

DEEP LEARNING VS. TRADITIONAL COMPUTER VISION FOR UAV LANDING SITE DETECTION IN ADVERSE WEATHER

Rebecca Schmidt[1] & Joachim Rüter[1]

[1]German Aerospace Center (DLR), Institute of Flight Systems, 38108 Brunswick, Germany

Abstract

Autonomous Unmanned Aerial Vehicles are increasingly being considered for a wide range of tasks, including cargo transport and environmental monitoring. Many of these applications require automated landing in all weather conditions, making reliable landing site detection essential. Computer vision methods, both deep learning–based and traditional, have shown promising results for this task. Although deep learning is often assumed to outperform traditional techniques in detection problems, no direct comparison has yet been conducted specifically for landing site detection under adverse environmental conditions. To address this gap, this study implements and compares state-of-the-art detection algorithms from both categories. To evaluate their robustness in realistic operational scenarios, augmented test data incorporating harsh environmental factors such as adverse weather are used. The comparative analysis examines key algorithmic characteristics, including detection performance, computational resources, and hardware requirements. By assessing the strengths and limitations of each approach, this work provides deeper insight into their suitability for landing site detection under adverse weather conditions.

Keywords: VTOL UAV, Landing Site Detection, Computer Vision, Image Augmentation, Adverse Weather

1. Introduction

With the rapid advancement of Vertical Take-off and Landing (VTOL) Unmanned Aerial Vehicle (UAV) technology, their integration across a wide range of application domains has accelerated significantly. In particular, tasks such as search and rescue operations [1], environmental monitoring [2], and cargo transport [3] can benefit from the key advantages of UAVs, including their high operational flexibility, suitability for continuous deployment, and enhanced cost efficiency. In many of these applications, missions start and end at designated take-off and landing areas. Consequently, the ability to execute safe take-offs and landings in constrained environments, particularly under adverse environmental conditions, is a fundamental requirement for reliable UAV operation. At the same time, landing is one of the most critical and technically challenging phases of flight, as errors at this stage can result in severe damage to both the UAV and nearby infrastructure. Achieving automated landing requires real-time relative positioning between the UAV and the landing site. Although Global Navigation Satellite Systems (GNSS) are widely used for positioning, their utility in this context is limited. Under ideal conditions, GNSS typically provides positional accuracy on the order of 3 to 5 m [4], which is insufficient for precise landing procedures. Moreover, GNSS signals are vulnerable to natural disturbances, such as space weather events [5], as well as intentional interference, including jamming and spoofing attacks [6].

Onboard perception systems provide a promising alternative to satellite-based positioning by first performing the detection of the landing target and subsequently estimating the UAVs relative position. Camera-based sensor systems have become one of the most widely adopted technologies for landing applications [7, 8] and computer vision (CV) techniques, such as object detection and semantic



Figure 1 – Exemplary images from the real-world dataset, featuring the landing marker under adverse weather augmentations, including rain, fog, and varying illumination

segmentation, provide powerful tools for landing site detection, including traditional, i.e. non-deep learning (DL), and deep learning-based approaches. DL approaches have demonstrated significant promise, particularly in tasks such as semantic segmentation and object detection. The literature indicates that DL methods outperform traditional techniques in applications like pedestrian detection [9] and multiple segmentation tasks [10]. However, there are well-founded reasons why DL-based CV approaches do not render traditional methods obsolete. Traditional techniques offer greater algorithmic explainability and do not require large training datasets, making them highly effective for problems that can be addressed using expert systems [11]. Landing site detection can benefit from traditional methods, as landing site markers are often well defined and consist of geometric shapes with standardized dimensions, such as those specified in marker designs provided by regulatory bodies [12, 13].

Currently, comparative research between DL and traditional techniques for landing site detection remains limited, particularly under adverse weather conditions. Therefore, in this work, we compare state-of-the-art detection algorithms from both DL and traditional object detection approaches. To challenge these algorithms under target domain conditions, test data is used that simulate harsh operational domain factors, such as adverse weather and varying illumination by utilizing image augmentation methods, as illustrated in Fig. 1. The comparative analysis evaluates key algorithm characteristics, including detection performance, runtime performance, and computational resources. Through this work, we want to identify the strengths and weaknesses of each method and work towards the identification of a suitable approach for the landing site detection in both, good and adverse weather conditions.

The key contributions of this paper are as follows. First, we introduce a test dataset that simulates domain-relevant environmental conditions using adverse weather image augmentation for algorithm evaluation. Second, we propose a landing site detection algorithm for a defined landing marker design based on traditional CV techniques. Third, we conduct a comparative analysis between the traditional method and two state-of-the-art DL-based YOLO architectures, by evaluating performance, runtime, and computational requirements.

2. Related Work

The use of vision-based onboard perception systems on UAVs is receiving growing interest because of strong performance in UAV-relevant tasks such as environment perception [14, 15] and localisation [16], as well as continuing advances in enabling methods such as safety monitoring [17, 18, 19] of UAV vision systems. Vision-based systems are also among the most widely adopted sensor con-

figurations for UAV landing, driven by their widespread integration across platforms and the rapid advancement of CV. In recent years, numerous landing site detection methods have been developed, including traditional techniques and modern DL-based approaches.

Traditional, i.e., non–DL–based, approaches rely on classic feature-based detection methods. For example, [20, 21] employ CV techniques for landing marker detection, both using a marker consisting of a cross inside a circle. In [20], robustness is enhanced by detecting the outer circle and inner cross separately to avoid false positives, while Guo–Hall thinning [22] is used for robust line detection. In contrast, [21] focus on ellipse-fitting algorithms, emphasizing robustness against partial occlusions of the marker design. Both approaches demonstrate real-time capability, with processing times in the millisecond range. Such expert systems are often straightforward to deploy, as they do not require powerful or expensive hardware and can operate efficiently on small companion computers in real time. Expert-designed systems typically yield good results in well-defined and less complex environments, often achieving high detection rates under controlled conditions [11, 23].

DL–based systems leverage semantic information to achieve state-of-the-art performance in detection tasks. Object detection is one of the most common techniques applied to landing site detection. A primary challenge in DL networks is the trade-off between performance and runtime efficiency. While larger models generally offer higher performance, they require substantial computational resources. Object detectors such as YOLO are designed to balance this trade-off. For instance, YOLO has been successfully applied to marker-based ship-deck landing site detection [24, 25] and vertipad marker detection [26]. Additionally, [27] proposes a CNN-based ellipse detector that extracts shape information directly from markers, not using standard bounding box regression. While these models typically achieve high performance across diverse conditions, they require training datasets that accurately reflect the operational domain environments.

Most methods for landing site detection are evaluated primarily on small datasets due to a lack of large-scale, real-world data containing adverse weather conditions, making it difficult to reliably assess their performance in harsh environments. In contrast, more extensive investigations into the effects of adverse weather have been conducted for other UAV-centic tasks such as vehicle [28, 29] and UAV detection [30]. These studies frequently utilize image augmentation techniques to synthetically apply controlled weather conditions to existing real-world data. For example, [30] investigate the performance of DL models, including YOLO, Faster R-CNN, and RetinaNet, for UAV detection in aerial imagery. By augmenting clear data with rain and distortions, they demonstrate significant performance degradation under adverse conditions. Similarly, [28] address maritime ship detection under adverse weather, e.g. fog, low light, flares, introducing a weather degradation ratio based on distortion and occlusion levels. They test YOLO and various DETR models against this ratio, observing consistent performance declines as degradation increases. Conversely, research on snowy conditions remains limited. [29] evaluate vehicle detection using real-world snowy data, demonstrating significant performance drops when deploying state-of-the-art networks, such as YOLOv5s, YOLOv8s and Faster R-CNN, on low-contrast, snow-covered images.

However, these applications typically involve detecting small objects from high altitudes, demanding distinct requirements and configurations compared to landing site detection. This difference makes it difficult to generalize the results to the landing site detection task. Moreover, there is a notable lack of comparative studies evaluating traditional and DL approaches on identical datasets under adverse weather conditions. To address these gaps, this work contributes to the UAV domain by introducing a traditional landing site detection approach and systematically benchmarking it against state-of-the-art DL detectors.

3. Landing Site Detection

Vision-based landing site detection plays a crucial role in enabling automated landings of VTOL UAVs. However, it remains a highly challenging task due to the characteristics of the landing area, e.g., the limited available space and the high precision required, combined with potentially adverse environmental conditions. To aid in vertical landings, visual markers are often deployed to facilitate the detection of the target. In addition to various marker designs found in the literature [31], regulatory bodies such as the European Union Aviation Safety Agency (EASA) [12] and the Federal Aviation Administration (FAA) [13] have proposed landing site designs for VTOL operations. An example of

a design, as recommended by EASA, is shown in Fig. 1, and is used in this study for landing site detection. The goal of landing site detection is to accurately recognize the landing site marker in images under varying conditions. To evaluate and compare DL and traditional detection approaches, we implemented three algorithms for detecting the landing marker design: one based on traditional methods and two YOLO-based models. YOLO nano versions are employed since real-time operation on the UAV hardware is desired.

3.1 Traditional Approach

The traditional detection approach relies on a ellipse detection method. We apply the detection approach to the landing marker design proposed by EASA [12], shown in Fig 1. The marker consists of multiple colored circular elements, which can be detected as elliptical features from varying distances, making the ellipse-based approach suitable for the considered pattern. The algorithm is implemented using the python opencv library [32]. The detection approach comprises three main steps, as summarized in Algorithm 1. First, the input frame is preprocessed to enhance feature visibility and reduce noise. Next, candidate ellipses are extracted from the preprocessed frame. Finally, these candidates are evaluated and ranked based on a fitting criterion, with the highest-ranking ellipse selected as the final estimate. The detection result is represented by a bounding box fitted around the chosen ellipse. Further details of the algorithm are given in the following sections.

Algorithm 1 Traditional Detection Approach

```
function DetectLandingMarker(frame, threshold)
    ▷ Preprocess input frame
    frame_pp ← PreProcess(frame, threshold)

    ▷ Detect ellipses in pre processed frame
    ellipses ← DetectEllipses(frame_pp)

    ▷ Rank ellipses and extract bounding box
    bestFittingEllipse ← RankEllipses(ellipses)
    boundingBox ← BBXFromEllipse(bestFittingEllipse)
end function
```

(1) The first step, i.e. preprocessing, aims to reduce noise in the frame and prepare it for subsequent contour-based ellipse detection. The grayscale frame is converted to a binary frame using adaptive thresholding. Depending on the image noise level, a morphological erosion is applied to the binary frame to refine the contours for ellipse detection. Next, small components below a predefined size threshold are removed by identifying all connected components and eliminating those smaller than the threshold to further reduce noise.

(2) In the second step, the ellipse detection is performed using a method similar to [21], as detailed in Algorithm 2. First, contours are extracted from the preprocessed input frame and subsequently evaluated against several criteria based on reasonable real-world assumptions. For each contour, an initial filtering step is applied based on the contour area. Contours with an area below a predefined threshold are rejected, as they are likely caused by noise. If the contour passes this check, an ellipse fitting procedure is performed to determine the ellipse that best fits the set of 2D contour points. Next, several ellipse characteristics are evaluated. Ellipses with a large vertical angle, i.e. a ellipse with oval shape whose major axis runs vertically, are discarded, since such appearances are unlikely to occur from a UAV perspective when observing a circular pattern. Furthermore, the eccentricity of each detected ellipse is evaluated to quantify how much the shape deviates from a perfect circle. Ellipses with very high eccentricity values are considered more likely to originate from noise and are therefore rejected. In addition, two overlap-based filtering checks are applied to evaluate how well the fitted ellipse corresponds to the underlying contour. First, the overlap between the ellipse area and the corresponding contour area is determined, and the ratio with respect to the contour area is calculated. The ellipse is rejected if this ratio falls below a predefined threshold. Second, the overlap between the ellipse perimeter and the contour length is evaluated. The resulting overlap length is compared to the contour length to calculate another ratio. Ellipses with ratios below the specified threshold are discarded. These filtering steps remove ellipses that do not sufficiently match the underlying contour and are therefore unlikely to correspond to the desired detection object. The remaining set of valid ellipses is then forwarded to the ranking stage.

(3) In the final step, the detected ellipses are ranked, and the best-fitting ellipse is used to extract the bounding box as the detection result. For ranking, a score is calculated for each ellipse. The ellipse score is composed of three components: ellipse circularity and two overlap ratios with the contour.

Ellipse circularity measures how closely the ellipse resembles a perfect circle, quantified by the ratio of the minor to the major axis. The overlap ratio of the ellipse area with the contour area is low if the ellipse poorly fits the contour, for example, if the ellipse is significantly smaller than the contour. Similarly, the overlap ratio of the ellipse perimeter with the contour length is low if the ellipse perimeter deviates substantially from the contour length, indicating a poor fit. The raw ellipse score is further boosted by clustering and color information. All detected ellipses are clustered based on their center points. Since the ellipse detection method relies on contours, it tends to detect multiple circular elements when applied to the used landing marker design. Typically, the most outer contour, the yellow circular region, and the blue circular region are all detected as ellipses, often with closely aligned centers. Therefore, ellipses that belong to clusters with multiple close center points are considered more likely to correspond to the desired marker design and are assigned higher scores. Finally, the score of an ellipse is increased if the RGB frame area inside the ellipse contains a sufficient amount of blue and yellow pixels, further emphasizing ellipses corresponding to the markers expected color pattern.

Algorithm 2 Ellipse Detection Approach

function DETECTELLIPSES($frame$, $threshold$)
    ▷ Initialize empty set of detections
    $Ellipses \leftarrow \emptyset$

    ▷ Extract contours
    $contours \leftarrow$ FINDCONTOURS($frame$)

    for all $c \in contours$ **do**
        ▷ Ignore if area below threshold
        if $c.area < minContourArea$ **then Continue**

        ▷ Fit ellipse to contour
        $e \leftarrow$ FITELLIPSE(c)

        ▷ Ignore ellipse if not in accepted angles range
        if $e.angle < minAngle$ or $e.angle > maxAngle$ **then Continue**

        ▷ Ignore ellipse with too high eccentricity value
        if $e.eccentricity > maxEccentricity$ **then Continue**

        ▷ Ignore ellipses with insufficient contour area overlap
        $overlap \leftarrow c.area \cap e.area$
        if $|overlap|/c.area < minAreaOverlap$ **then Continue**

        ▷ Ignore ellipse with insufficient contour length overlap
        $overlap \leftarrow c.length \cap e.perimeter$
        if $|overlap|/c.length < minLengthOverlap$ **then Continue**

        ▷ Add ellipse to set
        $Ellipses$.append(e)
    end for

    return $Ellipses$
end function

3.2 YOLOv8n-based Approach

As the first DL-based model, we deploy a YOLOv8n object detection model using Ultralytics [33], initializing it with given pre-trained weights. Given the computational constraints of UAV-mounted companion computers, the nano-scale architecture is selected. The model is fine-tuned for a single-class task targeting landing markers. Training is done over 300 epochs and the model with best validation loss is used for evaluation. We use a batch size of 16 and an input resolution of 640x640 pixels. To enhance generalization, we apply Ultralytics YOLOv8n default random image augmentations throughout the training process using color space augmentations and geometric augmentations. The dataset consists of 1073 images, with 858 used for training and 215 for validation. It includes images from seven recordings, five from aerial flights and two from experimental ground-based captures. The images cover a range of heights, from very close to the target pattern up to 30 m, and are collected under clear weather conditions. Data acquisition is performed at different times of day, including early morning to late afternoon, and across seasons, mostly summer and few winter. The EASA landing marker design is used in all recordings. Both the training and validation sets contain a representative subset of images from each recording.

3.3 YOLO26n-based Approach

In addition to YOLOv8n, we train the more recent YOLO26n using Ultralytics [33], which is engineered for edge and low-power devices by reducing complexity to ensure a lighter and faster profile. To ensure comparability, we utilize the same training parameters as YOLOv8n. The model is initialized with given pre-trained weights and fine-tuned for single-class detection, specifically targeting the landing marker. Training is again done over 300 epochs, and the model with best validation loss is used. A batch size of 16, and an input resolution of 640x640 pixels is used. Ultralytics default image augmentations are applied randomly throughout the training process using color space augmentations and geometric augmentations. For training and validation we use the same dataset as introduced for YOLOv8n.

Table 1 – Parameters for adverse weather image augmentation using imgaug library

	Parameters	Value Ranges
Rain (Drizzle | Medium | Heavy)	Density	[0.005, 0.01] | [0.02, 0.04] | [0.1, 0.3]
	Drop Size	[0.3,0.4] | [0.6, 0.8] | [0.7, 0.85]
	Speed	[0.015, 0.02] | [0.05, 0.1] | [0.15, 0.2]
	Angle	[-30, 30] | [-30, 30] | [-30, 30]
Fog (Haze | Medium | Heavy)	Intensity Color	[190, 255] | [190, 255] | [190, 255]
	Alpha Min.	0.2 | 0.3 | 0.45
	Alpha Multiplier	[0.3, 0.6] | [0.3, 0.9] | [0.4, 0.75]
	Sparsity	1 | 1 | 1
	Density Multiplier	[0.25, 0.5] | [0.25, 0.75] | [0.5, 1]
Illumination (High | Low)	Multiplier	[1.3, 1.5] | [0.5, 0.7]
	Linear Contrast	[1.1, 1.3] | [0.85, 0.95]

4. Adverse Weather Augmentation

To generate evaluation datasets, we apply image augmentations that simulate adverse weather conditions. The evaluation set is augmented with varying levels of illumination changes, rain, and fog. We utilize the python imgaug library [34], which provides interfaces for applying these augmentations across multiple intensity levels. The parameters used to control each weather condition are summarized in Table 1. During data generation, these parameters are sampled randomly within their predefined ranges for each intensity level.

Rain augmentations are applied at three distinct intensities: drizzle, medium, and heavy. This is achieved by applying an imgaug RainLayer with four key parameters: density, controlling the amount of rain in the layer, drop size, i.e., the size of individual drops, speed controlling the perceived falling speed that determines rain streak length, and angle, i.e. the direction of the rain within the image.

Fog augmentations follow a similar pattern across three intensity levels: haze, medium, and heavy. Similar to rain, this utilizes an imgaug FogLayer with specific parameters. Intensity color controls the fogs tone, ranging from darker to brighter. Alpha min and Alpha multiplier work together to control the blending of the fog layer with the background, where the min value sets the base opacity and the multiplier scales it for fine-tuning. Additionally, sparsity adjusts the coarseness of the cloud pattern, while the density multiplier defines the fog concentration in visible areas.

Furthermore, two illumination augmentations, bright and dark, are applied to the evaluation dataset. These are controlled by two parameters: multiplier, which raises or lowers global illumination, and linear contrast, which adjusts the image contrast to simulate varying visibility conditions.

5. Evaluation of Detection Approaches

We evaluate the traditional, YOLOv8n, and YOLO26n detection approaches on both the original dataset and augmented versions featuring rain, fog, and illumination conditions. By employing varying intensities of weather augmentations, we assess the robustness of each approach. Furthermore, we analyze detection performance across different object sizes.

5.1 Evaluation Datasets

An evaluation dataset is generated comprising 691 real-world images from four recordings distinct from the training data. The dataset consists of two flight-based recordings and two ground-based experimental captures. Images are recorded under clear weather conditions, covering a range of target heights from close-up to 50 m. Data collection is performed between early morning and late afternoon across multiple seasons, primarily spring and some winter. The EASA landing marker is treated as a single class throughout all recordings and each image contains a maximum of one object instance. The dataset distribution is as follows: 110 images contain no objects, 292 contain large objects, 155 contain medium objects, and 134 contain small objects. Object sizes are categorized by pixel area: large $[96^2, \infty]$, medium $[32^2, 96^2]$, and small $[0, 32^2]$. All images have a resolution of 640x640 pixels. In the following this baseline dataset will be referred to as *Clear*.

Image augmentations are applied to the baseline Clear dataset as described in Section 4.. This process generates three main augmented datasets *Rain*, *Fog*, and *Illumination*, comprising eight

Table 2 – Comparison of detection performance in mAP, P and R (at $IoU = 0.50$). Rain results are averaged over drizzle, medium, and heavy; Fog results are averaged over haze, medium, and heavy; Illumination results are averaged over high, clear, and low

		Clear			Rain			Fog			Illumination		
		mAP	P	R	mAP	P	R	mAP	P	R	mAP	P	R
Traditional	all	0.526	0.680	0.742	0.364	0.434	0.503	0.256	0.309	0.403	0.419	0.524	0.604
	large	0.670	0.787	0.818	0.635	0.742	0.793	0.462	0.552	0.644	0.593	0.705	0.749
	medium	0.458	0.648	0.735	0.233	0.279	0.359	0.251	0.191	0.284	0.359	0.442	0.572
	small	0.349	0.520	0.582	0.006	0.011	0.03	0.007	0.01	0.02	0.241	0.282	0.326
YOLOv8n	all	**0.945**	**0.974**	**0.985**	**0.938**	**0.965**	**0.975**	0.802	0.841	**0.912**	**0.894**	**0.933**	**0.948**
	large	0.963	0.981	0.993	0.95	0.967	0.978	0.751	0.788	0.878	0.894	0.915	0.929
	medium	0.95	0.96	0.968	0.952	0.962	0.97	0.922	0.933	0.942	0.929	0.949	0.961
	small	0.895	0.977	0.985	0.894	0.97	0.975	0.874	0.943	0.95	0.85	0.959	0.973
YOLO26n	all	0.935	0.96	0.979	0.926	0.956	0.972	**0.828**	**0.876**	0.905	0.884	0.921	0.944
	large	0.954	0.97	0.986	0.942	0.962	0.978	0.776	0.824	0.865	0.891	0.914	0.933
	medium	0.933	0.95	0.968	0.935	0.952	0.968	0.907	0.927	0.942	0.916	0.940	0.961
	small	0.901	0.965	0.978	0.882	0.954	0.963	0.864	0.936	0.948	0.842	0.930	0.948

sub-datasets in total. The Rain dataset includes *Rain Drizzle*, *Rain Medium*, and *Rain Heavy*, while the Fog dataset consists of *Fog Haze*, *Fog Medium*, and *Fog Heavy*. The Illumination dataset comprises *Illumination High*, *Illumination Low* and the original baseline Clear images. Each of these sub-datasets contains images generated by applying the corresponding augmentation parameters from Table 1 to the full baseline Clear dataset.

5.2 Performance Metrics

We measure detection performance using Precision (P), Recall (R), and mean Average Precision (mAP) using the COCO protocol [35], based on the Intersection over Union (IoU) between predicted B_p and ground-truth B_{gt} bounding boxes:

$$IoU = \frac{\text{area}(B_p \cap B_{gt})}{\text{area}(B_p \cup B_{gt})}. \tag{1}$$

Precision measures prediction accuracy as the ratio of correctly detected positives to the total number of predicted detections, whereas Recall assesses the models ability to detect all actual positive instances as the proportion of ground truth objects correctly identified. These metrics are defined as:

$$P = \frac{TP}{TP+FP}, \tag{2}$$

$$R = \frac{TP}{TP+FN} \tag{3}$$

where TP, FP, and FN represent true positives, false positives, and false negatives, respectively. Finally, mAP is computed as the mean of the Average Precision (AP) values, where each AP corresponds to the area under the precision–recall curve, evaluated across ten IoU thresholds ranging from 0.50 to 0.95. Since the model detects a single class, the calculation simplifies to:

$$mAP = \frac{1}{10} \sum\nolimits_{t \in \{0.5, 0.55, \ldots, 0.95\}} (AP_t). \tag{4}$$

5.3 Detection Performance

We first evaluate all detection approaches on the Clear, Rain, Fog, and Illumination datasets. Performance is measured across all object sizes, as well as individually for large, medium, and small categories. Subsequently, we investigate the impact of weather intensity by evaluating detection performance under different levels of degradation. For evaluation, we use mAP and report P and R at an IoU threshold of 0.50.

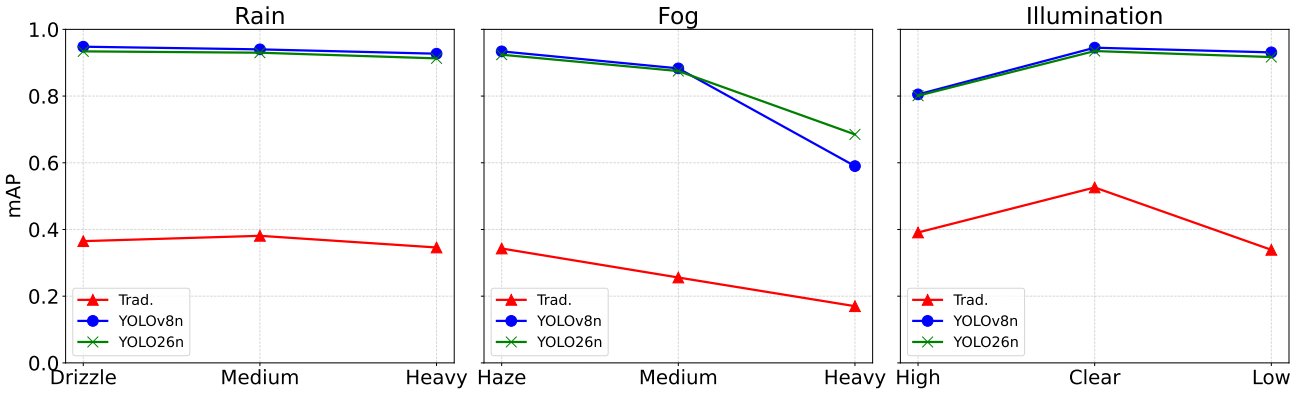


Figure 2 – Comparison of detection performance in mAP across degradation levels for rain, fog, and illumination

The results of the first evaluation step are presented in Table 2, with the best-performing models across all object sizes on the full dataset highlighted. Under clear conditions, YOLOv8n achieves the best overall performance, particularly on large and medium objects, while YOLO26n leads on small objects. Both DL-based approaches maintain high mAP across all sizes, whereas the traditional approach performs best on large objects. In rainy conditions, YOLOv8n maintains the best performance across all object sizes, despite a slight decrease in mAP compared to clear-weather conditions. YOLO26n follows closely behind, also exhibiting a drop in mAP under rainy conditions, while the traditional approach shows significant degradation, particularly for medium and small objects. Under foggy conditions, YOLO26n performs best overall and on large objects, with YOLOv8n showing the strongest results on medium and small sizes. Both DL approaches remain comparable, while the traditional approach suffers significant performance drops on smaller objects. Finally, under varying illumination, YOLOv8n leads across all object sizes, with YOLO26n showing comparable performance.

The detection performance of all approaches across varying levels of weather degradation is illustrated in Fig. 2. In different rain conditions, YOLO variants demonstrate comparable performance, with all approaches exhibiting only marginal degradation in drizzle and medium-to-heavy rain. Under medium-to-heavy fog conditions, all approaches experience significant performance drops as fog density increases. While the tested YOLO models perform similarly under haze and medium fog, their results diverge in heavy fog, with YOLO26n achieving the best performance. On illumination variations, both YOLO models perform consistently, generally yielding better results in low-light conditions compared to bright environments. In contrast, the traditional approach performs more accurately under high illumination conditions than in low illumination.

Overall, across all tested approaches, fog conditions tend to present the most significant challenge, causing the highest performance degradation among the adverse weather scenarios evaluated, while rain only showed limited influence.

5.4 Runtime Performance and Computational Requirements

We evaluate the runtime performance of all detection approaches on a UAV-mounted companion computer, using the NVIDIA® Jetson AGX Orin© module running in a 30 W power mode. Runtime is measured as the end-to-end latency from frame input to the generation of bounding box outputs. As shown in Table 3, the traditional approach achieves a mean runtime of 18.57 ms, outperforming both YOLOv8n and YOLO26n, which achieve mean runtimes of 25.59 ms and 23.68 ms, respectively.

Beyond runtime, we compare the hardware and data requirements of each approach. The traditional approach can operate on low-power edge devices without Graphics Processing Unit (GPU) acceleration, while DL models require dedicated GPU hardware to achieve real-time performance. Furthermore, DL-based approaches depend on large and diverse training datasets that closely reflect the target domain, as well as comprehensive ground-truth annotations. In contrast, traditional approaches rely on significantly less data, instead leveraging domain knowledge to develop rule-based or expert systems.

Table 3 – Comparison of runtime performance (ms) and resource requirements of detection approaches on NVIDIA® Jetson AGX Orin©

	Runtime	GPU Requirement	Data Requirement
Traditional	18.57	✗	low
YOLOv8n	25.59	✓	high
YOLO26n	23.68	✓	high

5.5 Discussion

We evaluate three object detection algorithms on adverse weather images comprising rain, fog, and varying illumination conditions. Weather degradations are introduced using image augmentation techniques. While such methods offer notable advantages, they also present distinct limitations. On the one hand, they enable easy access to test data and the controlled generation of datasets, while the availability of paired clean and augmented images facilitates direct performance comparison. On the other hand, augmented data often falls short in replicating real-world complexity. Natural weather phenomena interact to produce complex environmental effects that simple noise-layer augmentation cannot fully capture. Furthermore, widely available augmentation libraries typically do not account for aerial imaging perspectives, where cameras mounted on UAVs are frequently oriented downward rather than forward. Nevertheless, given that prior studies [36, 30] have demonstrated the utility of adverse weather augmented images in training DL models to enhance performance in real-world adverse conditions, it can be assumed that the images reflect the reality up to a certain level and the use of augmented images can still reveal meaningful initial trends that warrant further investigation.

The evaluation results show that all detection approaches exhibited the most significant performance degradation under foggy conditions compared to rain and illumination variations. We hypothesize this stems from the higher degree of occlusion caused by fog compared to rain. Furthermore, fog introduces bright regions within the image that closely resemble the visual signature of the landing marker, potentially leading to false positive detections.

While the proposed traditional detection approach yields moderate performance on larger objects, its strong reliance on object scale leads to significant performance degradation when handling smaller targets. However, a key advantage of the traditional method lies in its transparency. Unlike DL-based approaches, which often operate as black boxes, traditional methods are fully interpretable, offering distinct advantages for aviation applications. Furthermore, these approaches demonstrate notable hardware efficiency. By eliminating the need for GPU acceleration, they remain a viable option for resource-constrained onboard UAV systems.

In contrast to the traditional approach, both YOLO models achieved high performance values across most of the tested conditions, with only minor performance variations between the two architectures. These results stand in contrast to related works on UAV detection [30] and vehicle detection [28, 36] under similar adverse weather conditions, where YOLO variants typically demonstrated lower robustness. We assume that, for marker-based landing site detection, our selected training setup, consisting of standardized markers captured across multiple flight recordings, diverse environments, and varying altitudes and viewing angles, enhances the models ability to generalize effectively to unseen test data, resulting in high performance values. A significant drawback of DL-based approaches is their limited explainability, which complicates their adoption in aviation applications. Furthermore, YOLO-based approaches rely on GPU hardware to achieve real-time performance. This highlights a trade-off between detection performance and the computational resources required for deployment. In addition to computational resources, large amounts of training data are required for the use of DL-based approaches.

However, when interpreting the results of the detection approaches, it is important to note that this work compares only two of many available DL models, both based on similar architectures that have been progressively refined over the years. In contrast, the proposed traditional approach was specifically developed for this work and still has considerable potential for further development and performance improvement.

6. Conclusion and Future Work

In this work, we evaluate three landing site detection algorithms consisting of a proposed traditional, i.e. non-DL, approach, and two DL-based approaches utilizing YOLOv8n and YOLO26n architectures. We introduce a dataset comprising imagery of a defined landing marker captured at varying distances, angles, and across diverse environmental backgrounds. We apply adverse image augmentation to the test data to compare the detection algorithms across different degradation levels of rain, fog, and illumination, as well as varying object scales.

Our results indicate that the DL models achieve high performance across varying conditions and object sizes. In contrast, the traditional approach yields moderate performance, showing a performance drop when processing smaller objects. Notably, all detection algorithms experience their most substantial performance degradation under foggy conditions. However, it is important to note that this work compares established YOLO-based approaches with the traditional detection method proposed by us, which still offers potential for further development and performance enhancement. Traditional approaches are fully transparent and interpretable, which is a significant advantage for aviation applications. In contrast, DL approaches are generally considered black-box models. Furthermore, we evaluate the computational requirements of the approaches. While the traditional detection approach does not rely on GPU hardware or extensive training datasets, the DL-based approaches require powerful GPU hardware for real-time execution during flight and depend on large amounts of training data.

Future work might focus on enhancing the traditional approach by developing alternative variants and comparing them with additional DL-based model architectures, aiming toward a more comprehensive evaluation. Furthermore, the realism of adverse weather image augmentations should be addressed, as these techniques, despite their benefits, present significant limitations. Photorealistic simulation environments, such as Unreal Engine, show potential for easily generating adverse weather datasets that include automatic ground truth labels. Furthermore, such simulation environments are well-suited for generating more diverse datasets by simulating varied background conditions and employing physics-based weather simulation.

7. Contact Author Email Address

The corresponding authors email address is rebecca.schmidt@dlr.de.

8. Copyright Statement

The authors confirm that they, and/or their company or organization, hold copyright on all of the original material included in this paper. The authors also confirm that they have obtained permission, from the copyright holder of any third party material included in this paper, to publish it as part of their paper. The authors confirm that they give permission, or have obtained permission from the copyright holder of this paper, for the publication and distribution of this paper as part of the ICAS proceedings or as individual off-prints from the proceedings.

References

[1] J. Li, G. Zhang, C. Jiang, and W. Zhang, “A survey of maritime unmanned search system: Theory, applications and future directions,” *Ocean Engineering*, vol. 285, p. 115359, 2023.

[2] S. Yuan, Y. Li, F. Bao, H. Xu, Y. Yang, Q. Yan, S. Zhong, H. Yin, J. Xu, Z. Huang, *et al.*, “Marine environmental monitoring with unmanned vehicle platforms: Present applications and future prospects,” *Science of The Total Environment*, vol. 858, p. 159741, 2023.

[3] J. Wang, K. Zhou, W. Xing, H. Li, and Z. Yang, “Applications, evolutions, and challenges of drones in maritime transport,” *Journal of Marine Science and Engineering*, vol. 11, no. 11, p. 2056, 2023.

[4] W. Magiera, I. Vārna, I. Mitrofanovs, G. Silabrieds, A. Krawczyk, B. Skorupa, M. Apollo, and K. Maciuk, “Accuracy of code gnss receivers under various conditions,” *Remote Sensing*, vol. 14, no. 11, p. 2615, 2022.

[5] European Union Aviation Safety Agency, “Effects of space weather on aviation,” tech. rep., 2021. accessed May 2025.

[6] European Union Aviation Safety Agency, “Global navigation satellite system outage and alterations leading to navigation / surveillance degradation,” tech. rep., 2023. accessed May 2025.

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

[8] L. G. L. de Paula, V. S. Dwivedi, H.-S. Shin, and A. Tsourdos, "A review on vtol autonomous landing strategies on naval dynamic surfaces," *CEAS EuroGNC 2024*, 2022.

[9] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[10] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, "A review on deep learning techniques applied to semantic segmentation," *arXiv preprint arXiv:1704.06857*, 2017.

[11] N. O'Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh, "Deep learning vs. traditional computer vision," in *Science and information conference*, pp. 128–144, Springer, 2019.

[12] European Union Aviation Safety Agency, "Prototype technical specifications for the design of vfr vertiports for operation with manned vtol-capable aircraft certified in the enhanced category (pts-vpt-dsn)," tech. rep., 2022. accessed May 2025.

[13] Federal Aviation Administration, "Engineering brief no. 105, vertiport design," tech. rep., 2022. accessed May 2025.

[14] J. Leng, Y. Ye, M. Mo, C. Gao, J. Gan, B. Xiao, and X. Gao, "Recent advances for aerial object detection: a survey," *ACM Computing Surveys*, vol. 56, no. 12, pp. 1–36, 2024.

[15] C. Hinniger and J. Rüter, "Synthetic training data for semantic segmentation of the environment from uav perspective," *Aerospace*, vol. 10, no. 7, 2023.

[16] R. Schmidt, J. Rüter, S. Krause, and S. Schubert, "Vision-based self-localization for uavs using semantic features and openstreetmap," in *2025 IEEE Aerospace Conference*, 2025.

[17] N. Hartmann, F. Jünger, and J. Rüter, "Ensuring safety of deep learning components using improved image-level property selection for monitoring," in *AIAA SCITECH 2025 Forum*, 2025.

[18] R. Schmidt, J. Rüter, S. Schirmer, S. Krause, and J. C. Dauer, "A concept for increasing trustworthiness in deep learning perception for uas using map data," in *AIAA SCITECH 2025 Forum*, 2025.

[19] S. Narteni, M. Mongelli, J. Rüter, C. Torens, and U. Durak, "Transparent assessment of automated human detection in aerial images via explainable ai," in *AIAA SCITECH 2025 Forum*, 2025.

[20] T. Baca, P. Stepan, V. Spurny, D. Hert, R. Penicka, M. Saska, J. Thomas, G. Loianno, and V. Kumar, "Autonomous landing on a moving vehicle with an unmanned aerial vehicle," *Journal of Field Robotics*, vol. 36, no. 5, pp. 874–891, 2019.

[21] A. Keipour, G. A. Pereira, and S. Scherer, "Real-time ellipse detection for robotics applications," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7009–7016, 2021.

[22] Z. Guo and R. W. Hall, "Parallel thinning with two-subiteration algorithms," *Communications of the ACM*, vol. 32, no. 3, pp. 359–373, 1989.

[23] S. O. Semerikov, P. P. Nechypurenko, T. A. Vakaliuk, I. S. Mintii, and A. O. Kolhatin, "Vision-based autonomous uav landing: A comprehensive review of technologies, techniques, and applications," *Journal of Intelligent & Robotic Systems*, vol. 111, no. 4, pp. 1–39, 2025.

[24] T. Do Trong, Q. T. Hai, M. V. Van, B. N. Thai, T. N. Chi, and T. N. Quang, "Autonomous detection and approach tracking of moving ship on the sea by vtol uav based on deep learning technique through simulated real-time on-air image acquisitions," in *2021 8th NAFOSTED Conference on Information and Computer Science (NICS)*, pp. 374–380, IEEE, 2021.

[25] B. Lee, V. Saj, D. Kalathil, and M. Benedict, "Intelligent vision-based autonomous ship landing of vtol uavs," *Journal of the American Helicopter Society*, vol. 68, no. 2, pp. 113–126, 2023.

[26] E. Miccio, P. Veneruso, R. Opromolla, G. Fasano, G. Gentile, and C. Tiana, "Vision-aided navigation for uam approach to vertiports with multiple landing pads," in *2024 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1184–1191, IEEE, 2024.

[27] R. Jin, H. M. Owais, D. Lin, T. Song, and Y. Yuan, "Ellipse proposal and convolutional neural network discriminant for autonomous landing marker detection," *Journal of Field Robotics*, vol. 36, no. 1, pp. 6–16, 2019.

[28] B. Liu, J. Jin, Y. Zhang, and C. Sun, "Wrrt-detr: weather-robust rt-detr for drone-view object detection in adverse weather," *Drones*, vol. 9, no. 5, p. 369, 2025.

[29] H. Mokayed, A. Nayebiastaneh, K. De, S. Sozos, O. Hagner, and B. Backe, "Nordic vehicle dataset (nvd): Performance of vehicle detectors using newly captured nvd from uav in different snowy weather conditions.," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5314–5322, 2023.

[30] A. Munir, A. J. Siddiqui, S. Anwar, A. El-Maleh, A. H. Khan, and A. Rehman, "Impact of adverse weather

and image distortions on vision-based uav detection: A performance evaluation of deep learning models," *Drones*, vol. 8, no. 11, p. 638, 2024.

[31] L. Xin, Z. Tang, W. Gai, and H. Liu, "Vision-based autonomous landing for the uav: A review," *Aerospace*, vol. 9, no. 11, p. 634, 2022.

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

[33] G. Jocher, J. Qiu, and A. Chaurasia, "Ultralytics YOLO." https://github.com/ultralytics/ultralytics, 2023.

[34] A. B. Jung, K. Wada, J. Crall, S. Tanaka, J. Graving, C. Reinders, S. Yadav, J. Banerjee, G. Vecsei, A. Kraft, Z. Rui, J. Borovec, C. Vallentin, S. Zhydenko, K. Pfeiffer, B. Cook, I. Fernández, F.-M. De Rainville, C.-H. Weng, A. Ayala-Acevedo, R. Meudec, M. Laporte, *et al.*, "imgaug." https://github.com/aleju/imgaug, 2020.

[35] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*, pp. 740–755, Springer, 2014.

[36] H. Gupta, O. Kotlyar, H. Andreasson, and A. J. Lilienthal, "Robust object detection in challenging weather conditions," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 7523–7532, 2024.