SF2SE3: Clustering Scene Flow into SE(3)-Motions via Proposal and Selection

Leonhard Sommer, Philipp Schröppel, and Thomas Brox

University of Freiburg, Germany
{sommerl, schroepp, brox}@cs.uni-freiburg.de
https://lmb.informatik.uni-freiburg.de

Abstract. We propose SF2SE3, a novel approach to estimate scene dynamics in form of a segmentation into independently moving rigid objects and their $SE(3)$-motions. SF2SE3 operates on two consecutive stereo or RGB-D images. First, noisy scene flow is obtained by application of existing optical flow and depth estimation algorithms. SF2SE3 then iteratively (1) samples pixel sets to compute $SE(3)$-motion proposals, and (2) selects the best $SE(3)$-motion proposal with respect to a maximum coverage formulation. Finally, objects are formed by assigning pixels uniquely to the selected $SE(3)$-motions based on consistency with the input scene flow and spatial proximity.

The main novelties are a more informed strategy for the sampling of motion proposals and a maximum coverage formulation for the proposal selection. We conduct evaluations on multiple datasets regarding application of SF2SE3 for scene flow estimation, object segmentation and visual odometry. SF2SE3 performs on par with the state of the art for scene flow estimation and is more accurate for segmentation and odometry.

Keywords: low-level vision and optical flow · clustering · pose estimation · segmentation · scene understanding · 3D vision and stereo.

1 Introduction

Knowledge about dynamically moving objects is valuable for many intelligent systems. This is the case for systems that take a passive role as in augmented reality or are capable of acting as in robot navigation and object manipulation.

In this work, we propose a novel approach for this task that we term Scene-Flow-To-$SE(3)$ (SF2SE3). SF2SE3 estimates scene dynamics in form of a segmentation of the scene into independently moving objects and the $SE(3)$-motion for each object. SF2SE3 operates on two consecutive stereo or RGB-D images. First, off-the-shelf optical flow and disparity estimation algorithms are applied to obtain optical flow between the two timesteps and depth maps for each timestep. The predictions are combined to obtain scene flow. Note that the obtained scene flow is noisy, especially in case of occlusions. SF2SE3 then iteratively (1) samples pixel sets to compute $SE(3)$-motion proposals, and (2) selects the best $SE(3)$-motion proposal with respect to a maximum coverage formulation. Finally, objects are created for the selected $SE(3)$-motions by grouping pixels

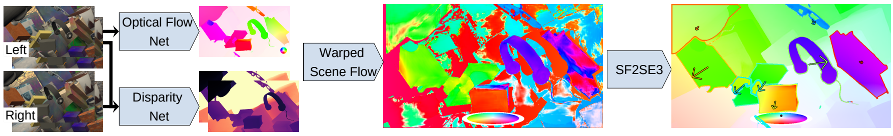

Fig. 1. Overview of SF2SE3: Optical flow and disparity are estimated with off-the-shelf networks and combined to an initial scene flow estimate. The resulting scene flow is noisy due to inaccurate estimations and due to occlusions (center image). SF2SE3 optimizes a set of objects with corresponding $SE(3)$-motions and an assignment from pixels to objects such that the initial scene flow is best covered with a minimal number of objects. Finally, an improved scene flow estimate can be derived from the object segmentation and $SE(3)$-motions, as shown in the last image. Furthermore, SF2SE3 obtains the camera egomotion by determining the $SE(3)$-motion of the background.

based on concistency of the input scene flow with the object $SE(3)$-motion. Further, SF2SE3 derives scene flow and the camera egomotion from the segmentation and $SE(3)$-motions. The described pipeline is illustrated in Fig 1.

Regarding related work, SF2SE3 is most similar to ACOSF [12] in that it approaches the problem as iteratively finding $SE(3)$-motion proposals and optimizing an assignment of pixels to the proposals. However, SF2SE3 introduces several improvements. Firstly, instead of randomly accumulating clusters that serve to estimate $SE(3)$-motion proposals, we propose a more informed strategy that exploits a rigidity constraint, *i.e.* forms clusters from points that have fixed 3D distances. Secondly, we propose a coverage problem formulation for selecting the best motion proposal such that (a) the input scene flow of all data points is best covered, and (b) irrelevant or similar $SE(3)$-motions are prohibited. The iterative process ends when no proposal fulfills the side-constraints from (b), whereas ACOSF iteratively selects a fixed number of $SE(3)$-motions.

We evaluate SF2SE3 and compare to state-of-the-art approaches for multiple tasks and datasets: we evaluate scene flow estimation on the KITTI [17] and FlyingThings3D [16] datasets, moving object segmentation on FlyingThings3D, and visual odometry on FlyingThings3D, KITTI and TUM RGB-D [19]. We use the state-of-the-art approaches CamLiFlow [13], RAFT-3D [22], RigidMask [30] and ACOSF as strong baselines for the comparison. CamLiFlow, RAFT-3D and RigidMask are currently the best approaches on the KITTI leaderboard and ACOSF is the most similar baseline to SF2SE3.

Compared to RAFT-3D, SF2SE3 obtains similar scene flow outlier rates on KITTI (-0.45%) and FlythingThings3D ($+0.47\%$). However, the advantage is additional output information in form of the object segmentation as opposed to pixel-wise motions.

Compared to RigidMask, the scene flow outlier rate of SF2SE3 is slightly worse on KITTI ($+0.43\%$) but significantly better on FlyingThings3D (-6.76%), which is due to assumptions about blob-like object shapes within RigidMask. Regarding object segmentation, SF2SE3 achieves higher accuracy than RigidMask on FlyingThings3D ($+2.59\%$).

Compared to ACOSF, SF2SE3 decreases the scene flow outlier rate on KITTI (-2.58%). Further, the runtime is decrease from 5 minutes to 2.84 seconds.

On all evaluated dynamic sequences of the TUM and Bonn RGB-D dataset, SF2SE3 achieves, compared to the two-frame based solutions RigidMask and VO-SF, the best performance.

To summarize, SF2SE3 is useful to retrieve a compressed representation of scene dynamics in form of an accurate segmentation of moving rigid objects, their corresponding $SE(3)$-motions, and the camera egomotion.

2 Related Work

In the literature, different models for the scene dynamics between two frames exist: (1) non-rigid models estimate pointwise scene flow or $SE(3)$-transformations, and (2) object-rigid models try to cluster the scene into rigid objects and estimate one $SE(3)$-transformation per object. Regarding the output, all models allow to derive pointwise 3D motion. Object-rigid models additionally provide a segmentation of the scene into independently moving objects. Furthermore, if an object is detected as static background, odometry information can be derived.

Our work falls in the object-rigid model category, but we compare to strong baselines from both categories. In the following, we give an overview of related works that estimate scene dynamics with such models.

2.1 Non-Rigid Models

Non-rigid models make no assumptions about rigidity and estimate the motion of each point in the scene individually as scene flow or $SE(3)$-transformations.

The pioneering work of Vedula *et al.* [24] introduced the notion of scene flow and proposed algorithms for computing scene flow from optical flow depending on additional surface information.

Following that, multiple works built on the variational formulation for optical flow estimation from Horn and Schunk [5] and adapted it for scene flow estimation [6,28,1,23,25,9].

With the success of deep learning on classification tasks and with the availability of large synthetic datasets like Sintel [3] and FlyingThings3D [16], deep learning models for the estimation of pointwise scene dynamics have been proposed [11,14,7,29,18,22]. In particular, in this work, we compare with RAFT-3D [22], which estimates pixel-wise $SE(3)$-motions from RGB-D images. RAFT-3D iteratively estimates scene flow residuals and a soft grouping of pixels with similar 3D motion. In each iteration, the residuals and the soft grouping are used to optimize pixel-wise $SE(3)$-motions such that the scene flow residuals for the respective pixel and for grouped pixels are minimized.

2.2 Object-Rigid Models

Object-rigid models segment the scene into a set of rigid objects and estimate a $SE(3)$-transformation for each object. The advantages compared to non-rigid models are a more compressed representation of information, and the availability of the object segmentation. The disadvantage is that scene dynamics cannot be correctly represented in case that the rigidity assumption is violated.

Classical approaches in this category are PRSM [26,27] and OSF [17], which split the scene into rigid planes based on a superpixelization and assign $SE(3)$-motions to each plane. MC-Flow [10] estimates scene flow from RGB-D images by optimizing a set of clusters with corresponding $SE(3)$-motions and a soft assignment of pixels to the clusters. A follow-up work [8] splits the clusters into static background and dynamic objects and estimates odometry and dynamic object motion separately.

An early learned object-rigid approach is ISF [2], which builds on OSF but employs deep networks that estimate an instance segmentation and object coordinates for each instance. The later approach DRISF [15] employs deep networks to estimate optical flow, disparity and instance segmentation and then optimizes a $SE(3)$-motion per instance such that is consistent with the other quantities.

In this work, we compare to the more recent learned approaches ACOSF [12] and RigidMask [30]. ACOSF takes a similar approach as OSF but employs deep networks to estimate optical flow and disparity. RigidMask employs deep networks to estimate depth and optical flow and to segment static background and dynamic rigid objects. Based on the segmentation, $SE(3)$ motions are fit for the camera egomotion and the motion of all objects. A key difference between RigidMask and our approach is that RigidMask represents objects with polar coordinates, which is problematic for objects with complex structures. In contrast, our approach takes no assumptions about object shapes.

3 Approach

In the following, we describe the proposed SF2SE3 approach. SF2SE3 takes two RGB-D images from consecutive timestamps τ_1 and τ_2 and the associated optical flow as input. While the optical flow is retrieved with RAFT [21], the depth is retrieved with a depth camera or LEAStereo [4] in the case of a stereo camera. Using the first RGB-D image as reference image, the corresponding depth at τ_2 is obtained by backward warping the second depth image according to the optical flow. Further, occlusions are estimated by applying an absolute limit on the optical flow forward-backward inconsistency [20]. The depth is indicated as unreliable in case of invalid measurements and additionally for the depth at τ_2 in case of temporal occlusions. SF2SE3 then operates on the set $\mathcal{D}$ of all image points of the reference image:

$$\mathcal{D} = \{\ D_i = (\underbrace{x_i, y_i, z_i, \rlap{$\overbrace{\phantom{p_i^{\tau_1}, p_i^{\tau_2}, r_i^{\tau_1}, u_i, v_i, d_i, r_i^{\tau_2}}}^{motion}$}p_i^{\tau_1}, p_i^{\tau_2}, r_i^{\tau_1}}_{spatial}, u_i, v_i, d_i, r_i^{\tau_2})\ \}, \tag{1}$$

where each image point D consists of its pixel coordinates (x, y), its depth z at τ_1, its 3D points (p^{τ_1}, p^{τ_2}), its optical flow (u, v), its warped disparity d at τ_2 and its depth reliability indications r^{τ_1} and r^{τ_2}.

The objective then is to estimate a collection of objects $\mathcal{O}$ where each object O consists of a point cloud $\mathcal{P}$ and a $SE(3)$-motion (R, t):

$$\mathcal{O} = \{\, O_k = (\underbrace{\mathcal{P}_k}_{spatial}, \overbrace{R_k, t_k}^{motion})\,\}. \tag{2}$$

3.1 Algorithm Outline

SF2SE3 aims to estimate objects $\mathcal{O}$ which explain or rather cover the downsampled image points $\mathcal{D}$. To quantify the coverage of an image point D by an object O, we introduce a motion inlier model $P_I^{motion}(D, O)$ and a spatial inlier model $P_I^{spatial}(D, O)$. These models are described in detail in Section 3.2. Based upon these models, objects $\mathcal{O}$ are retrieved iteratively, see Figure 2.

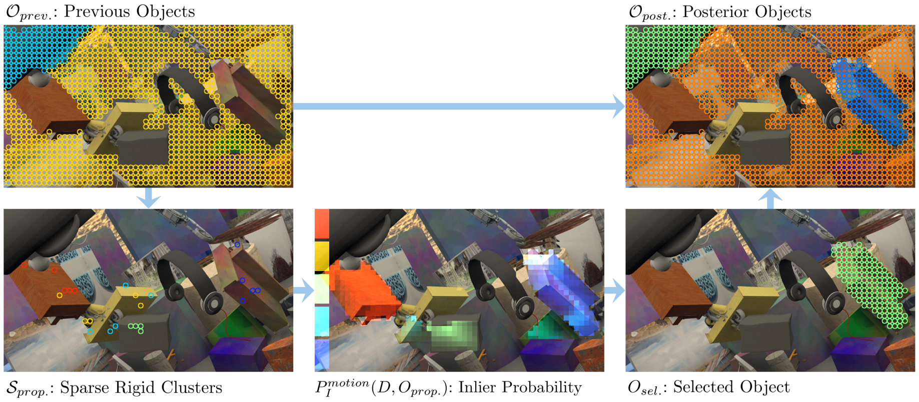
$\mathcal{O}_{prev.}$: Previous Objects
$\mathcal{O}_{post.}$: Posterior Objects
$\mathcal{S}_{prop.}$: Sparse Rigid Clusters
$P_I^{motion}(D, O_{prop.})$: Inlier Probability
$O_{sel.}$: Selected Object

Fig. 2. Cycle of Single Object Estimation: SF2SE3 estimates objects $\mathcal{O}$ iteratively. All image points which are not covered by any previous objects $\mathcal{O}_{prev.}$, and which depth is reliable, are accumulated to obtain sparse rigid clustered $\mathcal{S}_{prop.}$. This is described in Section 3.3. Fitting an $SE(3)$-motion to each cluster results in the proposed objects $\mathcal{O}_{prop.}$ which do not contain a spatial model. Based on the inlier probabilities for the proposed and the previous objects, $P_I^{motion}(D, O_{prop.})$ and $P_I(D, O_{prev.})$, the one object is selected which maximizes the coverage objective. The coverage problem is described in Section 3.4. After selecting a single object $O_{sel.}$, the image points which are covered based on the motion model $P_I^{motion}(D, O_{sel})$ are forming the point cloud which serves from then on as spatial model. Further, the selected object is split into multiple objects by splitting the point cloud into multiple spatially connected point clouds. This is not illustrated in the diagram for the sake of clarity. In case no proposed object has a sufficiently high coverage, the iterative process ends.

Finally, based on the obtained rigid objects $\mathcal{O}$, SF2SE3 derives odometry, segmentation and scene flow, which is described in Section 3.5. For this, one object is determined as background and each image point is assigned to one object based on the likelihood $f(D|O)$, which is introduced in Section 3.2.

For further implementation details, including parameter settings, we publish the source code at https://www.github.com/lmb-freiburg/sf2se3.

3.2 Consensus Models

To quantify the motion consensus between the scene flow of a data point and the $SE(3)$-motion of an object, we define a motion inlier probability $P_I^{motion}(D,O)$ and likelihood $f^{motion}(D|O)$. These are defined by separately imposing Gaussian models on the deviation of the data point's optical flow in x- and y-direction and the disparity d of the second time point from the respective projections π_u, π_v, π_d of the forward transformed 3D point p^{τ_1} according to the object's rotation R and translation t. Formally, this can be written as

$$\Delta u = u - \pi_u(Rp^{\tau_1} + t) \sim \mathcal{N}(0, \sigma_u^2) \tag{3}$$

$$\Delta v = v - \pi_v(Rp^{\tau_1} + t) \sim \mathcal{N}(0, \sigma_v^2) \tag{4}$$

$$\Delta d = d - \pi_d(Rp^{\tau_1} + t) \sim \mathcal{N}(0, \sigma_d^2). \tag{5}$$

Spatial proximity of the data point and the object's point cloud is measured with the likelihood $f^{spatial}(D|O)$ and the inlier probability $P_I^{spatial}(D,O)$. Therefore, we separately impose Gaussian models on the x-, y-, and z- deviation of the data point's 3D point p^{τ_1} from its nearest neighbor inside the object's point cloud $\mathcal{P}$. More precisely, we define the models

$$\Delta x = x - x^{NN} \sim \mathcal{N}(0, \sigma_{geo-2D}^2) \tag{6}$$

$$\Delta y = y - y^{NN} \sim \mathcal{N}(0, \sigma_{geo-2D}^2) \tag{7}$$

$$\Delta z_{rel} = \frac{z - z^{NN}}{\frac{z + z^{NN}}{2}} \sim \mathcal{N}(0, \sigma_{geo-depth-rel}^2). \tag{8}$$

The joint inlier probability for the spatial model yields

$$P_I^{spat.}(D,O) = \begin{cases} P_{I,Gauss.}(\Delta x) P_{I,Gauss.}(\Delta y) P_{I,Gauss.}(\Delta z_{rel}) & , r^{\tau_1} = 1 \\ P_{I,Gauss.}(\Delta x) P_{I,Gauss.}(\Delta y) & , r^{\tau_1} = 0 \end{cases}, \tag{9}$$

likewise the spatial likelihood $f^{spatial}(D|O)$ is calculated. Details for the calculation of the Gaussian inlier probability $P_{I,Gauss.}$ are provided in the Supplementary.

Regarding the motion model, the joint inlier probability yields

$$P_I^{mot.}(D,O) = \begin{cases} P_{I,Gauss.}(\Delta u) P_{I,Gauss.}(\Delta v) P_{I,Gauss.}(\Delta d) & , r^{\tau_1} = 1, r^{\tau_2} = 1 \\ P_{I,Gauss.}(\Delta u) P_{I,Gauss.}(\Delta v) & , r^{\tau_1} = 1, r^{\tau_2} = 0 \\ 1 & , else, \end{cases} \tag{10}$$

the same applies for the motion likelihood $f^{motion}(D|O)$.

Joining the motion and the spatial model, under the assumption of independence, results in

$$f(D|O) = f^{spatial}(D|O)f^{motion}(D|O) \tag{11}$$

$$P_I(D,O) = P_I^{spatial}(D,O)P_I^{motion}(D,O). \tag{12}$$

3.3 Proposals via Rigidity Constraint

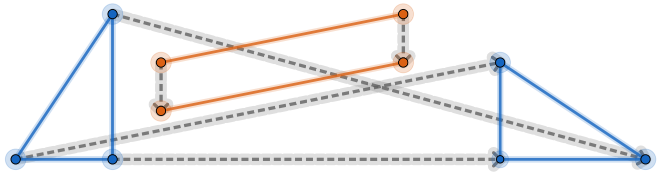

Fig. 3. Rigidity: Points of the same color are rigid, which means that the distance between each pair remains constant despite the movement. Two accumulated points are already sufficient to calculate new $SE(3)$-motion proposals.

Each proposed object is found by fitting an $SE(3)$-motion to a cluster of scene flow points which fulfill the rigidity constraint, see Figure 3. For each pair of points (p_i, p_j) it must hold

$$|\left\|p_i^{\tau_2} - p_j^{\tau_2}\right\| - \left\|p_i^{\tau_1} - p_j^{\tau_1}\right\|| < \delta_{rigid-dev-max}. \tag{13}$$

Clusters are instantiated by single scene flow points, which are sampled uniformly. Further points are iteratively added while preserving rigidity. Even though two points are already sufficient to estimate a $SE(3)$-motion, additional points serve robustness against noise.

3.4 Selection via Coverage Problem

Having obtained the $SE(3)$-motion proposals $\mathcal{O}_{prop.}$, we select the one which covers the most scene flow points which are not sufficiently covered by previous objects $\mathcal{O}_{prev.}$. Coverage is measured for the proposed objects with the motion model $P_I^{motion}(D,O)$. For previously selected objects, the spatial model is available in form of a point cloud. This allows us to use the joint model $P_I(D,O)$, consisting of motion and spatial model. Formally, we define the objective as

$$\max_{O \in \mathcal{O}_{prop.}} \frac{1}{|\mathcal{D}|} \sum_{D \in \mathcal{D}} \max \left[P_I^{motion}(D,O), \max_{\tilde{O} \in \mathcal{O}_{prev.}} P_I(D,\tilde{O}) \right]. \tag{14}$$

Separating the previous coverage results in

$$\max_{O \in \mathcal{O}_{prop.}} P_{contribute}(\mathcal{D}, O, \mathcal{O}_{prev.}) + \frac{1}{|\mathcal{D}|} \sum_{D_i \in \mathcal{D}} \max_{\tilde{O} \in \mathcal{O}_{prev.}} P_I(D, \tilde{O})), \tag{15}$$

with the contribution probability $P_{contribute}(\mathcal{D}, O, \mathcal{O}_{prev.})$ defined as

$$P_{contribute}(\mathcal{D}, O, \mathcal{O}_{prev.}) = \frac{1}{|\mathcal{D}|} \sum_{D \in \mathcal{D}} \max \left[P_I^{motion}(D, O) - \max_{\tilde{O} \in \mathcal{O}_{prev.}} P_I(D, \tilde{O}), 0 \right]. \tag{16}$$

To exclude irrelevant objects, we impose for each object a minimum contribution probability.

Moreover, to exclude duplicated objects, we impose for each pair of objects a maximum overlap probability, which we define as

$$P_{overlap}(\mathcal{D}, O_1, O_2) = \frac{\sum_{D \in \mathcal{D}} P_I(D, O_1) P_I(D, O_2)}{\sum_{D \in \mathcal{D}} P_I(D, O_1) + P_I(D, O_2) - P_I(D, O_1) P_I(D, O_2)}. \tag{17}$$

This overlap probability constitutes an extension of the intersection-over-union metric for soft assignments, e.g., probabilities.

Taken together, we formulate the optimization problem as

$$\max_{O \subseteq \mathcal{O}_{prop.}} P_{contrib.}(\mathcal{D}, O, \mathcal{O}_{prev.}) \tag{18a}$$

$$\text{subject to } P_{contrib.}(\mathcal{D}, O, \mathcal{O}_{prev.}) \geq \delta_{contrib.-min} \tag{18b}$$

$$P_{overlap}(\mathcal{D}, O, O_{prev.}) \leq \delta_{overlap-max} \quad \forall O_{prev.} \in \mathcal{O}_{prev}. \tag{18c}$$

An example for the calculation of contribution as well as overlap probability is provided in Figure 4.

Automatically the algorithm ends when the contribution probability falls below the minimum requirement.

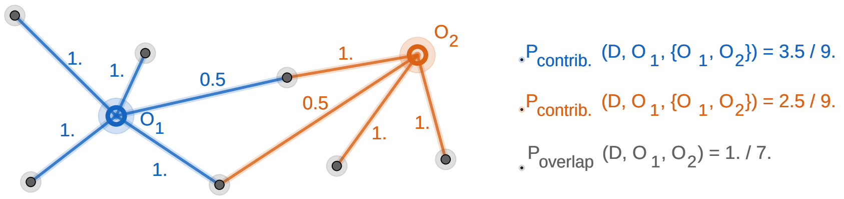
1.
1.
0.5
1.
O2
1.
0.5
O1
1.
1.
1.
1.
$P_{contrib.}$ (D, O 1, {O 1, O2}) = 3.5 / 9.
$P_{contrib.}$ (D, O 1, {O 1, O2}) = 2.5 / 9.
$P_{overlap}$ (D, O 1, O2) = 1. / 7.

Fig. 4. Contribution and Overlap Probability for two Objects: The edge weights equal the inlier probabilities $P_I(D, O)$, a missing edge indicates $P_I(D, O) = 0$. Corresponding probabilities for contribution of each object and the overlap probability of both objects are calculated on the right side.

3.5 Deduction of Odometry, Image Segmentation, and Scene Flow

For estimating the odometry we determine the dynamic rigid object that equals the background. Assuming that the background equals the largest object, we choose the one that yields the largest contribution probability

$$O_{background} = \arg\max_{O \in \mathcal{O}} P_{contribute}(\mathcal{D}, O, \mathcal{O}). \tag{19}$$

Based on the maximum likelihood, we assign each pixel from the high-resolution image to one of the objects

$$\phi_{D,O} = \begin{cases} 1 & , \arg\max_{O_k} f(D|O_k) = O \\ 0 & \end{cases} . \tag{20}$$

Given the object assignment the scene flow s can be retrieved for each 3D point p^{τ_1} as

$$s = Rp^{\tau_1} + t - p^{\tau_1}. \tag{21}$$

4 Results

We compare the performance of our method against the state of the art regarding scene flow, segmentation, odometry, and runtime.

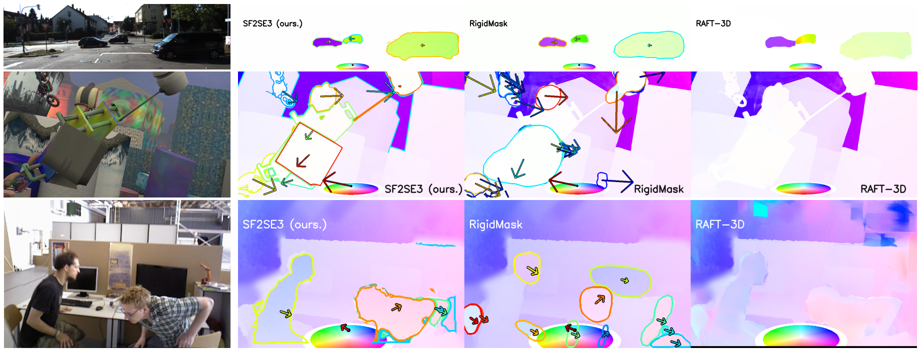

Fig. 5. Qualitative results of our approach, RigidMask, and Raft-3D (left to right) on KITTI-2015, FlyingThings3D, and TUM RGB-D (top to bottom). The scene flow is color coded for the x- and z-directions, as indicated by the color wheel. The edges of the object segmentation are highlighted and the motions of object centroids are indicated with arrows. The odometry is indicated with an arrow starting from the center of the color wheel. Note that RAFT-3D estimates no segmentation and odometry.

Table 1. Listed are the outlier percentages for disparities at both timestamps, optical flow, and scene flow. An outlier for optical flow and disparity implies a deviation from the ground truth of > 3 pxl absolutely and $> 5\%$ relatively. An outlier for scene flow implies an outlier for either disparity or optical flow.

Method	Dataset	D1 Out. [%]	D2 Out. [%]	OF Out. [%]	SF Out. [%]
ACOSF [12]	KITTI - test	3.58	5.31	5.79	7.90
DRISF [15]	KITTI - test	2.55	4.04	4.73	6.31
RigidMask [30]	KITTI - test	1.89	3.23	3.50	4.89
RAFT-3D [22]	KITTI - test	1.81	3.67	4.29	5.77
CamLiFlow [13]	KITTI - test	1.81	2.95	3.10	**4.43**
SF2SE3 (ours.)	KITTI - test	1.65	3.11	4.11	5.32
Warped Scene Flow	FT3D - test	2.35	16.19	9.43	19.09
RigidMask	FT3D - test	2.35	6.98	15.42	15.49
RAFT-3D	FT3D - test	2.35	4.40	8.47	**8.26**
SF2SE3 (ours.)	FT3D - test	2.35	4.86	8.76	8.73

Scene Flow To evaluate the performance of estimating scene flow, we measure the outlier percentages of disparity, optical flow and scene flow, in the same way as the KITTI-2015 benchmark [17]. The results for KITTI-2015 and FlyingThings3D are listed in Table 1.

Segmentation For the segmentation evaluation, we retrieve a one-to-one matching between predicted and ground truth objects with the Hungarian method and report the accuracy, *i.e.* the ratio of correctly assigned pixels. In addition to the accuracy, we report the average number of extracted objects per frame.

In Table 2 the results are listed for the FlyingThings3D dataset. The original ground truth segmentation can not be directly used, as it splits the background into multiple objects even though they have the same $SE(3)$-motion (Fig. 6 left). To resolve this, we fuse objects that have a relative pose error, as defined in Equation 24 and 25, below a certain threshold (Figure 6 right).

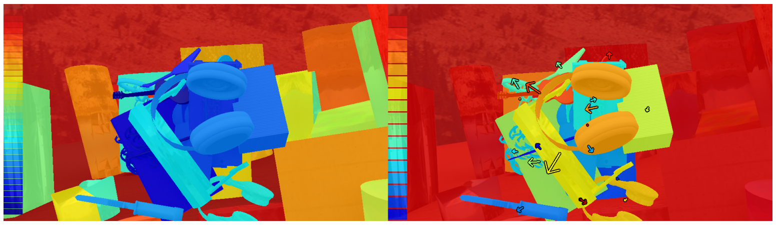

Fig. 6. Based on the segmentation of objects from the FlyingThings3D dataset, illustrated on the left side, the segmentation for evaluation, shown on the right side, is retrieved. To achieve this, we fuse objects with similar $SE(3)$-motion.

Table 2. Results for segmenting frames into moving objects. Metrics are the segmentation accuracy and the average objects count in each frame.

Method	Dataset	Segmentation Acc. [%]	Objects Count [#]
RigidMask	FT3D - test	80.71	16.32
SF2SE3 (ours.)	FT3D - test	**83.30**	7.04

Odometry We evaluate the odometry with the relative pose error, which has a translational part RPE_{transl} and a rotational part RPE_{rot}. These are computed from the relative transformation T_{rel} between the ground truth transformation ${}^{t_1}\hat{T}_{t_0}$ and the estimated transformation ${}^{t_1}T_{t_0}$, which is defined as follows:

$$T_{rel} = {}^{t_1}\hat{T}_{t_0}^{-1} \, {}^{t_1}T_{t_0} \tag{22}$$

$$T_{rel} = \begin{bmatrix} R_{rel} & t_{rel} \\ 0 & 1 \end{bmatrix}. \tag{23}$$

The translational and rotational relative pose errors RPE_{transl} and RPE_{rot} are computed as follows:

$$RPE_{transl} = \frac{\|t_{rel}\|}{t_1 - t_0} \text{ in } \frac{m}{s} \tag{24}$$

$$RPE_{rot} = \frac{\|w(R_{rel})\|}{t_1 - t_0} \frac{360}{2\pi} \text{ in } \frac{deg}{s}, \tag{25}$$

with $w(R_{rel})$ being the axis-angle representation of the rotation. We report the results on FlyingThings3D and TUM RGB-D in Table 3.

Table 3. Results for odometry estimation on FlyingThings3D and TUM RGB-D using the translation and rotational relative pose errors RPE_{transl} and RPE_{rot}.

Method	Dataset	RPE transl. [m/s]	RPE rot. [deg/s]
Static	FT3D - test	0.364	2.472
RigidMask	FT3D - test	0.082	0.174
SF2SE3 (ours.)	FT3D - test	**0.025**	**0.099**
Static	TUM FR3	0.156	18.167
RigidMask	TUM FR3	0.281	4.345
SF2SE3 (ours.)	TUM FR3	**0.090**	**3.535**

Runtime We report average runtimes of SF2SE3 and the baselines in Table 4. The runtimes were measured on a single Nvidia GeForce RTX 2080Ti.

5 Discussion

Our method performs on par with state-of-the-art methods of the KITTI-2015 scene flow benchmark, achieving a scene flow outlier rate similar to RigidMask

Table 4. Runtimes for different approaches on FlyingThings3D, KITTI-2015, and TUM RGB-D. If depth and optical flow are estimated separately and the runtime is known, it is listed. Runtimes in red are from the original authors on different hardware.

Method	Dataset	Depth [s]	Optical Flow [s]	Total [s]
ACOSF	KITTI - test	-	-	300.00
DRISF	KITTI - test	-	-	0.75
RigidMask	KITTI - test	1.46	-	4.90
RAFT-3D	KITTI - test	1.44	-	2.73
CAMLiFlow	KITTI - test	-	-	1.20
SF2SE3 (ours.)	KITTI - test	1.43	0.42	2.84
RigidMask	FT3D - test	1.60	-	8.54
RAFT-3D	FT3D - test	1.58	-	2.92
SF2SE3 (ours.)	FT3D - test	1.58	0.40	3.79
RigidMask	TUM FR3	0.23	-	2.34
RAFT-3D	TUM FR3	0.23	-	1.15
SF2SE3 (ours.)	TUM FR3	0.23	0.36	2.29

(+0.43%), CamLiFlow (+0.89%) and RAFT-3D (−0.45%). Further, on FlyingThings3D it achieves similar scene flow performance as the pointwise method RAFT-3D (−0.47%) and outperforms RigidMask significantly (−6.76%) while also achieving an higher segmentation accuracy (+2.59%). In contrast to RigidMask and others, our method generalizes better because supervision is only applied for estimating optical flow and depth. Therefore, we detect the pedestrians in Figure 7.

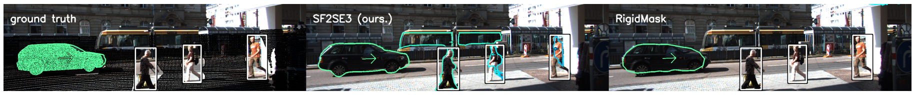

Fig. 7. Segmenting pedestrians on KITTI-2015: ground truth, estimate from SF2SE3, estimate from RigidMask (left to right). In contrast to RigidMask, the proposed SF2SE3 approach detects the pedestrians (marked with white bounding boxes) because it is not visually fine-tuned for cars. This problem of RigidMask is not reflected in the quantitative results because ground truth is missing for these points.

Moreover, our representation does not geometrically restrict object shapes. Thus, we are able to fit objects with complex shapes, as shown in Figure 8.

Compared to ACOSF, which is the most accurate method in scene flow on KITTI-2015 that estimates segmentation and takes no assumptions about object shapes, our method reduces the scene flow outlier percentage by (−2.58%) and the runtime from 300 seconds to 2.84 seconds. Furthermore, we expect ACOSF to perform even worse in case of more objects, e.g. in FlyingThings3D, as it uses random sampling for retrieving initial $SE(3)$-motions and assumes a fixed number of objects.

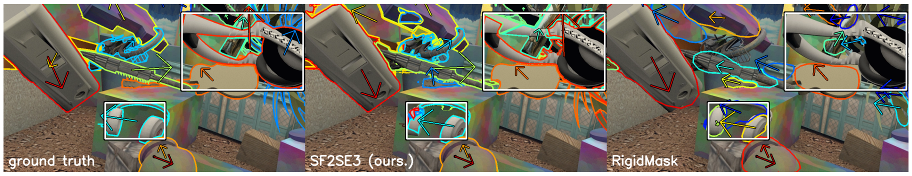

Fig. 8. Segmenting objects with complex shapes on FlyingThings3D: ground truth, estimate from SF2SE3, estimate from RigidMask (left to right). RigidMask over-segments the semi-circular shaped headphones on the bottom and misses the headphones on the right (marked with white bounding boxes).

6 Conclusion

We have proposed SF2SE3: a novel method that builds on top of state-of-the-art optical flow and disparity networks to estimate scene flow, segmentation, and odometry. In our evaluation on KITTI-2015, FlyingThings3D and TUM RGB-D, SF2SE3 shows better performance than the state of the art in segmentation and odometry, while achieving comparative results for scene flow estimation.

References

1. Basha, T., Moses, Y., Kiryati, N.: Multi-view scene flow estimation: A view centered variational approach. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 1506–1513 (2010). https://doi.org/10.1109/CVPR.2010.5539791
2. Behl, A., Hosseini Jafari, O., Karthik Mustikovela, S., Abu Alhaija, H., Rother, C., Geiger, A.: Bounding boxes, segmentations and object coordinates: How important is recognition for 3d scene flow estimation in autonomous driving scenarios? In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2574–2583 (2017)
3. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: European conference on computer vision. pp. 611–625. Springer (2012)
4. Cheng, X., Zhong, Y., Harandi, M., Dai, Y., Chang, X., Drummond, T., Li, H., Ge, Z.: Hierarchical neural architecture search for deep stereo matching. arXiv preprint arXiv:2010.13501 (2020)
5. Horn, B.K., Schunck, B.G.: Determining optical flow. Artificial intelligence **17**(1-3), 185–203 (1981)
6. Huguet, F., Devernay, F.: A variational method for scene flow estimation from stereo sequences. In: 2007 IEEE 11th International Conference on Computer Vision. pp. 1–7. IEEE (2007)
7. Hur, J., Roth, S.: Self-supervised monocular scene flow estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7396–7405 (2020)
8. Jaimez, M., Kerl, C., Gonzalez-Jimenez, J., Cremers, D.: Fast odometry and scene flow from rgb-d cameras based on geometric clustering. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 3992–3999. IEEE (2017)

9. Jaimez, M., Souiai, M., Gonzalez-Jimenez, J., Cremers, D.: A primal-dual framework for real-time dense rgb-d scene flow. In: 2015 IEEE international conference on robotics and automation (ICRA). pp. 98–104. IEEE (2015)
10. Jaimez, M., Souiai, M., Stückler, J., Gonzalez-Jimenez, J., Cremers, D.: Motion cooperation: Smooth piece-wise rigid scene flow from rgb-d images. In: 2015 International Conference on 3D Vision. pp. 64–72. IEEE (2015)
11. Jiang, H., Sun, D., Jampani, V., Lv, Z., Learned-Miller, E., Kautz, J.: Sense: A shared encoder network for scene-flow estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3195–3204 (2019)
12. Li, C., Ma, H., Liao, Q.: Two-stage adaptive object scene flow using hybrid cnn-crf model. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 3876–3883. IEEE (2021)
13. Liu, H., Lu, T., Xu, Y., Liu, J., Li, W., Chen, L.: Camliflow: Bidirectional camera-lidar fusion for joint optical flow and scene flow estimation. arXiv preprint arXiv:2111.10502 (2021)
14. Liu, X., Qi, C.R., Guibas, L.J.: Flownet3d: Learning scene flow in 3d point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 529–537 (2019)
15. Ma, W.C., Wang, S., Hu, R., Xiong, Y., Urtasun, R.: Deep rigid instance scene flow. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3614–3622 (2019)
16. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4040–4048 (2016)
17. Menze, M., Geiger, A.: Object scene flow for autonomous vehicles. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3061–3070 (2015)
18. Schuster, R., Unger, C., Stricker, D.: A deep temporal fusion framework for scene flow using a learnable motion model and occlusions. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 247–255 (2021)
19. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgb-d slam systems. In: 2012 IEEE/RSJ international conference on intelligent robots and systems. pp. 573–580. IEEE (2012)
20. Sundaram, N., Brox, T., Keutzer, K.: Dense point trajectories by gpu-accelerated large displacement optical flow. In: European conference on computer vision. pp. 438–451. Springer (2010)
21. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: European conference on computer vision. pp. 402–419. Springer (2020)
22. Teed, Z., Deng, J.: Raft-3d: Scene flow using rigid-motion embeddings. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8375–8384 (2021)
23. Valgaerts, L., Bruhn, A., Zimmer, H., Weickert, J., Stoll, C., Theobalt, C.: Joint estimation of motion, structure and geometry from stereo sequences. In: European Conference on Computer Vision. pp. 568–581. Springer (2010)
24. Vedula, S., Baker, S., Rander, P., Collins, R., Kanade, T.: Three-dimensional scene flow. In: Proceedings of the Seventh IEEE International Conference on Computer Vision. vol. 2, pp. 722–729. IEEE (1999)

25. Vogel, C., Schindler, K., Roth, S.: 3d scene flow estimation with a rigid motion prior. In: 2011 International Conference on Computer Vision. pp. 1291–1298. IEEE (2011)
26. Vogel, C., Schindler, K., Roth, S.: Piecewise rigid scene flow. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1377–1384 (2013)
27. Vogel, C., Schindler, K., Roth, S.: 3d scene flow estimation with a piecewise rigid scene model. International Journal of Computer Vision **115**(1), 1–28 (2015)
28. Wedel, A., Rabe, C., Vaudrey, T., Brox, T., Franke, U., Cremers, D.: Efficient dense scene flow from sparse or dense stereo data. In: European conference on computer vision. pp. 739–751. Springer (2008)
29. Yang, G., Ramanan, D.: Upgrading optical flow to 3d scene flow through optical expansion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1334–1343 (2020)
30. Yang, G., Ramanan, D.: Learning to segment rigid motions from two frames. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1266–1275 (2021)