Unmanned Systems, Vol. 14, No. 4 (2026) 761–774

DOI: 10.1142/S2301385026500172

UNMANNED SYSTEMS
https://www.worldscientific.com/worldscinet/us

Continual Learning via Multiple Support Vector Models for Localization with a Single Aerial Image

Aldrich A. Cabrera-Ponce [*], Manuel I. Martin-Ortiz [*], Jose Martinez-Carranza [†,‡]

[*]Benemrita Universidad Autnoma de Puebla, Puebla, Mexico
[†]Instituto Nacional de Astrofisica, Optica y Electronica, Puebla, Mexico

Aerial localization using images captured by Unmanned Aerial Vehicles (UAVs) opens the door to different applications where the GPS may become inaccessible or lost. Deep learning approaches using Convolutional Neural Networks (CNNs) have shown promising localization results in known and static environments. However, these approaches can produce inaccurate pose estimations in scenarios with dynamic changes occurring over time. Therefore, we propose a methodology to generate multiple models that are trained online, and that can be immediately available for localization inference during the same operation flight. To this end, we study the use of Support Vector Regression (SVR) based on a Continual Learning Rehearsal Strategy. Using the SVR, we seek to generate compact models that can be integrated into a search strategy based on sub-maps created along the operation flight. Our method receives a single aerial image as input, thus leveraging features extracted by backbone networks, which will be used to regress the localization variables. For evaluation purposes, we compare our approach with PoseNet, ORB-SLAM2 and single-model training using the entire dataset, comprising no more than 300 images. The experiments were carried out on four trajectories, with our approach achieving a percentage error of around 7% of the total trajectory, with a processing time of 67 ms.

Keywords: Aerial localization; continual learning; multi-models; pose estimation; SVR.

1. Introduction

Localization is an important part of aerial robotics, enabling the positioning of UAVs during flight missions. Its applications span different fields such as inspection [1], crop field monitoring [2], and autonomous navigation [3]. As a result, computer vision techniques, mapping [4], and training methods have gained popularity in system development. Deep learning has become a widely used approach in robotics to obtain localization from a single image [5, 6].

However, deep learning approaches face several challenges, such as long training times, high computational costs, and the need for a large volume of data to build accurate localization models. These models must be frequently retrained when new data are acquired, especially in dynamic environments, making the process impractical. An alternative is to use compact networks to accelerate the training process and get a model in time [7]. Nonetheless, these methods still require a specific amount of data to build a training model and obtain a suitable result for the localization.

Existing solutions, such as methods involving LiDAR sensors or depth cameras [8], offer enhanced scene interpretation but are not always feasible due to their hardware complexity and costs. Similarly, approaches using hypernetworks [9] aim to reduce data requirements but still struggle to adapt to new environments without retraining, while traditional methods fail to address learning under dynamic conditions with limited computational resources. Overcoming these limitations requires continual localization in environments where the scene is constantly changing, optimising the training time, and minimising data dependency, especially when UAVs need to adapt in real-time.

Received 29 June 2024; Revised 17 February 2025; Accepted 17 February 2025; Published 24 April 2025. This paper was recommended for publication in its revised form by editorial board member, Zhi Gao.
Email Addresses: [‡]carranza@inaoep.mx
[‡]Corresponding author.

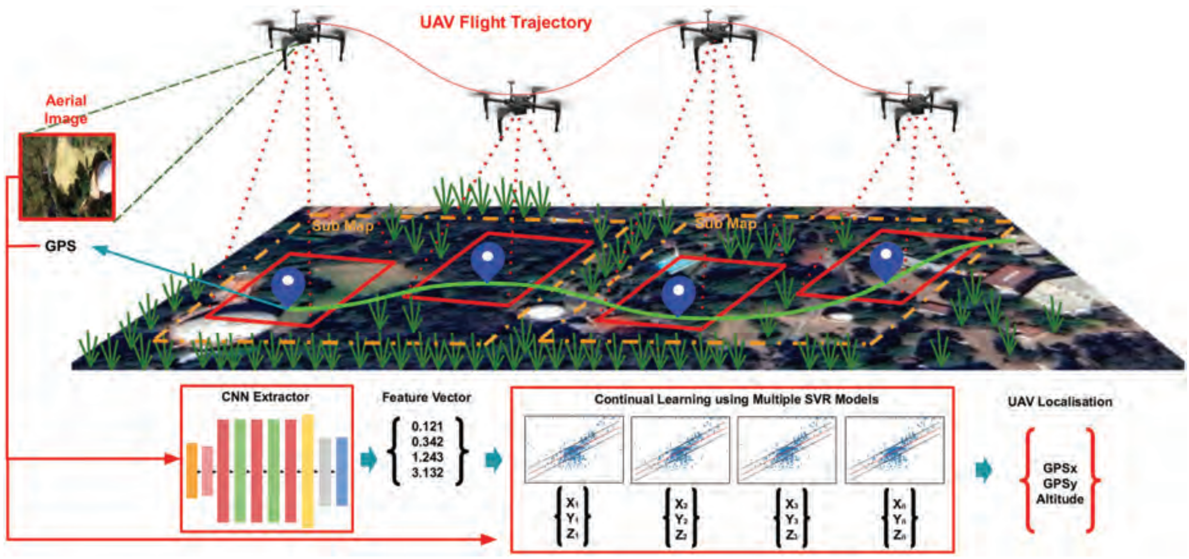


Fig. 1. Proposal methodology: We generate the dataset during the UAV flight mission, where the captured images are processed through a backbone network acting as a feature extractor. We propose using these features to train 3 SVR models, each representing the *x*, *y*, and *z* coordinates. In this way, we train new SVR models when new data arrive. We use this representation for our continual training and evaluation processes.

To tackle this issue, a branch of deep learning known as Continual Learning (CL) aims to acquire new information for updating learning without forgetting previous knowledge. These methods implement strategies within the network, such as optimizers and external memories [10] to preserve past information alongside the current data, refining the model. However, these approaches are designed for classification tasks and object detection, using partial data to accelerate the learning process. On the other hand, the external memories method has been applied to topological geolocalization with GPS and aerial images captured by monocular cameras onboard UAVs [11], enabling aerial localization.

Motivated by the above, this work aims to develop a localization methodology for UAVs using SVR across multiple models while minimising the training dataset to reduce processing and learning time. Drawing inspiration from the concept of sub-mapping in SLAM systems and leveraging recent information to create a model without forgetting the previous data, we present an approach that seeks to determine the corresponding sub-map in which the UAV is situated. Through multiple SVRs, we aim to estimate the UAV's pose close to the GPS ground truth. Thus, our methodology offers an alternative localization system in scenarios where the GPS signal is lost, enabling localization using a single image. Figure 1 illustrates our proposed framework for aerial localization using multiple SVRs.

The main contributions of this paper can be summarized as follows:

- We propose a sub-map search methodology to identify the area corresponding to the aerial image, leveraging the features extracted by a CNN.
- We propose a continual learning approach based on the progressive training of multiple SVR models to estimate the aerial pose. This methodology uses features extracted using backbone networks such as DeepPilot4Pose [12] and ResNet18 [13].
- To demonstrate the effectiveness of continual learning using a dataset of no more than 300 images, accelerating the training process with subsets to obtain aerial localization.

The rest of the paper is organized as follows. Section 2 presents related works on localization using learning methods; Section 3 details the methodology used for sub-map search and pose estimation. Experimental setup and results obtained are presented in Sec. 4. Conclusions and future work are presented in Sec. 5.

2. Related Work

Localization tasks have been conducted on deep learning (DL) methodologies, leveraging sensor data from cameras and LiDAR systems to determine the poses of UAVs across diverse scenarios, particularly in GPS-denied environments [8, 14]. The integration of DL techniques with Simultaneous Localization and Mapping (SLAM) systems has enabled high-frequency pose estimation within indoor settings, using Inertial Measurement Units (IMUs), point clouds, and imagery data [15]. This fusion of data sources achieves real-world UAV localization incorporating visual odometry (VO) techniques [16] and the refinement of global map consistency through correspondences in sparse point clouds [17].

PoseNet [5], a CNN-based approach, has been used to train models for estimating camera poses from images [18]. These networks leverage visual, spatial, and temporal cues within interaction modules for pose determination [19]. Likewise, CNNs have been applied to place recognition tasks, often in combination with satellite imagery [20]. More recently, advancements such as those in [21] propose CNN architectures that integrate past experiences with current iterations to improve localization in novel and unknown scenarios. This has enabled the exploration of new places or dynamic scenarios where visual information may change over time.

PoseNet has also been used for navigation and aerial localization with images captured by UAVs [3, 6]. In GPS-denied environments, integrating Kalman Filters has improved pose accuracy [1]. However, CNN-based localization systems face a trade-off between accuracy and computational efficiency, leading to long processing times and high resource demands. To address these challenges, recent works [7, 12] have explored compact PoseNet architectures to enable UAV pose estimation with smaller datasets and faster inference. Yet, real-time localization during flight missions remains an open challenge for CNN-based approaches.

An alternative approach to CNNs is to use a Support Vector Machine (SVM) to determine the localization without an extensive training dataset. An example of this is presented in [22], where they used an SVM to classify the position of the human body using eigenvalues. Feature vectors with SVM can be helpful for object localization and state estimation, as shown in [23]. Besides, SVMs have been applied in tasks such as human body position [24] and object detection with overlapping regions in images [25].

In contrast, some works have utilized SVM for place recognition [26] and beacon localization using Received Signal Strength (RSS) measurements via Radio Frequency (RF) on UAVs [27]. Support Vector Regression (SVR), an alternative to SVM, estimates the localization of mobile targets in indoor environments [28]. Thus, SVR-based techniques are suitable for autonomous navigation tasks on optimized paths [29]. However, SVR requires continuous training to use a model during a mission. To address this challenge, [30] introduced an incremental training approach that divides the dataset and uses support vectors to extend learning.

Continual learning (CL) is a training method to acquire knowledge without losing past information. For instance, [31, 32] employ discriminative learning to extend labeling instances and utilize similarity between samples and features. Other works such as [9, 33] use regularization strategies to modify loss and update it with parameters through small models in a dynamic architecture. Another strategy, known as latent data replay, creates an internal memory in a neural network to store previous information [10]. This strategy is employed for class increment via Knowledge Distillation (KD) [34], dual models [35], and for depth estimation using experiences from a source and destination [36].

Some work, such as [37, 38], employ CL strategies like KD for grasp and object detection in autonomous driving. On the other hand, [39,40] used these methods for identification and following tasks in service robots. In these contexts, CL trains robots to acquire knowledge of the environment using limited datasets and selecting historical keyframes that act as experiences retained in the learning process. In addition, CL methods, in combination with neural graphs and latent replay, allow place recognition and localization in NeRF systems [41], and using KD with LiDAR [42]. These methods used input parameters from new instances to update the learning model. For example, [43] presents an evolution of knowledge between current and historical identity knowledge, incrementing training and updating the model with new information. Even with the use of interpolation and extrapolation of features, it enables the improvement of map density with enhanced positions for localization [44].

Replay-based strategies have been applied to SLAM systems in unknown environments using UAVs [45]. iMAP [46] enhances the associated 3D map with color and depth by storing keyframe structures in memory, allowing pose determination in the region. In [47], a multicamera system is trained with multiple images continuously using latent replay. These strategies help preserve prior information, enabling the continuous learning of new data, including experiences for camera localization [48]. This approach has proven effective for pose estimation with monocular cameras. However, continuous dictionaries of consecutive images and 3D camera geometry are required for SLAM systems. A recent work in CL with SLAM, [49], retains information from visited places and predicts scale trajectories using videos and loop closures. Despite the use of latent memories, a dual-network SLAM system is still needed to obtain the camera pose.

The most significant advancements in CL and aerial localization with monocular cameras are detailed in [50], where existing methodologies for pose estimation from an image are outlined. Among the novel works, [51] presents an approach based on VO that explores experiences stored in latent memory. This method selects significant weights and transfers them to generalized knowledge, enabling the update of the previously trained model. In this way, camera localization is achieved in various scenarios using a triplet of images in combination with dual networks and replay memory. Finally, focusing on aerial localization with UAVs, a methodology based on sub-mapping and topological geo-localization using dual networks is presented in [11,52]. The works aim to acquire localization from an image, seeking the map corresponding to the current image and thus classifying the nearest mean pose to the sub-map.

Therefore, we employ CL in a sub-mapping approach using a traditional learning method, such as SVR. This can be leveraged to establish the localization of a UAV in case a GPS signal is lost and to estimate the pose inside of the scenario instead of classifying the nearest mean pose.

3. Methodology

The main idea of our work is the aerial localization of a UAV by implementing a CL learning method that integrates multiple SVR models alongside a sub-mapping search scheme. Our approach is structured into identifying the corresponding sub-map and estimating the UAV pose. The first part of our methodology searches the sub-maps through continual training of representative keyframes. Afterward, in the second part, we employ continual training of aerial images paired with their respective GPS coordinate labels, creating multiple SVR models corresponding to a sub-map identified in the initial step. Therefore, our proposed methodology aims to localize a UAV within a specific sub-map and estimate its pose using multiple SVR models.

3.1. *Dataset acquisition*

For data acquisition, we used the Matrice 100 drone integrated with the Robot Operating System (ROS) to establish communication between the ground control station (GCS) and the onboard UAV computer. This configuration enabled real-time acquisition of aerial images during flight missions. Four different flight missions were executed, each corresponding to individual scenarios with lengths of 1.0 km for trajectory 1, 3.0 km for trajectory 2, 4.6 km for trajectory 3, and 6 km for trajectory 4, making it the longest mission conducted over the designated area. We captured aerial images at an altitude of 50 metres with a 1280 × 720 resolution, where each captured image was tagged with its corresponding GPS coordinates.

Figure 2 depicts a representative trajectory followed by the UAV during a flight mission, delineated into sections referred to as sub-maps. This enables targeted training across multiple models, with each model focusing on the data to its designated sub-map. Two datasets of training were collected to facilitate this approach: aerial images with GPS coordinates and altitude as labels and keyframes annotated with the respective sub-map indexes. In this way, we used the images with GPS information to train SVR models for the pose estimation process. Additionally, we stored keyframes for training a classification CNN, which aids in associating each image with its corresponding sub-map within the trajectory. This dual-training strategy ensures the learning of both pose and sub-maps for UAV localization.

The generation of each sub-map occurs as the UAV moves between 50 to 100 m from its preceding location, resulting in an average of 5 to 20 poses captured within each sub-map. This process is derived from the concept of the SLAM system, wherein incremental data acquisition enables camera localization within each mapped section. Ten sub-maps were delineated for each of the four trajectories, with approximately 30 aerial images allocated for training purposes per sub-map. This process yielded a set of 3 SVR models representing *x*, *y*, and *z* coordinates alongside a classification model tasked with determining the appropriate sub-map for a given image.

Table 1. Summary of captured images for training, testing, and keyframes stored for each sub-map. Aerial images include GPS coordinates, while keyframes are indexed according to sub-map number.

	Aerial images		Keyframe storage	
Traj.	Training	Testing	1 Sub-map	10 Sub-maps
1	282	144	3	30
2	188	117	3	30
3	236	84	3	30
4	7826	607	5	50

We resized the images and keyframes to 224 × 224 for the training process, with GPS coordinates converted to metres. The dataset employed is the same detailed in [11], but we used all images and associated pose labels for estimation purposes rather than topological localization based on mean poses. Furthermore, to evaluate the efficacy of our approach, we constructed test trajectories similar to training trajectories. In Table 1, we show the information about the dataset generated for each trajectory with the number of images for training, testing, sub-maps and keyframe.

It should be noted that the dataset is not continuous due to GPS loss in certain capture areas, resulting in fewer than 300 samples in the first three trajectories. In contrast, trajectory 4 consists of a continuous dataset comprising over 7,000 images with constant coordinates. This dataset was collected to demonstrate the continual learning process involving the subdivision of a large dataset into subsets.

3.2. *Support Vector Regression* (*SVR*)

SVM and SVR are machine-learning tools adapted to different problems. SVMs specialize in classifying training samples into discrete classes, whereas SVRs fit functions to predict continuous numerical values used in regression tasks. Consequently, the training and optimization procedures for SVMs and SVRs diverge. While SVRs are optimized for unidimensional regression tasks, SVMs are trained to tackle multi-class classification.

Our methodology focuses on adapting SVRs for UAV pose estimation. To achieve this, we preprocess aerial images to extract features such as borders, textures, colors, shapes

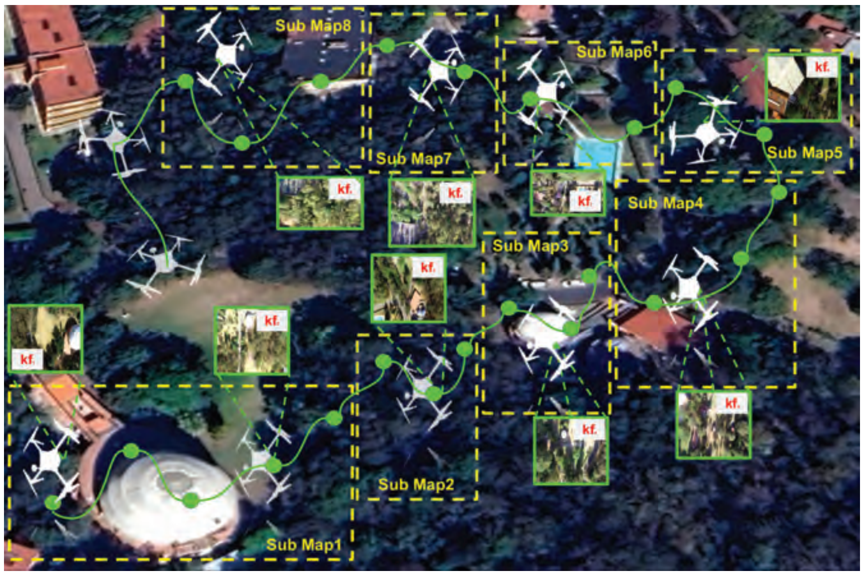


Fig. 2. General diagram for sub-map creation and keyframe storage. The flight trajectory is shown in green, green circles represent GPS coordinates, and keyframes in green outlines are images with information on the index of the sub-maps.

and patterns using a CNN as a feature extractor. Given the uni-dimensional nature of the SVR model, we employ a strategy wherein 3 SVR models are generated for each sub-map. Each of these models is dedicated to predicting one of the coordinates (x, y, z), simplifying the regression and enhancing the interpretability of the model.

We divided the dataset into sub-maps, aligning with the trajectory of the UAV during the training process. This strategy enables efficient data processing and management in large environments. Thus, we embrace a multi-model approach by training SVR models for individual sub-maps, enabling UAV localization in dynamic outdoor scenarios. This process is inspired by the concept of SLAM systems, where a map is constructed through sub-maps while simultaneously obtaining localization.

The selection of SVR stems from its advantageous features, such as low-processing training and the capacity to encapsulate information within support vectors. Moreover, SVRs minimize error within a margin, rendering them robust to noise and outliers within the dataset. In this context, we argue that UAV pose estimation using images based on SVR models can offer a promising approach due to managing high-dimensional data and capturing nonlinear relationships between input features and output variables.

Hence, we trained multiple SVR models using the following specifications: an RBF kernel with $C = 100$ and feature vectors extracted from the DeepPilot4pose [12] and ResNet18 [13] networks. The labels correspond to the coordinates (x, y, z) and are paired with each image feature vector. Table 2 provides an overview of the number of images used for each subset corresponding to every sub-map and model size expressed in bytes. These model sizes are lighter than pose estimation models such as PoseNet [5], which typically range between 300 to 700 Megabytes.

Table 2. Images used for training each sub-map and model size.

Traj.	Images	Model Size
1	29	25–240 kB
2	19	34–157 kB
3	24	58–198 kB
4	783	1.1–6.1 MB

3.3. *Continual multi-model learning*

We employed two continual training processes to achieve UAV localization and pose estimation from aerial images: (1) Training multiple SVR models with x, y, and z poses as labels; (2) Training InceptionV4 with keyframes extracted from the sub-maps with indexes as labels. In this way, both processes are trained separately as the UAV progresses during a flight.

We employ a Rehearsal Strategy based on progressive learning to train SVR models. This strategy involves creating new models with additional information from the pose using GPS coordinates. We train three SVR models for each sub-map generated corresponding to the x, y, and z coordinates using the feature vector extracted by DeepPilot4pose or ResNet18. Simultaneously, we store keyframes per each sub-map created for the second training step. The training complexity of SVR models is generally $O(n_s^3)$, where n_s is the number of support vectors for each SVR. For k sub-maps, each with three SVR models, this gives a total complexity of $O(k \cdot 3 \cdot n_s^3)$.

In the second training phase, we implemented a latent replay strategy using an external memory to retain past data. This approach allows the previous information to merge with the new data, preserving knowledge over time. For that, we used a sample vector of dimension 10, with each sample representing a different sub-map. The second training consists of extracting features from keyframes using InceptionV4 and storing them in the sample vector. Besides, InceptionV4 has a complexity of $O(n_f \cdot d_v)$, where n_f is the number of floating-point operations, and d_v is the network depth for InceptionV4. Afterward, we extract feature vectors from aerial images and compare them with the stored samples. If the extracted features match those from nearby keyframes, the CNN returns the index corresponding to the sub-map to which the image belongs.

In Fig. 3, we illustrate the diagram of our continual training methodology, where subsets are fed into a backbone network to extract features from aerial images. Hence, we implement two training processes that work separately. First, we train SVR models with features extracted using ResNet18 or DeepPilot4Pose while creating sub-maps and storing the keyframes. Then, we train InceptionV4 with these keyframes, acquiring features in a vector for a sub-map search in the test step. For backbone networks, we used DeepPilot4pose [12], a variant of PoseNet [5] chosen for its distilled architecture, producing a vector of 1,024 size features in its output. In addition, we used ResNet18 [13] due to its high-quality feature extraction compared to other CNNs, with a vector of 512 features.

Finally, we conducted the training processes using a computer equipped with CUDA 11.1, PyTorch 1.9, 8GB of RAM, and a GeForce 960M Nvidia card. Besides, we implemented progressive learning and latent replay to preserve previous data. Our methodology works as a localization and pose estimation system, where the sub-map containing the UAV's image is identified. This process is helpful in environments where traditional SLAM systems face challenges due to the need for features or their pattern repeatability. Our approach is the first focus in pose estimation and UAV localization using continual learning with aerial images, which is suitable for flight missions.

4. Experiments

We conducted two experiments to evaluate the effectiveness of our approach in localizing UAVs using aerial images. The first experiment focused on identifying the sub-map corresponding to a given input image by employing the InceptionV4 network. In the second experiment, we aimed to estimate the UAV's pose from a single aerial image by identifying the relevant sub-map and loading the corresponding SVR model to compute the localization. These experiments were carried out across four trajectories, each distinct from the training path. Additionally, we compared our results with PoseNet, ORB-SLAM2, and a Neural Network (NN). For comparison, we tested both our multi-model approach and single-model versions of SVR and NN, referred to as SVR-Single and NN-Single.

Table 3. Sub-map search accuracy results with the test dataset using color histograms and InceptionV4.

		Histogram Color		InceptionV4	
Traj.	Testing	Kfs Found	Acc.	Kfs Found	Acc.
1	144	87	0.60	117	0.81
2	117	65	0.55	84	0.71
3	84	56	0.66	60	0.71
4	607	310	0.51	467	0.76

4.1. *Sub-map results*

For the first experiment, we searched the sub-map using aerial images from the test dataset. The experiment involves finding the index corresponding to the nearest sub-map to the aerial image. To achieve this, the InceptionV4 network extracts maximum and minimum features from the image, which consist of a value range of each feature vector dimension in that corresponding sub-map. We use this range of features to compare the features extracted from the current image with those stored in the sample vector mentioned in the continual learning section. Thus, if the image matches the corresponding sub-map, we consider it correct; otherwise, it will be incorrect.

We performed a comparison with a search method using color histograms, extracting features from both the aerial image and keyframes to compare them and identify the corresponding sub-map. The results of this comparison are presented in Table 3, showing that InceptionV4 outperformed the color histogram method by successfully identifying the sub-maps corresponding to the aerial images. However, because the feature extractor with a color histogram is a traditional method, we decided to do the sub-map search using known neural networks. In Table 4, we present the search results using the MobileNetV2 and ResNet18 networks, where we can observe that it identifies more corresponding sub-maps concerning the keyframe than using the color histogram.

We argue that using CNN as a search system can offer advantages due to its capability to extract high-quality features, resulting in the best search performance. Also, using MobileNetV2 or ResNet18 as feature extractors gives us an advantage in finding sub-maps for UAV localization. Nonetheless, InceptionV4 extracts minimum and maximum features, which provides more features to search for than the features of the other methods. The results obtained with

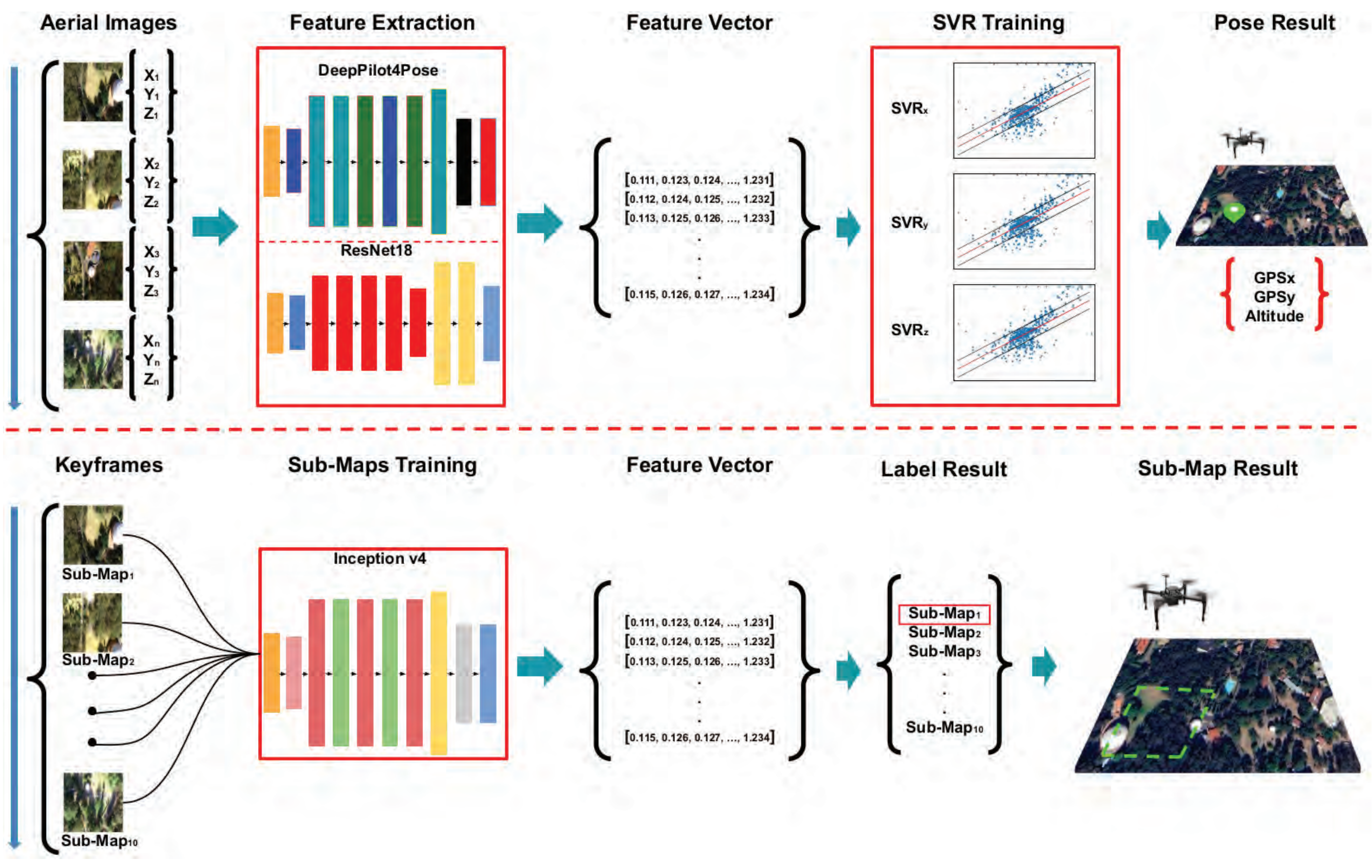


Fig. 3. Our localization methodology consists of two parts. First, we use a CNN to extract features from aerial images. These features, along with pose labels, are then used to train multiple SVR models. Second, we train the InceptionV4 network using keyframes generated during the dataset acquisition and initial training. This enables the UAV to identify the sub-map and estimate its pose relative to the ground truth, thus achieving aerial localization.

Table 4. Sub-map search accuracy results with the test dataset using MobileNetV2 and ResNet18.

		MobileNetV2		ResNet18	
Traj.	Testing	Kfs Found	Acc.	Kfs Found	Acc.
1	144	96	0.66	104	0.75
2	117	71	0.60	77	0.65
3	84	55	0.65	62	0.73
4	607	381	0.62	427	0.70

InceptionV4 will help search for the sub-map and load the SVR models for aerial localization.

4.2. *Aerial localization results*

For the second experiment, we performed UAV localization using aerial test images. In this process, the images pass through our system, where we initially extract maximum and minimum features using the InceptionV4 network. We compared these extracted features with the features stored in the sample vector, which contains information from representative keyframes of the sub-maps. Thus, if the features extracted from the aerial image match with a keyframe, the output class will indicate the corresponding sub-map. Following identifying the corresponding sub-map, we loaded the SVR models into a loading vector indexed as SVR_0 to SVR_9. Each index in this vector holds the three SVR models for the sub-map representing the x, y, and z coordinates.

This initial process involves loading the corresponding models for the identified sub-map based on the index found in the first part of the evaluation. Subsequently, once the three SVR models are loaded, the image passes through either DeepPilot4Pose or ResNet18, extracting 1024 or 512 features, respectively. These features are evaluated with the SVR models, thereby estimating the pose of the UAV using the aerial image. This process is repeated continuously as each image from the test set arrives. Figure 4 illustrates the experimental setup for localization, depicting the stages of sub-map search, model loading, and pose estimation.

For comparison, aerial localization is implemented with a 4-layer neural network (NN) using a single-model and multi-model approach. We also employed PoseNet and ORB-SLAM2 methods to localize a single aerial image. Likewise, we evaluated our approach to SVR-multiple models, applying

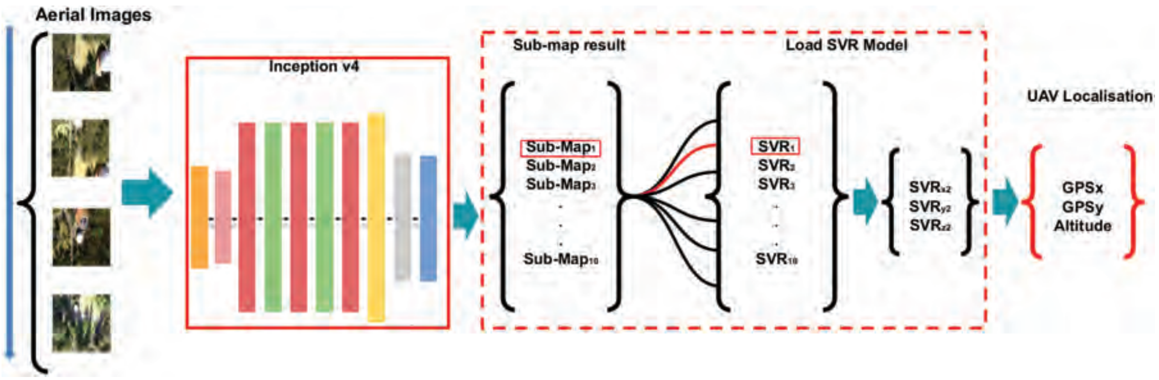


Fig. 4. General diagram for aerial localization. Initially, the aerial image passes through the InceptionV4 network to extract features. These features are used for sub-map search, where the identified sub-map (index) is obtained. Subsequently, the corresponding SVR model is loaded based on this index. Finally, the loaded SVR model is used for pose estimation.

Table 5. Mean Error Euclidean Distances in metres using DeepPilot4Pose as a feature extractor. The best results are highlighted in bold.

	Trajectory 1			Trajectory 2			Trajectory 3			Trajectory 4		
Approach	*x*	*y*	*z*	*x*	*y*	*z*	*x*	*y*	*z*	*x*	*y*	*z*
PoseNet	47.4	15.3	5.07	94.1	82.3	12.7	59.9	38.3	19.0	146.1	130.8	2.09
ORB-SLAM2	Breaks	Breaks	Breaks	**13.04**	**10.68**	6.90	Breaks	Breaks	Breaks	Breaks	Breaks	Breaks
NN-Single	48.0	18.8	11.3	104.0	93.7	18.0	54.2	45.5	11.2	158.5	136.6	16.8
NN-Multiple	46.4	26.7	23.5	82.51	100.8	30.9	77.2	32.3	25.8	94.86	87.51	36.1
SVR-Single	43.0	15.9	0.16	101.4	85.8	0.16	57.9	52.1	0.21	143.2	130.3	0.27
SVR-Multiple	**5.85**	**3.41**	**0.11**	16.77	19.1	**0.050**	**14.6**	**12.1**	**0.05**	**27.1**	**29.6**	**0.22**

the training with subsets and sub-map searching. In contrast, we implement an SVR-Single that trains the whole dataset without splitting it into mini-batches and without using sub-maps. Using SRV-Sigle means one model for *x*, one for *y*, and one for *z* without searching the corresponding model to the image representing the sub-map. In this way, we evaluated the approaches in terms of Mean Euclidean Distance Error in metres. In Table 5, we present the results obtained with these approaches across the four trajectories, demonstrating the effectiveness of our approach using multiple models and features extracted with DeepPilot4Pose.

Similarly, in Table 6, we present the error percentage results, where a higher percentage represents a larger error in estimation, indicating that the image did not obtain a correct localization or the sub-map was not found. The error percentage is calculated for each axis, where the *z*-axis shows the lowest values due to all flights being conducted at a consistent height of 50 m. In contrast, the percentages in the *x* and *y* axes show wider errors due to the fast training and the reduced number of training data. Despite this, our approach achieves lower errors than alternative approaches, even if the training data are small.

Finally, in Table 7, we present the total percentage error, which is the sum of the translation error across the entire length of the real trajectory. Our approach obtained a mean error of 7.08, which is a suitable result considering that we used fewer than 300 images and divided the dataset into multiple subsets without applying epochs or training cycles like conventional networks. In contrast, PoseNet obtained a higher error, which is coherent given the low number of training examples. ORB-SLAM2 achieved an error rate close to ours on trajectory 2; however, it could not complete the mapping on the other trajectories because the data are not continuous and have sharp turns.

In Fig. 5, we present the visual results of three approaches using features extracted with DeepPilot4Pose: PoseNet, a well-known network for pose estimation; our multi-model approach with a 4-layer NN; and our multi-model approach using SVR. Thus, we illustrate the real and estimated trajectories with these approaches, where PoseNet in column 1 exhibits several estimation jumps, resulting in erroneous localization results. In the second column, we present the multiple models with NN, obtaining suitable estimations at the beginning of the trajectories but with jumps in the final. Our approach in column 3 estimates the poses closer to the test trajectory, showing better performance than the others.

In Table 8, we present the mean results of the Euclidean distance error obtained using ResNet18 as the feature extractor, where our approach demonstrates a better performance than the others. Table 9 shows the error percentage per axis, obtaining lower error rates across all trajectories

Table 6. Percentage Error Axis (%) using DeepPilot4Pose as a feature extractor. The best results are highlighted in bold.

	Trajectory 1			Trajectory 2			Trajectory 3			Trajectory 4		
Approach	*x*	*y*	*z*	*x*	*y*	*z*	*x*	*y*	*z*	*x*	*y*	*z*
PoseNet	48.7	30.8	9.91	60.4	43.0	12.7	19.0	33.6	18.9	76.9	53.2	2.11
ORB-SLAM2	Breaks	Breaks	Breaks	16.6	**5.74**	7.56	Breaks	Breaks	Breaks	Breaks	Breaks	Breaks
NN-Single	49.3	37.8	22.4	66.8	49.0	17.9	17.1	39.9	11.1	83.4	55.5	16.7
NN-Multiple	47.7	53.7	46.6	52.9	52.7	30.8	24.4	28.4	25.7	49.9	35.5	35.9
SVR-Single	44.2	31.9	0.32	65.1	44.9	0.16	18.3	45.9	0.21	75.3	53.0	0.27
SVR-Multiple	**6.02**	**6.85**	**0.22**	**10.7**	9.99	**0.05**	**4.64**	**10.6**	**0.05**	**14.2**	**12.0**	**0.22**

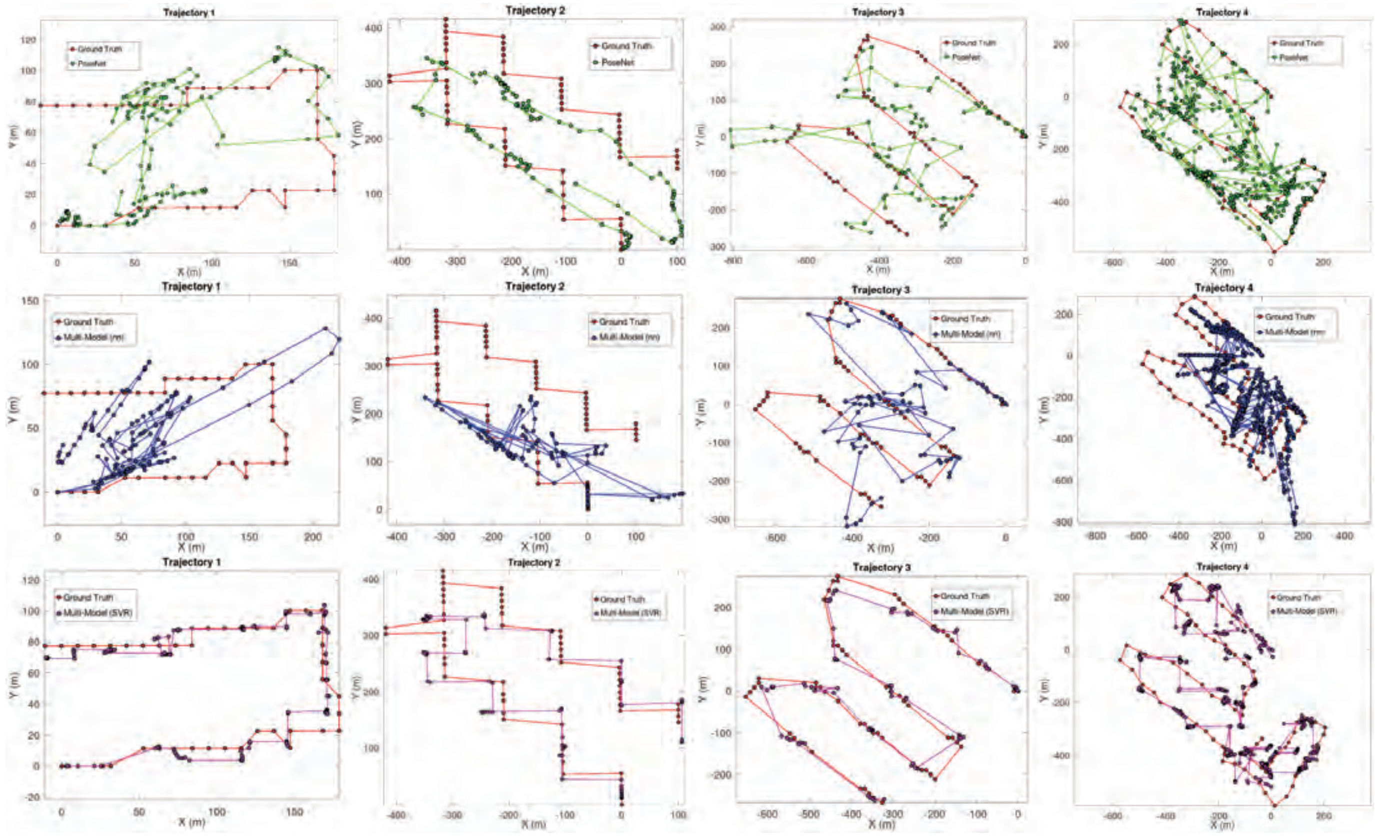


Fig. 5. Estimation results using DeepPilot4Pose as the backbone network for extracting features. We show the testing trajectory in red and circle the ground truth poses. Our multi-model SVR approach obtained better pose estimation results than the other.

Table 7. Total Percentage Error (%) using DeepPilot4Pose as a feature extractor. The best results are highlighted in bold.

Approach	Traj.1	Traj.2	Traj.3	Traj.4	Mean
PoseNet	34.3	42.2	22.1	52.0	37.6
ORB-SLAM2	Breaks	11.7	Breaks	Breaks	—
NN-Single	39.5	48.2	20.9	58.1	41.6
NN-Multiple	49.0	47.8	25.5	40.7	40.7
SVR-Single	29.9	41.9	20.8	51.0	35.9
SVR-Multiple	4.75	8.03	5.06	10.5	**7.08**

using our methodology. Unlike DeepPilot4Pose, ResNet18 provides more high-quality feature acquisition, benefiting both SVR and NN approaches. The features extracted by ResNet18 improve the training of both methods, reducing the error compared to DeepPilot4Pose. Table 10 displays the total percentage error, demonstrating a slight reduction in error compared to DeepPilot4Pose, with a mean error 7.02. Figure 6 illustrates the results obtained with PoseNet, multiple NNs, and multiple SVRs, highlighting the effectiveness of ResNet18 as a feature extractor.

ResNet18 demonstrates the ability to extract high-quality features suitable to training processes. Our analysis reveals that the multiple NN method improves estimations compared to features extracted using DeepPilot4Pose, resulting in poses closer to ground truth. Similarly, our

Table 8. Mean Error Euclidian Distances in metres using ResNet18 as a feature extractor. The best results are highlighted in bold.

	Trajectory 1			Trajectory 2			Trajectory 3			Trajectory 4		
Approach	*x*	*y*	*z*	*x*	*y*	*z*	*x*	*y*	*z*	*x*	*y*	*z*
PoseNet	47.4	15.3	5.07	94.1	82.3	12.7	59.9	38.3	19.0	146.1	130.8	2.09
ORB-SLAM2	Breaks	Breaks	Breaks	**13.04**	**10.68**	6.90	Breaks	Breaks	Breaks	Breaks	Breaks	Breaks
NN-Single	39.4	9.72	0.10	82.09	75.7	5.80	34.4	37.9	6.06	140.7	120.6	7.30
NN-Multiple	12.8	10.9	4.07	45.8	48.47	13.5	27.8	21.5	7.05	52.64	52.75	23.2
SVR-Single	35.3	9.59	0.18	87.9	74.0	0.20	28.1	39.7	0.22	141.2	119.2	0.27
SVR-Multiple	**6.12**	**3.85**	**0.10**	19.61	17.2	**0.04**	**13.6**	**10.9**	**0.04**	**25.22**	**29.35**	**0.23**

Table 9. Percentage Error Axis (%) using ResNet18 as a feature extractor. The best results are highlighted in bold.

	Trajectory 1			Trajectory 2			Trajectory 3			Trajectory 4		
Approach	*x*	*y*	*z*	*x*	*y*	*z*	*x*	*y*	*z*	*x*	*y*	*z*
PoseNet	48.7	30.8	9.91	60.4	43.0	12.7	19.0	33.6	18.9	76.9	53.2	2.11
ORB-SLAM2	Breaks	Breaks	Breaks	16.6	**5.74**	7.56	Breaks	Breaks	Breaks	Breaks	Breaks	Breaks
NN-Single	40.5	19.5	5.54	52.6	39.6	5.70	10.9	33.3	6.03	74.0	49.0	7.26
NN-Multiple	13.1	22.0	8.07	29.4	25.3	13.4	8.82	18.8	7.02	27.7	21.4	12.1
SVR-Single	36.3	19.2	0.36	56.4	38.9	0.20	8.91	34.9	0.22	74.3	48.5	0.24
SVR-Multiple	**6.29**	**7.75**	**0.21**	**12.5**	17.2	**0.04**	**4.34**	**9.58**	**0.04**	**13.2**	**11.9**	**0.04**

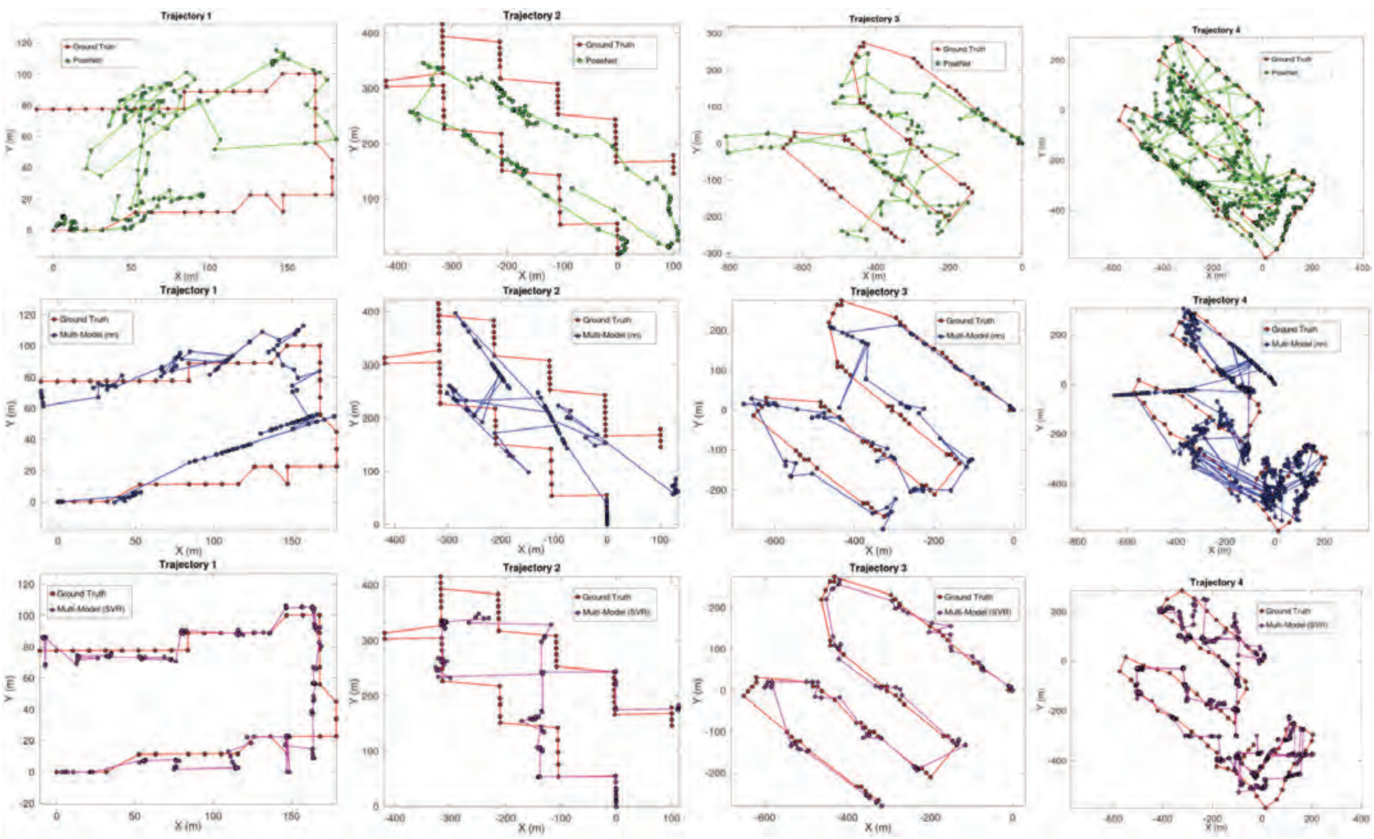


Fig. 6. Estimation results using ResNet18 as the backbone network for extracting features. We show the testing trajectory in red and circle the ground truth poses. Our multi-model SVR approach obtained better pose estimation results than the other methods.

Table 10. Total Percentage Error (%) using ResNet18 as a feature extractor. The best results are highlighted in bold.

Approach	Traj.1	Traj.2	Traj.3	Traj.4	Mean
PoseNet	34.3	42.2	22.1	52.0	37.6
ORB-SLAM2	Breaks	11.7	Breaks	Breaks	—
NN-Single	26.3	46.5	14.8	50.0	34.4
NN-Multiple	14.1	24.0	10.6	21.9	17.6
SVR-Single	22.8	36.3	12.8	48.6	30.1
SVR-Multiple	5.10	8.24	4.65	10.1	**7.02**

multiple SVR approach demonstrates a light improvement in estimation accuracy, achieving localization nearer to ground truth and maintaining consistency with the poses around the sub-maps where the images were acquired. We need to use a larger dataset with more aerial images to achieve smoother estimation results with fewer jumps.

Therefore, we evaluate the approaches in the fourth trajectory with 7 thousand images, where the multiple SVR tries to obtain poses close to the real path. Thus, our multiple SVR approach obtains slightly perceptible estimates while avoiding pose jumps along the UAV trajectory. We can also see that the poses are grouped around the sub-maps, resulting in smoother localiztion and fewer incorrect poses compared to PoseNet and the multiple NN models. In this way, we can use our methodology with a few or several images on long or small flight missions to acquire localization more quickly, serving as a helpful alternative localization system in case the GPS signal of the UAV is lost.

4.3. *Discussions*

To discuss the results obtained, we present Error Tables comparing our method with well-known methods such as ORB-SLAM2 and PoseNet. We noticed that PoseNet and ORB-SLAM2 present challenges to acquiring aerial localization. In the case of SLAM, the constrained number of images and the drastic scene changes pose significant difficulties. These abrupt changes often lead to feature tracking failures, which cause SLAM to stop generating the map properly. While SLAM may initially work correctly, breaking when encountering a sudden change, resulting in the test not being completed correctly.

In contrast, PoseNet, a popular network for pose estimation and localization from images, has demonstrated effectiveness across several works, including applications in aerial localization. However, achieving accurate pose estimations with PoseNet requires extensive training epochs and a large dataset to develop a robust model. While PoseNet and its variants, such as DeepPilot4Pose [12], have shown promising results, their reliance on extensive datasets limits their suitability for continual learning with small data usage.

On the other hand, our approach leverages SVR to enable continuous pose estimation through the concept of multiple models. Our approach involves a strategy of progressive learning to create multiple models with incoming information from aerial images inspired by SLAM systems. In this way, we divide the process into two distinct phases: sub-map search and pose estimation, thereby achieving localization. To achieve this, we employed two networks as feature extractors: DeepPilot4Pose [12] and ResNet18 [13], observing a slight performance improvement when we used ResNet18 as the feature extractor.

Table 11. Mean Feature Extraction Time (ms) using ORB-SLAM2, DeepPilot4Pose and ResNet18 as a feature extractor. The best result is highlighted in bold.

Extractor	Traj.1	Traj.2	Traj.3	Traj.4	Mean
ORB-SLAM2	77.17	70.94	72.97	74.90	73.99
DeepPilot4	54.55	55.10	55.65	58.07	55.84
ResNet18	43.35	39.91	41.65	40.09	**41.25**

In our first steps, our methodology holds promise for in-flight tasks that necessitate a reliable localization system, especially in scenarios where a UAV experiences a loss of GPS signal. By continuously learning from incoming information during flight, our approach can operate as a backup localization system during these occurrences, ensuring the UAV's safe positioning until the GPS signal is restored. Furthermore, we aim to validate the efficacy of our approach in real-time tasks, ensuring that it operates at an optimal speed to provide accurate pose estimation during flight.

Finally, to evaluate the efficiency of our approach, we measured time processing in milli seconds (ms), comparing two key elements: (1) We assessed the feature extraction time using ORB-SLAM2, DeepPilot4Pose and ResNet18, measuring how long the extraction process takes; (2) We analyzed the localization time, which contains the entire process of sub-map identification, pose estimation, and aerial localization, providing insights into the overall efficiency of the methodology.

We present the extraction time in ms in Table 11. It is worth noting that this analysis was carried out under the same conditions using ROS in a node to evaluate the images as video, allowing frame-by-frame processing for final localization. While ResNet18 and DeepPilot4Pose extract abstract features from the image using CNN, ORB-SLAM2 relies on an ORB detector to identify specific key-points. Although these methods differ in their approach to feature extraction, we included ORB-SLAM2 in the comparison to offer a reference point from a traditional keypoint-based approach. Our observations indicate that ResNet18 demonstrates a higher feature extraction speed than the DeepPilot4Pose and ORB-SLAM2. This was done using a computer with a non-current graphics card, resulting in slower extraction processing than if tested with a recent computer.

In Tables 12 and 13, we show the total time using the employed approaches. The single model approach with

Table 12. Total Processing Time (ms) using DeepPilot4Pose as a feature extractor. The best result is highlighted in bold.

Approach	Traj.1	Traj.2	Traj.3	Traj.4	Mean
ORB-SLAM2	Breaks	79.12	Breaks	Breaks	—
PoseNet	71.94	73.74	73.52	72.15	72.83
NN-Single	58.59	58.41	53.74	54.01	56.18
NN-Multiple	60.75	62.08	61.06	60.90	61.19
SVR-Single	49.92	51.13	49.90	51.10	**50.51**
SVR-Multiple	64.20	63.33	69.21	71.53	67.06

Table 13. Total Processing Time (ms) using ResNet18 as a feature extractor. The best result is highlighted in bold.

Approach	Traj.1	Traj.2	Traj.3	Traj.4	Mean
ORB-SLAM2	Breaks	79.12	Breaks	Breaks	—
PoseNet	71.94	73.74	73.52	72.15	72.83
NN-Single	56.01	50.25	45.22	49.82	50.32
NN-Multiple	61.45	61.43	64.60	64.18	62.91
SVR-Single	52.92	40.65	41.51	53.91	**47.24**
SVR-Multiple	70.83	64.37	64.63	75.15	68.74

NN and SVR exhibits faster processing time by handling a single model without sub-maps searching to get the aerial localization. Unlike the multi-model approach with NN and SVR, which requires loading models w.r.t. the sub-map index found. Besides, the one-dimensional nature of SVRs requires loading three models to obtain the poses separately for each sub-map when NN uses only one model. Despite this, our approach achieves good performance and a time not longer than other approaches, suitable for real-time localization tasks with non-continuous and non-large data. In contrast, PoseNet achieves the slowest processing time than the other approaches, while ORB-SLAM2 only finishes trajectory 2 with a processing time similar to PoseNet.

With these results, we argue that our proposed multi-model SVR approach offers several advantages over traditional algorithms through its progressive learning mechanism. Instead of using a single model trained on the entire dataset, our method trains multiple SVR models on mini-batches, which allows for better generalization and reduces the risk of overfitting. Moreover, the method retains and reuses previously learned features by incorporating latent replay, providing continuous learning. Traditional algorithms often lack this capability, whereas our multi-model system adapts dynamically, ensuring more accurate localization while maintaining computational efficiency.

5. Conclusions

We proposed a methodology for aerial localization based on pose estimation with multiple models. This approach entails the identification of sub-maps by comparing the features extracted from aerial images with those derived from representative keyframes. Our main contribution lies in implementing a multi-model SVR approach using a continual learning strategy. Thus, we trained the SVR models on a dataset of no more than 300 images divided into ten subsets. Each subset represents 3 SVR models for the x, y, and z coordinates, enabling UAV localization without requiring model retraining and relying on a large dataset.

To evaluate the effectiveness of our approach, we conducted comparative analyses with ORB-SLAM2, PoseNet, and a neural network with four layers. Besides, we trained a single model for both SVR and NN using the entire dataset without division. The evaluation results demonstrated that our approach consistently outperformed the others across all four trajectories, exhibiting a total percentage error around 7% for both, when using DeepPilot4Pose or ResNet18 as the feature extractor.

Finally, we evaluated the processing time of the methodology, showing a suitable result for localization purposes. Although the processing time with the multi-model approach is somewhat higher than the single model, the latter is not suitable for continual learning. In contrast, our methodology produces multiple models that can be available during the same operation flight, which is helpful for large-scale test scenarios prone to GPS signal loss, thus, acting as a backup localization system. As a future work, we will carry out the methodology on board a UAV to capture aerial images while training lightweight models in real-time during the flight mission.

Acknowledgments

The first author is thankful for his scholarship funded by Consejo Nacional de Humanidades, Ciencia y Tecnología (CONAHCYT) under grant 802791.

ORCID

Aldrich A. Cabrera-Ponce
https://orcid.org/0000-0002-9998-7444
Manuel I. Martin-Ortiz
https://orcid.org/0000-0002-4725-2059
Jose Martinez-Carranza
https://orcid.org/0000-0002-8914-1904

References

[1] D. Benjumea, A. Alcántara, A. Ramos, A. Torres-Gonzalez, P. Sánchez-Cuevas, J. Capitan, G. Heredia and A. Ollero, Localization system for lightweight unmanned aerial vehicles in inspection tasks, *Sensors* **21**(17) (2021) 5937.

[2] D. Bohnenkamp, J. Behmann and A.-K. Mahlein, In-field detection of yellow rust in wheat on the ground canopy and uav scale, *Remote Sens.* **11**(21) (2019) 2495.

[3] M. Müller, S. Urban and B. Jutzi, Squeezeposenet: Image based pose regression with small convolutional neural networks for real time uas navigation, *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **4** (2017) 49.

[4] R. Mur-Artal and J. D. Tardós, Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras, *IEEE Trans. Robot.* **33**(5) (2017) 1255–1262.

[5] A. Kendall, M. Grimes and R. Cipolla, Posenet: A convolutional network for real-time 6-dof camera relocalization, in *Proc. IEEE Int. Conf. Computer Vision*, (IEEE, 2015), pp. 2938–2946.

[6] A. A. Cabrera-Ponce and J. Martinez-Carranza, Aerial geo-localisation for mavs using posenet, *2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS)*, (IEEE, 2019), pp. 192–198.

[7] A. A. Cabrera-Ponce and J. Martinez-Carranza, Convolutional neural networks for geo-localisation with a single aerial image, *J. Real-Time Image Process.* **19**(3) (2022) 565–575.

[8] J. Qian, K. Chen, Q. Chen, Y. Yang, J. Zhang and S. Chen, Robust visual-lidar simultaneous localization and mapping system for uav, *IEEE Geosci. Remote Sens. Lett.* **19** (2021) 1–5.

[9] S. Auddy, J. Hollenstein, M. Saveriano, A. Rodríguez-Sánchez and J. Piater, Scalable and efficient continual learning from demonstration via hypernetwork-generated stable dynamics model, preprint (2023), arXiv:2311.03600.

[10] L. Pellegrini, G. Graffieti, V. Lomonaco and D. Maltoni, Latent replay for real-time continual learning, *2020 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, (IEEE, 2020), pp. 10203–10209.

[11] A. A. Cabrera-Ponce, M. Martin-Ortiz and J. Martinez-Carranza, Continual learning for topological geo-localisation, *J. Intell. Fuzzy Syst.* **44**(6) (2023) 10369–10381.

[12] L. O. Rojas-Perez and J. Martinez-Carranza, Deeppilot4pose: a fast pose localisation for mav indoor flight using the oak-d camera, *J. Real Time Image Process.* **20**(1) (2023) 8.

[13] K. He, X. Zhang, S. Ren and J. Sun, Deep residual learning for image recognition, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, (IEEE, 2016), pp. 770–778.

[14] J. F. Chow, B. B. Kocer, J. Henawy, G. Seet, Z. Li, W. Y. Yau and M. Pratama, Toward underground localization: Lidar inertial odometry enabled aerial robot navigation, preprint (2019), arXiv:1910.13085.

[15] J.-C. Yang, C.-J. Lin, B.-Y. You, Y.-L. Yan and T.-H. Cheng, Rtlio: Real-time lidar-inertial odometry and mapping for uavs, *Sensors* **21**(12) (2021) 3955.

[16] J. Bednář, M. Petrlík, K. C. T. Vivaldini and M. Saska, Deployment of reliable visual inertial odometry approaches for unmanned aerial vehicles in real-world environment, *2022 Int. Conf. Unmanned Aircraft Systems (ICUAS)*, (IEEE, 2022), pp. 167–176.

[17] Y. Pan, X. Zhong, L. Wiesmann, T. Posewsky, J. Behley and C. Stachniss, Pin-slam: Lidar slam using a point-based implicit neural representation for achieving global map consistency, preprint, arXiv:2401.09101 (2024).

[18] B. Wattanacheep and O. Chitsobhuk, Camera pose estimation using cnn, in *Proc. 3rd Int. Conf. Control and Computer Vision*, (2020), pp. 84–88.

[19] C. Qiao, Z. Xiang, Y. Fan, T. Bai, X. Zhao and J. Fu, Transapr: Absolute camera pose regression with spatial and temporal attention, *IEEE Robot. Autom. Lett.* **8** (2023) 4633–4640.

[20] Y. He, I. Cisneros, N. Keetha, J. Patrikar, Z. Ye, I. Higgins, Y. Hu, P. Kapoor and S. Scherer, Foundloc: Vision-based onboard aerial localization in the wild, preprint, arXiv:2310.16299 (2023).

[21] A. Saviolo, J. Frey, A. Rathod, M. Diehl and G. Loianno, Active learning of discrete-time dynamics for uncertainty-aware model predictive control, *IEEE Trans. Robot.* **40** (2023) 1273–1291.

[22] E. Ardizzone, A. Chella and R. Pirrone, Pose classification using support vector machines, in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, Vol. 6 (IEEE, 2000), pp. 317–322.

[23] C. Ionescu, L. Bo and C. Sminchisescu, Structural svm for visual localization and continuous state estimation, *2009 IEEE 12th Int. Conf. Computer Vision*, (IEEE, 2009), pp. 1157–1164.

[24] K. Chen, S. Gong and T. Xiang, Human pose estimation using structural support vector machines, *2011 IEEE Int. Conf. Computer Vision Workshops (ICCV Workshops)*, (IEEE, 2011), pp. 846–851.

[25] J. R. Madhukaro and D. Rao, Svm-ga based a novel technique for the detection of the vehicle in an optimized overlapped multi-camera system, *Int. J. Intell. Syst. Appl. Eng.* **12**(1) (2024) 810–818.

[26] Y. Qiao, C. Cappelle and Y. Ruichek, Place recognition based visual localization using lbp feature and svm, *Advances in Artificial Intelligence and Its Applications: 14th Mexican Int. Conf. Artificial Intelligence, MICAI 2015, Cuernavaca, Morelos, Mexico, October 25–31, 2015, Proceedings, Part II 14*, (Springer, 2015), pp. 393–404.

[27] R. Lima, K. Das and D. Ghose, Support vector regression based sensor localization using uav, in *Proc. 34th ACM/SIGAPP Symp. Applied Computing* (2019), pp. 938–945.

[28] S. R. Jondhale, V. Mohan, B. B. Sharma, J. Lloret and S. V. Athawale, Support vector regression for mobile target localization in indoor environments, *Sensors* **22**(1) (2022) 358.

[29] M. Wang, J. Xu, J. Zhang and Y. Cui, An autonomous navigation method for orchard rows based on a combination of an improved a-star algorithm and svr, *Precis. Agric.* (2024) 1–25.

[30] N. A. Syed, H. Liu and K. K. Sung, Handling concept drifts in incremental learning with support vector machines, in *Proc. fifth ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, (IEEE, 1999), pp. 317–321.

[31] M. Kilickaya and J. Vanschoren, Are labels needed for incremental instance learning?, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, (IEEE, 2023), pp. 2400–2408.

[32] U. Michieli and M. Ozay, Online continual learning for robust indoor object recognition, *2023 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, (IEEE, 2023), pp. 3849–3856.

[33] Y. Shao, Y. Guo, D. Zhao and B. Liu, Class-incremental learning based on label generation, preprint, arXiv:2306.12619 (2023).

[34] E. Camuffo and S. Milani, Continual learning for lidar semantic segmentation: Class-incremental and coarse-to-fine strategies on sparse data, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, (IEEE, 2023), pp. 2447–2456.

[35] H. Chawla, A. Varma, E. Arani and B. Zonooz, Continual learning of unsupervised monocular depth from videos, in *Proc. IEEE/CVF Winter Conf. Applications of Computer Vision*, (IEEE, 2024), pp. 8419–8429.

[36] N. Vödisch, K. Petek, W. Burgard and A. Valada, Codeps: Online continual learning for depth estimation and panoptic segmentation, preprint (2023), arXiv:2303.10147.

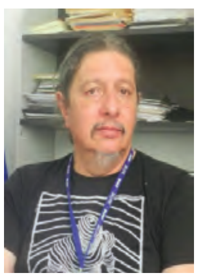
[37] Z. Wang, G. Cao, X. Xi and J. Wang, Opennet: Incremental learning for autonomous driving object detection with balanced loss, *2023 IEEE Int. Conf. Systems, Man, and Cybernetics (SMC)*, (IEEE, 2023), pp. 2675–2682.

[38] R. Yang, M. Grard, E. Dellandréa and L. Chen, When continual learning meets robotic grasp detection: a novel benchmark on the jacquard dataset, *18th Int. Conf. Computer Vision Theory and Applications (VISAPP)*, (SCITEPRESS-Science and Technology Publications, 2023).

[39] A. Ayub, C. Nehaniv and K. Dautenhahn, Interactive continual learning architecture for long-term personalization of home service robots, preprint (2024), arXiv:2403.03462.

[40] H. Ye, J. Zhao, Y. Zhan, W. Chen, L. He and H. Zhang, Person re-identification for robot person following with online continual learning, preprint (2023), arXiv:2309.11727.
[41] Z. Cai and M. Müller, Clnerf: Continual learning meets nerf, in *Proc. IEEE/CVF Int. Conf. Computer Vision*, (IEEE, 2023), pp. 23185–23194.
[42] J. Cui and X. Chen, Ccl: Continual contrastive learning for lidar place recognition, *IEEE Robot. Autom. Lett.* **8** (2023) 4433–4440.
[43] H. Yao, L. Yu, J. Luo and C. Xu, Camera-incremental object re-identification with identity knowledge evolution, e-prints (2023), arXiv–2305.
[44] M. Zaffar, L. Nan and J. F. P. Kooij, Copr: Toward accurate visual localization with continuous place-descriptor regression, *IEEE Trans. Robot.* **39** (2023) 2825–2841.
[45] A. Safa, T. Verbelen, I. Ocket, A. Bourdoux, H. Sahli, F. Catthoor and G. Gielen, Learning to slam on the fly in unknown environments: A continual learning approach for drones in visually ambiguous scenes, preprint (2022), arXiv:2208.12997.
[46] E. Sucar, S. Liu, J. Ortiz and A. J. Davison, imap: Implicit mapping and positioning in real-time, in *Proc. IEEE/CVF Int. Conf. Computer Vision*, (2021), pp. 6229–6238.
[47] A. A. Cabrera-Ponce, M. Martin-Ortiz and J. Martínez-Carranza, Continual learning for multi-camera relocalisation, *Mexican Int. Conf. Artificial Intelligence* (Springer, 2021), pp. 289–302.
[48] S. Wang, Z. Laskar, I. Melekhov, X. Li and J. Kannala, Continual learning for image-based camera localization, in *Proc. IEEE/CVF Int. Conf. Computer Vision*, (IEEE, 2021), pp. 3252–3262.
[49] N. Vödisch, D. Cattaneo, W. Burgard and A. Valada, Continual slam: Beyond lifelong simultaneous localization and mapping through continual learning, preprint (2022), arXiv:2203.01578.
[50] A. A. Cabrera-Ponce, M.-O. Manuel and J. Martinez-Carranza, Continual learning for camera localization, in *Machine Learning for Complex and Unmanned Systems* (CRC Press, Boca Raton, 2024), pp. 14–33.
[51] N. Vödisch, D. Cattaneo, W. Burgard and A. Valada, Covio: Online continual learning for visual-inertial odometry, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, (IEEE, 2023), pp. 2464–2473.
[52] A. A. Cabrera-Ponce, M. I. Martin-Ortiz and J. Martinez-Carranza, Hierarchical continual learning for single image aerial localisation, in *14th Annual International Micro Air Vehicle Conference and Competition*, ed. D. Moormann, Aachen, Germany (2023), pp. 40–48.

Aldrich A. Cabrera-Ponce is a PhD student at the Computer Science Department of the Benemrita Universidad Autnoma de Puebla. He obtained his Master of Science degree in Computer Science at the Instituto Nacional de Astrofsica ptica y Electrnica in 2021. His research interests include robotics and artificial intelligence, and he is currently working on aerial localization using continual learning with aerial images.



Manuel I. Martin-Ortiz is Full-time Principal Researcher C in the Computer Science Department at the Benemrita Universidad Autnoma de Puebla. In 2015, he was named Academic Coordinator of the National Supercomputing Laboratory of Southeast Mexico. His main research topic is digital image, parallel and distributed processing, for which he develops applications in specialized systems for natural and exact sciences.

Jose Martinez-Carranza holds the position of Senior Researcher C (Equivalent to Full Professor) in the Computer Sciences Department at INAOE. In 2015, he received the Royal Society-Newton Advanced Fellowship distinction, awarded by the Newton Fund. He is the founder and leader of the Unmanned Aerial Systems research group, and his team, QuetzalC++, has participated in highly prestigious international competitions, winning various awards, notably the 1st Place in the IROS 2017 Autonomous Drone Racing competition and the 1st Place Regional (Latin America) in the OpenCV AI Competition 2021. He served as the General Chair of the 12th International Micro Air Vehicle Conference (IMAV) 2021 and as Associate Editor for the IEEE IROS international conference in 2022–2023. He has published 150 articles in scientific journals, book chapters, and international conference proceedings. He also edited the book: Machine Learning for Complex and Unmanned Systems. His articles have received awards for the best paper at the IBERAMIA 2022 conference, finalist for the best paper at IMAV 2019 and 2024, and third place for the best paper at the MICAI 2018 conference. He is an IEEE Senior Member and Level 2 in the National Researchers System of Mexico.