An approach to matching satellite and UAV images for visual place recognition using color normalization and YOLO

Volodymyr Vozniak[1],*, Yuriy Ushenko[2] and Orken Mamyrbayev[3]

[1]Khmelnytskyi National University, 11, Institutes str., Khmelnytskyi, 29016, Ukraine
[2]Yuriy Fedkovych Chernivtsi National University, Chernivtsi, 58012, Ukraine
[3]Institute of Information and Computational Technologies, 125, Pushkin str., Almaty, 050010, Republic of Kazakhstan

Abstract

Accurate localization of Unmanned Aerial Vehicles (UAVs) in GPS-denied urban environments is a critical challenge for autonomous navigation and disaster response. The primary difficulty lies in the reliable cross-view matching of onboard UAV imagery with geo-referenced satellite databases, which is often hindered by significant discrepancies in viewpoint, scale, and illumination. In this work, we propose a robust visual place recognition framework that integrates a fine-tuned YOLO11 object detection model with a novel statistical distribution alignment method to bridge the domain gap between aerial and satellite views. Our approach specifically targets building segmentation to extract semantically meaningful vector representations, achieving an F1-score of 0.722 on a dedicated building dataset. Furthermore, we introduce a cumulative distribution function (CDF) alignment technique that standardizes pixel intensity distributions, significantly enhancing visual consistency across modalities. Experimental evaluation on the VPAIR dataset demonstrates that the proposed pipeline achieves a Recall@1 score of 0.195 with a localization radius of 3 in urban scenarios, outperforming existing CNN-based methods such as CosPlace. The significant conclusion of this study is that leveraging YOLO-derived vector representations in conjunction with rigorous statistical preprocessing provides a computationally efficient and accurate solution for global UAV localization in complex urban terrains.

Keywords

Visual place recognition (VPR), UAV, YOLO, image preprocessing, deep learning, image segmentation

1. Introduction

Accurate localization of Unmanned Aerial Vehicles (UAVs) in complex environments is essential for applications such as disaster response, environmental monitoring, precision agriculture, and urban planning. While Global Navigation Satellite Systems (GNSS) such as GPS remain the standard solution, their performance is often compromised by signal blockage, interference, or multipath effects, particularly in urban canyons or areas with dense infrastructure [1]. In such GPS-denied scenarios, vision-based localization emerges as an attractive alternative, offering low-cost, information-rich positioning that does not suffer from cumulative drift [2].

Visual Place Recognition (VPR) has become a prominent vision-based approach, allowing UAVs to localize themselves by matching onboard camera views to geo-referenced imagery [3]. Of particular interest is cross-view matching between UAV images and satellite data, since satellites provide global coverage and accessible references. However, this task is highly challenging due to substantial viewpoint, scale, and resolution differences, as well as variations in color, illumination, weather, and seasonal conditions [4]. Repetitive structures such as rooftops and fields further reduce the availability of distinctive landmarks. Classical feature-based methods (e.g., SIFT, SURF) have proven unreliable in such scenarios, highlighting the need for more robust alternatives.

ExplAI-2025: Advanced AI in Explainability and Ethics for the Sustainable Development Goals, November 07, 2025, Khmelnytskyi, Ukraine

*Corresponding author.

vozniakvz@khmnu.edu.ua (V. Vozniak); y.ushenko@chnu.edu.ua (Y. Ushenko); morkenj@mail.ru (O. Mamyrbayev)
0009-0008-3055-5257 (V. Vozniak); 0000-0003-1767-1882 (Y. Ushenko); 0000-0001-8318-3794 (O. Mamyrbayev)

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2025-12-22

Recent progress in deep learning has enabled the extraction of viewpoint- and appearance-invariant embeddings that significantly improve cross-view recognition. Convolutional neural networks (CNNs), when trained on large datasets, can learn semantically meaningful features and achieve high recall despite extreme perspective differences [5]. In addition, preprocessing methods, including geometric rectification, color and illumination normalization, and statistical distribution alignment, further reduce the domain gap between UAV and satellite imagery, thereby stabilizing visual descriptors across diverse conditions [6].

Considering the limited computational resources on UAV platforms, solutions must combine accuracy with efficiency. The YOLO family of object detection networks provides an effective balance, offering real-time inference and high accuracy on resource-constrained hardware [7]. YOLO models deliver multi-scale feature representations while focusing on salient objects, making them suitable backbones for UAV localization pipelines.

The contributions of this research include:

- development of a method to obtain robust image embeddings from the convolutional layers of a fine-tuned YOLO11 model trained on a dataset of segmented buildings;
- incorporation of preprocessing and distribution alignment techniques (e.g., cumulative distribution function normalization) to mitigate domain discrepancies between UAV and satellite imagery;
- demonstration of an efficient pipeline for global UAV localization that maintains real-time performance on edge devices while enhancing robustness under GPS-denied conditions.

The paper is structured as follows: Section 2 presents related work and defines the research objectives. Section 3 outlines the proposed UAV localization framework, with emphasis on preprocessing and distribution alignment methods. Section 4 reports experimental results, analyzing the performance of the fine-tuned YOLO model and comparing it with existing approaches. Section 5 concludes the paper and discusses directions for future work.

2. Related works

Modern visual place recognition (VPR) is commonly formulated as an image retrieval problem: a query image (e.g., UAV snapshot) is compared against a large database of geo-referenced references (e.g., satellite tiles), and the closest match indicates the UAV's location. The central element of this process is the image descriptor. Early approaches relied on handcrafted global descriptors or bag-of-visual-words built from local features such as SURF or SIFT (e.g., FAB-MAP [8], DBoW2 [9]). While effective under moderate viewpoint changes, these methods proved unreliable in UAV-to-satellite scenarios, where viewpoint, scale, and illumination differences are severe.

The emergence of deep learning brought about a step-change in descriptor quality. CNN-based embeddings demonstrated robustness to lighting and viewpoint variations, substantially performing engineered features. A seminal contribution was NetVLAD [10], which combined a CNN backbone with a VLAD aggregation layer, trained on large datasets such as Pitts-250k, to produce compact global descriptors. Subsequent works refined this idea: Patch-NetVLAD [11] incorporated multi-scale feature aggregation for improved viewpoint invariance, though at the cost of high-dimensional vectors and increased memory requirements.

Alongside CNN-based models, research has explored specialized datasets and cross-view learning paradigms. For example, the University-1652 dataset [12] introduced UAV imagery for building-level geo-localization, enabling Siamese and triplet networks to learn a common embedding space for UAV and satellite views. Other benchmarks such as VIGOR [13], SUES-200 [14], ALTO [15], and VPAIR [16] have further pushed evaluation toward multi-altitude, multi-terrain, and rotation-robust settings, highlighting the importance of large-scale, curated datasets for advancing cross-view VPR.

Lightweight and multimodal descriptors have also been proposed. MinkLoc employed sparse 3D convolutions for large-scale recognition, effective with LiDAR or depth data, though impractical for UAVs due to payload and energy constraints. More recently, CosPlace [17] reframed VPR as a classification problem, training on the massive San Francisco XL dataset of 40 million directional images, while MixVPR [18] introduced an MLP-based feature mixer trained on GSV-Cities (530,000 images from 62,000 locations) [19]. These advances underline the trend toward dataset-driven improvements and the need for scalable embeddings.

The rise of transformer architectures has added further capabilities through attention-based modeling. LoFTR [20] removed explicit keypoint detectors, directly matching local features, while AnyLoc [21] leveraged self-supervised DinoV2 [22] features and combined local and global attention modules. Such approaches achieve high benchmark scores but remain computationally heavy, often unsuitable for real-time UAV deployment due to high-dimensional descriptors and slow inference.

Another important strand of research addresses photometric discrepancies between UAV and satellite imagery. Preprocessing techniques such as histogram matching, mean-variance color normalization, and style transfer help reduce domain gaps [23]. More advanced strategies, such as cumulative distribution-based color alignment, standardize UAV imagery toward satellite-like appearance, improving feature consistency across modalities. In addition, style augmentation during training has been shown to improve robustness against weather, seasonal, and illumination variations.

In summary, state-of-the-art VPR methods range from compact CNN-based global descriptors to transformer-based hybrid models and self-supervised embeddings. While transformers and foundation models provide highly discriminative features, CNN-based pipelines remain attractive for UAV applications due to their computational efficiency. Surprisingly, despite YOLO's established balance of accuracy and speed in object detection, it has not yet been systematically explored for VPR.

Based on the literature analysis, the identified research gaps are as follows:

- limited deployment of real-time VPR pipelines optimized for UAV hardware with constrained computational resources;
- high-dimensional descriptors in methods such as NetVLAD [10] and Patch-NetVLAD [11] hinder memory efficiency and scalability;
- transformer-based approaches achieve strong accuracy but remain impractical for UAVs due to inference latency and resource demands;
- color and style normalization techniques exist but are often dataset-specific; generalizable domain alignment methods are still lacking;
- YOLO-based architectures, despite their proven efficiency in detection tasks, have not been systematically adapted or evaluated for UAV–satellite VPR.

The objective of this work is to build an enhanced visual place recognition system that aligns UAV images with satellite views, integrating deep embedding models and color preprocessing techniques to achieve greater robustness and precision, especially in urban cross-view conditions. In particular, the target system is designed to reliably recognize places from a low-altitude UAV perspective by matching against satellite imagery, even under significant viewpoint and appearance changes.

To achieve this goal, we have formulated the following tasks:

- fine-tune the YOLO11 model on a dataset with segmented buildings to obtain vector representations for UAV global location determination under challenging conditions;
- develop a method for aligning the statistical distributions of satellite imagery and UAV images to achieve visual similarity;
- compare results obtained by the proposed method against existing CNN-based methods across different terrain types.

3. Materials and methods

3.1. Methods

Determining the position of Unmanned Aerial Vehicles (UAVs) is commonly formulated as a Visual Place Recognition (VPR) problem. In the proposed approach, a query image captured by the UAV is compared against a database of geo-referenced satellite imagery. The UAV's global location is first estimated by retrieving the most visually similar satellite image (or set of candidates). This is then refined by aligning the query image with the selected satellite image to obtain precise geographic coordinates.

Accordingly, the task can be divided into two main stages:

- global localization: retrieving the most similar satellite tile from a large reference database;
- coordinate refinement: computing the exact position (latitude and longitude) within the matched tile.

The overall processing workflow for UAV localization is depicted in Figure 1.

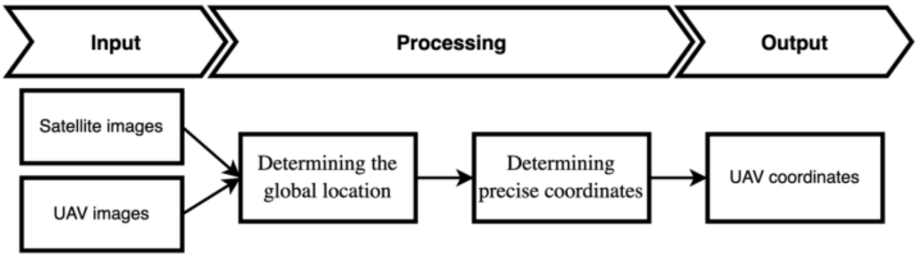

Input
Processing
Output
Satellite images
UAV images
Determining the global location
Determining precise coordinates
UAV coordinates

Figure 1: General data processing scheme for determining UAV location coordinates.

In this study, convolutional neural network (CNN) architectures are employed to address the problem of UAV global localization in urban environments. The overall structure of the proposed approach is illustrated in Figure 2.

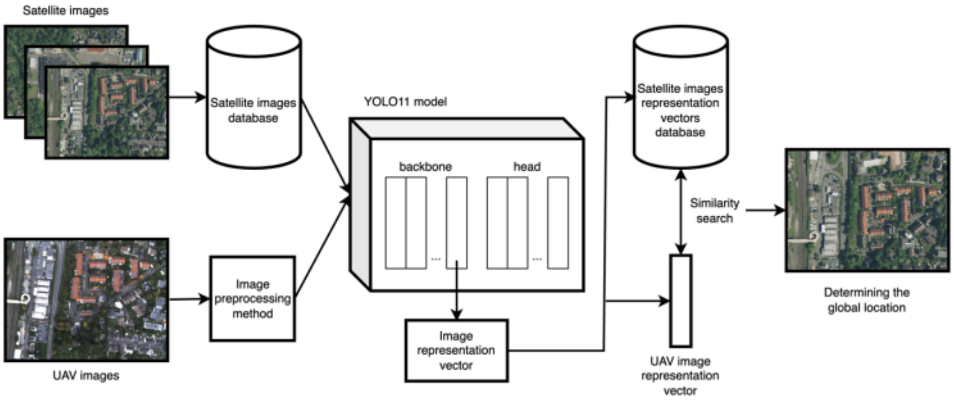

Satellite images
Satellite images database
YOLO11 model
backbone
head
...
...
Satellite images representation vectors database
Similarity search
Image preprocessing method
Image representation vector
UAV image representation vector
Determining the global location
UAV images

Figure 2: Processing scheme for UAV global localization using an image preprocessing method and a YOLO11 model fine-tuned on a dataset of segmented buildings.

We adopt the following notation: let $Q = \{q\}$ denote a query image captured by the UAV at a specific position and orientation, and let $\mathcal{R} = \{r_1, r_2, \dots, r_n\}$ represent a database of reference images with known geographic coordinates (typically satellite imagery).

To obtain image descriptors, we define a mapping function from the image domain I to the feature space V.

$$F : I \to V, \tag{1}$$

where I is the image, V is its corresponding feature vector.

The task of UAV global localization is then reduced to finding the reference image $r_j \in \mathcal{R}$ whose location $\ell(r_j)$ is closest to that of the query image $\ell(q)$. Formally, this can be expressed as:

$$k = \operatorname*{argmin}_{1 \le j \le n} d\left(F(q), F(r_j)\right), \tag{2}$$

where $d(\cdot,\cdot)$ is a distance metric (e.g., Euclidean distance) applied to feature vectors, and k denotes the index of the retrieved satellite image.

To enable global UAV localization, a database of satellite images along a predefined route must first be established. Using deep learning models, each image can be transformed into a vector representation (feature embedding) as defined in (1). In this study, the YOLO11 model is fine-tuned on a dataset of segmented buildings, and descriptors are extracted from its final backbone layer, which encodes the most comprehensive information produced by YOLO11's convolutional filters. The same model is applied to both satellite and UAV images, ensuring consistent feature representations across domains. Consequently, the fine-tuned YOLO11 network generates embeddings for the satellite database and, in real time, processes incoming UAV frames to produce query embeddings (1). The UAV's global location is then identified by selecting the satellite image whose descriptor minimizes a similarity measure (e.g., Euclidean distance) with the UAV embedding (2).

To further reduce domain discrepancies between UAV and satellite imagery, we propose a distribution alignment method that enhances visual similarity at the pixel level. Unlike prior approaches that directly apply cumulative distribution functions (CDF) from satellite to UAV images – which can introduce distortions due to mismatched distributions – our method computes transformations individually for each UAV frame, taking into account its unique statistical properties.

The approach is grounded in probability theory. For a random variable ξ with distribution function $F_\xi(x)$, one can obtain a uniformly distributed variable $\gamma \sim U(0,1)$ by applying the transformation $\gamma = F_\xi(\xi)$. Conversely, applying the inverse CDF to $\gamma \sim U(0,1)$ yields a variable ξ with the distribution $F_\xi(x)$. Satellite images, acquired using consistent sensors, exhibit stable color characteristics, while UAV imagery varies significantly depending on illumination and acquisition conditions. Treating pixel intensities as discrete random variables, we denote UAV and satellite intensities by Y, and X respectively. By computing $F_X(x)$ from the satellite dataset and estimating $F_Y(y)$ from UAV images, the alignment transformation is given by:

$$y' = F_X^{-1}\left(F_Y(y)\right), \tag{3}$$

where y' denotes the transformed UAV intensity aligned to the satellite domain.

Figure 3 illustrates the proposed alignment process based on these probabilistic principles.

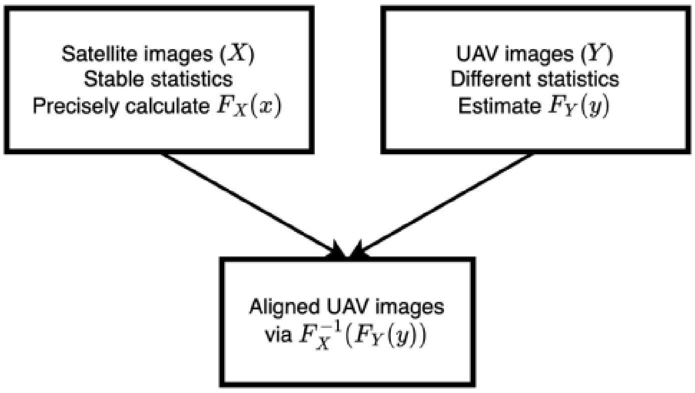


Figure 3: Diagram of the proposed method based on the probability theory.

The proposed method comprises two stages:

1. Computation of the averaged cumulative distribution function from the satellite image set.
2. Application of the averaged function individually to UAV frames for appearance normalization.

Algorithm 1 provides pseudocode for computing the averaged satellite CDF, while Algorithm 2 outlines its application to UAV images.

Listing 1: Computing the averaged cumulative distribution function from satellite images

```
Input: R = {r_1, r_2, ..., r_n} - n satellite images.
Initialization: $F^{sum} \leftarrow 0_{x \times c}, x \in [0;255], c \in \{R,G,B\}$.
For i in 1..n
    Step 1. Compute the normalized histogram (probability density function) for each color channel $c \in \{R,G,B\}$:
        $E_{c,i}(x) = \frac{H_{c,i}(x)}{\sum_{x=0}^{255} H_{c,i}(x)}$,
        where $H_{c,i}(x)$ is the number of pixels with value $x$ in channel $c$
            of the $i$-th satellite image.
    Step 2. Compute the cumulative distribution function (CDF) for each channel:
        $F_{c,i}(x) = \sum_{k=0}^{x} E_{c,i}(k)$.
    Step 3. Add the cumulative distribution function values to $F^{sum}$ to later derive the average cumulative distribution function:
        $F_i(x) = [F_{R,i}(x), F_{G,i}(x), F_{B,i}(x)]$;
        $F^{sum}(x) = F^{sum}(x) + F_i(x)$.
End
Step 4. Compute the averaged cumulative distribution function (CDF):
    $\hat{F}(x) = \frac{F^{sum}(x)}{n}$.
Output: $\hat{F}(x)$ - averaged cumulative distribution function of satellite images, $x \in [0;255]$.
```

Listing 2: Applying averaged cumulative distribution function of satellite images individually to UAV images

```
Input: Q = {q_1, q_2, ..., q_m} - m UAV images.
For j in 1..m
    Step 1. Compute the normalized histogram (probability density function) for each color channel $c \in \{R,G,B\}$:
        $D_{c,j}(y) = \frac{S_{c,j}(y)}{\sum_{y=0}^{255} S_{c,j}(y)}$,
        where $S_{c,j}(y)$ is the number of pixels with value $y$ in channel $c$
            of the $j$-th UAV image.
    Step 2. Compute the cumulative distribution function (CDF) for each channel:
        $G_{c,j}(y) = \sum_{n=0}^{y} E_{c,j}(n)$.
    Step 3. Define the transformation function for each channel $c$ by finding the value at which the cumulative distribution function (CDF) from the UAV images aligns with the averaged cumulative distribution function of satellite images:
        $M_{c,j}(y) = \hat{F}_c^{-1}(G_{c,j}(y))$,
        where $\hat{F}_c^{-1}$ is the inverse function of the averaged cumulative
            distribution function for satellite images in channel $c$.
            Because $\hat{F}_c^{-1}$ might be non-analytical, interpolation is used
             as an approximation:
        $M_{c,j}(y) = \mathrm{interp}(G_{c,j}(y), \hat{F}_c(z), z)$,
        where interp is an interpolation function that finds the
            corresponding $z$ for each $G_c(y)$, such that $\hat{F}_c(z) \approx G_c(z)$.
End
Output: $M_{c,j}(y)$ - the resulting function that transforms the input
```

pixels of the j-th UAV image for the given color channel $c, y \in [0; 255]$.

Histogram alignment standardizes pixel intensity distributions between UAV and satellite images by adjusting brightness, contrast, and tonal characteristics. This reduces intra-class variability and improves consistency for feature extraction, thereby enhancing matching accuracy and recognition performance.

3.2. Evaluation

The performance of image segmentation models is commonly evaluated using metrics such as mean Average Precision (mAP), Precision, Recall, and F1-score. Precision, Recall, and F1-score [24] are widely used across machine learning tasks, including binary classification and image segmentation. A distinctive aspect of segmentation evaluation is the absence of true-negative counts in the confusion matrix, which, however, does not affect the computation of these metrics.

Among these measures, mean Average Precision (mAP) is of particular importance and can be formally defined as:

$$AP_c = \sum_{n=1}^{N} (R_n - R_{n-1}) P_n, \tag{4}$$

$$mAP = \frac{1}{C} \sum_{c=1}^{C} AP_c, \tag{5}$$

where P_n and R_n are Precision and Recall at the threshold n with $R_0 = 0$ and $R_N = 1$, C is the number of classes, AP_c – average precision for the class c.

Conceptually, AP_c corresponds to the area under the Precision–Recall curve for a given class. Since this study focuses on segmentation of a single class (buildings), the evaluation reduces to $mAP = AP_b$.

For localization tasks, Recall@N [25] is a standard metric. A query is considered correctly localized if at least one relevant image from the database appears within the top-N retrieved results:

$$\text{Recall@N} = \frac{M_Q}{N_Q}, \tag{6}$$

where N_Q is the total number of query images, and M_Q is the number of queries with at least one correct match within the top-N results. This measure is particularly suitable when subsequent processing steps can further refine or filter false positives.

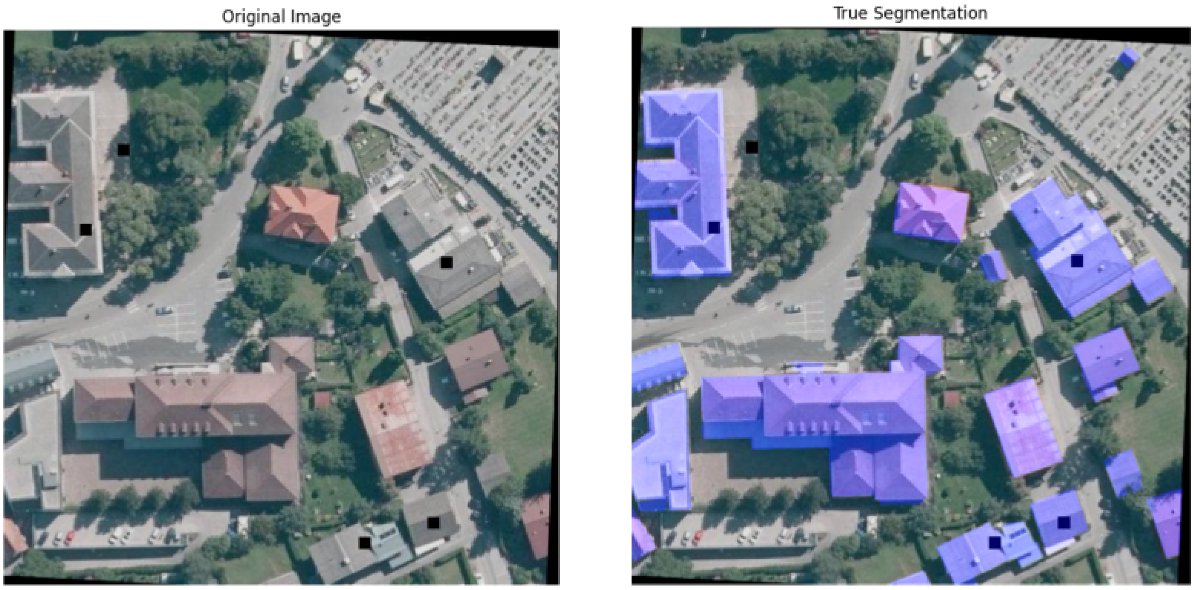
An extended definition of Recall@N incorporates a localization radius, whereby a result is deemed correct if the distance between the query and retrieved images falls within a predefined threshold. This radius can be specified either in physical units (e.g., meters) or in frame indices when satellite images are sequentially ordered. Such a formulation is especially useful in scenarios with overlapping reference imagery, allowing precise UAV localization even in the absence of an exact image match.

4. Results and discussion

4.1. YOLO finetuning

Since the standard YOLO model does not include buildings as a predefined class, this study fine-tunes the YOLO11 [26] architecture using a dedicated building segmentation dataset [27]. The dataset contains 9,665 images, with major contributions from cities such as Tyrol (2,999), Tripoli (1,078), Kherson (1,053), Donetsk (999), Mekelle (951), Mykolaiv (739), and Kharkiv (602). An example of annotated building segmentation from the dataset is shown in Figure 4.

Fine-tuning was carried out with the YOLO11 implementation provided by the open-source ultralytics library [28], executed on an Ubuntu environment with an Nvidia MSI RTX 3060 GPU. The training



(a) Original Image

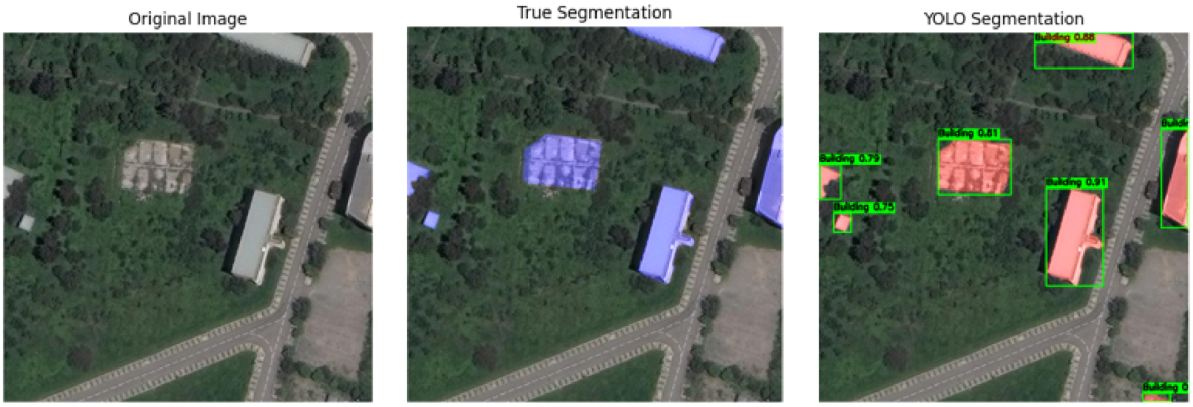
(b) True Segmentation

Figure 4: An example of segmented buildings from one image of the training dataset [27].

targeted only building segmentation, using the default YOLO11 configuration with 100 epochs and an input resolution of 640 pixels.

Model performance was evaluated using mAP, Precision, Recall, and F1-score. Among these, the F1-score was emphasized as the primary metric due to its balanced consideration of false negatives (missed buildings) and false positives (incorrect detections). To ensure reliability, results were averaged (Avg) and standard deviations (Std) were calculated across seven randomized 80/20 splits of the dataset into training and testing subsets.

Figure 5 presents an example of segmentation output from the test set used during YOLO11 fine-tuning, while Figure 6 demonstrates segmentation performance on an image from the VPAIR dataset.



(a) Original Image

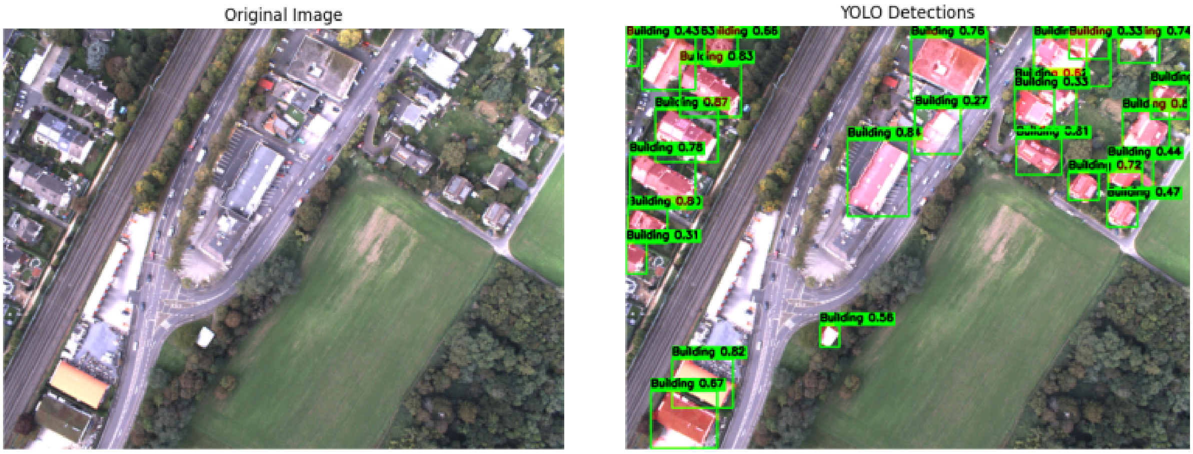
(b) True Segmentation

(c) YOLO Segmentation

Figure 5: An example of building segmentation on an image from the test dataset used for fine-tuning YOLO11.

Figure 7 illustrates the training and validation loss curves obtained during fine-tuning of the YOLO11 model on the optimal train/test split. The YOLO architecture employs a composite loss function, calculated as a weighted sum of several components, each addressing a different aspect of the segmentation task:

- `box_loss` – emphasizes the accurate placement of bounding boxes around detected objects



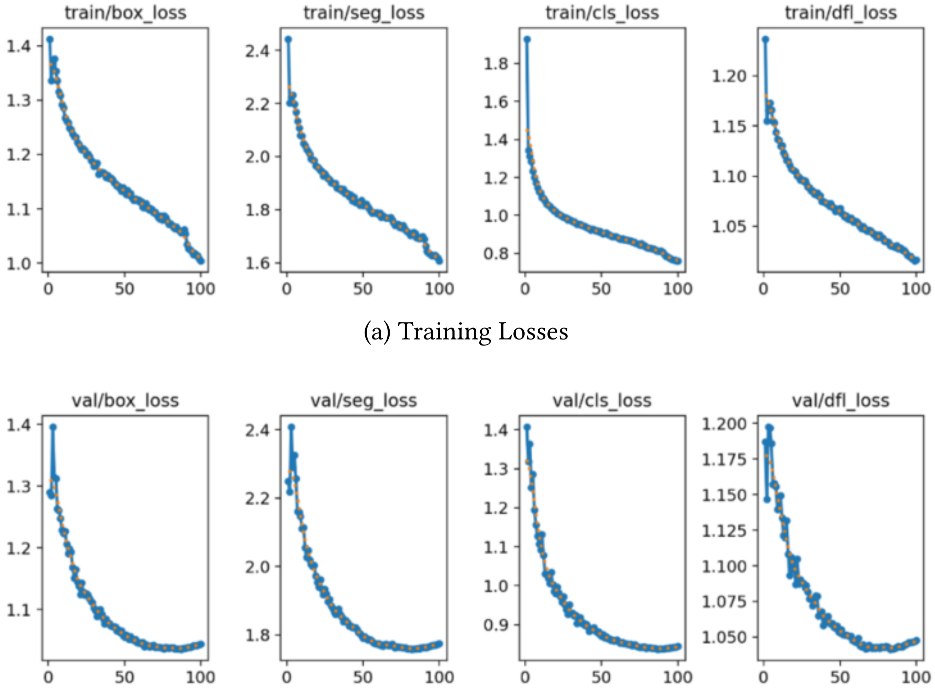
(a) Original Image

(b) YOLO Detections

Figure 6: An example of building segmentation on an image from the VPAIR dataset.

(weight coefficient: 7.5);

- `seg_loss` – enforces accurate delineation of object segmentation masks (weight: 7.5);
- `cls_loss` – accounts for correct classification of detected objects (weight: 0.5);
- `dfl_loss` – improves discrimination between visually similar or ambiguous objects by emphasizing distinctive features (weight: 1.5).



(a) Training Losses

(b) Validation Losses

Figure 7: Loss function plots for the training (train) and validation (val) sets.

The loss curves demonstrate the model's ability to effectively learn from the training data and generalize to unseen examples. This is reflected in the steady decrease of training loss and the subsequent stabilization of validation loss, indicating convergence without significant overfitting. Figure 8 presents the confusion matrix, with true negatives omitted, as their definition is not well-defined in the context of image segmentation.

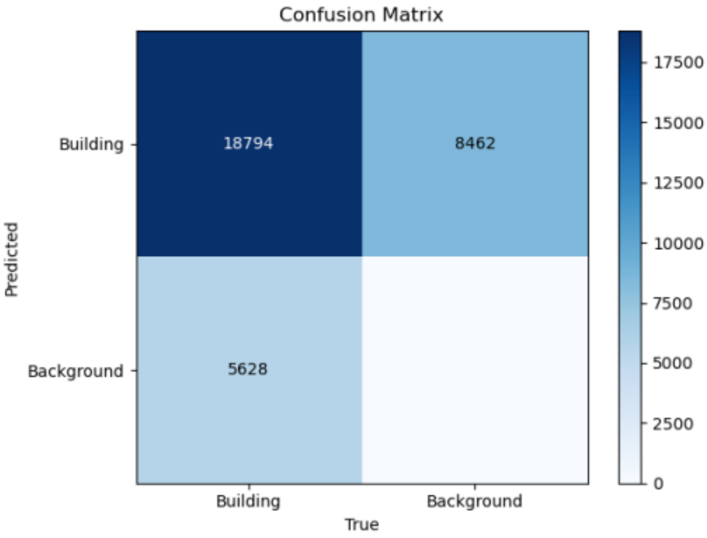


Figure 8: The confusion matrix for the YOLO11 fine-tuned model on the segmented buildings dataset.

Table 1 reports the evaluation metrics for the YOLO11 model fine-tuned on the building segmentation dataset. The results include mean Average Precision (mAP), Precision, Recall, and F1-score, calculated across seven distinct train/test splits. To assess robustness, both average (Avg) values and standard deviations (Std) are provided, reflecting the stability of the model's performance.

Table 1
Evaluation metrics obtained from the fine-tuned YOLO11 model on the segmented building dataset. Text in **bold** represents higher values.

Metric		Random splitting							Avg	Std
		1	2	3	4	5	6	7		
mAP	Train	0.813	0.821	0.820	0.823	0.813	0.817	0.814	0.817	0.0043
	Test	0.748	0.760	0.757	0.755	0.757	0.753	0.749	**0.754**	0.0043
Recall	Train	0.727	0.737	0.736	0.738	0.728	0.730	0.728	0.732	0.0047
	Test	0.677	0.685	0.681	0.679	0.681	0.680	0.673	**0.680**	0.0037
Precision	Train	0.809	0.819	0.815	0.820	0.809	0.813	0.813	0.814	0.0042
	Test	0.764	0.773	0.775	0.768	0.778	0.770	0.769	**0.771**	0.0047
F1	Train	0.766	0.775	0.773	0.777	0.766	0.769	0.768	0.771	0.0044
	Test	0.718	0.727	0.725	0.721	0.726	0.722	0.718	**0.722**	0.0037

Figure 9 shows the Precision–Recall (PR) curve for the best-performing split, with an achieved area under the curve (AUC [29]) of 0.76. The PR curve is especially informative in segmentation tasks, as it highlights the model's ability to correctly detect positive samples (buildings) under conditions of significant class imbalance between foreground structures and background regions.

For building segmentation tasks involving partially occluded structures or complex architectural outlines, the fine-tuned YOLO11 model achieved an F1-score of 0.722 on the test set. This result is particularly encouraging given the additional advantage of real-time inference inherent to YOLO-based architectures. The outcome indicates that the model can reliably detect buildings, a capability essential for generating vector data required in UAV-based global localization pipelines. Moreover, the standard deviation of all evaluation metrics remained below 0.5% across both training and testing splits,

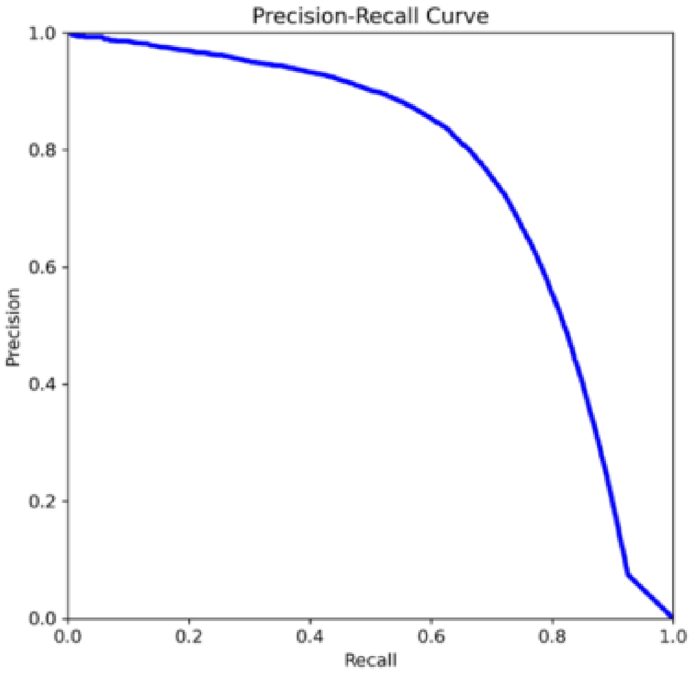


Figure 9: Precision-Recall curve for the YOLO11 fine-tuned model on the segmented buildings dataset.

confirming the robustness and stability of model performance. Future research may explore architectural modifications to YOLO11 and fine-tuning of hyperparameters to further improve segmentation accuracy, with a focus on challenging building recognition scenarios.

4.2. Visual place recognition with image preprocessing method

To validate the proposed method for aligning statistical distributions between UAV and satellite images, experiments were conducted using the VPAIR dataset [16]. This dataset was specifically designed for UAV localization under challenging real-world conditions. Data were collected during a 107-kilometer flight from Bonn into the mountainous Eifel region of Germany on October 13, 2020. The imagery spans diverse terrain types, including urban areas, agricultural fields, and forests. Data acquisition was performed with a single-lens color camera at a resolution of 1600×1200 pixels, later downsampled to 800×600 pixels for dataset inclusion. Each image is paired with highly accurate GNSS/INS ground truth (rotational error: 0.05°, positional accuracy: <1 m). The dataset consists of 2,706 UAV query images, an equal number of corresponding satellite references, and an additional 10,000 distractor images collected near Düsseldorf.

Since the original dataset did not provide terrain-type annotations, a new classification scheme was introduced, grouping images into four categories: urban (dominated by buildings and roads), field, forest, and water.

Validation of the proposed distribution alignment method was performed using the Recall@1 metric with a localization radius of 3. Experiments evaluated performance across all terrain categories, with a particular focus on urban environments, reflecting the specialization of the fine-tuned YOLO11 model on building segmentation for feature extraction.

Three comparative experiments were carried out to benchmark the proposed approach against state-of-the-art methods such as CosPlace:

- using the original, unprocessed VPAIR images;
- using grayscale-converted images;
- using UAV imagery preprocessed with the proposed averaged cumulative distribution function (CDF) alignment method.

Figure 10 illustrates the visual outcomes of the CDF-based preprocessing, showing UAV images transformed to statistically match the distribution of corresponding satellite imagery.

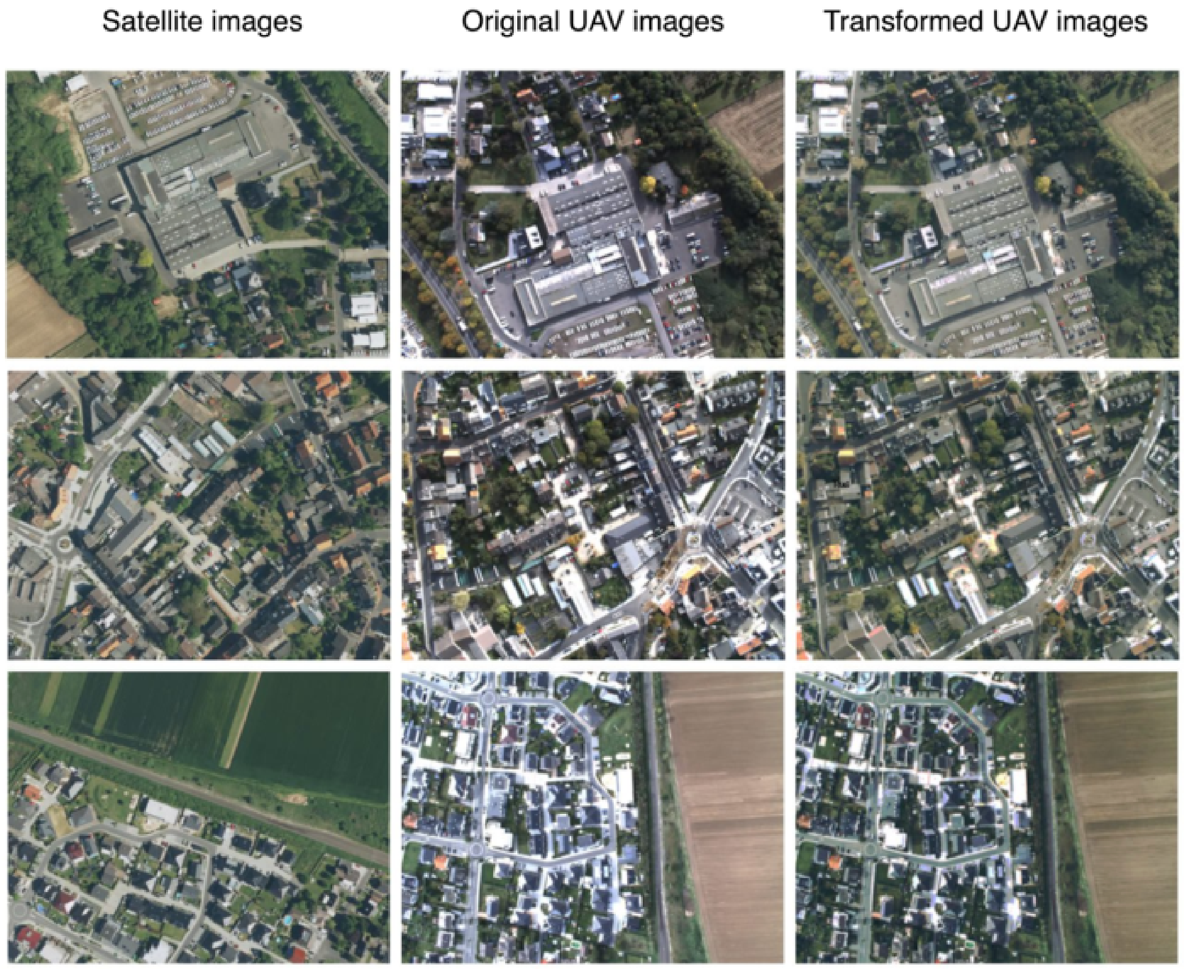


Figure 10: Visual results of the proposed averaged cumulative distribution function (CDF) method applied to UAV images, aligning their statistical distributions with those of satellite images. From left to right: satellite images – original UAV images – transformed UAV images.

Table 2 provides a concise yet comprehensive overview of how each stage of the evaluation pipeline influences final localization accuracy. For every UAV query image in the VPAIR dataset [16], the following sequence was executed under three color-handling variants, after which Recall@1 with a localization radius of 3 was recorded:

- color handling – retain the original RGB image, convert to single-channel grayscale, or apply the proposed averaged cumulative distribution function (CDF) transfer to align UAV pixel statistics with the satellite domain;
- embedding extraction – process the pre-processed image using either CosPlace or the fine-tuned YOLO11 model;
- nearest-neighbor retrieval – perform L2-based similarity search against the 2,706-image satellite reference set, selecting the closest match as the predicted location;
- localization test – classify the prediction as correct if the matched satellite tile lies within three reference frames of the ground-truth tile; otherwise mark as incorrect;
- metric aggregation – compute Recall@1 for each terrain category (Urban, Field, Forest, Water) by aggregating correct matches over all 2,706 queries.

Table 2
Recall@1 metric results with a localization radius of 3 across different terrain types from the VPAIR dataset. Text in **bold** represents higher values.

Method	Urban	Field	Forest	Water
Without color preprocessing				
CosPlace [17]	0.140	**0.097**	**0.122**	**0.364**
YOLO (Our)	**0.181**	0.049	0.055	0.345
Using greyscale preprocessing				
CosPlace [17]	0.132	**0.093**	**0.114**	**0.366**
YOLO (Our)	**0.175**	0.030	0.038	0.255
Using proposed averaged cumulative distribution function				
CosPlace [17]	0.145	**0.108**	**0.134**	0.374
YOLO (Our)	**0.195**	0.068	0.070	**0.545**

The results demonstrate that CNN-based localization methods (CosPlace, YOLO11) benefit from the proposed preprocessing strategy, confirming its effectiveness in improving UAV global localization accuracy. While CosPlace performs better across non-urban terrains such as fields, forests, and water, the YOLO11-based approach achieves superior results in urban settings, reflecting its fine-tuning on building segmentation data.

In urban scenarios, the proposed method achieved a Recall@1 score of 0.195 (19.5%) within a localization radius of 3, a promising outcome given the inherent difficulty of sustaining high accuracy at top ranks in UAV global localization. This performance not only underscores the robustness of the YOLO11-based technique but also demonstrates its competitiveness against established approaches such as CosPlace.

Nevertheless, the method has certain limitations. Its applicability is currently constrained to urban areas under daytime and favorable weather conditions, and it requires prior availability of satellite imagery along predefined UAV flight routes.

5. Conclusions

The primary objective of this study was successfully achieved: the development of a robust UAV–satellite image matching framework that leverages deep embeddings and color normalization to improve precision and robustness under challenging cross-view and urban conditions. Fine-tuning the YOLO11 model on a dataset of segmented buildings enabled the extraction of vector representations that substantially enhanced UAV-to-satellite image matching accuracy.

The proposed preprocessing strategy, based on aligning statistical distributions between UAV and satellite imagery, further improved visual consistency, outperforming established methods such as CosPlace, particularly in urban terrain. Quantitative results confirm this advantage, with the model achieving an F1-score of 0.722 for building segmentation and a Recall@1 of 19.5% within a localization radius of 3, surpassing existing benchmarks for urban UAV localization.

Key strengths of this work include the ability to achieve accurate and efficient UAV localization in GPS-denied environments, supported by YOLO11's inherent real-time inference capability. This makes the approach highly relevant for practical applications such as emergency response, urban surveillance, and infrastructure monitoring.

Despite these advances, several limitations remain. The method is currently optimized for urban scenarios under daytime and favorable weather conditions, and its effectiveness depends on the availability of pre-collected satellite imagery aligned with potential UAV flight paths. To mitigate seasonal variability, reference datasets should ideally be captured during late spring, summer, or early autumn, when environmental appearance is most stable. Additionally, to ensure robust feature extraction, imagery should meet a minimum resolution of 640×640 pixels, i.e., consistent with the YOLO input requirements, which also normalize resolution differences between UAV and satellite imagery.

Future work will focus on extending the versatility and precision of the YOLO11-based pipeline through advanced augmentation strategies, architectural refinements, and hyperparameter optimization. Expanding localization capabilities beyond urban settings to include forests, agricultural regions, and aquatic environments will further enhance adaptability, broadening the practical scope of UAV mission scenarios.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] O. Melnychenko, L. Scislo, O. Savenko, A. Sachenko, P. Radiuk, Intelligent integrated system for fruit detection using multi-UAV imaging and deep learning, Sensors 24 (2024) 1913. doi:10.3390/s24061913.

[2] Y. Wang, X. Li, R. Zhang, T. Huang, L. Zhao, Lightweight visual localization algorithm for uavs, Sci. Rep. 15 (2025) 6069. doi:10.1038/s41598-025-88089-y.

[3] Z. Cui, H. Huang, M. Zheng, L. Zhang, F. Xu, A novel geo-localization method for uav and satellite images using cross-view consistent attention, Remote Sens. 15 (2023) 19. doi:10.3390/rs15194667.

[4] Y. Yao, C. Sun, T. Wang, J. Yang, E. Zheng, Uav geo-localization dataset and method based on cross-view matching, Sensors 24 (2024) 6905. doi:10.3390/s24216905.

[5] O. Zalutska, M. Molchanova, O. Sobko, O. Mazurets, O. Pasichnyk, O. Barmak, I. Krak, Method for sentiment analysis of Ukrainian-language reviews in e-commerce using RoBERTa neural network, in: Proceedings of the 7th International Conference on Computational Linguistics and Intelligent Systems (CoLInS 2023), Volume I: Machine Learning Workshop, volume 3387, CEUR-WS.org, Aachen, 2023, pp. 344–356. URL: https://ceur-ws.org/Vol-3387/paper26.pdf.

[6] O. Melnychenko, O. Savenko, P. Radiuk, Apple detection with occlusions using modified YOLOv5-v1, in: Proceedings of the 12th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS 2023), IEEE, New York, NY, USA, 2023, pp. 107–112. doi:10.1109/IDAACS58523.2023.10348779.

[7] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 779–788. doi:10.1109/CVPR.2016.91.

[8] M. Cummins, P. Newman, Fab-map: probabilistic localization and mapping in the space of appearance, Int. J. Robot. Res. 27 (2008) 647–665. doi:10.1177/0278364908090961.

[9] D. Gálvez-López, J. D. Tardós, Bags of binary words for fast place recognition in image sequences, IEEE Trans. Robot. 28 (2012) 1188–1197. doi:10.1109/TRO.2012.2197158.

[10] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, J. Sivic, Netvlad: Cnn architecture for weakly supervised place recognition, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 5297–5307.

[11] S. Hausler, S. Garg, M. Xu, M. Milford, T. Fischer, Patch-netvlad: multi-scale fusion of locally-global descriptors for place recognition, in: Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2021, pp. 14141–14152.

[12] Z. Zheng, Y. Wei, Y. Yang, University-1652: a multi-view multi-source benchmark for drone-based geo-localization, in: Proc. 28th ACM Int. Conf. Multimedia, 2020, pp. 1395–1403. doi:10.1145/3394171.3413896.

[13] S. Zhu, T. Yang, C. Chen, Vigor: cross-view image geo-localization beyond one-to-one retrieval, in: Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2021, pp. 3640–3649.

[14] R. Zhu, L. Yin, M. Yang, F. Wu, Y. Yang, W. Hu, Sues-200: a multi-height multi-scene cross-view image benchmark across drone and satellite, IEEE Trans. Circuits Syst. Video Technol. 33 (2023) 4825–4839. doi:10.1109/TCSVT.2023.3249204.

[15] I. Cisneros, P. Yin, J. Zhang, H. Choset, S. Scherer, Alto: a large-scale dataset for uav visual place recognition and localization, arXiv preprint arXiv:2207.12317 (2022). doi:10.48550/arXiv.2207.12317.

[16] M. Schleiss, F. Rouatbi, D. Cremers, Vpair: aerial visual place recognition and localization in large-scale outdoor environments, arXiv preprint arXiv:2205.11567 (2022). doi:10.48550/arXiv.2205.11567.

[17] G. Berton, C. Masone, B. Caputo, Re-thinking visual geo-localization for large-scale applications, in: Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2022, pp. 4878–4888.

[18] A. Ali-bey, B. Chaib-draa, P. Giguère, Mixvpr: feature mixing for visual place recognition, in: Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV), 2023, pp. 2998–3007.

[19] A. Ali-bey, B. Chaib-draa, P. Giguère, Gsv-cities: toward appropriate supervised visual place recognition, Neurocomputing 513 (2022) 194–203. doi:10.1016/j.neucom.2022.09.127.

[20] J. Sun, Z. Shen, Y. Wang, H. Bao, X. Zhou, Loftr: detector-free local feature matching with transformers, in: Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2021, pp. 8922–8931.

[21] N. Keetha, R. Arief, S. Adhikari, et al., Anyloc: towards universal visual place recognition, IEEE Robot. Autom. Lett. 9 (2024) 1286–1293. doi:10.1109/LRA.2023.3343602.

[22] M. Oquab, T. Darcet, T. Moutakanni, et al., Dinov2: learning robust visual features without supervision, arXiv preprint arXiv:2304.07193 (2023). doi:10.48550/arXiv.2304.07193.

[23] J. Shao, L. Jiang, Style alignment-based dynamic observation method for uav-view geo-localization, IEEE Trans. Geosci. Remote Sens. 61 (2023) 1–14. doi:10.1109/TGRS.2023.3337383.

[24] O. Rainio, J. Teuho, R. Klén, Evaluation metrics and statistical tests for machine learning, Sci. Rep. 14 (2024) 6086. doi:10.1038/s41598-024-56706-x.

[25] M. Zaffar, S. Ehsan, L. Momeni, et al., Vpr-bench: an open-source visual place recognition evaluation framework with quantifiable viewpoint and appearance change, Int. J. Comput. Vis. 129 (2021) 2136–2174. doi:10.1007/s11263-021-01469-5.

[26] Ultralytics, Yolo11 new, Available at: https://docs.ultralytics.com/models/yolo11, ???? Accessed 04.05.2025.

[27] Roboflow, Buildings instance segmentation – v1 raw-images, Available at: https://universe.roboflow.com/roboflow-universe-projects/buildings-instance-segmentation/dataset/1, ???? Accessed 04.05.2025.

[28] G. Jocher, J. Qiu, A. Chaurasia, Ultralytics yolo, github repository, Available at: https://github.com/ultralytics/ultralytics, 2023. Accessed 04.05.2025.

[29] Ş. K. Çorbacıoğlu, G. Aksel, Receiver operating characteristic curve analysis in diagnostic accuracy studies: a guide to interpreting the area under the curve value, Turk. J. Emerg. Med. 23 (2023) 182–187. doi:10.4103/tjem.tjem_182_23.