A NEW OBJECT SCENE FLOW ALGORITHM BASED ON SUPPORT POINTS SELECTION AND ROBUST MOVING OBJECT PROPOSAL

Zhengyang Sun[1], *Zongqing Lu*[1,*], *Jing-Hao Xue*[2], *Qingmin Liao*[1]

[1]Graduate School at Shenzhen, Tsinghua University, China
[2]Department of Statistical Science, University College London, UK
luzq@sz.tsinghua.edu.cn

ABSTRACT

Recent algorithms of object scene flow estimation suffer from low computational efficiency or unstable moving object proposals. To tackle these two problems simultaneously, in this paper we propose a new, efficient and robust algorithm for object scene flow estimation, through making two technical contributions. Firstly to improve the efficiency, we propose to select only a few pixels termed support points for matching cost calculation rather than using all pixels. The support points are defined as those pixels with high confidence in feature matching. Secondly to attain stable moving object proposals, we propose a motion magnitude-adaptive thresholding scheme for ego-motion outlier detection, after patch matching on CNN-extracted high quality features. These two contributions, though simple, ensure a remarkable improvement in both efficiency and accuracy from the original object scene flow method, as well as making the proposed algorithm a strong practicable alternative to much more sophisticated state-of-the-art competitors.

***Index Terms*—** Scene flow, Patch matching, Moving object proposal, Energy estimation.

1. INTRODUCTION

Scene flow is commonly defined as a flow field describing 3D motion at every point in the scene. Because 3D object motion is a fundamental input for many valuable high-level tasks such as autonomous driving, estimating scene flow is gaining increasing attentions. This paper focuses on estimating 3D scene flow of changes in both depth and optical flow.

A considerable number of interesting investigations [1–10] have been conducted in this area, facilitated by the availability of challenging benchmarks like KITTI [11]. The early work on estimating scene flow was often formulated as a variational problem; for example, Cech et al. [1] proposed a seed-growing method for stereoscopic scene flow. Recently, depth and optical flow were combined in the 3D motion solutions, specifically by following a strategy which first partitions the scene into planar segments with rigid movements. Vogel et al. [2–5] explored a model of piecewise rigid scene flow in the two-frame [4] and multi-frame [5] settings. They treated scenes as a collection of planar segments with rigid motion and jointly estimated superpixel segmentation, 3D geometries and rigid movements. The object scene flow method (OSF) [6] segmented scenes into planar superpixels with rigid movements. The OSF achieved impressive results but suffered from high computational costs. Hence many improvements of OSF have been proposed afterwards. Lv et al. [7] proposed a method using continuous optimization, faster than [6] but with lower accuracy. Neoral et al. [8] explored multi-frame temporal consistency to get a higher accuracy. Behl et al. [9] exploited object recognition to mitigate the problem of large displacement and local ambiguity. They used a convolutional neural network (CNN) to generate a high quality prior, and achieved the state of the art on the KITTI 2015 benchmark. Other methods such as Ren et al. [10] used semantic segmentation to improve scene flow prediction. However, most of these methods still suffer from an unsatisfactory balance between computational efficiency and estimation accuracy, the latter often affected by unstable moving object proposals.

This paper aims to enhance both computational efficiency and estimation accuracy of OSF, as well as providing a better balance between efficiency and accuracy than sophisticated state-of-the-art performers. As illustrated in the diagram of our method in Fig. 1, we propose a new, efficient and robust algorithm for object scene flow estimation (SPSOSF), through making two technical contributions. To attain stable moving object proposals, we propose a motion magnitude-adaptive thresholding scheme for ego-motion outlier detection (Section 2.2.2), after patch matching on CNN-extracted high quality features (Section 2.2.1). To improve the efficiency, we propose to select only a few pixels termed support points for matching cost calculation rather than using all pixels. The support points are the pixels with high confidence in feature matching (Section 2.3). These two contributions ensure a remarkable improvement in both efficiency and accuracy from the original OSF method (Section 3.1), as well as making the proposed algorithm a practicable alternative to more sophisticated state-of-the-art competitors (Section 3.2).

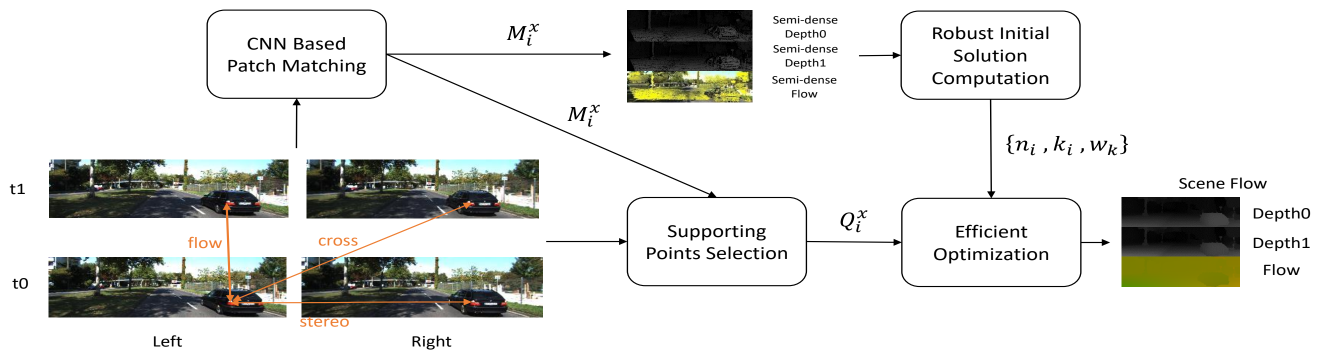


Fig. 1. Diagram of our method. The CNN-based patch matching is described in Section 2.2.1; the robust initialization including robust moving object proposals in Section 2.2.2; and the support points selection and the efficient optimization in Section 2.3.

2. OUR METHOD

2.1. Scene flow model

We follow the hypothesis that the 3D structure of a scene can be approximated by a set of piecewise planar superpixels [4], and that there are some finite number of foreground moving objects in the scene against the static background.

Let S denote the set of planar superpixels and O denote the set of moving objects. Each planar superpixel $s_i \in S$ corresponds to a region B_i in the image and a vector $\mathbf{v}_i = (\mathbf{n}_i, k_i)^T$, where $\mathbf{n}_i \in \mathbb{R}^3$ describes a 3D plane ($\mathbf{n}_i^T\mathbf{X} = 1$ for points $\mathbf{X} \in \mathbb{R}^3$ on the plane, and $k_i \in \{1, ..., |O|\}$ indexes the moving object with which the planar superpixel s_i is associated. Each object $o_k \in O$ corresponds to a variable $\mathbf{w}_k \in SE(3)$ composed of rotation $\mathbf{R}_k \in SO(3)$ and translation $\mathbf{t}_k \in \mathbb{R}^3$. The rotation and translation can describe the object's moving strategy and induce the scene flow.

With the input of left and right images of two consecutive frames, we aim to find the proper 3D geometry $\mathbf{n}_i$ and the object index k_i of each planar superpixel, and the rigid body motion $\mathbf{w}_k$ of each object, by minimizing the following two-term energy function of conditional random fields:

$$E(\mathbf{v}, \mathbf{w}) = \sum_{s_i \in S} \underbrace{\phi_i(\mathbf{v}_i, \mathbf{w})}_{\text{data}} + \sum_{s_i \sim s_j} \underbrace{\psi_{ij}(\mathbf{v}_i, \mathbf{v}_j)}_{\text{smoothness}}, \tag{1}$$

where $\mathbf{v} = \{\mathbf{v}_i | s_i \in S\}$, $\mathbf{w} = \{\mathbf{w}_k | o_k \in O\}$ and $s_i \sim s_j$ denotes the set of adjacent superpixels in S.

The data term in (1) is to ensure that the corresponding points across the four images should have similar features:

$$\phi_i(\mathbf{v}_i, \mathbf{w}) = \sum_{o_k \in O} [k_i = k] \cdot D_i(\mathbf{n}_i, \mathbf{w}_k), \tag{2}$$

where for each planar superpixel s_i, $[\cdot]$ denotes the Iverson bracket and $D_i(\mathbf{n}_i, \mathbf{w}_k)$ is its matching cost computed by using its plane geometry $\mathbf{n}_i$ and the rigid body motion $\mathbf{w}_k$:

$$D_i(\mathbf{n}_i, \mathbf{w}_k) = \sum_{x} \sum_{p \in B_i} C^x(\mathbf{p}, H(\mathbf{n}_i, \mathbf{w}_k) \cdot \mathbf{p}). \tag{3}$$

Here $x \in \{\text{stereo, flow, cross}\}$ identifies the three pairs of images (the red arrows) in Fig. 1: The reference image is the left image at t_0; the stereo pair stands for the reference image and the right image at t_0; the flow pair stands for the reference image and the left image at t_1; the cross pair stands for the reference image and the right image at t_1. In (3), $H(\mathbf{n}, \mathbf{w}) = \mathbf{K}(\mathbf{R}_x(\mathbf{w}) - \mathbf{t}_x(\mathbf{w}) \cdot \mathbf{n}^T)\mathbf{K}^{-1} \in \mathbb{R}^{3\times3}$ denotes the homography matrix between a pixel at location $\mathbf{p} \in \mathbb{R}^2$ in the reference image and at location $\mathbf{q} \in \mathbb{R}^2$ in the target image, in which $\mathbf{K} \in \mathbb{R}^{3\times3}$ is the camera intrinsic matrix and $[\mathbf{R}_x(\mathbf{w}), \mathbf{t}_x(\mathbf{w})] \in \mathbb{R}^{3\times4}$ maps a 3D point from the reference view to a 3D point in the target view. The details of calculating the matching cost $C^x(\mathbf{p}, \mathbf{q})$ will be given in Section 2.3.1.

The smoothness term in (1) is to enhance the coherence of adjacent superpixels. For our scene flow model, we use the smoothness term defined in [6] which is verified to be a good regularization for adjacent superpixels.

2.2. Robust moving object proposals

2.2.1. CNN-based patch matching

For optimizing the condition random fields in (1), we need to find a set of appropriate initial solutions $\{\mathbf{n}_i, k_i, \mathbf{w}_k\}$, where $\mathbf{n}_i$ can be estimated by the depths of pixels in the superpixel, k_i can be induced by a moving object proposal, and $\mathbf{w}_k$ can be estimated from optical flow and depth by least squares fitting. However, in many cases of sparse optical flow estimated by a method like OSF, some true moving object proposals are missing and such misses often appear together, which leads to high foreground scene flow errors. To overcome this issue, we first adopt CNN to extra quality features and use these features to attain more accurate semi-dense optical flow estimation, and then develop a motion magnitude-adaptive thresholding scheme to make robust moving object proposals.

We use CNN as a feature extractor to generate features robust to geometric and radiometric distortions. Our network has four convolution layers similar to the fast version of [12]. For the optical flow and depth feature extractions, we adopt the same network but with different training data and loss functions: for optical flow estimation, we follow the work in [13]; for depth computation, we follow the work in [12]. To train the network, we randomly select 30 images from the KITTI 2015 scene flow training dataset as the validation set and use the remaining images as the training data. We use the obtained semi-dense depth as the initial value of sps-stereo [14] which estimates the superpixel segmentation needed by other steps. Moreover, we build $M^x = \{M_i^x | s_i \in S\}$, a set of pixels with high-confidence matches, to be used later for robust matching cost calculation.

2.2.2. *Robust moving object proposal*

Moving objects are usually extracted as ego-motion outliers, where the endpoint error between our matching and the ego-motion correspondence is greater than a fixed threshold (denoted by $\|\mathbf{\Delta m}\|_2^2 > \gamma_1$; see below). This strategy, however, works poorly near image borders (Fig. 2(a)), where the endpoint error often has a large magnitude due to large ego-motion magnitude in both optical flow and depth change.

Therefore, we use a dynamic threshold adapted to the motion magnitude. That is, a matching is regarded as an outlier if the endpoint error $\Delta m = (\Delta u, \Delta v, \Delta d)$ between our matching and the ego-motion-induced correspondence satisfies

$$\|\mathbf{\Delta m}\|_2^2 > \max(\gamma_1 \|m_e\|_2, \gamma_2 \|m_e\|_2 / \overline{m}_e), \tag{4}$$

where $\mathbf{m_e} = (u_e, v_e, d_e)$ is obtained by [15]: the optical flow caused by the camera ego-motion is denoted by (u_e, v_e), and the consequent change in depth induced by the optical flow is denoted by d_e; $\gamma_1 = \sqrt{2}$ and $\gamma_2 = 12$ are constants selected by cross-validation; and $\overline{m}_e$ is the mean value of $\|\mathbf{m_e}\|_2$.

The aim of the factor $||\mathbf{m_e}||_2$ in (4) is to scale up threshold when the absolute magnitude of ego-motion $||\mathbf{m_e}||_2$ is too large, in which case a larger threshold is desired to suppress false alarms like that near image borders (Fig. 2(b)). The aim of the factor $\|\mathbf{m_e}\|_2/\overline{m}_e$ is to scale up threshold when the relative magnitude of ego-motion $||\mathbf{m_e}||_2$ is too large compared with the mean value around, in which case a larger threshold is desired to suppress noise. If a planar superpixel s_i satisfies (4), it will be proposed as a moving object in the scene.

2.3. Efficient optimization by selecting support points

After obtaining the robust initial solution, the max-product particle belief propagation (MP-PBP) [16, 17] with sequential tree-reweighted message passing (TRW-S) [18] can be used to solve the optimization problem (1).

However, MP-PBP is very computationally expensive (e.g., taking 50 minutes for one scene on a single i7 core running at 3.0 GHz), the time of which is mostly consumed by the calculation of matching cost (3). Therefore, we propose to use the support points only for the matching cost calculation.

Algorithm 1 Select support points

```
Input: $I, I^x, B_i, M_i^x, \eta$
Output: $Q_i^x$
 1: Pixels without matching: $N_i^x \leftarrow B_i \setminus M_i^x$;
 2: Minimum number of support points: $s_{min} \leftarrow |B_i|/\eta$;
 3: Set iterative parameters: $c = 0, C, \delta$;
 4: if $|M_i^x| \geq s_{min}$ then
 5:     $Q_i^x \leftarrow M_i^x$;
 6: else
 7:     NUM $\leftarrow \min(s_{min} - |M_i^x|, |N_i^x|)$;
 8:     repeat
 9:         $\hat{Q}_i^x \leftarrow M_i^x +$ randomly selected NUM points in $N_i^x$;
10:         Compute difference: $e \leftarrow \beta(I(B_i), I(\hat{Q}_i^x))$;
11:         Save $\hat{Q}_i^x$ which has minimum $e$ as $Q_i^x$;
12:         $c = c + 1$;
13:     until $c > C$ or $e < 0$
14: end if
```

2.3.1. *Matching cost*

The total matching cost (3) involves, for each pixel p in region B_i, the individual matching cost $C^x(\mathbf{p}, \mathbf{q})$, which represents the dissimilarity between pixel $\mathbf{p}$ in the reference image and pixel $\mathbf{q}$ in the target image.

As with OSF, we define this matching cost as a weighted combination of two types of cost, the projection cost $C_{pro}^x(\mathbf{p}, \mathbf{q})$ and the refinement cost $C_{ref}^x(\mathbf{p}, \mathbf{q})$:

$$C^x(\mathbf{p}, \mathbf{q}) = \theta_{1,x} C_{pro}^x(\mathbf{p}, \mathbf{q}) + \theta_{2,x} C_{ref}^x(\mathbf{p}, \mathbf{q}), \tag{5}$$

where $\theta_{1,x}$ and $\theta_{2,x}$ are weights decided via cross-validation.

The projection cost C_{pro}^x is defined on the assumption that the feature of pixel $\mathbf{p}$ should be similar to that of pixel $\mathbf{q}$ obtained through projection by the homography matrix:

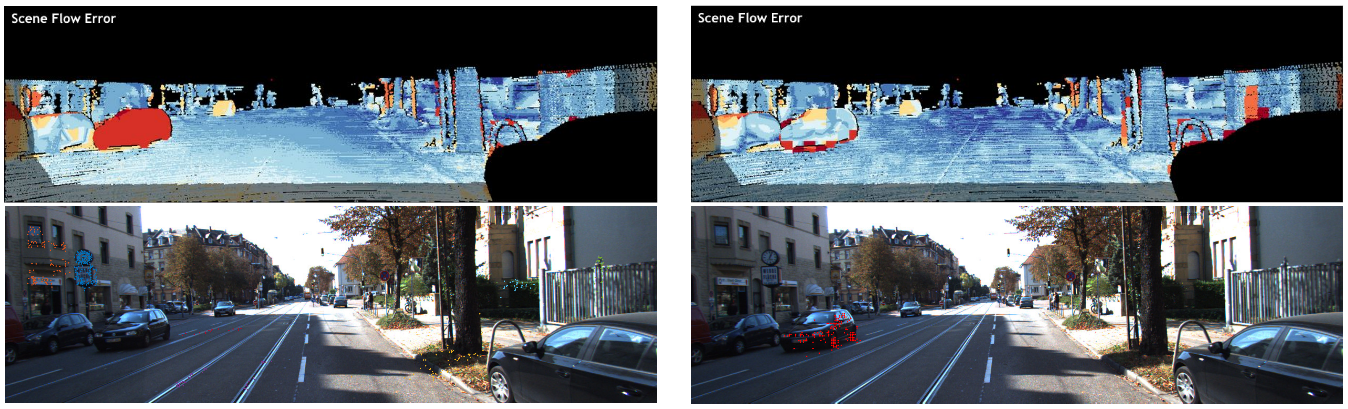
$$C_{pro}^x(\mathbf{p}, \mathbf{q}) = \begin{cases} \min(f(\mathbf{p}, \mathbf{q}), C_{max}) & \text{if } \mathbf{p} \in Q^x; \\ 0 & \text{otherwise}, \end{cases} \tag{6}$$

where $f(\mathbf{p}, \mathbf{q})$ denotes the Hamming distance of the respective 5×5 Census descriptors [19] between pixel $\mathbf{p}$ in the reference image and its corresponding pixel $\mathbf{q}$ in the target image; for the pixel $\mathbf{q}$ outside of the target image domain, we use the outlier value C_{max} as C_{pro}^x; and $Q^x = \{Q_i^x | s_i \in S\}$ is the set of support points for image pair x. The selection of Q_i^x will be given in Section 2.3.2.

The refinement cost C_{ref}^x follows the assumption that the pixel's refined motion should be close to the initial motion for the pixel with high-confidence matching. It is defined as

$$C_{ref}^x(\mathbf{p}, \mathbf{q}) = \begin{cases} \rho_\tau(||m^x(\mathbf{p}) - \mathbf{q}||_2) & \text{if } \mathbf{p} \in M^x; \\ 0 & \text{otherwise}, \end{cases} \tag{7}$$

where $m^x(\mathbf{p})$ is the pixel in the target image corresponding to a pixel $\mathbf{p}$ in M^x, in which M^x is the set of pixels in the reference image which has high-confidence matching as obtained in Section 2.2.1; and $\rho_\tau(x) = \min(|x|, \tau)$.



(a) OSF (b) Ours

Fig. 2. Lower row: moving object proposal (in red). Upper row: scene flow error (redder indicates higher). (a) OSF proposes wrong moving objects causing high foreground scene flow errors (e.g. the car highlighted in red). (b) Our method properly proposes moving objects achieving low foreground scene flow errors (the car more in blue).

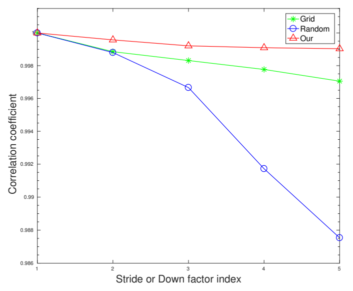


Fig. 3. Correlation (between the energies of three down-sampling strategies and the original energy without down-sampling) versus down factor. Regarding the down factor η of our method in Algorithm 1, indexes 1-5 on the horizontal axis correspond to $\eta = 1, 3, 5, 7, 10$, respectively.

2.3.2. *Support points selection*

As region B_i always contains a lot of matches with low confidence and duplicated calculation, we aim to find a subset of B_i to improve the computational efficiency for matching cost calculation while not at the expense of accuracy.

Intuitively, we can perform down-sampling to attain a subset by randomly or regularly (e.g., via grid) selecting points from the superpixel. However, such a way of selection will reduce the accuracy in practice. To illustrate this, we plot in Fig. 3 the correlation coefficient between the energy of each of three down-sampling strategies and the original energy in (1) obtained without down-sampling. We can observe that the random or regular grid-based down-sampling has clearly decreasing correlation with the original result when sampling fewer points (i.e., with larger down factors). In contrast, our new down-sampling method, which is based on support points selection, has more stable correlation for various down factors than other down-sampling strategies.

In our down-sampling strategy, we first hold the points that have high-confidence matching in the target image into Q_i^x, the set of support points for superpixel s_i in image pair x. Furthermore, because holding enough points from a superpixel is vital to the enhancement of robustness, but some superpixels may not have enough points with high-confidence matching, we have to select some points with low-confidence matching from these superpixels to meet the minimum number of support points for a superpixel. When selecting such points, we would like to maintain the structure of a superpixel in the sampling process, because we will need to convert the pixel values to other feature space by census transform.

Before we introduce the mathematical details how we select these support points, let us establish some notation. Firstly, I denotes the reference image, and I^x stands for the corresponding image for $x \in \{\text{stereo}, \text{flow}, \text{cross}\}$; for example, I^{stereo} is the right image at t_0 in Fig. 1. Secondly, B_i denotes the pixel set for superpixel i in the reference image, with $|B_i|$ denoting the number of pixels in B_i and $I(B_i)$ representing pixel values for pixels belong to B_i in the reference image I; M_i^x is the pixel set of pixels having matching with high confidence for superpixel s_i as in Section 2.2.1. Thirdly, η is the down factor that control the minimum number s_{min} of support points.

In our algorithm, the structure of superpixel is simplified to the probability distribution of pixel values. For every superpixel, we can assume the probability distribution of pixel values is $P(I(B_i))$. We would like to make $P(I(Q_i^x))$ similar to $P(I(B_i))$, where $P(I(Q_i^x))$ is probability distribution of support point values in the reference image I:

$$P(I(B_i)) \sim P(I(Q_i^x))$$

If we assume that $P(I(B_i))$ follows the Gaussian distribution

Table 1. Evaluation of proposed strategies on KITTI. Top 2 results are in bold.

Alg. \Error(%)	D1-all	D2-all	F1-all	SF-all	time(s)
OSF [6]	**4.53**	6.97	8.32	9.66	**3000**
pm	4.65	5.81	7.03	8.54	3002
pm + rop	**4.63**	**5.63**	**6.70**	**8.20**	3002
pm + rop + eff	4.72	**5.70**	**6.79**	**8.32**	**600**

with mean $E(I(B_i))$ and variance $Var(I(B_i))$, we will have

$$Var(I(Q_i^x)) \approx Var(I(B_i)), \quad E(I(Q_i^x)) \approx E(I(B_i)).$$

In the algorithm we randomly select points to get the Q_i^x whose sample mean and variance are close enough to the original values. To measure the difference, we use

$$\beta(I(B_i), I(\hat{Q_i^x})) = \frac{(E(I(B_i)) - E(I(\hat{Q_i^x})))^2 - \delta^2}{(E(I(B_i)) + \epsilon)^2} + \frac{(Var(I(B_i)) - Var(I(\hat{Q_i^x})))^2 - \delta^2}{(Var(I(B_i)) + \epsilon)^2}, \quad (8)$$

where ϵ is a constant used for normalization, and δ controls the difference between B_i and Q_i^x. In our experiments, C, ϵ and δ are empirically set to 20, 0.1 and 0.2, respectively. The pseudo code of our algorithm is in Algorithm 1.

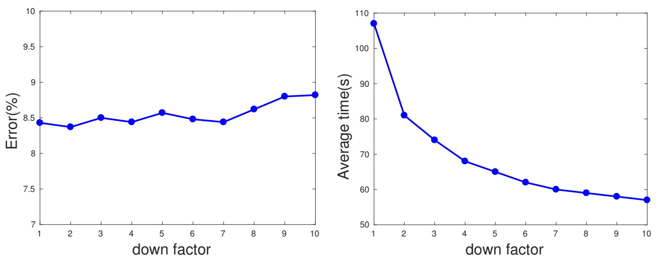
3. EXPERIMENTAL RESULTS

We evaluate our approach (SPSOSF) by comparing it with OSF and other state-of-the-art methods on the KITTI 2015 benchmark dataset. The error is computed by the 3 pixels evaluation threshold used by KITTI. Our experiments are all completed on a single i7 core running at 3.5 GHz.

3.1. The proposed method vs OSF

The experimental results from comparison with OSF on the KITTI validation set is listed in Table. 1, in which "pm" means CNN-based patch matching, "rop" stands for robust moving object proposal, and "eff" indicates efficient matching cost estimation by using support points only. In the experiments, the number of shape particles, motion particles and MP-PBP iterations are set to 30, 10 and 50, respectively. From Table. 1, our partial-version algorithm (pm+rop) including robust moving object proposals can largely reduce the error rate of OSF; and moreover, our full-version algorithm with both robust proposals and support points selection (pm+rop+eff) can further reduce the computational time remarkably with little loss in accuracy.

To illustrate this further, we draw for our algorithm the curve of SF-all error versus down factor η, along with the curve of average computational time versus η, in Fig. 4. We can observe from Fig. 4(a) that the accuracy of our algorithm



(a) (b)

Fig. 4. Evaluation of our algorithm on the validation set from the KITTI 2015 scene flow training dataset. (a) shows the SF-all error vs the down factor; and (b) shows average running time vs the down factor.

Table 2. Results on KITTI. Top 2 results are in bold.

Alg. \Error(%)	D1-all	D2-all	F1-all	SF-all	time(s)
OSF [6]	5.79	7.77	7.83	10.23	3000
ISF [9]	**4.46**	**5.95**	**6.22**	**8.08**	600
PRSF [4]	6.24	12.69	13.83	16.44	150
CSF [7]	5.98	10.06	12.96	15.71	**80**
SPSOSF	**5.02**	**6.68**	**7.46**	**9.55**	**70**

is barely harmed even when the down factor η goes up to 7, which can largely improve the computational efficiency (with much less computational time needed; see Fig. 4(b)). Hence, also for illustrative purposes, we set the minimum number of support points as $s_{min} = |B_i|/7$ in Section 3.2.

3.2. The proposed method vs state-of-the-art algorithms

We compare our proposed method with other state-of-the-art two-frame methods of the best-ranked KITTI 2015 submissions in the scene flow category. The results are listed in Table 2, in which our algorithm is denoted as SPSOSF, which is the same algorithm as pm+rop+eff but with different values of iteration parameters for faster computation: the number of shape particles, motion particles and MP-PBP iterations are now set to 10, 5 and 15, respectively.

From the accuracy perspective, we reach 9.55% EPE compared with 10.23% of the OSF algorithm. Although ISF of [9] achieves the best results, it needs to segment instances by object detection and segmentation computed by CNN. Our algorithm does not need complicated pre-processing like that, but still can achieve competitive accuracy and needs much less computational time than ISF. From the efficiency perspective, our algorithm only needs 70 seconds to run with only slightly lower accuracy than the best performer, ISF. Algorithms such as PRSF [4] and CSF [7] not only are slower than our algorithm, but also have much higher errors.

In short, the results show that our new algorithm with support points selection and robust moving object proposal can be a strong alternative to much more sophisticated state-of-the-art in practice, in terms of a better balance between efficiency and accuracy.

4. CONCLUSIONS

In this paper we proposed a new, efficient and robust algorithm for object scene flow estimation, through developing support points selection to reduce computational time, and through proposing robust moving object proposals. These two contributions, though simple, ensure a remarkable improvement in both efficiency and accuracy from the original object scene flow method, as well as making the proposed algorithm a strong practicable alternative to much more sophisticated state-of-the-art competitors.

5. REFERENCES

[1] Jan Čech, Jordi Sanchez-Riera, and Radu Horaud, "Scene flow estimation by growing correspondence seeds," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 3129–3136.

[2] Christoph Vogel, Stefan Roth, and Konrad Schindler, "View-consistent 3D scene flow estimation over multiple frames," in *European Conference on Computer Vision*. Springer, 2014, pp. 263–278.

[3] Christoph Vogel, Konrad Schindler, and Stefan Roth, "3D scene flow estimation with a rigid motion prior," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1291–1298.

[4] Christoph Vogel, Stefan Roth, and Konrad Schindler, "Piecewise rigid scene flow," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1377–1384.

[5] Christoph Vogel, Konrad Schindler, and Stefan Roth, "3D scene flow estimation with a piecewise rigid scene model," *International Journal of Computer Vision*, vol. 115, no. 1, pp. 1–28, 2015.

[6] Moritz Menze and Andreas Geiger, "Object scene flow for autonomous vehicles," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[7] Zhaoyang Lv, Chris Beall, Pablo F Alcantarilla, Fuxin Li, Zsolt Kira, and Frank Dellaert, "A continuous optimization approach for efficient and accurate scene flow," in *European Conference on Computer Vision*. Springer, 2016, pp. 757–773.

[8] Michal Neoral and Jan Åăochman, "Object scene flow with temporal consistency," in *Proc. of the Computer Vision Winter Workshop (CVWW)*, 2017.

[9] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger, "Bounding boxes, segmentations and object coordinates: How important is recognition for 3D scene flow estimation in autonomous driving scenarios?," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2574–2583.

[10] Zhile Ren, Deqing Sun, Jan Kautz, and Erik B Sudderth, "Cascaded scene flow prediction using semantic segmentation," *arXiv preprint arXiv:1707.08313*, 2017.

[11] Andreas Geiger, Philip Lenz, and Raquel Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3354–3361.

[12] Jure Zbontar and Yann LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *Journal of Machine Learning Research*, vol. 17, no. 1-32, pp. 2, 2016.

[13] Jia Xu, René Ranftl, and Vladlen Koltun, "Accurate optical flow via direct cost volume processing," *arXiv preprint arXiv:1704.07325*, 2017.

[14] Koichiro Yamaguchi, David McAllester, and Raquel Urtasun, "Efficient joint segmentation, occlusion labeling, stereo and flow estimation," in *European Conference on Computer Vision*. Springer, 2014, pp. 756–771.

[15] Andreas Geiger, Julius Ziegler, and Christoph Stiller, "Stereoscan: Dense 3D reconstruction in real-time," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*. Ieee, 2011, pp. 963–968.

[16] Jason Pacheco, Silvia Zuffi, Michael Black, and Erik Sudderth, "Preserving modes and messages via diverse particle selection," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1152–1160.

[17] Hoang Trinh and David McAllester, "Unsupervised learning of stereo vision with monocular cues," in *Proc. of the British Machine Vision Conf.(BMVC)*, 2009, vol. 4.

[18] Vladimir Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 10, pp. 1568–1583, 2006.

[19] Ramin Zabih and John Woodfill, "Non-parametric local transforms for computing visual correspondence," in *European conference on computer vision*. Springer, 1994, pp. 151–158.