Left-Right Comparative Recurrent Model for Stereo Matching

Zequn Jie* Pengfei Wang† Yonggen Ling* Bo Zhao§ Yunchao Wei≀ Jiashi Feng† Wei Liu*

*Tencent AI Lab †National University of Singapore
§University of British Columbia ≀University of Illinois Urbana-Champaign
{zequn.nus, wpfhtl, lingyg2008, zhaobo.cs, wychao1987, jshfeng}@gmail.com wliu@ee.columbia.edu

Abstract

Leveraging the disparity information from both left and right views is crucial for stereo disparity estimation. Left-right consistency check is an effective way to enhance the disparity estimation by referring to the information from the opposite view. However, the conventional left-right consistency check is an isolated post-processing step and heavily hand-crafted. This paper proposes a novel left-right comparative recurrent model to perform left-right consistency checking jointly with disparity estimation. At each recurrent step, the model produces disparity results for both views, and then performs online left-right comparison to identify the mismatched regions which may probably contain erroneously labeled pixels. A soft attention mechanism is introduced, which employs the learned error maps for better guiding the model to selectively focus on refining the unreliable regions at the next recurrent step. In this way, the generated disparity maps are progressively improved by the proposed recurrent model. Extensive evaluations on KITTI 2015, Scene Flow and Middlebury benchmarks validate the effectiveness of our model, demonstrating that state-of-the-art stereo disparity estimation results can be achieved by this new model.

1. Introduction

This paper aims at the problem of computing the dense disparity map between a rectified stereo pair of images. Stereo disparity estimation is core to many computer vision applications, including robotics and autonomous vehicles [4, 5, 16]. Finding local correspondences between two images plays a key role in generating high-quality disparity maps. Modern stereo matching methods rely on deep neural networks to learn powerful visual representations and compute more accurate matching costs between left and right views.

However, it is still challenging for the current methods to deal with ill-posed regions, such as object occlusions, reflective regions, and repetitive patterns. It is also observed that the mismatched pixels between the left and right views usually appear in the error-prone regions, including occluded objects, textureless areas, and sophisticated image borders (see Fig. 1). Taking advantage of disparity information from both the views to verify the left-right mutual consistency is an effective strategy to identify the unreliable regions. By doing this, the stereo disparity estimation results can be selectively improved by being refined on the mismatched regions. Traditional left-right consistency check is performed only as an offline post-processing step after the disparity map generation. Moreover, it is highly hand-crafted and hardwired—it only refines the pixels having errors above a mismatching threshold by interpolating their fixed local neighbors. The results are thus fragile due to low-quality local features and potential errors in the matching cost computation. As such, the traditional pipelines, whose regularization steps involve hand-engineered, shallow cost aggregation and disparity optimization (SGM or

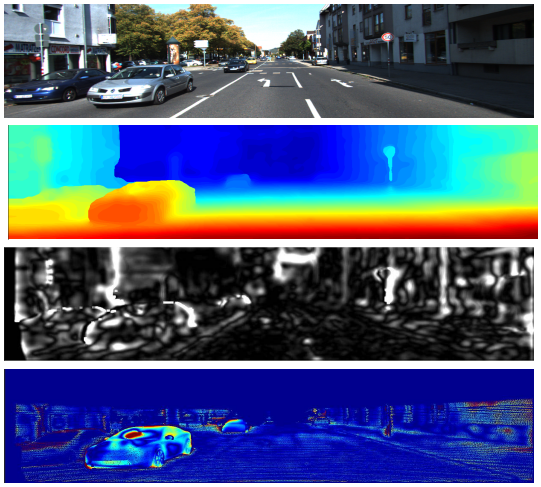

Figure 1: From top to bottom: the input image, predicted disparity map, learned attention map showing the potential erroneously labeled regions by the LRCR model, and the real error map (difference between the predicted and ground-truth maps).

MRF), have been proven to be inferior [28] and sub-optimal for stereo disparity estimation.

To overcome the above issue, we propose a novel *Left-Right Comparative Recurrent* (LRCR) model to integrate the left-right consistency check and disparity generation into a unified pipeline, which allows them to mutually boost and effectively resolve the drawbacks of offline consistency check. Formulated as a recurrent neural network (RNN)-style model, the LRCR model can learn to progressively improve the disparity estimation for both the left and right images by exploiting the learned left-right consistency. In this way, both disparity maps for two views favorably converge to stable and accurate predictions eventually. LRCR creatively introduces an attention mechanism accompanying recurrent learning to simultaneously check consistency and select proper regions for refinement. Concretely, at each recurrent step, the model processes both views in parallel, and produces both disparity maps as well as their associated error maps by performing the online left-right comparison. The error maps reflect the correctness of the obtained disparities by "referring" the disparity map of the other view. Treating the error maps as soft attention maps, the model is guided to concentrate more on the potential erroneously predicted regions in the next recurrent step. Such an error-diagnosis self-improving scheme allows the model to automatically improve the estimation of both views without tedious extra supervision over the erroneously labeled regions. Moreover, incorporating the left-right consistency into the model training achieves the desirable consistency between the training and inference (application) phases. Thus, LRCR can improve the disparity estimation performance in a more straightforward and expected manner, with better optimization quality.

The proposed left-right comparative recurrent model is based on a convolutional Long-Short Term Memory (ConvLSTM) network, considering its superior capability of incorporating contextual information from multiple disparities. Besides, the historical disparity estimation stored in LSTM memory cells automatically flows to following steps, and provides a reasonably good initial disparity estimation to facilitate generating higher-quality disparities in later steps. The proposed LRCR model replaces the conventional hand-crafted regularization methods [27] plus the "Winner Takes All" (WTA) pipeline to estimate the disparity, offering a better solution.

To summarize, our contributions are as follows.

1. We propose a novel deep recurrent model for better handling stereo disparity estimation tasks. This model generates increasingly consistent disparities for both views by effectively learning and exploiting online left-right consistency. To the best of our knowledge, this is the first end-to-end framework unifying consistency check and disparity estimation.
2. A soft attention mechanism is introduced to guide the network to automatically focus more on the unreliable regions learned by the online left-right comparison during the estimation.
3. We perform extensive experiments on the KITTI 2015 [19, 20], Scene Flow [17] and Middlebury [26] benchmarks to validate the effectiveness of our proposed model, and show that it can achieve state-of-the-art results on these benchmarks.

2. Related Work

Traditional stereo matching methods usually utilize the low-level features of image patches around the pixel to measure the dissimilarity. Local descriptors such as absolute difference (AD), sum of squared difference (SSD), census transform [12], or their combination (AD-CENSUS) [18] are often employed. For cost aggregation and disparity optimization, some global methods treat disparity selection as a multi-label learning problem and optimize a corresponding 2D graph partitioning problem by graph cut [1] or belief propagation [3, 31, 33]. Semi-global methods [10] approximately solve the NP-hard 2D graph partitioning by factorizing it into independent scan-lines and leveraging dynamic programming to aggregate the matching cost.

Deep learning has been used in stereo matching recently [34, 35, 15, 2, 29, 32]. Zbontar *et al*. [35] trained a siamese network to extract patch features and then compared patches accordingly. Luo *et al*. [15] proposed a faster network model which computes local matching costs as performing multi-label classification over disparities. Chen *et al*. [2] presented a multi-scale embedding model to obtain faithful local matching costs. Guney *et al*. [7] introduced object-category specific disparity proposals to better estimate the disparity. Shaked *et al*. [29] proposed a constant highway network to learn both 3D cost volume and matching confidence. Taniai *et al*. [32] over-parameterized each pixel with a local disparity plane and constructed an MRF to estimate the disparity map. Combining the matching cost from [35] and their proposed local expansion move algorithm also achieves good performance on the Middlebury stereo evaluation [26].

End-to-end deep learning methods have also been proposed to take the raw images as input and output the final disparity results with deep neural networks without offline post-processing. Mayer *et al*. [17] leveraged a large dataset to train a convolutional neural network (CNN) to estimate the disparity directly from the image appearances. In [17], the cost volume was constructed with the extracted features and the disparity was learned with 3D convolution. GC-NET [13] combines the geometric and contextual information with 3D convolution to learn the disparity.

Another type of work focuses on getting more reliable measurements of disparity. A review [11] has summarized

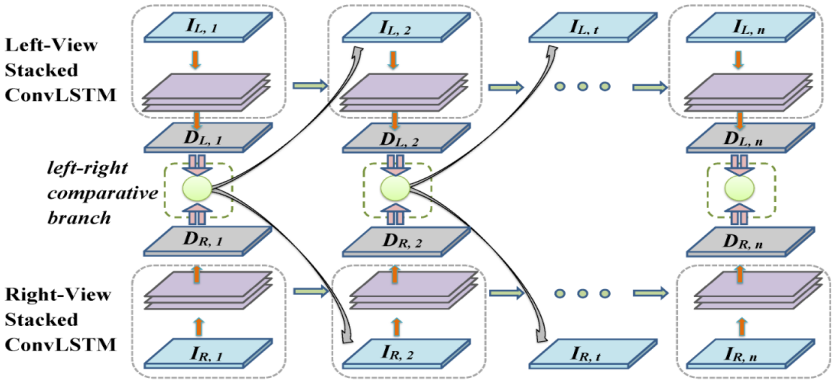

Figure 2: An illustration of the LRCR model. At each step, the Stacked ConvLSTMs of both views take as input the corresponding matching cost volume and the error map of that view obtained at the last step. Then, the generated disparity maps are compared to produce the error maps of both views to be fed into the model at the next step, which serve as the soft attention guidance to allow the model to selectively improve the regions.

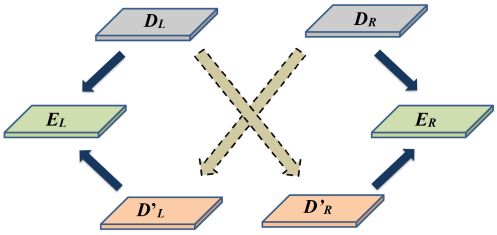

Figure 3: An illustration of the left-right comparative branch. The predicted disparity maps (*i.e.*, D_L and D_R) are first converted to the opposite coordinates to obtain the induced disparity maps (*i.e.*, D'_L and D'_R). Then D_L and D'_L are fed into a simple network with several convolutional layers to learn the error map E_L for the left view, which are the comparison outputs. The same process also applies for the right view.

and compared an exhaustive list of hand-crafted confidence measurements for disparity maps. CNN has also been implemented to improve the accuracy of a confidence prediction by exploiting the local information [30, 9, 24, 28, 22, 25, 23]. Haeusler *et al*. [9] combined different confidence measurements with a random forest classifier to measure the reliability of disparity maps. Seki and Pollefeys [28] estimated the confidence map by extracting patches from both the left and right disparity maps with a CNN. Poggi and Mattoccia [24, 23] predicted the confidence from only one disparity map within a deep network.

The most closely related work to ours is [6], which also considers incorporating error localization and improvement. [28] is also related in the sense of detecting left-right mismatched regions as potential erroneous regions. However, these methods are only for refinement and require external disparity results as inputs. By comparison, our method is a unified model integrating disparity generation and left-right comparison without requiring the initial disparity. Our proposed model can generate disparity maps with increasing left-right consistency, by learning to focus on erroneously predicted regions for disparity enhancement. [8] and [14] consider left-right consistency in the monocular depth estimation under unsupervised and semi-supervised settings, respectively.

3. Disparity Estimation with LRCR

This paper focuses on computing the disparity map for given rectified stereo pair images. Given left and right input images, we aim to produce the corresponding disparity results. The proposed left-right comparative recurrent (LRCR) model takes the matching cost volume of each pixel at all possible disparities as input. It outputs the disparity map containing the disparity values of all the pixels. The matching cost volume is a 3D tensor of size $H \times W \times d_{\max}$, where H, W, and $d_{\max}$ are the height, the width of the original image, and the maximal possible disparity, respectively. A pixel in the cost volume shows the matching cost between a patch centered around $\mathbf{p} = (x, y)$ in the left image and a patch centered around $\mathbf{pd} = (x - d, y)$, for every pixel $\mathbf{p}$ in the left image and every possible disparity d.

We deploy the constant highway networks [29] to produce the matching cost volume which serves as the input to our proposed LRCR model for disparity estimation. The constant highway network is trained on pairs of small image patches whose true disparity is known. It adopts a siamese architecture, where each branch processes the left or right image patches individually and generates its own description vector. Each branch contains several highway residual blocks with shared weights [29]. Subsequently, two pathways are exploited to compare the two patches and produce a matching cost. In the first pathway, the two feature vectors are concatenated into a single one, and then passed through several fully-connected layers to obtain a binary decision trained via a cross-entropy loss. The second pathway employs a hinge loss to the dot product between the two feature vectors. Both pathways provide the matching cost volumes that can be taken as input by the later LRCR model for the disparity estimation. During inference, the fully-connected layers are cast into 1×1 convolutional layers. Taking the whole images as inputs, the constant highway network outputs the matching cost volumes with the same spatial dimensions in one feed-forward pass efficiently.

3.1. LRCR Model

Based on the computed matching cost volume, conventional stereo matching pipelines apply several regularization techniques to incorporate information from neighboring pixels to obtain smoothed matching costs. Then a simple "Winner Takes All" (WTA) rule is implemented to determine the disparity for each pixel. However, such pipelines

are still highly hand-crafted and merely uses shallow functions to pool the contextual information from neighboring pixels.

In contrast, we propose a deep model, called as LRCR, which automatically learns the local contextual information among the neighborhood for each pixel based on the obtained matching cost volume. Specifically, LRCR simultaneously improves the disparity results of both views by learning to evaluate the left-right consistency. Formulated as an RNN-style model, the LRCR model can receive the past predicted disparity results and selectively improve them by learning information about the potential left-right mismatched regions. The progressive improvements facilitate predicting disparity maps with increasing left-right consistency as desired.

Fig. 2 shows the architecture of the proposed LRCR model, which contains two parallel stacked convolutional LSTM networks. The left network takes the cost volume matching left image to right as input and generates a disparity map for the left view. Similarly, the right network generates the disparity map for the right view. At each recurrent step, the two stacked convolutional LSTMs process the two input cost volumes and generate the corresponding disparity maps individually. The two generated disparity maps are then converted to the opposite coordinates [28] for comparison with each other. Formally, let $D_{\text{left},t}$ and $D_{\text{right},t}$ denote the disparity maps derived from the left-view and right-view matching cost volumes at the t^{th} step, respectively. $D_{\text{right},t}$ can be converted to the left-view coordinates, providing a right-view induced disparity map $D'_{\text{left},t}$. Then, a pixel-wise comparison can be conveniently performed between $D_{\text{left},t}$ and $D'_{\text{left},t}$ with the information from both views. For doing the comparison, several convolutional layers and pooling layers are added on the top of $D_{\text{left,t}}$ and $D'_{\text{left,t}}$, producing the error map of the left-view disparity. Taking such an error map, which serves as a soft attention map (together with the left-view matching cost volume), as input at the next step, the LRCR model suffices to selectively focus more on the left-right mismatched regions at the next step.

Learning to generate error maps is not trivial. An alternative is to impose the supervision by adopting the difference between the predicted and corresponding ground-truth disparity maps as the regression target. To enhance robustness and solve the aforementioned issues, we propose not to impose explicit supervision on the error map generation, in order to prevent the left-right comparative branch from learning the simple element-wise subtraction when comparing the two disparity maps. Instead, allowing the network to automatically learn the regions that need to be focused more at the next step may be beneficial to capturing the underlying local correlation between the disparities in the neighborhood. Moreover, we aim to provide a soft attention map whose values serve as confidence or attention weights for the next step prediction, which can hardly be interpreted as a simple element-wise difference between two disparity maps. It is observed that the learned error maps indeed provide reliable results in detecting the mismatched labels which are mostly caused by repetitive patterns, occluded pixels, and depth discontinuities (see Fig. 1). For the right view, the model performs the exactly same process including the conversion of left-view disparity, right-view error map generation, and using the error map as the attention map in the next step of right-view prediction.

3.2. Architecture

We describe the architecture of LRCR model in details, including the stacked convolutional LSTM and the left-right comparative branch.

3.2.1 Stacked Convolutional LSTM

Convolutional LSTM (ConvLSTM) networks have an advantage in encoding contextual spatial information and reducing the model redundancy, compared to conventional fully-connected LSTM networks. The LRCR model contains two parallel stacked ConvLSTM networks that process the left and right views respectively. The inputs to each stacked ConvLSTM include the matching cost volume of that view and the corresponding error map generated at the last step. The matching cost volume is a 3D tensor of size $H \times W \times d_{\max}$ and the error map is a 2D map of size $H \times W$. They are first concatenated along the disparity dimension to form a tensor of size $H \times W \times (d_{\max}+1)$. ConvLSTM is similar to usual fully-connected LSTM, except that the former applies spatial convolution operations on the 2D map in several layers to encode spatial contexts. The detailed unit-wise computations of the ConvLSTM are shown below:

$$
\begin{aligned}
i_t &= \sigma\big(W_{xi} * X_t + W_{hi} * H_{t-1} + W_{ci} \circ C_{t-1} + b_i\big),\\
f_t &= \sigma\big(W_{xf} * X_t + W_{hf} * H_{t-1} + W_{cf} \circ C_{t-1} + b_f\big),\\
o_t &= \sigma\big(W_{xo} * X_t + W_{ho} * H_{t-1} + W_{co} \circ C_{t-1} + b_o\big),\\
C_t &= f_t \circ C_{t-1} + i_t \circ \tanh\big(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c\big),\\
H_t &= o_t \circ \tanh(C_t).
\end{aligned}
\tag{1}
$$

Here $*$ denotes the spatial convolution operation, and X_t is the input tensor concatenated by the matching cost volume and the error map generated at the t^{th} step. H_{t-1} and C_{t-1} are the hidden state and memory cell of the $(t-1)^{th}$ step, respectively. i_t, f_t and o_t are the gates of the ConvLSTM. Feeding the hidden states of a ConvLSTM as input to another ConvLSTM successively builds a stacked ConvLSTM.

The hidden state tensor of the last ConvLSTM is then passed through simple convolutional layers to obtain the cost tensor of size $H \times W \times d_{\max}$. Taking the negative of each value in the cost tensor results in a score tensor. Applying softmax normalization to the score tensor leads to a

probability tensor that reflects the probability on each available disparity for all pixels. Finally, a differentiable $\arg\min$ layer [13] is used to generate the predicted disparity map by summing all the disparities weighted by their probabilities. Mathematically, the following equation describes how one can obtain the predicted disparity d^* given the costs on each available disparity c_d via the cost tensor for a certain pixel:

$$d^* = \sum_{d=0}^{d_{\max}} d \times \sigma(-c_d). \tag{2}$$

3.2.2 Left-Right Comparative Branch

Fig. 3 shows an illustration of the left-right comparative branch. The left-right comparative branch takes as input the left and right disparity maps (*i.e.*, D_{left} and D_{right}) generated by the left-view and right-view stacked ConvLSTMs, respectively. This branch first converts both maps to the coordinates of the opposite view (*i.e.*, D'_{left} and D'_{right}). Next, the original disparity map and converted disparity map (*e.g.*, D_{left} and D'_{left}) are concatenated and fed into a simple branch consisting of several simple convolutional layers and a final sigmoid transformation layer to obtain the corresponding error map for that view. This error map is then used as the soft attention map in the next step to guide the focused regions for the model.

3.3. Training

The supervisions of the whole ground-truth disparity maps are imposed on the predicted disparity maps for both left and right views at each recurrent step. The ℓ_1 norm of the difference between the ground-truth and the predicted disparity is used as the training loss. The averaged loss over all the labeled pixels is optimized for a particular sample. The loss is mathematically defined in (3), where N is the number of labeled pixels for the sample, and d_n and d_n^* are the predicted disparity and the ground-truth disparity for the n^{th} pixel, respectively.

$$L = \frac{1}{N} \sum_{n=1}^{N} \left|\left|d_n - d_n^*\right|\right|_1. \tag{3}$$

Due to the complexity of the model architecture, we train the whole LRCR model in two stages. We expect the model trained by the first stage to provide reliable initial disparity estimations, while the second stage trains the model to progressively improve the estimations. At the first stage, we train the stacked ConvLSTMs with only one recurrent step. Indeed, the stacked ConvLSTM performs as a feed-forward network that receives only the original matching cost volumes and predicts disparity by pooling local context via spatial convolutions, similar to [13] and [36]. In this stage, H_{t-1} and C_{t-1} are set as constant zero tensors. Therefore, only W_{xi}, W_{xf} and W_{xc} are trained in this stage, and the rest weights in (1) will not be trained. This stage results in a non-recurrent version of the LRCR model. In the second stage, the non-recurrent version of the LRCR model trained previously is used as the initialization to the full LRCR model. All the weights in (1) as well as the left-right comparative branch will be trained in this stage.

4. Experiments

We extensively evaluate our LRCR model on three largest and most challenging stereo benchmarks: KITTI 2015, Scene Flow and Middlebury. In Sec. 4.1, we experiment with different variants of our model and provide detailed component analysis on the KITTI 2015 dataset. In Sec. 4.2, we conduct the experiments on the Scene Flow dataset. In Sec. 4.3, we report the results on the Middlebury benchmark and also compare our model against the state-of-the-arts. In all the experiments, after generating the disparity maps by the LRCR model, sub-pixel enhancement plus median and bilateral filterings is used to refine the results.

4.1. Experiments on KITTI 2015

Dataset. KITTI 2015 is a real-world dataset with dynamic street views captured by a driving car. It provides 200 stereo pairs with sparse ground-truth disparities for training and 200 pairs for testing through online leaderboard. We randomly split a validation set with 40 pairs from the training set. When evaluated on the validation set, our model is trained on the training set containing the rest 160 pairs. When doing the evaluation on the testing set, we train the model on the whole training set.

Implementation details. The constant highway network [29] is used to learn the matching cost volumes as input to our LRCR model. It has two versions, *i.e.,* the hybrid one and the fast one. The hybrid version contains two pathways trained by both the cross-entropy loss and the hinge loss in the matching cost learning, while the fast one has only the dot-product pathway trained by the hinge loss. $d_{\max}$ is set to 228. In our LRCR model, each of the parallel stacked ConvLSTMs contains four ConvLSTM layers. The numbers of output channels in the four ConvLSTM layers are $d_{\max}$, $2d_{\max}$, $2d_{\max}$, and $d_{\max}$, respectively. All the convolution kernel sizes in the four ConvLSTM layers are $3*3$, and $1*1$ paddings are applied to keep the feature map sizes. After the four ConvLSTM layers, additional three $1*1$ convolutional layers are included and followed by the final differentiable $\arg\min$ layer. The numbers of output channels in the three $1*1$ convolutional layers are $d_{\max}$, $d_{\max}$, and $d_{\max}$, respectively. The left-right comparative branch has two $3*3$ convolutional layers with $1*1$ padding.

As explained in Sec. 3.3, we first perform the training of the non-recurrent version of our LRCR model. This

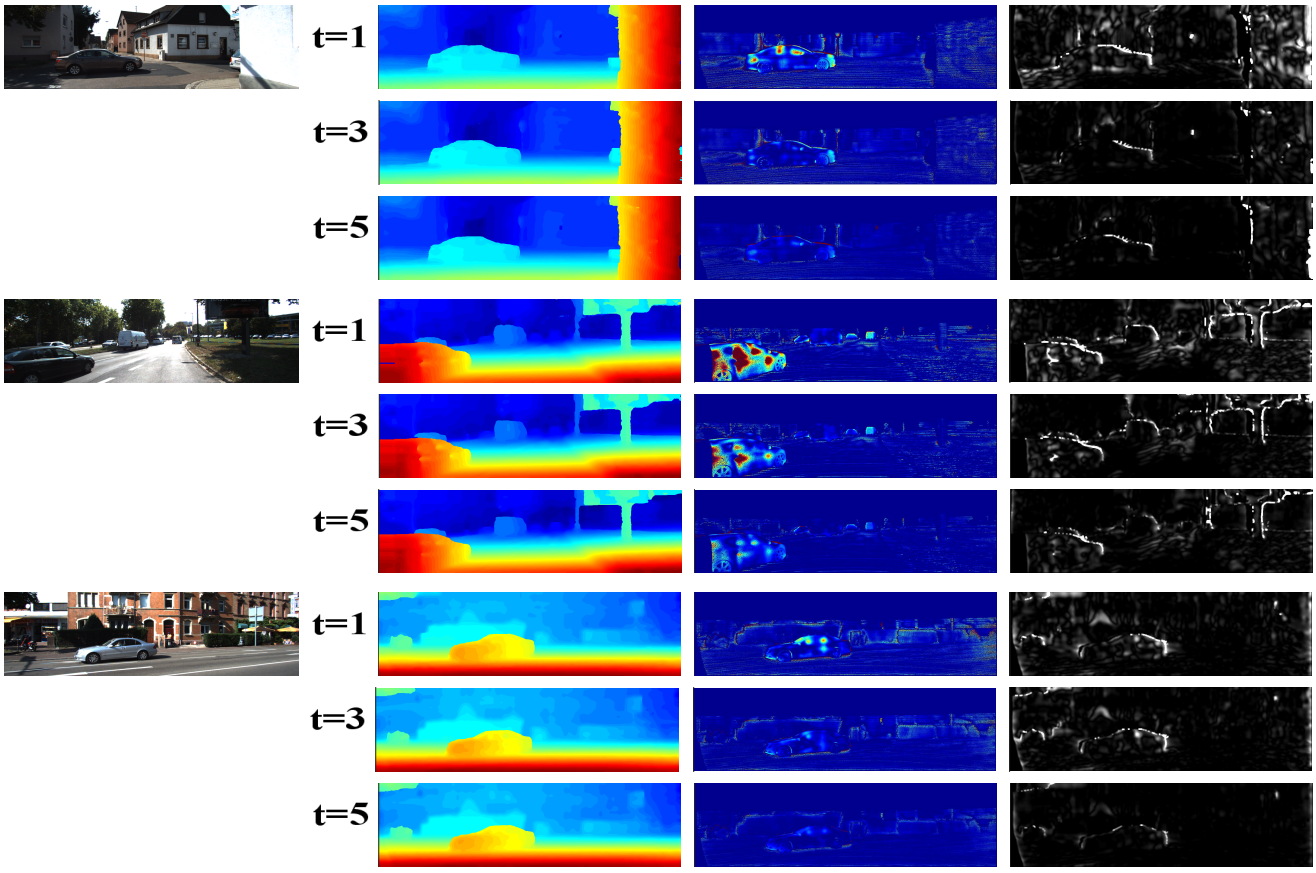

Figure 4: Visualization of input images, predicted disparity maps, real error maps (difference from the ground-truth disparity maps), and the learned attention maps at different recurrent steps. Note that the attention maps of the 5th step are obtained by forwarding the final disparity (obtained at the 5th step) to the left-right comparative branch once more.

stage lasts for 15 epochs. The initial learning rate is set to 0.01 and decreased by a factor of 10 every 5 epochs. In the second stage, the well trained non-recurrent version of the LRCR model is used for the weight initialization. We train the recurrent model for 5 recurrent steps. We train 30 epochs in this stage. The initial learning rate is set to 0.01 and decreased by a factor of 10 every 10 epochs.

Results. Based on the matching cost volumes produced by the constant highway network, we may directly apply the "Winner Takes All" (WTA) strategy, or choose different ablation versions of the LRCR model to generate the disparity maps. We thus compare WTA, full LRCR, and its multiple variants including non-recurrent, non-comparison, and non-attention. The non-recurrent version is a one-step feed-forward network explained in Sec. 3.3. The non-comparison version does not perform left-right comparison at each recurrent step. Instead, it just updates the left-view disparity based on only the left-view matching cost volumes at each step. The non-attention version does not include the left-right comparative branch to learn the attention map. Instead, it replaces the learned attention maps with a simple subtraction on the two disparity maps (*e.g.*, D_{left} and D'_{left}).

Table 1 shows the results of different versions of the LRCR model on the KITTI 2015 validation set. It can be seen that "WTA" performs worse than all the deep network based methods. All the recurrent versions achieve better results in the final step than the non-recurrent models. The results of the early steps (*e.g.*, 1st or 2nd steps) of the recurrent versions are slightly worse than the non-recurrent version. The reason may be that the recurrent versions are optimized at all the steps and the gradients of the later steps are propagated back to the early steps, making the prediction of early steps less accurate. Among the three recurrent versions, the full LRCR model consistently performs the best at all the recurrent steps. The non-comparison version produces the worst results without the guiding information from the opposite view.

To better demonstrate the effectiveness of the proposed

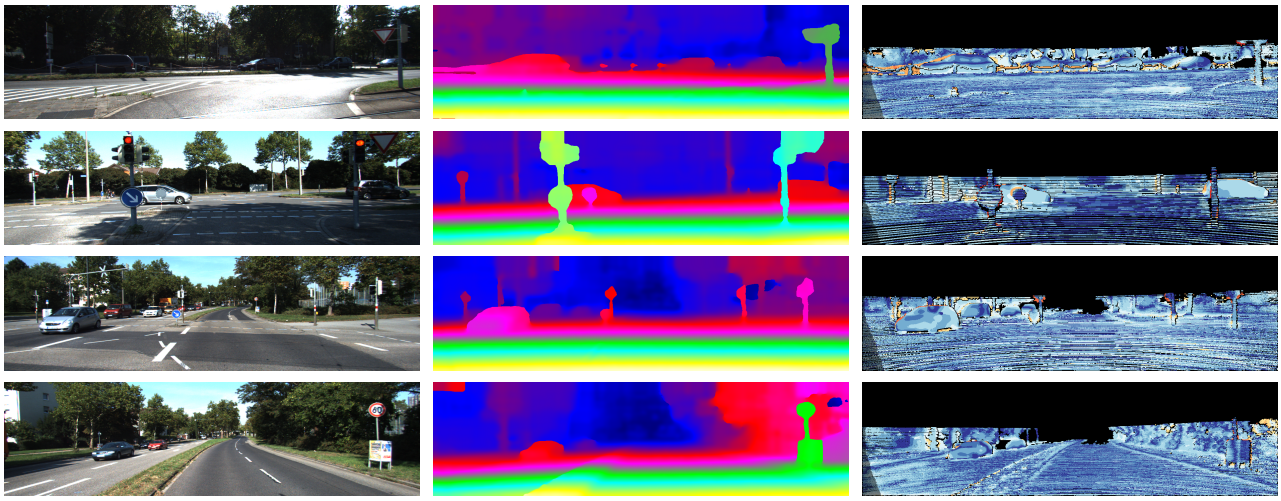

Figure 5: Qualitative results of: (left) input images, (center) predicted disparity maps, and (right) stereo errors.

Table 1: Stereo matching results (end point error and pixel percentages with errors larger than 2, 3 and 5 pixels) of different ablation versions of the LRCR model on the KITTI 2015 validation set. The results of different steps for the recurrent versions are shown.

Model	Recurrent Type	>2 px	>3 px	>5 px	EPE
WTA	–	7.94	4.78	3.11	1.144
Non-recurrent	–	5.70	3.69	2.44	0.939
t=1	w/o comp	6.24	3.92	2.54	0.972
	w/o atten	6.15	3.87	2.53	0.969
	LRCR	6.20	3.90	2.54	0.971
t=2	w/o comp	5.88	3.74	2.46	0.950
	w/o atten	5.79	3.72	2.45	0.945
	LRCR	5.61	3.66	2.42	0.935
t=3	w/o comp	5.64	3.67	2.43	0.936
	w/o atten	5.05	3.44	2.29	0.891
	LRCR	4.85	3.35	2.15	0.877
t=4	w/o comp	5.43	3.61	2.38	0.922
	w/o atten	4.54	3.22	2.08	0.855
	LRCR	4.33	3.12	2.02	0.839
t/=5	w/o comp	5.32	3.58	2.36	0.913
	w/o atten	4.14	3.05	1.99	0.825
	LRCR	3.92	2.96	1.93	0.806

LRCR model, we illustrate the learned attention maps and the real error maps at different recurrent steps in Fig. 4. The real error maps are the difference maps between the predicted and ground-truth disparity maps. As can be seen, the learned attention maps have high consistency with the real error maps at different steps. Both the learned attention maps and the real error maps mainly highlight the error-prone regions, including reflective regions, occluded objects, and sophisticated boundaries. The consistency between the erroneously predicted regions and the left-right mismatched regions is crucial for the designing of the LRCR model. This makes it possible that the left-right comparative branch can localize the erroneously labeled pixels with only the predicted disparities from both views, without explicit supervision over the erroneously labeled regions. With more recurrent steps, the learned attention maps become less highlighted, indicating the increasing consistency of the disparity maps from the two views. We also show the qualitative visualization of the final predicted disparity maps by the LRCR model on the KITTI 2015 testing set in Fig. 5.

Table 2: Stereo matching results (error rates of the non-occluded pixels and all the pixels) of the LRCR model and the state-of-the-art methods on the KITTI 2015 testing set.

Type	Method	NOC	ALL	Runtime
Others	MC-CNN-acrt [35]	3.33	3.89	67s
	Content-CNN [15]	4.00	4.54	1s
	Displets v2 [7]	3.09	3.43	265s
	DRR [6]	2.76	3.16	1.4s
End-to-end CNN	GC-NET [13]	2.61	2.87	0.9s
	CRL [21]	2.45	2.67	0.47s
LRCR and the baseline	λ-ResMatch (fast) [29]	3.29	3.78	2.8s
	λ-ResMatch (hybrid) [29]	2.91	3.42	48s
	Ours (fast)	2.79	3.31	4s
	Ours (hybrid)	2.55	3.03	49.2s

We then compare the LRCR model against the state-of-the-arts on the KITTI 2015 testing set. The top-ranked methods of the online leaderboard is shown in Table 2. Since our model is based on the matching cost volumes obtained by [29], we show the comparisons to [29] based on both the fast and hybrid versions. From Table 2, the proposed LRCR model outperforms [29] significantly while has similar running time to [29]. It is worth mentioning that the main time consumption lies in the matching cost computation at all possible disparities, especially when fully-connected layers are used in the matching cost generation.

Table 3: Stereo matching results (end point error and pixel percentages with errors larger than 1, 3 and 5 pixels) of different variants of the LRCR model, and the state-of-the-art methods on the Scene Flow testing set.

Model	>1 px	>3 px	>5 px	EPE
WTA	29.68	18.47	12.16	8.07
GC-NET [13]	16.9	9.34	7.22	2.51
Non-recurrent	23.32	14.08	9.69	5.26
LRCR (t=1)	25.61	15.53	10.57	6.33
LRCR (t=2)	21.36	12.80	9.03	4.54
LRCR (t=3)	18.59	10.84	7.86	3.18
LRCR (t=4)	16.74	9.54	7.03	2.43
LRCR (t=5)	15.63	8.67	6.56	2.02

Table 4: Stereo matching results (error rates) on the Middlebury testing set.

Method	Error Rate
MC-CNN-acrt[35]	8.08
λ-ResMatch (hybrid)[29]	9.08
LRCR (hybrid)	7.43

End-to-end models may have slightly better results than the LRCR model, but they are all trained with a huge amount of extra training data [17]. For example, [13] and [21] utilize the large Scene Flow training set to train their models (*i.e.*, extra 34k pairs of training samples for GC-NET [13] and extra 22k pairs of training samples for CRL [21]). The demand for the huge training set is due to the very deep networks that they use to achieve the end-to-end disparity estimation. For example, the CRL model [21] contains 47 convolutional layers with around 74.95M learnable parameters in its two-stage cascaded architecture. In comparison, our LRCR model has only about 30M learnable parameters, including both the hybrid version of the constant highway networks for matching cost learning and the stacked ConvLSTMs for disparity estimation. We also claim the possibility of the LRCR model in achieving better disparity results when leveraging stronger networks to learn the matching costs, *e.g.*, embedding the LRCR model to the existing end-to-end networks.

4.2. Experiments on Scene Flow

Dataset. The Scene Flow dataset [17] is a large scale synthetic dataset containing around 34k stereo pairs for training and 4k for testing.

Implementation details. First, the hybrid version of the constant highway network [29] is used to learn the matching cost volumes. $d_{\max}$ is set to 320. Each parallel ConvLSTM in the LRCR model has four ConvLSTM layers with $3 * 3$ convolutions. Three additional $1 * 1$ convolutional layers are also added after the ConvLSTM layers. The numbers of output channels in the four ConvLSTM layers and the three $1 * 1$ convolutional layers are $d_{\max}$, $2d_{\max}$, $2d_{\max}$, $d_{\max}$, $d_{\max}$, $d_{\max}$, and $d_{\max}$, respectively. The numbers of epochs for both training stages and the learning rate setting are the same as those in the KITTI 2015 experiments.

Results. We show the Scene Flow testing set results of applying "WTA", the non-recurrent version model, and the full LRCR model in Table 3. The LRCR model outperforms the state-of-the-art end-to-end GC-NET [13] noticeably. One can also find that the improvement trend on the Scene Flow dataset is similar to that on KITTI 2015. The relative improvements between consecutive steps become smaller with the increasing of the recurrent steps. The LRCR model has better results than the non-recurrent version model at the later steps, while performs slightly worse at the 1st step. The "WTA" strategy is inferior to all the deep network based methods.

4.3. Experiments on Middlebury

The Middlebury dataset contains five subsets [26] of indoor scenes. The image pairs are indoor scenes given a full, half and quarter resolutions. Due to the small number of training images in Middlebury, we learn the matching costs by fine-tuning the constant highway network model in the Scene Flow on the Middlebury training set. For the LRCR model training, we directly train the full LRCR model with 5 recurrent steps by fine-tuning the LRCR model trained on the Scene Flow dataset. The training lasts for 10 epochs with the initial learning rate set to 0.001 and decreased by a factor of 10 after 5 epochs.

Table 4 shows the result comparison between the LRCR model and [29] which shares the same matching cost as our method. The proposed LRCR model outperforms [29] significantly. The LRCR model achieves better results than another deep learning based method which shows outstanding performance on this dataset, *i.e.*, MC-CNN-acrt [35].

5. Conclusion

In this paper, we proposed a novel left-right comparative recurrent model, dubbed LRCR, which is capable of performing left-right consistency check and disparity estimation jointly. We also introduced a soft attention mechanism for better guiding the model itself to selectively focus on the unreliable regions for subsequent refinement. In this way, the proposed LRCR model is shown to progressively improve the disparity map estimation. We conducted extensive evaluations on the KITTI 2015, Scene Flow and Middlebury datasets, which validate that LRCR outperforms conventional disparity estimation networks with offline left-right consistency check, and achieves comparable results with the state-of-the-arts.

Acknowledgments

Jiashi Feng was partially supported by NUS startup R-263-000-C08-133, MOE Tier-I R-263-000-C21-112, NUS IDS R-263-000-C67-646 and ECRA R-263-000-C87-133.

References

[1] M. Bleyer and M. Gelautz. Graph-cut-based stereo matching using image segmentation with symmetrical treatment of occlusions. *Signal Processing: Image Communication*, 22(2):127–143, 2007. 2

[2] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang. A deep visual correspondence embedding model for stereo matching costs. In *ICCV*, 2015. 2

[3] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. *International journal of computer vision*, 70(1):41–54, 2006. 2

[4] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 1

[5] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 1

[6] S. Gidaris and N. Komodakis. Detect, replace, refine: Deep structured prediction for pixel wise labeling. *arXiv preprint arXiv:1612.04770*, 2016. 3, 7

[7] F. Gney and A. Geiger. Displets: Resolving stereo ambiguities using object knowledge. In *CVPR*, 2015. 2, 7

[8] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 3

[9] R. Haeusler, R. Nair, and D. Kondermann. Ensemble learning for confidence measures in stereo vision. In *CVPR*, 2013. 3

[10] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2008. 2

[11] X. Hu and P. Mordohai. A quantitative evaluation of confidence measures for stereo vision. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2121–2133, 2012. 2

[12] M. Humenberger, T. Engelke, and W. Kubinger. A census-based stereo vision algorithm using modified semi-global matching and plane fitting to improve matching quality. In *CVPR Workshops*, pages 77–84, 2010. 2

[13] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry. End-to-end learning of geometry and context for deep stereo regression. *arXiv preprint arXiv:1703.04309*, 2017. 2, 5, 7, 8

[14] Y. Kuznietsov, J. Stückler, and B. Leibe. Semi-supervised deep learning for monocular depth map prediction. In *CVPR*, 2017. 3

[15] W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *CVPR*, 2016. 2, 7

[16] W. Maddern and P. Newman. Real-time probabilistic fusion of sparse 3d lidar and dense stereo. In *IROS*, 2016. 1

[17] N. Mayer, E. Ilg, P. Hausser, 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 *CVPR*, 2016. 2, 8

[18] X. Mei, X. Sun, M. Zhou, S. Jiao, H. Wang, and X. Zhang. On building an accurate stereo matching system on graphics hardware. In *ICCV Workshops*, 2011. 2

[19] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *CVPR*, 2015. 2

[20] M. Menze, C. Heipke, and A. Geiger. Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015. 2

[21] J. Pang, W. Sun, J. Ren, C. Yang, and Q. Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *ICCV Workshops*, 2017. 7, 8

[22] M.-G. Park and K.-J. Yoon. Leveraging stereo matching with learning-based confidence measures. In *CVPR*, pages 101–109, 2015. 3

[23] M. Poggi and S. Mattoccia. Learning from scratch a confidence measure. In *BMVC*, 2016. 3

[24] M. Poggi and S. Mattoccia. Learning to predict stereo reliability enforcing local consistency of confidence maps. In *CVPR*, 2017. 3

[25] M. Poggi, F. Tosi, and S. Mattoccia. Quantitative evaluation of confidence measures in a machine learning world. In *ICCV*, 2017. 3

[26] D. Scharstein and R. Szeliski. Middlebury stereo vision page. *Online at http://www. middlebury. edu/stereo*, 2, 2002. 2, 8

[27] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42, 2002. 2

[28] A. Seki and M. Pollefeys. Patch based confidence prediction for dense disparity map. In *BMVC*, 2016. 2, 3, 4

[29] A. Shaked and L. Wolf. Improved stereo matching with constant highway networks and reflective confidence learning. *arXiv preprint arXiv:1701.00165*, 2016. 2, 3, 5, 7, 8

[30] A. Spyropoulos, N. Komodakis, and P. Mordohai. Learning to detect ground control points for improving the accuracy of stereo matching. In *CVPR*, 2014. 3

[31] J. Sun, N.-N. Zheng, and H.-Y. Shum. Stereo matching using belief propagation. *IEEE Transactions on pattern analysis and machine intelligence*, 25(7):787–800, 2003. 2

[32] T. Taniai, Y. Matsushita, Y. Sato, and T. Naemura. Continuous stereo matching using local expansion moves. *arXiv preprint arXiv:1603.08328*, 2016. 2

[33] Q. Yang, L. Wang, and N. Ahuja. A constant-space belief propagation algorithm for stereo matching. In *CVPR*, 2010. 2

[34] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015. 2

[35] J. Zbontar and Y. LeCun. Computing the stereo matching cost with a convolutional neural network. In *CVPR*, 2015. 2, 7, 8

[36] Y. Zhong, Y. Dai, and H. Li. Self-supervised learning for stereo matching with self-improving ability. *arXiv preprint arXiv:1709.00930*, 2017. 5