RAFT-MSF: Self-Supervised Monocular Scene Flow Using Recurrent Optimizer

Bayram Bayramli · Junhwa Hur · Hongtao Lu

Received: 29 May 2022 / Accepted: 17 May 2023 / Published online: 24 June 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract

A popular approach to estimate scene flow is to utilize point cloud data from various Lidar scans. However, there is little attention to learning 3D motion from camera images. Learning scene flow from a monocular camera remains a challenging task due to its ill-posedness as well as lack of annotated data. Self-supervised methods demonstrate learning scene flow estimation from unlabeled data, yet their accuracy lags behind (semi-)supervised methods. In this paper, we introduce a self-supervised monocular scene flow method that substantially improves the accuracy over the previous approaches. Based on RAFT, a state-of-the-art optical flow model, we design a new decoder to iteratively update 3D motion fields and disparity maps simultaneously. Furthermore, we propose an enhanced upsampling layer and a disparity initialization technique, which overall further improves accuracy up to 7.2%. Our method achieves state-of-the-art accuracy among all self-supervised monocular scene flow methods, improving accuracy by 34.2%. Our fine-tuned model outperforms the best previous semi-supervised method with 228 times faster runtime. Code will be publicly available to ensure reproducibility.

Keywords Scene flow · Disparity estimation · Self-supervised learning · Autonomous driving

1 Introduction

For safe navigation, autonomous systems (e.g., self-driving cars, robots) need to comprehend the 3D motion of dynamic scenes, including multiple moving objects. Scene flow estimation is the task of estimating the 3D motion of points in the world coordinate system (Vedula et al., 2005). Recent state-of-the-art methods (Wang et al., 2020; Teed & Deng, 2021) estimate scene flow from 3D points or RGB-D cameras in a supervised manner. However, supervised learning requires a large amount of annotated training datasets. It is expensive to obtain such datasets because labeling the motion of every point in the real world is highly laborious. Therefore, the supervised learning-based method primarily used large-scale synthetic datasets for training and real-world dataset for fine-tuning only. Hence, such expensive requirements restrict the effectiveness of supervised methods in a real-world setting. Moreover, using different types of input sensors (e.g., RGB-D videos, stereo videos (Teed & Deng, 2021), and point clouds (Wang et al., 2020)) brings further limitations for each sensor configuration (e.g., indoor usage only, calibration for a stereo rig, costly LiDAR).

To overcome the above-mentioned limitations, we focus on scene flow estimation using a monocular camera in a self-supervised fashion, which is a simple, low-cost setup that resolves those limitations. Scene flow estimation from monocular images is a challenging, ill-posed task due to depth ambiguity. Several self-supervised multi-task learning approaches (Yang et al., 2018; Luo et al., 2020; Zou et al., 2018) demonstrate the possibility of learning scene flow from only monocular video frames, yet their complicated training pipelines, as well as architectures, limit the practicality of their methods. Recently, a couple of methods (Hur & Roth, 2020, 2021) propose more effective approaches to estimate monocular scene flow: directly estimating scene flow from a single network, requiring simple training schedules, and substantially improving the accuracy. Nevertheless, their
Recent state-of-the-art self-supervised depth estimation methods are learning-based. Godard et al. (2017) proposed to learn monocular depth estimation from stereo pairs. Later, Godard et al. (2019) proposed an improved proxy loss for handling occlusion and moving objects. Watson et al. (2019) introduced Depth Hints using stereo pairs as complementary supervision to boost the overall accuracy of existing self-supervised solutions. Ramamonjisoa et al. (2021) used wavelet representation with deep learning to learn monocular depth maps. The lack of ground truth depth maps led to the investigation of knowledge distillation techniques for self-supervised depth estimation. Peng et al. (2021) proposed a data augmentation and self-teaching loss based on the distillation method to enhance the performance of the monocular depth estimation task. In contrast to those methods that rely on complicated strategies, we demonstrate very competitive results by using a considerably simpler network without additional supervision, such as pseudo labels from knowledge distillation or stereo images, or complicated training procedures.

2.2 Scene Flow

Some approaches (Liu et al., 2019b; Wu et al., 2020; Gu et al., 2019) estimate scene flow from 3D Lidar scans using deep learning techniques. Liu et al. (2019b) learns scene flow based on PointNet++ (Qi et al., 2017). Gu et al. (2019) introduces a sparse convolution network for scene flow, and Wu et al. (2020) proposes coarse-to-fine scene flow estimation network based on PWCNet (Sun et al., 2018). Compared to these methods, we do not learn scene flow directly from 3D point clouds but we attempt to learn scene flow from lightweight, inexpensive cameras.

Recent approaches to self-supervised monocular scene flow show that a 3D motion field can be directly learned from a CNN module without requiring joint training of depth, optical flow, and ego-motion. First, Hur and Roth (2020) proposed to learn scene flow using a single joint decoder with proxy loss functions based on PWC-Net (Sun et al., 2018) architecture. Hur and Roth (2020) was the first method to learn 3D motion vectors directly from the CNN module using only a sequence of images. Later, Hur and Roth (2021) proposed a multi-frame monocular scene flow network to improve the previous two-view monocular scene flow model Hur and Roth (2020) by proposing a split decoder and multi-view training strategy using convolutional LSTM. Despite the simplicity of the above works compared to joint multi-task learning methods, the accuracy of these works is still lower. Inspired by Hur and Roth (2020) two-view approach, we propose RAFT-MSF and substantially improve on previous two-view and multi-view models by only using two-view images as input. Additional to self-supervised methods, a semi-supervised method (Brickwedde et al., 2019) estimates scene flow from a monocular camera using a probabilistic optimization framework, and Schuster et al. (2020) proposes...
Fig. 1 Overview of RAFT-MSF. Feature networks encode input images to feature vectors. Correlation volume is constructed by using encoded features. We initialize the 3D motion field to zeros everywhere, and the disparity is initialized using 2 convolutional layers. During each iteration, the update operator uses the current estimates of scene flow and disparity to index from the correlation pyramid. The output of the update operator is fed to scene flow and disparity heads. The scene flow can be projected to optical flow

a monocular combination of depth estimation and optical flow with interpolation and refinement techniques to estimate monocular scene flow. On the contrary, our fine-tuned monocular scene flow method with simpler network architecture outperforms the above semi-supervised works with faster run-time. Note that, the self-supervised learning technique does not involve labeled data, the model predicts intermediate disparity and scene flow through view synthesis. However, the semi-supervised methods still require small annotated data.

3 Method

Given two temporally consecutive images, $I_t$ and $I_{t+1}$, our model estimates a disparity map and a dense 3D motion field with respect to the camera. Inspired by Hur and Roth (2020) and Teed and Deng (2020), our model iteratively estimates a 3D point $P = (P_x, P_y, P_z)$ and scene flow $s = (s_x, s_y, s_z)$ for each pixel $p = (p_x, p_y)$ in the source frame, $I_t$. Assuming a pinhole camera model with known camera intrinsic, optical flow can be recovered by projecting scene flow and depth onto the image plane.

3.1 Network Architecture

Our network consists of three main components: i) a feature encoder, ii) a correlation pyramid, and iii) a GRU-based update operator with scene flow and depth head modules. Our model first constructs a correlation pyramid using features extracted from the encoder. Then, the GRU-based update operator takes correlation features as input and updates the 3D motion field and depth map iteratively. The correlation features are parsed by using intermediate estimates of scene flow and depth. Here, we use the depth ($d_t$) and disparity interchangeably; however, our model learns disparity in a hypothetical stereo setup following (Godard et al., 2019), the depth values can be obtained given the baseline and focal length of the camera. The overview of our RAFT-MSF is shown in Fig. 1.

3.1.1 Feature Encoder

Given the two input images, our model extracts two kinds of features: matching feature and context feature. For each image, the feature encoder $f_\theta$ extracts a 256-dimensional matching feature at $1/8$ resolution, which is then used for constructing a cost volume. The context encoder, the same architecture as in $f_\theta$ but with different trainable weights, extracts the context feature for the source image $I_t$. The context feature provides semantic and contextual information about the source image to the update operator.

3.1.2 Correlation Pyramid

We build a full 4D correlation volume that contains visual similarity between all possible matching pairs. Given the per-pixel matching feature for each image, $f_\theta(I_t), f_\theta(I_{t+1}) \in \mathbb{R}^{H\times W \times D}$, the correlation volume is computed as:

$$C_{ijkh}(I_t, I_{t+1}) = \langle f_\theta(I_t)_{ij}, f_\theta(I_{t+1})_{kh} \rangle \in \mathbb{R}^{H\times W \times H \times W},$$

by using a dot product between matching features. $\langle \rangle$ represents a dot product operation. $ij$ and $kh$ refers pixel index of $I_t$ and $I_{t+1}$, respectively.

Then, we construct a 4-level correlation pyramid $\{C_1, C_2, C_3, C_4\}$ by pooling the last two dimensions of the full correlation volume,

$$C_k \in \mathbb{R}^{H\times W \times H/2^k \times W/2^k}, k = 1, 2, 3, 4.$$
This multi-scale correlation pyramid provides information for both large and small displacements while maintaining high-resolution information.

3.1.3 Update Operator

Unlike RAFT that estimates optical flow, our update operator predicts a sequence of forward and backward scene flow \{s_1, ..., s_N\}, and disparity map of source and target frame \{d_1, ..., d_N\}. The bi-directional (Meister et al., 2018) estimation allows the utilization of occlusion cues. For simplicity, we only show the forward scene flow and the disparity of the source frame. At each iteration, the update operator residually updates the estimates,

\[ s_{n+1} = \Delta s + s_n, d_{n+1} = \Delta d + d_n. \]  

Equation (3).

Figure 2 illustrates an overview of the update operator. The update operator is based on a GRU and takes features from the correlation pyramid via an indexing operator, current estimates of scene flow and disparity, and a hidden state from the previous iteration step, then outputs the residual update \( \Delta s \) and \( \Delta d \) for scene flow and disparity, respectively. It additionally outputs an upsampling mask from the mask head. The input hidden state of the update operator is initialized by the context encoder with the \text{tanh} \ function as activation.

**Initialization** We initialize scene flow to zeros (i.e., \( s_0 = 0 \)) but disparities are initialized with convolutional layers. We found that initializing disparities with two convolutional layers yields better accuracy for both disparity and scene flow compared to initializing with zeros. Given the feature map of the source image, we initialize our disparity:

\[ d_0 = g_w(f_0(I_s)). \]  

Equation (4).

with two convolutional layers \( g_w() \).

**Inputs** At each iteration step, we retrieve correlation features for the residual updates from the pre-computed 4D correlation pyramid \( C_k \). Given current estimations scene flow \( s_n \), depth \( \hat{d}_n \), and the camera intrinsic \( K \), we first calculate the corresponding pixel \( \hat{p}' \) for each pixel \( p \),

\[ \hat{p}' = K(d_t(p) \cdot K^{-1} p + s_{t\rightarrow t+1}(p)). \]  

Equation (5).

and retrieve correlation features of a set of pixels \( N_{p'} \) neighboring the corresponding pixel \( \hat{p}' \), within a range of \([-r, r] \),

\[ N_{p'} = \{(p_x + d_{p_x}, p_y + d_{p_y}) | d_{p_x}, d_{p_y} \in [-r, ..., r]\}. \]  

Equation (6).

Of course, optical flow can be naturally obtained by \( F = \hat{p}' - p \). Given the current estimates and the retrieved correlation feature, we pass them through the motion feature encoder to get a motion feature. Then the GRU takes a concatenation of the motion features and the context features (from the context encoder) as an input.

**Predictions** The crucial part of the update operator is GRU units with convolution layers instead of fully connected layers. The encoded motion features are iteratively decoded to predict residual scene flow and disparity. As illustrated in Fig. 2, the output of the GRU is fed to scene flow and disparity heads to estimate corresponding outputs. We employ three convolutional layers to predict scene flow and disparity residuals. The final prediction is the sum of all residual outputs with initial values (ref. Eq. 3).

**Upsampling Module** The predicted disparity and scene flow estimates are at 1/8 of the input resolution. We employ an optimization-based upsampling module using convex upsampling (Teed & Deng, 2020) to upsample the coarse resolution estimates to the input image resolution. The upsampling module takes the feature map from the output of the GRU and processes it using two convolutional (Fig. 2, mask head) layers to generate a convex mask. However, we found that in the self-supervised learning setup, direct usage of the convex upsampling operator results in checkerboard artifacts on predicted outputs at later stages, and the model fails to converge. To address it, we detach the gradients of the hidden state outputted by the GRU in each upsampling operation. In Fig. 2, the arrow \( \not\rightarrow \) represents detaching the output during back-propagation.
of GRU for the mask head at each iteration. In the ablation study in Fig. 1, we demonstrate that the gradient detaching technique further improves the accuracy. Moreover, in Fig. 4, using detaching technique results in a smooth and artifact-free depth map, while without detaching, we get distorted and checkerboard artifacts on the depth map.

### 3.2 Self-Supervised Learning

We follow the self-supervised loss function proposed by (Hur & Roth, 2020, 2021), which uses view synthesis as a proxy task for learning depth and scene flow in a self-supervised manner. We briefly describe the loss below. Given stereo image pairs, our model takes a pair of source and target frames \( \{ I^l_t, I^l_{t+1} \} \) as an input and outputs disparity map \( d^l_t, d^l_{t+1} \) for each frame and bi-directional scene flow \( (s^l_{fw}, s^l_{bw}) \). Bi-directional scene flow can be achieved by simply altering the temporal order of the input images. At training time, the temporal right images \( (I^r_t, I^r_{t+1}) \) are only used to guide the loss functions, but during the test time, our model only accepts monocular temporal images. Given depth map and scene flow estimates, we synthesize the reconstructed left source image \( \hat{I}^l_t \) from the left target frame \( I^l_{t+1} \) and penalize the photometric differences between the source image \( I^l_t \) and the reconstructed source image \( \hat{I}^l_t \), so that the network outputs a correct combination of depth and scene flow.

Our disparity loss is based on Godard et al. (2017). The loss calculates a difference between the left image \( I^l_t \), and the synthesized left image \( \hat{I}^l_t \) that is obtained by the estimated disparity map \( d^l_t \) and the right image \( I^r_t \) using bilinear interpolation. It calculates the loss only for non-occluded pixels. The photometric loss is a weighted sum of an 1st-order edge-aware smoothness loss and the structural similarity index (SSIM) (Wang et al., 2004):

\[
L_{d, ph} = \frac{\sum_p (1 - O^l_{1, disp}(p)) \cdot \rho(I^l_t(p), \hat{I}^l_{t}(p))}{\sum_q (1 - O^l_{t, disp}(q))},
\]

where \( O^l_{1, disp} \) is the binary mask for disparity occlusion. Following Godard et al. (2017), we use the right image \( I^r_t \) with its corresponding disparity map to obtain the occlusion mask. \( \rho() \) is function of SSIM:

\[
\rho(a, b) = a \frac{1 - SSIM(a, b)}{2} + (1 - \alpha) \| a - b \|_1,
\]

where \( \alpha \) is set to 0.85.

As a common standard, 2nd-order edge-aware smoothness loss is used to have locally smooth disparity estimates,

\[
L_{d, sm} = \frac{1}{N} \sum_p \sum_{i \in \{x, y\}} \| \nabla^2 d^l_t(p) \| \cdot \epsilon^\beta \| \nabla I^l_t(p) \|,
\]

where \( N \) is the number of pixels and \( \beta = 10 \). Given \( L_{d, ph} \) and \( L_{d, sm} \), we define the total disparity loss function as:

\[
L_d = L_{d, ph} + \lambda_{d, sm} L_{d, sm},
\]

where \( \lambda_{d, sm} \) is fixed to 0.1.

Similarly, the scene flow loss is made up of a photometric loss \( L_{sf, ph} \), scene flow smoothness loss \( L_{sf, sm} \), and a 3D point reconstruction loss \( L_{sf, pt} \).

Given the left source image \( I^l_t \), reconstructed left image \( \hat{I}^l_{t, sf} \) is obtained by employing the disparity map \( d^l_t \), scene flow \( s^l_{fw} \), and the target image \( I^l_{t+1} \). The corresponding \( p' \) in \( I^l_t \) is calculated by projecting the pixel \( p \) in \( I^l_t \) back to 3D coordinate systems using camera parameters \( K \), depth prediction \( \hat{d}^l_t(p) \), and scene flow \( s^l_{fw} \). The corresponding pixel \( p' \) is finally obtained by projecting the translated points into the image coordinate system using Eq. 5.

The scene flow photometric loss is used to penalize the difference between \( \hat{I}^l_t \) and \( I^l_{t, sf} \) by using the similar occlusion-aware loss as in Eq. 7.

\[
L_{sf, ph} = \frac{\sum_p (1 - O^l_{1, sf}(p)) \cdot \rho(I^l_t(p), \hat{I}^l_{t, sf}(p))}{\sum_q (1 - O^l_{t, sf}(q))},
\]

scene flow occlusion mask \( O^l_{1, sf} \) is calculated by using backward scene flow \( s^l_{bw} \) estimates.

The 3D point reconstruction loss minimizes the difference between two corresponding 3D points:

\[
L_{sf, pt} = \frac{\sum_p (1 - O^l_{1, sf}(p)) \cdot \| P'_t - P'_{t+1} \|_2}{\sum_q (1 - O^l_{t, sf}(q))},
\]

where \( P'_t, P'_{t+1} \) are the two corresponding 3D points,

\[
P'_t = \hat{d}^l_t(p) \cdot K^{-1} p + s^l_{fw}(p)
\]

\[
P'_{t+1} = \hat{d}^l_{t+1}(p) \cdot K^{-1} p'.
\]

Smoothness loss for scene flow is similar to the Eq. 9.

\[
L_{sf, sm} = \frac{1}{N} \sum_p \sum_{i \in \{x, y\}} \| \nabla^2 s^l_{fw}(p) \| \cdot \epsilon^\beta \| \nabla \hat{I}^l_t(p) \|,
\]

Based on the Eqs. 11, 12 and 15, the total scene flow loss is defined as

\[
L_{sf} = L_{sf, ph} + \lambda_{sf, pt} L_{sf, pt} + \lambda_{sf, sm} L_{sf, sm},
\]

with the weights being \( \lambda_{sf, pt} = 0.2 \), and \( \lambda_{sf, sm} = 200 \).

The overall objective function for self-supervised learning is a weighted sum of disparity Eq. 10 and scene flow Eq. 16.
losses,
\[ L_{\text{total}} = L_d + \lambda_{\text{sf}} L_{\text{sf}}, \]  
(17)

where the regularization parameter \( \lambda_{\text{sf}} \) is dynamically decided to balance the optimization of the join task of scene flow and disparity estimation.

Finally, given \( n \) intermediate predictions, we apply the self-supervised loss Eq. 17 on the entire sequence of predictions. We exponentially decay the weights of each loss in Eq. 17,
\[ L_{\text{sequence}} = \sum_{i=1}^{n} \gamma^{n-i} L_i, \]  
(18)

where \( n \) is the number of disparity and scene flow iterations, \( \gamma \) is the decay factor, and \( L_i \) is the loss at each \( i^{th} \) iteration step.

For brevity sake, we only show the loss functions are applied in one direction (from \( t^j \) to \( t^j+1 \)). During experiments, disparity loss Eq. 10 is applied both on \( d^j \) and \( d^{j+1} \). Similarly, scene flow loss Eq. 16 is applied on both forward and backward scene flow directions.

4 Experiments

For a fair comparison with the state-of-the-art methods, we follow the same training dataset (KITTI raw (Geiger et al., 2013)) and protocols from our direct previous work (Hur & Roth, 2020, 2021). For the main experiment and ablation study, we use KITTI split (Godard et al., 2017) consisting of 25801 training pairs and 1684 validation pairs. Then, we test our model on the KITTI Scene Flow Training, which contains scene flow and disparity ground truth for 200 image pairs. For monocular depth evaluation, we additionally use the Eigen Split (Eigen et al., 2014) that consists of 20120 training pairs and 1338 validation pairs. Note that there is no overlap between the training data and testing data. Following (Hur & Roth, 2020), we also perform fine-tuning of our monocular scene flow model.

4.1 Implementation Details

For training, we use Adam optimizer (Kingma & Ba, 2015) \( (\beta_1 = 0.9, \beta_2 = 0.999) \) and clip gradients to the range of \([-1, 1]\) with initial learning rate of 0.0001. The training takes about two days on 4 GPUs. We train our model for 200k iterations with the mini-batch size of 8. Our method does not require a complicated stage-wise pre-training; thus, it can be trained at once. Unless otherwise noted, we set the number of update iteration \( n \) in Eq. 18 to 10 and the weight decay factor

| Ablation study on our contributions: We first propose a monocular scene flow RAFT baseline (RAFT baseline) which already outperforms the previous state of the arts |
|---------------------------------|--------|--------|--------|--------|
| Grad. detach | Disp. init | D1-all | D2-all | Fl-all | SF-all |
| RAFT baseline | 20.06 | 26.71 | **16.59** | 34.01 |
| ✓ | 20.44 | **23.62** | 17.61 | 31.85 |
| ✓ ✓ | **18.34** | 23.65 | 17.51 | **30.97** |

Bold indicates the best performance and italics shows the second best performance.

The disparity of RAFT-baseline is initialized by zeros. The gradient detaching on the upsampling module and disparity initialization with convolutional layers further improve the accuracy.

| Ablation results using different upsampling module: Quantitative results of using bilinear upsampling (Bilinear Ups) and learned upsampling (Learned Ups) module with the gradient detaching technique |
|---------------------------------|--------|--------|--------|--------|
| Type of upsampling module | D1-all | D2-all | Fl-all | SF-all |
| Bilinear ups | 21.02 | 26.72 | 21.71 | 35.73 |
| Learned ups | **18.34** | **23.65** | **17.51** | **30.97** |

Bold shows the best performance.

4.2 Experiments

For a fair comparison with the state-of-the-art methods, we follow the same training dataset (KITTI raw (Geiger et al., 2013)) and protocols from our direct previous work (Hur & Roth, 2020, 2021). For the main experiment and ablation study, we use KITTI split (Godard et al., 2017) consisting of 25801 training pairs and 1684 validation pairs. Then, we test our model on the KITTI Scene Flow Training, which contains scene flow and disparity ground truth for 200 image pairs. For monocular depth evaluation, we additionally use the Eigen Split (Eigen et al., 2014) that consists of 20120 training pairs and 1338 validation pairs. Note that there is no overlap between the training data and testing data. Following (Hur & Roth, 2020), we also perform fine-tuning of our monocular scene flow model.

4.1 Implementation Details

For training, we use Adam optimizer (Kingma & Ba, 2015) \( (\beta_1 = 0.9, \beta_2 = 0.999) \) and clip gradients to the range of \([-1, 1]\) with initial learning rate of 0.0001. The training takes about two days on 4 GPUs. We train our model for 200k iterations with the mini-batch size of 8. Our method does not require a complicated stage-wise pre-training; thus, it can be trained at once. Unless otherwise noted, we set the number of update iteration \( n \) in Eq. 18 to 10 and the weight decay factor

| Ablation study on disparity initialization: Initializing disparity with convolutional layers outperforms initializing with other fixed initialization methods such as ones, zeros, etc |
|---------------------------------|--------|--------|--------|--------|
| Type of disp init | D1-all | D2-all | Fl-all | SF-all |
| Ones | 44.59 | 49.27 | 19.97 | 57.74 |
| 0.01 | 20.49 | 23.65 | 17.63 | 31.88 |
| 0.001 | 20.44 | 23.63 | 17.61 | 31.86 |
| Zero | 20.44 | **23.62** | 17.61 | **31.85** |
| 2 conv layers | **18.34** | 23.65 | **17.51** | **30.97** |

Bold indicates the best performance and italics shows the second best performance.

We choose to use two layers which give better accuracy without much computational cost.
Table 4  Ablation study on the impact of the number of iterations used in update operator.

| # iters | D1-all | D2-all | Fl-all | SF-all | Runtime |
|---------|--------|--------|--------|--------|---------|
| 1       | 44.56  | 48.22  | 35.33  | 65.37  | 0.03 s  |
| 2       | 37.54  | 42.99  | 25.75  | 57.76  | 0.06 s  |
| 4       | 26.66  | 30.08  | 19.69  | 41.13  | 0.09 s  |
| 6       | 21.85  | 24.32  | 17.90  | 33.21  | 0.11 s  |
| 8       | 19.51  | 23.49  | 17.53  | 31.24  | 0.14 s  |
| 10      | 18.34  | 23.65  | 17.51  | 30.97  | 0.18 s  |

Bold indicates the best performance and italics shows the second best performance. The increase in iteration steps consistently improves the accuracy γ to 0.8 for all the experiments conducted in our manuscript. Weights of all convolutional layers, including the ones used for disparity initialization, are initialized by Kaiming uniform initialization. Note that we train all components of our network at once from scratch, there is no stage-wise training implemented in our work as compared to other multi-task learning methods.

4.2 Ablation Study

To validate our contributions, we conduct ablation studies on our model, RAFT-MSF. We use the scene flow metrics for evaluation, which is defined as the outlier rate in %. A pixel is considered an outlier if it exceeds a threshold of 3 pixels and 5% w.r.t. the ground truth labels. The scene flow outlier rate (SF-all) is obtained after calculating the outlier rate of the disparity (D1-all), disparity change (D2-all), and optical flow (Fl-all).

Table 1 demonstrates the ablation study of our major contributions. Given the vanilla RAFT optical flow backbone, we demonstrate a self-supervised monocular scene flow pipeline on RAFT (RAFT baseline) in the first row. This first contribution already outperforms the previous best two-frame self-supervised method (Hur & Roth, 2020) by a large margin (34.01 % vs. 47.05 %) in scene flow error metric (SF-all).

Our contributions using gradient detach for convex upsampling layers and disparity initialization further improve the accuracy up to 7.2% from our monocular scene flow proposal on RAFT. Especially, the gradient detaching strategy effectively removes the checkerboard artifacts appearing during the upsampling process and outputs clear object boundaries. Specifically, in Fig. 4 (i.e., column (b)), the estimated disparities and disparity errors are checkered and distorted; however, in Fig. 4 (i.e., column (c)), the visualized results are smoother and checker free.

We compare our upsampling module with gradient detach- ing technique to bilinear upsampling in Table 2. Additionally,
Table 5  Monocular depth comparison: We show a superior accuracy on KITTI Split compared to published multi-task methods.

| Split       | Method                        | Abs Rel ↓ | Sq Rel ↓ | RMSE ↓ | RMSE log ↓ | δ < 1.25 ↑ | δ < 1.25² ↑ | δ < 1.25³ ↑ |
|-------------|-------------------------------|-----------|----------|--------|-------------|-------------|-------------|-------------|
| KITTI       | EPC (Yang et al., 2018)       | 0.109     | 1.004    | 6.232  | 0.203        | 0.853       | 0.937       | 0.975       |
|             | Liu et al. (2019a)            | 0.108     | 1.020    | 5.528  | 0.195        | 0.863       | 0.948       | 0.980       |
|             | Self-Mono-SF (Hur & Roth, 2020)| 0.106    | 0.888    | 4.853  | 0.175        | 0.879       | 0.965       | 0.987       |
|             | RAFT-MSF (ours)               | **0.082** | **0.726** | **4.165** | **0.148** | **0.921** | **0.971** | 0.986       |
| Eigen       | CC (Ranjan et al., 2019)      | 0.155     | 1.296    | 5.857  | 0.233        | 0.793       | 0.931       | 0.973       |
|             | GLNet (Chen et al., 2019)     | 0.135     | 1.070    | 5.230  | 0.210        | 0.841       | 0.948       | 0.980       |
|             | EPC (Yang et al., 2018)       | 0.127     | 1.239    | 6.247  | 0.214        | 0.847       | 0.926       | 0.969       |
|             | EPC++ (Luo et al., 2020)      | 0.127     | 0.936    | 5.008  | 0.209        | 0.841       | 0.946       | 0.979       |
|             | Self-Mono-SF (Hur & Roth, 2020)| 0.125    | 0.978    | 4.877  | 0.208        | 0.851       | 0.950       | 0.978       |
|             | Monodepth2 (Godard et al., 2019)| 0.105 | 0.822    | 4.692  | 0.199        | 0.876       | 0.954       | 0.977       |
|             | Depth Hints (Watson et al., 2019)| 0.099 | 0.723    | 4.445  | 0.187        | 0.886       | 0.961       | 0.982       |
|             | WaveletMonodepth (Ramamonjisoa et al., 2021)| 0.102 | 0.739    | 4.452  | 0.188        | 0.883       | 0.960       | 0.981       |
|             | R-MSFM (Zhou et al., 2021)    | 0.112     | 0.753    | 4.530  | 0.189        | 0.881       | 0.961       | 0.982       |
|             | EPCDept (Peng et al., 2021)   | **0.093** | **0.671** | **4.297** | **0.178** | **0.899** | **0.965** | **0.983**   |
|             | RAFT-MSF (ours)               | **0.093** | 0.781    | 4.321  | 0.186        | **0.901**   | 0.960       | 0.981       |

Bold indicates the best performance and italics shows the second best performance.

In Eigen split, our method outperforms multi-task methods and delivers competitive results with single monocular depth estimation tasks. (↓: lower is better, and ↑: higher is better)
Table 6  Monocular scene flow evaluation on KITTI 2015 Scene Flow Training: Our self-supervised method significantly outperforms both multi-task CNN methods and recently published monocular scene flow methods on all the metrics. “>” means that the scene flow outlier rate of the method is at least bigger than the number stated.

| Method | D1-all | D2-all | Fl-all | SF-all |
|--------|--------|--------|--------|--------|
| DF-Net (Zou et al., 2018) | 46.50 | 61.54 | 27.47 | 73.30 |
| GeoNet (Yin & Shi, 2018) | 49.54 | 58.17 | 37.83 | 71.32 |
| EPC++ (Luo et al., 2020) | 23.84 | 34.86 | 19.64 | (>60.32) |
| Self-Mono-SF (Hur & Roth, 2020) | 31.25 | 34.86 | 23.49 | 47.05 |
| Multi-Mono-SF (Hur & Roth, 2021) | 27.33 | 30.44 | 18.92 | 39.82 |
| RAFT-MSF (ours) | **18.34** | **23.65** | **17.51** | **30.97** |

Bold indicates the best performance and italics shows the second best performance.

Fig. 5 Qualitative comparison of our model (RAFT-MSF) with Self-Mono-SF (Hur & Roth, 2020): top row shows two temporal input images ($I_t, I_{t+1}$), second and third rows compare the depth and depth error maps of our method (RAFT-MSF) and Self-Mono-SF (Hur & Roth, 2020), and last two rows compare scene flow and scene flow error maps of the same methods, respectively.

we show the qualitative comparison of these two approaches in Fig. 3. It can be observed that the learned upsampling module achieves better performance at all the scene flow metrics and produces a sharper depth map, particularly around motion boundaries.

In Table 3, we further investigate the different types of disparity initialization. Comparing to simply initializing with zero, 0,01, 0.001, using convolutional layers for initialization yields better accuracy. We choose to use two convolutional layers for better performance with the minimal burden of computational cost. We also tried different initialization techniques, such as using ones everywhere, but with these training settings, the model is unable to learn disparity. By using convolutional layers for initialization, we hope that the model learns initial disparity values that help faster convergence and yield better accuracy. We have also tried the same initialization approaches for scene flow, but it did not bring obvious benefits for scene flow. It may be due to the scene flow having a high dimensionality and thus is difficult to learn. In the end, in order to favor a simpler model, we initialized the scene flow with zeros.

We have evaluated the proposed method using various update iteration numbers in Table 4. It can be seen that our method quickly converges and outperforms Hur & Roth (2020) after 4 updates. Although an increasing number of update iterations brings better results, it also affects the runtime of the proposed method. Therefore, we have opted for 10 update iterations as an optimal trade-off between the performance and inference time.
Table 7  KITTI 2015 Scene Flow Test: We compare our RAFT-MSF with fine-tuned or semi-supervised monocular methods (top-rows) and self-supervised methods (bottom rows). We report the outlier rate on the background (bg), foreground (fg), and all pixels (all).

| Method                        | D1 bg | D1 fg | D1 all | D2 bg | D2 fg | D2 all | Fl bg | Fl fg | Fl all | SF bg | SF fg | SF all |
|-------------------------------|-------|-------|--------|-------|-------|--------|-------|-------|--------|-------|-------|--------|
| Mono-SF (Brickwedde et al., 2019) | 14.21 | 26.94 | 16.32  | 16.89 | 33.07 | 19.50  | 11.40 | 19.64 | 12.77  | **19.79** | 39.57 | 23.08  |
| Mono expansion (Yang & Ramanan, 2020) | -     | -     | 25.36  | -     | -     | 28.34  | -     | -     | 6.30   | -     | -     | 30.96  |
| MonoComb (Schuster et al., 2020) | 17.89 | **21.16** | 18.44  | 22.34 | **25.85** | 22.93  | **5.84** | **8.67** | 6.31   | 27.06 | **33.55** | 28.14  |
| Self-Mono-SF-ft (Hur & Roth, 2020) | 20.72 | 29.41 | 22.16  | 23.83 | 32.29 | 25.24  | 15.51 | 17.96 | 15.91  | 31.51 | 45.77 | 33.88  |
| Multi-Mono-SF-ft (Hur & Roth, 2021) | 21.60 | 28.22 | 22.71  | 25.47 | 31.72 | 26.51  | 12.41 | 18.20 | 13.37  | 31.18 | 42.68 | 33.09  |
| RAFT-MSF-ft (ours)             | **14.19** | 24.79 | **15.95** | **16.51** | 30.07 | **18.77** | 8.35  | **11.02** | 8.80   | **19.79** | 36.07 | **22.50** |
| Self-Mono-SF (Hur & Roth, 2020) | 31.22 | 48.04 | 34.02  | 34.89 | 43.59 | 36.34  | 23.26 | 24.93 | 23.54  | 46.68 | 63.82 | 49.54  |
| Multi-Mono-SF (Hur & Roth, 2021) | 27.48 | 47.30 | 30.78  | 32.39 | 44.56 | 34.41  | 18.13 | 26.59 | 19.54  | 40.29 | 62.78 | 44.04  |
| RAFT-MSF (ours)                | **18.10** | **36.82** | **21.21** | **24.98** | **40.19** | **27.51** | **17.98** | **20.33** | **18.37** | **31.59** | **51.97** | **34.98** |

Bold indicates the best performance and italics shows the second best performance.
Fig. 6 Qualitative comparison of our fine-tuned model with state-of-the-art methods (Hur & Roth, 2020; Brickwedde et al., 2019) on the KITTI 2015 Scene Flow public benchmark. In the first row the source and target frames. From the second to the last row, we give a qualitative comparison with a Self-Mono-Sf (Hur & Roth, 2020), b (Brickwedde et al., 2019) and c Ours: the disparity map of the source image (D1) with its error map (D1 Error), disparity estimation at the target image mapped into the source frame (D2) along with its error map (D2 Error), optical flow (F1) with its error map (F1 Error), and the scene flow error map (SF1 Error). The color map of the respective error maps are depicted in the legend. Blue indicates a lower error, and red indicates a higher error. The outlier rates are overlayed on each error map (Color figure online).

To better understand how each contribution affects the predicted scene flow and disparity estimates, Fig. 4 provides qualitative examples of disparity, optical flow, and scene flow estimation with their corresponding error maps. According to Fig. 4, the qualitative results for each finding: (a) our full model including applying gradient detaching technique and disparity initialization with convolutional layers; (b) without performing gradient detach for the upsampling module; (c) with disparity initialization to zeros instead of using convolutional layers. The configurations (a-c) are trained in a self-supervised manner using the KITTI Split and evaluated on the KITTI Scene Flow Training set.

Without the proposed disparity initialization (i.e., column (c) in Fig. 4), the network outputs inaccurate disparity estimation for both source (D1) and target frame (D2), especially around the car bodies and ground area, which contributes to incorrect scene flow error results (SF1) in the end. Moreover, without detaching the gradients of the GRU output for
the upsampling module (i.e., column (b) in Fig. 4) causes checkerboard artifacts and incorrect estimates of disparity and scene flow which still leads to undesirable overall results. However, applying disparity initialization with convolutional layers and detaching the gradients of the GRU output (i.e., column (a) in Fig. 4) yield impressive qualitative results where the estimated disparities and flows are smoothed and artifact-free, especially errors around the car bodies and on the ground are reduced significantly which in the end results in much better scene flow.

4.3 Monocular Depth

Our method also demonstrates state-of-the-art monocular depth accuracy. We evaluate our RAFT-MSF on KITTI and Eigen splits and compare them with the state-of-the-art monocular depth estimation approaches in Table 5. During training and evaluation, we set the depth estimation to a fixed depth range of a minimum 0 m and a maximum 80 m. We compare the accuracy of monocular depth estimation of RAFT-MSF with recent state-of-the-art methods by using five broadly used evaluation metrics introduced in Eigen et al. (2014): Abs Rel, Sq Rel, RMSE, RMSE log, and threshold accuracy.

Our method surpasses previously published methods on KITTI split for monocular depth estimation. On Eigen split, our RAFT-MSF shows better accuracy compared to Zhou et al. (2021) a recent monocular depth estimation method that also uses an iterative module to update the inverse depth via embedding a multi-scale feature modulation. Moreover, despite having a simpler and trick-free network, we can obtain competitive accuracy with Watson et al. (2019) that leverages generated depth maps from the classical semi-global-matching (SGM) as proxy labels for supervision and with Peng et al. (2021) which employs self-distillation and extensive data augmentation.

4.4 Monocular Scene Flow

We compare our full model (RAFT-MSF) on the KITTI Scene Flow Training and Test benchmark. Table 6 shows the accuracy comparison with state-of-the-art self-supervised monocular scene flow methods on the KITTI Scene Flow Training set. Our method achieves the best scene flow accuracy among self-supervised multi-task CNN methods and self-supervised monocular scene flow methods by a large margin. Especially, our model outperforms, in SF-all metric, the best two-frame self-supervised method (Hur & Roth, 2020) by 34.17% and even multi-frame self-supervised method (Hur & Roth, 2021) by 22.23%. Qualitative results in Fig. 5 show the superiority of our method over Hur and Roth 2020, improving the disparity and scene flow estimation on the planar road surface and object boundaries.

On KITTI Scene Flow 2015 Test benchmark in Table 7, our self-supervised model consistently outperforms the state of the arts, reducing the scene flow error by 29.39% on two-view (Hur & Roth, 2020), and 20.57% on multi-view method (Hur & Roth, 2021), respectively. We also fine-tune our self-supervised model using 200 pairs of ground truth in a semi-supervised manner and test on KITTI Scene Flow 2015 benchmark. Our fine-tuned model (RAFT-MSF-ft) also achieves the best scene flow accuracy among all published semi-supervised methods, especially outperforms the best monocular method (Brickwedde et al., 2019) with 228 times faster runtime (0.18 (s) vs. 41 (s)). Although in Fig. 6 we use a much simpler network and fewer annotated images (200) in our fine-tuned model, we qualitatively and quantitatively (Table 7) outperform semi-supervised monocular scene flow method (Brickwedde et al., 2019) that exploits more than 20 000 pseudo ground truth depth labels to train their model.

5 Analysis

RAFT-MSF achieves superior self-supervised monocular scene flow estimation results. However, our method has two limitations. First, it requires stereo images for training. To keep the pipeline as simple as possible, we do not explicitly model ego-motion. Our future work considers integrating ego-motion into our pipeline without complicating the overall architecture. Another limitation is the assumption of known camera intrinsic parameters. The camera parameters are sensor dependent and are not usually included in the dataset, which might limit the generalization of our method to other datasets with different intrinsic parameters. In the future, we plan to address this limitation by introducing a self-calibration mechanism into our model.

6 Conclusion

In this paper, we proposed a new self-supervised monocular scene flow method that significantly outperforms previous methods. Our RAFT-MSF is based on RAFT and iteratively updates scene flow and disparity map using a GRU update operator. Our contributions on disparity initialization with convolutional layers and gradient detaching strategy for upsampling layers further improve the accuracy. Our fine-tuned version outperforms the best semi-supervised method while demonstrating 228 times faster runtime. We hope our promising result, which demonstrates the significant accuracy boost, encourages active follow-up research on monocular scene flow estimation.
Acknowledgements This paper is supported by NSFC, China (No. 62176155) and Shanghai Municipal Science and Technology Major Project, China (2021SHZDZX0102).

Data Availability The datasets generated during and/or analyzed during the current study are available in the The KITTI Vision Benchmark Suite, https://www.cvlibs.net/datasets/kitti/

References

Basha, T., Moses, Y., & Kiryati, N. (2010). Multi-view scene flow estimation: A view centered variational approach. In CVPR.

Brickwedde, F., Abraham, S., & Mester, R. (2019). Mono-SF: Multi-view geometry meets single-view depth for monocular scene flow estimation of dynamic traffic scenes. In ICCV, pp. 2780–2790.

Chen, Y., Schmid, C., & Sminchisescu, C. (2019). Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In ICCV, pp. 7062–7071.

Eigen, D., Puhrsch, C., & Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In NIPS.

Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. IJRR, 32, 1231–1237.

Godard, C., Aodha, O. M., & Brostow, G. J. (2017). Unsupervised monocular depth estimation with left-right consistency. In CVPR, pp. 6602–6611.

Gu, X., Wang, Y., Wu, C., Lee, Y. J., & Wang, P. (2019). HPLFlowNet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds. In CVPR, pp. 3249–3258.

Hur, J., & Roth, S. (2020). Self-supervised monocular scene flow estimation. In CVPR, pp. 7394–7403.

Hur, J., & Roth, S. (2021). Self-supervised multi-frame monocular scene flow. In CVPR, pp. 2683–2693.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In ICLR.

Liu, L., Zhai, G., Ye, W., & Liu, Y. (2019a). Unsupervised learning of scene flow estimation fusing with local rigidity. In IJCAI.

Liu, X., Qi, C., & Guibas, L. J. (2019b). FlowNet3D: Learning scene flow in 3D point clouds. CVPR, pp. 529–537.

Luo, C., Yang, Z., Wang, P., Wang, Y., Xu, W., Nevatia, R., & Yuille, A. (2020). Every pixel counts++: Joint learning of geometry and motion with 3D holistic understanding. IEEE TPAMI, 42, 2624–2641.

Meister, S., Hur, J., & Roth, S. (2018). Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In AAAI.

Peng, R., Wang, R., Lai, Y., Tang, L., & Cai, Y. (2021). Excavating the potential capacity of self-supervised monocular depth estimation. In ICCV.

Qi, C., Yi, L., Su, H., & Guibas, L. J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In NIPS.

Ramamonjisoa, M., Firman, M., Watson, J., Lepetit, V., & Tumukhambetov, D. (2021). Single image depth prediction with wavelet decomposition. In CVPR, pp. 11084–11093.

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.