Semantic Segmentation-Based Enhancement of Visual SLAM Loop Closure Detection in Dynamic Indoor Environments

Lu Wang[1], Chao Hu[2,*], Xiaoxia Lu[3]

{dqx_wl@163.com[1], 2022107283@zut.edu.cn[2], lxxwl2008@163.com[3]}

School of Computer Science, Zhongyuan University of Technology, Zhengzhou, China[1]
School of Computer Science, Zhongyuan University of Technology, Zhengzhou, China[2]
School of Computer Science, Zhongyuan University of Technology, Zhengzhou, China[3]
*corresponding author

Abstract. Current visual SLAM loop closure detection algorithms encounter significant challenges in dynamic environments, where moving objects such as pedestrians lead to inconsistencies in feature points, compromising map accuracy. This study proposes a novel visual SLAM loop closure detection algorithm leveraging semantic segmentation, specifically designed for complex indoor dynamic scenarios. The proposed approach introduces the Bottleneck with Squeeze and Excitation Block (BnSEBlock) to improve the U-Net++ semantic segmentation model by incorporating residual connections, dilated convolutions, and an adaptive attention mechanism. Dynamic weights are assigned to semantic information based on motion intensity and centroid coordinates, which are derived through adaptive HDBSCAN clustering. Loop closure is identified by assessing the similarity between keyframes and candidate frames using these weighted parameters. Experimental evaluations on publicly available datasets demonstrate that the enhanced U-Net++ model achieves a Mean Intersection over Union (MIoU) of 76.9% and reduces the loss to 0.172. In comparison, the traditional bag-of-words-based approach yields a maximum similarity of 0.273 for loop images. The proposed algorithm shows a 61.57% improvement in localization accuracy within dynamic indoor environments.

Keywords: Loop closure detection, dynamic environment, semantic segmentation, motion intensity, centroid coordinate, dynamic weight allocation.

1 Introduction

Simultaneous Localization and Mapping (SLAM) [1] is a core robotics technology that enables robots to self-localize and map unknown environments. Visual SLAM, favored for its rich image information and cost-effective cameras, comprises sensor data, front-end, back-end, loop closure detection, and mapping (Fig. 1). Loop closure detection is critical for correcting cumulative errors, as its absence would lead to inaccuracies akin to those of an odometer [2], especially in long-term navigation [3]. Although graph optimization algorithms [4] can reduce incorrect loops' impact, they cannot eliminate errors or avoid computational overhead, underscoring the importance of loop closure in map optimization.

CONF-MLA 2024, November 21, Adana, Turkey

DOI 10.4108/eai.21-11-2024.2354632

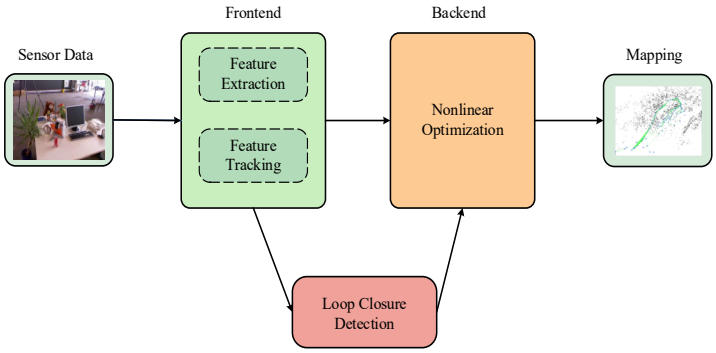


Fig. 1. Classic Visual SLAM Framework.

Recent research has focused on improving the performance of visual SLAM systems in dynamic environments, particularly in optimizing loop closure detection algorithms. Chen [5] proposed a Binocular SLAM system using an enhanced semi-global matching algorithm and superpixel segmentation to identify moving object boundaries, reducing dynamic object interference. However, small disparity differences between multiple moving objects can reduce boundary accuracy. Bojko et al. [6] trained networks using pre-generated dynamic object masks to differentiate static and dynamic elements, minimizing mismatches caused by dynamic objects, but the accuracy of mask generation can introduce errors in localization.

With advancements in deep learning and hardware, more studies have integrated object detection, semantic segmentation, and instance segmentation into visual SLAM to improve dynamic environment handling. Bescos et al. [7] introduced DynaSLAM, which uses Mask R-CNN [8] for semantic segmentation to exclude dynamic objects during loop closure detection. However, its high computational cost and time requirements limit real-time applicability. Yu et al. [9] developed DS-SLAM, using SegNet [10] for semantic segmentation to focus on static features for loop closure, but it also struggles with real-time performance. Zhang et al. [11] used instance segmentation and optical flow for object tracking in outdoor environments, but its reliance on lighting and weather conditions affects accuracy. Scona et al. [12] proposed StaticFusion, which reconstructs the background and isolates dynamic objects for more reliable mapping, though its performance is heavily dependent on data quality, particularly in low-light or low-quality camera conditions.

2 Visual Slam System Architecture for Enhanced Loop Closure Detection

The system framework of this paper is based on the classic open-source system ORB-SLAM2, which includes three parallel threads: the tracking thread, the local map thread, and the loop closure detection thread. While maintaining the original architecture of the system, a new semantic segmentation thread has been added as shown in Fig. 2. The purpose of this new semantic segmentation thread is to enhance the loop closure detection thread of the original system, enabling it to handle more complex environments.

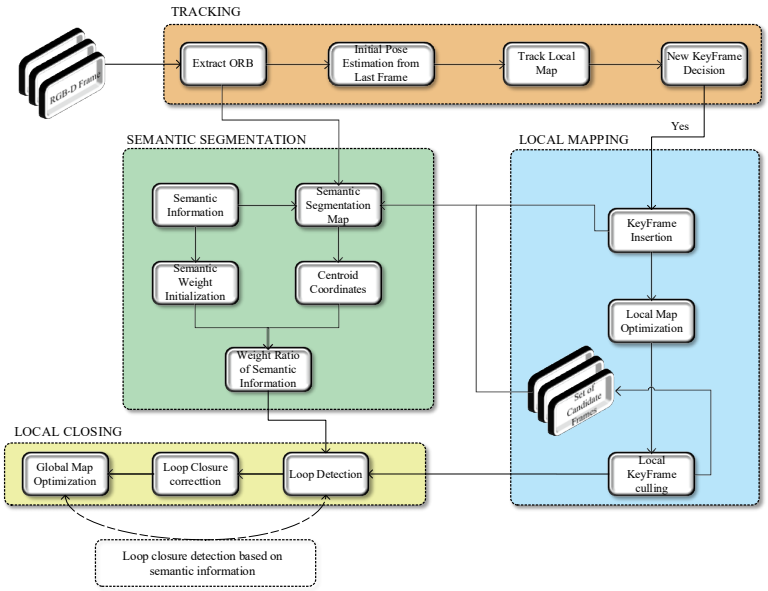


Fig. 2. Enhanced Loop Closure Detection Visual SLAM System Architecture Diagram.

3 The Semantic Segmentation Thread

3.1 Network Architecture of BnSEBlock

Convolutional Module of BnSEBlock. To replace the poorly performing VGGBlock [13] convolutional blocks in U-Net++, this paper designs a completely new BnSEBlock convolutional network. The convolutional structure of BnSEBlock is shown in Module 1 of Fig. 3, aiming to significantly enhance the model's feature extraction accuracy with only a small increase in parameters.

The BnSEBlock structure consists of four convolutional layers [14]. The first uses 512 1×1 kernels to expand the image channels from 3 to 512, enhancing feature representation and network expressiveness without significantly increasing parameters. The second layer dynamically adjusts the number of 1×1 kernels based on the downsampling level, using fewer kernels for shallow layers to capture low-level details and more for deeper layers to extract high-level semantics. This layer reduces feature map dimensions, lowering computational complexity while retaining key features.A 3×3 dilated convolution layer in the middle increases the receptive field without additional computational cost, preserving local details and capturing broader contextual information for large-scale structures. Finally, a 1×1 convolution layer adjusts the output channels to align with residual connections, ensuring efficient feature transfer, alleviating gradient vanishing, and enhancing the network's capacity to process complex data. Finally, a 1×1 convolution layer adjusts the output channels to align with residual connections, ensuring efficient feature transfer, alleviating gradient vanishing, and enhancing the network's capacity to process complex data.

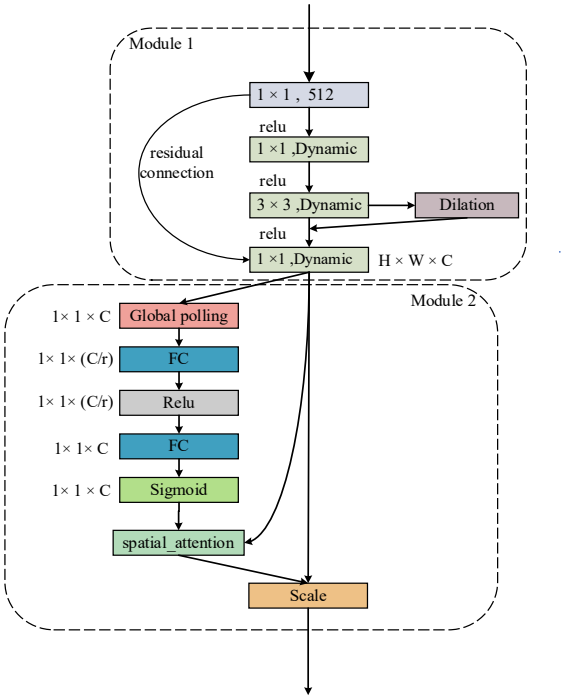


Fig. 3. BnSEBlock network structure.

Adaptive Attention Mechanism of BnSEBlock. Module 2 in Fig. 3 represents the adaptive attention mechanism of BnSEBlock. The key idea of this mechanism is to incorporate a channel-level attention module, which primarily includes dynamic Squeeze and Excitation.

The Squeeze operation reduces the spatial dimensions of the input feature map to 1×1 through global average pooling, compressing the features of each channel. This results in a vector Z for each channel of the input feature map X, representing the global features of that channel. As illustrated in Eq. (1), Z_c denotes the global feature of the c-th channel, and $X_c(i, j)$ represents the pixel value of the feature map at position (i, j).

$$Z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} X_c(i, j) \tag{1}$$

In Module 2, r is a hyperparameter used to dynamically control the dimensionality reduction operation in Excitation. The dimensionality reduction ratio r is predicted using Algorithm 1 based on the global feature vector Z.

Algorithm 1: DynamicRModule

Initialize: Set dropout probability p; Set channel reduction factor (h); Initialize trainable parameters

Compute: Global average pooling on input feature map x

While (DynamicRModule not converge) do

  For (each batch in x) do

    z = Global_Average_Pooling(x)

    z = Flatten(z)

    z = ReLU(Linear_Projection(z, in_features=C, out_features=C//h))

    z = Dropout(z, p)

    z = ReLU(Linear_Projection(z, in_features=C//h, out_features=C//h))

    z = Linear_Projection(z, in_features=C//h, out_features=1)

    r = Sigmoid(z)

  End For

End While

Return r

The excitation operation in the BnSEBlock uses two fully connected (FC) layers to generate a channel weight vector B, reducing and then restoring channel dimensionality, with a sigmoid activation constraining B to [0, 1]. By integrating spatial attention [15], the BnSEBlock surpasses SENet in capturing both channel and spatial relationships, adaptively adjusting attention weights to better handle input variations and disturbances. The spatial attention weight Spatial_Attention, calculated for the input feature vector A, contributes to the final module output, as shown in Eq. (2).

$$Output = \mathbf{A} \odot \boldsymbol{B} \odot Spatial\_Attention \tag{2}$$

3.2 Improved U-Net++ Model Structure

The basic encoding section of U-Net++ uses two convolution operations per module, sufficient for simple tasks like medical imaging but inadequate for feature-rich datasets, leading to low feature utilization and reduced segmentation accuracy. To integrate U-Net++ into ORB-SLAM2's loop closure detection module and meet its real-time demands, the BnSEBlock structure enhances feature extraction with minimal additional parameters. Fig. 4 illustrates the updated U-Net++ structure with BnSEBlock replacing the original convolution blocks.

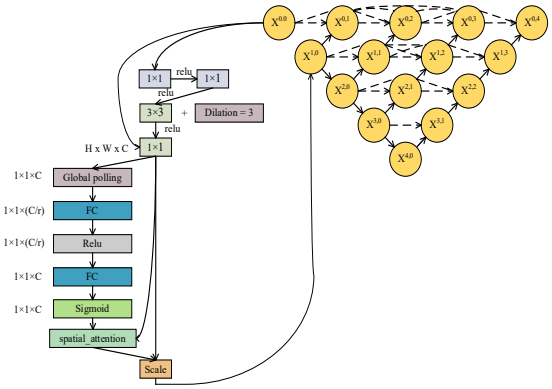


Fig. 4. Improved U-Net++ model structure.

4 Dynamic Update of Semantic Information Weights

4.1 Initialization of Semantic Information Weights

Based on the improved U-Net++ model, semantic segmentation information is obtained. Initial weights are assigned to each piece of semantic information according to its motion level, which is defined as follows:

$$Num\_class = \{person, cat, dog, sofa, \cdots\} \tag{3}$$

$$W_j = \{W_j^0, W_j^1, W_j^2, \cdots, W_j^{19}\} \tag{4}$$

$$W_j \in Candiframe\_class \tag{5}$$

The *Num_class* set consists of the 20 categories from the PASCAL VOC 2012 dataset, excluding background and object contour information. The number of weights W_j corresponds to the number of elements in the *Num_class*, representing the weight of each semantic category. These weights range from 0 to 10. The initial weights for each semantic category are presented in Table 1. Here, j represents the number of candidate frames formed by accumulating keyframes, with the keyframe data and their semantic segmentation information comprising the *Candidate_class* candidate frame collection.

Table 1. Initialization of Semantic Information Weights.

Semantic Information	Weights	Semantic Information	Weights
Aeroplane	0	Bicycle	3
Bird	0	Boat	0
Bottle	1	Bus	0
Car	0	Cat	0
Chair	4	Cow	0
Table	8	Dog	0

Horse	0	Motorbike	3
Person	0	Plant	4
Sheep	0	Sofa	9
Train	0	Monitor	3

4.2 Centroid coordinates of HDBSCAN clustering for masking semantic information in images

Before performing semantic segmentation on the PASCAL VOC 2012 dataset, the dataset underwent preprocessing. Each distinct color in the label images is represented by a tuple of three values (*R,G,B*) as shown in Eq. (6). The order in which each tuple appears is recorded as t . These tuples, along with their order of appearance, are stored as key-value pairs in the dictionary ω. The dictionary ω contains a total of 22 key-value pair entries.

$$\begin{cases} \beta_t = (R_t,\ G_t,\ B_t) \\ \{\ \beta_t\colon\ t\ \}\ \in \omega, \quad t \in [0\ ,21] \end{cases} \tag{6}$$

In the dictionary ω, there are 20 key-value pairs corresponding to specific semantic information, one pair for background information, and one for object contour information. The label files are transformed into grayscale images according to the following conversion rule: Each RGB tuple from the label image is matched against the tuples in the dictionary to find the corresponding key-value pair *t*. This value *t* is then used to set the grayscale value of the label image, resulting in a new grayscale image *gray_image*. This process scales the grayscale values in the PASCAL VOC 2012 dataset's label files to a range of 0 to 21, forming an array α with these 22 grayscale values.

A zero matrix *class_matrix* is created with the same shape as the label image (*H , W*), but with a third dimension of 22. The shape of the zero matrix is (*H, W, 22*). The third dimension of *class_matrix* is then iterated over, and the 22 matrices of shape (*H , W*) are assigned values from *gray_image*. Positions in *class_matrix* that match the grayscale values stored in the array α are set to 1, with all other positions set to 0. Thus, the third dimension of *class_matrix* essentially represents binary mask information for 22 semantic categories. Each binary mask is multiplied by 255 and saved as a grayscale image. However, only 21 categories including background are actually useful for the model, requiring the exclusion of contour class information, as it may interfere with model training.

After the aforementioned processing of the *class_matrix*, the resulting multi-channel image mask has a shape of (*H*, *W*, 21), where the third dimension indicates that there are 21 channels, each representing a different semantic category. Each source image corresponds to one multi-channel image mask. The semantic information δ that appears on the source image is marked as a hit in the corresponding channel of the third dimension of its multi-channel image mask. At this time, the channel image has two colors: black and white, where the white part of the image represents the semantic information δ. Semantic information that does not appear is indicated as a miss in the corresponding channel of the mask, where the channel image is entirely black.

The mask of the multi-channel image contains semantic information in two colors: black and white. However, there are still noise points that interfere with the data. The HDBSCAN

clustering algorithm is effective in removing noise points from the semantic segmentation map, allowing for more accurate calculation of the centroid coordinates of each type of semantic information.

For a given channel of the semantic segmentation map, the set of all points containing semantic information *c* forms the point set S_c, as shown in Eq. (7). To apply the HDBSCAN clustering algorithm to the point set S_c, the parameters need to be defined: *minPts* specifies the minimum number of neighbors required to define a core point, and *epsilon* represents the maximum distance threshold. HDBSCAN is then applied to the point set S_c as described in Eq. (8), where *Clusters* includes the clustering labels L_j for each point and noise markers. Noise points are labeled as -1. After clustering, noise points are removed from the results, retaining only the points labeled as valid clusters, as shown in Eq. (9).

$$S_c = \{(x_i, y_i) \mid I_c(x_i, y_i) = t\} \tag{7}$$

$$Clusters = HDBSCAN(S_c, minPts, epsilon) \tag{8}$$

$$S_c' = \{(x_i, y_i) \in S_c \mid L_j \neq -1\} \tag{9}$$

HDBSCAN partitions these data points into different clusters. Each cluster i has a unique label Li.The label Lj of of each point j indicates the cluster to which point j belongs. For each cluster i, the centroid coordinates Ci are computed as shown in Eq. (10), where Clusteri represents the set of all points belonging to cluster i, Ni is the number of points in cluster i, and (xi , yi) are the coordinates of point j.

$$C_i = (\vec{x}_i, \vec{y}_i) = (\frac{1}{N_i} \sum_{j \in Cluster_i} x_j, \frac{1}{N_i} \sum_{j \in Cluster_i} y_j) \tag{10}$$

Each cluster's centroid Ci serves as the basis for calculating the weighted average, where the weight is the number of points Ni in each cluster. The coordinates of the weighted average representative point are computed according to Eq. (11).

$$C_{weighted} = \left(\frac{\sum_{i=1}^{k} N_i \cdot \vec{x}_i}{\sum_{i=1}^{k} N_i}, \frac{\sum_{i=1}^{k} N_i \cdot \vec{y}_i}{\sum_{i=1}^{k} N_i}\right) \tag{11}$$

4.3 Semantic Information Weight Update Strategy

Based on the dictionary information described above, the Euclidean distance L_n between the centroid coordinates of the same semantic categories is calculated as shown in Eq. (12), where *n* represents the number of semantic categories that are the same between the two frames. *i* denotes the order of the channels, and *j* represents the number of mask images. $H_{keyframe}$ and $H_{candidate}$ respectively represent dictionaries of key-value pairs, where the keys are composed of

channel order and centroid coordinates from HDBSCAN clustering, and the values correspond to the keyframes and candidate frames.

$$\begin{cases} i(H_{keyframe}) = i(H_{candiframe}) \\ L_n = \left\| H_{keyframe}[i] - H_{candiframe}[i] \right\|_2 \\ M_{keyframe_j} = round(W_{keyframe_j}^{i} \times \left\| (X_{keyframe_j}^{i}, Y_{keyframe_j}^{i}) - (X_{candiframe_j}^{i}, Y_{candiframe_j}^{i}) \right\|_2, 2) \end{cases} \tag{12}$$

For semantic information present only in the candidate frame, if the initial weight is greater than 5, it is also determined that there is no loop closure between the two images. For semantic information with an initial weight of 5 or less, this category of semantic information, which originally does not exist in the keyframe, is virtually created, and the weight for this semantic category in both the keyframe and candidate frame is set to the initial weight.

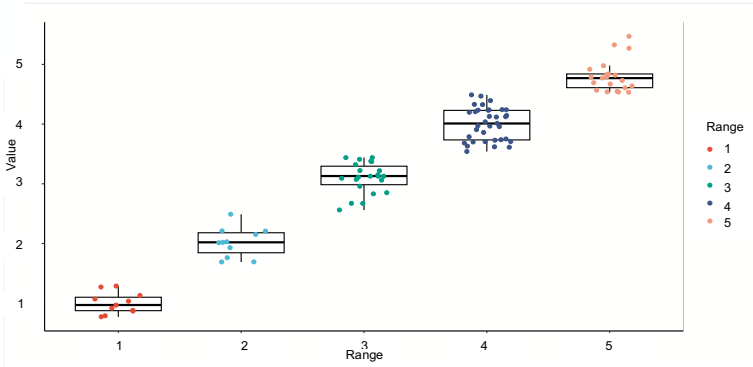


Fig. 5. Distribution of Similarity.

If the keys are the same, the Euclidean distance Ln between the centroid coordinates of the same semantic category in the compared frame and candidate frame is calculated after appropriately scaling the images based on their size. For the experiments, the image size of 640×480 is scaled down to 64×48. According to Eq. (12), the Euclidean distance Ln between the centroid coordinates of the two frames is calculated. If Ln is greater than 1, the weight of the semantic information on the keyframe is updated to Mkeyframe. If Ln is less than or equal to 1, the weights for the semantic information on both the keyframe and candidate frame are set to zero.

Select 20 pairs of loop-closing images from each dataset within the fr3 sequence: walking_rpy, walking_static, sitting_xyz, walking_halfsphere, and walking_xyz.Calculate the similarity between these 100 pairs of loop-closing images using Eq. (13). The experimental results, as depicted in Fig. 5, show that 97% of the similarity values between loop-closing images are distributed within the range of 0 to 5. Thus, set the threshold σ to 5 and determine if a loop has occurred based on whether the ratio calculated from semantic weight information is within this threshold.Additionally, to maintain consistency with the similarity assessment method used in

the original ORB-SLAM2 system's loop closure detection algorithm, this paper transforms into a similarity measure using the Gaussian formula Eq. (14), denoted as *Similarity*.

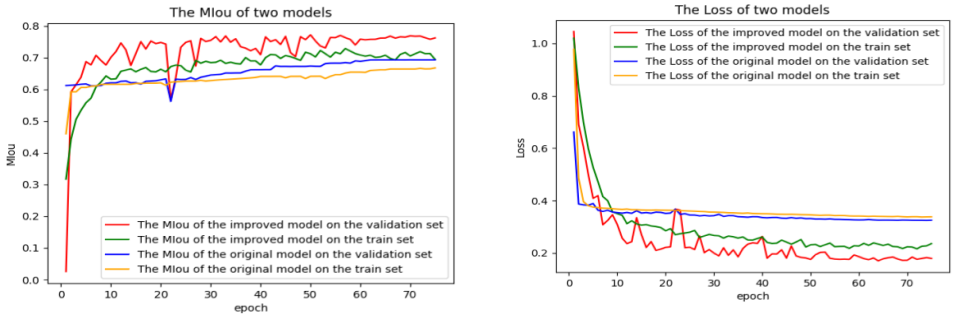
$$\eta = \frac{round(\sum_{k=1}^{20}(M_{keyframe_k}),2)}{round(\sum_{k=1}^{20}(M_{candiframe_k}),2)} \tag{13}$$

$$Similarity = e^{-\frac{\eta^2}{2\sigma^2}} \tag{14}$$

5 Experiments and Analysis

5.1 Performance Testing of the Improved U-Net++ Model

To validate the effectiveness of the improved U-Net++ model, we compare it with the U-Net++ model using the publicly available PASCAL VOC 2012 dataset. This dataset contains a total of 1,464 training sets and 1,449 validation sets specifically designed for semantic segmentation tasks.



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

Fig. 6. MIoU and Loss of Two Models on the PASCAL VOC 2012 Dataset.

Fig. 6 illustrates the performance comparison between the original U-Net++ model and the improved model. In Fig. 6(a), the improved model achieved higher MIoU scores of 71.8% (training set) and 76.9% (validation set) compared to the original model's 66.7% and 69.2%, respectively. Fig. 6(b) shows that the improved model reduced loss values to 0.341 (training set) and 0.319 (validation set), compared to 0.216 and 0.179 for the original model. The improvement resulted in a 5.1 percentage point increase in MIoU on the test set, with a loss reduction of 0.125. On the validation set, MIoU increased by 7.7 percentage points, and loss decreased by 0.140, highlighting the effectiveness of the proposed enhancements.

Table 2. Comparison of Six Classic Models on PASCAL VOC 2012.

Model Name	MIoU	Time Consumption
FCN	59.3%	404ms
SegNet	56.7%	68ms
DeepLabV3	**74.2%**	179ms
DFN	78.3%	1248ms
U-Net++	69.2%	**52ms**
Ours	**76.9%**	**56ms**

Table 2 presents the comparative results of the improved U-Net++ model and four other semantic segmentation models (FCN [16], Segnet, DeeplabV3, DFN [17]) on the PASCAL VOC 2012 dataset.The improved U-Net++ model is second only to DFN in terms of MIoU metrics, but DFN takes more than three times the processing time per frame compared to the algorithm in this paper, which is detrimental to the real-time performance of the ORB-SLAM2 system.Compared to the U-Net++ model, the processing time per frame for the improved model in this paper only increases by about 4ms, but the MIoU accuracy improves to 76.9%.Therefore, in situations where computational resources are limited or real-time performance is crucial, the model in this paper is more suitable.

5.2 Performance Testing of the Improved Loop Closure Detection Algorithm

Comparative experiment with ORB-SLAM2 algorithm. To validate the proposed algorithm, its performance is compared with the original ORB-SLAM2 using highly dynamic datasets. Fig. 7 highlights loop images from the sitting_xyz, walking_xyz, and walking_halfsphere datasets, with Table 3 showing their center-of-mass coordinates and Euclidean distances. Table 4 presents similarity results from the proposed loop detection algorithm, while Table 5 shows results from ORB-SLAM2's bag-of-words-based method. Keyframes and candidate frames (Images1-6) represent loop relationships across the datasets. The proposed algorithm achieves a maximum similarity of 0.872, significantly outperforming ORB-SLAM2's 0.273, demonstrating its robustness in dynamic indoor environments.

1

3

5

2

4

6

Fig. 7. Loop-closing images.

Table 3. Centroid coordinates and Euclidean distance of the fr3 sequence loop images.

fr3 sequence	Centroid coordinates	Candidate frame	Keyframe	Euclidean distance
sitting_xyz(Image1,2)	Person	0	0	0
	Monitor	(16.43,41.46)	(14.86,39.74)	2.33
	Chair	(34.32,17.76)	(31.76,14.323)	4.29
	Table	(20.48,34.76)	(18.78,35.32)	1.79
walking_xyz(Imag3,4)	Person	0	0	0
	Monitor	(27.74,34.25)	(29.82,37.14)	3.56
	Chair	(45.52,42.91)	(41.54,39.26)	5.40
	Table	(31.46,29.38)	(29.72,27.31)	2.71
walking_halfsphere(Imag5,6)	Person	0	0	0
	Monitor	(14.23,39.61)	(12.73,41.82)	2.67
	Chair	(36.91,20.94)	(33.57,17.15)	5.06
	Table	(25.32,31.33)	(23.56,27.37)	4.34

Table 4. Similarity between fr3 sequence loop photos (algorithm in this paper).

fr3 sequence	Semantic information weights	Candidate frame	Keyframe	η	Similarity
sitting_xyz(Image1,2)	Person	0.00	0	2.62	**0.872**
	Monitor	3.00	6.99		
	Chair	4.00	17.16		
	Table	7.00	12.53		
walking_xyz(Image3,4)	Person	0.00	0	3.66	0.766
	Monitor	3.00	10.68		
	Chair	4.00	21.60		
	Table	7.00	18.97		
walking_halfsphere(Image5,6)	Person	0.00	0	4.17	0.707
	Monitor	3.00	8.01		
	Chair	4.00	20.04		
	Table	7.00	30.38		

Table 5. Similarity between fr3 sequence loop closure photos (ORB-SLAM2).

ORB-SLAM2	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6
Image 1	1.000	0.237	0.239	0.214	0.229	0.215
Image 2	0.237	1.000	0.235	0.255	0.240	0.221
Image 3	0.239	0.235	1.000	0.262	0.221	0.236
Image 4	0.214	0.255	0.262	1.000	0.237	0.231
Image 5	0.229	0.240	0.221	0.237	1.000	**0.273**
Image 6	0.215	0.221	0.236	0.231	**0.273**	1.000

Table 6 compares the relativerotational trajectory errors of the fr3 sequences.The dynamic amplitude of the sitting_xyz dataset is significantly smaller than that of the other datasets. In this case, the semantic segmentation thread may exclude some static semantic information, leading to slightly lower trajectory accuracy in the improved ORB-SLAM2 compared to the original ORB-SLAM2.However, in high-dynamic scenarios such as walking_xyz and walking_halfsphere, the accuracy of loop closure detection has significantly improved. In high-dynamic scenarios, the root mean square error of the rotational component has improved by 80.61% to 87.16%. The improved system shows an average accuracy increase of 61.57% across five test datasets.

Table 6. Comparison of Relative Rotational Trajectory Errors for the fr3 Sequences.

fr3 sequence	Original ORB-SLAM2			Improved ORB-SLAM2 algorithm			Improvement Margin(%)		
	Mean	Median	RMSE	Mean	Median	RMSE	Mean	Median	RMSE
walking_rpy	5.672	3.513	9.342	**0.972**	**0.831**	**1.348**	82.80	76.28	85.84
walking_static	4.713	2.437	6.383	**0.673**	**0.654**	**0.817**	85.69	73.21	87.16
sitting_xyz	**0.441**	0.38	**0.547**	0.473	**0.373**	0.573	-7.26	1.84	-4.75
walking_halfsphere	6.792	4.475	9.853	**1.743**	**0.972**	**1.913**	74.38	78.20	80.61
walking_xyz	5.571	3.593	7.463	**0.975**	**0.893**	**1.136**	82.43	75.16	84.77

Comparative experiments with other SLAM algorithms. To further validate the superiority of the proposed algorithm, a comparison is conducted between the proposed algorithm and other state-of-the-art algorithms designed for dynamic environments, such as DS-SLAM and Detect-SLAM. The comparison results are presented in Table 7, where "-" indicates that the respective study did not provide pose accuracy analysis for the corresponding dataset. According to the comparison, the improved ORB-SLAM2 system in this paper performs well on most datasets.However, since DS-SLAM relies on optical flow for dynamic element removal, its performance may vary significantly across different datasets.

Table 7. Comparison of ATE between other advanced dynamic SLAM algorithms and the proposed algorithm

fr3 sequence	DS-SLAM		The algorithm in [18]		Detect-SLAM[19]	The algorithm in [20]	Improved ORB-SLAM2 algorithm	
	RMSE	Median	RMSE	Median	RMSE	RMSE	RMSE	Median
walking_rpy	0.444	0.283	0.133	0.083	0.296	0.097	**0.039**	**0.027**
walking_static	**0.008**	**0.006**	0.066	0.023	-	0.013	0.011	0.009
sitting_xyz	-	-	0.048	0.033	0.021	0.019	**0.017**	**0.014**
walking_halfsphere	**0.047**	**0.034**	0.125	0.067	0.054	0.029	0.057	0.051
walking_xyz	0.025	0.019	0.137	0.073	0.024	0.035	**0.019**	**0.016**

6 Conclusion

To enhance the loop closure detection algorithm of traditional visual SLAM systems in dynamic scenes, this paper proposes an improved visual SLAM loop closure detection algorithm integrated with semantic information based on the ORB-SLAM2 system, which performs well in recognizing loops in indoor dynamic environments. By leveraging the newly proposed BnSEBlock network, the U-Net++ model is enhanced to improve its semantic segmentation capabilities. The optimized loop closure detection algorithm, based on the improved U-Net++ model, initializes weights according to the motion levels of different semantic information. These weights are then dynamically updated based on the Euclidean distance of the centroids of HDBSCAN clusters of similar semantic information. Finally, comparisons are made using the TUM public dataset. The results show that, compared to the bag-of-words-based loop closure detection methods that perform poorly in dynamic scenes, the proposed algorithm effectively identifies loops in indoor dynamic environments. In the next phase, we will further leverage semantic information in visual SLAM systems and explore constructing semantic octree maps to support higher-level applications such as robot path planning and obstacle avoidance.

References

[1] Chen W, Shang G, Ji A, et al. An overview on visual slam: From tradition to semantic[J]. Remote Sensing, 2022, 14(13): 3010.

[2] Tourani A, Bavle H, Sanchez-Lopez J L, et al. Visual slam: What are the current trends and what to expect?[J]. Sensors, 2022, 22(23): 9297.

[3] Kazerouni I A, Fitzgerald L, Dooly G, et al. A survey of state-of-the-art on visual SLAM[J]. , 2022, 205: 117734.

[4] Cattaneo D, Vaghi M, Valada A. Lcdnet: Deep loop closure detection and point cloud registration for lidar slam[J]. IEEE Transactions on Robotics, 2022, 38(4): 2074-2093.

[5] Chen L, Fan L, Xie G, et al. Moving-object detection from consecutive stereo pairs using slanted plane smoothing[J]. IEEE Transactions on Intelligent Transportation Systems, 2017, 18(11): 3093-3102.

[6] Bojko, Adrian, et al. "Learning to segment dynamic objects using SLAM outliers." 2020 25th International Conference on Pattern Recognition (ICPR). IEEE, 2021.

[7] Bescos B, Cadena C, Neira J. Empty cities:A dynamic-objectinvariant space for visual SLAM[J]. IEEE Transactions on Robotics, 2021, 37(2): 433-451.
[8] He K M, Zhang X Y, Ren S Q, et al. Deep residual learning for image recognition[C]//IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, USA:IEEE, 2016:770-778.
[9] Yu C, Liu Z X, Liu X J, et al. DS-SLAM:A semantic visual SLAM towards dynamic environments[C]//IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, USA:IEEE, 2018:1168-1174.
[10] Badrinarayanan V, Kendall A, Cipolla R. SegNet:A deep convolutional encoder-decoder architecture for image segmentation[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(12): 2481-2495.
[11] Zhang J, Henein M, Mahony R, et al. VDO-SLAM:A visual dynamic object-aware SLAM system[DB/OL]. (2020-05-22)[2021-08-01]. https://arxiv.org/abs/2005.11052.
[12] Scona R, Jaimez M, Petillot Y R, et al. Staticfusion: Background reconstruction for dense rgb-d slam in dynamic environments[C]//2018 IEEE international conference on robotics and automation (ICRA). IEEE, 2018: 3849-3856.
[13] Wang Q, Park Y, Lu W D. Device Variation Effects on Neural Network Inference Accuracy in Analog In‐Memory Computing Systems[J]. Advanced Intelligent Systems, 2022, 4(8): 2100199.
[14] He K, Zhang X, Ren S, et al. Identity mappings in deep residual networks[C]//Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14. Springer International Publishing, 2016: 630-645.
[15] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. Advances in neural information processing systems[J]. Advances in neural information processing systems, 2017, 30(2017).
[16] Siano P, Cecati C, Yu H, et al. Real time operation of smart grids via FCN networks and optimal power flow[J]. IEEE Transactions on Industrial Informatics, 2012, 8(4): 944-952.
[17] Yu C, Wang J, Peng C, et al. Learning a discriminative feature network for semantic segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 1857-1866.
[18] Chen L, Fan L, Xie G, et al. Moving-object detection from consecutive stereo pairs using slanted plane smoothing[J]. IEEE Transactions on Intelligent Transportation Systems, 2017, 18(11): 3093-3102.
[19] Zhong F, Wang S, Zhang Z, et al. Detect-SLAM: Making object detection and SLAM mutually beneficial[C]//2018 IEEE winter conference on applications of computer vision (WACV). IEEE, 2018: 1001-1010.
[20] Ai Qinglin, Liu Gangjiang, Xu Qiaoning. Robot RGB-D SLAM algorithm based on improved geometric and motion constraints in dynamic environment[J]. Robotics, 2021, 43(2): 167-176.