Received June 22, 2020, accepted August 31, 2020, date of publication September 7, 2020, date of current version September 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3022506

Pixel-Wise Motion Segmentation for SLAM in Dynamic Environments

THORSTEN HEMPEL AND AYOUB AL-HAMADI
Neuro-Information Technology Group, Otto von Guericke University Magdeburg, 39106 Magdeburg, Germany

Corresponding author: Thorsten Hempel (thorsten.hempel@ovgu.de)

This work was supported by the Federal Ministry of Education and Research (BMBF) within the 3D Sensation Alliance under Grant 03ZZ0448G-L.

ABSTRACT Visual simultaneous localization and mapping (SLAM) is a key prerequisite for many mobile robotic systems. A common assumption for SLAM methods is a static environment. The interference of dynamic objects can lead to impairment of the camera pose tracking and permanent distortions of the map. This limits the use of many visual SLAM systems in real world scenarios, where dynamic environments are typical. We present a novel method for pixel-wise segmentation of dynamic image sequences based on a scene flow model estimation. We detect and eliminate outlying pixels sparsely by evaluating each pixel motion separately and maintain the most possible area of static scene background for SLAM. The evaluation with the public TUM dataset demonstrates that our proposed method outperforms other comparable state-of-the-art approaches for dynamic removal for SLAM systems.

INDEX TERMS Simultaneous mapping and localization, motion removal, dynamic environments, motion segmentation.

I. INTRODUCTION

Simultaneous localization and mapping has been intensively researched over the past decades and became a fundamental capability for mobile robots. In visual SLAM the most simplistic setup is a monocular camera, which is popular due to its low cost, size and fast calibration. Its downside is a prone initialization process and the unobservability of scale. With the development of more complex sensors RGB-D and stereo cameras gained extensive popularity for enabling a metric scaling of the environment and led to improvements of the robustness of SLAM algorithms. Recent SLAM systems [1]–[3] achieve excellent accuracy results in static scenes, but tend to forfeit accurateness in dynamic environments. A common reason is a flawed transformation estimation of the camera motion, as the matching between a frame with its previous reference is violated by dynamic objects. This significantly limits the use of many visual SLAM system in target applications, *e.g.* in mobile service robots and autonomous vehicles, where multiple dynamic elements in the environment are common.

The associate editor coordinating the review of this manuscript and approving it for publication was Rui-Jun Yan.

The handling of dynamic objects to improve the performance of SLAM in non-static environments is therefore an important problem. We tackle this by introducing a preprocessing step that segments dynamic pixels based on a scene flow derived projective transformation model. The scene flow is generated from the dense optical flow estimation of a deep neural network in combination with depth information of the corresponding depth map pair from a RGB-D camera. A best fitting projective transformation of the ego-motion induced motion field reveals salient pixels and enables a detailed segmentation. We show that our method of decoupling dynamic elements from the static background is remarkably efficient due its pixel-wise procedure and independence from further semantic analysis of the images (See Fig. 1). The contributions of the proposed work are as follows:

- The creation of scene flow by combining optical flow with depth map data
- A model for an iterative projective transformation estimation by using a constrained RANSAC for pixelwise segmentation of dynamic elements
- An extensive evaluation of the proposed method, embedded as preprocessing step in a state-of-the-art SLAM system, with a public benchmark dataset in comparison to other dynamical removal algorithms

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

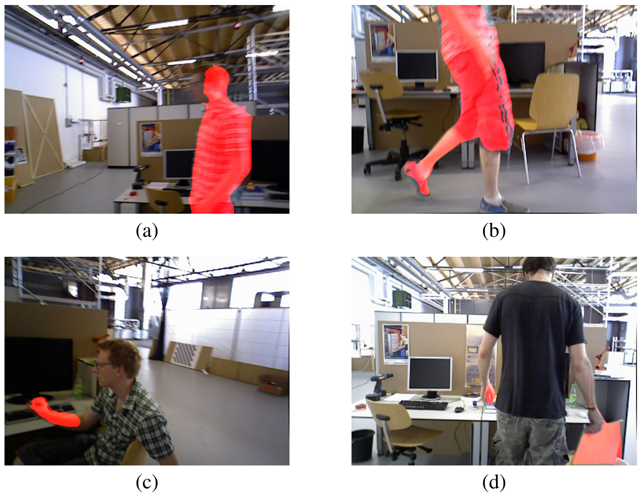

FIGURE 1. **Overview of the results: (a) Segmentation of a person walking by. (b) Acceptance as static of its momentary static leg. (c) Selective segmentation of a moving forearm. (d) Segmentation of a moved chair.**

The paper is structured as follows: We review different approaches for handling SLAM in dynamic environments and analyze recent state-of-the-art methods in section II.

Section III gives details about our proposed approach and is followed by a detailed result examination in section IV. Section V presents the conclusion of our work and states future directions for research.

II. RELATED WORK

Visual SLAM is typically classified in feature-based and direct methods. Feature based SLAM uses salient image points for matching while direct methods reconstruct the pose and map by the minimization of the photometric error. Both types commonly reject outliers using techniques such as RANSAC or robust cost functions. Unfortunately, these methods reach their limit when outliers increase from single feature point mismatches or pixel intensity changes to dynamic objects in form of image clusters. Multiple approaches target the visual SLAM in dynamic scenes. Some of the most relevant are:

- Alcantarilla *et al.* [4] propose, similarly to us, the detection of moving objects using dense scene flow from stereo cameras. However, their approach differs in operating the Mahalanobis metric directly on the scene flow to identify outliers. Jaimez *et al.* [5] calculate the scene flow via K-Means Clustering and warp it with a camera pose estimation.
- Kim and Kim [6] presented a background model-based dense-visual-odometry (BaMVO) algorithm that rejects dynamic objects by modeling a static background from multiple consecutive depth images.
- Li and Lee [7] calculate the probability of dynamic feature points using weighted depth edge points.
- Sun *et al.* [8] remove motion patches on RGB-D data by tracking particles on ego-motion compensated images.
- Wang *et al.* [9] segment dynamic image areas by combining epipolar constraints and clustered depth maps.

Many recent approaches use deep learning to enhance the dynamic object segmentation. The most relevant are:

- Bescos *et al.*. [10] use Mask R-CNN [11] to segment potential dynamic objects. Undefined dynamic objects are detected by multi view geometry.
- Liu *et al.* [12] apply YOLO [13] to detect potential dynamic objects. Target objects are further analyzed for dynamic behaviour using optical flow. Similarly, Yu *et al.* [14] construct a static background using SegNet.
- Zhang *et al.* [15] determines dynamic pixels by warping PWC-Net [16] generated optical flow with an ego-motion estimation.

All of this methods are tied to constraints that either influence their robustness or the accuracy of the odometry estimation. The ability of rejecting a priori dynamic image components by object detection [10], [12], [14] requires the objects to be known by the neural network in the first place. Secondly, segmenting a potential dynamic object omits its possibility of being scene-specific non-dynamic and therefore usable as part of the static background. *E.g.* a scene showing a crowded stand of an opera will likely cause a failure of the tracking due to the segmentation of all the image area of the spectators even though their overall movement is minimal. [12], [14] address this problem by additionally calculating the optical flow to examine a priori dynamic objects for frame specific movement. This can increase the robustness in scenes with many potential but non-moving dynamic objects, but doesn't take into account that also commonly static objects (*e.g.* books, chairs, desks) can be moved and become dynamic objects. A semantic based segmentation is therefore very tied to scene specific applications.

Additionally, using only planary information for either optical flow estimation [15], ego-motion estimation [8] or feature matching lacks the ability of accurately detecting motion of smaller image fragments that is orthogonal to the image plane. Many SLAM algorithms [1], [17] use depth data from IR sensors or stereo vision to enhance the ego-motion estimation in 3-dimensional space and enable absolute distance scaling for the map. So, it is reasonable to use this additional information channel also for dynamic scene analysis. Yet, the depth estimation tends to be noisy and become imprecise with increasing distance, which makes the depth based segmentation for motion removal very challenging, especially when camera-induced motion and object motion take affect in combination [6], [7].

Our proposal targets the frame specific and semi-dense segmentation of dynamic motion in consecutive images by calculating the reprojection error towards an estimated 3D homography. The transformation matrix is calculated from a scene flow, that is generated from a neural network backed optical flow and its corresponding depth maps. The pixel-wise segmentation ensures to obtain the maximum

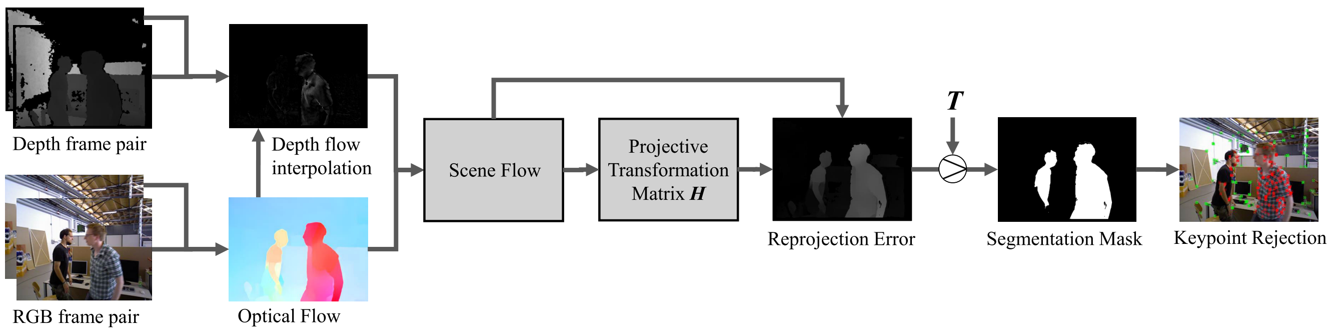

FIGURE 2. Block diagram of our proposal. The process starts by calculating dense optical flow from a consecutive frame pair. After that the depth flow is interpolated based on the optical flow and a depth pair for creation of a basic scene flow. In the next step a projective transformation matrix is robustly estimated from the scene flow that projects the camera induced flow. Flow vectors from dynamic pixels can be segmented afterwards by evaluating its reprojection error towards the transformation.

possible area of static background which retains optimal robustness and odometry precision.

III. METHODOLOGY

The idea of our method is straightforward. We want to map the precise motion field from the RGB-D sequence using scene flow. We think it is reasonable to assume that there are less dynamic pixels in the frame than static ones. Accordingly, the majority of the motion field is defined by the camera motion. We are therefore able to find a representative transformation in the scene flow that reflects the camera motion between the images pairs. Warping it with the scene flow shows us based on the reprojection error which pixels have an divergent motion flow towards the ego-motion and belong to dynamic objects. Figure 2 illustrates the procedure and will be further examined in the following.

A. SCENE FLOW ESTIMATION

Scene flow was introduced in [18] as three-dimensional motion field of points and since then multiple times approached on stereo [4], [19] and RGB-D systems [5], [20]–[23]. By analogy with optical flow methods novel approaches for scene flow [24]–[26] with high accuracy are based on deep learning. Despite their leading performance these methods suffer from high runtimes of between a few seconds and plenty of minutes for a single image pair. We reduce the complexity by combining similarly to Schuster *et al.* [27] a top-performing optical flow method with depth data to estimate the scene flow. As illustrated in Figure 2, we have two time-consecutive input pairs. We estimate the dense optical flow of the intensity image pair I_1, I_2 by a neural network, such as FlowNet2 [28]. In respect to its accuracy it maps the planary motion flow where the velocity vector $f_{I_1,I_2} = [v_x\ v_y]^T$ describes the motion between the point $\tilde{x} = [x\ y]^T$ and its correspondence $\tilde{x}' = [x'\ y']^T$. According to this we can equate that $\tilde{x}' = \tilde{x} + f_{I_1,I_2}$. In the next step we use the corresponding depth map pair D_1, D_2 and extend the equation as follows. We calculate $z_{D_1}(\tilde{x})$ as depth value in pixel units from the coordinates of $\tilde{x}$ and its depth correspondence $z'_{D_2}(\tilde{x}')$. As $\tilde{x}'$ is likely located in the subpixel area, we estimate z' by bilinear interpolation. The uncertainty of the measured depth grows with increasing distance to the camera. We factor this into the equation with the depth weight μ following Herbst *et al.* [21] who weight the depth against the optical flow. We notate $\mu = \frac{\sigma_z(1)}{\sigma_z(Z)}$ with $\sigma_z(Z)$ as depth resolution at depth Z. The uncertainty grows around quadratic with the distance in common RGB-D cameras such as Kinect [29]. As a result we receive a depth flow vector $v_z = \delta z = \mu\, z' - \mu\, z$ and extend the optical flow f to scene flow

$$s_{I_1,I_2,D_1,D_2} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}. \tag{1}$$

The motion point correspondence relation between the scene point $p_{I_1,D_1} = [x\ y\ z]^T$ and p'_{I_2,D_2} is therefore the following:

$$\begin{bmatrix} x' \\ y' \\ \mu\, z' \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} + \begin{bmatrix} x \\ y \\ \mu\, z \end{bmatrix} \tag{2}$$

However, the resulting scene flow map contains gaps where the optical flow vector exceeds the frame boundaries or the depth sensor doesn't provide any depth pixel information. We therefore smooth the depth map with a median filter to shrink or even close small gaps. As a result we receive a semi-dense scene flow map.

B. EGO-MOTION ESTIMATION

The calculated scene flow represents the motion flow of the scene in respect to the camera. It can be assumed that the dynamic objects take up a smaller area in the frame than the overall static background. If the velocity vectors on the static background are greater than zero, they must be induced by the ego-motion of the camera. Figure 3 illustrates this coherence with the ego-motion induced motion flow in green and the motion vectors from dynamic objects in orange. We want to find an appropriate transformation matrix that projects this ego-motion in respect to rotation and translation

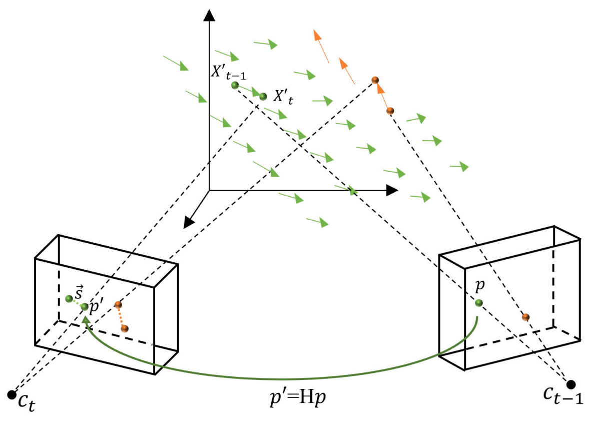

FIGURE 3. Geometric relation between the view of two spatial-temporal consecutive frames. The three dimensional movement of the projected points are estimated by a transformation matrix. Most scene flow vectors are caused by the camera motion between c_t and c_{t-1}. Vectors from dynamic objects (orange) tend to have conspicuous motion flows.

from the velocity vectors:

$$p'_i = Hp_i = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & h_{22} & h_{23} & h_{34} \\ h_{31} & h_{32} & h_{33} & h_{34} \\ h_{41} & h_{42} & h_{43} & h_{44} \end{bmatrix} \begin{bmatrix} x \\ y \\ \mu\, z \\ 1 \end{bmatrix} \tag{3}$$

H is a 4×4 transformation matrix with 15 degrees of freedom ($h_{44} = 1$). It can be determined with a set P of 5 point correspondences $p_{I_1,D_1} \leftrightarrow p'_{I_2,D_2}$ with homogeneous point coordinates $[x\ y\ z\ 1]^T$. Regular least squares methods for model estimations are very sensitive to outliers, in our case velocity vectors from dynamic objects. We therefore use the robust RANSAC method for rejecting outliers and make the assumption that a least 55 percent of the observed background is static. Considering that we use the minimum sample size of $s = 5$ point correspondences and want to gain a probability of $p = 0.98$ that we find a set without any outliers we come to following minimum number of iterations:

$$n = \frac{log(1 - 0.98)}{log(1 - (1 - 0.55)^5)} \approx 75 \tag{4}$$

We apply a preemptive RANSAC method with a time constrain of $n = 75$ for n model estimations and thus gain a chance of 98 percent for finding a valid sample from a static background without any outliers from dynamic objects. We verify the models by dividing the frame into $3 \cdot 4$ grid cells and taking 10 random samples from each of them. This sums up to an overall verifying set of 120 samples. The grid size is chosen in consideration of getting a good indication from all parts of the image while also sampling as sparsely as possible. Each estimated model is evaluated against a threshold and ranked according to their inlier count. First, the reprojection error vector of the model candidat against the samples is calculated:

$$\varepsilon = H \cdot p_{I_1,D_1} - p'_{I_2,D_2} \tag{5}$$

ε is the reprojection error vector between the model estimated $\tilde{p}'$ and the given p' from the point x' and can be described as $\varepsilon = [\tilde{v}_x\ \tilde{v}_y\ \tilde{v}_z\ 1]^T$. Formulated as euclidean distance this endpoint error (EPE) determines how close the projected flow is to our scene flow map.

We use a simple judgment rule

$$J(s_i) = \Big\{ inlier \mid ||\varepsilon_H(p_{I_1,D_1} \leftrightarrow p'_{I_2,D_2})|| < \tau \tag{6}$$

to check if the tested flow vector is consistent with the ego-motion velocity. The adaptive threshold T_1 can be described as

$$\tau = \begin{cases} \sigma(\hat{S}), & \text{if } \sigma(\hat{S}) \geq \gamma \\ \gamma, & \text{otherwise} \end{cases} \tag{7}$$

with $\hat{S} = \{||s_1||, ||s_2||, \ldots, ||s_N||\}$.

It consists of the standard deviation $\sigma(\hat{S})$ of the scene flow with respect to the median over all flow vectors in l_2 norm. We use the median instead of mean to diminish the influence of outliers and to receive a good indication about the distribution of the camera flow velocities. The standard deviation can reduce rapidly in scenes with low dynamic objects and minor ego-motion. As a result even small reprojection errors rise above the threshold that leads to the inlier rejection of many static background flow vectors. Depending on the distance of the objects in the scene the camera induced flow vectors can vary in velocity. We therefore use a max function to obtain a minimum threshold of the hyperparameter γ to ensure that also flow vectors on the edge of the ego-flow distribution retain as inliers.

The tested inliers are saved along with it transformation model. After n tested models we refine the best one by estimating it again with all its inliers using the single value decomposition (SVD). We then receive an appropriate estimation of the ego-motion scene flow on all levels from the overdetermined system

$$\hat{P}' \approx H\hat{P} \tag{8}$$

with $\hat{P} = [p_1\ p_2\ \ldots\ p_n]$ using all n inlier point correspondences.

C. DYNAMIC PIXEL SEGMENTATION

With the last step of refining the transformation matrix we calculate a general but precious projection model for the ego-motion indicating scene flow. We can now generate a projection error mask by calculating the reprojection error for every scene flow vector in the image using equation (5). The greater the error the more likely the corresponding flow vector belongs to a dynamic object. Consequently, we can set a threshold to distinguish between dynamic and static image pixels to obtain a segmentation mask M.

$$M_t = \{p_t \mid ||\varepsilon_H(p_{I_1,D_1} \leftrightarrow p'_{I_2,D_2})|| > T \tag{9}$$

where

$$T = \alpha \cdot \tau \tag{10}$$

The threshold T is combination of our prior used τ and the hyperparamter α. By tuning α the rigor of the judgement can be regulated to choose what level of dynamics

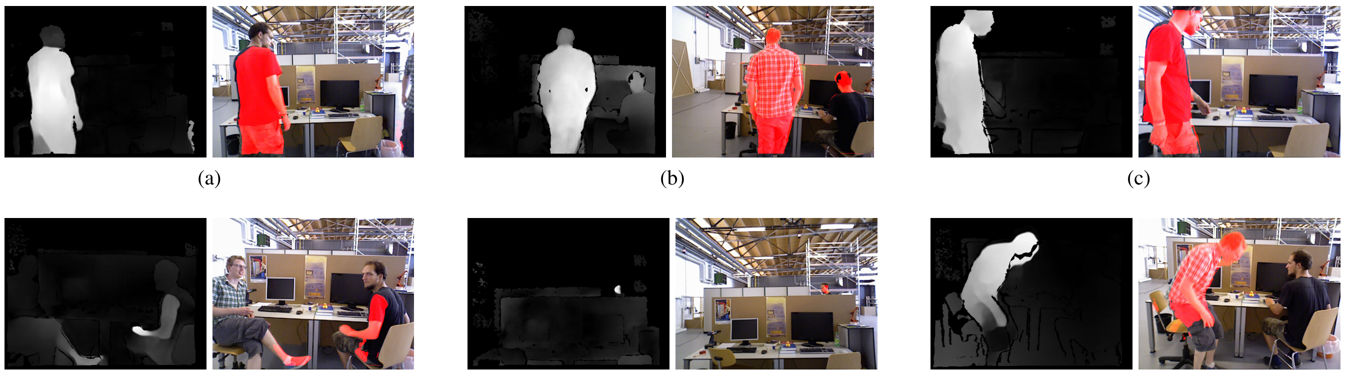

FIGURE 4. Qualitative results of our approach. Image pairs of normalized reprojection error maps and corresponding image masks.

is accepted. The lower the threshold the more pixels with even little motion are rejected. On the other hand a low threshold increases the risk of falsely declaring static pixels as dynamic. In general an ideal scene only contains two planes, where one plane belongs to one or multiple dynamic objects and the other one is static. In this case all static scene flow vectors would hold the same distance to the camera and therefore consists of very similar velocities. The calculated homography can project a very precise camera movement and would enable to reject dynamic pixels with a very strict threshold. The more static planes are in the scene the more general the homography has to be estimated to cover all of them. This can lead to a bad initial position for finding low-key dynamic outliers. We empirically found that $\gamma = 1.5$ and $\alpha = 0.7$ offer the best trade off for rejecting dynamic velocities sensitively while keeping falsely rejections of static parts to a minimum.

IV. EXPERIMENTAL RESULTS

We implemented our method as preprocessing step in the high-performant ORB-SLAM2 [1] system and evaluated it in the public TUM RGB-D dataset [30]. First, we examine the influence of the RANSAC iteration number on the estimation of a well fitting transformation matrix. In the next step we show the influence of our method on ORB-SLAM2 by performing it with and without our proposed method implemented. In a third step we compare our results to other state-of-the-art slam systems for dynamic environments. We chose those that similarly perform *adhoc* segmentation of dynamic image areas without making assumption about a priori potential dynamic objects by semantic interpretations. Finally we will analyze a computational costs of our method.

All experiments are executed on the TUM RGB-D dataset that contains 39 indoor sequences with varying dynamic environments and ego-motion. The sequences are recorded with a Microsoft Kinect camera and consist of RGB and depth images along with the ground-truth trajectory provided by a high accuracy motion capture system. 9 sequences contain scenes with dynamic objects and can be categorised in two different classes. Sequences titled with *sitting* show two people are sitting at a desk causing only little motion by gesturing and minor body movements. In *walking* scenes the same two persons from the desk are walking around which causes challenging dynamic motions. The camera motion is categorized in four different types: *static* - the camera is held in the hands statically, *xyz* - the camera moves along the x,y and z axes, *rpy* - the camera performs a rotation along the roll, pitch and yaw axes, *halfsphere (hs)* - the camera motion forms a half sphere.

A. RANSAC ITERATIONS

In 4 we calculated the RANSAC iteration parameter $n \approx 75$ based on the minimum sample set $s = 5$, the chance of finding a valid sample $p = 0.98$ and the assumption that the inlier ration is $p_s = 0.55$. This approximation can become very inaccurate, when dynamic objects get near the camera and thus are able to capture more than the arbitrary 45 percent of the frame. Given that the static background shrinks to only half of image area the desired iteration number would rapidly make n grow to 123. To get a better understanding of the RANSAC's behaviour towards the iteration number and the finding of an appropriate setting we will further analyse our RANSAC model in multiple real world scenarios. Figure 5 shows the results of running sequences from the TUM RGB-D dataset with increasing iteration numbers from $n = 1$ to $n = 150$ compared with its resulting average inlier count per sequence run. The number of inliers can change depending on the sequence. We therefore use normalized inlier rates in respect to the max inlier count of each sequence, which indicates the find of a best possible transformation model. It shows that in the very dynamic scenes *w-xyz* and *w-halfsphere* the inlier ratio is rapidly increasing from $n = 1$ to $n = 30$ and experiences a growing decline in the slope afterwards. At $n = 75$ the best model receives support from about 95% of the maximum number of inliers. The maximum is reached between $n = 85$ and $n = 90$. In the *walking-static* sequence with a low degree of ego-motion the first model candidate receive already over 90% of the inlier support. This ratio increase until it reaches its maximum at $n = 15$. At the sample number of 22, the the inlier ratio is stable at its maximum. In this scene a fixed $n = 22$ would be sufficient

TABLE 1. Comparison of different RMSE of absolute trajectory errors [m] of different state-of-the-art methods compared to ours.

Sequence	Motion Removal [8]			DynaSLAM(G[1]) [10]			Wang et al. [9]			Our method		
	w/o MD [m]	w/ MD [m]	Improv. [%]	w/o MD [m]	w/ MD [m]	Improv. [%]	w/o MD [m]	w/ MD [m]	Improv. [%]	w/o MD [m]	w/ MD [m]	Improv. [%]
sitting-hs	0.062	0.047	23.70	0.020	**0.018**	10.00	0.023	0.020	13.04	0.025	**0.018**	**28.00**
sitting-xyz	0.051	0.048	**4.55**	0.009	**0.009**	0	-	-	-	0.009	0.011	-22.22
walking-hs	0.529	0.125	76.32	0.351	0.035	90.03	0.689	0.312	54.72	0.511	**0.034**	**93.35**
walking-xyz	0.597	0.093	84.38	0.459	0.312	32.03	0.733	0.305	59.01	0.679	**0.018**	**97.35**
walking-rpy	0.730	**0.133**	81.75	0.662	0.251	59.64	0.552	0.498	9.78	0.801	0.144	**82.02**
walking-static	0.212	0.066	69.06	0.090	**0.009**	90.00	0.384	0.308	19.71	0.384	**0.009**	**97.66**

[1] Multi view geometry method.

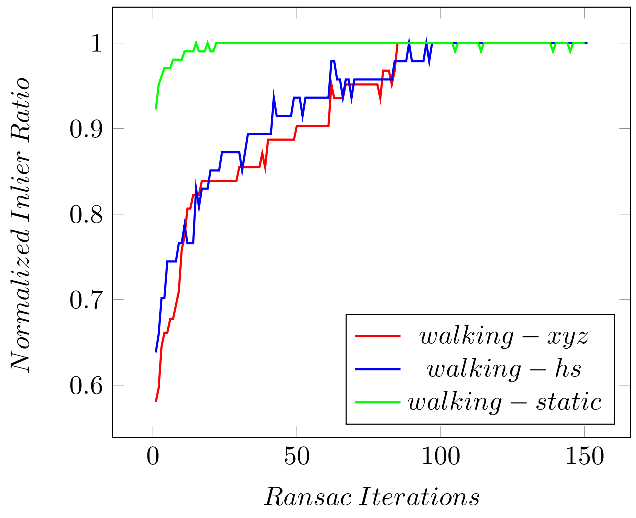

FIGURE 5. Inlier ratio towards different RANSAC iteration numbers *n*.

to gain the highest possibility of finding a valid sample that is supported by a maximum number of samples from the verifying set. The proposed $n = 75$ is therefore very conservative in this specific scene. In sequences with more ego-motion (*w-hs, w-xyz*) the chance of receiving a better fitting model can increase slightly with $n > 75$ with a declining improvement rate, but comes with a linear growing processing time rate as drawback (further discussed in section IV-C). Thus, we will retain the prior proposed RANSAC iteration number of $n = 75$ for finding a well fitting model in a reasonable time.

B. TUM RGB-D DATASET RMSE

Figure 4 shows qualitative results of our approach based on the *w-xyz* sequence. We were able to precisely segment dynamic image parts based on the visualized reprojection error maps as shown *e.g.* in figure 4a with a moving body, but static head and in figure 4b with remained static background between the arms and the upper body. In figure 4e even a far distant moving head could be segmented as dynamic image area. Figure 4f shows a black gap under the head of the successfully segmented person. In this area the depth map doesn't provide depth information and couldn't therefore analyzed for dynamic objects.

TABLE 2. RMSE of the absolute trajectory error [m] with and without our method.

Sequence	ORB-SLAM2		Our method		Improvement	
	median	std	median	std	median	std
s_hs	0.025	0.052	**0.018**	**0.039**	**28.00%**	**25.00%**
s_xyz	**0.009**	0.001	0.011	0.001	-22,22%	0%
w_hs	0.511	0.173	**0.034**	**0.091**	**93.35%**	**47,40%**
w_xyz	0.679	**0.079**	**0.018**	0.123	**97.35%**	-55,70%
w_rpy	0.801	0.128	**0.144**	**0.111**	**82.02%**	**13.28%**
w_static	0.384	0.031	**0.009**	**0.004**	**97.66%**	**87.10%**

Table 2 shows the results for five sequences of the open source ORB-SLAM2[1] [1] and in comparison with an extended version with our method implemented as preprocessing step. We use the median of ten iterations for each sequence to reduce the non-deterministic impact from the systems on the results. The used error metric is the commonly used absolute trajectory root mean square error (RMSE) and its standard deviation over ten iterations to examine the robustness. The absolute trajectory error between the estimated trajectory and the corresponding ground truth trajectory at time i is defined as

$$E_i := Q_i^{-1} S P_i, \tag{11}$$

where S is a rigid-body transformation that maps the estimated trajectory P onto the ground truth trajectory Q. The error of the sequence is the average deviation from the ground truth trajectory per frame:

$$ATE_{rmse} := \left(\frac{1}{n} \sum_{i=1}^{n} ||trans(E_i)||^2 \right)^{1/2}. \tag{12}$$

The results of the ORB-SLAM2 without motion removal show a strong increase of the error rate in high dynamic *walking* scenes towards the *sitting* sequences with a small degree of dynamic motion. This effect is intensified by scenes with high ego-motion rates such as *halfsphere* and *rpy* towards more static camera scenes like *static*. The combination of vigorous ego-motion and a dynamic environment leads besides the inaccurate odometry estimation to short time tracking losses and varying feature point tracking behaviour,

[1]https://github.com/raulmur/ORB_SLAM2

which reduces the robustness and results in a higher standard deviation.

The implementation of our proposed method leads to a significant improvement of the RMSE in almost all sequences. The best results are archived in scenes with low ego-motion and very dynamic objects. The accuracy dispersion is mostly improved in the same circumstances, but aggravates in scenes with low degree of motions. The reason can be found by the vanishing threshold that is induced by the low standard deviation of the flow velocities. This demands a fine-tuning of the γ parameter from (7). But due to the overall low dynamics a loose setted γ will accept a lot of dynamic pixels which results in a higher RMSE. A more tighter setting will squeeze out dynamic elements, but also noisy static ones. In this case the standard deviations of the results increase as shown in table 2 in the *sitting* sequences. We achieve an improvement of over 90% in all *walking* scenes except one, with two improvement rates of over 95%. The reliability of getting the same results in multiple iterations are enhanced in three of five cases and only worsened in one case. Examining this case showed us that the high standard deviation is caused by a few frames from the scene where a person is walking straight orthogonal towards the image plane. This motion is captured by the interpolated depth vectors, but due to its weight factor the velocity is reduced and lays right on the verge for getting rejected as dynamic motion. As a result the feature points on the walking person are accepted in a few iterations and lead to strong distortion of the trajectory.

Table 1 shows our results compared to other state-of-the-art motion removal approaches. In addition to the trajectory error error rate we also consider the relative improvement from each original SLAM system towards its motion detection (MD) enhanced version. Our method outperforms other state-of-the-art algorithm in most cases and achieves higher improvement rates in even more sequences. The most accurate outcomes are attained in scenes with translational or few ego-motion in combination with high dynamic environments. The biggest benefit is achieved in *w-xyz* with a RMSE of 0.018 m, which denotes an improvement of 80.65 % towards the second best motion removal approach [8] with 0.093 m. In sequences with less dynamic objects we achieve similar results as DynaSlam with its multi view geometry approach as shown in *s-halfsphere* and *w-static*. In *s-xyz* we get a slightly less accurate trajectory towards the state-of-the-art. In this scene dynamic motions are very small and have no impact on the trajectory estimation, which can be seen in the results from the other state-of-the-art methods. An additional preprocessing step for filtering is therefore not necessary and even brings the risk for impairing the results.

C. TIMING

The average computation cost of finding a model grows linear towards the increasing iteration number. With the proposed $n = 75$ the computational cost of the finding the transformation matrix is 60 ms. This refers only to the model estimation and the calculating of the scene flow, which gets generated on the fly. In case of a direct SLAM a segmentation map has to be calculated afterwards. For featured based SLAM method it might be more handy and quicker to analyze only those scene flow vectors, that are laying on a feature point. The average processing times for each step are the following:

• Model Estimation ($n = 75$):	60.0 ms
• Segmentation Mask:	15.7 ms
• Feature Rejection:	1.2 ms

Creating the segmentation mask consists of calculating the reprojection error for every valid scene flow vector in the frame. For dynamic keypoint rejection we analyze a $3 \cdot 3$ pixel patch around every feature point for dynamic behaviour. As for ORB-SLAM2 with default 1000 feature points we can reduce the computation time to averagely 1.19 ms. All processing steps are done on an Intel i7-6850K (6 cores @ 3.6GHz) with 32Gb RAM and optimized with the EIGEN library [31]. Additional computation costs are coming from generating the optical flow. The inference time depends on the used model and hardware.

V. CONCLUSION

We presented a novel method for the pixel-wise segmentation of dynamic elements to improve the robustness and accuracy of RGB-D SLAM systems. Our approach can be used in environments without a priori knowledge about possible dynamic objects and can be implemented in feature based SLAM as well as in direct SLAM thanks to its pixel-wise segmentation procedure. Our method outperforms almost any other comparable motion removal approaches on the TUM RGB-D dataset and achieves its best results in scenes with a high challenging dynamic objects and translatory or few ego-motion.

A limiting constraint is the choice of appropriate hyperparameters for the threshold calculation of 7 and 10, as an optimal refinement would require a priori information about the degree of the dynamics in the environment and of the ego motion. The dynamic adjustment of the parameters based on a measurement window of the rejection rates in recent frames could be a possible future improvement to counteract against over- and undersegmentation.

Another promising prospect is the integration of depth information from stereo images instead of RGB-D, which could increase the resolution and the accuracy of the scene flow. This would enable to leverage the depth weighting factor to exploit potential for further improvements of the robustness as discussed in section IV-B for the $xyz(w)$ sequence.

REFERENCES

[1] R. Mur-Artal and J. D. Tardos, "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.

[2] T. Whelan, S. Leutenegger, R. Moreno, B. Glocker, and A. Davison, *Elasticfusion*. Newbury Park, CA, USA: Sag, 2016, pp. 1697–1716, doi: 10.1177/0278364916669237.

[3] J. Engel, T. Schöps, and D. Cremers, "Lsd-SLAM: Large-scale direct monocular SLAM," in *Computer Vision*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham, Switzerland: Springer, 2014, pp. 834–849.

[4] P. F. Alcantarilla, J. J. Yebes, J. Almazan, 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.

[5] 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.

[6] D.-H. Kim and J.-H. Kim, "Effective background model-based RGB-D dense visual odometry in a dynamic environment," *IEEE Trans. Robot.*, vol. PP, pp. 1–9, 10 2016.

[7] 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.

[8] 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.

[9] 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.

[10] B. Bescos, J. M. Facil, 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.

[11] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, pp. 1–9, Oct. 2017.

[12] H. Liu, G. Liu, G. Tian, S. Xin, and Z. Ji, "Visual SLAM based on dynamic object removal," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2019, pp. 596–601.

[13] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, pp. 1–6, Apr. 2018.

[14] C. Yu, Z. Liu, X. Liu, F. Xie, Y. Yang, Q. Wei, and F. Qiao, "DS-SLAM: A semantic visual SLAM towards dynamic environments," in *Proc. Int. Conf. Intell. Robot. Syst.*, Dec. 2018, pp. 1168–1174.

[15] T. Zhang, H. Zhang, Y. Li, Y. Nakamura, and L. Zhang, "Flowfusion: Dynamic dense RGB-D SLAM based on optical flow," Tech. Rep., Mar. 2020.

[16] D. Sun, X. Yang, M. Liu, and J. Kautz, "Pwc-net: CNNs for optical flow using pyramid, warping, and cost," *CoRR*, vol. abs/1709.02371, pp. 1–9, Oct. 2017.

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

[18] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, 1999, pp. 722–729.

[19] A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers, "Stereoscopic scene flow computation for 3D motion understanding," *Int. J. Comput. Vis.*, vol. 95, no. 1, pp. 29–51, Oct. 2011.

[20] A. Letouzey, B. Petit, and E. Boyer, "Scene flow from depth and color images," *7th IEEE Int. Conf. Comput. Vis.*, Aug. 2011, pp. 46.1–46.11.

[21] E. Herbst, X. Ren, and D. Fox, "RGB-D flow: Dense 3-D motion estimation using color and depth," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 2276–2282.

[22] J. Quiroga, F. Devernay, and J. Crowley, "Local/global scene flow estimation," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2013, pp. 3850–3854.

[23] M. Jaimez, M. Souiai, J. Gonzalez-Jimenez, and D. Cremers, "A primal-dual framework for real-time dense RGB-D scene flow," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 98–104.

[24] W.-C. Ma, S. Wang, R. Hu, Y. Xiong, and R. Urtasun, "Deep rigid instance scene flow," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3614–3622.

[25] Y.-L. Qiao, L. Gao, Y.-K. Lai, F.-L. Zhang, M. Yuan, and S. Xia, "SF-net: Learning scene flow from RGB-D images with CNNs," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, Newcastle, U.K., Sep. 2018, p. 281.

[26] R. Schuster, O. Wasenmäller, C. Unger, G. Kuschk, and D. Stricker, "SceneFlowFields++: Multi-frame matching, visibility prediction, and robust interpolation for scene flow estimation," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 527–546, Feb. 2020.

[27] R. Schuster, C. Bailer, O. Wasenmäller, and D. Stricker, "Combining stereo disparity and optical flow for basic scene flow," in *Proc. CVT*, Kaiserslautern, Germany, Mar. 2018, pp. 1–5.

[28] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1647–1655.

[29] K. Khoshelham and S. Oude Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, pp. 1437–1454, Dec. 2012.

[30] 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. 1–5.

[31] G. Guennebaud. (2010). *Eigen v3*. [Online]. Available: http://eigen.tuxfamily.org

THORSTEN HEMPEL received the B.S. degree in industrial engineering from the West Coast University of Applied Sciences, Heide, Germany, in 2017, and the M.S. degree in industrial engineering from Otto von Guericke University Magdeburg, Magdeburg, Germany, in 2019, where he is currently pursuing the Ph.D. degree with the Neuro-Information Technology Group. His research interests include the development of algorithms in the field of robot vision, robot navigation, and human–robot interaction.

AYOUB AL-HAMADI received the Ph.D. degree in technical computer science, and the Habilitation degree in artificial intelligence and the Venia Legendi degree in pattern recognition and image processing from Otto von Guericke University Magdeburg, Germany, in 2001 and 2010, respectively. He is currently a Professor and the Head of the Neuro-Information Technology Group, Otto von Guericke University Magdeburg. He is the author of more than 350 papers in peer-reviewed international journals, conferences, and books. His research interests include computer vision, pattern recognition, and image processing. See `http://www.iikt.ovgu.de/al_hamadi`.html for more details.