Neurocomputing 267 (2017) 195–200

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

# The Euclidean embedding learning based on convolutional neural network for stereo matching

Menglong Yang[a,*], Yiguang Liu[b], Zhisheng You[b]

[a] School of Aeronautics and Astronautics, Sichuan University, Chengdu 610064, PR China
[b] School of Computer Science and Engineering, Sichuan University, Chengdu 610064, PR China

ARTICLE INFO

*Article history:*
Received 3 May 2016
Revised 3 February 2017
Accepted 2 June 2017
Available online 8 June 2017

Communicated by Sanqing Hu

*Keywords:*
Stereo matching
Convolutional neural network
Semiglobal Matching

ABSTRACT

Stereo matching is one of the most important and fundamental topics in computer vision. The calculation of matching cost plays a very important role for stereo matching algorithms. The stereo matching algorithm proposed by Zbontar and LeCun focusing on the training of the matching cost has showed the good performance of the convolutional neural network. Unfortunately, computing a convolutional neural network for matching cost is computationally very expensive. This paper proposes a method based on learning a Euclidean embedding using a convolutional neural network with a triplet-based loss function, where the matching cost is directly computed by the squared L2 distances between two vectors in the embedding space. The cost is refined by Semiglobal Matching with an adaptive smoothness constraint based on multi-scale segmentations. The proposed method has a comparable performance with the state-of-the-art algorithms, and it overcomes a problem of heavy computation. The proposed method takes only about 5 s for predicting a single image pair, where the computing of convolutional neural networks needs less than 2 s with CPU, that is much faster than the algorithm by Zbontar and LeCun where the computing of convolutional neural network takes 67 s with GPU.



## 1. Introduction

As one of the most fundamental and important topics in computer vision, stereo vision attracts many researchers and has been widely researched since there was the publicly available performance testing such as the Middlebury [1] and KITTI stereo benchmark [2], which allow researchers to compare their algorithms against all the state-of-the-art algorithms.

Different from the feature matching [3,4], which matches sparse feature points in two images, the stereo matching can densely match the pixels. Scharstein and Szeliski [1] summarized four steps for a typical stereo algorithm, i.e., matching cost computation, cost aggregation, optimization, and disparity refinement, respectively. Local stereo methods focused on the first two steps [5,6] often fail in challenging scenarios of weakly textured, saturated or reflective regions. Many global methods based on the research on steps (3) and (4) are thus researched and perform well on Middlebury benchmark, such as graph cuts [7–9] and belief propagation (BP) [10–14]. The stereo is achieved in these global algorithms, essentially, by solving a Markov Random Field (MRF) model, including the assumptions of photo-consistency and smoothness. However, stereo for the complex outdoor scenarios is still a challenging issue [15].

Recently, Zbontar and LeCun [16] trained a convolutional neural network (CNN) to compute the matching cost. Their method outperformed on KITTI benchmark but need about 67 s for calculating a single image pair, where the majority of time during prediction is spent in the forward pass of the CNN with a Nvidia GeForce GTX Titan GPU.

Our method is based on learning a Euclidean embedding using a convolutional neural network. Different with [16] where the matching cost is computed by five full connected layers, the network in this work is trained such that the squared L2 distances in the embedding space directly correspond to matching cost between a pair of patches. Thus our method takes only about 5 s for predicting a single image pair, where the computing of convolutional neural networks only needs 1.2 s with a Nvidia GeForce GTX 880 GPU or 2 s with a Intel i7 CPU.

### 1.1. Related work

Many learning-based stereo algorithms [17–23] have been proposed since the introduction of large stereo datasets [21,24].

A class of training methods aim to compute or refine matching cost besies Zbontar and LeCun's convolutional neural network

* Corresponding author.
*E-mail address:* steinbeck@163.com (M. Yang).

http://dx.doi.org/10.1016/j.neucom.2017.06.007
0925-2312/

[16]. For example, Kong and Tao [17,18] initialized the matching cost with sum of squared distances, and using a trained model to predict the probability that whether the initial disparity is correct. The initial matching cost was refined based on the predicted probabilities. Some other works [22,23] focused on estimating the confidence of the computed matching cost.

Similarly to our training strategy, Schroff et al. [25] trained a deep convolutional neural network called FaceNet for face recognition and clustering. They used triplet loss to separate the positive pair from the negative by a distance margin. Similar work proposed by Liu et al. [26] used similar framework for face recognition and achieved the state-of-the-art accuracy of Labeled Faces in the Wild (LFW)[1] database. The main difference is that FaceNet only trained one network, whereas this work trained two networks.

Except the matching cost, smoothness constraint is an important factor as well, where encouraging self-similar pixels to be assigned to the same label is an effective strategy. A typical way of taking advantage of self-similarity is performing a color segmentation on the image and regarding the pixels within each segment as self-similar pixels. Some segmentation-based algorithms use a hard constraint that the pixels within a single segment must be assigned to the same plane [10,27,28], and some others use a soft constraint that the neighboring two pixels sharing the same segment are only encouraged to lie on the same plane [29,30]. The scale of segmentation is important for these algorithms. A large scale of segmentation over-constrains the small objects and a small scale of segmentation is hard to constrain the large objects. Bleyer et al. [31] use a soft segmentation term to encourage the stereo result to consistent with a precomputed segmentation, but optimizing these higher-order cliques is difficult and time consuming.

### 1.2. Contributions

A typical stereo matching algorithm involves the matching cost and smoothness constraint. Both aspects are studied in this paper, which are in correspondence to two main contributions of this work that are summarized as follows, respectively.

Firstly, we proposed a method of learning a Euclidean embedding using a convolutional neural network. Different with [16] where the matching cost is computed by five full connected layers, the network in this work is trained such that the squared L2 distances in the embedding space directly correspond to matching cost between a pair of patches. Thus our method takes less than 1 s for computing the matching cost for a single image pair.

Secondly, as mentioned before, the positive effect of image segmentation has been shown in many stereo matching algorithms. However in the methods of hard constraint, an appropriate scale of segmentation is hard to find for different scenarios. In the methods of soft constraint, optimizing the higher-order cliques is difficult and time consuming. To make a tradeoff, we adopt an adaptive smoothness penalty for each pair of neighboring pixels that depends on multi-scale segmentations. If two neighboring pixels are always in same segment with different segmentation scales, they are naturally expected to be assigned as a same label.

## 2. Matching cost

Following the definition of [16], the first step of a typical stereo algorithm is computing the matching cost $C(\mathbf{p}, d)$ at each position $p$ for all disparities $d$. For example, the sum of absolute differences

$$C(\mathbf{p}, d) = \sum_{q \in \mathcal{N}_\mathbf{p}} | I^L(\mathbf{q}) - I^R(\mathbf{qd}) | \tag{1}$$

aims at measuring the cost associated with matching a patch from the left image whose center is located at $\mathbf{q}$ in the original image, with a patch from the right image, whose center is located at $\mathbf{qd}$ in the original image. Here $\mathcal{N}_\mathbf{p}$ is the set of locations within a fixed rectangular window centered at $\mathbf{p}$. $I^L(\mathbf{q})$ and $I^R(\mathbf{q})$ are image intensities at position $\mathbf{q}$ of the left and right image, respectively. The $\mathbf{d}$ means that if the center point of left image is $\mathbf{q} = (x, y)$, then the center point of right image is $\mathbf{qd} = (x - d, y)$.

This work aims at solving the matching problem by a supervised learning approach through amount of samples. Different with [16], where a convolutional neural network was used to predict the degree of two image patches matching, we use convolutional neural networks to embed the image patches into Euclidean space, and matching cost is simply corresponded to the Euclidean distance between two features.

Our method uses two deep convolutional neural networks, handling the left and right image patches, respectively. As Fig. 1 shows, the blue network processes the left image patches, and the red network processes the right image patches. We use a triplet-based loss function to train the networks, where the triplets consist of two matching image patches and a mismatching image patches. The loss aims to separate the positive pair from the negative by a distance margin.

### 2.1. Creating the training dataset

As Fig. 1 shows, a training sample includes three image patches, one from the left and the other two from the right image, which can be denoted as

$$< \mathcal{P}_{15\times15}(\mathbf{p}), \mathcal{P}'^{p}_{15\times15}(\mathbf{q}), \mathcal{P}'^{n}_{15\times15}(\mathbf{r}) > \tag{2}$$

where $\mathcal{P}_{15\times15}(\mathbf{p})$ is a $15 \times 15$ patch from the left image, whose center is located at $\mathbf{p}$ in the original image. Similarly, $\mathcal{P}'^{p}_{15\times15}(\mathbf{q})$ and $\mathcal{P}'^{n}_{15\times15}(\mathbf{r})$ are the positive and negative sample, which are $15 \times 15$ patches from the right image, centered at $\mathbf{q}$ and $\mathbf{r}$ in the original image, respectively. For each location $\mathbf{p} = (x, y)$ where the true disparity $d$ is known, we extract one positive $\mathcal{P}'^{p}_{15\times15}(\mathbf{q})$ and one negative sample $\mathcal{P}'^{n}_{15\times15}(\mathbf{r})$ from the right image.

Similarly to [16], a positive sample is obtained by setting

$$\mathbf{q} = (x - d + o_{pos}, y) \tag{3}$$

where $o_{pos}$ is chosen randomly from the set $\{-P_{hi}, \ldots, P_{hi}\}$.

Similarly, a negative sample is extracted by

$$\mathbf{q} = (x - d + o_{neg}, y) \tag{4}$$

where $o_{neg}$ is chosen randomly from the set $\{-N_{hi}, \ldots, -N_{lo}, N_{lo}, \ldots, N_{hi}\}$. $P_{hi}$, $N_{lo}$, $N_{hi}$ are hyperparameters of the method.

### 2.2. Triplet loss

As mentioned above, the triplet loss aims at shortening the L2 distance of the matching samples and enlarging it between mismatching samples. Suppose that the embedding is represented by $f(\mathcal{P}), f'(\mathcal{P}') \in \mathcal{R}_d$, corresponding to the left network and right network, respectively. They embed the image patches $\mathcal{P}$ and $\mathcal{P}'$ into a $d$-dimensional Euclidean space. Here we want to ensure that an left image patch $\mathcal{P}$ of a specific position is closer to its matching right image patch $\mathcal{P}'^{p}$ than it is to any mismatching right image patch $\mathcal{P}'^{n}$. This can be mathematically expressed as

$$\|f(\mathcal{P}) - f'(\mathcal{P}'^{p})\|_2^2 + m < \|f(\mathcal{P}) - f'(\mathcal{P}'^{n})\|_2^2, \quad \forall(\mathcal{P}, \mathcal{P}'^{p}, \mathcal{P}'^{n}) \in \Gamma \tag{5}$$

where $m$ is a margin that is enforced between matching and mismatching pairs. $\Gamma$ is the set of all possible triplets in the training set and has cardinality $N$.

[1] http://vis-www.cs.umass.edu/lfw/.