SUPPLEMENTARY MATERIALS
Bilateral Cyclic Constraint and Adaptive Regularization for Unsupervised Monocular Depth Prediction

Alex Wong, Stefano Soatto
UCLA Vision Lab
University of California, Los Angeles, CA 90095
{alexw, soatto}@cs.ucla.edu

1. Problem Formulation

In this section, we give the formulation for predicting the disparities for a single view using stereo imagery as supervision. Given a single image I^0, our goal is to estimate a function $d = f(I^0, \omega) \in \mathbb{R}_+$ that represents the disparity of I^0, where f is a network parameterized by ω. We assume I^0 belongs to a stereo-pair (I^0, I^1) with which we exploit I^1 to learn a representation d for predicting scene geometry from I^0 by maximizing the posterior distribution:

$$p(d|I^0, I^1) \propto p(I^1|I^0, d) \cdot p(I^0, d) \tag{1}$$

We assume that both the likelihood $p(I^1|I^0, d)$ and the prior $p(I^0, d)$ follow a Laplacian distribution. The likelihood can be approximated by a data fidelity term $g(I^0, I^1, d)$ and the prior by a regularization term $h(I^0, d)$ and will have the form:

$$p(I^1|I^0, d) \approx \exp(-\frac{g(I^0, I^1, d)}{a}) \tag{2}$$

$$p(I^0, d) \approx \exp(-\frac{h(I^0, d)}{b}) \tag{3}$$

We then take the negative log to form our data and regularization terms:

$$\begin{aligned} -\log p(d|I^0, I^1) &\propto -\log \exp(-\frac{g(I^0, I^1, d)}{a}) \cdot \exp(-\frac{h(I^0, d)}{b}) \\ &\propto \frac{1}{a} g(I^0, I^1, d) + \frac{1}{b} h(I^0, d) \\ &\propto \underbrace{g(I^0, I^1, d)}_{\text{data fidelity}} + \alpha \underbrace{h(I^0, d)}_{\text{regularization}} \end{aligned} \tag{4}$$

Given I^1 and d, we can derive $\hat{I}^0 = I^1_{xy+d_{xy}}$. We substitute $g(I^0, I^1, d)$ with a generic image reconstruction function:

$$g(I^0, I^1, d) = \sum_{(x,y)\in\Omega} |I^0_{xy} - \hat{I}^0_{xy}| \tag{5}$$

We can similarly substitute $h(I^0, d)$ with a generic prior such as local smoothness with edge-awareness:

$$h(I^0, d) = \sum_{(x,y)\in\Omega} |(\lambda_{xy}(I^0)|\partial_X d^0_{xy}| + \lambda_{xy}(I^0)|\partial_Y d^0_{xy}|)| \tag{6}$$

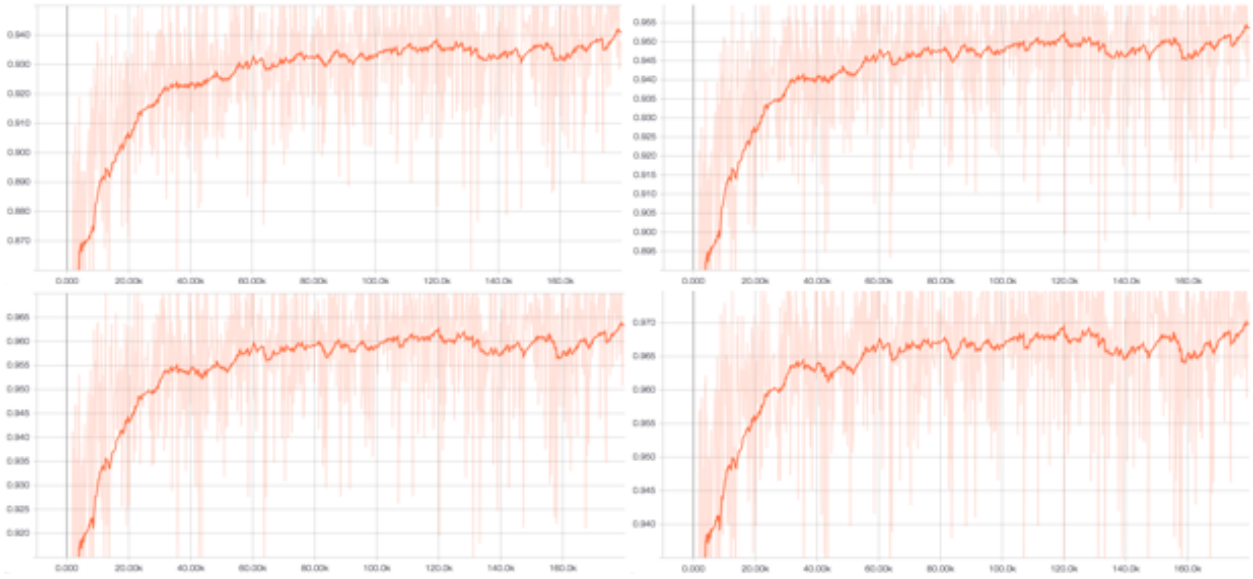

Figure 1: An example of the behavior of our residual-based adaptive weighting scheme (as describe in Sec. 3.2 from main text) in the process of training our model. Top row from left to right: the mean of α applied to resolution $r = 0$ (full resolution) and resolution $r = 1$ (one-half resolution) of the loss pyramid. Bottom row from left to right: mean of α applied to resolution $r = 2$ (one-quarter resolution) and resolution $r = 3$ (one-eighth resolution). The value recorded (light orange) is the mean of the weights that is being applied to the regularization terms of our model. A weight α_{xy} is assigned to each (x, y) position of the solution; hence, the set of weights α will vary spatially across the image domain. The trend line (dark orange) shows that the average weight for regularization is increasing over time (implying that residual is decreasing over time) – the training time varying property exhibited by our adaptive weighting scheme.

2. Adaptive Regularization

Our adaptive regularization weighting scheme α (Sec. 3.2 from main text) allows us to have a model-driven approach to regularization. While a number of literature have exploited a data-driven approach to regularization by weighting the amount of regularity imposed by the structure of the data (e.g. gradients of an image [1, 2]), this class of approaches are still static in terms of weights for a given example as the same image will always give the same weights. Our approach is both model-driven as well as data-driven as our regularization weighting scheme (Eqn. 6 from main text) is a function of the output of the model and the data-fidelity residual (as determined by reconstruction of the images).

Traditionally, the weight of the regularization terms is a static scalar. However, we argue that imposing the same amount of regularity to the entire solution fails to address corner cases. We propose that α should be spatially varying and inversely proportional to the local residual. While the notion of "trusting" the prior (or the regularizers) when data-fidelity fails to explain the scene is intuitive, this assumption is only valid once we are able to sufficiently satisfy the data-fidelity term; otherwise we are restricting our model to a biased set of solutions without having fully explored the solution space. This is apparent in the example given in Sec. 3.2 of the main text regarding the training time varying property of α – a solution proposed at the first time step cannot be trusted in terms of its data-fidelity and hence we should not impose regularity. Therefore, we propose that our weighting scheme $\alpha \to 0$ when residual is large and $\alpha \to 1$ as the residual tends to 0. Naturally the data-fidelity residual decreases over time as the training progresses, which we exploit instead of directly making α a function of training time. Thus, when the model converges, α will also converge.

We apply our adaptive weighting scheme to each of the regularization terms (Sec. 3.3 from main text) at each level of the loss pyramid for a total of four levels beginning from the full resolution to one-eighth resolution. Fig. 1 shows the behavior of the adaptive weights on each level of the loss pyramid. The figure was taken from a training process. The recorded value in dark orange is the trend line representing the mean of the weights that is being applied to our solution. We see that as the model improves our adaptive weights proportionally increase, which equivalently impose regularity on the model.

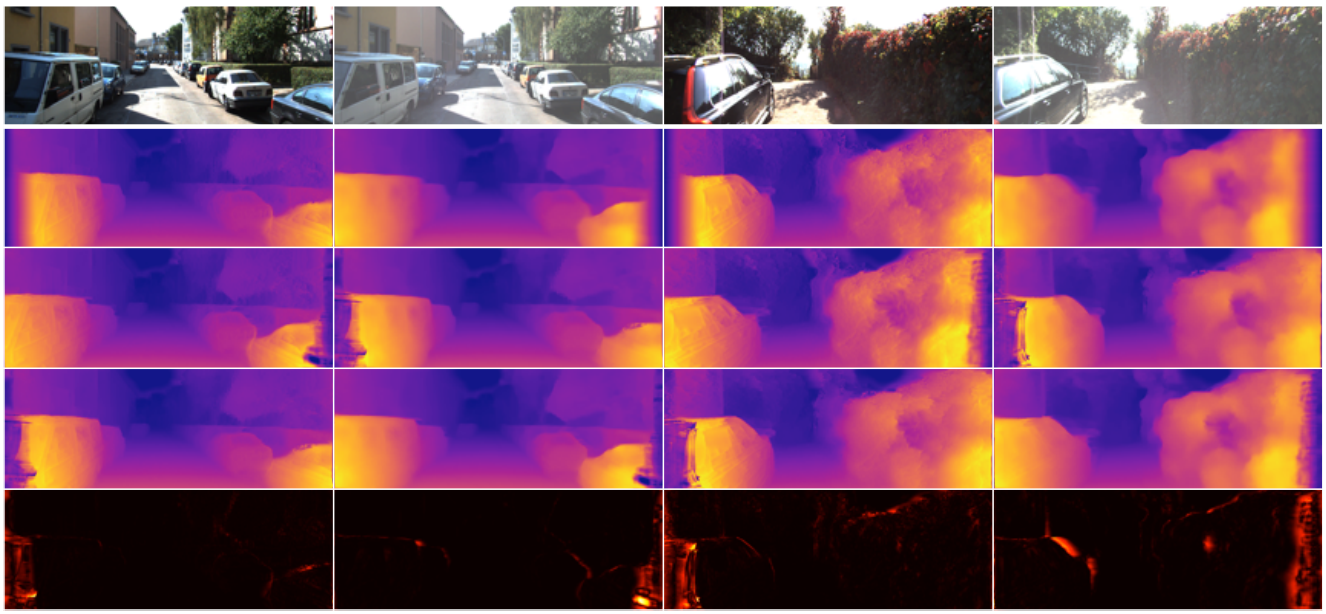

Figure 2: An illustration of our bilateral cyclic constraint with two examples (each example spans two columns). Top to bottom: left image that serves as input to the network with a whitened right image used only in the loss function, initial left and right disparities, projected left and right disparities, reconstructed left and right disparities through back-projection, and absolute difference between the initial disparities and their reconstructions. Our network predicts both the left and right disparities associated with the input left image to enable us to enforce our bilateral cyclic constraint. The right image (whitened) is not given to the network and is only shown to give a frame of reference for the right disparities predicted. From the heat maps (last row), we see that the regions of high intensity (error) is associated with occlusion boundaires (e.g. the missing sections of the cars in the left side of the imagse along with the edges of the cars). In such cases where there are no correspondences (yielding high reconstruction error), our adaptive regularization scheme will discount such regions.

3. Bilateral Cyclic Consistency

In this work, we propose using a bilateral cyclic constraint as a way of regularizing the behavior of the predicted disparities. Our model enforces bilateral cyclic consistency (Sec. 3.3 from main text) by projecting the set of disparities of a given image in a stereo pair to its counter-part and back-projected to itself as a reconstruction (Eqn. 9 from main text). We then apply an $L1$ penalty to the difference between the initial disparities and its reconstruction. By doing so, we are imposing regularity on the disparity map in its original frame of reference as opposed to a relative frame of reference (i.e. left-right consistency). By enforcing the cyclic application of the disparities to be the identity, we constrain that the correspondences found in both images are co-visible regions. In the case of occlusions, the constraint will be violated, yielding high loss. While a solution would be to discount such errors by setting an arbitrary threshold (as attempting to reconstruct a region that does not exist can never yield a correct solution), our adaptive weighting scheme (Sec. 3.2 from main text) provide an elegant solution to such scenarios. As one can never find correct correspondences for occluded regions of two images, this naturally lends to a high data-fidelity residual with which we adaptively discount using α.

We illustrate the cyclic application of the left and right disparities in Fig. 2. Although we are never given the right image (whitened image) of the stereo pair, our network successfully hallucinates its existence and predicts the corresponding disparities to enable our bilateral cyclic consistency check. The areas of inconsistency between the initial disparities and their reconstructions (last row of Fig. 2) exist near the occlusion boundaries, which we use α to effectively guide our model to regularize these regions to improve depth consistency.

4. Ablation Studies on Eigen Split

Table 1 shows an ablation study on the KITTI Eigen split benchmark. The two rows following our full model and our full modeling using the proposed two-branch decoder (Sec. 4 of main paper) denotes the percentage improvement over [2] and [3]. Our full model using a generic encoder with a single-branch decoder outperforms the top performing methods [2, 3]

Method	Error Metrics				Accuracy Metrics		
	Abs Rel	Sq Rel	RMS	logRMS	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
80 Meter Depth Cap							
$ph + st + \lambda^G sm + lr$ (Godard et al. [2])	0.148	1.344	5.927	0.247	0.803	0.922	0.964
Zhan et al. [3] (w/ video)	0.144	1.391	5.869	0.241	0.803	0.928	**0.969**
$ph + st + \alpha\lambda^G sm + \alpha lr$ ([2] w/ Our Adaptive Regularization)	0.145	1.302	5.790	0.245	0.807	0.923	0.965
$ph + st + \lambda^L sm + bc$ (Ours w/o Adaptive Regularization)	0.141	1.266	5.761	0.241	0.811	0.925	0.965
$ph + st + \alpha\lambda^L sm + \alpha lr$ (Ours w/o Bilateral Cyclic Consistency)	0.140	1.290	5.746	0.238	0.817	0.929	0.967
$ph + st + \alpha\lambda^G sm + \alpha bc$ (Ours w/o Bidirectional Edge-Awareness)	0.138	1.191	5.637	0.237	0.817	0.929	0.967
$ph + st + \alpha\lambda^L sm + \alpha bc$ (Ours Full Model)	0.135	1.157	5.556	0.234	0.820	0.932	0.968
Ours (Full Model) % Improvement over [2]	8.8%	13.9%	6.3%	5.3%	2.1%	1.1%	0.4%
Ours (Full Model) % Improvement over [3]	6.3%	16.8%	5.3%	2.9%	2.1%	0.4%	0.1%
$ph + st + \alpha\lambda^L sm + \alpha bc$ * (Ours Full Model w/ 2 Branch Decoder)	**0.133**	**1.126**	**5.515**	**0.231**	**0.826**	**0.934**	**0.969**
Ours (Full Model)* % Improvement over [2]	**10.1%**	**16.2%**	**7.0%**	**6.5%**	**2.9%**	**1.3%**	**0.5%**
Ours (Full Model)* % Improvement over [3]	**7.6%**	**19.1%**	**6.0%**	**4.1%**	**2.9%**	**0.6%**	**0.0%**
50 Meter Depth Cap							
$ph + st + \lambda^G sm + lr$ (Godard et al. [2])	0.140	0.976	4.471	0.232	0.818	0.931	0.969
Zhan et al. [3] (w/ video)	0.135	0.905	4.366	0.225	0.818	0.937	**0.973**
$ph + st + \alpha\lambda^G sm + \alpha lr$ ([2] w/ Our Adaptive Regularization)	0.138	0.957	4.417	0.230	0.824	0.933	0.970
$ph + st + \lambda^L sm + bc$ (Ours w/o Adaptive Regularization)	0.134	0.944	4.389	0.227	0.825	0.934	0.969
$ph + st + \alpha\lambda^L sm + \alpha lr$ (Ours w/o Bilateral Cyclic Consistency)	0.133	0.942	4.351	0.224	0.832	0.937	0.971
$ph + st + \alpha\lambda^G sm + \alpha bc$ (Ours w/o Bidirectional Edge-Awareness)	0.131	0.881	4.265	0.224	0.832	0.937	0.971
$ph + st + \alpha\lambda^L sm + \alpha bc$ (Ours Full Model)	0.128	0.856	4.201	0.220	0.835	0.939	0.972
Ours (Full Model) % Improvement over [2]	8.6%	12.3%	6.0%	5.2%	2.1%	0.9%	0.3%
Ours (Full Model) % Improvement over [3]	5.2%	5.4%	3.8%	2.2%	2.1%	0.2%	0.1%
$ph + st + \alpha\lambda^L sm + \alpha bc$ * (Ours Full Model w/ 2 Branch Decoder)	**0.126**	**0.832**	**4.172**	**0.217**	**0.840**	**0.941**	**0.973**
Ours (Full Model)* % Improvement over [2]	**10.0%**	**14.7%**	**6.7%**	**6.5%**	**2.7%**	**1.1%**	**0.4%**
Ours (Full Model)* % Improvement over [3]	**6.7%**	**8.1%**	**4.4%**	**3.6%**	**2.7%**	**0.4%**	**0.0%**

Table 1: Ablation study on the KITTI Eigen split [4]. We compare each variant of our model to the top performing methods in the monocular depth prediction task. Our full model using a generic single-branch decoder consistently outperforms the best previous methods [2, 3] in all metrics across both depth caps. Each of our partial models improves over the baseline [2] consistently across all metrics and depth caps. In fact, our model without using our bidirectionally informed edge-awareness is already able to exceed the performance of [3] on most metrics, despite not using temporal knowledge and multiple networks as [3] did, with the exception of $\delta < 1.25^3$, where [3] marginally beat us by approximately 0.1%. Our full model using our two-branch decoder (marked by *) outperforms all variants across all metrics and depth caps and is the state-of-the-art. We show the relative percentage boost in performance in all metrics in the two rows following the results of our full model using a single-branch decoder and our full model using a two-branch decoder(*).

across all metrics under both depth caps. Notably, we improve over [2] and [3] by an average of 8.7% and 5.75% in AbsRel, 13.1% and 11.1% in SqRel and even 5.25% and 2.55% in logRMS, respectively. Furthermore, we improve 2.1% over both in $\delta < 1.25$ (the hardest accuracy metric). [3] is only able to outperform our full model marginally in the $\delta < 1.25^3$ metric by 0.1% despite using multiple networks and stereo video streams for training as opposed to stereo pairs.

Moreover, each of our partial models using the generic encoder with the single-branch decoder (same network as [2]) shows improvement over the baseline [2]. Even by simply applying our adaptive regularization to [2], we improve con-

sistently across all metrics. More importantly, our model without bidirectionally informed edge-awareness is already able to outperform [3] on most metrics across both depth caps with the exception of $\delta < 1.25^2$ and $\delta < 1.25^3$ where we are comparable. Based on Table 1, we can see that each of our individual contribution improves the model. There is a significant performance gain when multiple contributions are applied to the model (e.g. adaptive regularization with bilateral cyclic consistency versus [2] with adaptive regularization). The strongest model is produced when all of the contributions are combined as each contribution complements the others to resolve inconsistencies in object boundaries, co-visible and occluded regions.

Furthermore when applying our full model using the proposed two-branch decoder, we further improve over all methods. Specifically, we improve over [2] and [3] by an average of 10.05% and 7.15% in AbsRel, 15.45% and 13.6% in SqRel, 6.85% and 5.2% in RMS, and 6.5% and 3.85% in logRMS, respectively. We even improve 2.8% over both in $\delta < 1.25$. [3] is comparably to us in $\delta < 1.25^3$ metric. Our full model using the proposed two-branch decoder is the state-of-the-art in the unsupervised single image depth prediction task.

5. Qualitative Comparison Between Single-Branch and Two-Branch Decoder

It is well-known that geometry can be recovered in co-visible regions (barring texture-less regions) simply by establishing correspondence between two views of the scene. Our proposed decoder (Fig. 1 from main text) dedicates one branch to learning the necessary features to satisfy data-fidelity, which in this case is the reconstruction between the stereo pairs. In doing so, we also produce an initial solution that satisfies data-fidelity. The second branch then aims to refine such a solution by learning the residual features from the skip connection necessary for adaptively imposing regularity. Fig. 3 gives a qualitative comparison between the single-branch decoder (row 3) and the proposed two-branch decoder (row 4). We see that the two-branch decoder consistently recovers more of the scene geometry, particularly thin structures and distant structures. In cases where a thin structure (e.g. pole) lies close to a larger structure (e.g. walls of building) or a structure is located far away, the single branch decoder often fail to recover their geometry. The two-branch decoder, however, is able to recover distant structures and distinguish thin structures from larger nearby structures.

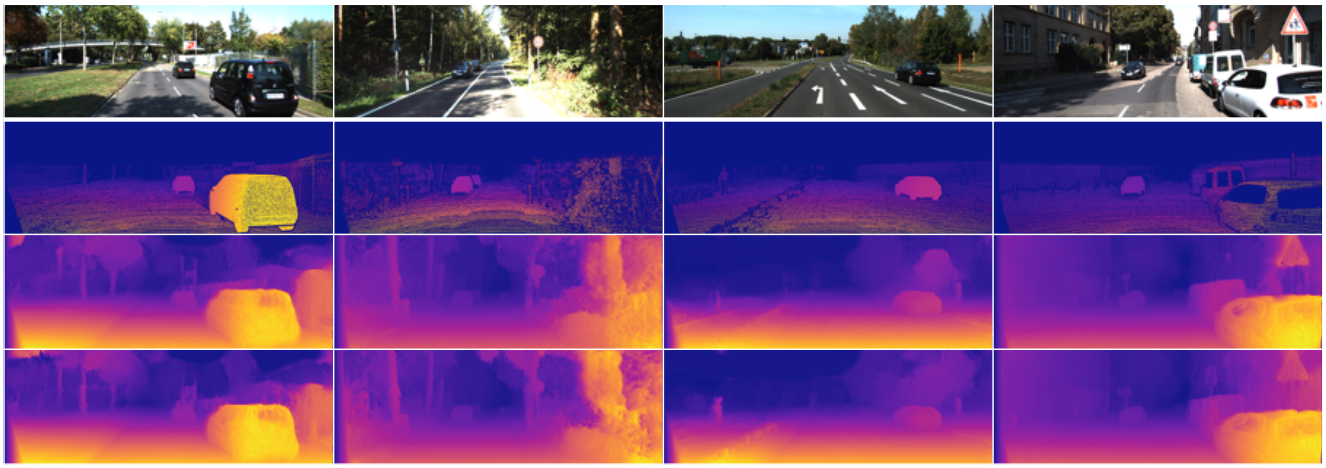

Figure 3: A qualitative comparison between the predictions of a single-branch decoder and the proposed two-branch decoder. Top to bottom: the input image, ground-truth disparities, the results of a single-branch decoder, and the results of the proposed two-branch decoder. The two-branch decoder consistently produces more detailed disparity maps that recovers more of the scene geometry. In the left-most (first) column, while both decoders are able to correct predict the scene globally, the single-branch decoder is unable to recover the details of sign in the distant whereas the two-branch decoder is able to fully recover it. In the second column, the two-branch decoder can recover both of the poles for the sign on the right. In the third column, the two-branch decoder is able to recover the small red pole on the left whereas the single-branch decoder can only recover it partially. In the right-most column, the two-branch decoder is able to recover the small pole on the right next to the wall, the sign in the distance next to the car, the sign above the van and the pole for the sign on the right.

6. Network Architectures

We trained our model using two architectures: 1) a generic encoder (Table 2) based on the VGGnet [5] architecture with a single branch decoder 2) a generic encoder (same as the aforementioned) with our proposed two-branch decoder (Table 3, Sec 4. from main paper).

	kernel		channels		downscale		
layer	size	stride	in	out	in	out	input
Encoder							
conv1	7	2	3	32	1	2	left
conv1b	7	1	32	32	2	2	conv1
conv2	5	2	32	64	2	4	conv1b
conv2b	5	1	64	64	4	4	conv2
conv3	3	2	64	128	4	8	conv2b
conv3b	3	1	128	128	8	8	conv3
conv4	3	2	128	256	8	16	conv3b
conv4b	3	1	256	256	16	16	conv4
conv5	3	2	256	512	16	32	conv4b
conv5b	3	1	512	512	32	32	conv5
conv6	3	2	512	512	32	64	conv5b
conv6b	3	1	512	512	64	64	conv6
conv7	3	2	512	512	64	128	conv6b
conv7b	3	1	512	512	128	128	conv7
Decoder							
upconv7	3	2	512	512	128	64	conv7b
iconv7	3	1	1024	512	64	64	upconv7‖conv6b
upconv6	3	2	512	512	64	32	iconv7
iconv6	3	1	1024	512	32	32	upconv6‖conv5b
upconv5	3	2	512	256	32	16	iconv6
iconv5	3	1	512	256	16	16	upconv5‖conv4b
upconv4	3	2	256	128	16	8	iconv5
iconv4	3	1	128	128	8	8	upconv4‖conv3b
disp4	3	1	128	2	8	8	iconv4
upconv3	3	2	128	64	8	4	iconv4
iconv3	3	1	130	64	4	4	upconv3‖conv2b‖disp4*
disp3	3	1	64	2	4	4	iconv3
upconv2	3	2	64	32	4	2	iconv3
iconv2	3	1	66	32	2	2	upconv2‖conv1b‖disp3*
disp2	3	1	32	2	2	2	iconv2
upconv1	3	2	32	16	2	1	iconv2
iconv1	3	1	18	16	1	1	upconv1‖disp2*
disp1	3	1	16	2	1	1	iconv1

Table 2: Our network architecture follows that of [2] and [6] and we are able to outperform the baseline [2]. "in" and "out" refers to the input and output channels and downscale factor due to striding for each layer. ‖ refers to the concatenation of multiple layers. $*$ refers to up-sampling disparity predictions at a given resolution. Batch normalization was not used.

	kernel		channels		downscale		
layer	size	stride	in	out	in	out	input
Encoder							
conv0	7	1	3	32	1	1	left
conv1	7	2	32	32	1	2	conv0
conv1b	7	1	32	32	2	2	conv1
conv2	5	2	32	64	2	4	conv1b
conv2b	5	1	64	64	4	4	conv2
conv3	3	2	64	128	4	8	conv2b
conv3b	3	1	128	128	8	8	conv3
conv4	3	2	128	256	8	16	conv3b
conv4b	3	1	256	256	16	16	conv4
conv5	3	2	256	512	16	32	conv4b
conv5b	3	1	512	512	32	32	conv5
conv6	3	2	512	512	32	64	conv5b
conv6b	3	1	512	512	64	64	conv6

Table 3: Our proposed network architecture, which achieves state-of-the-art. "in" and "out" refers to the input and output channels and downscale factor due to striding for each layer. $\|$ refers to the concatenation of multiple layers. $*$ refers to up-sampling disparity predictions at a given resolution. The branch prefixed with '`i`' makes the initial prediction and the branch prefixed with '`r`' makes the final prediction.

	kernel		channels		downscale		
layer	size	stride	in	out	in	out	input
Decoder							
iupconv6	3	2	512	512	64	32	conv6b
iconv6	3	1	1024	512	32	32	iupconv6||conv5b
iupconv5	3	2	512	256	32	16	iconv6
iconv5	3	1	512	256	16	16	iupconv5||conv4b
iupconv4	3	2	256	128	16	8	iconv5
iconv4	3	1	256	128	8	8	iupconv4||conv3b
idisp4	3	1	128	2	8	8	iconv4
sconv4	3	1	128	128	8	8	conv3b
sconv4b	3	1	128	128	8	8	sconv4
rskip4	3	1	128	128	8	8	conv3b+sconv4b
rconv4	3	1	258	128	8	8	iconv4||idisp4||rskip4
rdisp4	3	1	128	2	8	8	rconv4
iupconv3	3	2	128	64	8	4	iconv4
iconv3	3	1	130	64	4	4	iupconv3||conv2b||idisp4*
idisp3	3	1	64	2	4	4	iconv3
sconv3	3	1	64	64	4	4	conv2b
sconv3b	3	1	64	64	4	4	sconv3
rskip3	3	1	64	64	4	4	conv2b+sconv3b
rupconv3	3	2	128	64	8	4	rconv4
rconv3	3	1	196	64	4	4	iconv3||idisp3||rupconv3||rskip3||rdisp4*
rdisp3	3	1	64	2	4	4	rconv3
iupconv2	3	2	64	32	4	2	iconv3
iconv2	3	1	66	32	2	2	iupconv2||conv1b||idisp3*
idisp2	3	1	32	2	2	2	iconv2
sconv2	3	1	32	32	2	2	conv1b
sconv2b	3	1	32	32	2	2	sconv2
rskip2	3	1	32	32	2	2	conv1b+sconv2b
rupconv2	3	2	64	32	4	2	rconv3
rconv2	3	1	100	32	2	2	iconv2||idisp2||rupconv2||rskip2||rdisp3*
rdisp2	3	1	32	2	2	2	rconv2
iupconv1	3	2	32	16	2	1	iconv2
iconv1	3	1	18	16	1	1	iupconv1||idisp2*
idisp1	3	1	16	2	1	1	iconv1
sconv1	3	1	32	32	1	1	conv0
sconv1b	3	1	32	32	1	1	sconv1
rskip1	3	1	32	32	1	1	conv0+sconv1b
rupconv1	3	2	64	32	2	1	rconv2
rconv1	3	1	68	16	1	1	iconv1||idisp1||rupconv1||rskip1||rdisp2*
rdisp1	5	1	16	2	1	1	rconv1

References

[1] Hoiem, D., Efros, A.A., Hebert, M.: Recovering surface layout from an image. International Journal of Computer Vision **75**(1) (2007) 151–172

[2] Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: CVPR. Volume 2. (2017) 7

[3] Zhan, H., Garg, R., Weerasekera, C.S., Li, K., Agarwal, H., Reid, I.: Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 340–349

[4] Garg, R., BG, V.K., Carneiro, G., Reid, I.: Unsupervised cnn for single view depth estimation: Geometry to the rescue. In: European Conference on Computer Vision, Springer (2016) 740–756

[5] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

[6] Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: 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. (2016) 4040–4048