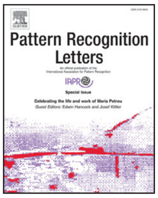
Pattern Recognition Letters 120 (2019) 75–81

Contents lists available at ScienceDirect

Pattern Recognition Letters

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



Widening siamese architectures for stereo matching

Patrick Brandao*, Evangelos Mazomenos, Danail Stoyanov

Wellcome/EPSRC Centre for Interventional and Surgical Sciences (WEISS) and Department of Computer Science, University College London, Charles Bell House, Foley Street, London W1W 7TS, UK

ARTICLE INFO

Article history:
Received 3 November 2017
Available online 4 December 2018

MSC:
62M45
62P30

Keywords:
Stereo matching
Convolutional neural network
Disparity
Computer vision

ABSTRACT

Computational stereo is one of the classical problems in computer vision. Numerous algorithms and solutions have been reported in recent years focusing on developing methods for computing similarity, aggregating it to obtain spatial support and finally optimizing an energy function to find the final disparity. In this paper, we focus on the feature extraction component of stereo matching architecture and we show standard CNNs operation can be used to improve the quality of the features used to find point correspondences. Furthermore, we use a simple space aggregation that hugely simplifies the correlation learning problem, allowing us to better evaluate the quality of the features extracted. Our results on benchmark data are compelling and show promising potential even without refining the solution.

1. Introduction

Computational stereo is one of the classical problems in computer vision systems whereby two cameras placed at different viewpoints can be used to extract 3D information by analyzing the relative position of the objects in the two perspectives of the scene. Finding relative displacements between image pairs from stereo cameras is usually called stereo matching [2,15]. By using the fundamental constrains in the two-view geometry of two perspective cameras, it is possible to reduce the stereo matching problem to a 1D search space in horizontally rectified images. Despite the reduced search space, accurately finding stereo correspondences in real world images is still very challenging because occlusions, reflective surfaces, repetitive patterns, textureless or low detail regions that can affect the similarity metric and underpins the search.

Recently, since the first winning entry in the ImageNet Large Scale Visual Recognition Challenge, deep learning has been at the forefront of most computer vision breakthroughs [13]. Convolutional neural networks (CNNs) are able to learn very complex nonlinear representations from raw visual data, creating effective and versatile models for complex problems. CNNs are now widely used across different vision problems and also in a vast range of applications, such as robotics and medical endoscopic imaging. Deep learning models have also recently been applied to stereo matching and are now among the most accurate methods reported on the public common evaluation datasets [4,9,14].

One of first successful uses of deep learning for stereo matching treats the problem as a binary classification [17], where different CNNs are trained to recognize two input patches centered around corresponding pixels. However, because each pixel is processed individually, and no spatial constrains are imposed in the decision, the resulting disparity map can be quite noisy. To mitigate noise, extensive post processing steps are used to smooth the result using hand-crafted regularization functions. Several improvements have been reported since then, usually by stacking extra convolution layers after the feature extraction, allowing the CNNs to learn their own spatial regularization. The current top stereo method ranked on the KITTI benchmark dataset also focuses on context and consistency by using a very deep end-to-end learned architecture with 3-D convolutions that is able to infer disparity maps capable of beating any hand-crafted regularized method [5].

While spatial consistency is essential for good stereo matching, there has been limited focus on the quality of the high-level representation learned to match corresponding points. Several methods proposed different architectures, correlation operations or regularization approaches but the majority of CNN stereo methods do not present any major discussion about the siamese architecture that it uses. The main objective of this work is to fill this gap and study the importance of the representations learned by a siamese architecture. We take a step back from deep complex CNN architectures and focus on the type of features that are used to find correspondences. We propose the use of pooling and deconvolution operations in the siamese architecture that allows the

* Corresponding author.
E-mail addresses: patrick.brandao.15@ucl.ac.uk (P. Brandao), danail.stoyanov@ucl.ac.uk (D. Stoyanov).

https://doi.org/10.1016/j.patrec.2018.12.002

extraction of features with a wider receptive field around the target pixels. The intuition is that, a wider context view allows the feature extraction of more visual cues, allowing better point correspondence. Furthermore, we study the effect of a simple feature space transformation that significantly simplifies the learning problem, allowing the CNNs to learn end-to-end correlation with a very shallow architecture. Our main objective is to show that improvements can be achieved simply by enhancing the way stereo features are extracted and aggregated. Because siamese architectures are part of most matching CNNs available, this work can easily be combined with more complex approaches (hand-crafted or deep learned) presented in the literature.

2. Related work

A range of approaches have been proposed to solve the stereo matching problem in the last decade. For the sake of brevity, we will focus on the work that exploits deep learning as a viable way to find point correspondences in image pairs [2,15].

The introduction of large scale, high resolution datasets, such as KITTI [4,9] and Middlebury [14], has opened the opportunity for the use of learning approaches in stereo matching. As stated before, Zbontar and LeCun [17] used a siamese CNN to binary classify matching or non-matching pairs of points. The method required an extensive post processing step, where edge and texture information were used as smoothness constrains.

More recently, Luo et al. [8] expanded on Zbontar's work and proposed a way to obtain disparity values for all possible displacements without manually pairing patch candidates. In other words, a wider image is passed though one of the branches of the siamese architecture and the computed features are correlated with the ones extracted from the target patch. This allows the computation of matching costs for all disparities with one-pass of the CNN. This work also shows that the inner product is an effective way to compute feature correlation. Again, because inference for each pixel is made independently, hand-crafted feature regularization is used to smooth the results.

Currently, the top performing stereo methods in the KITTI datasets [4,9] focus on end-to-end network learning with spatial regularization and do not use any type of hand-crafted post processing. Shaked and Wolf [16] employ a second network that is trained to smooth the matching cost obtained by a deep residual architecture. Kendall et al. use 37 layered network with multi-scale 3D convolutions to learn how to match a block of concatenated features from both images. Pang et al. [10] tackled the matching problem in two stages: first, a tweaked version of DispNet is used to estimate disparities with more detail and then a second network is used to rectify the results of the first stage. Knöbelreiter et al. [7] also achieved excellent performance by combining CNNs and conditional random fields into a hybrid model for stereo estimation. Despite the huge difference in architectures and training methodology, all these methods start roughly the same way, with a siamese architecture that acts as a feature descriptor for the stereo image pair. Most recent work chooses to focus on the spatial regularization rather than the feature extraction step. We argue that significant improvements can be achieved by simply increasing the amount of context that is extracted by the siamese architecture.

The work presented here is most similar to the one developed by Luo et al. [8] but with two major contributions. First, we show that the loss of the detail from pooling operations can be compensated with deconvolution operations if these are applied in the feature space, before computing correlation. This allows to hugely increase the global receptive field of the feature extractors, resulting in a more robust matching even before spatial regularization. Second, we show that a simple feature aggregation can be used to simplify the learning problem, resulting in effective, more easily learned, data driven correlation metric. To reiterate, we are studying the feature extraction step and how much it can influence correspondences by itself. Our aim is not to beat the current state-of-the-art for full stereo matching pipelines. Our contribution provides an effective stereo matching network that can easily be further improved by plugging it to most current CNN stereo matching models.

3. Methodology

Typically, stereo methods use a similarity function between handcrafted representations of small patches around the pixels [2]. Alternatively, CNNs can be used to learn complex, high dimensional feature extractors that allow a more robust patch comparison [17].

Some of the most accurate stereo algorithms proposed in recent years employ CNNs to score the patch similarity measure [5,8,11,16,17]. Even though these methods proceed with different approaches, every model starts with a siamese architecture that processes the left and the right images. While subsequent layers may allow more complex correlation inference or spatial regularization of the cost volume, the matching is still in essence based on the features extracted by the siamese branches. As a consequence, the architecture of the siamese CNN plays a crucial role in the quality of the stereo matching, much like the role of a traditional low level vision similarity metric. We therefore focus on enhancing the underlying siamese network in order to improve performance.

3.1. Siamese network architecture

We construct our network by layering sequential blocks of 2D convolutions, batch normalization and a rectifier linear unit (ReLU). Just like most architectures, we use layers with 64 neurons of 3×3 convolutions and the parameters between branches are shared. The last layers are added without batch normalization and ReLU operations.

Generally speaking, wider patches allow the extraction of more visual cues and help matching, especially in textureless regions or areas of aperture problems. The area around the target pixel that is considered in the matching process depends on the global receptive field of the CNN architecture. If we denote the input of the pth layer indexed by the coordinates i, j as $x^p(i, j)$, then a network with n layers will output $y(i, j) = x^n(i, j)$. Mathematically, we can define the global receptive field as the range of pixels in x^0 that affects each $y(i, j)$. Intuitively, the global receptive field is the size of the region that a CNN uses towards making a single prediction.

More convolution layers and bigger filters allow small increases in the global receptive field but cause an exponential increase in computation time and memory requirements. A common practice in classification CNNs is the use of strided pooling to downsample feature maps withing the network, allowing for much wider global receptive fields [11]. Pooling operations have also been reported to provide translation invariance to CNN models [11]. However, the properties that make pooling useful in classification tasks are not desirable for stereo matching, so most stereo algorithms avoid this operation. The loss of detail from feature downsampling makes it harder to recognize very small differences, something crucial for pixel-level matching. We address this problem by using transpose convolution (deconvolution) operations.

Deconvolution operations allow CNNs to learn filters capable of upsampling feature maps. The operation is especially useful in pixel-level applications, such as semantic segmentation or generative networks. For example, for optical flow, where the matching search space is bidimensional, the FlowNet [3] sequentially downsamples the features maps with pooling operations and uses a se-

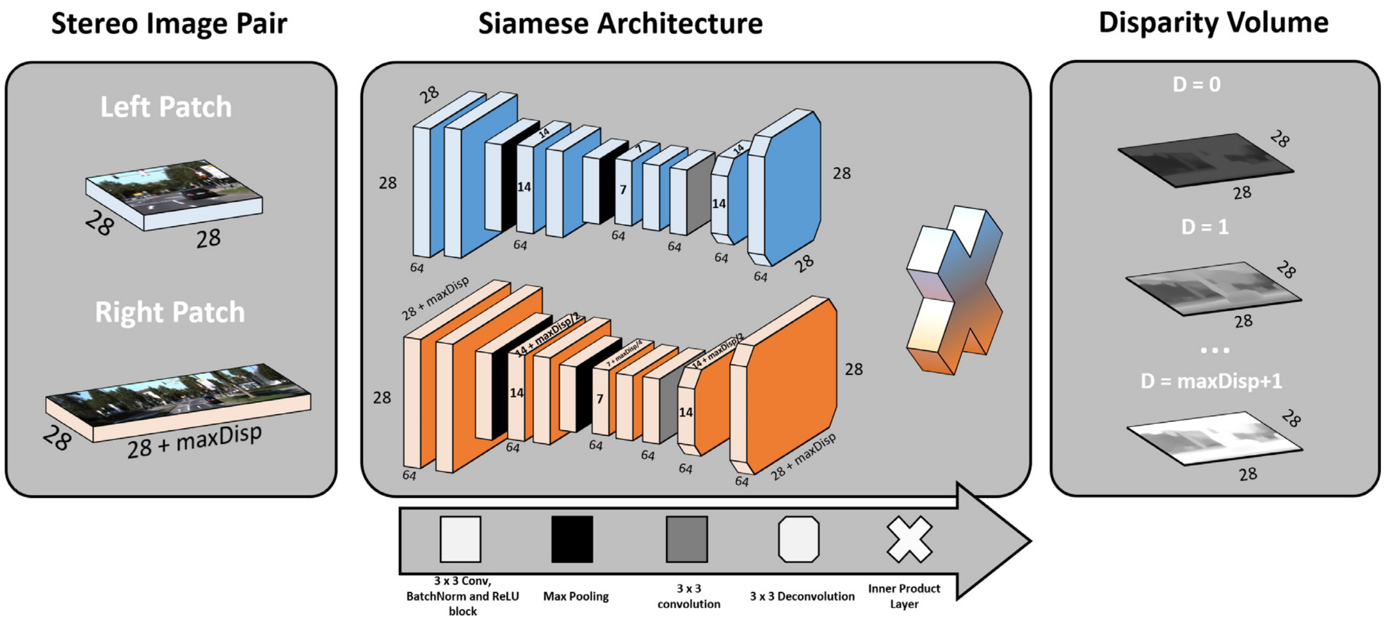


Fig. 1. Representation of our 7 layered stereo matching CNN. Patches extracted from the left and right stereo images are processed in the blue and orange branches, respectively. During training, the width of the right patch depends of the max disparity (D) considered. After feature extraction with the siamese architecture, the features are aggregated according to their relative displacement. The correlation between features for each disparity is computed by a simple two layer correlation architecture. The final disparity volume represents a correlation value of each possible integer disparity between zero and D for every left patch pixel.

ries of deconvolutions to obtain a dense prediction map. Unlike FlowNet, we argue that it is easier to match upsampled features than upsampling matching scores. Because of this we choose to implement deconvolution layers before computing any correlation metric. Just like represented in Fig. 1, we implement the same amount of 2 strided 3×3 deconvolutions as the number of max poolings within the CNN. This creates a dense feature space that can be used for computation of a correlation score for every possible disparity level.

3.2. Correlation layer

Several stereo matching CNNs use the inner product as a correlation metric between features vectors extracted from the siamese branches [8,11,17]. The operation is computationally efficient, fast and differentiable, which allows backpropagation during training. In these cases, the CNN learns feature extractors that maximize the inner product between two corresponding points. While this provides a fast and effective way to compute correlation, it would be preferable to allow the network to learn a correlation that best fits the stereo data. Note that the inner product only measures one direction/component of similarity between vectors. Whereas the network could learn more complex relationships/metrics.

Recent methods choose to concatenate the output from the siamese network along the feature dimension and follow it with more convolution layers [11,16,17]. To a certain extent, this allows the CNN to learn how to correlate matching points, but the maximum disparity that the network is able to find is intrinsically related to the global receptive field of the layers stacked after the siamese portion of the CNN.

Let's consider the case where we want to find the disparity map for a left stereo image I_l with $W \times H$ dimensions. Considering D, the maximum disparity possible between the stereo pair, correlation needs to be computed with all pixels within a $D+1$ range in the right stereo image I_r. By using a siamese network with a θ dimensional output its possible to extract two feature vectors with $W \times H \times \theta$ dimensions. To learn how to match pixels for $D+1$ possible disparities from the concatenated volume, the network needs to process 2θ values in its third dimension and to account for a range of $D+1$ pixels in the input second dimension. In other words, the correlation layers would need to start with 2θ neurons, and their global receptive field would need to be equal or superior to $D+1$ in the image width dimension. Using the common approach where we stack n layers of $w \times w$ convolution blocks the global receptive field of a network is equal to $n \times (w-1)+1$. In the KITTI dataset [4], for example, where $D = 256$, it would take at least 128 layers of 3×3 convolutions for a network to have a global receptive field wide enough to match 256 pixels apart without downsampling the feature space. This is not only challenging from a computational point of view but it greatly complicates the learning process. Beyond learning how to correlate features of matching points, the model would also need to correspond feature positions with the intended disparity. We use a feature space transformation that greatly simplifies the learning problem of a non-linear correlation metric through convolutional layers, needing as little as two convolution layers to compute a disparity map for any size D.

Defining the θ-dimensional feature vectors computed from I_l and I_r as the ψ_l and ψ_r, respectively, we construct a new feature space Ψ as:

$$\Psi(i, j) = \mid [\mid \psi_l(i, j)\psi_r(i, j-d) \mid], \forall d \in \mathbb{N}_0 \mid 0 \leqslant d \leqslant D \mid \quad (1)$$

where $|.|$ represents a concatenation operation. Note that we are still concatenating vectors along the feature dimension, but we replicate the left features and pair them with right features of every possible disparity. The new feature space Ψ has the dimensions $WH \times D+1 \times 2\theta$ where, for all (i, j) pixels, there is a paired 2θ-dimensional feature vector for all $D+1$ possible disparities. This simple transformation radically changes what kind of information convolution filters receive. Let's consider applying a single 1×1 convolution layer that outputs a single value from a 2θ dimensional input to the new feature space Ψ. Note that a single value would be computed for $D+1$ disparities for all (i, j) pixels, using only the corresponding right and left feature pairing as input. This way, the correlation layer only needs to learn how to correlate two concatenated θ-dimensional vectors, inde-

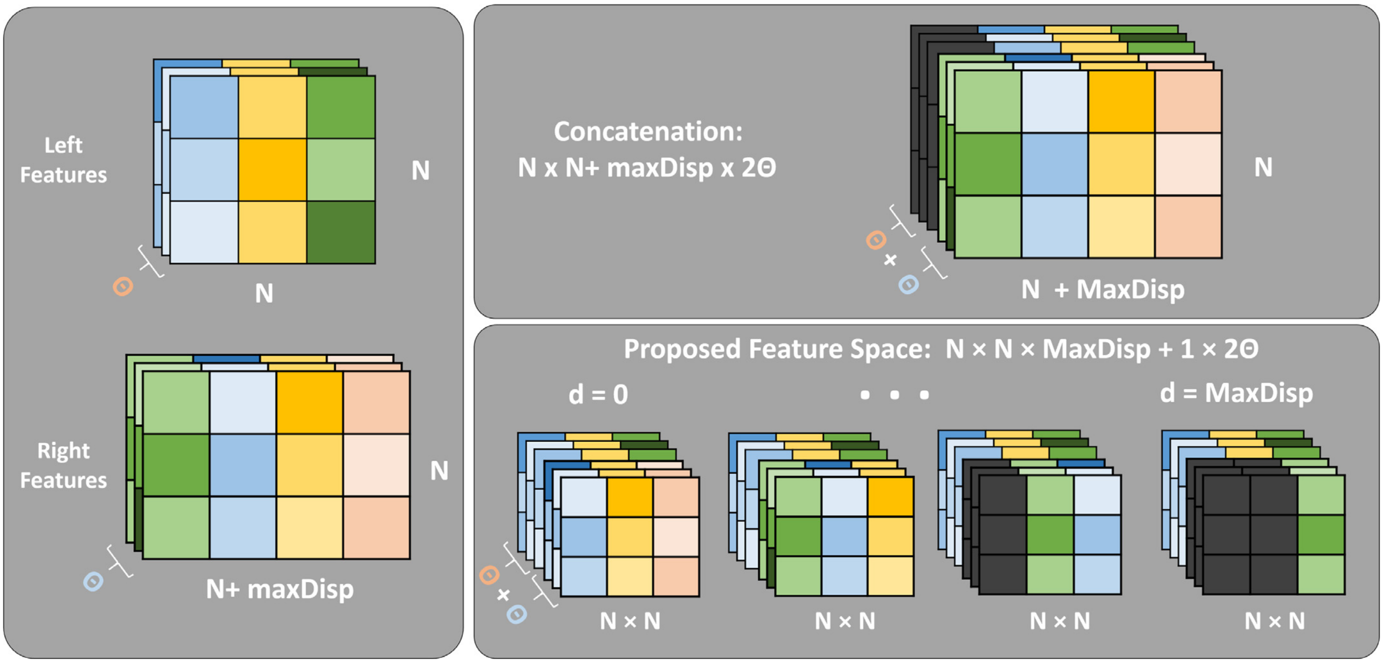


Fig. 2. Comparison between standard feature concatenation and a built feature space. The left and right Θ-dimensional features are computed by the siamese architecture. Similar color squares represent point correspondences between the stereo image pair. Differences in tone are just meant to represent small variations between both images. Black squares represent zero padding.

pendently of their original position, considerably simplifying the learning problem. This layer would output a $WH \times D + 1 \times 1$ map that can be easily transformed to the intended disparity volume with a $W \times H \times D + 1$ shape. Beyond this, in this feature space, filters of size $1 \times z$ allow the network to learn a correlation metric that accounts for z neighbor disparity pairs, creating the opportunity for a more robust disparity correlation. Finally, because the filters learned during training always correlate 2θ-dimensional feature pairs, Ψ can be rebuilt for a variable number of max disparities without needing to retrain the model.

The idea of aligning features is similar to the one presented by Kendall et al. [5]. However, this is followed with a second big 3D network that is responsible to learn not only a correlation metric, but to regularize the disparity map. While this is an obvious advantage for the global performance of a stereo matching network, it would make it harder to exclusively evaluate the quality of the features extracted. Our feature space transformation meaning that, each disparity is processed individually by the same learned correlation layer, making it that only the features learned from the stereo pair are taken into consideration.

In our experimental results, we compare the performance of the cost volumes computed with inner product and with our correlation layer. We use the simplest architecture that allows non-linear logical operations [12]. For our correlation layer, we use a single activated convolutional hidden layer with 2θ neurons and 1×3 filters, and an output convolutional layer with a singular output channel also with a 1×3 filter. A smaller filter wound not allow the correlation layer to take into account neighborhood information and bigger filters did not improve the results.

3.3. Training

We train our models with randomly extracted small patches from the left stereo image and the same coordinate patch from the right image extended by the maximum disparity under consideration. This allows to diversely sample training batches while being memory efficient. We treat each disparity value as a mutually exclusive classification problem. The values outputted from the correlation step are normalized using a softmax function and the network is trained by minimizing cross-entropy loss. All parameters are trained with stochastic gradient descent and gradients are backpropagated using the standard Adam optimization [6].

3.4. Testing

During testing, memory constrains us to compute disparity maps for high resolution images with big max displacements in a single network pass. Instead of processing subsections of the image individually, we follow the same procedure suggested by Luo et al. [8]. First, we extract the feature representation for all pixels of the stereo image pair with the siamese architecture. Then in the correlation step, the same feature values can be reused for computation of disparity maps of multiple pixels. This results in significant increases in the inference speed. The final disparity values are chosen with a winner-takes-all approach (Table 2).

4. Experimental evaluation

We train and evaluate our models using both the KITTI 2012 [4] and KITTI 2015 [9] datasets. Both are composed of rectified natural images captured by a stereo camera. KITTI 2012 consists only of static environments while moving objects are present in KITTI 2015. Just like most methods [5,8,16,17], we use the sparse available labels from non-occluded pixels for training.

We evaluate our methodology by training three different siamese architectures: S_4, S_7 and S_9, with 4, 7 and 9 convolution layers and with 1, 2 and 3 max pooling layers, respectively. We also compare all models trained with inner product and with the proposed correlation architecture. We verified no performance improvement by adding skip connections between the encoding, so we only present the results with non-skip architectures.

All parameters are randomly initialized with a normalized Gaussian distribution and input images are normalized to have zero mean and unit standard deviation. Every CNN is trained for

Table 1
Comparison of several error metrics in % of our three different siamese architectures trained with inner product (inner prod) and with our correlation architecture (learned) on the KITTI 2012 validation set.

		>2 pixel		>3 pixel		>5 pixel		Runtime (s)
Siamese CNN	Correlation	Non-Occ	All	Non-Occ	All	Non-Occ	All	
S_4	Inner prod	12.42	14.18	11.38	13.16	9.98	11.76	1.15
	Learned	11.27	13.05	10.39	12.13	9.08	10.82	5.25
S_7	Inner prod	7.57	9.45	6.72	8.61	5.64	7.53	1.15
	Learned	**6.65**	**8.23**	**5.84**	**7.58**	**4.80**	**6.48**	5.27
S_9	Inner prod	7.47	9.34	6.50	8.36	5.31	7.17	1.16
	Learned	7.57	10.29	6.59	9.05	5.34	7.80	5.28

Table 2
Comparison of several error metrics in % of our three different siamese architectures trained with inner product (inner prod) and with our correlation architecture (learned) on the KITTI 2015 validation set.

		>2 pixel		>3 pixel		>5 pixel		Runtime (s)
Siamese CNN	Correlation	Non-Occ	All	Non-Occ	All	Non-Occ	All	
S_4	Inner prod	11.19	12.68	10.01	11.50	8.57	10.05	1.15
	Learned	8.26	10.72	7.10	9.71	6.82	8.40	5.25
S_7	Inner prod	7.80	9.36	6.81	8.37	5.75	7.30	1.15
	Learned	**6.79**	**8.21**	**5.92**	**7.30**	**4.92**	**6.24**	5.27
S_9	Inner prod	6.89	8.47	6.02	7.61	5.18	6.74	1.16
	Learned	7.47	8.96	6.42	7.88	5.41	6.82	5.28

75K iterations with a $1e^{-3}$ starting learning rate. Training is done with randomly extracted patches from left image with sizes 10×10 for S_4, 28×28 for S_7 and 56×56 for S_9. We use the biggest batch size that our system allowed for each model. For CNNs trained with inner product, this translates to batches of 128, 32 and 20 for S_4, S_7 and S_9, respectively, and batches of 128, 20 and 8 for the same models trained without correlation architecture. All models were implemented in Tensorflow [1] and executed on a NVIDIA Titax-X GPU.

4.1. KITTI 2012

KITTI 2012 datasets consists of 194 image pairs for training and 195 for testing. Because no ground truth is given for the testing images, and multiple online submissions are not allowed, we evaluate our models by splitting the training data in a training and validation sets. As in the work developed by Luo et al. [8], we randomly use 160 image pairs for training and 34 for testing. Even though we do not guarantee the same split as [8], we argue that the difference in performance is big enough to prove the importance in widening the receptive field of the Siamese network, independently of the training/validation set split. Again, our main objective is to study and improve the siamese architecture that initializes most recent CNN stereo matching systems, so we do not implement an end-to-end system capable of competing with current state-of-the-art systems. The performance of our models in the validation set is shown in Table 1.

When we use the inner product for feature correlation, a direct comparison with the same depth architectures from [8] allow us to verify the effect of pooling and deconvolution layers. All our models outperform the corresponding networks proposed by Luo et al. [8], which shows the benefit of our pooling/deconvolution approach. Despite the overall increase in performance, Table 1 shows that there is a limit to the benefit of increasing the receptive field trough downsampling pooling layers. While the 2-pixel is reduced substantially from S_4 to S_7, the extra pooling layers in S_9 did not greatly decreased the matching error.

Table 1 also shows that slightly better matching was achieved by learning correlations from the transformed feature space. Matching improvements are present in S_4 and S_7 when the correlation layer is used, but a slightly worst performance is achieved in S_9. This indicates that the loss of detail from successive pooling might hinder the ability of the network to learn a good correlation function. The best results were achieved with S_7, where the receptive field is big enough for robust matching, but the lost of detail is not enough to stop the network from computing an effective correlation. Fig. 3 shows that, even without spatial regularization, our architecture is able to smoothly match low detail regions while maintaining sharp edges in cars and trees. Because the focus of our work is the evaluation of the feature extraction, we did not invest a huge amount of time in performance improvements. We used a slow naive implementation of the feature space transformation that is significantly slower than the inner product. However, this operation can still be greatly optimized with a GPU implementation.

4.2. KITTI 2015

KITTI 2015 has 200 image pairs for training and for testing. Again, just like Luo et al. [8], we randomly split the training set in 160 images for training and 40 for validation. This allows a better direct comparison with their method.

A similar analysis to the one made for KITTI 2012 is valid for the KITTI 2015 results. Bigger receptive fields allow lower matching errors for features learned with the inner-product implementation. When learning a correlation, a compromise between a wider global receptive field with less loss of detail is found in the S_7 architecture. In Fig. 4, we continue to predict big smooth disparities in low texture regions, even without any post-processing. This shows that wider global receptive fields allow a much more effective correlation computation. Furthermore, even with the downsampling operation within the networks, features capable of representing small structures like traffic signs, fences and trees can be successfully extracted. Stacking further layers should easily allow spatial regularization to be learned without significant increase in computation cost, since the concatenation and reshaping operations of the feature space transformation are the bottleneck of the method.

4.3. Comparisons with other methods

As stated before, we do not propose a full stereo pipeline for stereo matching. Our main objective is to improve a crucial part

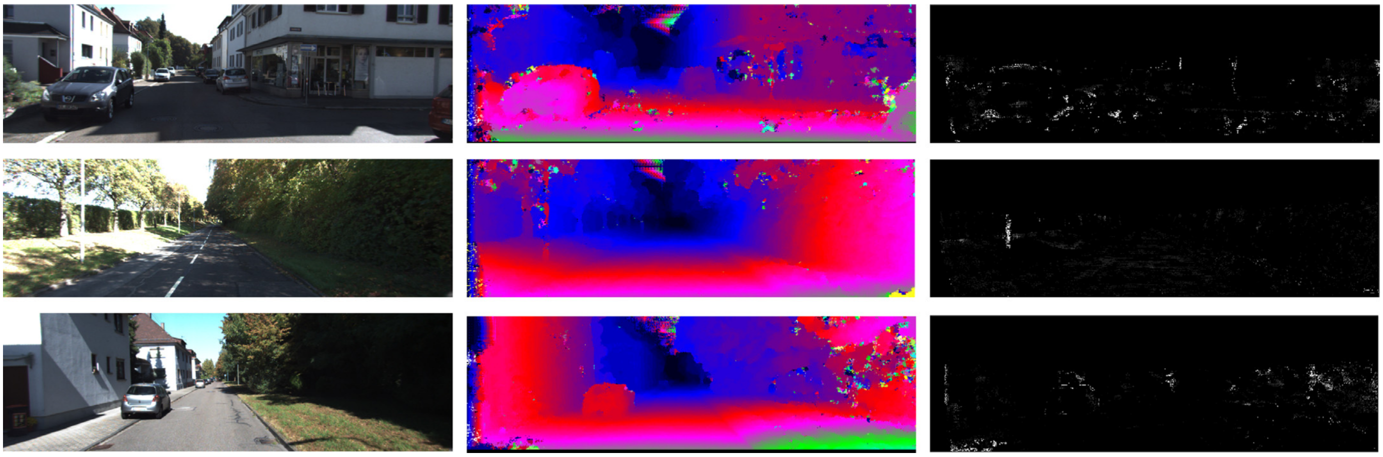


Fig. 3. Examples of non-regularized disparities (middle) and errors (right) of KITTI 2012 validation images (left) computed with the S_7 architecture and learned correlation.

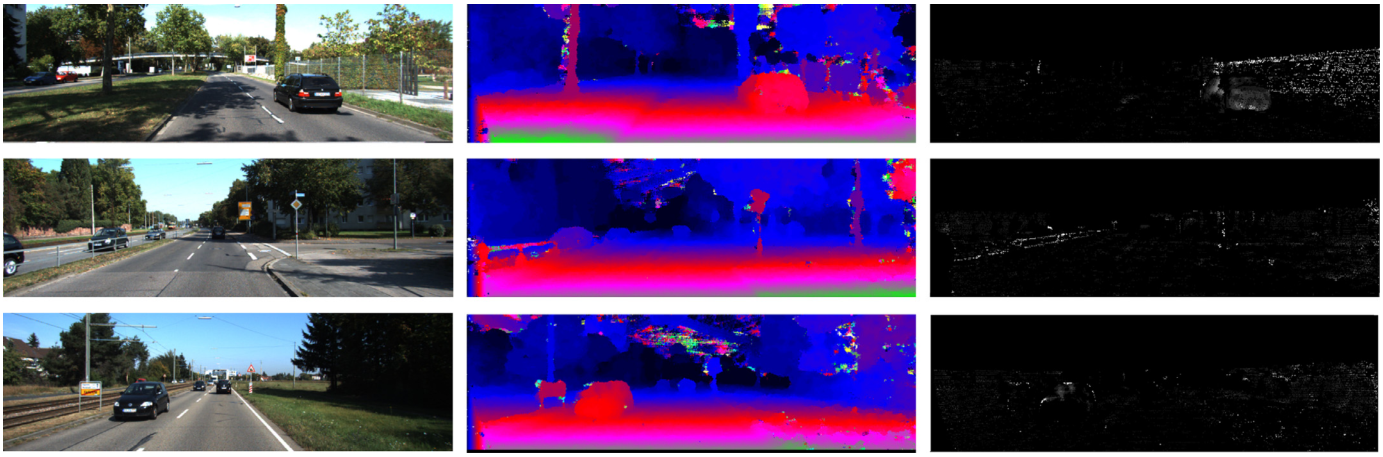


Fig. 4. Examples of non-regularized disparities (middle) and errors (right) of KITTI 2015 validation images (left) computed with the S_7 architecture and learned correlation.

Table 3
Comparison of the 2 pixel % error of different matching siamese architectures without post-processing on the 2012 and 2015 KITTI validation set.

Method	KITTI 2012		KITTI 2015	
	Non-Occ	All	Non-Occ	All
MC-CNN-acrt	15.02	16.92	15.20	16.83
MC-CNN-fast	17.72	19.56	18.47	20.04
Luo et al.	10.87	12.86	9.96	11.67
S_9 + inner product	7.57	10.29	6.89	8.47
S_7 + correlation	**6.65**	**8.23**	**6.79**	**8.21**

of most of the current CNN stereo matching models: the siamese architecture. Because of this, we compare our work with other non-spatial regularized architectures. These results are presented in Table 3.

Table 3 shows that when compared with other non regularized Siamese architectures, our wider models have a significantly lower 2-pixel error in both 2012 and 2015 KITTI datasets. Furthermore, the proposed space transformation allows S_7 to learn a shallow correlation layer which allows it to outperform all other siamese architectures.

The results reported do not guarantee that replacing the siamese architectures of more complex models, such as the one proposed by Kendall et al. [5], will improve matching performance, but they show promising potential even without spatial regularization. If nothing else, our models, just like the ones proposed by Luo et al. [8], provide a simple, fast and easy to train method, but much more accurate results.

5. Conclusion

Similar to so many areas in computing, deep learning has allowed us to move at an incredible speed towards a robust solution for stereo matching. As computation power increases, there is a natural tendency to move to bigger and more complex CNN models. In this work, we demonstrated that big improvements are still possible by small, problem-specific adaptations that simplify the learning problem. For future work, we plan to incorporate the recent approaches that use context for regularization, allowing us to take full advantage of the proposed feature extractor.

Acknowledgement

This work was supported by the EU-Horizon2020 project Endoo (H2020-ICT-2015-688592), the EPSRC (EP/P012841/1) and the Wellcome/EPSRC Centre for Interventional and Surgical Sciences (WEISS) at UCL (203145Z/16/Z).

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden,

M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: large-scale machine learning on heterogeneous systems, 2015, Software available from arXiv:1603.04467v2.

[2] M.Z. Brown, D. Burschka, G.D. Hager, Advances in computational stereo, IEEE Trans. Pattern Anal. Mach. Intell. 25 (2003) 993–1008.

[3] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, T. Brox, Flownet: learning optical flow with convolutional networks, arXiv:1504.06852 (2015).

[4] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? The kitti vision benchmark suite, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2012.

[5] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, End-to-end learning of geometry and context for deep stereo regression, in: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22–29, 2017, 2017, pp. 66–75.

[6] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, CoRR, 2014. abs/1412.6980.

[7] P. Knöbelreiter, C. Reinbacher, A. Shekhovtsov, T. Pock, End-to-end training of hybrid CNN-CRF models for stereo, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017, 2017, pp. 1456–1465.

[8] W. Luo, A.G. Schwing, R. Urtasun, Efficient deep learning for stereo matching, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5695–5703.

[9] M. Menze, A. Geiger, Object scene flow for autonomous vehicles, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[10] J. Pang, W. Sun, J. Ren, C. Yang, Q. Yan, Cascade residual learning: a two-stage convolutional neural network for stereo matching, ICCVW, 3, 2017.

[11] H. Park, K.M. Lee, Look wider to match image patches with convolutional neural networks, IEEE Signal Process. Lett. (2016).

[12] R. Rojas, Neural Networks: A Systematic Introduction, Springer Science & Business Media, 2013.

[13] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (2015) 211–252.

[14] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, P. Westling, High-resolution stereo datasets with subpixel-accurate ground truth, in: German Conference on Pattern Recognition, Springer, 2014, pp. 31–42.

[15] D. Scharstein, R. Szeliski, A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, Int. J. Comput. Vis. 47 (2002) 7–42.

[16] A. Shaked, L. Wolf, Improved stereo matching with constant highway networks and reflective confidence learning, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017, 2017, pp. 6901–6910.

[17] J. Zbontar, Y. LeCun, Stereo matching by training a convolutional neural network to compare image patches, J. Mach. Learn. Res. 17 (2016) 2.