SGM-Nets: Semi-global matching with neural networks

Akihito Seki[1]* Marc Pollefeys[2,3]
[1]Toshiba Corporation [2]ETH Zürich [3]Microsoft
akihito.seki@toshiba.co.jp, marc.pollefeys@inf.ethz.ch

Abstract

This paper deals with deep neural networks for predicting accurate dense disparity map with Semi-global matching (SGM). SGM is a widely used regularization method for real scenes because of its high accuracy and fast computation speed. Even though SGM can obtain accurate results, tuning of SGM's penalty-parameters, which control a smoothness and discontinuity of a disparity map, is uneasy and empirical methods have been proposed. We propose a learning based penalties estimation method, which we call SGM-Nets that consist of Convolutional Neural Networks. A small image patch and its position are input into SGM-Nets to predict the penalties for the 3D object structures. In order to train the networks, we introduce a novel loss function which is able to use sparsely annotated disparity maps such as captured by a LiDAR sensor in real environments. Moreover, we propose a novel SGM parameterization, which deploys different penalties depending on either positive or negative disparity changes in order to represent the object structures more discriminatively.

Our SGM-Nets outperformed state of the art accuracy on KITTI benchmark datasets.

1. Introduction

Stereo disparity estimation is one of the most important problems in computer vision. The disparity map is widely used, for example in object detection [13], surveillance [29], autonomous driving for cars [27], and unmanned air vehicles [24].

Many disparity estimation methods have been proposed for many years [32]. A standard pipeline for the dense disparity estimation starts by finding local correspondences between stereo images. Incorrect correspondences occur due to various reasons such as occlusion and pixel intensity noise. In order to refine the disparity map, regularization methods [15, 31, 33, 35] and some filters [36, 40, 38] are applied, and the fine dense disparity is finally obtained. In the KITTI website [1], many state of the art researches focus on accurate local correspondence methods with deep learning [38, 21, 3] and apply Semi-global matching (SGM) [15] as regularization. Recently, deep learning methods such as FlowNet [6] and DispNet [22] which play end-to-end of the pipeline have been proposed. However, the methods haven't achieved sufficient accuracy compared to the standard pipeline so far. We guess one of the reason for the lower accuracy comes from the differences between training and testing datasets as mentioned in [9, 26].

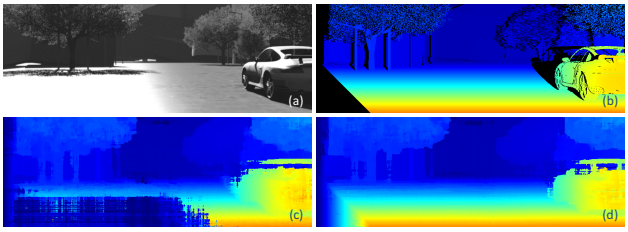


Figure 1. (a) Left image. (b) Ground truth disparity map. Occlusion in black. Disparity map by using SGM with (c) Hand tuned penalties and (d) SGM-Net. The difference of inputs is only SGM penalties.

In this paper, we focus on the regularization part of the standard pipeline since many sophisticated local correspondence methods have been proposed. SGM is a widely used regularization method due to its high accuracy while keeping low computation cost. Some papers have reported its real time computation even on mobile devices [16, 14]. SGM has penalty-parameters, we call them "penalties" in this paper, and they control the smoothness and discontinuity of the disparity map. So far, the penalties are designed empirically and are uneasy to be tuned.

We consider the penalties should be different depending on 3D object structures. For instance, the penalties should capture the fact that road is smooth. We propose a learning based penalties prediction method which uses CNNs. CNNs provide high performances from primitive level processing such as stereo correspondences to high level ones such as scene classification [2, 20] and object detection [11, 39]. Deep learning using a CNN offers a promising way for our purpose. However, it isn't straightforward to involve the CNN for the task, i.e. How to train and construct the CNNs for SGM?

The contributions of this paper are the following: (1) A

*This work has been done while the first author was visiting at ETH Zürich.

learning based penalties estimation for SGM. We propose a new loss function in order to train neural networks which inputs are small patches and their location. To the best of our knowledge, we are the first to leverage neural networks for SGM. Figure 1(c) shows a dense disparity map obtained with hand tuned SGM penalties. Erroneous pixels on road region are correctly estimated by our method in Fig. 1(d). (2) New SGM parameterization that separates the positive and negative disparity changes in order to represent object structures discriminatively. (3) Quantitative evaluation on both synthetic [22] and real scenery [10, 23]. The datasets are very challenging due to saturation of intensity, reflections, motion blurs, and image noises. SGM-Nets were able to outperform state of the art accuracy on KITTI datasets without the need for an explicit foreground shape prior such as a vehicle.

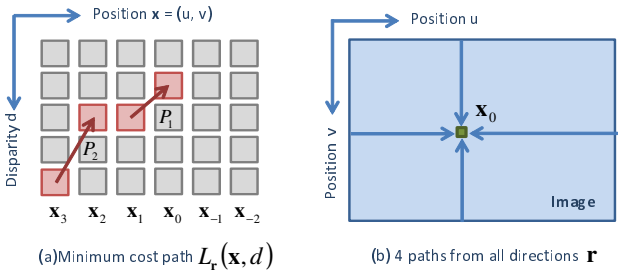
In the following sections, we first focus on related works in Sec. 2. Then, we explain SGM so that some equations are ready for our method (Sec. 3). In Sec. 4, SGM-Nets which predict SGM penalties are described. We address the implementation details in Sec. 5. An effectiveness of our method is demonstrated with both synthetic and real datasets in Sec. 6. Section 7 summarizes this paper.

2. Related works

A standard pipeline for dense disparity estimation consists of two parts, i.e. local correspondence and regularization. Learning based correspondence functions have been widely studied [38, 21, 3]. They leverage CNNs for local correspondence and hand tuned SGM for regularization. In this section, we will discuss hand tuned SGM and learning based Markov Random Field (MRF) which is a general case of SGM [7].

Hand tuned penalties for SGM. So far, SGM penalties have been manually tuned or designed [17, 15, 38, 28]. The simplest way is that the penalties are fixed over images [17]. Another assumption is that pixels which have a large gradient, i.e. edges, are more likely to be discontinuities, which means that the penalties at the pixels should be mitigated in order to allow disparity jumps [15]. In more advanced method, the penalties are set smaller not only when edges in a reference image are detected, but also they coincide with edges at the corresponding position in a target image [38]. In [28], stereo correspondence confidence is estimated. Then pixels with high confidence should be trusted and the penalties at the pixels are mitigated.

Learning based penalties for MRF. Conditional Random Field (CRF) parameters learning method for stereo was proposed [25], however the penalties are learned over manually tuned intervals of image gradients. Some papers which learn CRF parameters with CNN have been proposed [41, 19, 34]. However, [41, 19] aim for semantic segmentation, and their formulations and ideas are unable to be applied to learn SGM penalties. Very recently, the method for stereo was proposed [34], however some of energy terms (local smoothness and object potentials) are designed manually.



(a)Minimum cost path $L_{\mathbf{r}}(\mathbf{x}, d)$ (b) 4 paths from all directions $\mathbf{r}$

Figure 2. Aggregation of costs and estimated disparity.

Our method fully learns SGM penalties with CNN in order to improve disparity maps. Moreover, not only standard SGM parameterization but also new parameterization that separates the positive and negative disparity changes are able to be applied. We end up using CNNs for matching (based on [38]) AND for determining SGM penalties.

3. Semi-global matching

Before introducing SGM-Net, we first explain Semi-Global Matching (SGM) [15]. An energy function $\mathbb{E}$ for solving SGM is defined as

$$\mathbb{E}(D) = \sum_{\mathbf{x}} \Big(C(\mathbf{x}, d^{\mathbf{x}}) + \sum_{\mathbf{y} \in \mathbf{N}_{\mathbf{x}}} P_1 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| = 1] + \sum_{\mathbf{y} \in \mathbf{N}_{\mathbf{x}}} P_2 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| > 1] \Big). \quad (1)$$

$C(\mathbf{x}, d^{\mathbf{x}})$ represents a matching cost at pixel $\mathbf{x} = (u, v)$ of disparity $d^{\mathbf{x}}$. The first term represents the sum of matching costs at all pixels for the disparity map D. The second term represents slanted surface penalty P_1 for all pixels $\mathbf{y}$ in the neighborhood $\mathbf{N}_{\mathbf{x}}$ of $\mathbf{x}$. The third term indicates penalty P_2 for discontinuous disparity. P_2 is typically set small according to the magnitude of the image gradient, for example $P_2 = P_2' / |I(\mathbf{x}) - I(\mathbf{y})|$ so that the discontinuities are easily selected [15]. $T[\cdot]$ represents Kronecker delta function which gives 1 when a condition in the bracket is satisfied, otherwise 0.

In order to minimize $\mathbb{E}(D)$ in Eq. (1), a cost $L'_{\mathbf{r}}(\mathbf{x}, d)$ along a path in the direction $\mathbf{r}$ of the pixel $\mathbf{x}$ at disparity d as shown in Fig. 2(a) is formulated as

$$L'_{\mathbf{r}}(\mathbf{x}_0, d) = c(\mathbf{x}_0, d) + \min \Big(L'_{\mathbf{r}}(\mathbf{x}_1, d), L'_{\mathbf{r}}(\mathbf{x}_1, d-1) + P_1, L'_{\mathbf{r}}(\mathbf{x}_1, d+1) + P_1, \min_{i \neq d \pm 1} L'_{\mathbf{r}}(\mathbf{x}_1, i) + P_2 \Big).$$

$\mathbf{x}_1$ and $c(\mathbf{x}, d)$ represents the previous pixel $(\mathbf{x}_0 - \mathbf{r})$ and a pixel-wise matching cost, which is given by for instance ZNCC (Zero Mean Normalized Cross-Correlation), Census [37], or CNN based methods [38, 21, 30, 3]. In order to avoid very large values due to accumulation along the

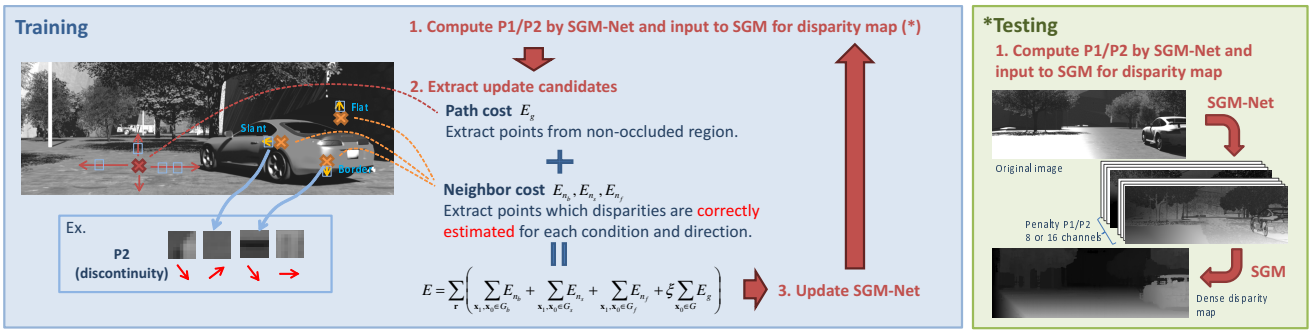


Figure 3. Overview of SGM-Net. SGM estimates dense disparity by incorporating penalty P_1 and P_2 from SGM-Net. SGM-Net is iteratively trained on each aggregation direction with image patches and their positions.

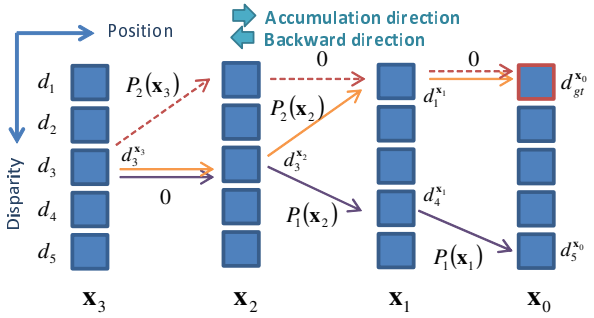


Figure 4. Consecutive 4 pixels and their 5 candidate disparities at each pixel. The orange and purple line represent the path from correct disparity $d_{gt}^{\mathbf{x}_0}$ and d_5 at root pixel $\mathbf{x}_0$, respectively.

path, the minimum path cost at the previous pixel $\mathbf{x}_1$ is subtracted, and we get

$$L_{\mathbf{r}}(\mathbf{x}_0, d) = c(\mathbf{x}_0, d) + \min\Big(L_{\mathbf{r}}(\mathbf{x}_1, d), L_{\mathbf{r}}(\mathbf{x}_1, d-1) + P_1, L_{\mathbf{r}}(\mathbf{x}_1, d+1) + P_1, \min_{i \neq d \pm 1} L_{\mathbf{r}}(\mathbf{x}_1, i) + P_2\Big) - \min_k L_{\mathbf{r}}(\mathbf{x}_1, k). \tag{2}$$

The disparity D at pixel $\mathbf{x}_0$ is computed by the winner-takes-all strategy of the aggregated costs of all directions $\mathbf{r}$ (4 in Fig. 2(b)) as below:

$$D(\mathbf{x}_0) = \arg\min_d \sum_{\mathbf{r}} L_{\mathbf{r}}(\mathbf{x}_0, d). \tag{3}$$

4. SGM-Net

Figure 3 illustrates an overview of our proposed method. Our neural network which we call SGM-Net provides P_1 and P_2 at each pixel. It consists of two phases: training and testing. During the training phase, SGM-Net is iteratively trained by minimizing two kinds of costs, which are "Path cost" in Sec. 4.1.1 and "Neighbor cost" in Sec. 4.1.2. In the testing, dense disparity is estimated by SGM the penalties of which are predicted by SGM-Net.

We first explain standard parameterization of SGM in Sec. 4.1. Then, more discriminative parameterization in Sec. 4.2. An architecture of SGM-Net is explained in Sec. 4.3.

4.1. Standard parameterization

4.1.1 Path cost

As shown in Eq. (3), a necessary condition to obtain the correct disparity is that a path traversing the correct disparity $d_{gt}^{\mathbf{x}_0}$ at pixel $\mathbf{x}_0$ should be smaller than any other paths, i.e. a cost $L_{\mathbf{r}}$ at pixel $\mathbf{x}_0$ must satisfy $L_{\mathbf{r}}(\mathbf{x}_0, d_i^{\mathbf{x}_0}) > L_{\mathbf{r}}(\mathbf{x}_0, d_{gt}^{\mathbf{x}_0})$, $\forall d_i \in [0, d_{max}] \neq d_{gt}$. We formulate it with a hinge loss function as below:

$$E_g = \sum_{d_i^{\mathbf{x}_0} \neq d_{gt}^{\mathbf{x}_0}} \max\Big(0, L_{\mathbf{r}}(\mathbf{x}_0, d_{gt}^{\mathbf{x}_0}) - L_{\mathbf{r}}(\mathbf{x}_0, d_i^{\mathbf{x}_0}) + m\Big), \tag{4}$$

where m means margin. The hinge loss function allows easier formulation of back-propagation compared to other functions such as softmax loss. In order to allow the back-propagation of the loss function, we should clarify the gradients of Eq. (4) with respect to P_1 and P_2. We first show with an example in Fig. 4. Here, we pay attention to the costs L of disparity at pixel $\mathbf{x}_0$. The costs L are accumulated between pixel $\mathbf{x}_3$ and $\mathbf{x}_0$ along the path. The traversed disparities from pixel $\mathbf{x}_0$ can be chased by tracking in a backward direction. In this figure, the costs of disparity $d_5^{\mathbf{x}_0}$ and $d_{gt}^{\mathbf{x}_0}$ at pixel $\mathbf{x}_0$ are represented as

$$\begin{aligned} L(\mathbf{x}_0, d_{gt}^{\mathbf{x}_0}) &= c(\mathbf{x}_0, d_{gt}^{\mathbf{x}_0}) + c(\mathbf{x}_1, d_1^{\mathbf{x}_1}) + c(\mathbf{x}_2, d_3^{\mathbf{x}_2}) \\ &\quad + c(\mathbf{x}_3, d_3^{\mathbf{x}_3}) + P_2(\mathbf{x}_2) - \beta \\ L(\mathbf{x}_0, d_5^{\mathbf{x}_0}) &= c(\mathbf{x}_0, d_5^{\mathbf{x}_0}) + c(\mathbf{x}_1, d_4^{\mathbf{x}_1}) + c(\mathbf{x}_2, d_3^{\mathbf{x}_2}) \\ &\quad + c(\mathbf{x}_3, d_3^{\mathbf{x}_3}) + P_1(\mathbf{x}_1) + P_1(\mathbf{x}_2) - \beta, \end{aligned} \tag{5}$$

where β means the minimum path cost in Eq. (2). To generalize them, the accumulated cost along the path becomes

$$\begin{aligned} L_{\mathbf{r}}(\mathbf{x}_0, d_i^{\mathbf{x}_0}) = \gamma + \sum_n \Big(&P_{1,\mathbf{r}}(\mathbf{x}_n) T[|\delta d^{\mathbf{x}_n \leftarrow d_i^{\mathbf{x}_0}}| = 1] \\ &+ P_{2,\mathbf{r}}(\mathbf{x}_n) T[|\delta d^{\mathbf{x}_n \leftarrow d_i^{\mathbf{x}_0}}| > 1]\Big) \end{aligned} \tag{6}$$

$\delta d^{\mathbf{x}_n \leftarrow d_i^{\mathbf{x}_0}}$ means the series of disparity difference $(d^{\mathbf{x}_k} - d^{\mathbf{x}_{k-1}}, \forall k \in [1, n])$ between consecutive pixels $\mathbf{x}_k$ and $\mathbf{x}_{k-1}$ along direction $\mathbf{r}$, the root of which is disparity $d_i^{\mathbf{x}_0}$ at

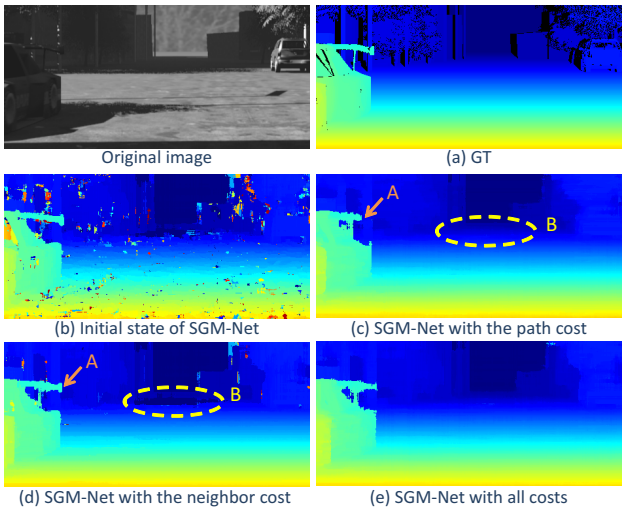


Figure 5. Comparison of the costs for the loss function.

pixel $\mathbf{x}_0$. γ represents accumulated matching costs and subtracted minimum costs at every pixels. Note that γ doesn't contain P_1 and P_2.

Eq. (6) is put into Eq. (4), then the loss function E_g with non-zero cost at pixel $\mathbf{x}_0$ is differentiated with respect to P_1 and P_2. Finally, below equations are obtained:

$$\frac{\partial E_g}{\partial P_{1,\mathbf{r}}} = \sum_{d_i^{\mathbf{x}_0} \neq d_{gt}^{\mathbf{x}_0}} \sum_n \left(T[|\delta d^{\mathbf{x}_n \leftarrow d_{gt}^{\mathbf{x}_0}}| = 1] - T[|\delta d^{\mathbf{x}_n \leftarrow d_i^{\mathbf{x}_0}}| = 1] \right)$$
$$\frac{\partial E_g}{\partial P_{2,\mathbf{r}}} = \sum_{d_i^{\mathbf{x}_0} \neq d_{gt}^{\mathbf{x}_0}} \sum_n \left(T[|\delta d^{\mathbf{x}_n \leftarrow d_{gt}^{\mathbf{x}_0}}| > 1] - T[|\delta d^{\mathbf{x}_n \leftarrow d_i^{\mathbf{x}_0}}| > 1] \right). \tag{7}$$

For example, a derivative of E_g in Eq. (5) is obtained as follows:

$$\frac{\partial E_g}{\partial P_1(\mathbf{x}_1)} = -1, \quad \frac{\partial E_g}{\partial P_2(\mathbf{x}_1)} = 0, \quad \frac{\partial E_g}{\partial P_2(\mathbf{x}_2)} = 1, \tag{8}$$
$$\text{when} \quad E_g = L_{\mathbf{r}}(\mathbf{x}_0, d_{gt}^{\mathbf{x}_0}) - L_{\mathbf{r}}(\mathbf{x}_0, d_5^{\mathbf{x}_0}) + m > 0.$$

With the equations, we are able to minimize the loss function by using the standard framework, i.e. forward and back propagation. We call this loss function "Path cost".

Note that the path cost does not require dense ground truth so that we can easily use the dataset taken under real environment such as KITTI [10]. On the other hand, the path cost has a potential problem. The intermediate paths aren't taken into account directly. For instance, the red dot lines in Fig. 4 indicate the paths which traverse the correct disparities at each pixel. Orange lines, which have paths before and after pixel $\mathbf{x}_2$ that are different from the correct ones, lead to wrong penalties at pixels $\mathbf{x}_3$ and $\mathbf{x}_2$.

The partially wrong penalties create artifacts as shown in Fig. 5. Fig. 5(c) shows a disparity map by SGM the penalties of which are predicted by SGM-Net trained only this loss function. Comparing to initial parameters of SGM-Net (Fig. 5(b)), the disparity map becomes better, however a detail such as A has disappeared.

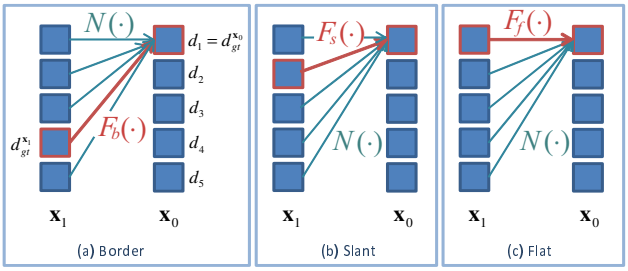


Figure 6. Considerable relations of disparities between consecutive pixels: (a) Border, (b) Slant, and (c) Flat. Correct path and wrong ones are represented in red and blue-green, respectively.

4.1.2 Neighbor cost

In order to remove the ambiguity of disparities traversed along the path, we introduce "Neighbor cost" function. The basic idea is that the path which traverses correct disparities at consecutive pixels must have the smallest among all paths as shown in Fig. 6. In this figure, a cost $F_b(\cdot)$, $F_s(\cdot)$, or $F_f(\cdot)$ along the path in red is smaller than the other costs $N(\cdot)$ in green. The neighbor cost is represented as

$$E_{n_X} = \sum_{d \neq d_{gt}^{\mathbf{x}_1}} \max\left(0, F_X(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1}) - N(\mathbf{x}_1, d_{gt}^{\mathbf{x}_0}, d) + m\right), \tag{9}$$

where $N(\cdot)$ means

$$N(\mathbf{x}_1, d_{gt}^{\mathbf{x}_0}, d) = L_{\mathbf{r}}(\mathbf{x}_1, d) + P_{1,\mathbf{r}}(\mathbf{x}_1) T[|d_{gt}^{\mathbf{x}_0} - d| = 1] + P_{2,\mathbf{r}}(\mathbf{x}_1) T[|d_{gt}^{\mathbf{x}_0} - d| > 1] \tag{10}$$

and $F_X(\cdot)$ is a function depending on relations of a disparity change between consecutive pixels: border $F_b(\cdot)$, slant $F_s(\cdot)$, and flat $F_f(\cdot)$.

Border is the case where there is a discontinuity in consecutive pixels as shown in Fig. 6(a). The path cost $F_X(\cdot)$ between $d_{gt}^{\mathbf{x}_0}$ and $d_{gt}^{\mathbf{x}_1}$ is defined as

$$F_b(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1}) = L_{\mathbf{r}}(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1}) + P_{2,\mathbf{r}}(\mathbf{x}_1). \tag{11}$$

Slant (Fig. 6(b)) represents a surface that has a small disparity change such as road plane. $F_X(\cdot)$ becomes

$$F_s(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1}) = L_{\mathbf{r}}(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1}) + P_{1,\mathbf{r}}(\mathbf{x}_1). \tag{12}$$

Flat (Fig. 6(c)) is a frontoparallel plane to a camera. In this case, none of the penalty is added. It is defined as

$$F_f(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1}) = L_{\mathbf{r}}(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1}). \tag{13}$$

Eq. (9) can be differentiated in a similar way to the path cost explained previous section. By using the neighbor cost, the detailed part A in Fig. 5(d) are preserved.

A necessary condition to apply the neighbor cost is that the disparity at pixel $\mathbf{x}_1$ has to be estimated correctly, i.e.

accumulated cost $L_{\mathbf{r}}(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1})$ must have the smallest accumulated cost $L_{\mathbf{r}}$ of all disparities. Otherwise, the disparity at pixel $\mathbf{x}_0$ is unlikely to be correctly predicted. The advantage of the neighbor cost is that the aggregated cost at both consecutive pixels is supposed to be minimized at the correct disparity. Meanwhile, it is difficult to apply the neighbor cost to all pixels because of the necessary condition. When SGM-Net is trained only with the neighbor cost, erroneous pixels occur (B in Fig. 5).

In order to compensate the advantage and difficulty of the path and neighbor costs, they are put together, and finally the loss function becomes

$$E = \sum_{\mathbf{r}\in R}\Big(\sum_{\mathbf{x}_1,\mathbf{x}_0\in G_b} E_{n_b} + \sum_{\mathbf{x}_1,\mathbf{x}_0\in G_s} E_{n_s} + \sum_{\mathbf{x}_1,\mathbf{x}_0\in G_f} E_{n_f} + \xi \sum_{\mathbf{x}_0\in G} E_g\Big), \tag{14}$$

where ξ means a blending ratio. We randomly extracted the same number of pixels for border G_b, slant G_s, and flat G_f on each direction $\mathbf{r}$. All G_* have annotation of true disparity. For the path cost, we randomly select from G which pixels have the ground truth. The magnitude of penalties P_1 and P_2 are related to accumulated costs $L_{\mathbf{r}}$. Meanwhile, the accumulated costs also depend on the penalties. Therefore, the penalties are estimated iteratively as shown in Fig. 3. The disparity map given by SGM-Net trained with Eq. (14) is shown in Fig. 5(e).

4.2. Signed parameterization

We have explained standard parameterization of SGM. In this section, we propose a new parameterization. Figure 7(a) shows a basic idea of this parameterization. P_1 and P_2 have different penalties depending on either positive or negative disparity change so we call it "signed parameterization". This strategy is observed to work well for structures such as road surface and side wall (Fig. 7(b)). Disparities along top to bottom direction on the road (red), which is able to be assumed as slanted plane, is more likely to become larger, so P_1^- tends to be larger than P_1^+. As disparities on the left side wall (green) can be considered the same way, P_1^+ is more likely to be larger than P_1^- along left to right direction.

In this parameterization, the cost $L'_{\mathbf{r}}$ is modified to

$$L_{\mathbf{r}}^{'\pm}(\mathbf{x}_0, d) = c(\mathbf{x}_0, d) + \min\Big(L_{\mathbf{r}}^{'\pm}(\mathbf{x}_1, d),$$
$$\min_{i=d\pm1} L_{\mathbf{r}}^{'\pm}(\mathbf{x}_1, i) + P_{1,\mathbf{r}}^{+}\underbrace{T[d-i=1]}_{T_1^+[\cdot]} + P_1^{-}\underbrace{T[i-d=1]}_{T_1^-[\cdot]},$$
$$\min_{i\neq d\pm1} L_{\mathbf{r}}^{'\pm}(\mathbf{x}_1, i) + P_{2,\mathbf{r}}^{+}\underbrace{T[i<d]}_{T_2^+[\cdot]} + P_{2,\mathbf{r}}^{-}\underbrace{T[i>d]}_{T_2^-[\cdot]}\Big).$$

The equation shows discriminative penalties depending on the sign of the disparity change.

A path cost $E_g^{\pm}$ is represented the same way as E_g in Eq. (4) by replacing $L_{\mathbf{r}}$ with $L_{\mathbf{r}}^{\pm}$. As in standard parameterization, $L_{\mathbf{r}}^{\pm}(\mathbf{x}_0, d)$ is computed simply by subtracting minimum value at a previous pixel from $L_{\mathbf{r}}^{'\pm}(\mathbf{x}_0, d)$. $L_{\mathbf{r}}^{\pm}$ is generalized as below:

$$L_{\mathbf{r}}^{\pm} = \gamma + \sum_n \Big(P_{1,\mathbf{r}}^{+}T_1^{+}[\cdot] + P_{1,\mathbf{r}}^{-}T_1^{-}[\cdot] + P_{2,\mathbf{r}}^{+}T_2^{+}[\cdot] + P_{2,\mathbf{r}}^{-}T_2^{-}[\cdot]\Big).$$

The derivative of $E_g^{\pm}$ can be derived from the above equation.

The neighbor cost $E_{n_X}^{\pm}$ deriving from Eq. (9) becomes more complex in this case. We have to consider five cases instead of three in the standard parameterization. There are two cases for border, two for slant, and one for flat. $N(\cdot)$ is replaced by

$$\begin{aligned} N^{\pm}(\mathbf{x}_1, d_{gt}^{\mathbf{x}_0}, d) =& L_{\mathbf{r}}^{\pm}(\mathbf{x}_1, d) \\ &+ P_{1,\mathbf{r}}^{+}(\mathbf{x}_1)T[\delta=1] + P_{1,\mathbf{r}}^{-}(\mathbf{x}_1)T[\delta=-1] \\ &+ P_{2,\mathbf{r}}^{+}(\mathbf{x}_1)T[\delta>1] + P_{2,\mathbf{r}}^{-}(\mathbf{x}_1)T[\delta<-1], \end{aligned}$$

where $\delta = d_{gt}^{\mathbf{x}_0} - d$. F_X of the border pixels is described as

$$\begin{aligned} F_b^{\pm}(\mathbf{x}_1, d_{gt}^{\mathbf{x}}) = L_{\mathbf{r}}(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1}) &+ P_{2,\mathbf{r}}^{+}(\mathbf{x}_1)T[d_{gt}^{\mathbf{x}_0} > d_{gt}^{\mathbf{x}_1}] \\ &+ P_{2,\mathbf{r}}^{-}(\mathbf{x}_1)T[d_{gt}^{\mathbf{x}_0} < d_{gt}^{\mathbf{x}_1}] \end{aligned}$$

F_X on slanted pixels is represented as

$$\begin{aligned} F_s^{\pm}(\mathbf{x}_1, d_{gt}^{\mathbf{x}}) = L_{\mathbf{r}}(\mathbf{x}_1, d_{gt}^{\mathbf{x}_1}) &+ P_{1,\mathbf{r}}^{+}(\mathbf{x}_1)T[d_{gt}^{\mathbf{x}_0} - d_{gt}^{\mathbf{x}_1} = 1] \\ &+ P_{1,\mathbf{r}}^{-}(\mathbf{x}_1)T[d_{gt}^{\mathbf{x}_1} - d_{gt}^{\mathbf{x}_0} = 1] \end{aligned}$$

F_X on flat pixels is the same function as F_f in Eq. (13).

In order to train the signed parameterization network, we minimize the loss function E in Eq. (14) in which we replace the cost functions with the extended costs for signed parameterization.

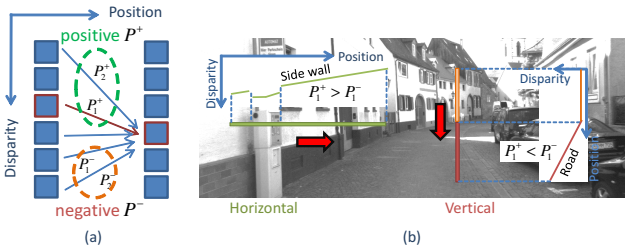


Figure 7. (a) Signed parameterization. (b) Slant structure penalty P_1 at disparities along green (side wall) and red (road) lines.

4.3. SGM-Net architecture

So far, we described the cost functions for both standard and signed parameterizations of SGM. SGM-Net architectures are explained in this section. A gray scale image patch of 5×5 pixels and its normalized position are input to the networks as shown in Fig. 8. It has two convolution layers both consisting of 16 filters with kernel size 3×3, Rectified Linear Unit (ReLU) layer after each convolution layer, concatenate layer for merging two kinds of information, two

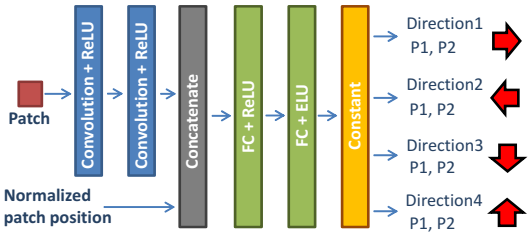


Figure 8. SGM-Net architecture for standard parameterization. Image patch and its position are input into the network. Eight parameters which are P_1 and P_2 for 4 aggregation directions.

fully connected (FC) layers of size 128 each, and ReLU after first FC layer. Additionally, we employ Exponential Linear Unit (ELU) [5] with $\alpha = 1$ and add constant value 1 so that SGM penalties keep positive value. ReLU has zero gradients on negative input values, however ELU alleviates the vanishing gradient problem. It means that ELU speeds up learning in the neural network and leads to higher accuracy.

As a preprocessing step, we subtract the mean from the image patch, and divide it by a max intensity of the image. The position of the patch is normalized by dividing it either by the width or the height of the image. In this paper, the costs are accumulated along 4 directions which are horizontal and vertical. Of course, we could add diagonal directions. The networks output 8 (= 2 $[P_1, P_2] \times 4$ [direction]) or 16 (= 4 $[P_1^+, P_1^-, P_2^+, P_2^-] \times 4$ [direction]) values, which correspond to standard and signed parameterization, respectively.

Predicted penalties are shown in Fig. 9. On the road, standard SGM-Net gives larger P_1 in a horizontal direction than that in a vertical direction at A because the road is usually flat in a horizontal direction and slanted in a vertical direction. P_2 becomes smaller at edges of B and C, which is the same as the assumption in [15]. D and E are the situations explained in Fig. 7(b). The penalties are trained as we expected.

5. Implementation

We refer the reader to the appendix for a summary of training process. SGM-Nets were implemented with Torch7 [4] on NVIDIA(R) Titan X. As an optimization method, we tested both Stochastic Gradient Descent and Adaptive Moment Estimation (Adam) [18], and finally we found the latter optimization method reached lower error. Adam is able to control learning rate properly for each parameter and keep past gradients in order to find optimal parameters quickly and stably. The networks are trained from scratch, which means they are initialized randomly, and their training took a couple of days. Most of time is consumed by loading matching cost volume files from disks, however it is much faster than computing MC-CNN [38] every iteration.

Method		Train [%] (scene_forwards)	Test [%] (scene_backwards)
Hand tuned	Fixed [17]	21.4 / 20.0	**24.0** / 23.2
	Dynamic [38]	**19.9 / 17.3**	**24.0 / 22.0**
Standard SGM-Net	Initial	29.4 / 28.9	32.9 / 32.8
	Neighbor cost	20.9 / 18.6	23.4 / 22.3
	Path cost	17.9 / 15.6	21.7 / 20.1
	All	**17.7 / 15.2**	**21.2 / 19.5**
	All (w/o pos.)	19.7 / 16.4	22.3 / 20.1
Signed SGM-Net	Initial	29.7 / 29.0	33.2 / 32.9
	Neighbor cost	21.4 / 18.2	24.3 / 22.5
	Path cost	16.8 / **14.0**	**20.4 / 18.3**
	All	**16.6 / 14.0**	**20.4 / 18.3**

Table 1. Overall Out-Noc error on synthetic dataset. Comparison of cost functions and matchers of ZNCC / MC-CNN.

6. Experimental results

In this section, we demonstrate the accuracy of our method by using "Driving" from SceneFlow datasets [22] as synthetic images and KITTI 2012 (K12) [10] and 2015 (K15) [23] datasets as real scenes. SceneFlow dataset provides pixel-wise disparity for ground truth so that the neighbor cost can be used. Actually, such dense ground truth is difficult to collect with laser sensors under real environments.

As SGM-Net parameters, we set $\xi = 0.1$ and $m = 2.5$ for all experiments.[1]

6.1. Synthetic images

Among some settings of "Driving", we selected 35mm focal length and slow motion images so that a layout and blur of the images look similar to KITTI datasets. Road regions on the bottom of the images have large disparities, hence it makes difficult to train the network on GPU due to memory size restriction. Therefore, we cut both top and bottom 100 pixels from the original images and got 960×340 pixels images. We extracted stereo pairs every 5 frames in order to remove similar scenes. Finally, we got each 160 frames from "forwards" and "backwards" scenes for training and testing, respectively.

As a comparison, we employ hand tuned penalties for SGM which are fixed [17] or dynamically determined [38]. The penalties are tuned so as to reach the minimum error over training images, and they are applied to testing images. On the latter method, the penalties are based on the magnitude of image intensity difference along an aggregation direction, which takes into account intensity difference of both the left image and the corresponding pixels in the right image. Additionally, there is a parameter which controls the penalties depending on the aggregation direction.

Here we compare not only to the hand tuning methods but also to behaviour of SGM-Nets with respect to combinations of the cost functions and advantage of positional in-

[1] Except MC-CNN in Sec. 6.1, we set $m = 5.0$ in this case.

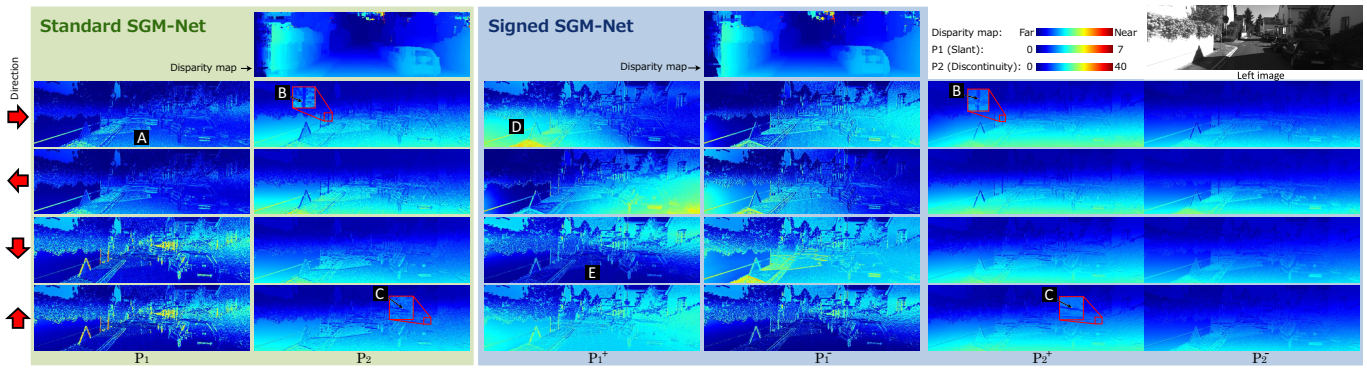


Figure 9. Predicted P_1 and P_2 by standard and signed SGM-Nets. The color of P1 and P2 encodes strength of respective values, blue and red mean small and large, respectively.

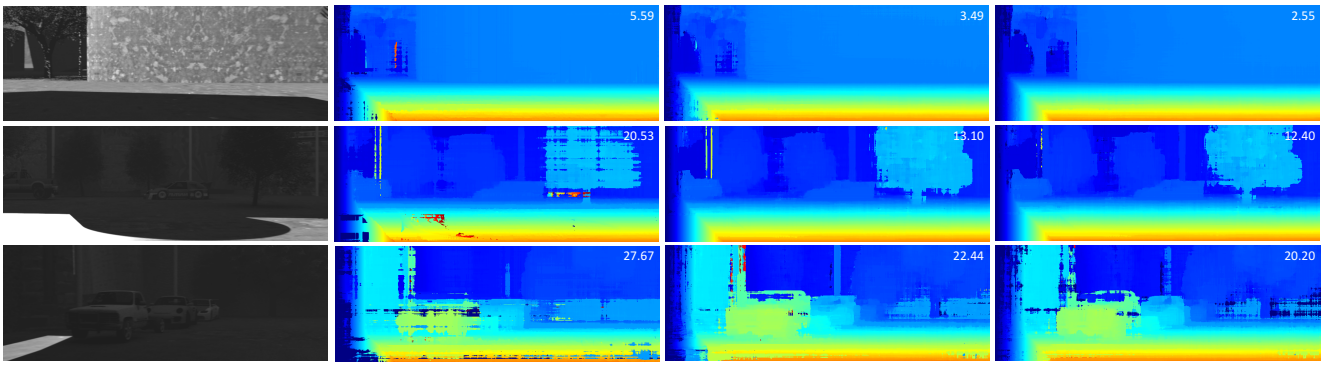


Figure 10. Disparity maps of test images (scene_backwards) on the synthetic dataset. An original image, hand tuned [38], standard SGM-Net (All), and signed SGM-Net (All) results are shown from left to right. Numbers of top right on the disparity map indicate Out-Noc error.

formation on two types of matching function (ZNCC which window size is 5×5 pixels and MC-CNN [38]). The value v of ZNCC is converted to $(1 - v)/2$ in order to fit the energy function. As an error criterion, we evaluate the percentage of erroneous pixels on non-occluded areas with an error threshold of 3 pixels. We didn't apply any post processing to the disparity maps. Table 1 shows the errors of each method. Bold font means the best accuracy in each group (hand tuned, standard and signed SGM-Nets). On both matching functions, signed SGM-Net gets the best, and hand tuned method [38] gets the worst. SGM-Nets are properly trained from initial random parameters. By using all cost functions (path and neighbor costs), both SGM-Nets got the best. Moreover, the positional information works well. In the following experiments, we picked up the best method from each of the group, i.e. hand tuned (Dynamic) and standard and signed SGM-Nets with all cost functions.

Figure 10 shows disparity maps of test images estimated by hand tuned [38], standard, and signed SGM-Nets. SGM-Nets are able to outperform accuracy, even though some parts of the disparity map are difficult to estimate correctly because of saturation, motion blurs, image noises, and etc. Hand tuned SGM is likely to be happen streaking artifacts that appear around a tree (middle row in Fig. 10), road and vehicles (bottom row in Fig. 10). SGM-Nets are able to mitigate them because our method estimates the penalties so as to have a margin m between true disparity and others in Eq. (4) and (9).

Training data	Test (K15 tr.) D1-all by Standard / Signed SGM-Net
Synthetic	5.09% / 5.58%
Synthetic + K12 tr.	4.49% / 4.38%

Table 2. Comparison of used training datasets.

6.2. Real images

Ground truth disparity maps are not provided for test images of KITTIs, hence we have to submit the disparity maps of the test images on the website [1]. However the number of the submission is restricted. Therefore, we use one of KITTIs training images as training for SGM-Nets and another as evaluation. MC-CNN provides its trained network for each K12 and K15. We use MC-CNN trained in corresponding training dataset. We employ the default error criterion of KITTIs and it is the same in Sec. 6.1.

First, we evaluated an advantage to use real images for training. K12 training images are used for training, and the networks are evaluated on K15 training images. As shown in Tab. 2, real images help to improve accuracy on both SGM-Nets. In the following experiments, we used both synthetic and real images for training SGM-Nets.

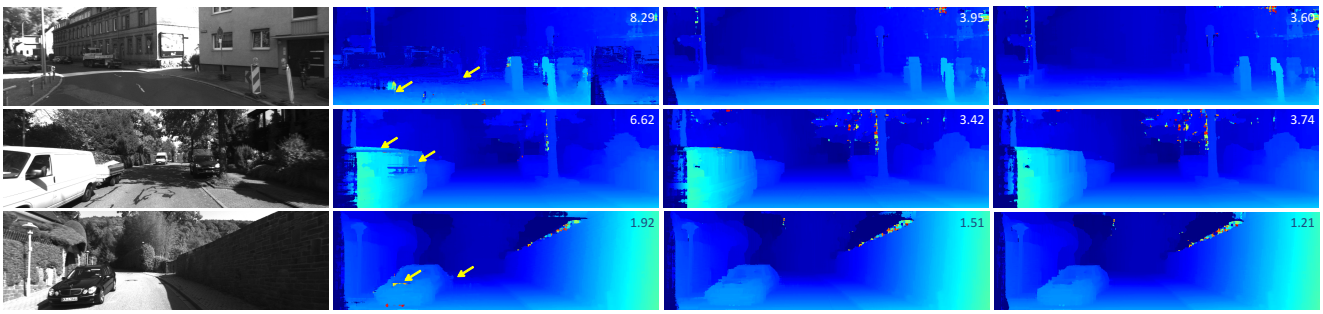


Figure 11. KITTI 2012 training images. An original image, hand tuned [38], standard SGM-Net (All), and signed SGM-Net (All) results are shown from left to right. Yellow arrows indicate noticeable errors. Numbers of top right on the disparity map indicate Out-Noc error. As there is none of ground truth on upper parts of the image, the parts are more likely to be influenced by synthetic images.

Method	Train (K12 tr.)		Test (K15 tr.) D1-all		Train (K15 tr.) D1-all		Test (K12 tr.)	
	SGM	Post proc.	SGM	Post proc.	SGM	Post proc.	SGM	Post proc.
Hand tuned [38]	3.19%	2.50%	4.72%	3.76%	3.23%	2.60%	5.18%	4.12%
Standard SGM-Net	3.10%	2.50%	4.49%	**3.63%**	3.04%	**2.49%**	4.80%	3.99%
Signed SGM-Net	**2.99%**	**2.47%**	**4.38%**	3.69%	**3.00%**	2.54%	**4.76%**	**3.96%**

Table 3. Out-Noc error on KITTI datasets. "SGM" in this table shows errors of pure SGM results, i.e. w/o post processing.

Rank	Method	Error	Time [sec.]
1	Signed SGM-Net	**2.29%**	67*
2	Standard SGM-Net	2.33%	67*
3	PBCP [28]	2.36%	68*
4	Displets v2 [12]	2.37%	265
5	MC-CNN-acrt [38]	2.43%	67*

Table 4. Out-Noc error on KITTI 2012 testing dataset by October 18th 2016[3]. "*" means GPU computation.

Rank	Method	D1-bg	D1-fg	D1-all
1	Standard SGM-Net	**2.23%**	7.44%	**3.09%**
1	Displets v2 [12]	2.73%	4.95%	**3.09%**
3	PBCP [28]	2.27%	7.72%	3.17%
4	Signed SGM-Net	2.24%	7.94%	3.18%
5	MC-CNN-acrt [38]	2.48%	7.64%	3.33%
7	DispNetC [22]	4.11%	**3.72%**	4.05%

Table 5. Out-Noc error on KITTI 2015 testing dataset by October 18th 2016[3]. The ranking is based on D1-all error.

We employ a post process similar to [38] for refining the disparity map by SGM. As consistency check from right disparity image is needed in the process, SGM-Nets with respect to the right image were also trained. We modified the process as following. Firstly, the bilateral filtering was changed in order to preserve object borders more strongly. Secondly, we added speckle filtering to remove small blobs consisting of outliers.

Figure 11 shows estimated dense disparity maps. We employed the penalty parameters provided by the authors of [38] for the hand tuned method. Erroneous pixels on objects such as road and vehicle are correctly estimated by SGM-Nets. Table 3 shows overall accuracy in the settings of training on K12 and testing on K15 and vice versa. As you can see, SGM-Nets outperform the hand tuned method in all cases. Signed SGM-Net is the most accurate on pure SGM results, however it loses accuracy after the post processing on K15 since some parts of the disparity maps were difficult to refine by the processing.

Then, we evaluated SGM-Nets with the test images on the website. Table 4 and 5 show estimated error and absolute ranking on K12 and K15, respectively. Signed and standard SGM-Net got the 1st rank on K12 and K15, respectively. Annotated density of foreground (vehicle) is much higher than that of background on K15. Additionally, the foregrounds which have transparent region such as windshield and reflected region are hard to be predicted to fit vehicles shape. They make an advantage for the methods which use object knowledge explicitly such as Displets [12]. However, our method achieved the same accuracy on the overall criterion (D1-all) without the prior knowledge.

Note that the most of computation time is consumed by stereo correspondence. SGM-Nets take only 0.02 seconds on the GPU.

7. Conclusion

In this paper, we proposed SGM-Nets which provide learned penalties for SGM. We employed a simple SGM-Net architecture, and there might be better architectures. And we have options that are involving some other information such as correspondence confidence [28] and semantic labels [8] for more accuracy. Moreover, when someone will develop much better matching functions in the future, we will be able to leverage them in our SGM-Nets.

Acknowledgement We thank Ian Cherabier and Dr. Torsten Sattler at Pollefeys' lab for comments that greatly improved the paper.

[3] At the time of CVPR submission & Excluding anonymous.

References

[1] The KITTI vision benchmark suite. http://www.cvlibs.net/datasets/kitti/index.php. 1, 7

[2] X. Chen and C. Lawrence Zitnick. Mind's Eye: A recurrent visual representation for image caption generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1

[3] Z. Chen, X. Sun, and L. Wang. A deep visual correspondence embedding model for stereo matching costs. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 1, 2

[4] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011. 6

[5] Djork-Arne, Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (ELUs). In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016. 6

[6] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 1

[7] A. Drory, C. Haubold, S. Avidan, and F. A. Hamprecht. Semi-global matching: A principled derivation in terms of message passing. In *Proceedings of the German Conference on Pattern Recognition(GCPR)*, pages 43–53, 2014. 2

[8] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, 2013. 8

[9] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[10] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2, 4, 6

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1

[12] F. Guney and A. Geiger. Displets: Resolving stereo ambiguities using object knowledge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 8

[13] S. Helmer and D. Lowe. Using stereo for object recognition. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3121–3127, 2010. 1

[14] D. Hernandez-Juarez, A. Chacón, A. Espinosa, D. Vázquez, J. C. Moure, and A. M. López. Embedded real-time stereo estimation via semi-global matching on the GPU. In *Proceedings of the International Conference on Computational Science*, pages 143–153, 2016. 1

[15] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, Feb. 2008. 1, 2, 6

[16] D. Honegger, H. Oleynikova, and M. Pollefeys. Real-time and low latency embedded computer vision hardware based on a combination of FPGA and mobile CPU. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 4930–4935, Sept 2014. 1

[17] Itseez. Open source computer vision library. https://github.com/itseez/opencv, 2015. 2, 6

[18] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015. 6

[19] A. Kirillov, D. Schlesinger, S. Zheng, B. Savchynskyy, P. H. Torr, and C. Rother. Joint training of generic CNN-CRF models with stochastic optimization. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2016. 2

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012. 1

[21] W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2

[22] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. 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 (CVPR)*, 2016. 1, 2, 6, 8

[23] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2, 6

[24] H. Oleynikova, D. Honegger, and M. Pollefeys. Reactive avoidance using embedded stereo vision for MAV flight. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 50–56, 2015. 1

[25] C. J. Pal, J. J. Weinman, L. C. Tran, and D. Scharstein. On learning conditional random fields for stereo. *International Journal of Computer Vision (IJCV)*, 99(3):319–337, 2012. 2

[26] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 102–118. Springer International Publishing, 2016. 1

[27] A. Seki and M. Okutomi. Robust obstacle detection in general road environment based on road extraction and pose estimation. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, 2006. 1

[28] A. Seki and M. Pollefeys. Patch based confidence prediction for dense disparity map. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016. 2, 8

[29] A. Seki, O. J. Woodford, S. Ito, B. Stenger, M. Hatakeyama, and J. Shimamura. Reconstructing fukushima: A case study. In *Proceedings of the International Conference on 3D Vision*, pages 681–688, 2014. 1

[30] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptor. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

[31] J. Sun, N.-N. Zheng, and H.-Y. Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 787–800, 2003. 1

[32] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag New York, Inc., 1st edition, 2010. 1

[33] T. Taniai, Y. Matsushita, and T. Naemura. Graph cut based continuous stereo matching using locally shared labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1613–1620, 2014. 1

[34] Z. Wang, S. Zhu, Y. Li, and Z. Cui. Convolutional neural network based deep conditional random fields for stereo matching. *Journal of Visual Communication and Image Representation*, 40:739–750, October 2016. 2

[35] O. Woodford, P. H.S. Torr, I. Reid, and A. Fitzgibbon. Global stereo reconstruction under second-order smoothness priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2115–2128, 2009. 1

[36] Q. Yang. Stereo matching using tree filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(4):834–846, 2015. 1

[37] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 151–158. Springer-Verlag New York, Inc., 1994. 2

[38] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17:1–32, 2016. 1, 2, 6, 7, 8

[39] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 818–833, 2014. 1

[40] K. Zhang, J. Lu, and G. Lafruit. Cross-based local stereo matching using orthogonal integral images. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(7):1073–1079, 2009. 1

[41] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

Product names mentioned in this paper are trademarks of their respective companies.