electronics

MDPI

Article

VSLAM Optimization Method in Dynamic Scenes Based on YOLO-Fastest

Zijing Song [1], Weihua Su [2,*], Haiyong Chen [1], Mianshi Feng [1], Jiahe Peng [1] and Aifang Zhang [3]

[1] School of Artificial Intelligence and Data Science, Hebei University of Technology, Tianjin 300401, China; szj1144040820@163.com (Z.S.); shaiyong.chen@hebut.edu.cn (H.C.); 202132803095@hebut.edu.cn (M.F.); 15927009419@163.com (J.P.)
[2] School of Mechanical Engineering, Hebei University of Technology, Tianjin 300401, China
[3] School of Civil and Architectural Engineering, Beijing University of Technology, Beijing 100124, China; 18822053736@163.com
* Correspondence: 13512062095@126.com

Abstract: Simultaneous localization and mapping (SLAM) is one of the core technologies for intelligent mobile robots. However, when robots perform VSLAM in dynamic scenes, dynamic objects can reduce the accuracy of mapping and localization. If deep learning-based semantic information is introduced into the SLAM system to eliminate the influence of dynamic objects, it will require high computing costs. To address this issue, this paper proposes a method called YF-SLAM, which is based on a lightweight object detection network called YOLO-Fastest and tightly coupled with depth geometry to remove dynamic feature points. This method can quickly identify the dynamic target area in a dynamic scene and then use depth geometry constraints to filter out dynamic feature points, thereby optimizing the VSLAM positioning performance while ensuring real-time and efficient operation of the system. This paper evaluates the proposed method on the publicly available TUM dataset and a self-made indoor dataset. Compared with ORB-SLAM2, the root-mean-square error of the Absolute Trajectory Error (ATE) can be reduced by 98.27%. The system successfully locates and constructs an accurate environmental map in a real indoor dynamic environment using a mobile robot. It is a VSLAM system that can run in real-time on low-power embedded platforms.

Keywords: visual SLAM; object detection; dynamic environments; real-time performance

Citation: Song, Z.; Su, W.; Chen, H.; Feng, M.; Peng, J.; Zhang, A. VSLAM Optimization Method in Dynamic Scenes Based on YOLO-Fastest. *Electronics* **2023**, *12*, 3538. https://doi.org/10.3390/electronics12173538

Academic Editor: Dah-Jye Lee

Received: 15 July 2023
Revised: 5 August 2023
Accepted: 10 August 2023
Published: 22 August 2023

1. Introduction

Robots utilize their onboard visual sensors to perform simultaneous localization and mapping (SLAM) [1] for perceiving the surrounding environment and exhibiting stable performances within specific static scenarios [2]. However, in dynamic scenarios, such as human–robot cooperation and multi-robot collaboration, the presence of other moving robots, personnel, and unstable objects within the environment can significantly reduce the accuracy and precision of mapping and localization. Dynamic objects can lead to erroneous data associations, and the feature point matching between two frames of dynamic objects can result in an incorrectly solved camera pose, thereby reducing system stability.

The robustness of VSLAM [3] systems in dynamic environments has become a focal point for many researchers. The key to solving this issue is to effectively detect and filter dynamic features, preventing the use of features extracted from moving objects during the tracking process. To address dynamic objects within the environment, this paper proposes the use of a deep learning network to extract dynamic components from input data and explicitly discard them as outliers, not participating in pose estimation and mapping. However, existing VSLAM systems require high computing power to support neural networks, making them unsuitable for robots with low computing power platforms, thereby failing to achieve real-time performance. This paper proposes the use of the fastest and lightest known YOLO object detection algorithm, YOLO-Fastest, to detect prior

dynamic objects in image frames within the VSLAM system. YOLO-Fastest is designed to break through the limitations of computing power, enabling real-time object detection calculations on low-cost edge devices. By incorporating YOLO-Fastest into the ORB-SLAM2 [4], the most classic visual SLAM framework, and using methods such as target detection during feature extraction, dynamic feature points within detection boxes can be removed, reducing the occurrence of erroneous epipolar constraints and improving accuracy through increased efficiency. This optimization method is referred to as YF-SLAM in this paper.

To address this challenge, the main insights and contributions of this article are summarized as follows:

- A single-stage YOLO-Fastest object detection network was designed in the VSLAM system, which can quickly locate the area information of dynamic targets in keyframes.
- The system combines depth geometry constraint methods to distinguish dynamic feature points from static feature points in the dynamic target area, effectively filtering out dynamic feature points using only one image information and improving the accuracy of mapping and localization.
- YF-SLAM can be easily deployed on low-computing platforms, which can significantly reduce the time consumption of the VSLAM system and achieve real-time operation of robots in dynamic environments.

This article is divided into four sections: Section 2 summarizes the relevant theoretical work in the field of VSLAM. Section 3 provides a detailed introduction to the framework and proposed methods of YF-SLAM. Section 4 reports the experimental results evaluated on the TUM dataset and post-disaster rescue scenarios and compares them with VSLAM frameworks such as DS-SLAM and DYNA-SLAM to complete the YF-SLAM experiment on the search and rescue robot platform. The final section summarizes the application scenarios and advantages of YF-SLAM in this article and looks forward to future improvement goals.

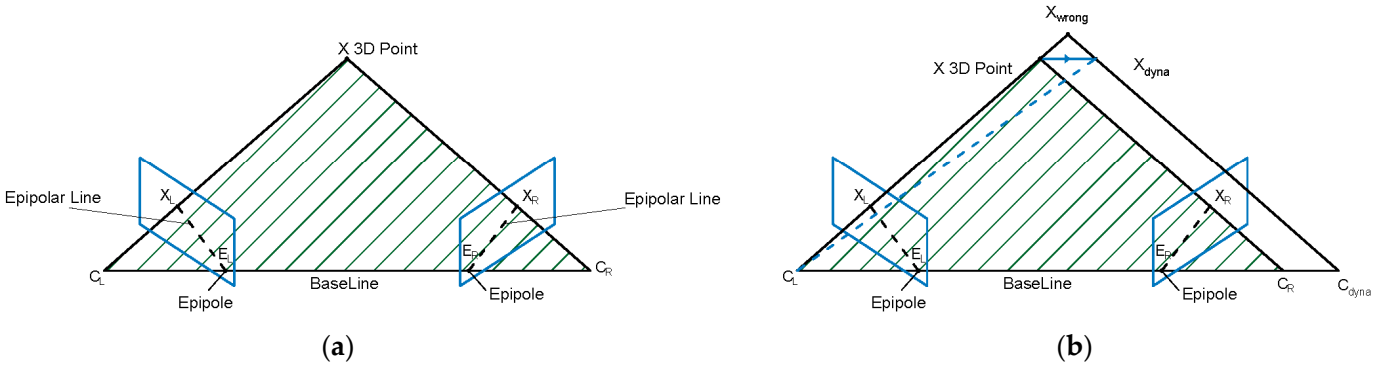
2. Related Work

The epipolar geometry [5] is the geometric constraint relationship between two-perspective geometric modeling, which is mainly used to achieve binocular stereo vision based on triangulation, depth estimation, etc. Figure 1a shows the polar constraints, wherein the real point X in the 3D world is called the object point, the $C_L C_R$ line connecting the optical centers between two cameras is called the baseline, and the intersection point between the baseline and the imaging surface is called the poles E_L and E_R. If the object point has an imaging point of X_L on one camera imaging plane, then the image point on the other camera plane must exist on the intersection line $E_R X_R$ with the plane $C_L C_R X$. as follows:

$$x_2^T F x_1 = 0 \tag{1}$$

$$F = K_2^{-T} \hat{t} R K_1^{-1} \tag{2}$$

The variables x_1 and x_2 represent the matching point positions between two consecutive frames of images, F is the fundamental matrix, K_1 and K_2 are the internal parameter matrices of camera 1 and camera 2, and R and t are the external parameters of coordinate systems 1 and 2. In Figure 1b, the point X is dynamic, and the target moves from X to X_{dyna}. At this point, the epipolar plane becomes $X_{wrong} C_L C_{dyna}$, and the backprojection rays connecting the camera center and projection points cannot intersect at a single point. The presence of dynamic features can lead to erroneous estimation of the fundamental matrix, resulting in incorrect camera pose calculation results.



(**a**) (**b**)

Figure 1. The influence of dynamic target on epipolar constraint relationship. (**a**) Shows the correct geometric constraint relationship, and (**b**) shows that dynamic features do not meet multi-view geometric constraints based on static features.

In typical VSLAM schemes, the RANSAC (random sample consensus) [6] algorithm can only cope with some scenarios where the proportion of dynamic features is relatively low. However, these methods may fail when there are too many dynamic objects occupying a significant portion of the image. At present, there are roughly three categories of approaches to accurately detecting and removing dynamic features: geometric methods, deep learning methods, and the method of combining deep learning with geometry.

2.1. Methods of Geometrical

Relying on geometric constraint technology to handle dynamic SLAM problems utilizes polar geometric features to segment static and dynamic features. Kundu et al. [7] set upper and lower boundaries for the displacement of feature points, and the feature points located outside the boundary in the detection results are likely to be dynamic features. Tan [8] projects the features between two frames and measures the distance between the current frame and the previous frame, while those with a larger reprojection distance are dynamic features. By removing abnormal features, the camera pose can be accurately estimated.

There are also some methods to distinguish dynamic points by processing feature points or map points. Kit [9] trained the classifier in advance and classified the feature points in the known environment to distinguish between static and dynamic feature points. The scheme based on geometry has good real-time performance. However, the geometric method only distinguishes based on the high geometric motion error of feature points, so it cannot handle situations where moving objects temporarily stop.

2.2. Methods of Deep Learning

In recent years, with the development of deep learning, the accuracy of object detection methods has significantly improved. This advancement has provided new possibilities for addressing the challenges of dynamic SLAM. One approach involves utilizing semantic segmentation to remove potential dynamic feature points.

Yu et al. [10] of Tsinghua University proposed the DS-SLAM system based on RGB-D in 2018 and added semantic segmentation and dense semantic octree [11] map construction threads on the basis of the ORB-SLAM2 system to combine the semantic segmentation network SegNet [12] and mobile consistency detection methods to filter the dynamic part of the scene. Specifically, the process begins by extracting ORB feature points and categorizing them as potential dynamic points. These points are then subjected to motion consistency checks to ensure their classification as actual dynamic feature points. Furthermore, a parallel semantic segmentation thread is introduced, where the segmentation results are combined with the ORB feature points from the tracking thread. To enhance stability, a dense 3D semantic octree map is employed. The map filters out unstable factors via probabilistic logarithmic filtering, resulting in the robot obtaining more accurate and

consistent perceptual information for executing the SLAM task. This approach effectively improves the performance of Visual SLAM systems in dynamic environments.

2.3. The Method of Combining Deep Learning with Geometric

In the combination of deep learning and geometry methods, complementary advantages of the two methods. This combined method first provides semantic segmentation information for dynamic targets and then uses geometric constraints to filter dynamic feature points while retaining static feature points.

The DYNA-SLAM [13] system is also improved over the ORB-SLAM2 system. When using monocular or binocular cameras, only prior dynamic objects in the frame are segmented pixel by pixel using the instance segmentation network Mask R-CNN [14], directly not extracting feature points from this region. When using an RGB-D camera, the multi-view geometry method [15] is used to detect potentially moving objects, such as chairs that have been moved by humans, and adds a background restoration function. This raises the accuracy and positioning accuracy for detecting moving objects, but it is time consuming and has poor real-time performance.

In order to address this challenge, YF-SLAM adopts a lightweight single-order network, YOLO-Fastest, and a tightly coupled geometric constraint method to reduce the time consumption and make the system faster and more stable.

3. Dynamic Target Point Elimination

3.1. Framework of YF-SLAM

The YF-SLAM dynamic target point elimination framework is an improvement on the traditional ORB-SLAM2 system. ORB-SLAM2 is known as one of the best SLAM systems based on feature extraction methods, including tracking, mapping, and loop closure detection, and it performs exceptionally well under static environmental conditions. The YF-SLAM system adds object detection and geometric depth filtering threads to the ORB-SLAM2 system, where the object detection thread uses the YOLO-Fastest network and adds bounding box markers to the detected dynamic targets to locate the position of dynamic feature points in each frame. The feature point screening thread utilizes geometric constraint methods to eliminate potential dynamic feature points within the bounding box, while preserving pure static feature points. In Figure 2, the system flowchart of YF-SLAM is shown. The original RGB image is first processed for target detection and feature extraction, followed by the application of geometric constraint methods to recognize and filter out dynamic features marked in red, finally preserving static feature points for pose estimation.

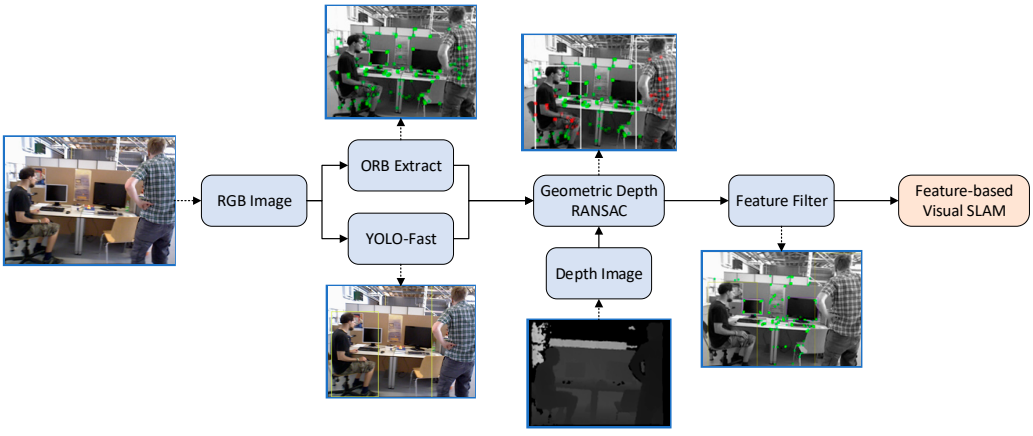


Figure 2. The optimization framework for dynamic target feature points in the YF-SLAM system. First, it extracts ORB features from the original camera and detects predefined dynamic targets. Then, geometric depth constraints are applied to separate these dynamic features marked in red. Finally, only static feature points are used for attitude estimation.

3.2. YOLO Fastest for Dynamic Target Recognition

Target detection models can automatically locate and identify various objects in an image. Currently, single- and two-stage networks are the most mainstream detection models. Faster-RCNN [16] and Mask R-CNN belong to the two-stage network, where target localization is prioritized in the first step by extracting RoI (Region of Interest) from the input image, followed by feature extraction by the network, and identification of the category of each RoI with a multi-class SVM. YOLO and SSD [17] belong to the single-stage network, where both target localization and classification are carried out by the RPN alone. Compared to the two-stage networks, single-stage networks have faster training and recognition speeds. YOLO-Fastest focuses on real-time inferencing performance on a single core, achieving low CPU usage while meeting real-time requirements. The YOLO-Fastest model structure shown in Figure 3. It decouples the regression, background classification, and detection category classification of the detection box into three different feature maps, where the background classification and detection category classification are shared using the same network branch parameters.

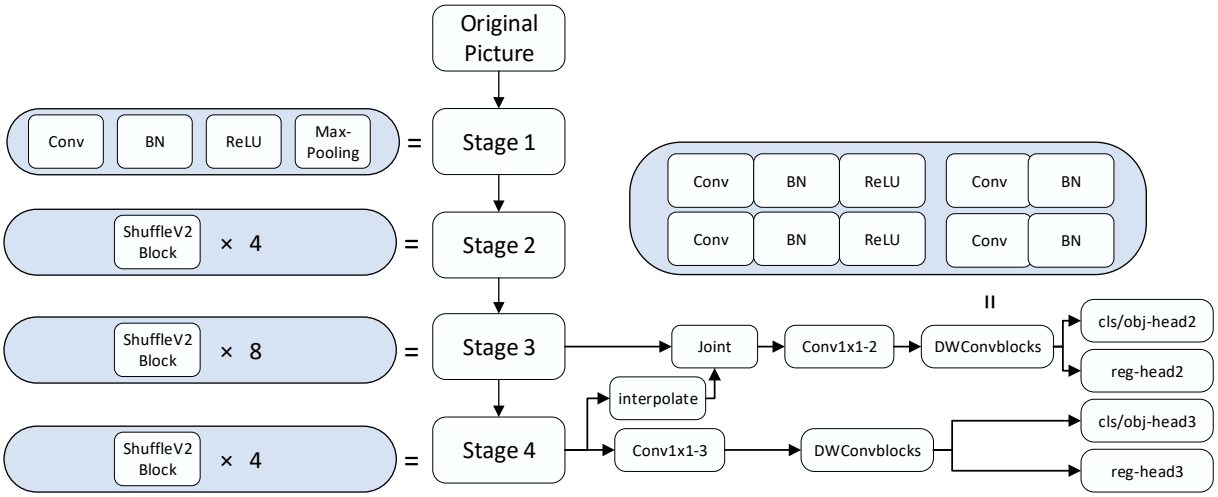


Figure 3. A neural model of a convolutional network from YOLO-Fastest. The backbone of the network is ShuffleNetV2, with four stages as the backbone. The head consists of three parts: classification, regression, and detection. The neck part is a lightweight FPNet network.

YOLO-FastestV2 [18] is the fastest and lightest version of the improved YOLO universal target detection algorithm known to be open source. Table 1 shows the benchmark and evaluation indicators under the framework of the Kirin 990 CPU platform ncnn. In practical applications, considering the power consumption and system resource occupation, it is generally not possible to use multiple cores to infer the model, so a single-core experiment is conducted. Not only does the reasoning of the model take less time, but the system resources consumed by model reasoning are also very low. Different object detection frameworks can affect the processing time of a single-frame image in the tracking thread of the VSLAM system. Compared to YOLOV4-Tiny, YOLO-FastestV2 consumes only one-tenth of the time in single-frame processing. Compared with YOLO-FastestV1.1, YOLO-FastestV2 uses 0.3% accuracy loss in exchange for a 30% improvement in reasoning speed and a 25% reduction in parameters. The mobile end can reach up to 300 FPS, with a parameter count of only 250 k. YOLO-FastestV2 is used to optimize the SLAM algorithm, paying more attention to the cost performance and inference efficiency of the algorithm.

Table 1. YOLO-Fastest evaluating indicator.

Network	COCO Map (0.5) %	Resolution	Run Time (1 × Core) (ms)	FLOPs (G)	Params (M)
YOLO-FastestV2	24.10	352 × 352	5.37	0.212	0.25 M
YOLO-FastestV1.1	24.40	320 × 320	7.54	0.252	0.35 M
YOLOv4-Tiny	40.20	416 × 416	55.44	6.900	5.77 M

3.3. Dynamic Feature Point Removal for YF-SLAM

Feature points are the key to the VSLAM system. High-quality feature points can improve system performance. YF-SLAM strictly screens feature points by means of semantic priors and tight coupling of geometric depth information. The training weight and label files are developed via YOLO-Fastest to identify dynamic targets, such as robots and personnel, and generate a bounding box, which can locate the most likely position of dynamic feature points in the image and return to a rectangle composed of four coordinates. The extracted feature points are mixed into dynamic and static feature points. At this time, the bounding box includes potential dynamic feature points such as chairs and rescue supplies that are easy to move by search and rescue personnel, as well as static feature points in the background. In some papers, for example, DMS-SLAM [19] will directly remove all feature points in the rectangle, thus deleting many static points still in the rectangle, especially when the proportion of dynamic objects is large. This will weaken the constraint of feature matching.

The depth difference of the inherent feature points of dynamic objects tends to be similar, which can be effectively resolved via RGB-D cameras. Firstly, all feature points in the regions of interest for dynamic objects are collected as a set. The variance of depth values is calculated for any two randomly selected feature points within the set. Subsequently, a model is generated by iterating over the set of variances of all feature points, wherein feature points with similar depth values are grouped together. The group with the largest number of feature points is identified as the dynamic feature points, and this method is referred to as the Geometric Depth RANSAC (Random Sample Consensus) approach. The process for filtering static feature points is illustrated in Figure 4. YF-SLAM optimizes the camera's pose and the position of 3D points by adding point clouds and obtaining global point clouds by minimizing reprojection errors.

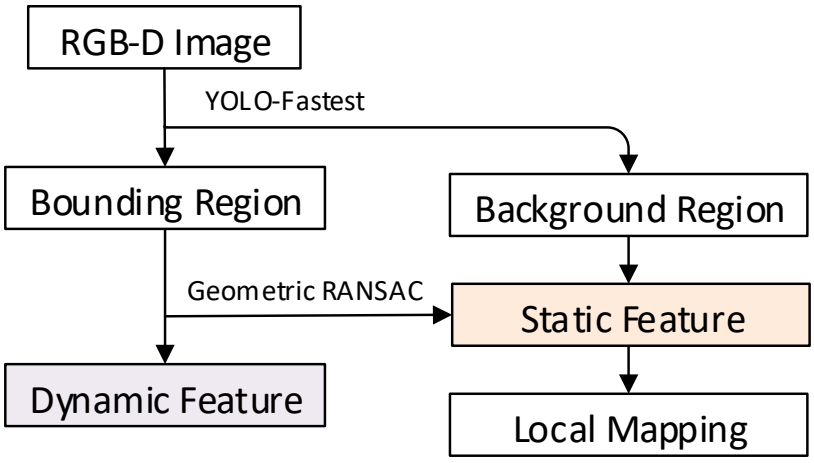


Figure 4. Dynamic feature point logic judgment diagram. In this process, the RGB-D image information is initially processed using the YOLO-Fastest network to generate dynamic regions. Subsequently, the RANSAC algorithm is employed to filter out feature points from the dynamic regions, resulting in the generation of pure static feature points. These pure static feature points, along with background feature points, are utilized to construct loop closure detection, thereby achieving more accurate camera poses and motion trajectory estimation.

4. Experimental Results and Analysis

To verify the effectiveness of this method, we conducted comparative experiments on YF-SLAM, ORB-SLAM2, DYNA-SLAM, and DS-SLAM algorithms in TUM datasets and simulated indoor disaster rescue environments. The evaluation metrics used to assess accuracy were the Absolute Trajectory Error (ATE) and the Relative Pose Error (RPE) [20]. ATE is the direct difference between the estimated pose and the actual pose, which can intuitively reflect the algorithm accuracy and global consistency of the trajectory. The RPE can be understood as a real-time comparison between the actual pose value and the estimated value, calculating the difference between the estimated value of the VSLAM system on two identical timestamps and the actual value of the camera pose at regular intervals. The robustness and stability of the system were represented by the Standard Deviation (S.D.) and the Root-Mean-Square-Error (RMSE) of both metrics.

The TUM dataset [21] contains texture-rich office scenes. For example, the fr3/sitting sequence is moderately dynamic, with two people sitting at a table with subtle movements. The fr3/walking sequence is highly dynamic, with two people walking around the table. The walking sequences are the most challenging as dynamic objects occupy a significant portion of the camera's field of view, and our comparative experiments primarily focused on these sequences. These walking sequences involve two people walking while the camera translates and rotates along the xyz axes, the camera moving on a half sphere with a 1 m diameter, and the camera being relatively stationary while the two people walk, aiming to evaluate the system's robustness to fast-moving dynamic objects.

In the simulated indoor experiments, we aimed to evaluate the system's performance in disaster scenarios such as earthquakes and chemical leaks, where rescue robots utilize visual sensors to perceive the surrounding environment in real-time and perform localization and map construction for search and rescue operations. In scenarios involving human–robot collaboration and multi-robot coordination, the presence of other moving robots, rescue personnel, and unstable objects in the environment can degrade the accuracy and precision of localization and mapping. The YF-SLAM algorithm is particularly suitable for disaster scenarios with fewer types of dynamic factors and limited computational resources but high real-time requirements for rescue robots. The experiments and evaluations were conducted to demonstrate the effectiveness and applicability of the YF-SLAM algorithm in both dynamic office environments (TUM dataset) and simulated indoor disaster scenarios.

4.1. Modular Evaluation

The YF-SLAM system is a combined framework consisting of a semantic module and a geometric module, each playing a different role in achieving accurate localization. Figure 5 illustrates a comparison between the YF-SLAM system and ORB-SLAM2, DS-SLAM, and DYNA-SLAM in terms of trajectory estimation and the deviation between estimated and ground truth roll, pitch, and yaw values in a dynamic scene. The red line represents the ground truth trajectory, while the blue line represents the estimated trajectory. From the first row, it is evident that ORB-SLAM2 fails to effectively handle dynamic environments. In scenes with a high proportion of dynamic features, the semantic module ensures real-time performance by removing more dynamic feature points, thereby improving trajectory accuracy. On the other hand, the geometric module retains valid static feature points to maintain system stability. In most cases, ORB-SLAM2 struggles to achieve reliable tracking due to the presence of dynamic objects. In scenes with a lower proportion of dynamic features, the results of YF-SLAM are similar to those of ORB-SLAM2, as the semantic module removes fewer dynamic feature points. The experimental results demonstrate that YF-SLAM exhibits reduced trajectory drift and outstanding performance in handling dynamic scenes. YF-SLAM and DYNA-SLAM both exhibit significant errors at the beginning of roll and pitch trajectory tracking, as object detection and feature filtering threads require several frames to initialize and converge to dynamic elements in the scene.

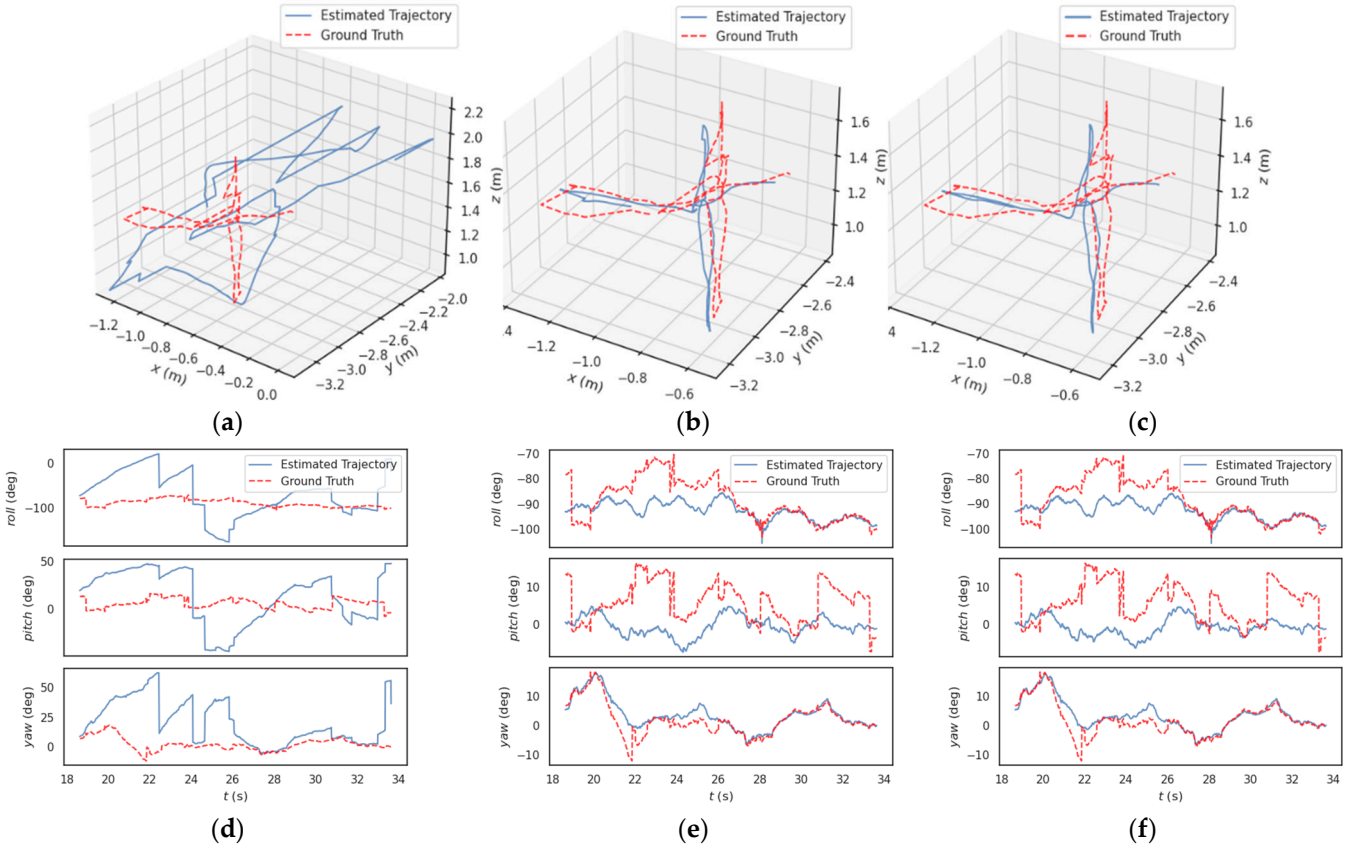


Figure 5. (**a**–**c**) shows the comparison of ORB-SLAM2, DYNA-SLAM, and YF-SLAM systems with the positioning trajectory and the real camera movement trajectory in the dynamic scene. (**d**–**f**) shows the deviation between the estimated values and the real values of roll, pitch, and yaw in the three-dimensional Cartesian coordinate system. The red line represents the ground truth, and the blue line represents the estimated trajectory.

4.2. Evaluation of Camera Localization

The accuracy of camera pose positioning is a key indicator for SLAM applications in positioning. In the experiments of each sequence, the estimated trajectories of different systems are compared with the ground truth values. ATE and RPE are often used as error metrics. The improvement effect of pose accuracy is illustrated by comparing the standard deviation data and root-mean-square deviation data of the system. The performance improvement effect is calculated according to (3):

$$Improve = \left(1 - \frac{E_{our}}{E_{ori}}\right) \times 100\% \tag{3}$$

where E_{ori} is the trajectory error of ORB-SLAM2, E_{our} is the trajectory error of YF-SLAM, and Improve is the optimization effect.

Table 2 shows that RANSAC, the method used by ORB-SLAM2, is effective in removing outliers in low-dynamic and static environments. In dynamic scenes, the ATE of YF-SLAM is much smaller than that of ORB-SLAM2, and the trajectory error of YF-SLAM is significantly reduced compared to ORB-SLAM2, with RMSE and SD improvement rates of up to 97.93% and 97.71%, respectively. Compared with DYNA-SLAM, YF-SLAM has an advantage in accuracy in high-dynamic sequence environments. We visually present the results of ATE's RMSE and SD indices for DS-SLAM, DYNA-SLAM, and YF-SLAM in Figure 6 using bar charts, demonstrating the Absolute Trajectory Error reaching millimeter- or centimeter-level accuracy in the sequences. Based on the RPE results in Tables 3 and 4, it

can be observed that the trend of error reduction is similar to ATE, performing exceptionally well in high-dynamic scenes with only slight limitations in low-dynamic scenes.

Table 2. Absolute Trajectory Error result table (ATE).

Sequence	ORB-SLAM2		DS-SLAM		DYNA-SLAM		YF-SLAM		Improve	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE (%)	S.D. (%)
fr3_walking_xyz	0.7214	0.2560	0.0247	0.0161	0.0164	0.0086	0.0156	0.0086	97.93	97.71
fr3_walking_half	0.4667	0.2601	0.0303	0.0159	0.0296	0.0157	0.0301	0.0157	93.55	93.96
fr3_walking_static	0.3872	0.1636	0.0081	0.0036	0.0068	0.0032	0.0067	0.0030	98.27	98.17
fr3_sitting_xyz	0.0092	0.0047	/	/	0.0127	0.0060	0.091	0.0049	/	/
fr3_sitting_half	0.0192	0.0110	/	/	0.0186	0.0086	0.0179	0.0075	6.77	31.82
fr3_sitting_static	0.0087	0.0042	0.0065	0.0033	/	/	0.0063	0.0032	27.59	21.43
Indoor scene	0.2618	0.1372	0.0117	0.0092	0.0165	0.0115	0.0096	0.0083	96.33	93.95

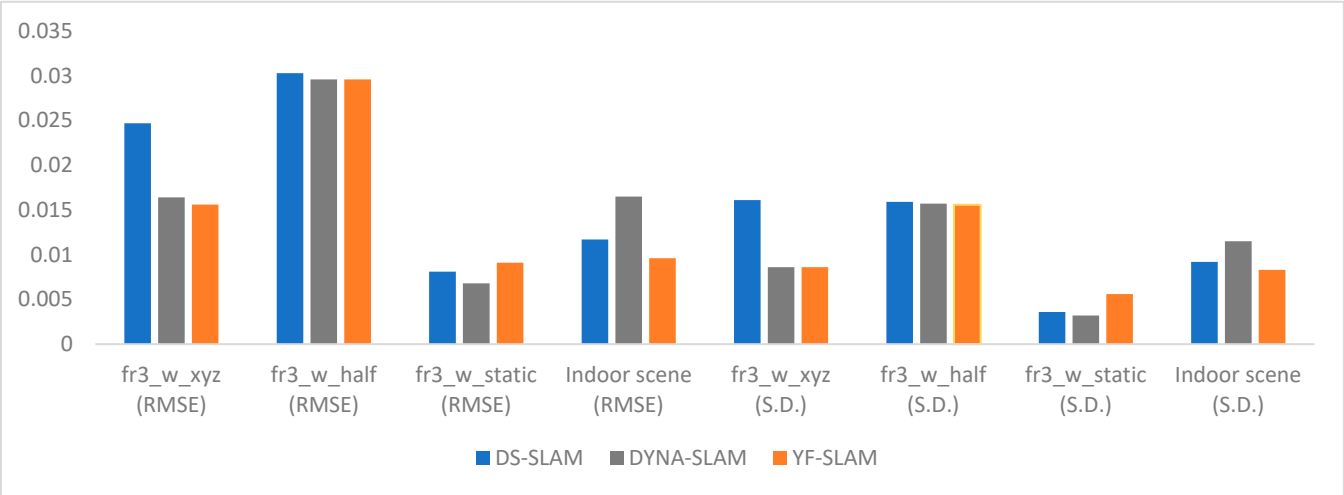


Figure 6. The index of Absolute Trajectory Error (ATE).

Table 3. Results of Metric Translation Drift (RPE).

Sequence	ORB-SLAM2		DS-SLAM		DYNA-SLAM		YF-SLAM		Improve	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE (%)	S.D. (%)
fr3_walking_xyz	0.3944	0.2964	0.0333	0.0229	0.0217	0.0119	0.0213	0.0109	94.60	96.32
fr3_walking_half	0.3480	0.2859	0.0297	0.0152	0.0284	0.0149	0.0277	0.0134	92.04	95.31
fr3_walking_static	0.2349	0.2151	0.0102	0.0048	0.0089	0.0044	0.0089	0.0064	96.21	97.02
fr3_sitting_xyz	0.0117	0.0060	/	/	0.0142	0.0073	0.0114	0.0055	2.56	8.33
fr3_sitting_half	0.0231	0.0163	/	/	0.0239	0.0120	0.0228	0.0160	1.30	95.92
fr3_sitting_static	0.0090	0.0043	0.0078	0.0038	/	/	0.0072	0.0035	20.00	18.60
Indoor scene	0.2961	0.1456	0.0201	0.0115	0.0172	0.0130	0.0121	0.0095	96.33	93.95

Table 4. Results of Metric Rotational Drift (RPE).

Sequence	ORB-SLAM2		DS-SLAM		DYNA-SLAM		YF-SLAM		Improve	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE (%)	S.D. (%)
fr3_walking_xyz	7.7846	5.8335	0.8266	0.5826	0.6284	0.3848	0.5997	0.3642	92.30	93.76
fr3_walking_half	7.2138	5.8299	0.8142	0.4101	0.7842	0.4012	0.7521	0.3576	89.57	93.87
fr3_walking_static	4.1856	3.8077	0.2690	0.1182	0.2612	0.1259	0.2610	0.1198	93.76	96.85
fr3_sitting_xyz	0.4890	0.2713	/	/	0.5042	0.2651	0.4844	0.2689	9.41	5.60
fr3_sitting_half	0.6015	0.2924	/	/	0.7045	0.3488	0.5971	0.2877	7.32	1.61
fr3_sitting_static	0.2850	0.1241	0.2735	0.1215	/	/	0.2725	0.1213	4.39	2.26
Indoor scene	4.7663	3.1141	0.3152	0.2089	0.3511	0.2787	0.2376	0.1208	95.02	96.12

4.3. Runtime Analysis

YF-SLAM's biggest advantage is its real-time performance, which has low CPU usage under real-time conditions. It is not only capable of achieving real-time performance on mobile devices with the Kirin 990 CPU tested, but it can also satisfy real-time requirements on low-cost, low-power devices such as the RK3399, Raspberry Pie, and various Cortex-A53 devices. This experiment ran YF-SLAM on a rescue robot equipped with a Jetson Nano. YOLO-FastestV2 can reach 38.4 FPS and an mAP of 24.1 on the Jetson Nano platform with objects of 80.

In order to evaluate the efficiency of the YF-SLAM system, we measured the average computation time for segmenting and tracking each frame and compared it with the state-of-the-art object detection-based DYNA-SLAM, DS-SLAM systems, and the benchmark ORB-SLAM2 system. DS-SLAM and DYNA-SLAM are also built on top of ORB-SLAM2. Table 5 shows that the improved YF-SLAM system only requires 10.12 ms for segmenting and tracking each frame, which is a significant improvement compared to DYNA-SLAM. DS-SLAM takes longer processing time when multiple dynamic points are present. The YF-SLAM system has strong robustness compared to other systems while ensuring fast performance.

Table 5. Time comparison table for each frame segmentation and tracking between YF-SLAM and similar systems on the Jetson Nano platform.

Framework	Network	Time (ms)		
		Segment	Track	Total
ORB-SLMAM2	/	/	24.66	37.96
DYNA-SLAM	Mask R-CNN	265	305	916.05
DS-SLAM	SegNet	42.11	30.54	98.63
YF-SLAM	YOLO-Fastest	10.12	26.05	57.11

4.4. Indoor Rescue Environment Assessment

The post-disaster environment dataset was captured using a calibrated Intel RealSense D435i depth camera. It contained 1000 images of search and rescue personnel and robots. The network training experiments were conducted on a desktop computer with an Intel I5-13400F processor and a GeForce GTX 1660 TI graphics card. The YF-SLAM system was deployed on a rescue robot platform powered by Jetson Nano. In the experiment, search and rescue personnel and rescue robots performed different activities, such as human–machine collaborative movement, moving static chairs, etc. The CPU running memory of VSLAM runtime is mainly used to store the data structures, variables, and temporary calculation results required for algorithm runtime. We use the Intel RealSense D435i depth camera as input to achieve real-time indoor synchronous positioning and mapping on the Jetson Nano rescue robot platform and complete rescue tasks.

Figure 7 shows the feature extraction results for the indoor dataset. The first row displays the detection results for search and rescue personnel, rescue robots, and dragged

chairs. The second row shows the feature points with dynamic individuals, robots, and occluded chairs removed while preserving the feature points from the static background within the rectangular bounding box. Under different indoor lighting conditions, search and rescue personnel, rescue robots, and potential dynamic objects can be clearly distinguished, and static feature points within the rectangular bounding box are also preserved. This further verifies the practicality of YF-SLAM in scenarios with known and unknown moving objects.

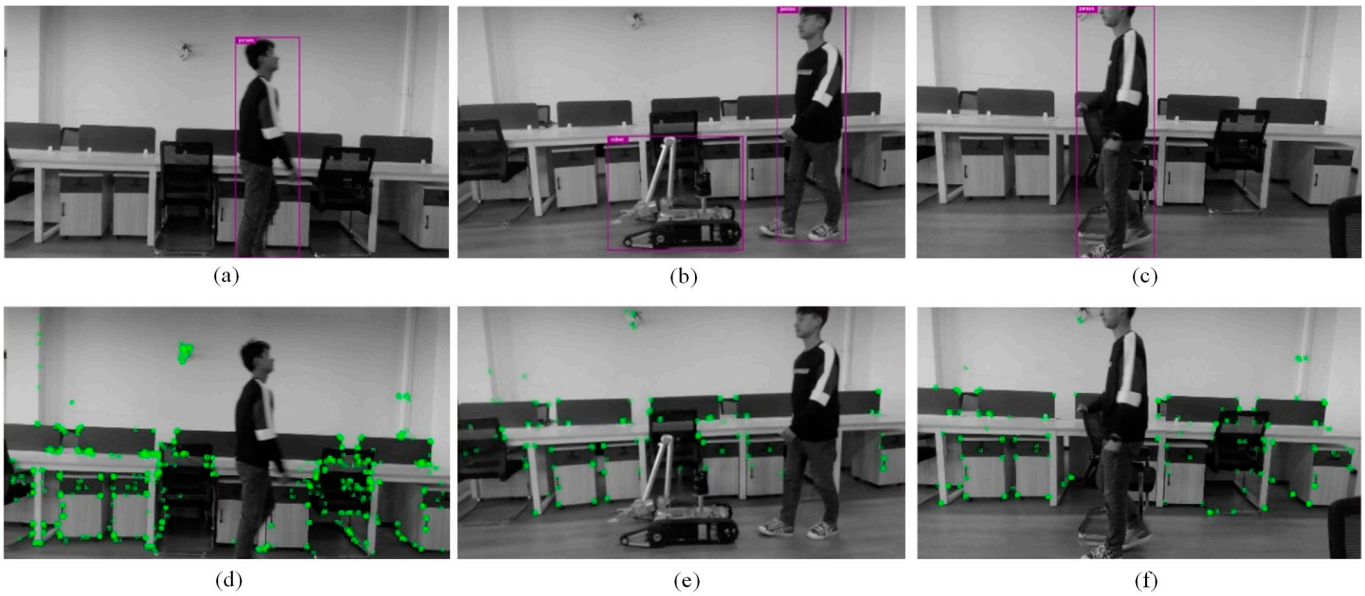


Figure 7. In the experiment on indoor datasets, the purple bounding box in (**a**–**c**) shows the position of dynamic objects identified by YOLO-FasterV2, including people and robots, and (**d**–**f**) shows the effect of YF-SLAM after removing dynamic feature points, including potential dynamic chairs in the box. Green dots are feature points.

5. Conclusions

This paper proposes a semantic SLAM system, called YF-SLAM, that works in dynamic environments. YF-SLAM is based on ORB-SLAM2 and primarily uses semantic information to assist the SLAM system in removing interference caused by moving objects. For the object detection thread, a lightweight version of YOLO-FastestV2 is proposed, which provides necessary semantic information in dynamic environments, thereby improving detection speed. Additionally, a deep feature point filtering method is proposed for the dynamic feature filtering thread, which retains static feature points extracted within the contours of dynamic objects. Based on the open TUM dataset and a self-made RGB-D indoor dynamic dataset, the proposed YF-SLAM achieves ATE errors that are reduced by over 90% compared to the original ORB-SLAM2 on most sequences. Compared with the currently advanced DYNA-SLAM and DS-SLAM systems, our approach also maintains its leading position in high dynamic sequences. On a rescue robot platform equipped with a Jetson Nano, the YF-SLAM system efficiently removes dynamic feature points and exhibits outstanding reliability in stability, accuracy, and speed under the walking sequence dataset where dynamic objects occupy a large part of the camera view. YF-SLAM greatly improves the operational capability of mobile robots in realistic dynamic environments.

There is still ongoing work in YF-SLAM. We will focus on improving its robustness under extreme conditions. Based on Figure 5, both YF-SLAM and DYNA-SLAM exhibit larger errors at the beginning of the trajectory tracking for roll and pitch. At the start, the detection bounding boxes may be inaccurate, letting through more unstable features and causing transient errors in pose estimation. As the system processes more frames, the detection and filtering improve, leading to more accurate tracking. This reveals an

intrinsic limitation of detection-based VSLAM methods, which require warm-up time for the semantic processing to stabilize. We will aim to address this issue in future work, for example, by propagating semantic predictions temporally to improve initialization.

Author Contributions: Software, M.F.; Validation, J.P.; Investigation, A.Z.; Writing—original draft, Z.S.; Writing—review & editing, W.S. and H.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received funding from the Major Research Program of National Natural Science Foundation of China (NSFC) under grant number 91948303, in part by the Hebei Science and Technology Innovation Foundation of China on Precise Identification Technology and Industrial Application of Small Targets in High-speed Movements. The article processing charges (APC) were covered by the Hebei University of Technology.

Data Availability Statement: Data associated with this article can be found in the online version, at https://blog.csdn.net/weixin_45947476.

Conflicts of Interest: All the authors do not have any possible conflict of interest.

References

1. Eason, G.; Noble, B.; Sneddon, I.N. On certain integrals of Lipschitz-Hankel type involving products of bessel functions. *Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Sci.* **1955**, *247*, 529–551. [CrossRef]
2. Scona, R.; Nobili, S.; Petillot, Y.R.; Fallon, M. Direct visual SLAM fusing proprioception for a humanoid robot. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Vancouver, BC, Canada, 24–28 September 2017; pp. 1419–1426. [CrossRef]
3. Fuentes-Pacheco, J.; Ruiz-Ascencio, J.; Rendón-Mancha, J.M. Visual simultaneous localization and mapping: A survey. *Artif. Intell. Rev.* **2015**, *43*, 55–81. [CrossRef]
4. Mur-Artal, R.; Tardos, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
5. Feng, X.-F.; Fang, B. Algorithm for epipolar geometry and correcting monocular stereo vision based on a plane mirror. *Optik* **2021**, *226*, 165890. [CrossRef]
6. Martínez-Otzeta, J.M.; Rodríguez-Moreno, I.; Mendialdua, I.; Sierra, B. RANSAC for Robotic Applications: A Survey. *Sensors* **2022**, *23*, 327. [CrossRef] [PubMed]
7. Kundu, A.; Krishna, K.M.; Sivaswamy, J. Moving object detection by multi-view geometric techniques from a single camera mounted robot. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009; Volume 2009, pp. 4306–4312. [CrossRef]
8. Zou, D.; Tan, P. CoSLAM: Collaborative Visual SLAM in Dynamic Environments. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 354–366. [CrossRef] [PubMed]
9. Kitt, B.; Moosmann, F.; Stiller, C. Moving on to dynamic environments: Visual odometry using feature classification. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 October 2010; pp. 5551–5556. [CrossRef]
10. Yu, C.; Liu, Z.; Liu, X.-J.; Xie, F.; Yang, Y.; Wei, Q.; Fei, Q. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1168–1174. [CrossRef]
11. Wang, X.; Oswald, M.R.; Cherabier, I.; Pollefeys, M. Learning 3D Semantic Reconstruction on Octrees. In *Pattern Recognition*; Springer: Cham, Switzerland, 2019. [CrossRef]
12. Badrinarayanan, V.; Kendall, A.; Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef] [PubMed]
13. Bescos, B.; Fácil, J.M.; Civera, J.; Neira, J. DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes. *IEEE Robot. Autom. Lett.* **2018**, *3*, 4076–4083. [CrossRef]
14. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R.B. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2980–2988.
15. Shi, J.; Zha, F.; Guo, W.; Wang, P.; Li, M. Dynamic Visual SLAM Based on Semantic Information and Multi-View Geometry. In Proceedings of the 5th International Conference on Automation, Control and Robotics Engineering (CACRE 2020), Dalian, China, 19–20 September 2020; pp. 671–679. [CrossRef]
16. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
17. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016.

18. dog-qiuqiu/YOLO-FastestV2: Based on YOLO's Low-Power, Ultra-Lightweight Universal Target Detection Algorithm, the Parameter is only 250k, and the Speed of the Smart Phone Mobile Terminal Can Reach ~300fps+. 2022. Available online: https://github.com/dog-qiuqiu/Yolo-FastestV2 (accessed on 14 July 2023).
19. Liu, G.; Zeng, W.; Feng, B.; Xu, F. DMS-SLAM: A General Visual SLAM System for Dynamic Scenes with Multiple Sensors. *Sensors* **2019**, *19*, 3714. [CrossRef] [PubMed]
20. Ngo, E.; Ramirez, J.; Medina-Soto, M.; Dirksen, S.; Victoriano, E.D.; Bhandari, S. UAV Platforms for Autonomous Navigation in GPS-Denied Environments for Search and Rescue Missions. In Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS), Dubrovnik, Croatia, 21–24 June 2022; pp. 1481–1488. [CrossRef]
21. 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. [CrossRef]