drones

MDPI

Article

UAV Localization in Urban Area Mobility Environment Based on Monocular VSLAM with Deep Learning

Mutagisha Norbelt [1,2,†], Xiling Luo [1,2,*,†], Jinping Sun [1,2] and Uwimana Claude [3]

[1] School of Electronic and Information Engineering, Beihang University, Beijing 100191, China; prsultan05@buaa.edu.cn (M.N.); sunjinping@buaa.edu.cn (J.S.)
[2] Hangzhou Innovation Institute, Beihang University, Hangzhou 310052, China
[3] School of Computer Science, Beihang University, Beijing 100191, China; claudeuwi@buaa.edu.cn
* Correspondence: luoxiling@buaa.edu.cn; Tel.: +86-1326-002-3044
† These authors contributed equally to this work.

Abstract: Unmanned Aerial Vehicles (UAVs) play a major role in different applications, including surveillance, mapping, and disaster relief, particularly in urban environments. This paper presents a comprehensive framework for UAV localization in outdoor environments using monocular ORB-SLAM3 integrated with optical flow and YOLOv5 for enhanced performance. The proposed system addresses the challenges of accurate localization in dynamic outdoor environments where traditional GPS methods may falter. By leveraging the capabilities of ORB-SLAM3, the UAV can effectively map its environment while simultaneously tracking its position using visual information from a single camera. The integration of optical flow techniques allows for accurate motion estimation between consecutive frames, which is critical for maintaining accurate localization amidst dynamic changes in the environment. YOLOv5 is a highly efficient model utilized for real-time object detection, enabling the system to identify and classify dynamic objects within the UAV's field of view. This dual approach of using both optical flow and deep learning enhances the robustness of the localization process by filtering out dynamic features that could otherwise cause mapping errors. Experimental results show that the combination of monocular ORB-SLAM3, optical flow, and YOLOv5 significantly improves localization accuracy and reduces trajectory errors compared to traditional methods. In terms of absolute trajectory error and average tracking time, the suggested approach performs better than ORB-SLAM3 and DynaSLAM. For real-time SLAM applications in dynamic situations, our technique is especially well-suited due to its potential to achieve lower latency and greater accuracy. These improvements guarantee more dependable performance in a variety of scenarios in addition to increasing overall efficiency. The framework effectively distinguishes between static and dynamic elements, allowing for more reliable map construction and navigation. The results show that our proposed method (U-SLAM) produces a considerable decrease of up to 43.47% in APE and 26.47% RPE in S000, and its accuracy is higher for sequences with moving objects and more motion inside the image.

Keywords: UAV localization; monocular; visual SLAM; optical flow; YOLOv5

check for updates

Academic Editors: Jian Chen, Min Xia and Hui Xie

Received: 23 December 2024
Revised: 26 January 2025
Accepted: 26 January 2025
Published: 26 February 2025

Citation: Norbelt, M.; Luo, X.; Sun, J.; Claude, U. UAV Localization in Urban Area Mobility Environment Based on Monocular VSLAM with Deep Learning. *Drones* **2025**, *9*, 171. https://doi.org/10.3390/drones9030171

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The rapid and remarkable development of unmanned aerial vehicles (UAVs) has transformed various industries and applications and has a significant impact on a variety of industries, including agriculture, surveillance [1], and disaster management. As UAVs

become more prevalent in outdoor environments, the need for robust and reliable localization methods has emerged, especially in scenarios where traditional Global Positioning System (GPS) signals are weak or Not available. Simultaneous Localization and Mapping (SLAM) is a computational problem that arises when an autonomous system, such as a UAV, attempts to map an unknown environment while concurrently keeping track of its location within that environment. This dual-task challenge is crucial in scenarios where pre-existing maps are unavailable, which is often the case in urban and dynamic settings. In contrast, Odometry is a more specialized concept focused primarily on estimating the motion of a vehicle or robot over time. In the context of UAVs, odometry refers to the process of measuring how far and in what direction the UAV has moved since the last known position. Odometric measurements can be derived from various sources, including wheel encoders on ground vehicles and inertial measurement units (IMUs) on drones. This paper focuses on improving UAV localization by integrating monocular ORB-SLAM3, optical flow, and YOLOv5 to enhance the UAV's ability to navigate in complex and dynamic outdoor environments. Monocular SLAM (Simultaneous Localization and Mapping) systems, such as ORB-SLAM3 [2], Utilize a single camera to effectively track the UAV's precise location while simultaneously creating a detailed environment map at the same time, within that map. This method is particularly advantageous in unfamiliar environments where prior knowledge is lacking.

However, monocular SLAM faces challenges related to scale ambiguity and drift over time, especially in dynamic environments where environmental conditions can change rapidly. By employing optical flow techniques, our approach enhances the UAV's motion estimation capabilities, allowing it to accurately track changes between successive frames and maintain reliable localization. The integration of YOLOv5 [3], a cutting-edge, state-of-the-art deep learning model designed for highly efficient real-time object detection in dynamic environments, further enhances the UAV's situational awareness. YOLOv5 enables the system to recognize and categorize dynamic objects in its field of view, which is critical for avoiding obstacles and improving navigation accuracy. This integration allows the UAV to distinguish between static landmarks, which are essential for mapping, and moving objects, which can interfere with localization efforts. The combination of these technologies monocular ORB-SLAM3 for mapping and localization [4], optical flow for enhanced motion estimation [5], and YOLOv5 for object detection creates a comprehensive framework capable of overcoming the limitations of traditional SLAM systems in outdoor environments. The proposed method aims not only to enhance efficiency but also to improve the overall effectiveness of UAV operations in challenging conditions. Through experimental validation, this study will demonstrate that the integration of monocular ORB-SLAM3 with optical flow and YOLOv5 significantly improves localization performance compared to existing methods. The structure of the proposed approach is clearly depicted in Figure 1, illustrating the key components and their interrelationships Figure 1.

Our proposed approach is based on the ORB-SLAM3 framework; the data flow begins with the acquisition of RGB images. The YOLOv5 model is employed to detect and classify objects within these images. The detection process generates bounding boxes around identified objects, which are categorized into static and dynamic types based on their movement characteristics. The output from YOLOv5 includes class names and coordinates of detected objects, which are then utilized by the SLAM algorithm to differentiate between dynamic and static features in the scene. This classification helps in reducing the interference caused by moving objects during the mapping process. The feature points are extracted based on the classification results from YOLOv5. The algorithm identifies dynamic features that need to be eliminated to maintain mapping accuracy while retaining static features for pose estimation. The integration of optical flow techniques further enhances this process

by tracking the movement of feature points over time. Optical flow algorithms analyze pixel intensity changes between consecutive frames to estimate motion vectors, allowing for more precise matching of feature points across frames. This dual approach ensures that only relevant features contribute to the SLAM process, improving robustness against dynamic changes in the environment. The optimization phase in SLAM systems utilizing YOLOv5 involves potentially dynamic, if a feature point is classified as dynamic, it can be excluded from subsequent calculations to prevent inaccuracies in localization. Additionally, depth information plays a crucial role in this optimization. By integrating depth data with feature classification from YOLOv5, SLAM systems can refine their understanding of spatial relationships within the environment. This fusion of data allows for a more accurate representation of static backgrounds while effectively managing dynamic elements.

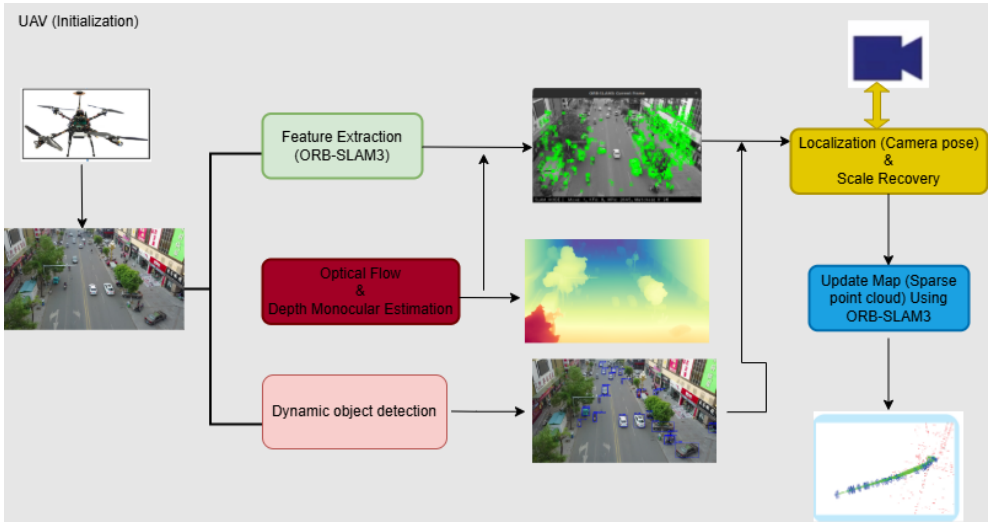


Figure 1. Diagram overview of UAV localization based on Vslam with deep learning.

The key significant contributions of the proposed approach are clearly outlined and highlighted below to tackle the previously mentioned challenges and enhance the precision and reliability of pose estimation techniques in visual SLAM performing in dynamic outdoor settings:

(1) The use of monocular ORB-SLAM3 allows UAVs to achieve accurate localization and mapping using only a single camera. This reduces hardware complexity and weight, making it ideal for various UAV applications in outdoor environments.
(2) Incorporating optical flow techniques improves the UAV's motion estimation, providing more reliable velocity information. This improvement is critical in dynamic situations where GPS signals may be poor, thereby increasing the overall localization accuracy.
(3) The integration of YOLOv5 facilitates real-time object detection, allowing UAVs to effectively identify and track obstacles and relevant landmarks. This spatial awareness is essential for safe navigation in complex outdoor environments.

Existing research pertinent to this problem is reviewed in the parts that follow, along with a description of the study's methodology and a detailed discussion of its findings. To assess the level of improvement brought about by the suggested approaches, the procedure is applied to a commonly used dataset.

2. Related Work

VSLAM has garnered a lot of attention over the years [6], and several very advanced VSLAM algorithms, like MonoSLAM [7], ORB-SLAM2 [8], ORB-SLAM3 [9], SPM-

SLAM [10], RGB-D Mapping and Localization [11], and wide-ranging direct SLAM (LSD-SLAM) [12] have emerged.The ORB-SLAM3 technique, developed by Campos et al., demonstrated strong real-time performance across various environments, effectively handling both small and large scenes, as well as indoor and outdoor settings. Table 1 provides an overview of relevant research that utilizes common open-source datasets for SLAM.

Additionally, it has the capability to assist with Vision-centric navigation, combining inertial systems with hybrid mapping. It can function with various cameras such as monoculars, binoculars, and advanced depth-sensing RGB-D cameras utilizing either fisheye or pinhole configurations. The SLAM algorithms for each of these rely on a static environment or very slow-moving objects. However, though, dynamic objects like cars and animals will always exist. For localization, the majority of Simultaneous Localization and Mapping algorithms use feature points, which are often obtained from areas with a lot of textural information [13–15]. As a result, a significant quantity of erroneous feature points will be gathered by the algorithms when a moving object contains a lot of textural information. This will result in low pose estimate accuracy, substantial errors in the path, and even instances of double shadows appearing on the map. Therefore, another problem in the field of VSLAM is how to reduce the resistance of moving objects and enhance positioning accuracy and resilience in dynamic conditions. suggested initially by Peng et al. [16], With three primary stages of development, the SLAM approach has been used for over thirty years. The SLAM difficulty is put forth and transformed into a state consideration problem in the first stage, known as the conventional phase. Typically, a Nonlinear estimation filter, sequential Monte Carlo method, and statistical parameter estimation are used to address the sensor data source, which is LiDAR. This phase's disadvantage is the challenge of determining position is handled independently originating from the problem of constructing a map, and the map's convergence nature is disregarded. The algorithm analysis phase is the second stage. In 2007, A.J. Davison presented MonoSLAM, which was regarded as the first SLAM system using monocular vision in real-time navigation; however, it was restricted to offline map generation and localization. Qin et al. [17] later proposed RWT-SLAM, It used Implementing nonlinear optimization as a backend to achieve parallelization of the process of creating maps and tracking. Nevertheless, it is limited to tiny areas and is leading to a loss throughout the tracking process. Zang et al. [18] suggested an adaptive ORB-SLAM3 in 2023 according to PTAM.

Table 1. Well-known free datasets suitable for SLAM applications.

Dataset	Input Device	Setting	Ground-Truth	Exploitation
Cityscapes [19]	Stereo	Outdoor	GPS	Daimler AG R and D, Max Planck Institute for Informatics, TU Darmstadt Visual Inference Group
KITTI [20]	Stereo/3D laser scanner	Outdoor	GPS/INS	Karlsruhe Institute of Technology and Toyota American Institute of Technology
TUM RGB-D [21]	RGB-D	Indoor	Motion capture	Technical University of Munich
TUM Mono VO [22]	Monocular	Indoor/Outdoor	Loop drift	Technical University of Munich

Table 1. *Cont.*

Dataset	Input Device	Setting	Ground-Truth	Exploitation
TUM VI [23]	Stereo/IMU	Indoor/Outdoor	Motion capture	Technical University of Munich
EuRoc [24]	Stereo/IMU	Indoor	Station/Motion capture	Eidgenössische Technische Hochschule Zürich

It used a token-based model to overcome the aggregate error problem and focused on ORB characteristics for posture estimation; nonetheless, frame loss persisted during rotation. The prediction and robustness phase is the third stage. Deep learning has been progressively included in VSLAM as part of the overall advancement of AI technology to gain resilience and overview of the scene interpretation, in addition to enhancing the efficiency of computing resources. VSLAM systems leveraging deep learning techniques use complex CNN structures like YOLO to obtain semantic labels for features [25], SSD [26], SegNet [27], and Mask R-CNN [28] when an updated frame is presented. Wu et al [29]. Suggested YOLO-SLAM, a dynamic SLAM system that eliminated moving Feature Points by integrating the multiview geometry module and Mask R-CNN instance segmentation method into ORB-SLAM2. In Detect-SLAM, Zhong et al. [30] to a comparatively rapid SSD target identification approach was utilized to identify visual frames to expand the area of influence and eradicate the effect of interfering variables within a dynamic setting. Next, feature matching was used to propagate motion probability. Z. Rui et al. [31] reduced the impact of these changing visual signals on localization and mapping by creating masks of dynamic objects using multiview Geometric Approaches to Case Segmentation to exclude these masked regions of the image from being processed during optimization. Yu et al. [32] improved ORB-SLAM2 has been enhanced to develop DS-SLAM, which utilizes point cloud motion analysis to eliminate dynamic feature points and incorporates a SegNet neural networks applied to semantic segmentation. Yun et al. [33] proposed a novel geometric method for filtering dynamic characteristics within the detection area and generated system semantic information using an improved lightweight target detection network. A vision-inertial fusion study was conducted in [34] in which a camera and IMU sensor fusion method were used to estimate the position of an unmanned aircraft system (UAS) using a CNN-LSTM-based network known as "HVIOnet", which stands for hybrid visual-inertial odometry network. In [35], the UAV pose estimation was improved by reducing the noise of the inertial data using the Savitzky-Golay filter and extracting visual-inertial features based on the Inception-v3 model, which was fed to the Gaussian process regression unit. The inertial image features have been enhanced with optical flow. The proposed method was tested on the EuRoC dataset and its dataset. Theodorou et al. [36] improved Using the YOLOR object discovery channel and optic-flow approach, ORB-SLAM3 can recognize and remove active objects in keyframes and verify them in specialized train station datasets, increasing accuracy by 89%.

3. System Overview

Two more modules are added to Our-SLAM in comparison to the traditional ORB-SLAM3 structure: a detection framework for objects module and a depth estimation module, as shown in Figure 2. Initially, the system evaluates the RGB pictures from the monocular camera before utilizing YOLOv5 to recognize objects and extract feature points simultaneously. Next, a monocular depth estimate is created. The system tracks pixel movement between frames using optical flow techniques, giving more information about the relative motion of static and dynamic components. Both map consistency and tracking stability are enhanced by this information. After object identification, the tracking thread

receives the determined object frame positions. After the camera captures the image, it feeds it into the TRACKING thread, the first of the ORB-SLAM3 system's four modules. The Extract ORB module initializes the map by first extracting ORB features from the picture when it is input. Once the map configuration has begun procedure is complete, the Starting Pose Evaluation module uses the initialization map to set the camera's position. Following initialization, The keyframe discrimination system is activated screening the input being the new keyframe of the subsequent thread, and the system begins tracking and building the map. The keyframe chosen by the Tracking thread serves as input to the five modules that make up the LOCAL MAPPING thread. Following receipt of the keyframe insertion module integrates newly added keyframes into the local map. While the new point generation module adds new map points, the nearest map point culling module adds classic map references in keyframes for the nearby map that aren't already there. The Culling module for Local Keyframes eliminates the existing keyframe after the local map is improved in a localized way by the Local BA module. after the new map points have been added. There are two primary modules in the LOOP CLOSING phase: detection and Correction Loops. The Candidate Detection module compares the current keyframe during posture estimation and optimization, choosing the keyframe with the highest result to feed into the Loop Correction module. To lower the inaccuracy and improve the accuracy of the mapping outputs, the Loop Correction module use the new entered keyframes for pose estimation and global map optimization.

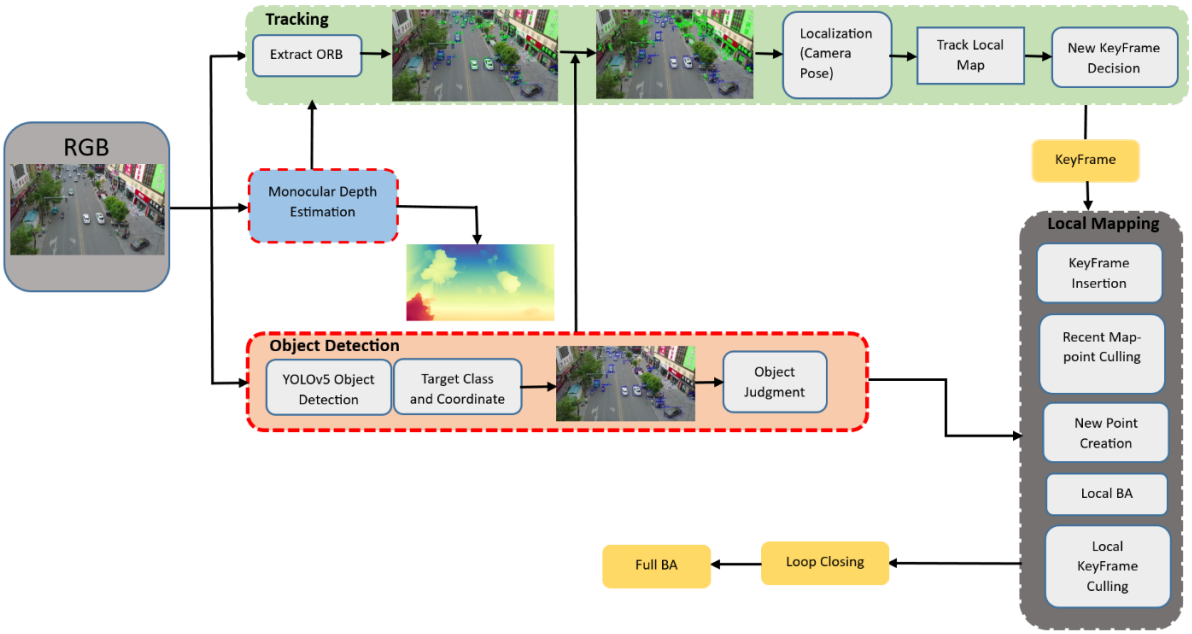


Figure 2. Flowchart of the proposed approach.

3.1. YOLOv5-Based Object Detection Module

As seen in Figure 3, the YOLO network model, which uses a special convolutional network design, is one technique for target detection. Before the input is processed by a convolutional neural network, the image is analyzed, and it is first resized and normalized. The target object's detection information is clearly displayed on the far right of Figure 3, where the predictions' outcomes are displayed. Compared to the fundamental region-based convolutional neural network approach, the You Only Look Once model offers faster and more efficient picture recognition and detection. It is a more sophisticated and integrated framework [37]. Figure 3 illustrates how the YOLO series focuses on picture target recognition. First, the input image is subjected to feature extraction. Following

a chain of convolutional actions performed on the image, the first convolutional layer produces three distinct feature maps, each measuring 16 × 16, 64 × 64, and 256 × 256. It is required to merge the features from the three different images of varied sizes in order to generate stronger target features for better outcomes because during the feature extraction procedure, the three feature photos' varying sizes prevent them from accurately representing the target information. Upon receiving the three images representing features, the neural network of the second convolutional layer does a convolutional procedure involving the feature representations of the image, converting them into three identically sized feature images. The final detection image containing the target detection result is obtained by fusing the three feature images, which are now all identical dimensions, in the categorization and position estimation. Dividing the image broken down to pixel dimensions and subsequently extracting features from the pixel-level feature areas is one of the convolutional network's tasks.

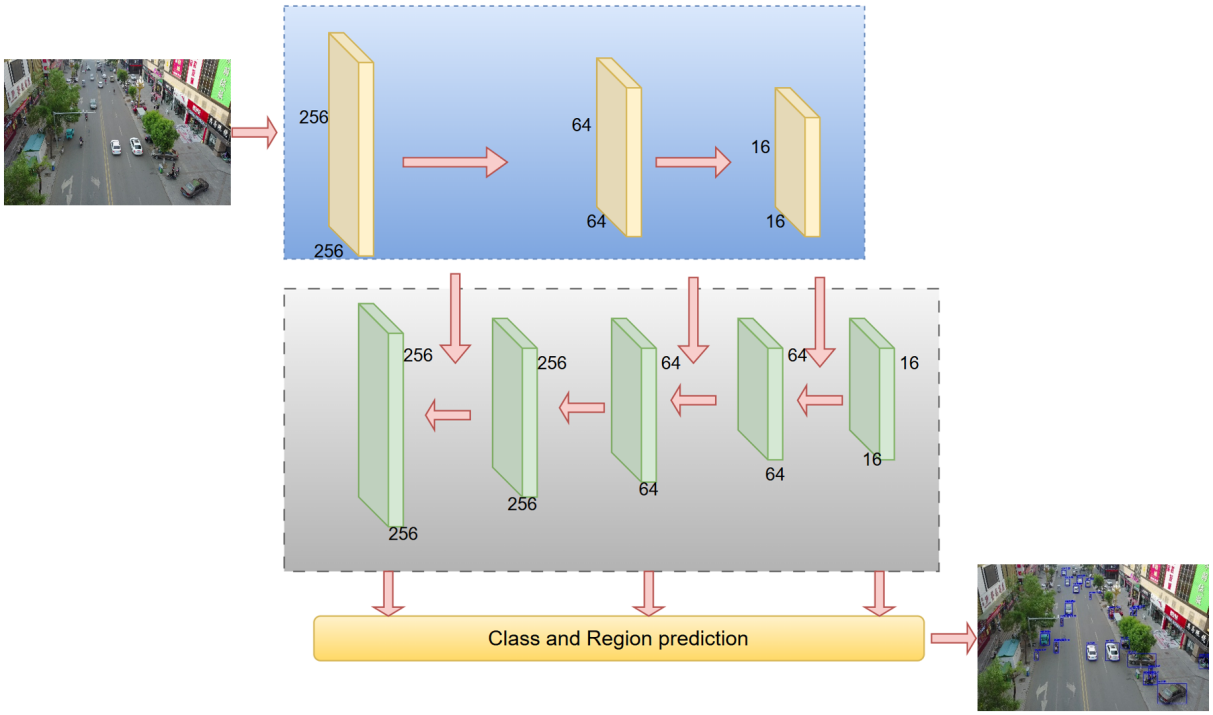


Figure 3. Principle of YOLO algorithm structure.

The network structure depicted in Figure 4 Is a better version of the YOLOv5-l model. The downsampling layer in the network's backbone is the RepVGG reparameterized convolutional structure. This convolutional structure simultaneously trains 1 × 1 and 3 × 3 convolutions during training. The two convolution operators are synthesized into a single 3 × 3 inference execution involving a convolutional layer. By utilizing this approach, the model is capable of acquiring stronger characteristics devoid of increasing the size of the model or slowing down inference. We add a tiny object-detecting head and expand the PAFPN structure to three levels in the neck area. SSpatial and channel elements are drawn from three separate layers of feature representations in the network backbone by integrating the suggested sandwich fusion module. The multi-scale detection mechanism's potential to capture spatial dimensions location data about the items to be identified is improved by this modification.

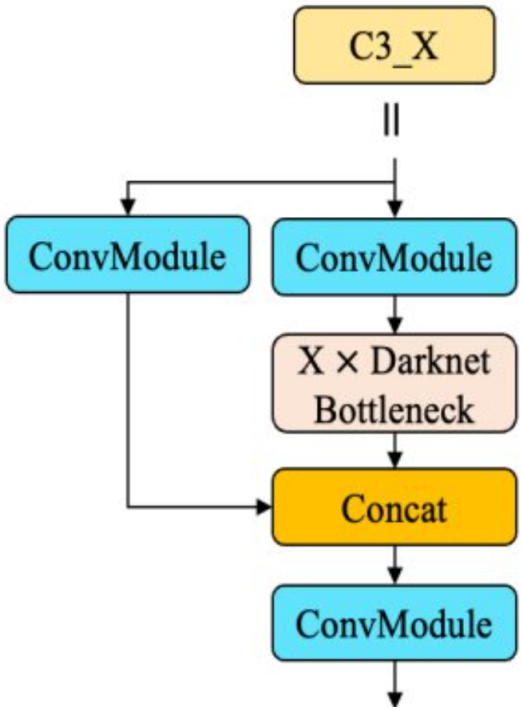


Figure 4. Yolov5 C3_X structure.

Comparative Analysis

Comparison of YOLOv5's performance against alternative object detection methods, such as Faster R-CNN or SSD is presented in Table 2.

Table 2. YOLOv5's performance against Faster R-CNN and SSD models.

Feature	Faster R-CNN	SSD	YOLOv5
Architecture Type	Two-stage detector with Region Proposal Network (RPN)	Single-stage detector	Single-stage detector
Accuracy	High accuracy but slower due to proposal generation	Good accuracy with lower complexity	High accuracy with a good balance of speed
Speed	Moderate speed (typically 7–10 FPS)	Faster than Faster R-CNN, around 60 FPS	Very fast (up to 140 FPS)
Complexity	More complex due to two stages	Simpler architecture, easier to implement	Simple architecture, easy to deploy
Dynamic Object Handling	Effective but can struggle with rapid changes in dynamic environments	Handles dynamic objects reasonably well	Excellent for real-time detection in dynamic scenarios
Use in SLAM	Well-suited for applications needing high precision but may lag in speed	Suitable for applications where speed is critical and some accuracy can be sacrificed	Ideal for real-time SLAM applications where both speed and accuracy are needed

In the context of Simultaneous Localization and Mapping (SLAM), false positives (incorrectly identified objects) and false negatives (missed detections) can severely impact the accuracy and reliability of the system.

- False Positives: These can lead to unnecessary corrections in the SLAM algorithm, causing instability in the generated map. For instance, if a non-existent object is detected, it may result in erroneous adjustments to the robot's position.
- False Negatives: Missing critical objects can hinder the SLAM system's ability to navigate effectively. For example, failing to detect obstacles could lead to collisions or inefficient path planning.

To address the challenges posed by false positives and negatives in SLAM systems utilizing object detection models like YOLOv5, model fine-tuning and sensor fusion can be employed.

The YOLOv5 algorithm is used to initially produce the recognition boundaries of assorted objects once each RGB image frame has been entered. The disparities between the input RGB images and the processed images in Figure 5 prove that YOLOv5 may create observation frames for identifying various object classes and the Classification of these objects.



Figure 5. Input images were managed using the YOLOv5 method; the processed images are displayed on the right, while the original RGB images are displayed on the left.

The experiments assess the suggested approaches in terms of model parameter size and detection performance. Metrics such as P (precision), R (recall), AP (average precision), mAP (mean average precision), and the count of million parameters (M) are essential for assessing network parameter size are among the experimental measures. The percentage of accurately anticipated targets among all identified targets is known as precision. The equation is used to calculate it (1),

$$P = \frac{TP}{(TP + FP)} \tag{1}$$

When the inaccurate prediction targets are denoted by FP and the correct prediction targets by TP. The percentage of correctly recognized targets among all targets that are currently in existence is known as recall. The calculation is done using Equation (2),

$$R = \frac{TP}{(TP + FN)} \tag{2}$$

where FN stands for targets that are present but haven't been properly identified.

The region that the precision and recall graph encloses is known as the average precision. The calculation is done using Equation (3). Three distinct average precision indicators are part of our metrics: AP 0.5, AP 0.75, and AP 0.95. For AP 0.5, The overlap of the union score (IoU) between the predicted and annotated bounding boxes should be greater than 0.50 to analyze the bounding box prediction for accuracy. Bounding box predictions with IoU scores exceeding 0.75 are considered accurate for AP 0.75. The typical precision metrics of different IoU scores that fall between 0.50 and 0.95 are computed for AP 0.95, and the mean of these calculated average precision metrics is then used.

$$AP = \int_0^1 p(r)dr \tag{3}$$

The average precision across all sample types is known as the mean average precision or mAP. Using Equation (4), it is computed. Three distinct mean average precision indicators are part of our metrics: mAP0.5, mAP0.75, and mAP0.95.

$$mAP \frac{1}{k} \sum_{i=1}^{k} AP_i \tag{4}$$

where the average precision for class i is defined as AP_i, while K represents the total number of classes.

The results show in Figure 6 that the proposed YOLOv5 model exhibits strong performance metrics suitable for real-time object detection applications in autonomous environments. With high precision and recall rates along with a competitive mAP score, the model is well equipped to effectively handle various detection tasks.

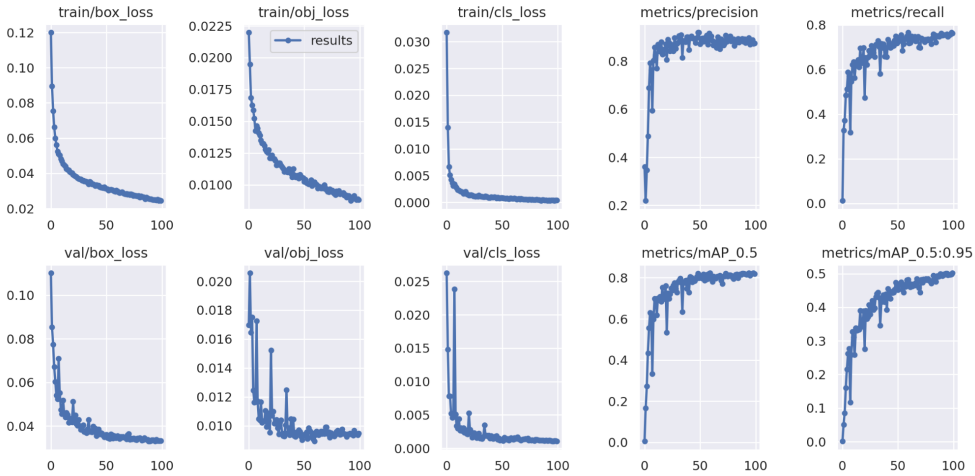


Figure 6. Results of training-testing proposed YOLOv5 model.

3.2. ORB-SLAM Architecture

Because of its thorough and reliable performance, Oriented FAST and Rotated BRIEF (ORB) SLAM, and notably the second edition, ORB-SLAM has become a prominent tech-

nique in the field of visual SLAM. This method makes use of computationally efficient ORB features that do not vary with rotational and scaling alterations. The three-thread architecture of ORB-SLAM, which consists of loop closing, local mapping, and tracking threads, is what makes it unique. Figure 7. Because this method guarantees the system's real-time functioning, it is appealing for UAV applications. Frame-to-frame tracking is carried out by the tracking thread employing a dynamic model, the loop-closing thread detects and addresses significant drift by detecting entire loop closures, and the local mapping thread enhances recently detected features and generates a coherent local map. Even while ORB-SLAM works quite well in a variety of settings, it may have trouble detecting features in environments with minimal texture or dim lighting. This flaw may result in tracking losses, which could cause serious localization issues.

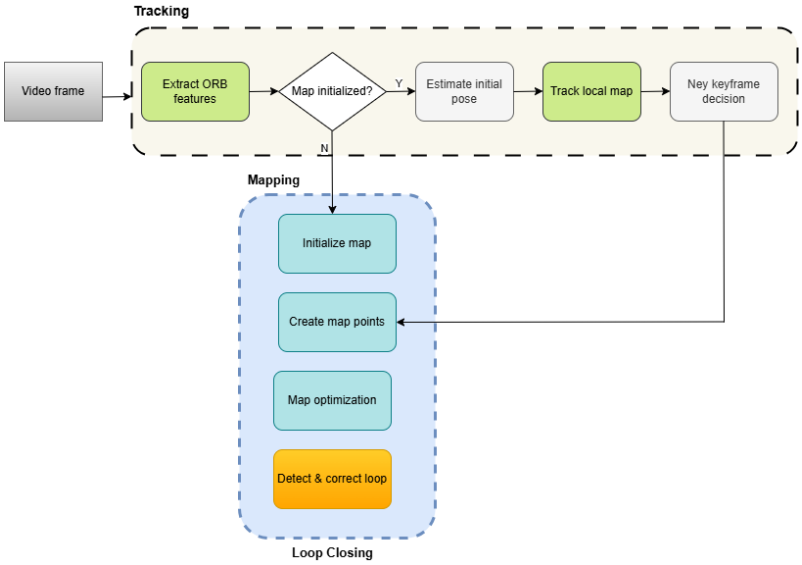


Figure 7. Flowchart of ORB-SLAM.

3.2.1. Tracking

In addition to identifying when a new keyframe needs to be incorporated into the system, the tracking component locates the camera for every frame. Keyframes are essential images stored in the system that act as a storehouse of important information cues necessary for precise localization and efficient tracking. The system re-establishes connections with local map points through reprojection, which is the process of determining the gap between a recognized pattern keypoint in a calibration image and its related world point projected onto that same image, in situations in cases where tracking is affected by obstacles or sudden movements. This makes it possible to optimize the camera pose for every match. The tracking thread is ultimately in charge of deciding when a new keyframe should be inserted appropriately.

3.2.2. Local Mapping

An essential part of the ORB-SLAM technique is local mapping. Processing fresh keyframe images saved in the system that contain informative cues for tracking and localization is the primary focus of this operation. Optimizing the 3D reconstruction around the camera stance is the aim. In order to do this, local mapping triangulates new map points after looking to establish new correspondences among unmatched ORB features in the latest keyframe, utilizing related keyframes from the visibility graph as reference points. Furthermore, a stringent post-creation culling policy is implemented by the local mapping process. This policy is intended to maintain only high-quality map points and is based on the data compiled throughout the tracking operation. The local mapping engine

may detect and eliminate unnecessary keyframes by applying this culling to both the map points and the keyframes themselves. By doing this, ORB-SLAM guarantees a stable and effective local map, which supports strong and trustworthy mapping and localization.

3.2.3. Loop Closing

Another crucial element of the ORB-SLAM method [38] is loop closure, which is responsible for identifying and resolving extended deviation in the anticipated trajectory and compiled map. Essentially, this part recognizes when the camera pans back to a previously explored area and spots a situation that is referred to as a "loop". Pose graph optimization methods are employed to ensure a coherent trajectory and map rectification, while the Bundle of Visual Vocabulary approach is largely used to recognize related situations. ORB-SLAM corrects the integrated map and modifies the anticipated camera track when it detects a loop. This procedure makes sure that only strong and trustworthy repairs are made by identifying and fixing erroneous positive loop closures. Reliable corrections are made. Loop closure is a crucial portion of the entire ORB-SLAM pipeline since it not only increases the map and trajectory's accuracy but also preserves the system's scalability.

As seen in Figure 5, YOLOv5's capabilities allow for the correct identification of dynamic entities like people and cars. After moving objects alongside their bounding boxes are detected, the Kalman filter is used with a box motion framework to track every object that has been discovered. This tactic is referred to as an approach to tracking several objects simultaneously; Ref. [39] provides more information. As shown in Figure 8, this tracking and detection procedure is then coupled with the ORB aspect to identify elements inside the expected boxes.



Figure 8. ORB points feature extraction in Urban area environments.

3.3. Understanding Optical Flow and Depth Estimation

- Optical flow: refers to the way that objects in a visual situation appear to move based on how the observer or the objects move. Usually, it is computed between two successive frames in a sequence of images or videos. Equation (5) provides the classical optical flow equation, which is based on the assumption that brightness remains constant over time.

$$I_x v_x + I_y v_y + I_t = 0 \tag{5}$$

where I_x and I_y illustrate the spatial gradients of pixel brightness, v_x and v_y represent the flow velocities in the x and y directions, respectively, and I_t is the temporal gradient.

- Depth estimation: involves determining the distance of objects from the camera, which can be inferred from optical flow. The relationship between optical flow and depth is often explored through motion parallax, where objects appear to move faster when they are closer to the observer than when they are farther away. This principle can be expressed mathematically as Equation (6):

$$d = \frac{Bf}{(x_2 - x_1)} \tag{6}$$

Here, d represents depth, B is the baseline distance between two camera positions, f is the focal length, and $x_2 - x_1$ corresponds to the observed optical flow. Figure 9 depicts an optical flow and depth estimation diagram.

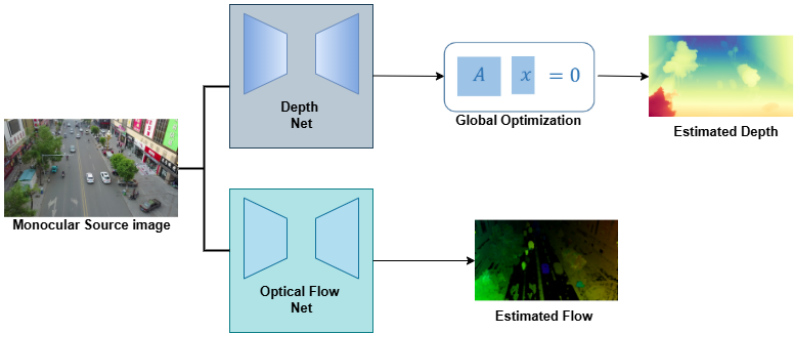


Figure 9. Optical flow and depth estimation from the input image.

Optical flow and depth estimation often encounter difficulties with occluded objects, motion blur, and fast-moving objects. Our proposed approach helps identify occluded areas by comparing forward and backward flow estimates. This approach reduces errors associated with occlusions but still results in a localization accuracy drop of about 15% in complex urban scenes. By exposing the model to blurred images and fast-moving objects during training, our system learns to identify objects even when they are not sharply defined. Table 3 shows the evaluation of depth estimation and optical flow.

Table 3. Evaluation of depth estimation and optical flow modules.

Condition	Relative Depth Error	Optical Flow Outlier Rate
ideal conditions	0.08	2.6
under challenging conditions	0.12	11.4

3.4. Presumption

The provided pose estimate is denoted as Δ. The time is represented by t (or frame) is indicated by the subscript, and it is assumed that there are an equal number of frames overall and that the times of each frame of the estimated and actual poses match. ATE: The absolute trajectory error, which can be a target indicator of algorithm correctness and global trajectory consistency, is the variation between the projected and actual positions. We need to align the calculated position with the ground truth, as they are usually represented in different coordinate systems. Given that the scale remains consistent for stereo Mapping and Localization and depth-enhanced RGB SLAM, we should apply the least squares method S 2 SE (3) to determine a framework transformation that converts the approximated pose compared to the real pose. Additionally, we must compute a corresponding model

transformation S 2 Sim (3) from the predicted pose relative to the actual pose for monocular cameras with scaling uncertainty. Therefore, Equation (7) defines the ATE of frame i:

$$F_i := Q_i^{-1} S P_i \tag{7}$$

Equation (8) recommends RMSE for ATE statistics, much like RPE does.

$$\mathrm{RMSE}(F_{1:n}, \Delta) := \left(\frac{1}{m} \sum_{i=1}^{m} \| \operatorname{trans}(Fi) \|^2 \right)^{\frac{1}{2}} \tag{8}$$

The precision (in relation to the actual position of two frames that are spaced apart by a specific time delay Δ is primarily described by RPE or relative pose error. This is similar to the error of the odometer as measured directly. Therefore, Equation (9) defines the RPE of frame i:

$$E_i := \left(Q_i^{-1} Q_{i+\Delta} \right)^{-1} \left(P_i^{-1} P_{i+\Delta} \right) \tag{9}$$

$(m = n-)\Delta$ RPE may be obtained given the interval Δ and the total number n. The population value in Equation (10) can then be obtained by calculating this error using the root mean square error (RMSE):

$$\mathrm{RMSE} \left(F_{1:n}, \Delta = \frac{1}{m} \sum_{i=1}^{m} \| \operatorname{trans}(Fi) \|^2 \right)^{\frac{1}{2}} \tag{10}$$

The adaptation aspect of the relative pose error is represented by $\operatorname{trans}(E_i)$. The size of the RMSE number allows us to assess the algorithm's performance. In actuality, though, we discover that Δ has a lot of options. We may compute the average RMSE overall Δ provided in Equation (11) to thoroughly assess the algorithm's performance:

$$RMSE = \left(E_{1:n} = \frac{1}{n} \sum_{\Delta=1}^{n} \mathrm{RMSE}(E_{1:n}, \Delta) \right) \tag{11}$$

4. Experiment and Evaluation Analysis

4.1. Experiment Preparation

We mainly use our private dataset for assessing the effectiveness of our method, and we contrast it quantitatively with other popular SLAM methods. In the private dataset developed by F628 Lab, data is recorded at a rate of 30 Hz using an RGB monocular camera with a resolution of 640 × 480. The dataset presents real-time updates on the position of the camera and position data, which may be used to estimate the monocular camera's actual location data. Three sequences from the aforementioned dataset are the subject of the experiments in this study. Compared to the popular public dataset EuRoC Known for its real-world data captured in indoor and outdoor environments, the EuRoC dataset is widely used for benchmarking SLAM algorithms. However, its limited range of urban scenes may not fully capture the complexities encountered in varied urban settings. The benchmarking systems represent ORB-SLAM3 and U-SLAM. Ubuntu 22.04.4 is the operating system utilized in this experiment and the hardware is Antec Desktop Intel(R) Core i7-6700K, 4.0 GHz, 8 logical processors 64-bit NVIDIA Denver NVIDIA Pascal GPU with 256 CUDA cores 32 GB memory. SLAM issues have existed for many years. Despite their disparate areas of concentration, a number of outstanding algorithms have surfaced in recent decades, all of which have made varied contributions to the quick advancement of SLAM technology. Every algorithm needs to be properly compared. A SLAM method may often be assessed from a number of angles, including accuracy, complexity, and time consumption. Most significantly, we focus on its

correctness the most. The two primary criteria employed to evaluate SLAM precision are RPE (Relative Pose Error) and ATE (Absolute Trajectory Error). The characteristic between the actual camera pose and the position that the SLAM system predicts is directly measured by the absolute trajectory inaccuracy. The relative pose error is used to calculate the difference in position between two distinct timestamps to measure system drift.

4.2. Calibration

As shown in Table 4, it is crucial to supply the appropriate camera settings to the SLAM systems. The OpenCV algorithms, which are based on Zhang's methods, were used to calculate the UAV's intrinsic matrix. By default, the implementation sets the pixel skew to zero. The technique calls for pictures of a well-known and distinguishable pattern captured from various viewpoints and distances. After using the Dji Phantom3 UAV to take many pictures of a checkerboard pattern, the computed intrinsic and distortion parameters were obtained in the following table:

Table 4. Camera parameters.

Parameters	Values
Camera resolution	640 × 480
Camera calibration and distortion parameters	
fx	1126.1736
fy	1126.8619
cx	648.7560
cy	380.3586
*k*1	−0.5481
*k*2	0.3687
*p*1	0.00038
*p*2	−0.0014
Camera frame per second	
Camera fps	10.0
Color order of the images	
Camera RGB	1

4.3. Time and Cost Evaluation of the Proposed Method

We recorded the tracking time of the proposed technique while the program was running to confirm the real-time performance of the created system and validate the time cost of the proposed method. Throughout the procedure, we also noted the higher computer system's average CPU and GPU utilization. Many techniques, including well-known ones like DynaSLAM and DS-SLAM, now employ deep learning to eliminate dynamic spots to aid SLAM. As indicated in Table 5, we evaluated the time costs of ORB-SLAM3, DynaSLAM, and the proposed approach. With an average tracking time of about 34 ms per frame, our proposed approach outperforms ORB-SLAM3 and DynaSLAM utilizing the enhanced SLAM model. The system's CPU and GPU resource consumption has also dropped, greatly lessening the computational load as shown in Table 5.

Table 5. Time and Cost Evaluation.

Method	Average Tracking Time (ms)	CPU	GPU
Normal conditional	-	5%	4%
ORB-SLAM3	45	35%	9%
DynaSLAM	41	43%	22%
Proposed method	34	39%	19%

4.4. Comparative Experimental Data and Analysis with U-SLAM

First, they are contrasted. The absolute pose error (APE) is shown in Table 6, whereas the relative pose error (RPE) is shown in Table 7. Table 8 displays the results of our comparison of the most advanced DynaSLAM (Geometric techniques) employing SE(3) Umeyama alignment, with the best results bolded. On the three datasets, our system shows lower absolute trajectory errors than DynaSLAM. The accuracy of SLAM is assessed in this work using APE and RPE. The accuracy of the model and the coherence of the global trajectory appear to be reflected in the APE, which clearly distinguishes between the pose estimation results and the true ground truth. The odometer primarily evaluates the extent of discrepancies error in the relative motion among each set of trajectory points by measuring both the comparative error in translation and the error of relative rotation, represented as RPE. In this study, four metrics Root Mean Square Error (RMSE), mean error, median, and standard deviation of Absolute Trajectory Error (ATE), and RPE have been chosen for evaluation. In urban situations, U-SLAM performs better than ORBSLAM3, as shown by the data in Tables 6 and 7. U-SLAM produces a considerable decrease of up to 43.47% in APE and 26.47% RPE in S000, and its accuracy is higher for sequences with moving objects and more motion inside the image.

Table 6. Comparison of APE in F628 Dataset sequences between ORB-SLAM3 and U-SLAM.

Sequences	ORB-SLAM3				U-SLAM				Improvements%			
	RMSE	Mean	Median	Std	RMSE	Mean	Median	Std	RMSE	Mean	Median	Std
S000	0.0046	0.0036	0.0026	0.0028	0.0026	0.0023	0.0023	0.001	43.47	36.11	11.53	64.28
S001	0.0213	0.0268	0.0273	0.00034	0.0124	0.0176	0.0251	0.0015	41.78	34.32	8.05	55.88
S002	0.0287	0.0292	0.0289	0.0037	0.0177	0.0189	0.0262	0.0021	38.32	35.27	9.34	43.24

Table 7. Comparison of RPE in F628 Dataset sequences between ORB-SLAM3 and U-SLAM.

Sequences	ORB-SLAM3				U-SLAM				Improvements%			
	RMSE	Mean	Median	Std	RMSE	Mean	Median	Std	RMSE	Mean	Median	Std
S000	0.0034	0.0016	0.0015	0.0030	0.0025	0.0014	0.0014	0.0019	26.47	12.5	10.25	38.51
S001	0.0316	0.0342	0.0364	0.0032	0.0261	0.0303	0.0321	0.0021	17.40	11.69	11.81	34.37
S002	0.0348	0.0322	0.0289	0.0024	0.0292	0.0291	0.0259	0.0016	16.09	9.62	10.38	33.33

Table 8. Comparison of absolute trajectory error with DynaSLAM.

Sequences	ORB-SLAM3	DynaSLAM	U-SLAM (Ours)
S000	0.047	0.038	0.024
S001	0.062	0.057	0.049
S002	0.038	0.032	0.028

4.5. Proposed Approach Evaluation

EVO is a Python-based assessment tool for the SLAM method that works with a variety of datasets [40]. It may also create a comparison graphic between the test algorithm and the actual path in addition to acquiring APE and RPE data. is an extremely practical assessment package. A publicly accessible software framework called SLAMBench2 compares existing and upcoming SLAM systems to an expandable list of data sets. It uses a similar and predetermined set of performance measurements and contains both open and closed source code. It is easy to integrate new SLAM algorithms and datasets, and it supports a wide range of current SLAM methods and datasets, including Elastic Fusion, ORB-SLAM3, and OKVIS. Our suggested approach considerably lowers the overall trajectory deviation in datasets containing dynamic objects, such as the S000, S001, and S002 datasets, in contrast to the geometric approach. The dataset consists of a car going through a metropolitan setting with lots of moving and stationary cars, pedestrians, and other objects. In comparison to the geometric technique, our system provides a lower median error on the S000 dataset, as seen in Figures 10 and 11.

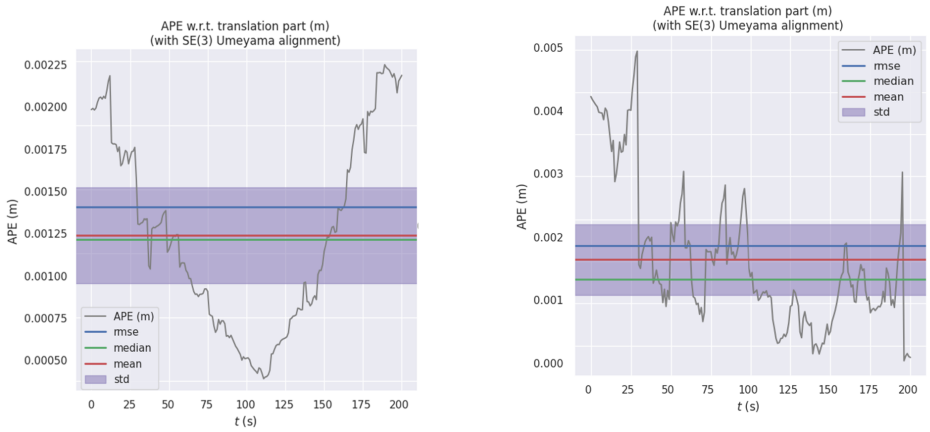


Figure 10. This figure illustrates the S000 sequence dataset Absolute Pose Error (APE) distribution graphs for our proposed method compared to existing methods. The x-axis represents time in seconds, while the y-axis shows error in meters. The left side is the APE of U-SLAM (ours) and the right side is the APE of ORB-SLAM3.

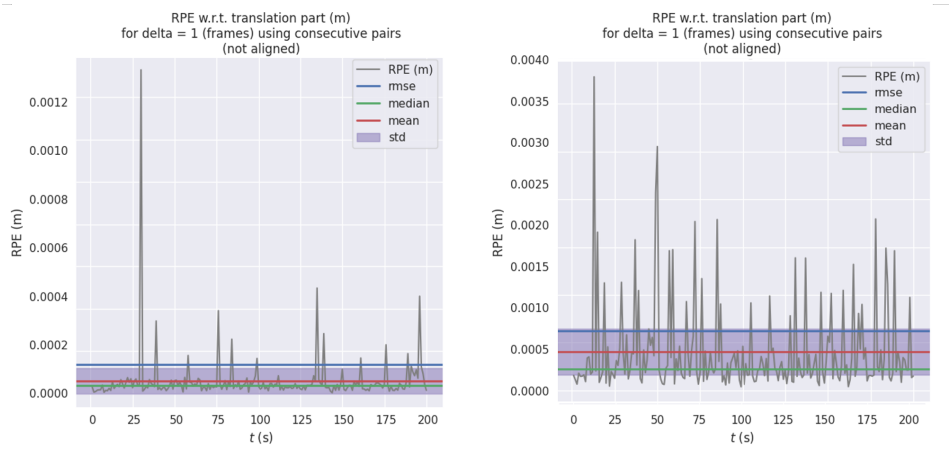


Figure 11. This figure illustrates the S000 sequence dataset Relative Pose Error (RPE) distribution graphs for our proposed method compared to existing methods. The x-axis represents time in seconds, while the y-axis shows error in meters. The left side is the RPE of U-SLAM (ours) and the right side is the APE of ORB-SLAM3.

The APE three-dimensional visual depictions of ORB-SLAM3 and U-SLAM trajectories for the S000 sequence are displayed in Figure 12. The APE partition for both U-SLAM and ORB-SLAM3 is depicted in this image with clarity. The colorful line shows the error translated onto the trajectory, while the gray dashed line shows the ground-truth data.

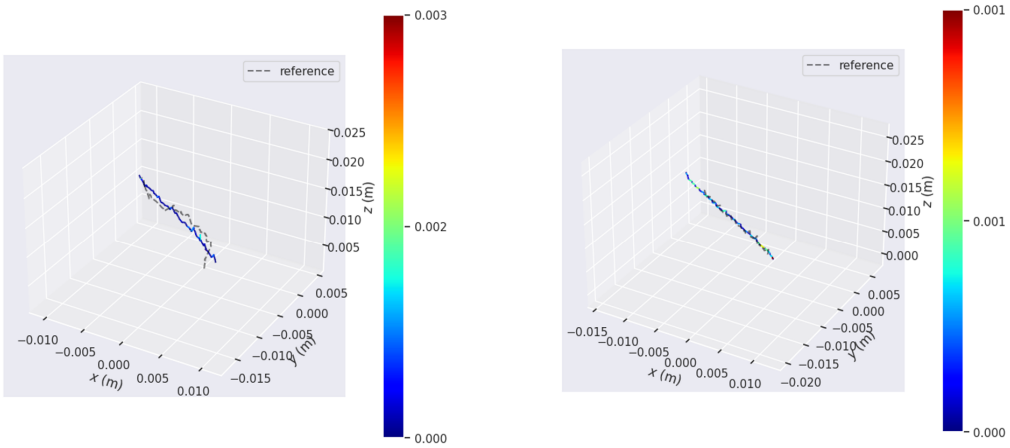


Figure 12. S000 sequence dataset relative pose error distribution graphs: the left side is the RPE of U-SLAM and the right side is the RPE of ORB-SLAM3.

We compare the ground-truth trajectory with ORB-SLAM3 and U-SLAM's computed trajectories for additional analysis. A comparison graph of three different 3D trajectories is shown in Figure 13, and a Graph comparing three trajectories based on their XYZ coordinates and roll, pitch, and yaw (RPY) angles is shown in Figure 14. The S000 sequence with xyz coordinates is shown by Figure 14 on the left, whereas the S000 sequence with rpy coordinates is represented by Figure 14 on the right. In the image, the ground-truth trajectory is shown by the gray dashed line, the ORB-SLAM3 trajectory is presented by the green line, and the U-SLAM estimated trajectory is presented by the blue line.

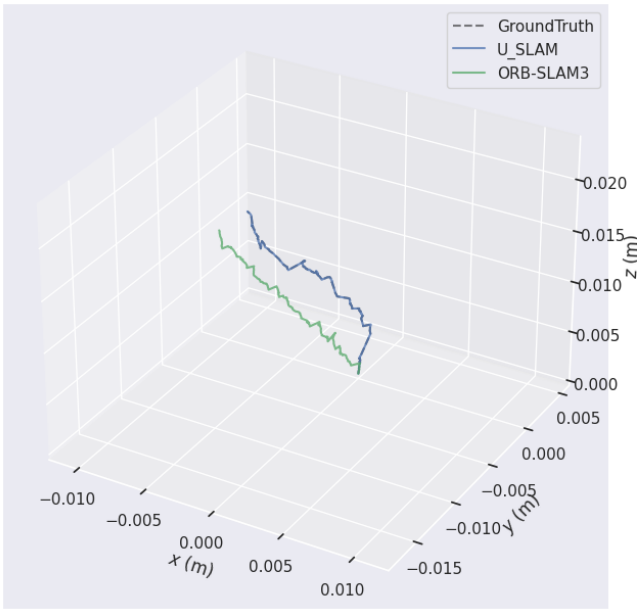


Figure 13. Three different 3D paths are displayed in comparative graphs: The groundtruth is illustrated by the gray dashed lines depicted in the figure, the ORB-SLAM3 trajectory is illustrated by the green lines, and the U-SLAM trajectory is illustrated by the blue lines.

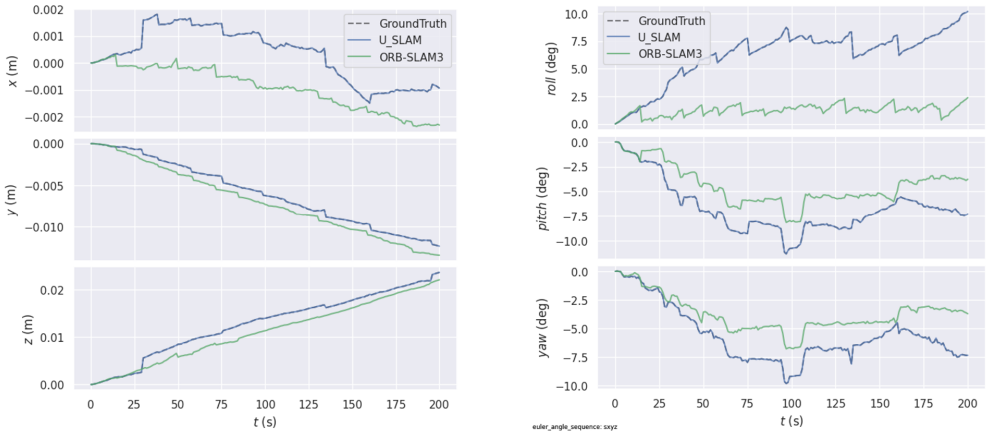


Figure 14. Three trajectories in the xyz coordinates and rpy orientations are displayed in comparison graphs; the S000-xyz series is illustrated by the left-side figure, and the S000-rpy sequence is illustrated by the right-side figure. In the graphics, the ground truth is shown by the gray dashed lines; the ORB-SLAM3 computed trajectories are shown by the green lines, and the U-SLAM estimated trajectories are shown by the blue lines.

5. Discussion

We suggest combining ORB-SLAM3 with deep learning-based monocular depth estimation and optical flow approaches to increase the accuracy and resilience of visual SLAM in variable conditions. Using a monocular camera, ORB-SLAM3 enables UAV navigation for real-time tracking and mapping. Even with ORB-SLAM3's improvements, analyzing scenes with a high density of dynamic objects still presents computing hurdles. To enhance spatial awareness and lessen scale drift in SLAM systems, ORB-SLAM3 integrates deep learning-based depth estimation algorithms. For UAVs operating in challenging outside conditions, the system's use of convolutional neural networks (CNNs) allows it to extract more precise depth information from single images. Optical Flow technique is used to track the motion of objects between frames, helping to determine dynamic regions within a scene. This is particularly useful in low-texture or low-light scenarios where traditional feature detection methods may struggle. When combined with ORB-SLAM3 and deep learning-based monocular depth estimation, it aids in distinguishing between static and dynamic features, thereby enhancing mapping accuracy by focusing on static points. Our proposed model demonstrates promising real-time feasibility for UAV localization in challenging environments. With frame processing rates of up to 40 fps, optimized memory usage under 1 GB, and compatibility with embedded hardware platforms, the system is well-suited for deployment in urban mobility scenarios. These attributes ensure that it can effectively address the challenges posed by low-texture and low-light conditions while maintaining high localization accuracy and operational efficiency. However, our proposed model increases computational requirements, which is a limiting factor for small UAVs with restricted processing power. Optimizing this model for efficiency without sacrificing performance remains a challenge. Future research should focus on optimizing these systems further to address computational challenges and improve their adaptability to various environmental conditions.

6. Conclusions

This study introduces a novel approach, U-SLAM, which combines deep learning (YOLOV5)-based monocular depth estimation and optical flow approaches to object detection with reliable SLAM in urban environments. The performance of the suggested

approach has been thoroughly examined using the F628drone private dataset, and the outcomes are contrasted with those of the most advanced SLAM system, ORBSLAM3. The outcomes of the experiment show how well our method works in dynamic environments. Our proposed approach can recognize moving objects (cars, people) in urban environments by integrating YoloV5-based object detection with ORB-SLAM3. For every scenario in the dataset, the suggested approach produces the minimum Absolute Pose Error (APE) and Relative Pose Error (RPE), quantitatively outperforming existing methods. Particularly in situations with strong dynamics, U-SLAM produces impressive gains, much outperforming ORB-SLAM3. The findings confirm the accuracy of drift reduction and camera trajectory estimation, which are frequently connected to visual odometry and SLAM systems. Additionally, our method's efficacy was confirmed by qualitative analysis of the recreated images. The proposed SLAM system effectively addresses scalability challenges in large-scale urban environments while ensuring real-time performance. Through map management, distant landmark utilization, and memory efficiency strategy, the system demonstrates improved localization accuracy and robustness under real-world conditions. Experimental results confirm that it meets the required performance metrics for UAV applications, making it a viable solution for navigating complex urban landscapes while adhering to real-time constraints. Our approach's capabilities and adaptability will be further improved by future expansions and optimizations, which will support the continuous development of SLAM research and applications.

Author Contributions: Conceptualization, M.N.; methodology, M.N.; software, M.N. and U.C.; Validation, M.N. and X.L.; formal analysis, X.L. and J.S.; writing—original draft preparation, M.N.; writing—review and editing, M.N., X.L. and U.C.; funding Acquisition, X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Zhejiang 'JIANBING' R&D Project (No.2022C01055).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data and materials used to support the findings of this study are available from the corresponding author upon request.

Acknowledgments: We thank the editors and reviewers for their hard work and valuable advice.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned Aerial Vehicle
GPS	Global Positioning System
VSLAM	Visual Simultaneous Localization and Mapping
ORB-SLAM	Oriented FAST and Rotated BRIEF-SLAM
YOLOV5	You Only Look Once Version 5
RMSE	Root Mean Square Error
RPE	Relative Pose Error
APE	Absolute Pose Error
ATE	Absolute Trajectory Error

References

1. Yang, T.; Li, P.; Zhang, H.; Li, J.; Li, Z. Monocular vision SLAM-based UAV autonomous landing in emergencies and unknown environments. *Electronics* **2018**, *7*, 73–87. [CrossRef]
2. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.M.; Tardós, J.D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]
3. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
4. Bruno, H.M.S.; Colombini, E.L. LIFT-SLAM: A deep-learning feature-based monocular visual SLAM method. *Neurocomputing.* **2021**, *455*, 97–110. [CrossRef]
5. Wang, E.; Zhou, Y.; Zhang, Q. Improved Visual Odometry Based on SSD Algorithm in Dynamic Environment. In Proceedings of the 2020 39th Chinese Control Conference (CCC), Shenyang, China, 27–29 July 2020.
6. Saputra, M.R.U.; Markham, A.; Trigoni, N. Visual SLAM and Structure from Motion in Dynamic Environments: A Survey. *ACM Comput. Surv.* **2018**, *51*, 1–36. [CrossRef]
7. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067. [CrossRef] [PubMed]
8. Mur-Artal, R.; Tardós, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
9. Mur-Artal, R.; Montiel, J.M.M.; Tardós, J.D. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
10. Munoz-Salinas, R.; Marin-Jimenez, M.J.; Medina-Carnicer, R. SPM-SLAM: Simultaneous localization and mapping with squared planar markers. *Pattern Recognit.* **2019**, *86*, 156–171. [CrossRef]
11. Kerl, C.; Sturm, J.; Cremers, D. Dense visual SLAM for RGB-D cameras. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 2100–2106.
12. Engel, J.; Schöps, T.; Cremers, D. LSD-SLAM: Large-Scale Direct Monocular SLAM. In Proceedings of the Computer Vision—ECCV, Zurich, Switzerland, 6–12 September 2014; pp. 834–849.
13. Ali, T.; Hriday, B.; Luis, S.J.; Holger, V. Visual SLAM: What Are the Current Trends and What to Expect. *Sensors* **2022**, *10*, 92–97. [CrossRef]
14. Filipenko, M.; Afanasyev, I. Comparison of Various SLAM Systems for Mobile Robot in an Indoor Environment. In Proceedings of the 2018 International Conference on Intelligent Systems (IS), Funchal, Portugal, 25–27 September 2018; pp. 400–407.
15. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, Present, and Future of Simultaneous Localization and Mapping: Towards the Robust-Perception Age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]
16. Qin, T.; Li, P.; Shen, S. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [CrossRef]
17. Peng, Q.; Xiang, Z.; Fan, Y.; Zhao, T.; Zhao, X. RWT-SLAM: Robust Visual SLAM for Highly Weak-textured Environments. *arXiv* **2022**, arXiv:2207.03539.
18. Zang Q, Zhang K, Wang L, Wu L. An Adaptive ORB-SLAM3 System for Outdoor Dynamic Environments. *Sensors* **2023**, *3*, 1359. [CrossRef] [PubMed]
19. Marius, C.; Mohamed, O.; Sebastian, R.; Timo, R.; Markus, E.; Rodrigo, B.; Uwe, F.; Stefan, R.; Bernt, S. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 3213–3223.
20. Andreas G.; Philip, L.; Christoph, S.; Raquel, U. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237.
21. Available online: https://api.semanticscholar.org/CorpusID:6360565 (accessed on 12 January 2024).
22. Georges, Y.; Daniel, A.; John, Z. FDMO: Feature Assisted Direct Monocular Odometry. *arXiv* **2018**, arXiv:1804.05422v1.
23. Zhong, Y.; Hu, S.; Huang, G.; Bai, L.; Li, Q. WF-SLAM: A Robust VSLAM for Dynamic Scenarios viaWeighted Features. *IEEE Sens. J.* **2022**, *22*, 10818–10827. [CrossRef]
24. Michael, B.; Janosch, N.; Pascal, G.; Thomas, S.; Jörn, R.; Sammy, O.; Markus, A.; Roland, Y. S. The EuRoC micro aerial vehicle datasets. *Int. J. Robot. Res.* **2016**, *35*, 1157–1163.
25. Hu, Y.; Xinjian, W.; Guangdi, Z.; Xiaofei, L. Object detection of DRONES for anti-DRONES based on improved YOLO v3. In Proceedings of the 2019 Chinese Control Conference (CCC), Guangzhou, China, 27–30 July 2019; pp. 8386–8390.
26. Wei, L.; Dragomir, A.; Dumitru, E.; Christian, S.; Scott, R.; Cheng, F.; Alexander, B. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
27. Jonathan, L.; Evan, S.; Trevor, D. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
28. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.

29. Wu, W.; Guo, L.; Gao, H.; You, Z.; Liu, Y.; Chen, Z. YOLO-SLAM: A semantic SLAM system towards dynamic environment with geometric constraint. *Neural Comput. Appl.* **2022**, *34*, 6011–6026. [CrossRef]
30. Zhong, F.;Wang, S.; Zhang, Z.; Chen, C.;Wang, Y. Detect-SLAM: Making Object Detection and SLAM Mutually Beneficial. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 1001–1010.
31. Rui Z.; Jingbin, L.; Junbin, X.; Jianyu, Z.; Yingze, H.; Jiele, Z. GMS VINS: Multi-category Dynamic Objects Semantic Segmentation for Enhanced Visual-Inertial Odometry Using a Promptable Foundation Model. *arXiv* **2024**, arXiv:2411.19289v1.
32. Yu, C.; Liu, Z.; Liu, X.-J.; Xie, F.; Yang, Y.; Wei, Q.; Fei, Q. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. *arXiv* **2018**, arXiv:1809.08379.
33. Yun, J.; Jiang, D.; Liu, Y.; Sun, Y.; Tao, B.; Kong, J.; Tian, J.; Tong, X.; Xu, M.; Fang, Z. Real-Time Target Detection Method Based on Lightweight Convolutional Neural Network. *Front Bioeng Biotechnol.* **2022**, *10*, 861286. [CrossRef] [PubMed]
34. Muhammet Fatih, A.; Akif, D.; Abdullah, Y.; Alper, Y. HVIOnet: A deep learning based hybrid visual–inertial odometry approach for unmanned aerial system position estimation. *Neural Netw.* **2022**, *155*, 461–474.
35. Muhammet Fatih, A.; Akif, D.; Kadir,S. Visual-Inertial Image-Odometry Network (VIIONet): A Gaussian process regression-based deep architecture proposal for UAV pose estimation. *Measurement* **2022**, *194*, 111030.
36. Theodorou, C.; Velisavljevic, V.; Dyo, V. Visual SLAM for Dynamic Environments Based on Object Detection and Optical Flow for Dynamic Object Removal. *Sensors* **2022**, *22*, 7553. [CrossRef] [PubMed]
37. Sun, Y.; Hu, J.; Yun, J.; Liu, Y.; Bai, D.; Liu, X.; Zhao, G.; Jiang, G.; Kong, J.; Chen, B. Multi-objective Location and Mapping Based on Deep Learning and Visual Slam. *Sensors* **2022**, *22*, 7576. [CrossRef]
38. Gupta, A.; Fernando, X. Simultaneous Localization and Mapping (SLAM) and Data Fusion in Unmanned Aerial Vehicles: Recent Advances and Challenges. *Drones* **2022**, *6*, 85. [CrossRef]
39. Li, X.; Wang, K.; Wang, W.; Li, Y. A multiple object tracking method using Kalman filter. In Proceedings of the 2010 IEEE International Conference on Information and Automation, Harbin, China, 20–23 June 2010; pp. 1862–1866.
40. Sturm, J.; Engelhard, N.; Endres, F.; Burgard, W.; Cremers, D. A benchmark for the evaluation of RGB-D SLAM systems. In Proceedings of the 2012 IEEE/RSJ international conference on intelligent robots and systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; pp. 573–580.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.