Learning-Based Image Denoising Methods

In recent years, deep learning has gained the favor of many researchers due to its powerful feature capturing capabilities and flexible network architectures. Burger et al. employed a Multi-Layer Perceptron (MLP) [83] to learn the mapping from noisy images to clean images, achieving performance comparable to BM3D. Chen et al. [86]

designed a trainable Nonlinear Reaction Diffusion (TNRD) denoising model. However, MLP and TNRD can only handle images with fixed noise levels and may not yield ideal results when applied to datasets with varying noise levels.

To enhance the model's ability to handle varying levels of noise, Zhang et al. [87] introduced the Denoising Convolutional Neural Network (DnCNN) model. This model not only addresses images with different noise levels but also utilizes residual learning and batch normalization techniques to expedite model training. Subsequently, Zhang et al. proposed the Fast and Flexible Denoising Network (FFDNet) model [88], which builds upon DnCNN by including noise levels as an additional input to the model. FFDNet is capable of handling spatially correlated noise and plays a crucial role in balancing noise reduction and image detail preservation.

However, these aforementioned models do not yield satisfactory results for real image denoising. To tackle this issue, Guo et al. [89] introduced the Convolutional Blind Denoising Network (CBDNet) and constructed a new noise model to simulate real noise. CBDNet consists of a denoising sub-network and a noise level estimation sub-network, enhancing the network's performance and generalization capacity by introducing an asymmetric loss function. Nevertheless, CBDNet's network structure is complex and comes with a significant computational cost, making it less suitable for practical applications. Therefore, Anwar and Barnes [90] proposed the Real Image Denoising Network (RIDNet) to address real-world denoising scenarios. RIDNet adopts a modular structure for the denoising network and introduces a channel attention mechanism for adaptive channel weight adjustment. The introduction of CBDNet and RIDNet has driven the development of image denoising research in real-world settings.

3.3.3 Assessment and Evaluation

The approaches we selected for comparison in this group of experiments are briefed as follows. Within these approaches, BM3D and WNNM are conventional filtering-based methods, and the others are deep-learning models.

- BM3D [82]: As a novel image denoising method, BM3D is based on an enhanced sparse representation in the transform-domain, where the enhanced sparsity is achieved by grouping similar 2-D fragments of the data within the transform domain into 3-D data arrays namely “groups”, followed by collaborative filtering being applied to these 3-D groups. It has three key steps, i.e. 3-D transformation of a group, shrinkage of transform spectrum, and inverse 3-D transformation, where the collaborative filtering can help to reveal the finest details shared by grouped fragments whilst preserving the essential unique features of each individual fragment;

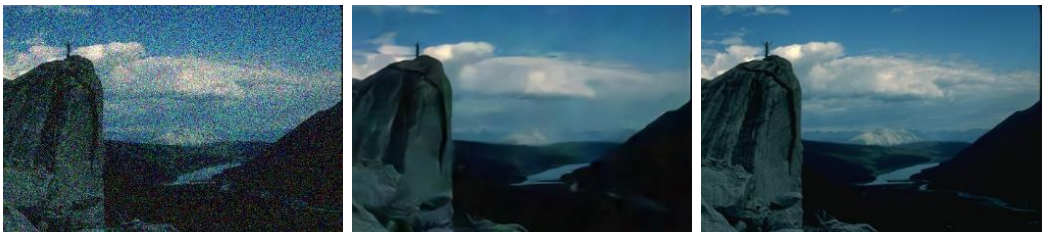
- WNNM (Weighted Nuclear Norm Minimization) [91]: The image is modelled as Y = X + N, where Y is also composed of samples with noise, forming a sample matrix. X and N are the corresponding noise-free sample matrices and the noise, respectively. The given constraint is that X is a low-rank matrix. Since the matrix composed of similar samples exhibiting low-rank characteristics, when the noise does not have low-rank characteristics, image denoising can be achieved through low-rank clustering;
- VDN (Variational Denoising Network) [94]: This model is capable of simultaneous image denoising and noise estimation. In typical work, Gaussian white noise is assumed to be present in the image, but this model is not limited to Gaussian assumption. The proposed generative model exhibits strong generalization capabilities and performs well even for noise not encountered in the test set. The model provides an explanation for the overfitting phenomenon often observed in deep learning methods trained using MSE loss. This issue is attributed to overfitting the prior of the underlying clean image while neglecting variations in noise. This VDN model explicitly models the generation of noise, thereby avoiding this drawback of deep learning methods;
- FFDNet [88]: The adjustable noise level mapping, denoted as M, is used as input to provide flexibility to the denoising model regarding noise levels. An invertible downsampling operator is introduced to reshape the input image of size W × H × C into four subsampled sub-images of size 4W/2 × H/2 × 4C, where W, H and C represent the width, height and the number of channels of the image, respectively. To ensure that noise level mapping robustly controls the trade-off between denoising and detail preservation without introducing significant visual artifacts, an orthogonal initialisation method is also applied to the convolution filters;

- NAFNet [96]: Taking inspiration from the Transformer architecture, the use of Layer Normalization (LN) is incorporated to facilitate smoother training. NAFNet also introduces LN operations, leading to significant performance gains on image denoising and deblurring datasets. In the Baseline approach, ReLU is jointly replaced with GELU and CA. GELU helps maintain denoising performance while significantly enhancing deblurring performance. Two new attention module compositions are proposed, namely CA (Channel Attention) and SCA (Spatial-Channel Attention).

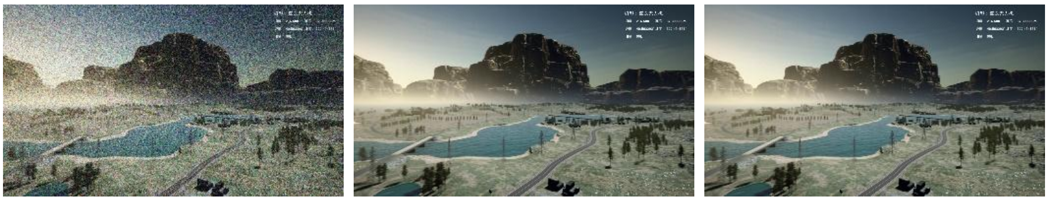
In Table 3.1, the quantitative results from one public dataset and the own dataset is compared, using the PSNR and SSIM as indicated before. As seen, BM3D and NAFNet have produced significant better results than other compared approaches, though BM3D performs better in the public dataset whilst NAFNet has higher results in the own dataset. This is mainly because our dataset is more challenging as it is 3D simulation of the real flight environment. As a result, the scene is more complex and larger (1,920×1,080), containing foggy and rainy scenes as well as images from different seasons and time of a day. That is why BM3D tends to work less effectively as it does in the CBSD68 dataset. Also note that BM3D is a filtering based conventional model and NAF3D is a DL based approach using the transformer model.

Table 3.1: Quantitative evaluation of the image denoising algorithms

Models	CBSD68 dataset		Own dataset	
	↑PSNR (dB)	↑SSIM	↑PSNR (dB)	↑SSIM
BM3D	**41.63**	**0.9936**	36.50	0.9301
WNNM	39.38	0.9750	35.02	0.9320
VDN	30.83	0.8533	25.83	0.8533
FFDNet	28.98	0.7969	26.16	0.7897
NAFNet	40.30	0.9621	**38.02**	**0.9502**



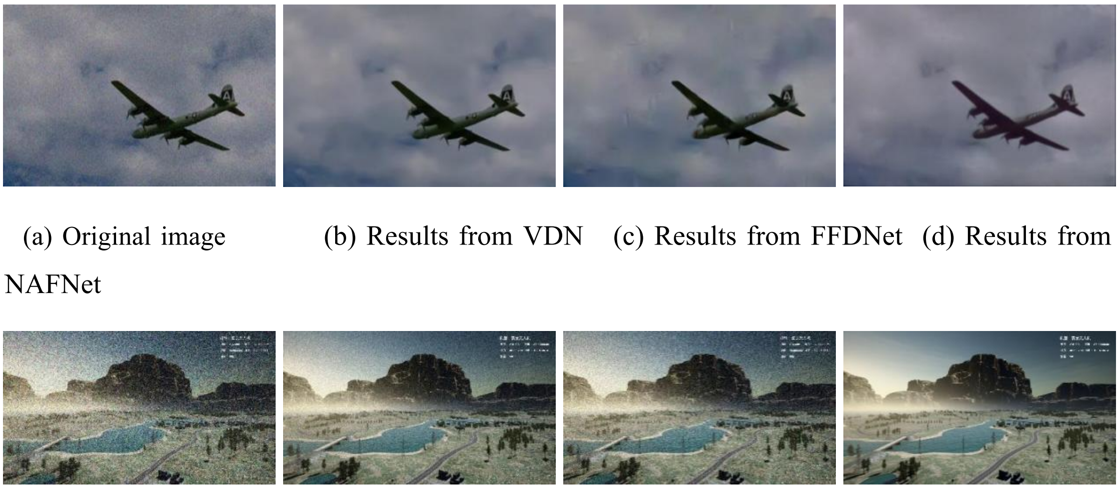
(a) Original image from CBSD68 (b) Result from BM3D (c) Result from WNNM



(d) Original image from own dataset (e) Result from BM3D (f) Result from WNNM

Figure 3.4. Denoising results from BM3D and WNNM.

In addition, it is worth noting that the evaluation results from PSNR and SSIM are not always consistent, though they seem to have an overall positive correlation. For example, NAFNet has a higher PSNR of 40.30 than WNNM at 39.38, yet its SSIM is 0.9621, which is lower than 0.9750 from WNNM in the public dataset. In the own dataset, BM3D has a higher PSNR than WNNM yet its SSIM is lower than that of WNNM. This inconsistency in different metrics on one hand, indicates the difficulty in evaluation of the denoised images. On the other hand, it shows great potential for future investigation in this context.



(a) Original image (b) Results from VDN (c) Results from FFDNet (d) Results from NAFNet

(e) Original image (f) Results from VDN (g) Results from FFDNet (h) Results from NAFNet

Figure 3.5. Comparison of denoising results from deep learning models of VDN, FFDNet and NAFNet. The original images used are from CSDB68 (top) and own dataset (bottom), respectively.

Furthermore, the denoised results of some test images are given for comparison, including those from conventional approaches of BM3D and WNNM, see in Fig. 3.4, and those from deep learning models, see in Fig. 3.5, respectively. In both Fig. 3.4

and Fig. 3.5, samples images from the publicly available dataset CBSD68 and own dataset are illustrated. As can be seen, the denoising images are significantly improved in terms of the visual quality.

As seen from the results above, BM3D apparently outperforms WNNM, thanks for considering the grouped fragments within a 3-D block along with the sparse representation based collaborative filtering, which has greatly improved the performance of denoising, including those using deep learning models. The relatively poor performance from deep learning can be due to two reasons, i.e. insufficient training samples and the inconsistency between the training set and the testing set, especially when the overall dataset is small. The latter may be caused by the random characteristics of the noise within the image, which has potentially affected the learning-based approaches. Nevertheless, in the best deep learning model, NAFNet, the transformer architecture has somehow mitigated such limitations, which can be further explored.

3.4 Image Dehazing Models

Fog can reduce the brightness and contrast of images, causing objects to appear blurry and degrading image quality. Image dehazing is a classic problem in the field of computer vision, with a history dating back to the early 1900s. The development of image dehazing can be broadly categorized into the following types: photometric-based methods, polarization-based [19], prior-based and learning-based methods [20].

3.4.1 Conventional approaches

The conventional approaches rely mainly on assumptions to model the degradation due to foggy or hazing effects, fall within prior based approaches as detailed below. In the mid-development phase of image dehazing, further success was achieved by utilizing improved assumptions and priors through physics-based dehazing methods. For instance, Tan et al. [20] used a Markov Random Field to maximise the local contrast of input hazy images to achieve effective dehazing results. However, this approach still left some minor halos and oversaturated areas in the generated images. Fattal et al. [21] introduced a novel method for estimating light transmission, which also addressed shadow issues and improved image clarity and contrast. Nevertheless, this method struggled to handle severely blurred images. Kratz and Nishino [22] employed a factorial Markov Random Field to estimate reflectance and depth with natural statistical priors. This approach yielded higher detail features and saturation but occasionally resulted in minor halos in scenes like lakes and streets.

To better eliminate the haze effects in real outdoor scenes, inspired by the dark object subtraction technique, the Dark Channel Prior (DCP) was proposed by He et al. [97] for estimating image transmission in outdoor scenarios. This method is suitable for most outdoor hazy images, but it struggles with handling the sky portion of the images and comes with high computational costs. To enhance the computational efficiency of DCP-based methods, standard median filtering, average median filtering, and guided image filtering have been used as alternatives to the time-consuming process of soft matting. Tang et al. [99] combined various haze-related features with random forests to estimate transmission maps. Berman et al. [100] proposed a pixel dehazing method based on non-local priors. While Tang's method [99] yielded significant dehazing effects compared to previous approaches, the computational demands make it difficult to achieve real-time results.

3.4.2 Learning-Based Methods

Research methods in learning-based image dehazing can be roughly categorized into two types: two-stage methods based on transmission estimation and end-to-end pipeline methods. The former involves first learning a non-linear mapping between input hazy images and their corresponding transmission maps and then using this mapping to estimate transmission maps, followed by solving the haze-free image using an atmospheric scattering model.

For example, Cai et al. [26] proposed DehazeNet for learning and estimating transmission maps, where the network takes hazy images as inputs and outputs estimated transmission maps, which are then used in an atmospheric scattering model to obtain haze-free images. Similarly, Ren et al. [102] introduced a multi-scale CNN-based image dehazing method, consisting of coarse-scale and fine-scale networks, to learn non-linear mappings from hazy images for estimating scene transmission maps. Li et al. [105] introduced AOD-Net, which optimized the end-to-end network from hazy images to clear images, where the AOD-Net model was also combined with the Faster R-CNN for object detection in the images before and after dehazing to evaluate AOD-Net's dehazing performance. Zhang et al. [27] proposed a densely connected pyramid dehazing network (DCPDN) that can jointly learn the transmission maps, atmospheric light, and dehazing. In Chen et al. [107], a new gated context aggregation network (GCANet) was proposed along with the smooth dilated convolutions for image dehazing. In Deng et al. [108], a haze-aware representation distillation GAN (HardGAN) was presented to fuse the global atmospheric luminance and local spatial structure for effective image dehazing. Liu et al. [109] introduced a multi-scale estimation image dehazing method based on attention mechanisms (GridDehazeNet). Qin et al. [110] proposed a feature fusion attention network (FFA-Net), in which the channel attention is combined with the pixel attention to directly recover hazy images into clear images.

3.4.3 Evaluation and Results

Based on the discussions above, the dehazing approaches being compared are briefed as follows.

- DehazeFormer [111]: DehazeFormer is a network based on a five-layer U-Net architecture, where convolution blocks are replaced with DehazeFormer blocks. Some optional components outlined within the dashed lines can also be considered. SK fusion and soft reconstruction are used to replace the original settings of the concatenation fusion and global residual methods.
- FFANet [110]: A feature fusion attention network (FFA-Net) is proposed for end-to-end direct restoration of haze-free images. It leverages attention-based FFA structures at different hierarchical levels, allowing for the adaptive learning of feature weights from the Feature Attention (FA) module, with higher weights assigned to more important features. FFANet can also help to retain shallow-level information and pass it to deeper layers;
- GridDehaze Net [109]: An attention-based multi-scale dehazing network is introduced, comprising of three modules: pre-processing, backbone, and post-processing. Within the backbone module, a novel attention-based multi-scale network is implemented to address the bottleneck issues commonly encountered in traditional multi-scale methods effectively;
- GCANet [107]: The model directly learns the residual between the original and hazy images, which has three convolutional blocks as the encoder, one deconvolutional block, and two convolutional blocks as the decoder. Smooth

dilated residual blocks are inserted between them to aggregate contextual information without causing a grid-like artifact. To fuse different-level features, an additional gate fusion sub-network is utilized. GCANet predicts the end-to-end target clean image's residual between the blurred input images.

- PMNet [112]: An attention-based Single Haze-aware Attenuation (SHA) method is proposed, which uses fewer parameters and computational resources to estimate the haze density map. The density map, along with the haze map, is further utilized to refine texture features for improved dehazing results.

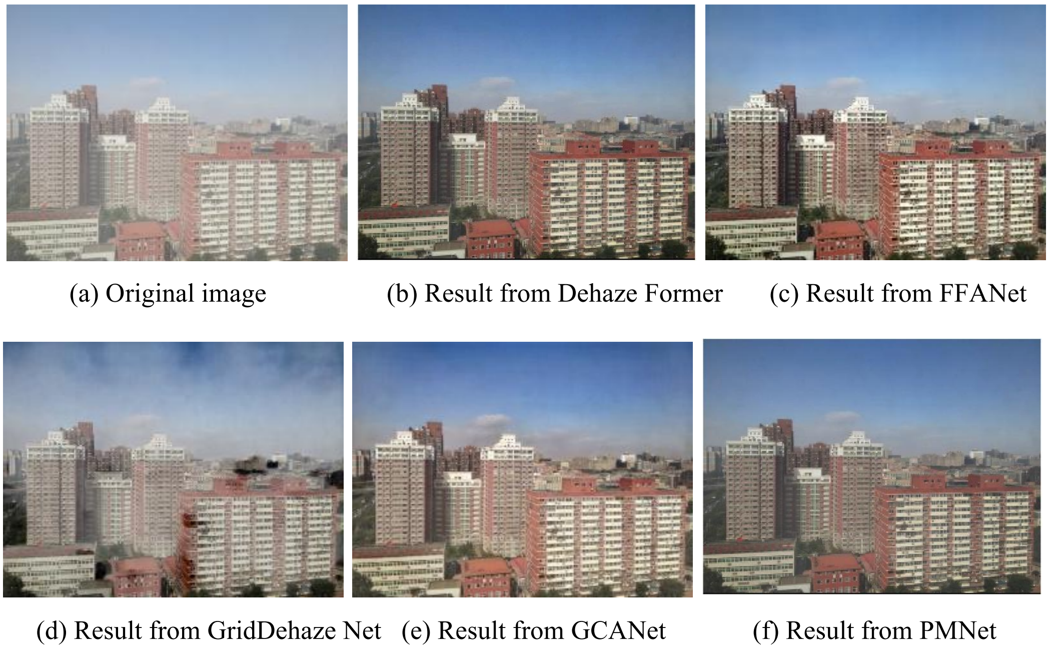
In Table 3.2, quantitative results in terms of PSNR and SSIM are given to compare the performance of various dehazing approaches as discussed above, using the images from one public dataset, SOTS-outdoor [74] and one own dataset. For the SOTS-outdoor dataset, Daheze Former seems to produce consistently better results than other compared approaches, thanks to the transformer-based model in effectively tackling with the variations within the dataset. However, in the more challenging self-collected dataset, Dehaze Former fails to produce the best results.

Table 3.2 Comparison of different dehazing approaches

Models	Public (SOTS-outdoor)		Own dataset	
	↑PSNR (dB)	↑SSIM	↑PSNR (dB)	↑SSIM
Dehaze Former	**36.80**	**0.9949**	32.04	0.9210
FFANet	36.39	0.9891	34.02	**0.9587**
GridDehaze Net	30.86	0.9818	**34.89**	0.9287
GCANet	30.23	0.9801	28.50	0.8905
PMNet	34.41	0.9905	29.29	0.9025

On the contrary, the highest PSNR and SSIM are yielded by GridDehaze Net and FFANet, respectively, though FFANet seems to produce more consistently better results than the GridDehaze Net with the much higher SSIM and slightly lower PSNR. This again shows the inconsistency of these two metrics in the quantitative assessment of the dehazing results, which has inevitably raised the open question on how to solve such inconsistency in the future.

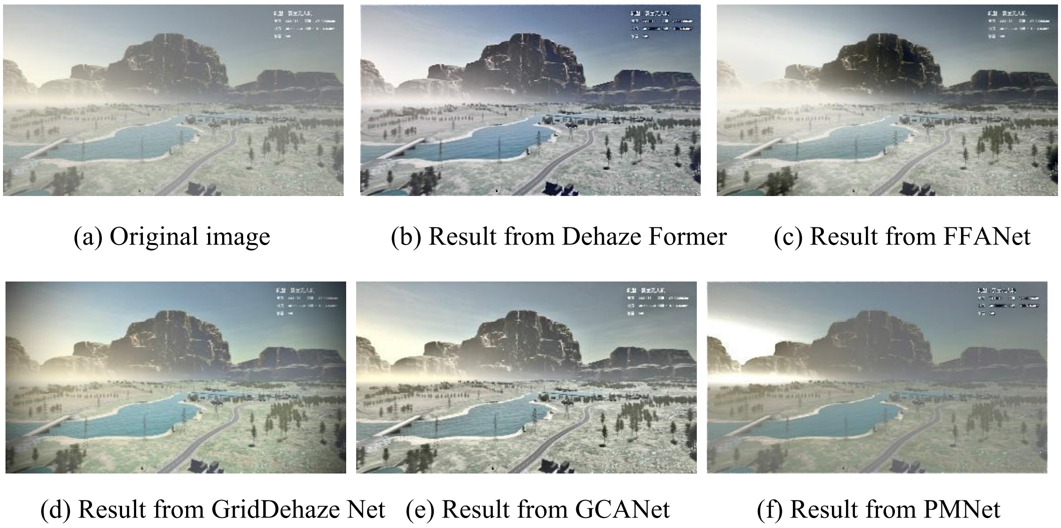
In addition, visual comparison of the dehazing results is also given in Fig. 3.6 and Fig. 3.7, which are for the results from the publicly available dataset and self-collected dataset, respectively. As can be seen, both Dehaze Former and FFANet have produced much better results than other compared approaches, where the dehazed images show improved contrast than the original ones. In other words, the Dehaze Former and FFANet algorithms exhibit the best dehazing performance according to the experiments.



(a) Original image (b) Result from Dehaze Former (c) Result from FFANet

(d) Result from GridDehaze Net (e) Result from GCANet (f) Result from PMNet

Figure 3.6. Comparison of dehazing results from publicly available dataset.



(a) Original image (b) Result from Dehaze Former (c) Result from FFANet

(d) Result from GridDehaze Net (e) Result from GCANet (f) Result from PMNet

Figure 3.7 Comparison of dehazing results from self-collected dataset.

3.5 Summary

In this chapter, a survey of the denoising models for UAV images in SLAM implementation is conducted, followed by a comprehensive evaluation. Several models are selected for both qualitative and quantitative assessment, including conventional approaches in the spatial domain and transform domain as well as deep learning models. By benchmarking on the publicly available datasets, it is found that the BM3D model, an extension to the NLM approach, outperforms all others, even the deep learning approaches, owing mainly to the local-similarity with a 3-D block and sparse representation enabled collaborative filtering in modelling. This shows the great potential of conventional vision - based perception models in image denoising. It also indicates the limitations of the DL models in this context, due mainly to the ill-posed problem in training the models. Furthermore, the great performance of the NAFNet has suggested that the transformer architecture can help to mitigate the limitations here and improve the modelling, thus is worth for further investigation.

As the degradation process of UAV images can be much more complicated, this chapter only covers a few tasks, where many other useful topics have not been addressed, such as deburring and normalisation of the lighting effects et al. These will be the future work, along with the deployment of other models.

Chapter 4. Effective Object Detection and Recognition with an Improved YOLOv7

4.1 Introduction

Object detection and recognition are crucial tasks in the field of computer vision. These tasks involve identifying and locating one or more target objects within a given image or video. Object detection and recognition techniques initially require the classification of objects, followed by the identification of their positions within the image or video, often marked with bounding boxes or contours [32][113][114].

In civilian applications, object detection technology can recognize and track individuals, vehicles, and other objects appearing in surveillance footage, providing effective means for security and prevention. In the realm of autonomous driving, object detection technology, using onboard cameras, can real-time detect obstacles on the road, offering reliable support for the autonomous navigation of self-driving vehicles. In the field of medical image analysis, object detection technology can facilitate the automatic diagnosis of diseases like tumors, offering crucial references and support for medical diagnosis and treatment.

In the military domain, object detection finds widespread use, primarily driven by special military needs and missions that involve aspects such as combat command, weapon systems, and computer vision-based object detection and military video surveillance. As for drones, object detection can be used to identify target objects, enhancing its detection, classification and tracking. In addition, the integration of such systems can be also applied to process real-time satellite monitoring data for civil, industrial, environmental and military applications towards smart decision making.

Among a number of existing approaches, the YOLO series of models [30] are selected for object detection in the thesis, due to the following reasons. The first is the high computational efficiency that can be applied for real-time processing. The second is the relatively high detection accuracy in comparison to many other methods. The third is the unique property of the YOLO series feature, as it adopts a one-stage processing to combine the detection and classification together. As a result, it also features good robustness and generalisation in various scenarios as well as an end to end model for extra efficiency.

4.2 Related Work

Before 2012, object detection and recognition primarily relied on traditional detection methods. With the development of GPUs and NPUs, object detection and recognition have shifted from traditional approaches to those based on DL methods, as depicted in Fig. 4.1. Detailed discussions of related models and algorithms, especially the DL based, are given as follows.

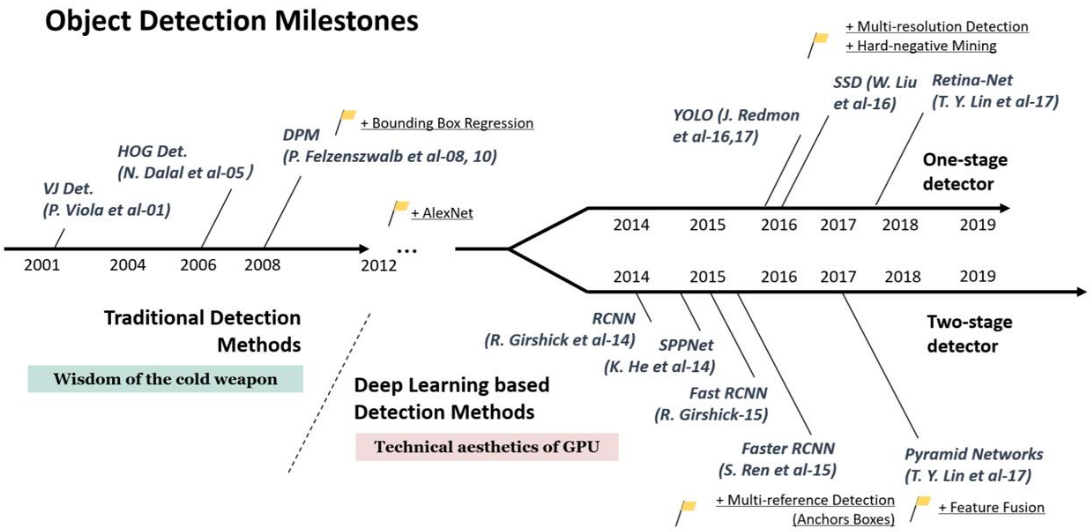

Object Detection Milestones
+ Multi-resolution Detection
+ Hard-negative Mining
SSD (W. Liu et al-16)
Retina-Net (T. Y. Lin et al-17)
YOLO (J. Redmon et al-16,17)
DPM (P. Felzenszwalb et al-08, 10)
+ Bounding Box Regression
HOG Det. (N. Dalal et al-05)
VJ Det. (P. Viola et al-01)
One-stage detector
+ AlexNet
2001
2004
2006
2008
2012
...
2014
2015
2016
2017
2018
2019
2014
2015
2016
2017
2018
2019
Traditional Detection Methods
Wisdom of the cold weapon
RCNN (R. Girshick et al-14)
SPPNet (K. He et al-14)
Two-stage detector
Deep Learning based Detection Methods
Technical aesthetics of GPU
Fast RCNN (R. Girshick-15)
Faster RCNN (S. Ren et al-15)
Pyramid Networks (T. Y. Lin et al-17)
+ Multi-reference Detection (Anchors Boxes)
+ Feature Fusion

Figure 4.1: The development history of object detection and recognition (source: https://wikidocs.net/167508).

4.2.1 Conventional Approaches

In 2001, P. Viola and M. Jones, among others, introduced the Viola-Jones (VJ) detector for face recognition [32]. The VJ detector employed a sliding window approach and demonstrated robustness across various skin tones. It began by calculating all different-sized windows within an image and then sequentially traversed the image using windows of various sizes to identify facial features.

In 2005, Dalal et al. [113] introduced the HOG (Histogram of Oriented Gradients) Detector method. This method primarily employed directional gradient histograms for detection and used Support Vector Machines (SVM) with manually crafted features to detect objects. The conventional HoG approach relies on a spatial window to extra the features of the object, where the optimal window size should match the dimension of the object. However, this detection method was not suitable for scenarios where the scale of the target object varies significantly, due to the difficulty to identify the optimal window size and resulting in poor detection performance. To address the issue of scale variation, the Deformable Part-based Model (DPM) algorithm emerged. It was introduced by P. Felzenszwalb et al. in 2008 [114]. DPM used a sliding window approach to extract features at different scales and employed SVM for classification. This method exhibited excellent detection performance, surpassing traditional algorithms that relied on manual feature extraction. In summary, traditional object detection algorithms typically involve three steps: sliding window traversal of the image, feature extraction (such as Harr or HOG), and classifier-based detection with bounding box regression. The workflow for object detection in conventional approaches such as [32][113][114] is illustrated in Fig. 4.2.

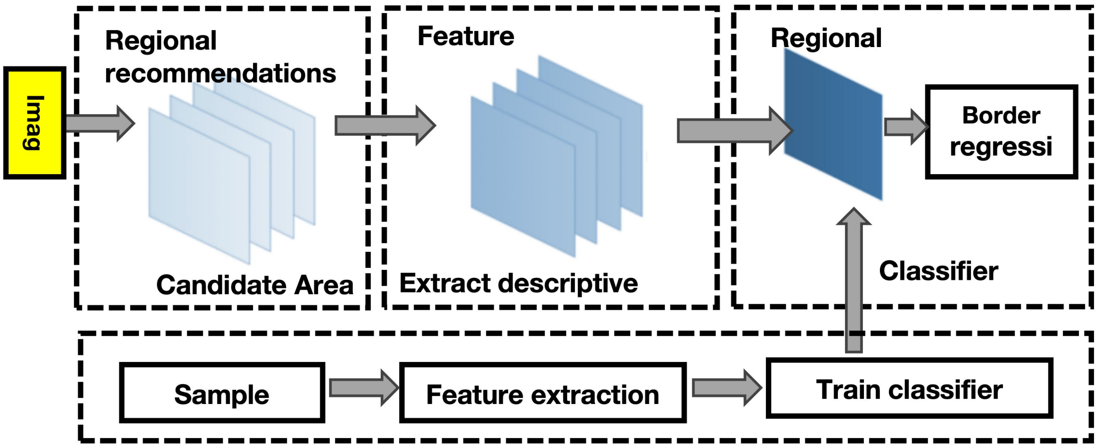


Figure 4.2. Workflow of conventional object detection approaches

Conventional object detection algorithms suffer from inherent limitations.

1) The selection of feature regions depends on manual extraction. When feature extraction is incomplete or erroneous, detection accuracy is low, and the time complexity is high.
2) Due to the complexity of natural scenes, traditional detection algorithms yield poor feature extraction results when objects are affected by factors such as lighting and occlusion, leading to reduced detection performance.

As a result, conventional methods struggle to fully meet the practical requirements of object detection and recognition applications.

4.2.2 Deep Learning - Based Approaches

Deep learning (DL) -based detection algorithms employ CNNs to perform multi-level convolution operations on images for more accurate feature extraction. They can address issues such as false positives and false negatives caused by object deformations and changes in the surrounding environment. These algorithms exhibit higher robustness compared to traditional methods that rely on manual feature extraction.

Currently, DL based object detection algorithms can be categorized into one-stage object detectors and two-stage object detectors. The primary difference between them lies in the fact that two-stage object detectors introduce a region proposal phase to generate candidate boxes, which are then used to predict object boxes. Notable one-stage detectors include YOLO [30], SSD [115], and RetinaNet [116], while representative two-stage detectors include R-CNN [29], Fast R-CNN, and Faster -RCNN [118].

(1) R-CNN Series Algorithms

RCNN algorithms utilize selective search for region proposal and then use convolutional networks to extract features from these regions. They combine SVM for candidate box classification and employ non-maximum suppression (NMS) for regression to identify objects. An example of the RCNN algorithm is illustrated in Fig. 4.3. RCNN achieved an average precision of 58.5% on the VOC07 test set, representing a significant improvement over traditional algorithm like DPM-v5.

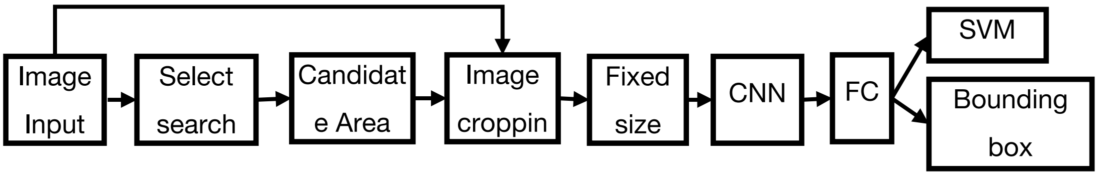


Figure 4.3. Diagram of the RCNN algorithm

However, RCNN has notable drawbacks. Since RCNN is a distributed two-stage object detection algorithm, it needs to crop and resize candidate region images during feature extraction to adapt to the input of CNNs. This process results in the loss of image feature information, leading to lower accuracy. Furthermore, RCNN requires a convolutional calculation for each candidate region, which significantly increases computational complexity, causing slower training and prediction speeds.

Fast RCNN is an improved algorithm based on the RCNN approach. It begins by transforming input images into feature maps using convolutional networks. Subsequently, it extracts candidate regions on the feature maps using the selective search algorithm. This approach addresses the issue of repetitive convolution calculations seen in RCNN, effectively improving computational speed. Fast RCNN introduces Region of Interest (ROI) pooling, which is inspired by SPP Net. After obtaining the feature maps, ROI pooling extracts features using blocks of different sizes, divided into 'n' blocks, resulting in an 'n'-dimensional feature vector. Fig. 4.4 provides an example illustration of the Fast RCNN algorithm. In Fig. 4.4, FCN represents feature maps generated using the output of the first five convolutional layers of the AlexNet. The bounding box regression involves calculating the difference between anchor boxes and predicted values.

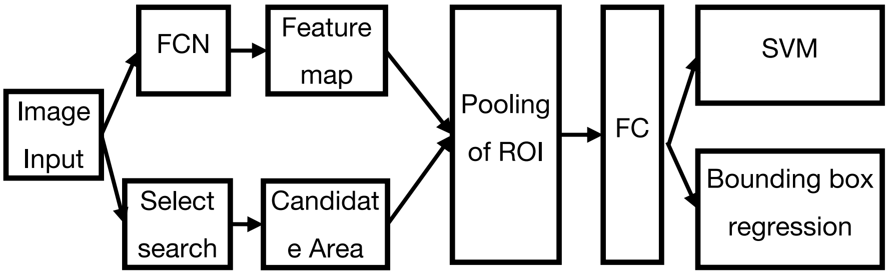


Figure 4.4. Diagram of the Fast RCNN algorithm

Fast RCNN performs feature extraction only once, utilizing a single network to simultaneously handle both classification and regression tasks. It combines the objective functions for target classification and bounding box prediction into a single multi-task objective function, improving both speed and accuracy. The network employs cross-entropy and smooth L1 loss for classification and bounding box regression, respectively. Due to its multi-scale training capability, Fast RCNN effectively performs data augmentation when given images of different sizes. Fast RCNN achieves a mean average precision (mAP) of 70.0% on the VOC07 dataset and is 200 times faster than RCNN. However, its detection speed is still constrained by the candidate region operation.

In order to further enhance detection speed, Faster RCNN replaces the candidate region operation in Fast RCNN with the Region Proposal Network (RPN) structure. The RPN network consists of two parts: one part obtains the positive and negative classifications of anchors using softmax, and the other part calculates the regression offsets for anchor boxes. The final proposals are generated by combining the positive anchors and the regression of the target boxes. These proposals then serve as input to Fast RCNN. An illustration of the Faster RCNN algorithm is provided in Fig. 4.5.

The common feature extraction network used in Faster RCNN is based on the structure of VGG16 with the last max-pooling layer removed. The introduction of the RPN network is to provide Fast RCNN with regions where objects may be located directly, eliminating the need for the selective search mechanism to extract candidate regions, thereby improving detection speed. Faster RCNN achieved a mAP of 70.4% on the VOC12 dataset, a significant improvement over Fast RCNN. Additionally, Faster RCNN is the first end-to-end detector among two-stage object detection algorithms.

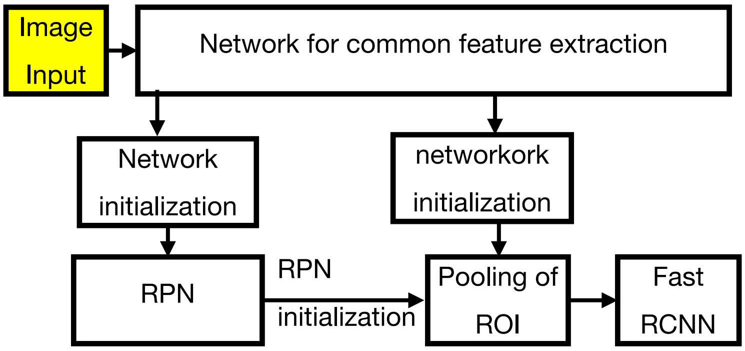


Figure 4.5. Example illustration of the Faster RCNN Algorithm.

Throughout the development of two-stage object detection, the detection system has evolved from independently operating components, such as region proposal, feature extraction networks, and bounding box regression, into an integrated network that shares computation at the feature layer level. This integration has greatly reduced computation time.

(2) SSD Series Algorithms

The SSD object detection algorithm [115] pioneered the use of multi-scale feature maps for detection. It placed default boxes at various scales on different feature maps, modifying the traditional VGG16 network. The Prior Box layer in SSD is responsible for deploying default boxes at each pixel location in the feature map. The default box coordinates are normalized based on the input image's top-left and bottom-right coordinates and specified values. Bounding box predictions are made through convolutional layer regression parameters. However, in SSD, the minimum and maximum values of prior boxes and aspect ratio values are manually set and not learned from the network. The size and number of default boxes depend on the designer's experience. These issues were later addressed in subsequent networks by using real box clustering before training to avoid manual shortcomings.

SSD also uses large feature maps to detect small objects, but the SSD network does not incorporate feature fusion networks, leading to subpar detection results for small objects. The DSSD algorithm aimed to improve the inadequate detection performance of small objects in SSD. It analysed the problem, stating that while shallow feature maps are suitable for localizing small objects, they lack sufficient semantic information for classification. To address this, DSSD up sampled feature maps using deconvolution and fused them with the original feature maps. The fused feature maps were then fed into a perception network for predictions. DSSD replaced SSD's VGG network with ResNet-101, introduced residual modules before classification and regression, and added deconvolution layers after the auxiliary convolution layers introduced by SSD, creating a "wide-narrow-wide" "hourglass" structure. This effectively improved small object detection but did not significantly enhance detection speed. On the VOC07 dataset with 513×513 input images, the mAP of DSSD achieved 81.5%, with a frame rate of around 6fps.

FSSD [119], as an improved version of SSD, introduces a feature fusion approach that takes feature maps of different scales produced by VGG16 and transforms them into the same scale as the conv4-3 layer using bilinear interpolation. It then concatenates all the feature maps, uses the obtained feature maps for resampling to obtain different feature map scales, and inputs them into a perception network for predictions. On the VOC07 dataset with 300*300 input images, the mAP of FSSD achieves 82.7%. On a single 1080Ti GPU, FSSD reaches a speed of 65.8FPS. FSSD also outperforms native SSD on the COCO dataset, achieving state-of-the-art performance in terms of both speed and accuracy within the SSD series at that time.

(3) The YOLO series of algorithms

The YOLO series of algorithms, as one-stage object detection methods, ensure high accuracy while maintaining exceptionally fast processing speeds. This series has evolved from YOLOv1 [30] to the latest version, YOLOv8, as depicted in Fig. 4.6.

The core concept of YOLOv1 is to input an entire image into the network and obtain detection results and bounding box offsets directly at the output layer through a convolutional neural network. Due to the simplicity of YOLO's network architecture, which consists of only a pre-trained fully convolutional network, four subsequent convolutional layers, and two fully connected layers, it achieves very fast detection speeds but at the cost of lower accuracy, leading to many missed objects. Subsequently, YOLOv2 [122] improved upon YOLOv1 by introducing Batch Normalization (BN) layers and anchor box mechanisms. It removed computationally expensive fully connected layers and pooling layers, which tend to lose feature information. On the VOC12 dataset, YOLOv2 achieved a mAP of 73.5%. However, it still struggled with detecting small objects effectively. YOLOv3 [124] adopted the darknet-53 backbone feature extraction network, implementing multiscale detection inspired by residual networks. To enhance object detection performance across various scales, it introduced a feature pyramid network and employed downsampling to expand the receptive field of each pixel. This resulted in improved performance in detecting small objects. YOLOv4 [125] introduced improvements in the detection head. Similar to YOLOv3, it used three scales to handle objects of different sizes but employed multiple anchor points for each real value.

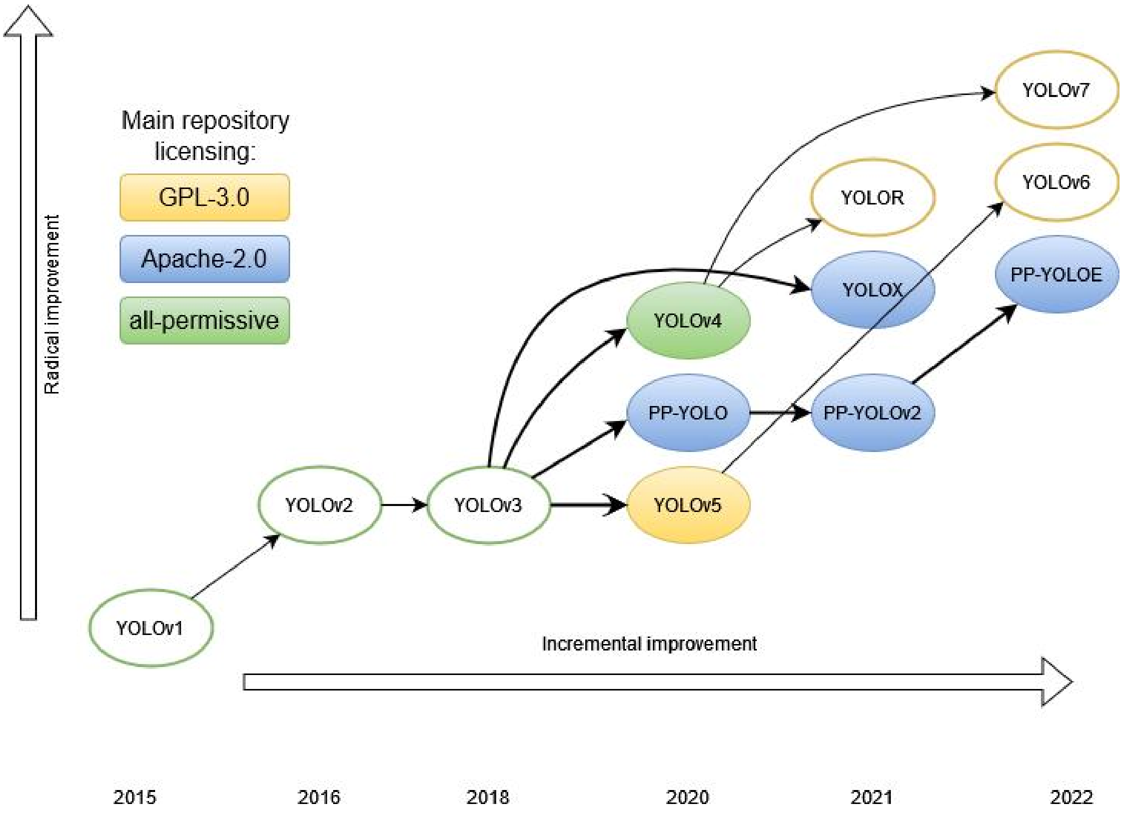


Figure 4.6. Evolution of YOLO Series Models [126].

YOLOv5, considered a leader in real-time object detection, introduced the Focus module, which preserves more complete feature information during image downsampling. It employed two different CSP structures in the backbone and neck, effectively addressing the gradient vanishing problem caused by increased network depth. This improved the network's ability to fuse features, resulting in richer and more accurate feature information. YOLOv5 is known for its speed, model lightweight design, and enhanced accuracy in locating small objects. It offers flexibility with four different network configurations, namely YOLOv5n (nano), YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), and YOLOv5x (extra-large), utilizing channel and layer control factors similar to EfficientNet.

YOLOv6 [127] adopted anchorless detection, featuring a new backbone based on RepVGG, which increased parallelism compared to previous YOLO backbones. For the neck, it used PAN enhancements with RepBlocks or CSPStackRep Blocks, particularly for large models. Performance was further improved through a self-distillation strategy for regression and classification tasks. YOLOv7 [128] introduced the Extended Efficient Layer Aggregation Network (E-ELAN), which improved accuracy without affecting inference speed, albeit extending training time. It achieved more effective learning and convergence in deep models by controlling the shortest and longest gradient paths. E-ELAN was applicable to models with infinitely stacked computational blocks, enhancing network learning through shuffling and merging cardinalities of different feature groups without disrupting the original gradient paths. As of Jan. 2023, Ultralytics (https://www.ultralytics.com/yolo), the company behind YOLOv5, released YOLOv8, offering five versions: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x.

4.3 Experimental Settings

4.3.1 Experimental Environment

The experimental environment in this study involves two platforms: **a conventional PC setting with CPU/GPU and also an embedded implementation**. For the server-side, the system runs on Windows 10 and is equipped with two NVIDIA GeForce RTX 3090 GPUs, 32GB of RAM, and an Intel Xeon Silver 4210 CPU. The total computational power of this setup is 35.6 teraflops. The embedded development platform operates on a Linux-based system NVIDIA Xavier NX. It features a 6-core NVIDIA Carmel 64-bit ARMv8.2 CPU, 384-core NVIDIA Volta GPU, and 8GB of RAM. The computational power reaches 14 teraflops at 10 watts and 21 teraflops at 15 watts. With accelerated computing power of up to 21 TOPS, it can run neural networks in parallel and process data from multiple high-resolution sensors. It is suitable for drones, portable medical equipment, small commercial robots, smart cameras, high-resolution sensors, automatic High-performance AI systems such as optical inspection and other IoT embedded systems. The module appearance of the Xaiver NX is given in Fig. 4.7, along with the block diagram of its process engine illustrated in Fig. 4.8.

Figure 4.7. The module appearance of the Xaiver NX

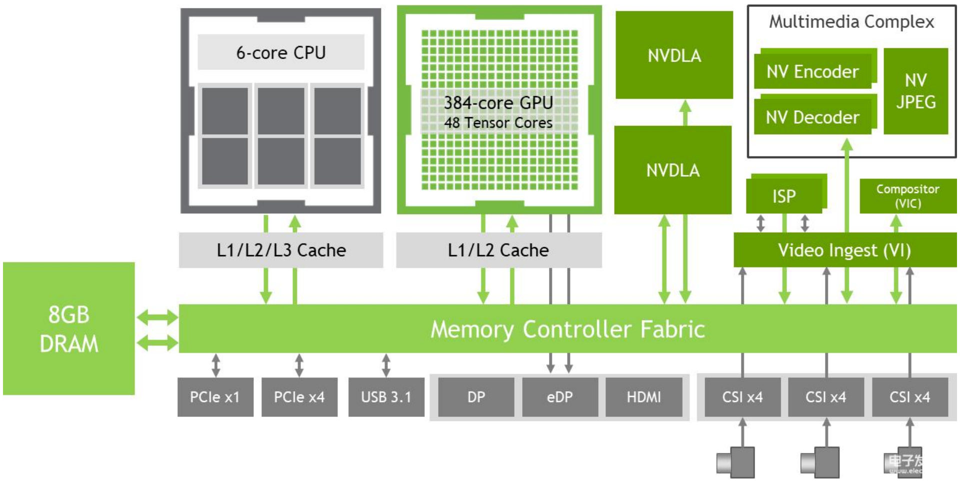


Figure 4.8. The block diagram of the process engine of the Xaiver NX.

As seen, the Xavier NX processor engine includes high-speed I/O and memory structures. Xavier NX's CPU uses the non-Arm public version of Nvidia's self-developed Carmel architecture, which is compatible with the Armv8.2 architecture. The maximum frequency is 1.4-1.9GHz. The Geekbench5 single-core running score is 473, which is twice as high as the Arm Cotoex-A72 in the Raspberry Pi 4. There are three groups of CPU Clusters on the NX chip. Each Cluster contains two Carmel CPUs and a shared 2MB L2 cache. The three Clusters share a 4MB L3 cache. The total Geekbench5 multi-core running score of the 6 CPUs is 2,307, which is the highest in the Raspberry Pi 4. 3x that of 4 Arm Cortex-A72 CPUs. Its GPU uses NVIDIA's previous generation architecture Volta, but the 21Tops computing power it provides can still beat other edge computing chips.

The complete technical specifications of the Jetson Xavier NX module are shown in Table 4.1. To ensure experimental consistency, the framework and software versions used for model development on both platforms remain identical, as listed in Table 4.2.

Table 4.1 Detailed technical specification of the Jetson Xavier NX

AI Performance	21 TOPS
GPU	384-core NVIDIA Volta™GPU equipped with 48 Tensor cores
CPU	6 Kernel NVIDIA Carmel ARM®v8.2 64-bit CPU, plus 6MB L2+4MB L3
Display Memory	8 GB 128bits LPDDR4x, 59.7GB/s

Memory	16 GB eMMC 5.1
Power Consumption	10w |15w |20w
PCIe	One x1(PCIe 3.0)+ one x4(PCIe 4.0), in total 44 GT/s*
CSI Camera	Up to 6 cameras (or 24 via virtual channel settings) 14 channels (3x4 or 6x2）MIPI CSI-2 D-PHY 1.2 (up to Gbps)
Video coding	2x 4K60|4x 4K30|10x 1080p60|22x 1080p30(H.265) 2x 4K60|4x 4K30|10x 1080p60|20x 1080p30(H.264)
Video decoding	2x 8K30|6x 4K60|12x 4K30|22x 1080p60|44x 1080p30(H.265) 2x 4K60|6x 4K30|10x 1080p60|22x 1080p30(H.264)
Display	Two multimodal DP 1.4/eDP 1.4/HDMI 2.0
DL Accelerator	2x NVDLA Engine
Visual Accelerator	7 channel VLIW visual processor
Network	10/100/1000 BASE-T Ethernet
Size & Specifications	69.6mmx 45mm 260 pins, SO-DIMM connector

Table 4.2 List of software versions used.

Software	Version used
Pytorch	1.7
CUDA	10.2
CuDNN	7.6.5
Python	3.8
OpenCV	3.3.1

4.3.2 Datasets

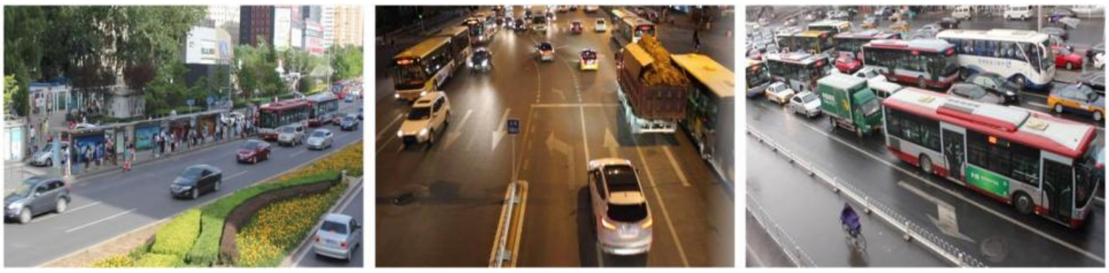
In this study, both real and virtual datasets are utilized. Real datasets possess genuine characteristics such as real backgrounds, lighting conditions, and noise. Using real datasets allows for a better reflection of how objects perform in actual environments, enhancing the accuracy and robustness of object recognition algorithms in real-world applications. Virtual datasets, on the other hand, are generated through computer simulations, enabling control over various data parameters and conditions. They provide a larger pool of data samples, expediting the algorithm's training and testing processes. Additionally, virtual datasets can be used to assess algorithm performance in different scenarios and conditions, providing a more comprehensive performance evaluation.

In summary, the use of both real and virtual datasets complements each other, improving the performance and robustness of object recognition algorithms.

(1) Real Dataset (Public)

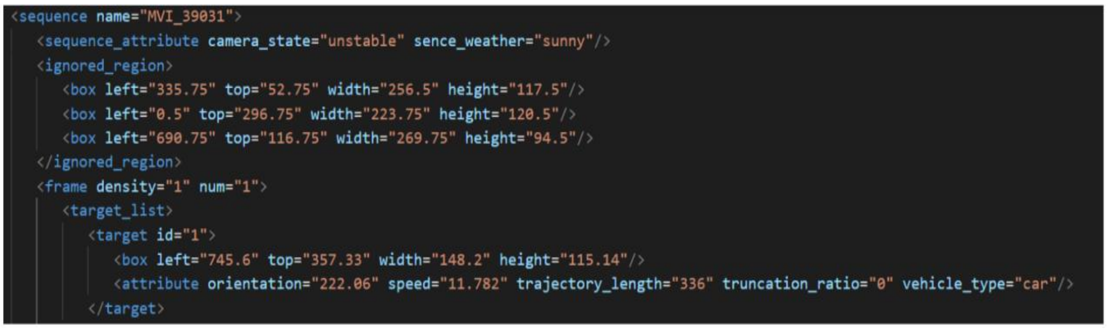
This study utilizes the publicly available UA-DETRAC [133] dataset for object recognition. The UA-DETRAC dataset consists of actual aerial footage recorded at a frame rate of 25 frames per second (fps) with an image resolution of 960×540 pixels. It comprises a total of 140,000 images with 8,250 annotated vehicles and 1.21 million vehicle bounding boxes. The dataset includes various traffic scenes such as intersections and highways and provides video sequences under different weather and time conditions, including daytime, night-time, and rainy conditions, as shown in Fig. 4.9. One advantage of this dataset is that it closely resembles the perspective of a high-altitude camera and is suitable for aerial drone front-end cameras. Additionally, it contains a diverse set of 8,250 vehicles with various shapes, making it suitable for detecting different types of vehicles in real-world scenarios. Therefore, using the UA-DETRAC dataset allows for the better training of object recognition algorithms for vehicle detection in practical applications.

The dataset's folder names correspond to video IDs and contain information about the camera's status, weather conditions, and details for each vehicle within the video sequences. This information includes the vehicle's direction, speed, trajectory length, occlusion rate, and vehicle type, as illustrated in Fig. 4.10.



Daytime Nighttime Rainy Day

Figure 4.9. Sample images from the UA-DETRAC dataset.



```
<sequence name="MVI_39031">
  <sequence_attribute camera_state="unstable" sence_weather="sunny"/>
  <ignored_region>
    <box left="335.75" top="52.75" width="256.5" height="117.5"/>
    <box left="0.5" top="296.75" width="223.75" height="120.5"/>
    <box left="690.75" top="116.75" width="269.75" height="94.5"/>
  </ignored_region>
  <frame density="1" num="1">
    <target_list>
      <target id="1">
        <box left="745.6" top="357.33" width="148.2" height="115.14"/>
        <attribute orientation="222.06" speed="11.782" trajectory_length="336" truncation_ratio="0" vehicle_type="car"/>
      </target>
```

Figure 4.10 Illustration of data labels in the UA-DETRAC dataset.

(2) Virtual Dataset (Non-Public)

This study utilizes a virtually simulated dataset for recognition, as depicted in Fig. 4.11. The dataset comprises a total of 2,035 images within virtual simulated scenes, encompassing various natural environments such as seasonal simulations, extreme weather simulations, and time simulations, as illustrated in Figs. 4.12-4.14. The dataset includes three categories of objects: airports, vehicles, and buildings, with variations in vehicle and building types.

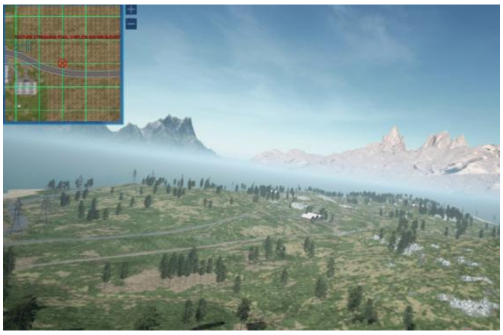
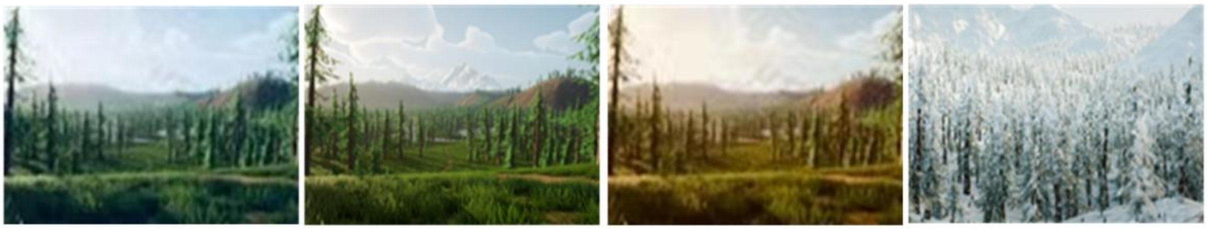


Figure 4.11. Example of one virtually simulated image.



(a) spring (b) summer (c) autumn (d) winter

Figure 4.12 Simulated different seasons of spring to winter (a-d, from left to right).

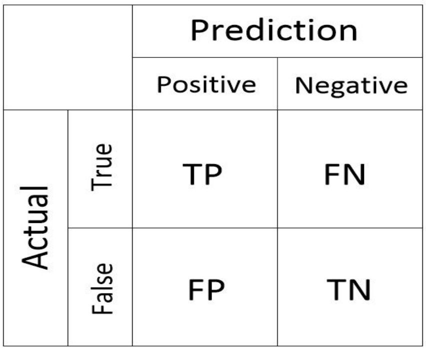
(a) early morning (b) noon (c) evening (d) late night

Figure 4.13 Simulated scenes at different times of a day.

Figure 4.14 Simulated rainy and foggy effects.

4.3.3 Evaluation Metrics

In this study, the evaluation metrics for object recognition primarily include the Precision (True Positives, TP), Recall (False Negatives, FN), Accuracy, Average Precision (AP), and Frames Per Second (FPS). True Positives (TP) represent predictions that are positive and actually true, while False Negatives (FN) represent predictions that are negative but actually true. On the other hand, False Positives (FP) indicate predictions that are positive but actually false, and True Negatives (TN) denote predictions that are negative and actually true. Among these, AP represents the average precision for a category in the dataset, while Accuracy, Precision, and Recall are mainly calculated using the confusion matrix, as shown in Fig. 4.15.



		Prediction	
		Positive	Negative
Actual	True	TP	FN
	False	FP	TN

Figure 4.15 Illustration of the confusion matrix.

$$ACC = \frac{TP + TN}{TP + FN + FP + TN} \qquad (4.1)$$

$$Precision = \frac{TP}{FP + TP} \quad , \quad Recall = \frac{TP}{FP + TN} \#(4.2)$$

4.4 Improved YOLO Implementation and Evaluation

4.4.1 Model Training

The recognition module employs the YOLOv7 algorithm. Initially, the training dataset from the virtual dataset is randomly split into training, validation, and test sets with a ratio of 6:2:2. Given the hardware information of the server, a batch size of 16 is set, while the other hyperparameters remain unchanged. The loss during the training process is depicted in Fig. 4.16, where 'L' represents the loss value (Loss).

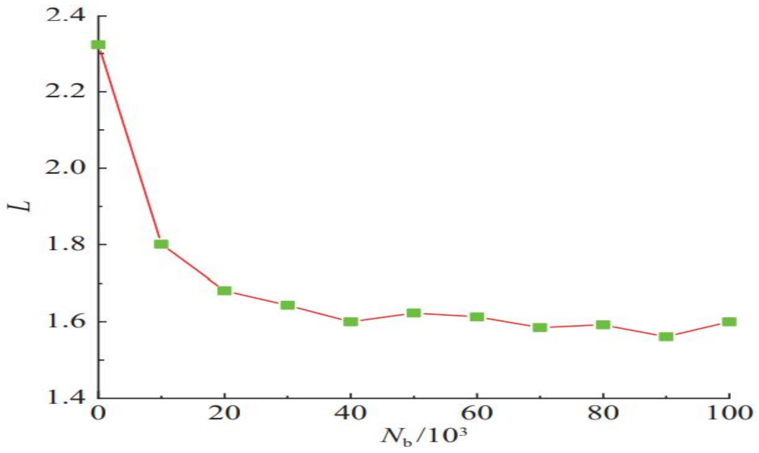


Figure 4.16 Training loss L under varying number of iterations N_b.

Simultaneously, training is conducted using cosine metric learning. A batch size of 32 is set, along with a learning rate of 0.001. The loss function incorporates parameters to prevent category center jitter, with α set to 0.5, and a balancing parameter γ set to 0.085. The training results are illustrated in Fig. 4.17. After multiple iterations, the classification accuracy starts to stabilize and reaches its peak at

94.5%. This improvement in accuracy is attributed to the significant distances between different objects in space, enhancing the model's stability.

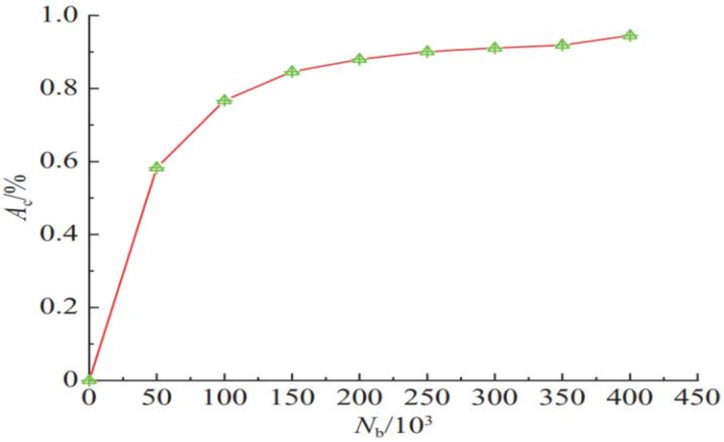


Figure 4.17 Evolutional variations of classification accuracy in training, where A_c and N_b denote the classification accuracy and the number of iterations, respectively.

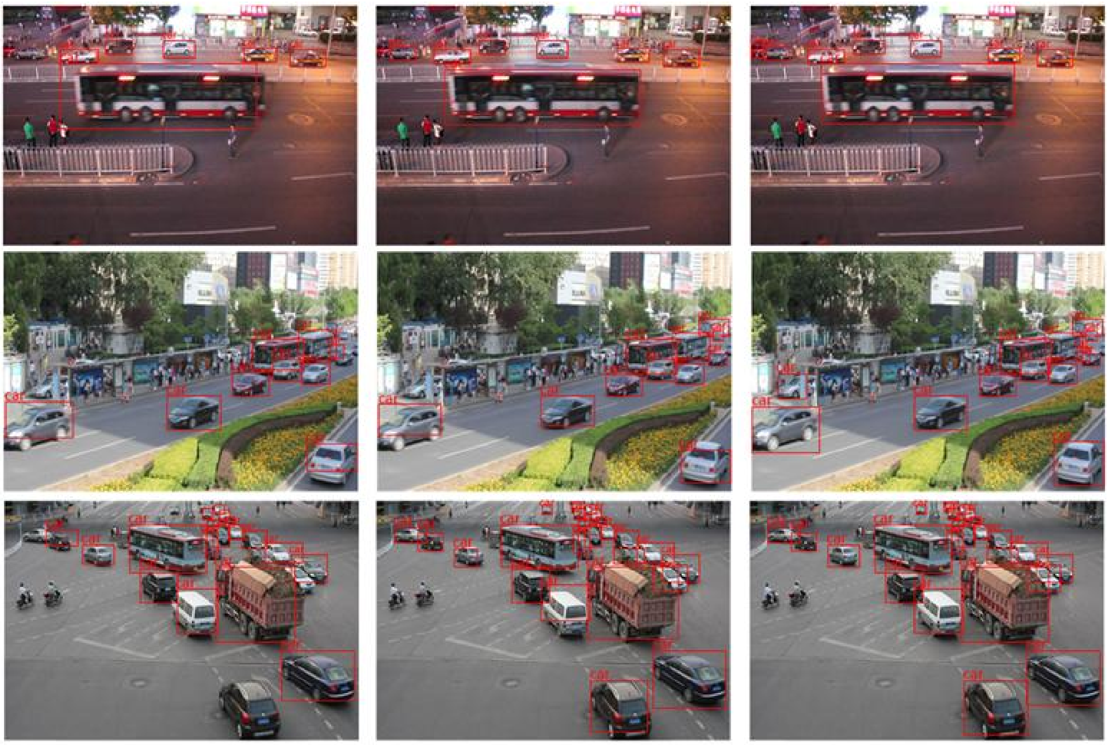
4.4.2 Results from Conventional YOLO in the Server Platform

According to the current research status, YOLO stands out due to its remarkable balance between speed and accuracy. It can rapidly and reliably identify objects in images while also possessing the advantage of model lightweight design, as depicted in Table 4.3.

Choosing the YOLOv3-tiny, YOLOv5s, and YOLOv7 algorithms for implementation on the server, the recognition results of these three models on a public dataset are illustrated in Fig. 4.18. After conducting experiments on the UA-DETRAC dataset, YOLOv7 yielded better results. Consequently, the YOLOv7 was trained on the virtual simulation dataset. However, the average precision (AP) at a threshold of 0.5 is only 0.897. Since the original YOLOv7's recognition accuracy is relatively low in virtual scenes, several improvements have been made.

Table 4.3 Comparison of YOLO Series Architectures and Results [134]

Year	version	Anchor	Framework	Backbone	AP (%)
2015	YOLO	No	Darknet	Darknet24	63.4
2016	YOLOv2	Yes	Darknet	Darknet24	63.4
2018	YOLOv3	Yes	Darknet	Darknet53	36.2
2020	YOLOv4	Yes	Darknet	CSPDarknet53	43.5
2020	YOLOv5	Yes	Pytorch	Modified CSP v7	55.8
2020	PP-YOLO	Yes	PaddlePaddle	ResNet50-vd	45.9
2021	Scaled-YOLOv4	Yes	Pytorch	CSPDarknet	56.0
2021	PP-YOLOv2	Yes	PaddlePaddle	ResNet101-vd	50.3
2021	YOLOR	Yes	Pytorch	CSPDarknet	55.4
2021	YOLOX	No	Pytorch	Modified CSP v5	51.2
2022	PP-YOLOE	No	PaddlePaddle	CSPRepResNet	54.7
2022	YOLOv6	No	Pytorch	EfficientRep	52.5
2022	YOLOv7	No	Pytorch	RepConvN	56.8
2022	DAMO-YOLO	No	Pytorch	MAE-NAS	50.0
2023	YOLOv8	No	Pytorch	YOLO v8	53.9



(a) yolov3-tiny (b) yolov5s (c) yolov7

Figure 4.18 Detection results from images in three datasets

4.4.3 Improvements of the YOLOv7

Several improvements have been introduced to the YOLOv7 model for enhancing its capability in feature extraction, efficiency and efficacy of object detection and classification as detailed below.

4.4.3.1 Add transformer for improved feature extraction

First, a Transformer module is added into the YOLOv7 backbone network to enhance its feature extraction capabilities, as shown in Fig. 4.19. The Transformer is used to replace the last part of C3 in the backbone network to extract image features, as highlighted in yellow in Fig. 4.19.

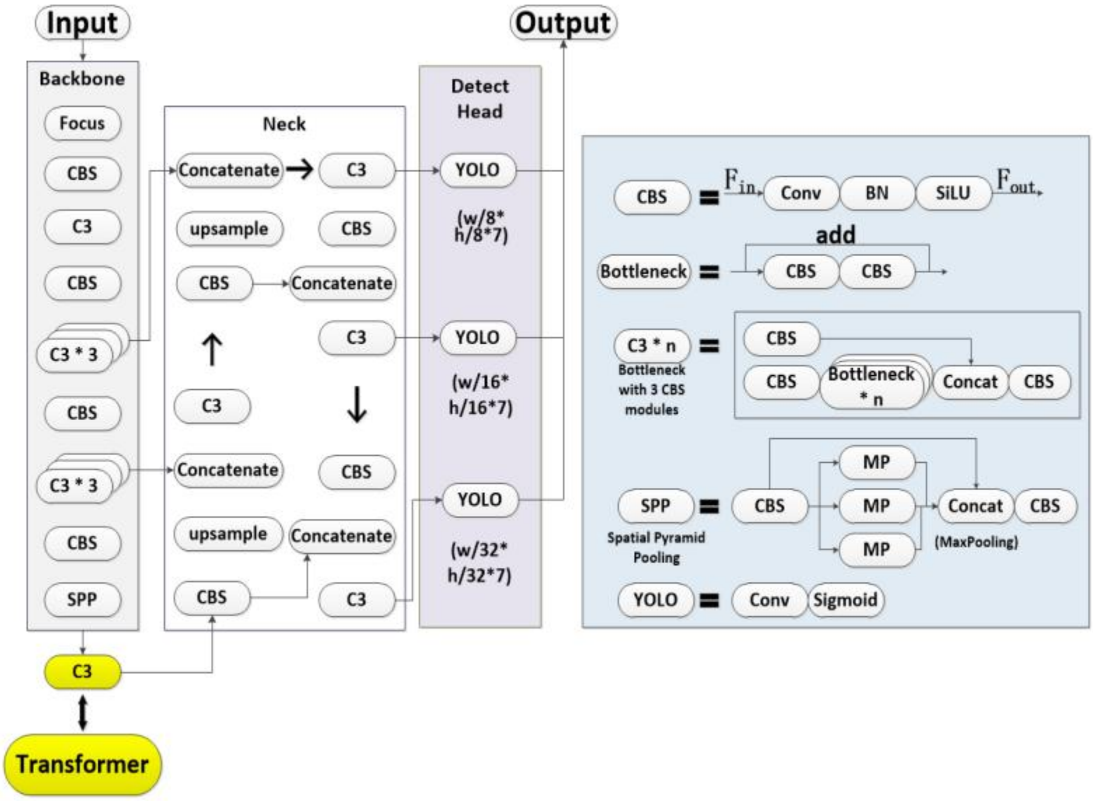


Figure 4.19 Network architecture by adding the Transformer to YOLOv7

The CSPDarknet53 in the original backbone network used for feature extraction and selection consists of a bottleneck structure and three convolutions (C3). In the C3 module, when the feature map Fin of the input of the previous layer is calculated as two parallel branches, as shown in Fig. 4.19, the number of dimensions of Fin is reduced by half and two new feature maps are generated. These two feature maps are concatenated as the output $Fout$ of the C3 module. Use Transformer to replace the last part of C3 in the backbone network to extract image features, which enhances the capability of the feature extraction network and improves detection accuracy.

As shown in Fig. 4.19, the replaced Transformer is connected to the CBS module. The CBS module is the basic module that makes up the network. It consists of the convolution layer (Conv), batch normalization (BN) and activation layer, and is combined with Sigmoid Weighted Linear Unit (SiLU).

4.4.3.2 Introducing the CIoU loss function

After modifying the model structure, in order to improve the training speed of the model, the CIoU loss function is used in the network structure. The loss function in YOLOv7 is the GIoU loss function. When the prediction box and the real box are completely surrounded, GIoU loses the ability to be further optimised. GIoU depends on the IoU term, which will cause oscillation during the training process and is difficult to converge. Therefore, it is necessary to improve the loss function. The GIoU formula is given as follows.

$$GIoU = IoU - \frac{A^c - u}{A^c} \text{ , } L_{GIoU} = 1 - GIoU \#(4.3)$$

where L_{GIoU} is the corresponding loss function.

In Eq. 4.3, IoU is the intersection ratio of the predicted box and the real box. The smallest rectangle containing both the real box and the predicted box is A^c, and u is the union of the two boxes. CIoU can simultaneously calculate the center offset, aspect ratio and overlap size of the real and predicted boxes, as defined below.

$$CIou = IoU - \frac{\rho^2(b, b^{gt})}{c^2} - \alpha v \text{ , } L_{CIoU} = 1 - CIoU \#(4.4)$$

Among them, b and b^{gt} represent the centers of the two frames, α is the balance parameter, v represents the consistency of the aspect ratio between the real frame and the detection frame, ρ^2 denotes the calculation of the Euclidean distance between the two frames, and c represents the minimum of the diagonal length of the rectangle. The definitions of α and v are given below, where w^{gt}, h^{gt} and w, h represent the width and height of PB and GT, respectively.

$$\alpha = \frac{v}{(1 - IoU) + v} \#(4.5)$$

$$v = \frac{4}{\pi^2}(arc\tan\frac{w^{gt}}{h^{gt}} - arc\tan\frac{w}{h})^2 \#(4.6)$$

As shown in Fig. 4.20, when the regression box has an inclusion relationship, the IoU Loss values are the same, and the GIoU Loss values of the three regression boxes are the same. It is impossible to judge which regression effect is better, but the CIoU Loss values are not the same. Obviously The regression of the third box is better.

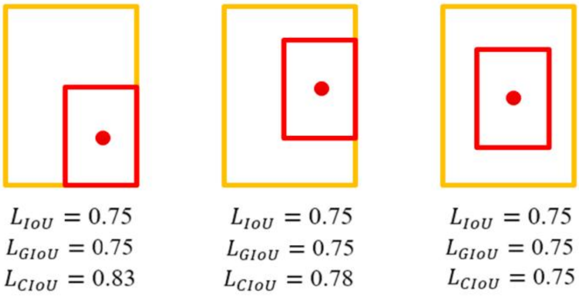


Figure 4.20 Comparison of IoU, GIoU and CIoU in different predicted boxes.

4.4.3.3 Add the GhostBottleneck module for improved efficiency

Meanwhile, the GhostBottleneck module is used in the network structure to reduce the number of parameters and the associated computational complexity, which has not only ensured the detection performance but also reduced the computational costs. Since the backbone network of YOLOv7 has a large amount of calculation, optimising the backbone network can reduce the number of model parameters to a certain extent. By replacing the CSP module in YOLOv7 and using GhostBottleneck in the lightweight network ghost network GhostNet [37], the amount of model parameters is reduced.

It found from the GhostNet experiments that in ordinary CNNs, the layered output feature maps have many similarities. Therefore, if a nonlinear convolution is used, linear convolution is then performed on this basis to obtain the features. Graph, on the basis of ensuring the feature extraction capability, can reduce a certain computational complexity. GhostNet first obtains a feature map through nonlinear changes, and then performs linear changes on this basis to obtain the ghost feature map. The GhostNet Bottleneck consists of two stacked Ghost modules, with the first Ghost module acting as an expansion layer, increasing the number of channels. The second Ghost module reduces the number of channels to facilitate superposition with the residual network. Finally, residual connections are used before and after the two Ghost modules to strengthen feature transfer and gradient return, ensuring the feature extraction capability of the network.

4.4.4 Results of the improved YOLOv7

In order to validate the effectiveness of the aforementioned improvements, ablation study experiments were conducted on the virtual simulation dataset by sequentially adding the Transformer module, CIoU Loss, and the GhostBottleneck module. The experiment weights were initialized randomly, and the Adam optimizer was employed with an initial learning rate of 0.001. Three learning rate decays were performed during the experiments, each reducing the learning rate to half of its current value. Additionally, weight decay was applied during training with a decay rate of 0.0001. The results of the YOLOv7 ablation experiments are presented in Table 4.4.

Table 4.4 Ablation study of the improved YOLOv7

Transformer	CIoULoss	GhostBottleneck	Precision	Recall	AP@0.5
×	×	×	0.901	0.942	0.897
√	×	×	0.929	0.966	0.924
√	×	√	0.927	0.966	0.922
×	√	×	0.908	0.945	0.909
×	√	√	0.908	0.945	0.909
√	√	×	0.939	0.969	0.933
√	√	√	0.938	0.969	0.932

In Table 4.4, the first row is the basic performance of the original YOLOv7 algorithm framework on the virtual data set. It can be seen that the average detection accuracy AP@0.5 is 0.897. Then only after the Transformer module was introduced in YOLOv7, both Precision and Recall improved, and AP@0.5 increased by 3.01%. When the GhostBottleneck module is introduced at the same time as the Transformer module, the change in AP@0.5 is small.

Only in YOLOv7, the loss function GIoULoss was improved to CIoULoss. Compared with the original YOLOv7 algorithm, although both Precision and Recall increased, the improvement of detection accuracy by only improving the loss function was very small, and AP@0.5 only increased by 1.33%. When the GhostBottleneck module is introduced when CIoULoss is introduced, there is no change in performance.

Finally, the Transformer, CIoULoss and GhostBottleneck modules were all introduced into YOLOv7. Compared with the original algorithm, Precision and Recall were greatly improved, and AP@0.5 increased by 3.90%. Although the Precision and AP@0.5 have slightly decreased after the introduction of the GhostBottleneck module, the overall model parameters have been reduced from the original 7.5M to 4.85M, as shown in Table 4.5. In other words, the GhostBottleneck module can maintain about the same detection accuracy yet significantly reduce the number of parameters and computational complexity to benefit the embedded implementation.

Table 4.5 Comparison of model complexity with the GhostBottleneck

Model	Main module	Parameters	CPU time	GPU time
Original	BottleneckCSP	7.5 M	1.25 s	0.037 s
Add GhostBottleneck	GhostBottleneck	4.85 M	0.98 s	0.031 s

To sum up, first, by introducing the Transformer module into the backbone network, the capability of the feature extraction network is enhanced and the detection accuracy when the target pixels are small is improved. Second, by adding the CIoU Loss loss function in the network, it speeds up the bounding box regression rate and improves the positioning accuracy. Finally, the GhostBottleneck module is used in the network structure to reduce the amount of parameters and computational complexity, which not only ensures detection performance but also reduces computational costs and alleviates the problem of limited device computing power. The experimental results have validated the efficacy of the improved three improvements. In addition, visual results from the improved YOLOv7 on the simulated virtual scenes are given in Fig. 4.21 for visual comparison.

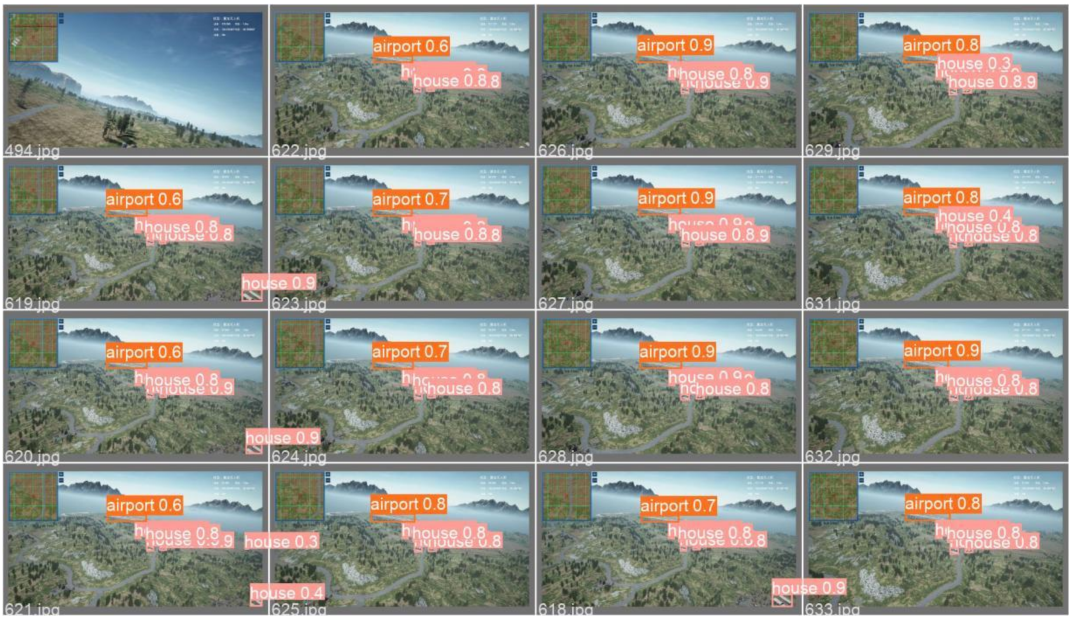


Figure 4.21 Detection results in virtually simulated scenes.

4.5 Further Comparison with Embedded Implementation

4.5.1 Quantitative Comparison

After training and recognition on the server side, the optimal weight files of YOLOv7 and the improved YOLOv7 algorithm were ported to the embedded development environment. The embedded experimental platform selected for this purpose is the NVIDIA Xavier NX. This platform is compact in size and provides powerful computing performance for edge computing applications. Leveraging its up to 21 TOPS (trillion operations per second) of acceleration computing power, it can parallelize the execution of neural networks and process data from multiple high-resolution sensors.

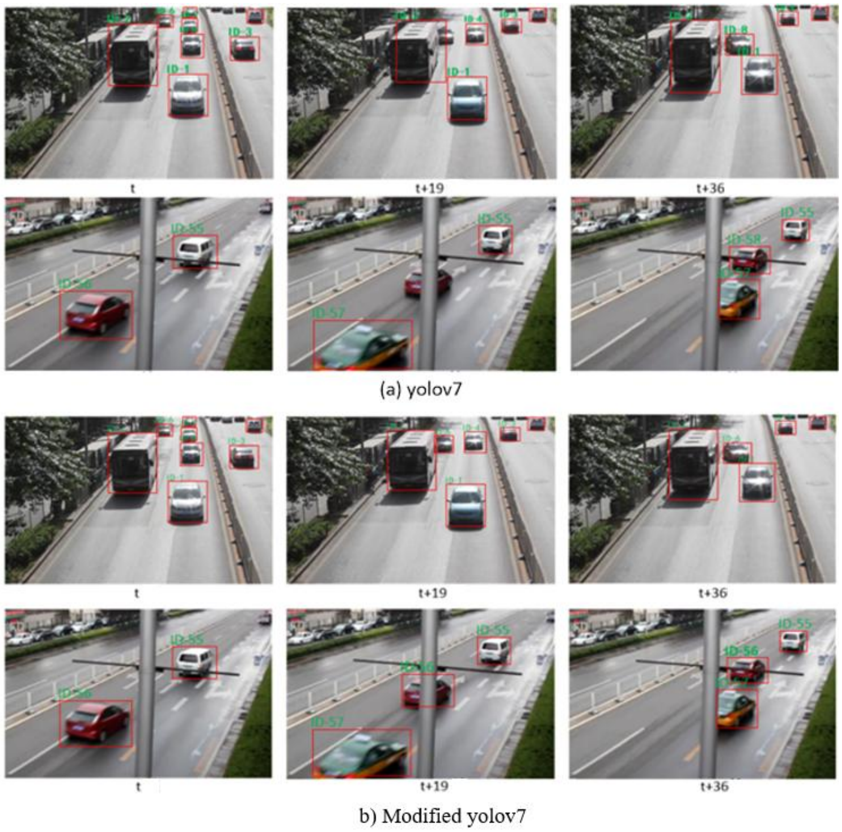
Herein, the original YOLOv7 and the improved YOLOv7 were compared on the embedded platform, where both YOLOv7 and the improved YOLOv7 were implemented under the PyTorch framework. The results were shown in Table 4.6 for comparison. As can be seen, the improved YOLOv7 has not only produced much better results in object detection and classification, as already shown in Table 4.4, the computational efficiency in terms of reduced CPU/GPU usage and increased frame rate of processing has further demonstrated its suitability for real-time applications.

Table 4.6 Results of YOLOv7 and improved YOLOv7 on the NX Platform

Algorithms	**Computational process**			
	Detection and recognition	GPU	CPU	Actual utilization rate
YOLOv7	Image detection time (seconds)	1.12	2.9	
	Real-time camera detection frame rate (FPS)	7.4	0.4	CPU Processing: 6 CPUs is consistently close to 65%. GPU Processing: 6 CPUs is approximately 20%, while the GPU utilization is at 99%.
	Video test frame rate (FPS)	7.4	0.4	CPU Processing: 6 CPUs is consistently close to 65%. GPU Processing: 6 CPUs is approximately 35%, while the GPU utilization is at 99%.
Modified YOLOv7	Image detection time (seconds)	0.63	0.36	
	Real-time camera detection frame rate (FPS)	22-28	3.4	CPU Processing: 6 CPUs is consistently close to 60%. GPU Processing: 6 CPUs is approximately 25%, while the GPU utilization is at 30-70%.
	Video test frame rate (FPS)	18-26	3.4	CPU Processing: 6 CPUs is consistently close to 58%. GPU Processing: 6 CPUs is approximately 60%, while the GPU utilization is at 30-70%.

4.5.2 Visual Comparison

In addition, visual comparison from the UA-DETRAC dataset and the simulated virtual scenes are shown in Fig. 4.22 and Fig. 4.23, respectively. As seen, the improved YOLOv7 has achieved much better the consistency of the object IDs, which has clearly demonstrated its efficacy in object detection and recognition under complex scenarios.



(a) yolov7

b) Modified yolov7

Figure 4.22. Visual results from UA-DETRAC dataset on the NX Platform

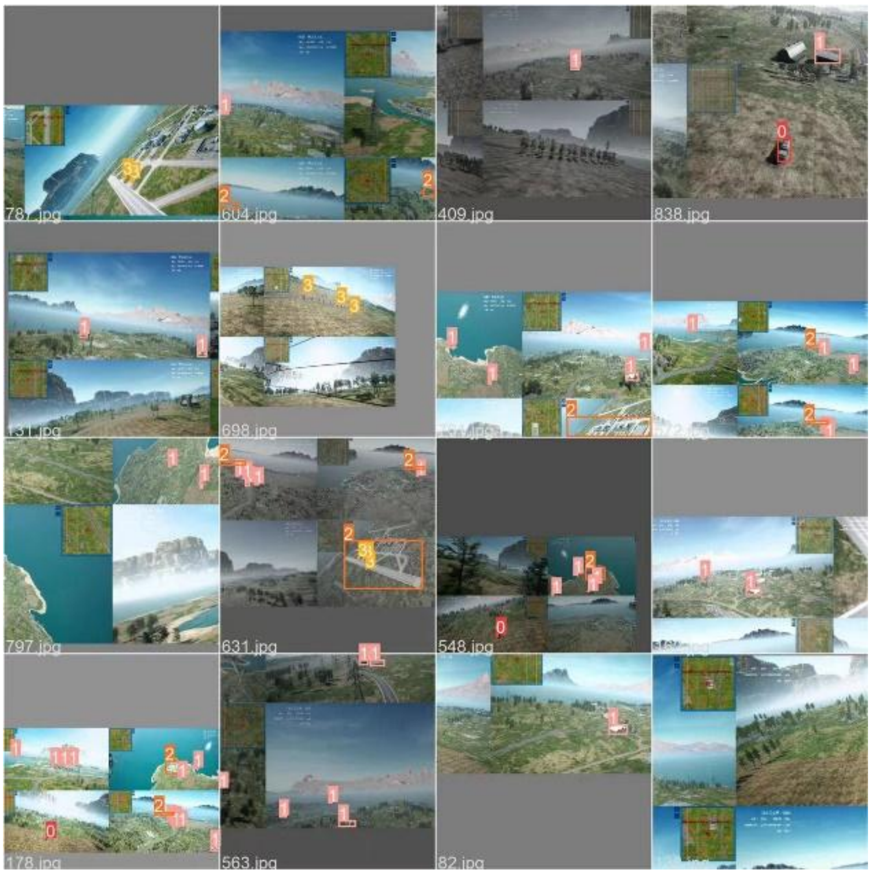


Figure 4.23. Visual results from the simulated virtual scenes.

4.6 Summary

In this chapter, an improved YOLOv7 is presented for more effective and efficient object detection and recognition from UAV images, by adding the following three key components, i.e. transformer, CIoULoss and GhostBottleneck. As a result, the precision, recall and AP@05 have all been improved, which have validated the enhanced performance by adding these modules.

In addition, embedded implementation of the YOLOv7 and the improved one is also compared. The CPU usage, memory consumption and running time for detection have all been significantly improved. Increasing the detection frame rate from 7.4 FPS to 22-28 has enabled more chance for real-time implementation in order to satisfy a wide range of inspection and surveillance applications.

Chapter 5. Conclusions and Future Work

5.1 Contributions and Conclusions

In this thesis, several challenges are focused in UAV based SLAM applications, aiming to provide efficient and effective solutions to enable edge-computing based deployment and thus benefit a wide range of inspection and survey applications. These include i) image enhancement of UAV images, ii) effective object detection and recognition, and iii) embedded implementation for improved efficiency. Accordingly, innovative solutions are developed in the thesis, along with research publications to highlight the value of the work.

The major contributions and useful findings of the thesis are summarised as follows.

1) In Chapter 2, a detailed survey of related work in UAV based SLAM applications is focused, covering the basic knowledge of UAVs and aerial image processing as well as computer vision techniques in UAV applications and current status of visual SLAM development. This can provide valuable insights for people who are interested in the area of work, e.g. vision-based UAV applications.
2) In Chapter 3, evaluation and assessment of the models for UAV image enhancement are performed, covering denoising and dehazing, respectively. Relevant approaches are first reviewed, including conventional ones in spatial domain and transform domain as well as DL models.

   For denoising of UAV images, filtering based approaches including BM3D, WNNM and a few DL based approaches are selected for assessment, the latter covers VDN, FFDNet and NAFNet for their better performance than the peers. Extensive evaluations are carried out in both publicly available and self-collected datasets. It is worth noting that the conventional method of BM3D seems to perform comparable or even slightly better than the DL models. This has verified the great potential of conventional approaches in UAV image enhancement, and has also demonstrated space for further exploration of the DL models. Useful discussions are provided to analyse these results.

For dehazing of UAV images, several DL models are selected for evaluation, including DehazeFormer, FFANet, GriddehazeNet, GCANet and PMNet, again using both publicly available and self-collected datasets. Overall, the DehazeFormer model outperforms all others in the publicly available dataset. For the self-collected dataset, FFANet seems to have superior performance than other compared models. These results indicate the potential limitations of the existing approaches, hence in-depth analysis of the key models and further integration of different approaches can be beneficial.

3) In Chapter 4, object detection and recognition from UAV images are studied, using an improved YOLOv7 model along with embedded implementation. Although the YOLO series of models seem to produce quite promising results of object detection in various applications, the performance was constrained in the tested datasets due mainly to the poor quality and less distinguishable features. Also, these YOLO algorithms suffer from too heavy computational loads and resource consumption to fit the edge computing. These issues have motivated the improvements on the YOLOv7 as detailed in the chapter.

   Specifically, three important improvements are introduced to the existing YOLOv7 model. These include adding the transformer to improve feature extraction, introducing the CIoU loss function to increase the training speed due to easy of convergence, and appending the GhostBottleneck module to further reduce the number of parameters and computational complexity. With these additions, all the quantitative measures in terms of the precision, recall and average precision have been improved by 2.7-3.7%, a significant achievement on top of the existing YOLOv7.

   In addition, embedded implementation of YOLOv7 and the improved version is conducted to further validate the efficiency of the improvements. The resources consumption of the GPU and CPU has been significantly reduced from 99% and 65% to 30-70% and less than 60%, respectively. The processing times for GPU and CPU based detection has been reduced by 48% and 85%, respectively. This has led to the increasing frame rate for real-time applications from 7.4 (GPU) or 0.4 (CPU) to 18-28 (GPU) and 3.4 (CPU), respectively. These have demonstrated the value of the proposed improvements to facilitate the edge-computing based real-time deployment of the functional modules.

5.2 Future Work

Although innovative solutions have been developed to address the challenging issues identified in the thesis in terms of image enhancement, object detection and recognition and embedded implementation for efficiency. There are utterly unsolved issues that would need further investigation as detailed below.

1) **Image enhancement**: Despite of the success in other tasks such as object detection and classification, DL based UAV image enhancement, particularly for denoising and dehazing, shows few improvements over conventional approaches, such as BM3D. This suggests remained space for improvements, and the potential to combine conventional approaches with DL in this particular task.

   In addition, there are other challenging topics in image enhancement in UAV applications, such as the low-light environment, poor stability caused blurring, and low spatial resolution. As a result, new models and approaches need to be derived to tackle these challenges.

   Moreover, as the quantitative metrics for quality assessment, using PSNR and SSIM, seem inconsistent in many cases. How to derive more consistent and effective metrics for quality assessment of the enhanced images should be investigated.

2) **Object detection and recognition**: Although the improved YOLOv7 model has shown significantly reduced computation load, increased processing efficiency, as well as the higher detection accuracy, the CPU usage and memory requirement are still high. As such, performance optimisation of the architecture including parallel implementation of the key modules will be explored in the future. This can further improve the efficiency and usability especially for edge computing.

   In addition, testing and implementation on most recent YOLO models such as YOLOv8 will be continued in the future work. As an extended and further optimised model from YOLOv7, YOLOv8 is claimed to have further enhanced accuracy and efficiency, though the exact performance improvements may vary, depending on the implementation and the application scenarios. Therefore, it is worth applying more work on YOLOv8 to further improve the performance.

Furthermore, integration of other DL modules for multi-task learning based joint optimisation can be investigated, such as to consider the image enhancement with the object detection and recognition module. In this way, the sequential workflow can be optimised simultaneously in order to improve the efficiency and efficacy.

3) **SLAM based autonomous deployment**: As a high-demanding task in UAV inspection, SLAM based autonomous deployment plays a key role in a wide range of application scenarios. As such, scene matching navigation can be focused, based on the outcomes from the thesis. Different from conventional approaches using the detected corners [135] and local features using the scale invariant feature transform (SIFT) [136], speeded up robust features (SURF) and Oriented FAST and Rotated BRIEF (ORB) [137], DL based scene matching provides a useful alternative for future exploration, especially with sub-pixel accuracy. The robustness, accuracy and efficiency will be the three key factors when designing, developing and evaluating such approaches.

Bibliography

[1] Y. Liu and Q. Dai, "Vision aided unmanned aerial vehicle autonomy: An overview," in *Proc. 3rd Int. Congress on Image and Signal Processing*, pp. 417-421, 2010.

[2] A. Couturier and M. A. Akhloufi, "A review on absolute visual localization for UAV," *Robotics and Autonomous Systems*, vol. 135, pp. 103666, 2021.

[3] Y. Ma, Q. Li, L. Chu, Y. Zhou and C. Xu, "Real-time detection and spatial localization of insulators for UAV inspection based on binocular stereo vision," *Remote Sensing*, vol. 13, (2), pp. 230, 2021.

[4] E. T. Dill, “GPS/Optical/Inertial Integration for 3D Navigation and Mapping using Multi-Copter Platforms,” PhD thesis, Ohio University, 2015.

[5] B. T. Baeder and J. L. Rhea, "GPS attitude-determination analysis for UAV," in *Navigation and Control Technologies for Unmanned Systems*, pp. 232-243, 1996.

[6] E. Petritoli, F. Leccese and M. Leccisi, "Inertial navigation systems for UAV: Uncertainty and error measurements," in *Proc. IEEE 5th Int. Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pp. 1-5, 2019.

[7] T. Guan, G. Zhang and P. Chen, "A terrain matching navigation algorithm for UAV," in *Proc. 33rd Chinese Control and Decision Conf.*, pp. 5203-5207, 2021.

[8] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," *IEEE Robotics & Autom. Magaz.*, vol. 13, (3), pp. 108-117, 2006.

[9] A. Steenbeek and F. Nex, "CNN-based dense monocular visual SLAM for real-time UAV exploration in emergency conditions," *Drones*, vol. 6 (3), pp. 79, 2022.

[10] A. Aghamohammadi, A. H. Tamjidi and H. D. Taghirad, "SLAM using single laser range finder," *IFAC Proceedings Volumes*, vol. 41, (2), pp. 14657-14662, 2008.

[11] N. Karlsson, E. Di Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian and M. E. Munich, "The vSLAM algorithm for robust localization and mapping," in *Proc. the IEEE Int. Conf. on Robotics and Automation*, pp. 24-29, 2005.

[12] P. Ivanov, “Visual Localization using a Three-Dimensional Model and Image Segmentation”, US11232582B2, 2022.

[13] P. Zhang, C. Zhang, B. Liu and Y. Wu, "Leveraging local and global descriptors in parallel to search correspondences for visual localization," *Pattern Recognition*, vol. 122, pp. 108344, 2022.

[14] https://www.iwm.org.uk/history/a-brief-history-of-drones, Date of access: 18/03/2023

[15] J. Wu, Q. Shi, Q. Lu, X. Liu, X. Zhu and Z. Lin, "Learning invariant semantic representation for long-term robust visual localization," *Engineering Applications*

of Artificial Intelligence, vol. 111, pp. 104793, 2022.

[16] https://www.loginextsolutions.com/blog/drones-uav-last-mile-delivery/, Date of access: 22/03/2023

[17] https://en.wikipedia.org/wiki/Unmanned_aerial_vehicle, Date of access: 28/03/2023

[18] J. Kinnari, R. Renzulli, F. Verdoja and V. Kyrki, "LSVL: Large-scale season-invariant visual localization for UAVs," *Robotics and Autonomous Systems*, vol. 168, pp. 104497, 2023.

[19] Y. Schechner, G. Srinivasa et al. "Polarization-based vision through haze," *Applied Optics*, vol. 42, no.3, pp.511-525, 2003.

[20] R. T. Tan, "Visibility in bad weather from a single image," In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1-8, 2008.

[21] R. Fattal, "Single image dehazing," *ACM Trans. on Graphics (TOG),* vol. 27, (3), pp. 1-9, 2008.

[22] L. Kratz and K. Nishino, "Factorizing scene albedo and depth from a single foggy image," in *Proc. IEEE 12th Int. Conf. Comput. Vis*. (ICCV), Sep./Oct., pp. 1701–1708, 2009.

[23] S. Gamini and S. Kumar, "Homomorphic filtering for the image enhancement based on fractional-order derivative and generic algorithm," *Computers and Electrical Engineering*, vol. 196: 108566, 2023.

[24] D. J. Jobson, Z. Rahman, and G. A. Woodell, "Properties and performance of a center/surround retinex," *IEEE Trans. on Image Processing*, vol. 6, pp. 451–462, March 1997.

[25] E. H. Land and J. J. McCann, "Lightness and retinex theory," *Josa,* vol. 61, no 1, pp. 1-11, 1971.

[26] B. Cai, X. Xu, K. Jia, et al, "DehazeNet: an end-to-end system for single image haze removal," *IEEE Trans. on Image Processing*, 25(11): 5187-5198, 2016.

[27] H. Zhang, V.M. Patel, "Densely connected pyramid dehazing network," in *Proc. 32nd IEEE Conf. on Computer Vision and Pattern Recognition*, Salt Lake City, USA， June 19-21, 2018, 2018:2261-2269.

[28] https://journals.sagepub.com/doi/abs/10.1068/p7297, Date of access: 02/04/2023

[29] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 580-587, 2014.

[30] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 779-788, 2016.

[31] K. Okuma, A. Taleghani, N. De Freitas, J. J. Little and D. G. Lowe, "A boosted

particle filter: Multitarget detection and tracking," in Proc. *8th European Conf. on Computer Vision, Prague, Czech Republic, May 11-14, 2004*, pp. 28-39.

[32] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. the IEEE Computer Society Conf. on Computer Vision and Pattern Recognition,* pp. I, 2001.

[33] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu and A. C. Berg, "Ssd: Single shot multibox detector," in Proc. *14th European Conf. on computer Vision, Amsterdam, the Netherlands, October 11–14,* pp. 21-37, 2016.

[34] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *Proc. ICLR*, 2014.

[35] R. Girshick, "Fast R-CNN," *In Proc. the IEEE Int. Conf. Computer Vision*, 2015.

[36] T. Huang, M. Cheng, Y. Yang, X. Lv and J. Xu, "Tiny object detection based on YOLOv5," in *Proc. ICIGP*, pp. 45-50, 2022.

[37] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in Proc. *6th IEEE and ACM Int. Symp. on Mixed and Augmented Reality*, pp. 225-234, 2007.

[38] R. A. Newcombe, S. J. Lovegrove and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in Proc. *Int. Conf. on Computer Vision*, pp. 2320-2327, 2011.

[39] N. Houshangi and F. Azizi, "Accurate mobile robot position determination using unscented Kalman filter," in *Canadian Conference on Electrical and Computer Engineering*, pp. 846-851, 2005.

[40] G. Rigatos, P. Siano and G. Raffo, "Distributed control of unmanned surface vessels using the derivative-free nonlinear Kalman filter," *Intelligent Industrial Systems*, vol. 1, pp. 99-126, 2015.

[41] C. Fu, W. Liu, A. Ranga, A. Tyagi and A. C. Berg, "DSSD: Deconvolutional single shot detector," *arXiv Preprint arXiv:1701.06659,* 2017.

[42] M. Dai, J. Hu, J. Zhuang and E. Zheng, "A transformer-based feature segmentation and region alignment method for UAV-view geo-localization," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 32, (7), pp. 4376-4389, 2021.

[43] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The Int. Journal of Robotics Research*, vol. 5, (1), pp. 90-98, 1986.

[44] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proc. IEEE Int. Conf. Robotics and Autom.*, pp. 3310-3317, 1994.

[45] D. Tsishkou, T. Yin, A. Moreau, "Devices and methods for visual localization". *WO2022117192A1*, 2022.

[46] W. Zeng, R.L. Church, “Finding shortest paths on real road networks: the case for A*”, *Int. Journal of Geographical Information Science*, 23 (4): 531–543, 2009.

[47] L.C. Markley, J. F. Lindne, "Artificial gravity field," *Results in Physics*, Volume 3, Pages 24-29, 2013.

[48] I. Leduc-Cummings, K. M. Werner, M. Milyavskaya, J. K. Dominick, S. Cole, “Experiencing obstacles during goal pursuit: The role of goal motivation and trait self-control,” *Journal of Research in Personality*, vol. 99, 2022.

[49] D. H. Kim, D. H. Lee and W. Y. Kim. “Method of Generating Map and Visual Localization System using the Map”, *US2022139032A1*, 2022.

[50] L Zhu, J Wang, Y Wang, Y Ji, J Ren， "DRL-RNP: Deep reinforcement learning-based optimized RNP flight procedure execution," *MDPI Sensors*, 22 (17), 6475, 2022.

[51] https://apps.dtic.mil/sti/citations/ADA475002, Date of access: 03/04/2023

[52] H. Bao, G. Zhang, Y. U. Hailin, “Methods for visual localization and related apparatus”, *US2022148302A1*, 2022.

[53] A. R. Siddiqui, On Fundamental Elements of Visual Navigation Systems, *PhD thesis*, Blekinge Institute of Technology, 2014.

[54] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. on Robotics*, vol. 33, (5), pp. 1255-1262, 2017.

[55] C. Forster, M. Pizzoli and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in Proc. *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pp. 15-22, 2014.

[56] Y. Hu, P. Liu, H. Chen and S. Li, "Autonomous Scene Matching Navigation Based on Range Optimization of Reference Image by Interframe Matching," In *Proc. 5th Int. Conf. Pattern Recognition and Artificial Intelligence*, pp. 903-909, doi: 10.1109/PRAI55851.2022.9904193, 2022.

[57] D. H. Kim, H. L. Dong, W. Y. Kim, “Method and system for visual localization”, *US2022148219A1*, 2022.

[58] R. Chatila and J. Laumond, "Position referencing and consistent world modeling for mobile robots," in *Proc. IEEE Int. Conf. Robotics and Automation*, 1985, pp. 138-145.

[59] R. Smith, M. Self and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous Robot Vehicles*, Springer, 1990, pp. 167-193.

[60] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. on Robotics,* vol. 32, (6), pp. 1309-1332, 2016.

[61] H. Strasdat, J. M. Montiel and A. J. Davison, "Visual SLAM: why filter?" *Image Vision Comput.,* vol. 30, (2), pp. 65-77, 2012.

[62] J. Kinnari, F. Verdoja and V. Kyrki, "Season-invariant GNSS-denied visual localization for UAVs," *IEEE Robotics and Automation Letters*, vol. 7, (4), pp. 10232-10239, 2022.

[63] J. Qian, K. Chen, Q. Chen, Y. Yang, J. Zhang and S. Chen, "Robust visual-LiDAR simultaneous localization and mapping system for UAV," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1-5, 2021.

[64] K. Celik, S. Chung and A. Somani, "Mono-vision corner SLAM for indoor navigation," in *Proc. IEEE Int. Conf. on Electro/Inform. Techn.,* 2008, pp. 343-348.

[65] I. Dryanovski, R. G. Valenti and J. Xiao, "Fast visual odometry and mapping from RGB-D data," in *IEEE Int. Conf. on Robotics and Automation,* pp. 2305-2310, 2013.

[66] R. G. Valenti, I. Dryanovski, C. Jaramillo, D. P. Ström and J. Xiao, "Autonomous quadrotor flight using onboard RGB-D visual odometry," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA),* pp. 5233-5238, 2014.

[67] J. Conroy, G. Gremillion, B. Ranganathan and J. S. Humbert, "Implementation of wide-field integration of optic flow for autonomous quadrotor navigation," *Autonomous Robots,* vol. 27, pp. 189-198, 2009.

[68] M. J. Milford, F. Schill, P. Corke, R. Mahony and G. Wyeth, "Aerial SLAM with a single camera using visual expectation," in *Proc. IEEE Int. Conf. on Robotics and Automation,* pp. 2506-2512, 2011.

[69] C. Forster, Z. Zhang, M. Gassner, M. Werlberger and D. Scaramuzza, "SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems," *IEEE Trans. on Robotics*, vol. 33, no. 2, pp. 249-265, April 2017.

[70] F. Zangeneh, L. Bruns, A. Dekel, A. Pieropan and P. Jensfelt, "A Probabilistic Framework for Visual Localization in Ambiguous Scenes," *arXiv Preprint arXiv*:2301.02086, 2023.

[71] R. O. Anisimov, N. V. Goloburdin, K. A. Kulagin, T. Y. Gladkikh and Y. D. Vorobiev, "Visual localization system algorithm for UAV," in *Proc. Int. Conf. on Information, Control, and Communication Technologies (ICCT)*, pp. 1-5, 2022.

[72] D. Martin, C. Fowlkes, D. Tal and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. on Computer Vision,* pp. 416-423, 2001.

[73] A. Abdelhamed, S. Lin and M. S. Brown, "A high-quality denoising dataset for smartphone cameras," in *Proc. the IEEE Conf. on Computer Vision and Pattern Recognition,* pp. 1692-1700, 2018.

[74] X. Hu, C. Fu, L. Zhu, J. Qin and P. Heng, "Direction-aware spatial context features for shadow detection and removal," *IEEE Trans. Pattern Anal. Mach.*

Intell., vol. 42, (11), pp. 2795-2808, 2019.

[75] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. on Image Processing*, 13 (4): 600–612, 2004.

[76] J. Engel, T. Schöps and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *European Conference on Computer Vision*, pp. 834-849, 2014.

[77] V. Balntas, L. Tang and K. Mikolajczyk, "Binary online learned descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 40, (3), pp. 555-567, 2017.

[78] J. Chan, J. A. Lee and K. Qian, "F-sort: An alternative for faster geometric verification," in Proc. *13th Asian Conf. on Computer Vision, Taipei, Taiwan, Nov. 20-24,* pp. 385-399, 2016.

[79] R. Cai, "Research progress in image denoising algorithms based on deep learning," *Journal of Physics*, vol. 1345: 042055, 2019.

[80] F. Banterle, M. Corsini, P. Cignoni, R. Scopigno, "A Low-Memory, Straightforward and Fast Bilateral Filter Through Subsampling in Spatial Domain," *Computer Graphics Forum*, 31 (1): 19–32, 2011

[81] A. Buades, "A non-local algorithm for image denoising," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'05)*, pp. 60–65, 2005

[82] K. Dabov, A. Foi, V. Katkovnik and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.,* vol. 16, (8), pp. 2080-2095, 2007.

[83] H. C. Burger, C. J. Schuler, S. Harmeling, "Image denoising with multi-layer perceptrons, part 1: comparison with existing algorithms and with bounds," *Computer Science*, 2012, 8-30.

[84] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, M. Yang and L. Shao, "Cycleisp: Real image restoration via improved data synthesis," in *Proc. the IEEE/CVF Conf. on Computer Vision and Pattern Recognition,* pp. 2696-2705, 2020.

[85] J. M. Prewitt, "Object enhancement and extraction," *Picture Processing and Psychopictorics,* vol. 10, (1), pp. 15-19, 1970.

[86] W. Chen, Z. Huang, C. Tsai, H. Yang, J. Ding and S. Kuo, "Learning multiple adverse weather removal via two-stage knowledge learning and multi-contrastive regularization: Toward a unified model," in *Proc. the IEEE/CVF Conf. on Computer Vision and Pattern Recognition,* 2022, pp. 17653-17662.

[87] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising, " *IEEE Trans. Image Process*., vol. 26, no. 7, pp. 3142-3155, Jul. 2017.

[88] K. Zhang, W. Zuo and L. Zhang, "FFDNet: Toward a fast and flexible solution for CNN-based image denoising," *IEEE Trans. Image Process.,* vol. 27, (9), pp.

4608-4622, 2018.

[89] S. Guo, Z. Yan, K. Zhang, W. Zuo, and L. Zhang, “Toward convolutional blind denoising of real photographs,” in *Proc. CVPR*, pp. 1712-1722, 2019

[90] S. Anwar and N. Barnes, “Real Image Denoising with Feature Attention”, in *Proc. ICCV*, IEEE, pp. 3155-3164, 2019.

[91] S. Gu, L. Zhang, W. Zuo and X. Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proc. the IEEE Conf. on Computer Vision and Pattern Recognition,* pp. 2862-2869, 2014.

[92] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man Cybern.,* vol. 9, (1), pp. 62-66, 1979.

[93] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, (6), pp. 679-698, 1986.

[94] Z. Yue, H. Yong, Q. Zhao, D. Meng and L. Zhang, "Variational denoising network: Toward blind noise modeling and removal," *Advances in Neural Information Processing Systems,* vol. 32, 2019.

[95] B. Triggs, P. F. McLauchlan, R. I. Hartley and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," in Vision Algorithms: Theory and Practice: *Int. Workshop on Vision Algorithms, Corfu, Greece, Sept. 21–22, 1999*, pp. 298-372.

[96] L. Chen, X. Chu, X. Zhang and J. Sun, "Simple baselines for image restoration," in *European Conference on Computer Vision,* pp. 17-33, 2022.

[97] K. He, J. Sun and X. Tang, "Single image haze removal using dark channel prior," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 33, (12), pp. 2341-2353, 2010.

[98] D. Nistér, O. Naroditsky and J. Bergen, "Visual odometry for ground vehicle applications," *Journal of Field Robotics*, vol. 23, (1), pp. 3-20, 2006.

[99] K. Tang, J. Yang, and J. Wang, “Investigating haze-relevant features in a learning framework for image dehazing,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 2995-3000, 2014.

[100] D. Berman, T. Treibitz and S. Avidan, “Non-local Image Dehazing”, in *Proc. IEEE Conf. Computer Vision and Pattern Recogn.* (CVPR), pp. 1674-1782, 2016.

[101] A. Howard, "Real-time stereo visual odometry for autonomous ground vehicles," in *Proc. IEEE/RSJ Int. Conf. on Intell. Robots and Systems*, pp. 3946-3952, 2008.

[102] W. Ren, J. Pan, H. Zhang et al, “Single Image Dehazing via Multi-scale Convolutional Neural Networks with Holistic Edges,” *Int. Journal of Computer Vision*, 2020, 128(1):240-259, 2020

[103] B. Li, J. Yan, W. Wu, Z. Zhu and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proc. the IEEE Conf. on Computer Vision and Pattern Recognition,* pp. 8971-8980, 2018.

[104] A. J. Davison, I. D. Reid, N. D. Molton and O. Stasse, "MonoSLAM: Real-

time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, (6), pp. 1052-1067, 2007.

[105] B. Li, X. Peng, Z. Wang, J. Xu and D. Feng, "Aod-net: All-in-one dehazing network," in *Proc. the IEEE Int. Conf. on Computer Vision,* pp. 4770-4778, 2017.

[106] M. Montemerlo, S. Thrun, D. Koller and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," *Aaai/Iaai*, vol. 593598, 2002.

[107] D. Chen, M. He, Q. Fan, J. Liao, L. Zhang, D. Hou, L. Yuan and G. Hua, "Gated context aggregation network for image dehazing and deraining," in Proc. *IEEE Winter Conf. on Applications of Computer Vision (WACV),* pp. 1375-1383, 2019.

[108] Q. Deng, Z. Huang, C.C. Tsai et al, “HardGAN: A Haze-Aware Representation Distillation GAN for Single Image Dehazing,” In *Proc. ECCV, LNCS 12351*, pp. 722-738, 2020.

[109] X. Liu, Y. Ma, Z. Shi and J. Chen, "Griddehazenet: Attention-based multi-scale network for image dehazing," in *Proc. the IEEE/CVF Int. Conf. on Computer Vision,* pp. 7314-7323, 2019.

[110] X. Qin, Z. Wang, Y. Bai, X. Xie and H. Jia, "FFA-net: Feature fusion attention network for single image dehazing," in *Proc. the AAAI Conf. on Artificial Intelligence,* pp. 11908-11915, 2020.

[111] Y. Song, Z. He, H. Qian and X. Du, "Vision transformers for single image dehazing," *IEEE Trans. Image Process.,* vol. 32, pp. 1927-1941, 2023.

[112] T. Ye, M. Jiang, Y. Zhang, L. Chen, E. Chen, P. Chen and Z. Lu, "Perceiving and modeling density is all you need for image dehazing," *arXiv Preprint arXiv:2111.09733,* 2021.

[113] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. on Computer Vision and Pattern Recog.,* pp. 886-893, 2005.

[114] P. Felzenszwalb, D. McAllester and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition,* p. 1-8, 2008.

[115] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu and A. C. Berg, "SSDd: Single shot multibox detector," in *Proc. 14th European Conf., Amsterdam, the Netherlands, October 11–14,* pp. 21-37, 2016.

[116] T. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal loss for dense object detection," in *Proc. the IEEE Int. Conf. on Computer Vision,* pp. 2980-2988, 2017.

[117] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proc. of the 18th SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems,* pp. 270-279, 2010.

[118] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards real-time

object detection with region proposal networks," *Advances in Neural Information Processing Systems,* vol. 28, 2015.

[119] Z. Li and F. Zhou, "FSSD: feature fusion single shot multibox detector," *arXiv Preprint arXiv:1712.00960,* 2017.

[120] S. M. Smith and J. M. Brady, “A New Approach to Low Level Image Processing Tech”, *Int. J. Computer Vision*, 23(1), pp. 45-78, 1997.

[121] P. R. Beaudet,"Rotationally invariant image operators,” in *Proc. the Int. Joint Conf. on Pattern Recognition*, pp.579-583, 1978.

[122] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. the IEEE Conf. on Computer Vision and Pattern Recognition,* pp. 7263-7271, 2017.

[123] H. Bay, T. Tuytelaars, L.V. Gool, “SURF: speeded up robust features,” in *Proc. European Conf. on Computer Vision*, Springer-Verlag, pp. 404-417, 2006.

[124] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv Preprint arXiv:1804.02767,* 2018.

[125] A. Bochkovskiy, C. Wang and H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv Preprint arXiv:2004.10934,* 2020.

[126] Maxim Ivanov, "The evolution of the YOLO neural networks family from v1 to v7," https://medium.com/deelvin-machine-learning/the-evolution-of-the-yolo-neural-networks-family-from-v1-to-v7-48dd98702a3d, Date of access: 18/11/2022

[127] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng and W. Nie, "YOLOv6: A single-stage object detection framework for industrial applications," *arXiv Preprint arXiv:2209.02976,* 2022.

[128] C. Wang, A. Bochkovskiy and H. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proc. the IEEE/CVF Conf. on Computer Vision and Pattern Recognition,* pp. 7464-7475, 2023.

[129] M. I. Lourakis and A. A. Argyros, "SBA: A software package for generic sparse bundle adjustment," *ACM Trans. on Mathematical Software (TOMS),* vol. 36, (1), pp. 1-30, 2009.

[130] G. Wang, J. Chen, M. Dai and E. Zheng, "WAMF-FPI: A Weight-Adaptive Multi-Feature Fusion Network for UAV Localization," *Remote Sensing*, vol. 15, (4), pp. 910, 2023.

[131] M. Son and K. Ko, "Learning-based essential matrix estimation for visual localization," *Journal of Computational Design and Engineering*, vol. 9, (3), pp. 1097-1106, 2022.

[132] J. Tarel and N. Hautiere, "Fast visibility restoration from a single color or gray level image," in *Proc. IEEE 12th Int. Conf. on Computer Vision,* pp. 2201-2208, 2009.

[133] L. Wen, D. Du, Z. Cai, Z. Lei, M. Chang, H. Qi, J. Lim, M. Yang and S. Lyu,

"UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking," *Comput. Vision Image Understanding,* vol. 193, pp. 102907, 2020.

[134] J. R. Terven and D. M. Cordova-Esparza, "A comprehensive review of YOLO: From Yolov1 and Beyond," *arXiv:2304.00501v4*, Aug. 2023

[135] C. A. Harris, "Combined corner and edge detector," In *Proc. Alvey Vision Conf.*, (3):147-151, 1988

[136] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. of Computer Vision*, 60(2), pp. 91-110, 2004

[137] E. Rublee, V. Rabaud, K. Konolige et al. "ORB: An efficient alternative to SIFT or SURF," in *Proc. IEEE Int. Conf. Computer Vision*, pp. 2564-2571, 2011.

[138] J. Liang, Y. Wang, Y. Chen, B. Yang and D. Liu, "A triangulation-based visual localization for field robots," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, (6), pp. 1083-1086, 2022.

[139] C. O. Ancuti, C. Ancuti, R. Timofte and C. De Vleeschouwer, "O-haze: A dehazing benchmark with real hazy and haze-free outdoor images," in *Proc. the IEEE Conf. on Computer Vision and Pattern Recog. Workshops,* pp. 754-762, 2018.

[140] M. Dai, J. Chen, Y. Lu, W. Hao and E. Zheng, "Finding point with image: an end-to-end benchmark for vision-based UAV localization," *arXiv Preprint arXiv*:2208.06561, 2022.

[141] N. Peterfreund, "System and method for visual localization", WO2022070184A1, 2022.

[142] B. Bescos, J. M. Fácil, J. Civera and J. Neira, "DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robotics and Automation Letters*, vol. 3, (4), pp. 4076-4083, 2018.

[143] M. Maimone, Y. Cheng and L. Matthies, "Two years of visual odometry on the mars exploration rovers," *Journal of Field Robotics*, vol. 24, (3), pp. 169-186, 2007.

[144] M. Dorigo, V. Maniezzo and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics*, Part B (Cybernetics), vol. 26, (1), pp. 29-41, 1996.

[145] A. Geiger, P. Lenz, C. Stiller and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The Int. J. of Robotics Research,* vol. 32, (11), pp. 1231-1237, 2013.

[146] J. Engel, V. Koltun and D. Cremers, "Direct sparse odometry," *IEEE Trans. on Pattern Analysis and Machine Intelligence*., vol. 40, (3), pp. 611-625, 2017.

[147] R. Mur-Artal, J. M. M. Montiel and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, (5), pp. 1147-1163, 2015.

[148] G. Silveira, E. Malis and P. Rives, "An efficient direct approach to visual SLAM," *IEEE Transactions on Robotics*, vol. 24, (5), pp. 969-979, 2008.

[149] C. Engels, H. Stewénius and D. Nistér, "Bundle adjustment rules," *Photogrammetric Computer Vision,* vol. 2, (32), 2006.

[150] R. Smith, M. Self and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous Robot Vehicle,* Springer, pp. 167-193, 1990.

[151] M. Montemerlo, S. Thrun, D. Koller and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proc. 18th Int. Joint. Conf. Artificial Intelligence*, pp. 1151-1156, 2003.

Appendix

List of the online codes used for comparison.

1) NLM: https://github.com/Linwei-Chen/NL-means.git, Access date: 18/05/2022
2) WNNM: https://github.com/csjunxu/WNNM_CVPR2014.git, Access date: 26/04/2022
3) VDN: https://github.com/zsyOAOA/VDNet.git, Access date: 11/06/2022
4) FFDNet: https://github.com/cszn/FFDNet.git, Access date: 22/07/2022
5) NAFNet: https://github.com/megvii-research/NAFNet.git, Access date: 29/06/2022
6) CycleISP: https://github.com/swz30/CycleISP.git, Access date: 05/08/2022
7) TSKL: https://github.com/fingerk28/Two-stage-Knowledge-For-Multiple-Adverse-Weather-Removal.git, Access date: 23/08/2022
8) DehazeFormer: https://github.com/IDKiro/DehazeFormer.git, Access date: 03/09/2022
9) FFANet: https://github.com/zhilin007/FFA-Net.git, Access date: 10/12/2022
10) GridDehazeNet: https://github.com/proteus1991/GridDehazeNet.git, Access date: 29/08/2022
11) GCANet: https://github.com/cddlyf/GCANet.git, Access date: 26/10/2022
12) PMNet: https://github.com/Owen718/ECCV22-Perceiving-and-Modeling-Density-for-Image-Dehazing.git, Access date: 17/06/2022