IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 17 April 2023, accepted 4 May 2023, date of publication 12 May 2023, date of current version 18 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3275739

RESEARCH ARTICLE

SP-VO: RGB-D Visual Odometry Using Static Parts Toward Dynamic Environments

HYEONGJUN JEON AND JUNGHYUN OH, (Member, IEEE)
Department of Robotics, Kwangwoon University, Seoul 01890, South Korea

Corresponding author: Junghyun Oh (jhyunoh@kw.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) funded by the Korean Government (MSIT) under Grant 2022R1A4A3033961 and Grant 2020R1F1A1076667; and in part by the Research Grant of Kwangwoon University, in 2021.

ABSTRACT Estimating visual odometry in dynamic environments is a challenging problem, as features of moving objects prevent accurate image matching. Typical approach to deal with dynamic environments is to remove features of moving objects. However, this results in a lot of information loss and makes it impossible to use static parts of moving objects that could be fully utilized for image matching process. To address this issue, we propose a geometrical inference approach that utilizes the static parts of moving objects and background to achieve accurate feature matching. Moreover, we propose the concept of matching confidence that is calculated by comparing the squared residual motion likelihood with the chi-square distribution. For each frame, the geometric model and the semantic model are selected according to the proposed confidence, so that visual odometry could be estimated more accurately. Our algorithm was evaluated on RGB-D datasets, including dynamic environments. The results show better performance than prior algorithms.

INDEX TERMS Visual odometry, robotics, computer vision, mobile robots, SLAM.

I. INTRODUCTION

Accurate odometry estimation is one of the fundamental techniques for Simultaneous Localization And Mapping (SLAM) and path planning as it helps determine the location of a mobile robot. In particular, odometry estimation using visual information has been studied extensively because cameras provide abundant information about their surroundings and are reasonably price. Many algorithms [1], [2] have been examined for static environments and have been demonstrated in various areas. However, in environments with moving objects, static constraints are not maintained by these dynamic objects. Despite optimization techniques and outlier removal, performance degradation by dynamic objects has not been significantly improved.

Many algorithms have been proposed for dynamic environments. Some of these algorithms [3], [4], [5], [6], [7] use neural networks to recognize predefined dynamic objects (such as people, bicycles, and cars) and additional dynamic objects (such as chairs or books carried by people) based on geometric constraints. In the pre-processing step of tracking in SLAM, the recognized dynamic objects are removed as outliers. Other recent algorithms [8], [9] detect the semantic type of objects based on neural networks and try to use some movable objects if they are stopped. However, these methods do not distinguish between static and dynamic parts of moving objects, which leads to errors in pose estimation. Furthermore, other algorithms [10], [11], [12] attempt to exploit the object parts did not try to consider the confidence of their geometric inference, which can sometimes lead to estimation failures.

We propose a RGB-D Visual Odometry (VO) algorithm for accurate estimation of camera motion in dynamic environments. The proposed algorithm uses a combination of the Conditional Random Field (CRF) model, deep neural networks, and scene flow to recognize static objects and static parts present in dynamic objects such as Fig. 1. The proposed algorithm recognizes static objects and static parts present in dynamic objects and serves as a pre-processor of keypoint extraction for VO. It calculates the geometry model dynamics probability, which means that a particular part in the scene is moving, as a geometric inference using geometric constraints

The associate editor coordinating the review of this manuscript and approving it for publication was Gustavo Olague.

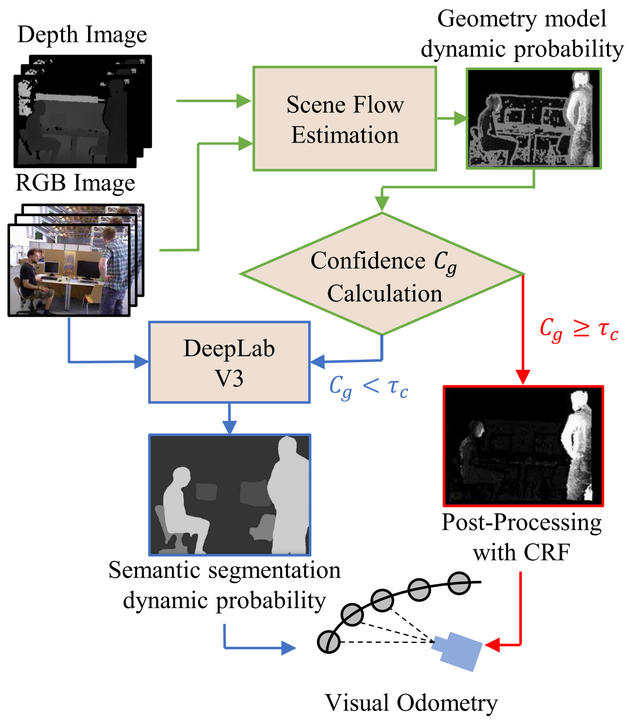

FIGURE 1. Framework of SP-VO. The proposed algorithm calculates the scene flow from the RGB and depth images in consecutive scenes and the geometry model dynamic probability. We computed the confidence C_g for that probability. If C_g is greater than τ_c, we use the CRF model to recognize the dynamic part. In the opposite case, we use DeepLab V3 to recognize it. The dynamic part is excluded from VO.

based on scene flow. Furthermore, the proposed algorithm obtains the confidence C_g about the inference. If C_g is greater than threshold τ_c, the probability is modified using the CRF model. In the opposite case, we calculate the semantic segmentation dynamic probability using the semantic segmentation model DeepLab V3 [13]. Keypoints are extracted only from the remaining parts with low probability of motion, and used for VO.

Proposed algorithm has the following main contributions:

- It uses more keypoints than prior algorithms, which use only static parts within static objects. This contributes to estimating an accurate camera relative pose.
- Unlike prior algorithms, it recognizes pixel-wise dynamic parts based on geometric inference and employs the CRF model. By using CRF model, our algorithms is robust for recognizing dynamic parts around boundaries between objects or occluded dynamic ones in consecutive scenes.
- It calculates the confidence of the geometric inference and recognizes the dynamic parts using a semantic segmentation model if the confidence is too low. The confidence calculation makes us deal with motion blur caused by fast movement of the camera and uncertainty of relative pose estimation.

The rest of this paper is structured as follows: Section II explains many visual SLAM algorithms that are robust to dynamic environments and how moving objects are used for VO in other algorithms. Then section III interprets processing of the scene flow to recognize static parts inside the objects and how to employ these parts for VO. Subsequently, section IV provides evaluations of the proposed and prior algorithms and demonstrates the reason for the performance improvement with additional experiments. Section V discusses our algorithm, its performance, limitations, and future work. Section VI presents the conclusion and future plans.

II. RELATED WORK

A. DYNAMIC SLAM BASED ON GEOMETRIC METHODS FOR VO

Before deep neural network models are used to get semantic information, many SLAM algorithms [14], [15], [16], [17] were developed toward dynamic environments. D.-H. Kim & Kim [14] proposed an algorithm that estimates camera pose and nonparametric background model by calculating energy function from continuous RGB-D images. The algorithm was combined with DVO SLAM [1] as a preprocessor. For calculating a camera ego-motion and detecting a dynamic object, Sun et al. [15] performed a particle-based tracking and MAP algorithm using perspective transformation between continuous RGB-D images. However, the algorithm was able to find only one dynamic area within the scene. Moreover, because of inaccurate image segmentation running K-Means, there was a limitation in that it could recognize incorrect dynamic areas. Li and Lee [16] found correspondences of depth edge points between continuous images and estimated camera pose toward dynamic environments. Wang et al. [17] clustered 3D points calculated by RGB-D camera to areas using K-Means and calculated the change in the number of inlier keypoints in the areas with epipolar constraint between consecutive scenes. The areas with small numbers of inliers were regarded as dynamic ones. Xu et al. [18] constructed triangles using features and calculated whether the features composing the triangles were dynamic by computing the changes of the triangles between keyframes.

There were some algorithms [10], [12], [19], [20], [34] using optical flow to detect dynamic parts. Namdev et al. [20] calculated the motion potential of scene using optical flow of mono camera images. The motion potential contributed to detecting dynamic parts of the image by using graph-based clustering. Alcantarilla et al. [10] estimated scene flow of the 3D points from optical flow and calculated residual motion likelihood considering noise of stereo camera. Keypoints extracted in areas of low likelihood were used for SLAM. Jaimez et al. [11] ran geometric clustering using K-Means in an RGB-D scene and separated the scene into regions as rigid bodies. They determined whether each region is static using a scene flow. The static ones were applied as background to estimate camera poses. Hempel and Al-Hamadi [12] generated a pixel-wise dynamic segment mask by using optical flow and homography matrix between continuous scenes. The mask was applied as pre-processor of ORB-SLAM2 [2]. Because these algorithms based on optical flow were not able to deal with noise, uncertainty in boundaries of objects, and occlusion, they had limitations on detecting dynamic parts. Jeon et al. [34] tried to deal with noisy scenes and estimate confidence of ego-motion using only CRF. However, when noisy scenes persisted for a long time, the dynamic masks created by their algorithms did not clearly distinguish the dynamic parts.

B. DYNAMIC SLAM USING DEEP NEURAL NETWORKS FOR VO

There were many SLAM algorithms [3], [4], [5], [6], [7] using deep neural networks to detect and remove movable objects for VO in dynamic scenes. DS-SLAM [3] based on ORB-SLAM2 is a keypoint-based SLAM that recognizes dynamic keypoints with semantic information using Seg-Net [21] and calculates moving consistency check based on epipolar constraint. Similarly, DynaSLAM [4] detects dynamic objects using Mask R-CNN [22], which is the semantic segmentation model. By calculating a depth change of keypoints between consecutive scenes, this algorithm also determines whether the keypoints in a scene are dynamic ones. The object evaluated as dynamic is replaced with background image parts in image inpainting. For crowded environments, Soares et al. [5] proposed Crowd-SLAM based on a specialization model of YOLOv3 [23]. The SLAM can deal with the situation when people take up most of the scene, but frequently causes tracking lost when the bounding boxes of people are too big to occlude the scene. For real-time SLAM using semantic segmentation models with high time-consuming, Liu and Miura [6] proposed RDS-SLAM. The framework of RDS-SLAM based on ORB-SLAM3 [24] recognizes keypoints of movable objects by initializing the moving probability of ones using segmentation results of the models, propagating the probability to following frames. By assigning a robust weight to keypoints of the scene, Wen et al. [7] detected dynamic objects using Mask R-CNN and built a map containing semantic information. These algorithms use insufficient keypoints for accurate VO because movable objects are removed in the overall SLAM process. The performance of these algorithms also decreased when the objects that the model can not recognize move in dynamic environments.

Some algorithms [25], [26] did not remove dynamic objects and tracked objects by defining the state of objects. MaskFusion [25] is an algorithm that uses the depth information of scenes obtained from RGB-D cameras and Mask R-CNN to distinguish objects and track objects moving from the background. The tracking was performed by optimizing the geometric and photometric errors for the 3D points in the scene. EM-Fusion [26] made a fixed-shape dynamic object using a local volumetric signed distance function and continuously tracked the dynamic objects based on the expectation-maximization method.

Some algorithms [8], [9], [27] used movable objects for VO when the objects do not move. Vincent et al. [8] classified dynamic objects using real-time semantic segmentation models YOLACT [30] and YOLACT++ [31] and tracked them using the extended Kalman filter. They classified dynamic objects as moving or idle based on the speed of each object. The moving dynamic objects were excluded from the VO and loop closure detection of RTAB-MAP [32], and the idle dynamic objects were only used for VO. However, because the algorithm used the center point of the bounding box obtained by the model as the state of the object, it was significantly influenced by the recognition performance of the model and failed to recognize the static parts of the dynamic object within the bounding box. Ballester et al. [9] proposed an algorithm for recognizing dynamic objects using the neural network Detectron [33]. The algorithm included object poses tracking algorithm, which minimizes photometric repair errors for each recognized object. In the algorithm, the dynamic disparity and differential entropy of an object were obtained using the tracked pose, and the object's motion was then evaluated. However, similar to [8], the algorithm did not recognize the static part inside the object. Kuo et al. [27] proposed the framework that assigns an attention weight to semantic labels detected by mono camera, using an attention module based on neural network for pose estimation. Cho and Kim [28] utilized optical flow and semantic segmentation models to calculate the changes of objects that move within a scene, and generated a dynamic mask. Kim et al. [29] proposed the SimVODIS++ network, which utilizes a self-supervised approach to select salient regions while excluding moving objects.

III. PROPOSED ALGORITHM

A. SCENE FLOW ESTIMATION

The proposed algorithm applies scene flow estimation techniques between consecutive scenes, as applied to the stereo camera in [10] and the RGB-D camera in [34]. The scene flow is a 3D vector representing the change in points between scenes based on the coordinate system of the current scene. This scene flow is obtained by using the coordinates of the corresponding pixels between the scenes, which is matched using the optical flow from the current to the previous scene. The corresponding 3D points between consecutive scenes are computed in (1) and (2). Scene flow M, a change between the matched two 3D points, is computed as a 3D vector in (3).

$$P_{t+1} = \begin{pmatrix} x_{t+1} \\ y_{t+1} \\ z_{t+1} \end{pmatrix} = K^{-1} \begin{pmatrix} u_{t+1} \\ v_{t+1} \\ 1 \end{pmatrix} d_{t+1} \tag{1}$$

$$P_t = \begin{pmatrix} x_t \\ y_t \\ z_t \end{pmatrix} = R \cdot K^{-1} \begin{pmatrix} u_t \\ v_t \\ 1 \end{pmatrix} d_t + T \tag{2}$$

$$M = P_{t+1} - P_t \tag{3}$$

where pixel (u_{t+1}, v_{t+1}) in the $t+1$ scene is points P_{t+1} in 3D space in (1). Similarly, the pixel (u_t, v_t) in the t scene corresponding to (u_{t+1}, v_{t+1}) is point P_t in the coordinate system of the $t+1$ in (2). The two corresponding points are obtained by calculating the optical flow from $t+1$ scene to t scene. The

d_t and d_{t+1} are the depth values for the corresponding pixel in each scene, K is the intrinsic parameters of the camera, and R and T are the rotational and translational matrices, which are the external parameters.

The ego-motion between scenes is estimated using keypoints belonging to the inlier for camera motion. To obtain inliers with a small distance, which indicates that it is an inlier point, we first calculate the Fundamental matrix between scenes using all keypoints and obtain the Sampson distance of each corresponding points to obtain inliers. Using a Random Sample Consensus (RANSAC) algorithm that is robust to the outlier, we solve the perspective n-point problem between the inlier 3D points in the t scene and the matched keypoints in the $t+1$ scene. By solving the problem, the initial ego-motion between scenes is estimated.

B. RESIDUAL MOTION LIKELIHOOD ESTIMATION

Residual motion likelihood L is used to calculate the movement of image parts in consecutive scenes from the scene flow. The likelihood L represents the translation error of 3D points between scenes considering sensor noise, as proposed in [10]. The consideration of sensor noise is reflected by using the Mahalanobis distance for 3D points in (4). The covariance Σ_{SF} of scene flow is obtained from the covariance matrix S_{SF} of the sensor and Jacobian matrix J_{SF} of the sensor parameters for scene flow M in (5).

$$L = \sqrt{M^T \Sigma_{SF}^{-1} M} \tag{4}$$

$$\Sigma_{SF} = J_{SF} S_{SF} J_{SF}^T \tag{5}$$

The covariance matrix S_{SF} of the sensor is composed of elements for the sensor parameters in (6). The sensor parameters are focal length f_u, f_v and principal point c_u, c_v and depth value d_t, d_{t+1} of the camera. The uncertainty of depth value increasing in proportion to the distance from the camera is modeled in a form proportional to the square of depth, similar to the covariance model of the Kinect camera proposed in [35]. σ_n is the standard deviation of the normalized disparity.

$$S_{SF} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & (\frac{1}{f_v}\sigma_n d_{t+1}^2)^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & (\frac{1}{f_v}\sigma_n d_t^2)^2 \end{pmatrix} \tag{6}$$

C. GEOMETRY DYNAMIC MODEL PROBABILITY CALCULATION

In this study, we define the possibility that a part of an object is moving in a scene as *geometry dynamic model (GDM) probability*. This probability is computed as the output of the sigmoid function with the likelihood L as an input.

Because of the properties inherent in optical flow, we do not calculate the likelihood L of certain points with a large error in the scene flow: points that do not match the consecutive scenes, points with too large depth value and boundary points between objects. For example, if a pixel in the current scene has too far and inaccurate depth value, it is inappropriate to determine whether the point is a geometrically dynamic part. Therefore, pixels with depth value larger than the threshold τ_d do not calculate the likelihood L.

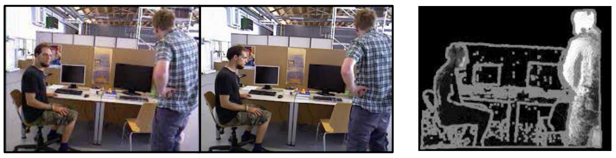
(a) (b)

FIGURE 2. *GDM* probability. (a) The sample of consecutive scenes. (b) Calculated *GDM* probability: the brighter it is, the more dynamic it is.

The boundary extraction algorithm of object proposed in [36] is used to determine the boundary of objects. Moreover, as error of a pixel for the optical flow near the boundary affect neighborhood of the pixel, we apply a dilation operation to the boundary. The pixels, which are dilated boundaries, are included in the boundary set E.

The $GDM(u, v)$, which is the GDM probability of a pixel (u, v), which has the likelihood $L(u, v)$, is calculated as (7). If the $GDM(u, v)$ is large, the pixel will be a dynamic part; otherwise, it will be a static part. Because the pixels with a large depth value are generally a static background with a small influence in tracking module of VO, $GDM(u, v)$ of a pixel with a depth $d(u, v)$ larger than the threshold τ_d is set to 0. τ_l and σ are fixed parameters that are used to adjust $GDM(u, v)$. The set U is a set of unmatched pixels between consecutive scenes. To obtain the unmatched pixels, we calculate the optical flow from the t scene to the $t + 1$ scene to determine the pixels that are not projected into the $t+1$ scene, and these pixels are included in the set U. We initialize $GDM(u, v)$ of the pixel in set E to 0.5.

$$GDM(u, v) = \begin{cases} 0, & \text{if } d(u, v) > \tau_d \\ \dfrac{1}{1 + e^{-\sigma(L(u,v)-\tau_l)}}, & \text{if } (u, v) \notin E, U \\ 0.5, & \text{else} \end{cases} \tag{7}$$

The GDM probability for consecutive scenes in Fig. 2(a) is represented as Fig. 2(b). The closer to white, the higher the GDM probability, and the boundary parts or unmatched ones of the object are gray. The black points are static parts with depths within the threshold τ_d or parts with depths larger than the threshold τ_d.

D. CONFIDENCE CALCULATION OF GDM PROBABILITY FOR MODEL SELECTION

Depending on scene characteristics, the proposed GDM probability calculation may be unreliable. For example, when dynamic objects take up a lot of the field of view or the RGB-D camera movement is extremely fast, the reliable

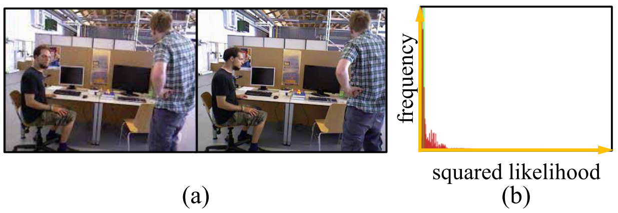

FIGURE 3. Histogram *H* in typical situations. (a) The sample of consecutive scenes. (b) Histogram of squared residual motion likelihood. The red lines are the frequency at each likelihood. The green lines are the location of the median in the histogram.

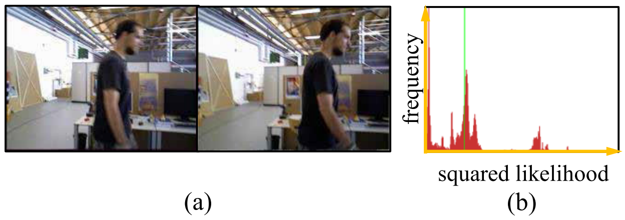

FIGURE 4. Histogram *H* in noisy scenes. (a) The sample of consecutive scenes. (b) Histogram of squared residual motion likelihood. The red lines are the frequency at each likelihood. The green line are the location of the median in the histogram.

estimation of ego-motion and scene flow becomes challenging. Therefore, calculating the confidence C_g of the *GDM* probability and employing other alternative algorithms is necessary.

For the confidence calculation, we propose using the histogram H of squared residual motion likelihood L. The likelihood L is Mahalanobis distance, as described in section III-B. Since the squared Mahalanobis distance is chi-squared distribution, the square of the likelihood approximates chi-squared distribution with 3 degrees of freedom and the likelihood of dynamic parts are distributed to the right of the histogram H as outliers. If the estimated ego-motion is close to the real one and no motion blur appears in the scene, the distribution of square of the likelihood will be similar to chi-squared distribution. For example, in general consecutive scenes of Fig. 3(a), the histogram of squared likelihood has a high frequency near 0 as Fig. 3(b). Additionally, most of the objects in the scenes are static ones: a desk, chairs, and cubicle walls, which are contributed to accurate ego-motion estimation. This means that the actual distribution of squared likelihood is approximated as chi-squared distribution.

However, in scene of Fig. 4(a), the estimated ego-motion is different with real one because moving person occupies in the field of view. This difference causes the histogram H to be skewed to the right in Fig. 4(b). Therefore, we consider the distribution characteristics of the histogram H for evaluating the confidence C_g.

We calculate the confidence C_g using the median of squared likelihood $M_s(L)$ as (8), where M_3 is the median of chi-squared distribution with 3 degrees of freedom ≈ 2.381. If the confidence C_g in the *GDM* probability calculation is lower than threshold τ_c, we calculate the *semantic*

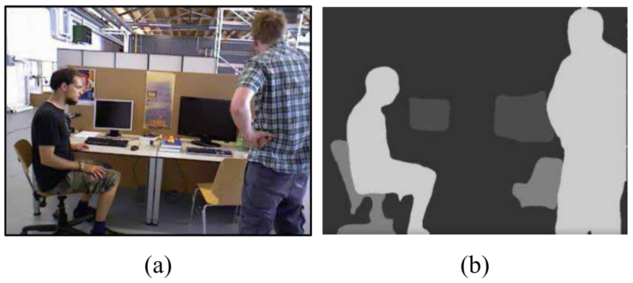

FIGURE 5. *SSD* probability. (a) RGB image. (b) Calculated *SSD* probability. Compared to the monitor, chairs and people appear brighter. This indicates that these objects are more dynamic than monitors.

segmentation dynamic (SSD) probability, which we define, using the semantic segmentation model, DeepLab V3 [13]. The model is a neural network that includes astrous spatial pyramid pooling to recognize the multiscale features of the image. We use RGB images, such as in Fig. 5(a) as input to the model and calculate a predetermined *SSD* probability from 0 to 1 for each type of object such as Fig. 5(b).

$$C_g = e^{-\frac{|M_s(L)-M_3|}{10}} \tag{8}$$

E. POST-PROCESSING BASED ON CRF

In this study, when the confidence C_g about *GDM* probability is larger than τ_c, we used DenseCRF [37] model, one of the CRF's techniques. It is a discriminant undirected probabilistic graphical model that can represent the relationship between different variables and is used to infer *GDM* probability of the part where the likelihood L cannot be evaluated.

Optimization with CRF minimizes energy functions such as (9) to infer whether the part is dynamic. $X = (x_1, x_2, \cdots, x_K)$ is an observation variable corresponding to the input and $Y = (y_1, y_2, \cdots, y_K)$ is the joint output variable indicating whether some part of the scene is dynamic. K is the total number of pixels in the current scene. We used RGB values of each pixel as similarity information between neighboring pixels. the binary variable y_i indicates that the pixel i is a dynamic part when it is 0 and a static part when it is 1. The first term of the energy function, the unitary term, is the penalty term for the wrong label, and the second term, the pairwise term, induces the neighboring pixels with similar color to have similar labels.

$$E(Y|X) = \Sigma_{i\in K}\psi_u(y_i) + \Sigma_{i,j\in K}\psi_p(y_i, y_j) \tag{9}$$

We compute the unary potential $\psi_u(y_i)$ about the occluded part by applying the Occlusion Processing method proposed in [34] to infer the *GDM* probability of the part unmatched previous scene. When we project an unmatched 3D point in the current scene into the previous scene, we determine the point as a static part if the difference $\Delta z = z_{proj}$ between the actual depth z' and the expected depth z_{proj} of the pixel projected in the previous scene exceeds the threshold τ_b. As per this method, we define the unary potential $\psi_u(y_i)$

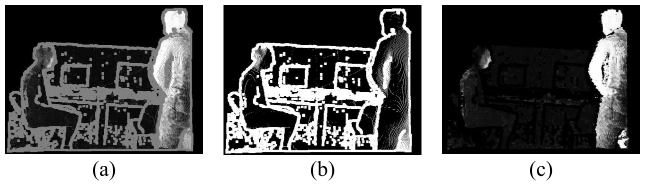

FIGURE 6. Post-processing based on CRF. (a) *GDM* probability. (b) Unmatched points and boundaries between objects (white). (c) Post-processed *GDM* probability.

like (10), (11). *clip* is an operation that clips the input value between 0.1 and 0.9 such that the $GDM(u, v)$ does not become 0 or 1. The pairwise potential is the weighted mix of Gaussian function about the position and RGB value of neighboring pixels adopted in [34] and [37].

$$\psi_u(0) = \begin{cases} -\log(clip(GDM)), & (u,v) \notin U \\ -\log(0.4), & \Delta z \geq \tau_b \\ -\log(0.6), & \Delta z < \tau_b \end{cases} \quad (10)$$

$$\psi_u(1) = \begin{cases} -\log(clip(1-GDM)), & (u,v) \notin U \\ -\log(0.6), & \Delta z \geq \tau_b \\ -\log(0.4), & \Delta z < \tau_b \end{cases} \quad (11)$$

The energy function defined in (9) is optimized by applying the mean field approximation proposed in [37]. For *GDM* probability shown in Fig. 6(a), the white color in Fig. 6(b) represents the unmatched part and the boundary between objects. After mean field approximation, the *GDM* probability of the white part in Fig. 6(b) has a probability close to 0 or 1, as shown in Fig. 6(c).

F. VO USING THE DYNAMIC PROBABILITY

In this study, we extract keypoints to be used for VO by selecting one of the two models as Fig. 7(a). If the confidence C_g about *GDM* probability is higher than threshold τ_c, keypoints are extracted from the static part, where *GDM* probability post-processed with the CRF model is less than the threshold τ_g. Otherwise, *SSD* probability is calculated using the DeepLab V3 model, and keypoints are extracted from the part where the probability is lower than the threshold τ_s. These keypoints were used to calculate VO in the tracking thread of ORB-SLAM2 such as Fig. 7(b).

IV. EXPERIMENTS

We evaluated our algorithm with dynamic environments and compared its performance with ORB-SLAM2 [2] and DynaSLAM [4] using TUM RGB-D dataset [38] and Bonn RGB-D dynamic dataset [39]. In all experiments, loop closing does not occur so that only the performance of VO can be compared.

In the TUM RGB-D dataset, we compared our algorithm with the result of other algorithm [3], [4], [8], [12], [25] which are reported in previous studies. We compared the proposed algorithm performance with the algorithm using only *SSD*

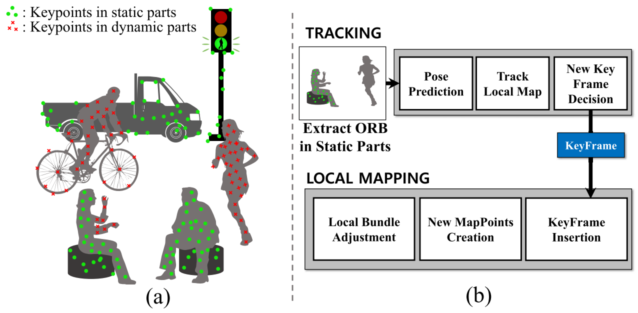

FIGURE 7. VO using static parts. (a) Extracted keypoints in scene. Red and green points are extracted from static and dynamic parts, respectively. (b) Extracted static keypoints is processed in tracking thread of ORB-SLAM2.

probability in our algorithm and analyzed the post-processed *GDM* probability generated by the algorithm.

In the Bonn RGB-D dynamic dataset, we compared the proposed algorithm with other algorithms for dynamic environments with moving boxes or crowds of people. Through analysis of the results obtained from these algorithms, we conducted a study on the performance differences and limitations between the proposed algorithm and algorithms that rely continuously on semantic segmentation models.

A. TUM RGB-D DATASET

We used a TUM RGB-D dataset [38] that includes a dynamic environment. The dataset comprises color and depth images having a resolution of 640 × 480. The ground truth trajectory of the dataset was obtained from the tracking camera. We used 8 sequences to evaluate our algorithm for multiple dynamic environments. The sequences have sitting and walking sequences. The sitting sequence comprises two people sitting in front of a desk to assess whether SLAM and VO algorithms are robust to low-dynamic objects. The walking sequence comprises two people walking in the office and it is used to assess whether the SLAM and VO algorithms are robust to high-dynamic objects. The sitting and walking sequences in tables are marked as "s" and "w", respectively.

The performance criteria for the algorithm are Absolute Trajectory Error (ATE) and Relative Pose Error (RPE). ATE is the error between the camera trajectory of the ground truth and the estimated camera trajectory; it shows the overall performance of the VO and SLAM systems. RPE refers to the difference between the relative pose and the actual relative pose between consecutive frames; it is used to measure the drift of the VO. Root Mean Square Error (RMSE) and Standard Deviation (SD) are the error metrics for the two criteria.

Table 1 and 2 are the results of comparing our algorithm with ORB-SLAM2 and DynaSLAM. Our algorithm demonstrated high performance improvements over ORB-SLAM2 in highly dynamic environments. In highly dynamic environments, the performance is similar to that of DynaSLAM because the person, the dynamic object in the sequence, moves the whole body except at the beginning and end of the

TABLE 1. **Comparison of the RMSE of ATE [m] of SP-VO against ORB-SLAM2, DynaSLAM in TUM dataset.**

Seq.	ORB-SLAM2		DynaSLAM		Proposed	
	RMSE	SD	RMSE	SD	RMSE	SD
s/half	0.020	0.011	0.017	0.008	**0.012**	**0.005**
s/rpy	**0.019**	**0.012**	0.024	0.015	0.024	0.017
s/static	0.008	0.004	**0.006**	**0.003**	0.008	0.004
s/xyz	**0.008**	**0.004**	0.013	0.005	**0.008**	**0.004**
w/half	0.598	0.309	0.027	0.013	**0.024**	**0.011**
w/rpy	0.607	0.228	-	-	**0.036**	**0.020**
w/static	0.594	0.272	**0.006**	**0.002**	0.009	0.004
w/xyz	0.944	0.549	**0.014**	0.009	**0.014**	**0.007**

TABLE 2. **Comparison of the RMSE of RPE [m] of SP-VO against ORB-SLAM2, DynaSLAM in TUM dataset.**

Seq.	ORB-SLAM2		DynaSLAM		Proposed	
	RMSE	SD	RMSE	SD	RMSE	SD
s/half	0.020	0.014	0.021	0.010	**0.013**	**0.006**
s/rpy	**0.024**	**0.014**	0.031	0.019	0.031	0.020
s/static	0.009	0.004	**0.007**	**0.003**	0.009	0.005
s/xyz	**0.010**	**0.005**	0.014	0.007	0.011	0.005
w/half	0.333	0.264	**0.025**	**0.012**	**0.025**	**0.012**
w/rpy	0.362	0.265	-	-	**0.048**	**0.027**
w/static	0.226	0.194	**0.008**	**0.003**	0.011	0.006
w/xyz	0.487	0.322	**0.019**	**0.009**	**0.019**	**0.009**

sequence; therefore, there were no static parts in the dynamic object for a long time. However, in low dynamic environments, it performs better than the DynaSLAM algorithm. Because static parts (e.g., legs) exist in the body of the person sitting for a long time, rich keypoints could be utilized in part for VO.

Fig. 8 shows the trajectory estimated by the proposed algorithm. The blue line represents the estimated path, and the red line represents the difference between the actual and estimated paths. The proposed algorithm had higher path estimation performance than the ORB-SLAM2.

To compare our algorithm with various others, we compared the results of algorithms that used the TUM dataset for experiments, as shown in Table 3. Since the codes for these algorithms were not all publicly available, we cited the RMSE or median value of the ATE from each paper. If results for a specific sequence were not reported in the paper, we marked that entry with '-'. The algorithm proposed by Hempel and Al-Hamadi [12] used sceneflow from RGB-D images, similar to our algorithm, but showed a large ATE. We interpreted this performance difference as a positive effect of the selective geometric approach using confidence C_g.

In the overall high and low dynamic environments, our algorithm demonstrates good performance compared to the algorithm using only semantic information, as shown in Table 4. This difference is because the algorithm using only semantic information cannot distinguish between static and dynamic parts within a dynamic object and cannot cope with moving objects that are semantically static or unrecognized by the model. Our algorithm alleviates this problem through the use of semantic information only when the confidence C_g of the probability inferred by the likelihood is low.

Fig. 9 shows the reason our algorithm in the sitting xyz sequence in Table 1 outperformed other algorithms. Between the consecutive scenes in Fig. 9(a), only the right arm of the left person is moving. The post-processed *GDM* probability in this scene is high in the dynamic right arm part, and this probability about the black region in Fig. 9(b) is excluded from the VO measurement. The proposed algorithm extracts keypoints except for the dynamic arm, and uses them for VO. Therefore, the loss of information is less.

B. BONN RGB-D DATASET

The Bonn RGB-D dynamic dataset is a dataset of dynamic environments that includes 23 sequences of moving objects, as well as people. Because objects such as balloons and boxes move within the dataset, it is difficult to accurately recognize whether they are changing objects through semantic segmentation. Additionally, unlike the TUM dataset, people often occupy the majority of the scene, making this dataset evaluated for more general environments.

As the dataset uses the same data structure and file format as the TUM dataset, we used ATE and RPE as performance criteria in the same way. Additionally, we defined the Success Rate of Tracking (SRT) in visual odometry as an additional criterion. Since ATE and RPE do not include camera poses that were not successfully tracked in the metric calculation, this rate was used as an indicator of how robustly the algorithm succeeded in estimating poses in image sequences.

The experiments on the proposed algorithm and ORB SLAM2, Dyna SLAM were conducted using four different sequences. First, the Crowd sequence contains a scene where three people are walking. The Kidnapping Box (KB) 2 sequence captures a scene where a box disappears between camera viewpoint changes. The Placing Nonobstructing Box (PNB) sequence includes a scene where a person carries a box of appropriate size and places it down while walking. Lastly, the Moving Obstructing Box (MOB) sequence shows a person pushing a very large box while moving.

The results of the experiments showed different performances depending on the characteristics of each sequence, as shown in Table 5, 6 and 7. In the Crowd sequence, the proposed algorithm showed slightly higher RMSE in ATE compared to DynaSLAM, indicating a limitation of the proposed ego-motion-based method in accuracy when there are many people walking in the scene. On the other hand, the proposed algorithm showed robust performance in the KB 2 sequence compared to DynaSLAM, as the proposed algorithm estimates odometry based on the change between the current and previous frames, unlike DynaSLAM, which considers geometric changes between keyframes. Therefore, even if a certain object suddenly disappears, the proposed algorithm showed robust visual odometry performance. In the PNB sequence, the proposed algorithm showed robust performance for objects that are not pre-determined to move, as it recognizes object motion as scene flow. In the MOB sequence, DynaSLAM showed low RMSE in ATE but low SRT. This is because the proposed algorithm utilizes

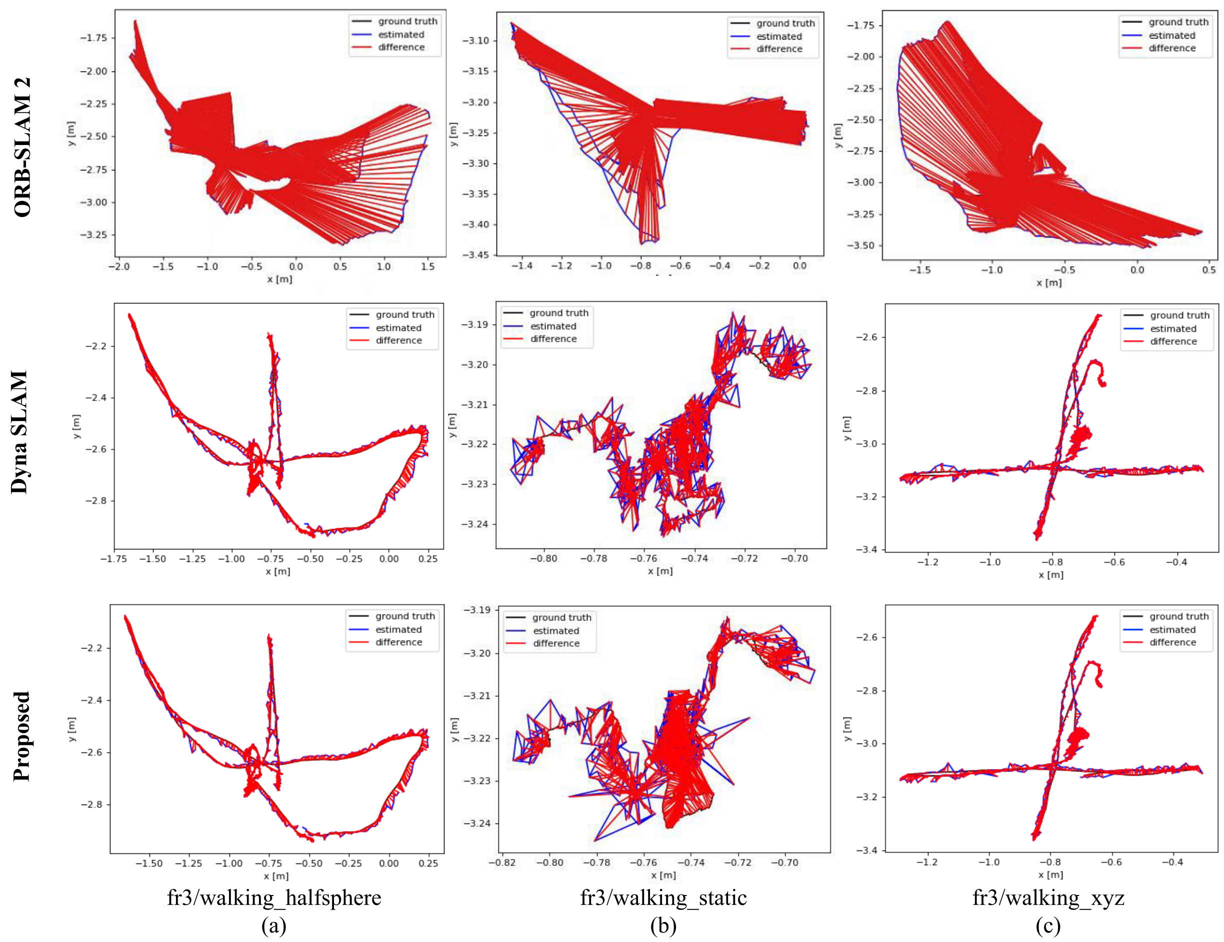

FIGURE 8. Trajectories of sequences walking halfsphere, static, xyz.

TABLE 3. Comparison of ATE [m] of SP-VO against other algorithms in TUM dataset.

Seq.	Mask Fusion [25]	DyanSLAM [4]	DS-SLAM [3]	Vincent *et al.* [8]	Hempel and Al-Hamadi [12]	Proposed					
	RMSE	RMSE	RMSE	RMSE	median	RMSE	mean	median	std	min	max
s/half	0.052	0.017	-	-	0.018	**0.013**	0.011	**0.010**	0.005	0.001	0.041
s/rpy	-	-	-	-	-	0.024	0.017	**0.013**	0.017	0.000	0.110
s/static	0.021	-	**0.007**	0.060	-	0.008	0.007	**0.006**	0.004	0.000	0.039
s/xyz	0.031	0.015	-	0.018	0.011	**0.008**	0.007	**0.006**	0.004	0.001	0.056
w/half	0.106	0.025	0.030	0.040	0.034	**0.024**	0.021	**0.019**	0.012	0.001	0.084
w/rpy	-	**0.035**	0.444	0.053	0.144	0.036	0.030	**0.024**	0.020	0.004	0.140
w/static	0.035	**0.006**	0.008	0.008	0.009	0.009	0.008	**0.008**	0.004	0.001	0.056
w/xyz	0.104	0.015	0.025	0.021	0.018	**0.014**	0.013	**0.012**	0.007	0.000	0.053

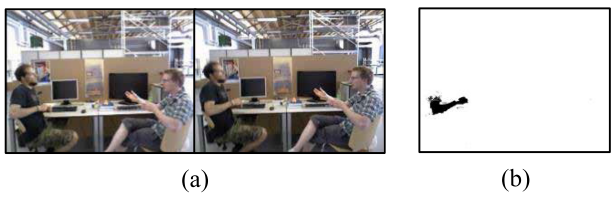

FIGURE 9. Example of good performance in sitting xyz sequence, (a) The sample consecutive scenes. (b) Post-processed *GDM* probability. Only the dynamic arms of the human body are expressed in black.

information about people in VO estimation, whereas DynaSLAM has limitations in geometric verification based on depth comparison between keyframes.

V. DISCUSSION

In this section, we discuss the performance, limitations, and future research of proposed algorithm. Our algorithm detects static parts in dynamic scenes and utilizes them to estimate the camera's position, showing better performance compared to algorithms that remove movable objects using semantic models. We interpret that increasing the number of features extracted from the static parts in the scene is related to the improvement in performance. As the number of static features increases, the correspondences necessary for calculating relative poses between frames also grow. the relation has also been reported in previous research [40] on static environments. Additionally, in dynamic scenes where moving objects mostly occupy the camera's field of view,

TABLE 4. Comparison of the RMSE of ATE [m] of SF-VO against Algorithm using only *SSD* probability.

Seq.	SF-VO		*SSD*	
	RMSE	SD	RMSE	SD
s/half	**0.012**	**0.005**	0.017	0.008
s/rpy	**0.024**	**0.017**	0.025	0.016
s/static	**0.008**	**0.004**	0.030	0.018
s/xyz	0.008	0.004	**0.007**	**0.003**
w/half	**0.024**	**0.011**	0.026	0.012
w/rpy	0.036	0.020	**0.033**	**0.019**
w/static	**0.009**	**0.004**	0.013	0.006
w/xyz	**0.014**	**0.007**	0.015	0.007

TABLE 5. Comparison of ATE [m] of SP-VO against other algorithms in Bonn dynamic dataset.

Seq.	ORB-SLAM2		DynaSLAM		Proposed	
	RMSE	SD	RMSE	SD	RMSE	SD
Crowd	-	-	**0.020**	**0.010**	0.026	0.021
KB 2	0.025	**0.012**	0.026	**0.012**	**0.024**	**0.012**
PNB	0.881	0.288	0.317	0.123	**0.050**	**0.043**
MOB	0.434	0.152	-	-	**0.258**	**0.087**

TABLE 6. Comparison of RPE [m] of SP-VO against other algorithms in Bonn dynamic dataset.

Seq.	ORB-SLAM2		DynaSLAM		Proposed	
	RMSE	SD	RMSE	SD	RMSE	SD
Crowd	0.771	0.503	**0.136**	0.065	**0.136**	**0.063**
KB 2	**0.340**	**0.147**	**0.340**	**0.147**	0.341	0.148
PNB	0.366	0.300	0.208	0.136	**0.167**	**0.088**
MOB	0.324	0.165	-	-	**0.291**	**0.134**

TABLE 7. Comparison of ATE [m] and SRT [%] of SP-VO against other algorithms in Bonn dynamic dataset.

Seq.	ORB-SLAM2		DynaSLAM		Proposed	
	RMSE	SRT	RMSE	SRT	RMSE	SRT
Crowd	0.884	87.163	**0.020**	99.029	**0.026**	**99.676**
KB 2	0.025	**99.611**	0.026	**99.611**	**0.024**	**99.611**
PNB	0.881	**99.722**	0.371	**99.722**	**0.050**	**99.722**
MOB	0.434	**99.830**	**0.161**	50.594	0.258	**99.830**

semantic segmentation model-based approaches removed a large number of features and often resulted in tracking lost. These factors contributed to the improved performance of our algorithm.

Our proposed algorithm has some limitations. If semantic segmentation model is chosen based on confidence, the proposed algorithm uses fixed probability to represent whether the object is moving. However, if this probability takes into account the situation of the scene, it would be more suitable for representing the moving probability. For example, if a person is holding a box, it would be natural to assign a higher dynamic probability to the box than usual. we are attempting to recognize dynamic objects based solely on the situation information of a scene without information on geometric changes between scenes. Additionally, our algorithm uses static features as map points between scenes, which is suitable for accuracy in visual odometry, but inappropriate for creating a map for SLAM that can be used in the long term. To create a map that can be used by a robot continuously, it would be helpful to recognize map points that cannot be used over a long period and to remove them at an appropriate time during SLAM (e.g., loop closure detection).

VI. CONCLUSION

In this study, we presented a VO algorithm that uses static parts present in dynamic objects and static background in dynamic environments. The proposed algorithm uses scene flow to calculate the *GDM* probability that a particular part of a scene is moving. Moreover, depending on the confidence of the probability, we proposed an algorithm that uses either the CRF model or the semantic segmentation model for accurate VO. Our algorithm demonstrated robust performance in high-dynamic environments of TUM and Bonn dynamic dataset with moving objects, exhibiting lower pose estimation errors than existing DynaSLAM or ORB-SLAM2.

In future, our work will include real-time performance. Additionally, we will recognize dynamic parts within a scene based not only on semantic information but also on situational information of the scene.

REFERENCES

[1] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 2100–2106.

[2] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[3] C. Yu, Z. Liu, X. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "DS-SLAM: A semantic visual SLAM towards dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1168–1174, doi: 10.1109/IROS.2018.8593691.

[4] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 4076–4083, Oct. 2018, doi: 10.1109/LRA.2018.2860039.

[5] J. C. V. Soares, M. Gattass, and M. A. Meggiolaro, "Crowd-SLAM: Visual SLAM towards crowded environments using object detection," *J. Intell. Robot. Syst.*, vol. 102, no. 2, p. 50, Jun. 2021.

[6] Y. Liu and J. Miura, "RDS-SLAM: Real-time dynamic slam using semantic segmentation methods," *IEEE Access*, vol. 9, pp. 23772–23785, 2021.

[7] S. Wen, P. Li, Y. Zhao, H. Zhang, F. Sun, and Z. Wang, "Semantic visual SLAM in dynamic environment," *Auto. Robots*, vol. 45, no. 4, pp. 493–504, May 2021, doi: 10.1007/s10514-021-09979-4.

[8] J. Vincent, M. Labbé, J.-S. Lauzon, F. Grondin, P.-M. Comtois-Rivet, and F. Michaud, "Dynamic object tracking and masking for visual SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 4974–4979.

[9] I. Ballester, A. Fontán, J. Civera, K. H. Strobl, and R. Triebel, "DOT: Dynamic object tracking for visual SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 11705–11711.

[10] P. F. Alcantarilla, J. J. Yebes, J. Almazán, and L. M. Bergasa, "On combining visual SLAM and dense scene flow to increase the robustness of localization and mapping in dynamic environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 1290–1297.

[11] M. Jaimez, C. Kerl, J. Gonzalez-Jimenez, and D. Cremers, "Fast odometry and scene flow from RGB-D cameras based on geometric clustering," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 3992–3999.

[12] T. Hempel and A. Al-Hamadi, "Pixel-wise motion segmentation for SLAM in dynamic environments," *IEEE Access*, vol. 8, pp. 164521–164528, 2020.

[13] L. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *CoRR*, vol. abs/1706.05587, pp. 1–14, Jun. 2017.

[14] D.-H. Kim and J.-H. Kim, "Effective background model-based RGB-D dense visual odometry in a dynamic environment," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1565–1573, Dec. 2016.

[15] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving RGB-D SLAM in dynamic environments: A motion removal approach," *Robot. Auto. Syst.*, vol. 89, pp. 110–122, Mar. 2017.

[16] S. Li and D. Lee, "RGB-D SLAM in dynamic environments using static point weighting," *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2263–2270, Oct. 2017.

[17] R. Wang, W. Wan, Y. Wang, and K. Di, "A new RGB-D SLAM method with moving object detection for dynamic indoor scenes," *Remote Sens.*, vol. 11, no. 10, p. 1143, May 2019. [Online]. Available: https://www.mdpi.com/2072-4292/11/10/1143

[18] G. Xu, Z. Yu, G. Xing, X. Zhang, and F. Pan, "Visual odometry algorithm based on geometric prior for dynamic environments," *Int. J. Adv. Manuf. Technol.*, vol. 122, no. 1, pp. 235–242, Sep. 2022.

[19] A. Talukder and L. Matthies, "Real-time detection of moving objects from moving vehicles using dense stereo and optical flow," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, vol. 4, Sep./Oct. 2004, pp. 3718–3725.

[20] R. K. Namdev, A. Kundu, K. M. Krishna, and C. V. Jawahar, "Motion segmentation of multiple objects from a freely moving monocular camera," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 4092–4099.

[21] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder–decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[22] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[23] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.

[24] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021.

[25] M. Runz, M. Buffier, and L. Agapito, "MaskFusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *Proc. IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*, Oct. 2018, pp. 10–20.

[26] M. Strecke and J. Stueckler, "EM-fusion: Dynamic object-level SLAM with probabilistic data association," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5864–5873.

[27] X.-Y. Kuo, C. Liu, K.-C. Lin, and C.-Y. Lee, "Dynamic attention-based visual odometry," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 160–169.

[28] H. M. Cho and E. Kim, "Dynamic object-aware visual odometry (VO) estimation based on optical flow matching," *IEEE Access*, vol. 11, pp. 11642–11651, 2023.

[29] U.-H. Kim, S.-H. Kim, and J.-H. Kim, "SimVODIS++: Neural semantic visual odometry in dynamic environments," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 4244–4251, Apr. 2022.

[30] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "YOLACT: Real-time instance segmentation," in *Proc. ICCV*, Oct. 2019, pp. 9156–9165.

[31] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "YOLACT++ better real-time instance segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 2, pp. 1108–1121, Feb. 2022.

[32] M. Labbé and F. Michaud, "RTAB-map as an open-source LiDAR and visual simultaneous localization and mapping library for large-scale and long-term online operation," *J. Field Robot.*, vol. 36, no. 2, pp. 416–446, Mar. 2019.

[33] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. (2019). *Detectron2*. [Online]. Available: https://github.com/facebookresearch/detectron2

[34] H. Jeon, C. Han, D. You, and J. Oh, "RGB-D visual SLAM algorithm using scene flow and conditional random field in dynamic environments," in *Proc. 22nd Int. Conf. Control, Autom. Syst. (ICCAS)*, Nov. 2022, pp. 129–134.

[35] K. Khoshelham and S. O. Elberink, "Accuracy and resolution of Kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, pp. 1437–1454, Feb. 2012. [Online]. Available: https://www.mdpi.com/1424-8220/12/2/1437

[36] K. Tateno, F. Tombari, and N. Navab, "Real-time and scalable incremental segmentation on dense SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 4465–4472.

[37] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected CRFs with Gaussian edge potentials," in *Proc. 24th Int. Conf. Neural Inf. Process. Syst. (NIPS)*. Red Hook, NY, USA: Curran Associates, 2011, pp. 109–117.

[38] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 573–580.

[39] E. Palazzolo, J. Behley, P. Lottes, P. Giguère, and C. Stachniss. (2019). *ReFusion: 3D Reconstruction in Dynamic Environments for RGB-D Cameras Exploiting Residuals*. [Online]. Available: https://www.ipb.uni-bonn.de/pdfs/palazzolo2019iros.pdf

[40] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Visual SLAM: Why filter?" *Image Vis. Comput.*, vol. 30, no. 2, pp. 65–77, Feb. 2012, doi: 10.1016/j.imavis.2012.02.009.

HYEONGJUN JEON was born in Dobong-gu, Seoul, South Korea, in 1996. He received the B.S. degree in robotics engineering from Kwangwoon University, Seoul, in 2020, where he is currently pursuing the master's degree. His research interests include multi robot task allocation, SLAM, and artificial intelligence for robotics.

JUNGHYUN OH (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering from Seoul National University, Seoul, South Korea, in 2012, 2014, and 2018, respectively. From 2018 to 2019, he was a Senior Engineer with Samsung Research of Samsung Electronics Company Ltd., Seoul. Since 2019, he has been an Assistant Professor with the Department of Robotics, Kwangwoon University, Seoul. His research interests include long-term robot autonomy, SLAM, and artificial intelligence for robotics.