HITNet: Hierarchical Iterative Tile Refinement Network for Real-time Stereo Matching
Supplementary Material

Vladimir Tankovich Christian Häne Yinda Zhang Adarsh Kowdle Sean Fanello
Sofien Bouaziz
Google
{vtankovich, chaene, yindaz, adarshkowdle, seanfa, sofien}@google.com

In this supplementary material, we provide additional details, ablation studies, evaluations and visualizations.

1. Training Details

In this section we add additional details regarding the training procedure and discuss difference among datasets.

1.1. Training Setup

The SceneFlow dataset consists of 3 components (Flyingthings, Driving and Monkaa) and comes with a predefined train and test split with ground truth for all examples. Following the standard practice with this dataset we use the predefined train and test split for all experiments. We also used only FlyingThings part of the dataset, as Driving and Monkaa don't have corresponding TEST sets and including them into training hurts accuracy for both Sceneflow and when it's used to pre-train for Middlebury. When all 35k images are used to train a model, the PSM EPE of XL model is 0.41 on "finalpass". We considered random crops of 320×960 and a batch size of 8, and a maximum disparity of 320. We trained for 1.42M iterations using the Adam optimizer, starting from a learning rate of $4e^{-4}$, dropping it to $1e^{-4}$, then to $4e^{-5}$, then to $1e^{-5}$ after 1M, 1.3M, 1.4M iterations respectively. The general robust loss for SceneFlow experiments was applied with, $\alpha = 0.9, c = 0.1$. For all other experiments, $\alpha = 0.8, c = 0.5$.

For real world datasets such as KITTI 2012 and 2015 a training set with ground truth and a test set where the ground truth is not available is provided. For the benchmark submission we trained the network on all 394 images available from both datasets. For ablation studies on the KITTI dataset we split training set into a train and validation set with 75% of the data in the training set and 25% of the data in the validation set. We trained with data augmentation, batch-size of 4 and random crops of 311×1178 and a maximum disparity of 256. The training schedule followed the following step: 400k iterations with learning rate $4e^{-4}$, followed by 8k iterations with learning rate $1e^{-4}$, followed by 2k iterations with learning rate $4e^{-5}$. Note that the network is not pre-trained on any other datasets as in [15], and a small training set is sufficient for our method to achieve good performance.

Indeed, empirically we found that using a small initial learning rate $1e^{-4}$ and training for longer achieves the best results on multiple datasets without showing sign of overfitting. In Figure 2 we show the evolution of the training HITNet L for more than 200 epochs (learning rate change to $1e^{-5}$ after 200 epochs) on the SceneFlow cleanpass dataset. We also compared this scheme with using a higher starting learning rate ($1e^{-3}$): after 10 epochs we observed EPE of 0.85 for $1e^{-4}$ and 0.66 for $1e^{-3}$. Although $1e^{-3}$ achieved smaller error within a few epochs, our experiments confirm that longer training with a small learning rate is beneficial to achieve higher quality results without overfitting. See also generalization experiment showing that the method has very good cross-dataset performance.

The training set for the real world ETH3D stereo dataset [11] contains just a few stereo pairs, so additional data is needed to avoid overfitting. For the benchmark submission we trained the network on all 394 images from both KITTI datasets, as well as all half and quarter resolution training images from Middlebury dataset V3 [10] and training images from ETH3D dataset. We used the same training parameters as for KITTI submission and stopped training after 115k iterations, which was picked using 4 fold cross-validation on ETH3D training set. Note that there is no additional training, pre-training, finetuning.

Similarly, the Middlebury dataset [10] contains a limited training set. To avoid overfitting, we pre-trained the model on SceneFlow's FlyingThings TRAIN set with data augmentation, then fine-tuned on the 23 Middlebury14-perfectH training images, while keeping all data augmentations on. Specifically, we used a HITNet Large model with initialization at 6 scales (M=5), pre-trained it for 445k itera-

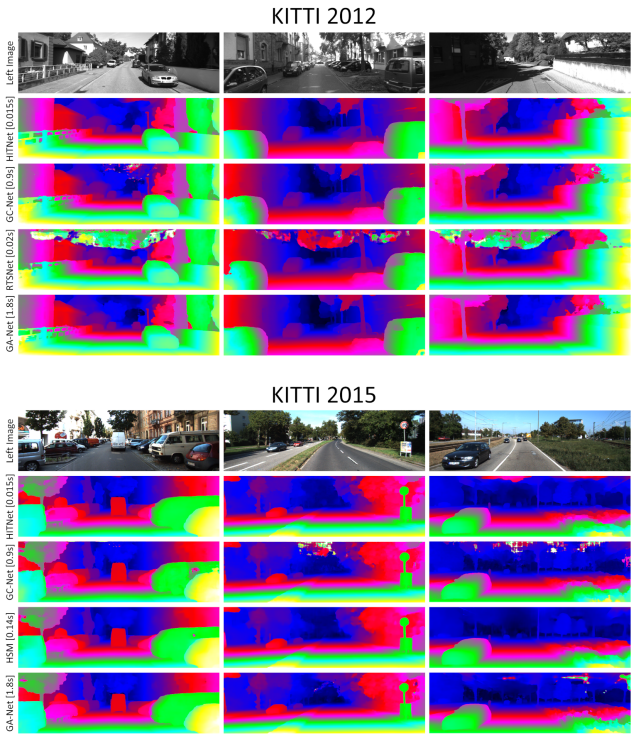

Figure 1: Qualitative Results on KITTI 2012 and 2015. Note how HITNet is able to recover fine structures and crisp edges using a fraction of the computational cost required by other competitors.

tions, using batch size of 8 and random crops of 512×960. We initialize the learning rate to $4e^{-4}$, then gradually drop it to $1e^{-4}$, $4e^{-5}$ and $1e^{-5}$ after 300K, 400K, 435K iterations respectively. Finally, we fine-tuned the model for 5K iterations at $1e^{-5}$ learning rate. These parameters were selected by using 4-fold cross validation on Middlebury training set.

1.2. Data Augmentation

The training data available may not be fully representative of the actual test sets for small real world datasets such as KITTI, ETH3D and Middlebury. Indeed, we often observed substantial differences at test time, such as changes in brightness, unexpected reflections and mis-calibrations. In order to improve the network robustness we performed the following augmentations. We first perturb the brightness and contrast of left and right images by using random symmetric and asymmetric multiplicative adjustments. Symmetric adjustments are sampled within [0.8, 1.2] interval and asymmetric between [0.95, 1.05]. Similar to [16], We then replace random areas of the right image with random crops taken from another portion of the right image: this helps the network to deal with occluded areas and encourages a better "inpainting". The crop size to be replaced is randomly sampled between [50,50] and [180, 250].

Finally, the Middlebury images contains a substantially different color distribution compared to other datasets. To mitigate this we used the approach from [12] that brings color distribution of training images closer to that of Middlebury set and during test time we normalize color distribution between left and right images of a stereopair. Additionally, similar to [16], in order to deal with miscalibrated pairs of this dataset, we augmented the training data with random y offset between $[-2, 2]$ pixels. The random values for y offset are generated at a low resolution [H/64, W/64], and then bilineraly up-sampled to full resolution [H, W] of the input image. To simulate different noise levels images with different exposure contain, we add Gaussian random noise with variance sampled between [0 and 5] intensity levels once for the whole image.

2. Additional Evaluations

In this section we show additional qualitative results on real-world datasets. In Figure 1 we show comparisons of our method with other approaches. We consider multiple representative competitors such as: GC-Net [5], which uses the full cost volume and 3D convolutions to infer context, RTS-Net [6] that has similar inference time than HITNet, and finally GA-Net [17], as one of the best performing methods in terms of accuracy.

Our method compares very favorably to other approaches such as GC-Net and fast methods like RTSNet and is on par with the state-of-the-art approaches, e.g. GA-Net [17]. Note how our method retrieves fine structures and crisp edges, while only training on the KITTI datasets, which exhibit significant edge fattening artifacts.

Similarly, in Figure 4 we show qualitative results on the Middlebury dataset [10]. For each image, we compare HITNet with the best performing competitor on the Bad 0.5 metric. Note how our method is able to produce crisp edges, correct occlusions and thin structures in all the considered cases.

2.1. Intermediate Outputs

We show intermediate outputs from within our network in Fig 3. We observe that with increasing resolution the disparity gets more fine grained and the details from the higher resolution initialization gets merged into the global context that is coming from the lower resolutions. Note that our results on the KITTI 2015 dataset are only trained on the KITTI datasets from scratch without any pre-training on other data sources. This means the network has not been supervised on the top one third of the image as these datasets do only provide ground truth for the bottom two thirds of the image.

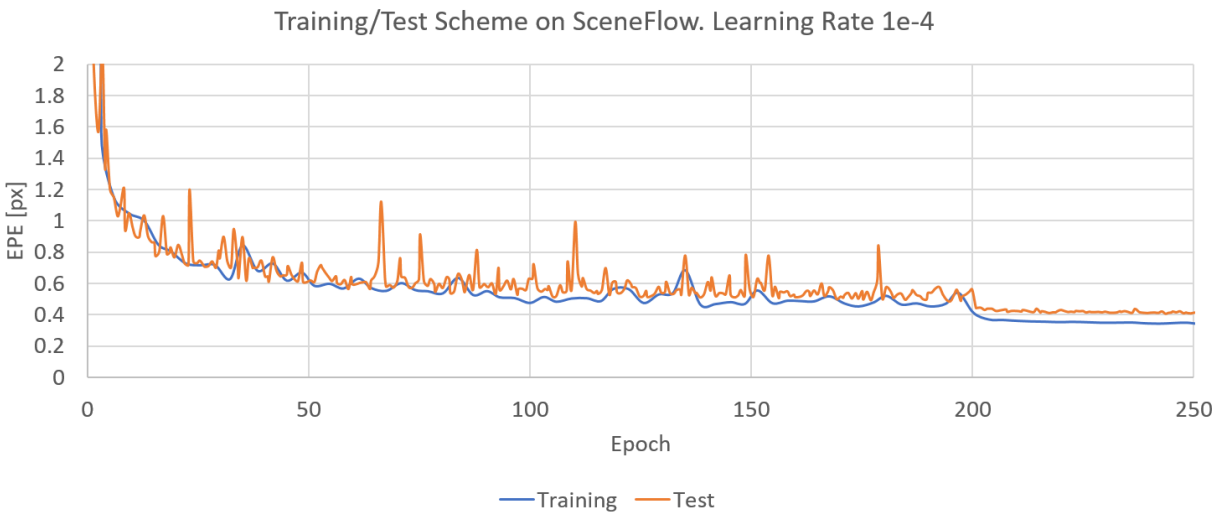

Figure 2: We show the evolution of the training reporting the EPE on training and test set respectively. Note how the scheme reduces the error on both training and test set without showing signs of overfitting.

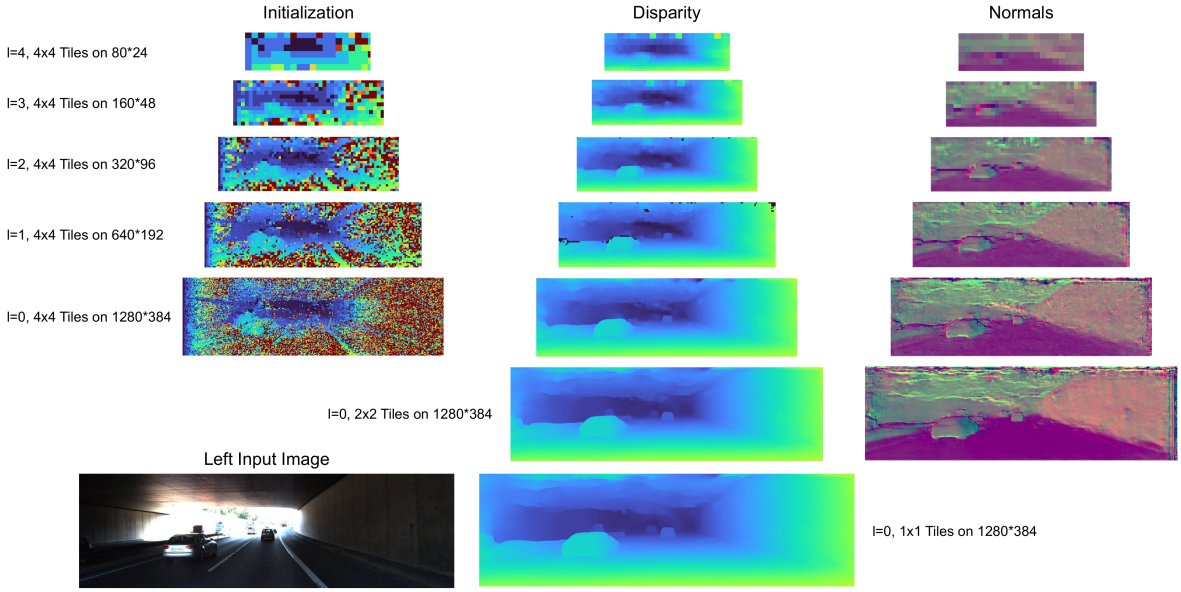

Figure 3: Intermediate results of our network on the left side we show the disparity maps that the matching of the initialization stage provides. On the right hand side we show the final disparity and normals for each resolution. The final two resolutions are 2x2 and 1x1 tiles of the highest resolution feature map, while the initialization is always computed on 4x4 tiles of the feature maps.

2.2. Generalization.

We finally demonstrate the cross-domain adaptation capabilities of our method. Following the protocol in [13], we trained HITNet on SceneFlow with data augmentations and tested on KITTI 2012 and KITTI 2015 respectively. We also considered multiple competitors as in [13] and report the results in Tab. 1: note how our method shows superior generalization results compared to all the other state-of-the-art approaches. This shows that our method is able to effectively generalize to unseen dataset even without explicit fine-tuning.

Dataset	HITNet	CRL [9]	iResNet [7]	PSMNet [2]	EdgeStereo [13]
KITTI 2012 EPE	1.06	1.38	1.27	5.54	1.96
KITTI 2012 $> 3px$	6.44	9.07	7.89	27.33	12.27
KITTI 2015 EPE	1.36	1.35	1.21	6.44	2.06
KITTI 2015 $> 3px$	6.49	8.88	7.42	29.86	12.46

Table 1: Generalization Experiment. We trained each method on SceneFlow with data augmentation and tested on KITTI 2012 and 2015. Note how our method outperforms the others.

	SceneFlow finalpass [8]				KITTI 2012 [4]		
Model	EPE	0.1 px	1 px	3 px	EPE	2 px	3 px
HITNet	0.529 px	24.0 %	5.52 %	3.00 %	0.484 px	2.91 %	2.00 %
4 Scales	-	-	-	-	0.507 px	3.10 %	2.20 %
No Multi-scale	-	-	-	-	0.747 px	4.76 %	3.62 %
4x4x4 Downsampled	0.561 px	26.4 %	5.88 %	3.15 %	0.526 px	3.16 %	2.19 %
16x16x8 Downsampled	0.615 px	27.5 %	6.36 %	3.39 %	0.536 px	3.35 %	2.30 %
16x16x1 Downsampled	0.651 px	31.9 %	7.28 %	3.62 %	0.554 px	3.60 %	2.51 %
No Warping	0.588 px	31.6 %	5.88 %	3.10 %	0.602 px	3.72 %	2.54 %
No Slant Prediction	0.548 px	25.2 %	5.74 %	3.08 %	0.513 px	3.23 %	2.18 %
No Tile Features	0.538 px	24.7 %	5.64 %	3.01 %	0.488 px	3.04 %	2.06 %
HITNet L	0.43 px	20.7 %	4.70 %	2.57 %	0.490 px	2.98 %	2.13 %
HITNet XL	0.36 px	18.2 %	4.09 %	2.21 %	0.492 px	3.11 %	2.20 %

Table 2: Ablation study of the proposed HITNet on SceneFlow [8] and KITTI 2012 [4] datasets. Lower is better.

2.3. Ablation Study

We analyze the importance of our proposed components. The full HITNet is considered as baseline and compared with a version where features are removed. The ablation study is performed on the SceneFlow "finalpass" data and KITTI 2012. See Figure 5 for a qualitative evaluation.

Multi Scale Prediction. The multi-scale feature affects both initialization and propagation stages. In Tab. 2, we report the results for the full model (HITNet) on KITTI 2012, with 5 scales, results for 4 scales and finally we removed the multi-resolution prediction completely. When we evaluated the same settings on the synthetic SceneFlow dataset we did not find a substantial differences between a single scale or multiple ones: clearly the synthetic dataset contains much more textured regions that do not benefit of additional context during propagation, whereas real world scenarios are full of textureless scenes (e.g. walls), where the multi-resolution approach is naturally performing better.

4**x**4**x**4 **Downsampled.** Initialization at full disparity resolution provides a compelling starting point to the network, which can focus mostly on refining the prediction. In Tab. 2 we show that using tile resolution for disparity (cost volume is 4X downsampled in H, W and D dimensions), the accuracy substantially drops. This demonstrates the importance of our proposed fast high resolution initialization.

16**x**16**x**8 **Downsampled.** Decreasing the resolution of the cost volume for all dimensions similar to [14] degrades accuracy (16X downsampled in H and W, 8X in D).

16**x**16**x**1 **Downsampled.** Using larger tiles, while maintaining disparity resolution degrades accuracy even more, as the network is not able to reason about precise disparity at low spatial resolution during initialization.

Slant Prediction. In this experiment, we forced tile hypotheses to always be fronto parallel by setting d_x and d_y to 0 and using bilinear interpolation for upsampling. As showed in Tab. 2, removing the slant prediction leads to a substantial drop in precision for both SceneFlow and KITTI 2012. Moreover the network loses its inherent capability of predicting some notion of surface normals that can be useful for many applications such as plane detection.

Tile Features. Here we removed the additional features predicted on each tile during the initialization and propagation steps. This turns out to be a useful component and without it we observe a decrease in accuracy for both datasets.

Warping. The image warps are used to compute the matching cost during the propagation. Removing this step hurts the subpixel precision as demonstrated in Tab. 2.

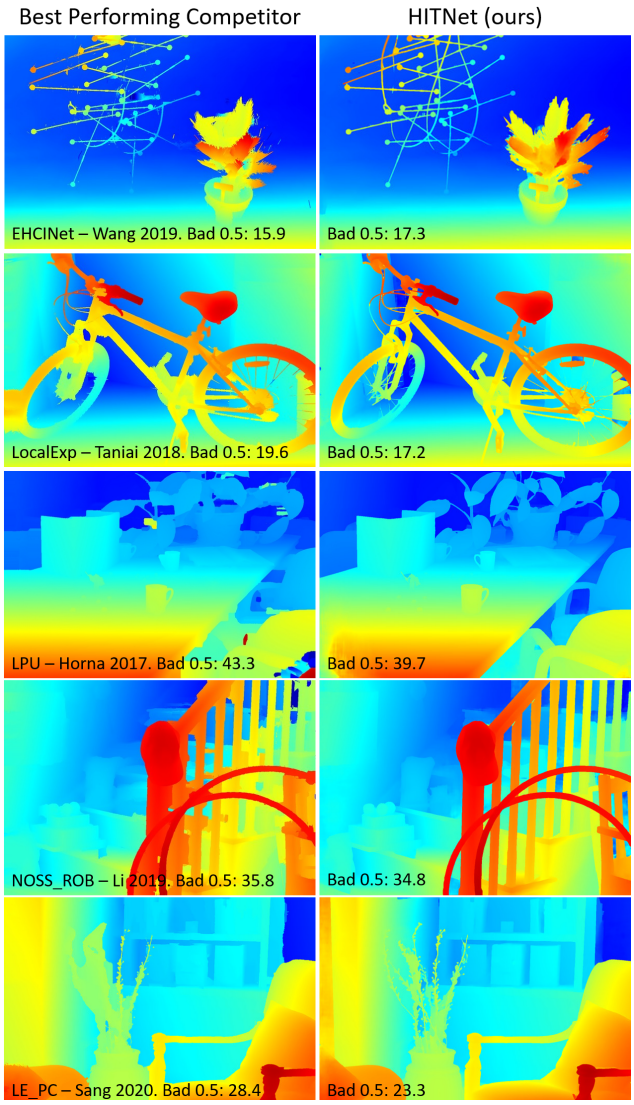

Figure 4: Qualitative comparisons on Middlebury dataset. For each image we compare our method with the best performing competitor following the Bad 0.5 metric. Note how our method is able to produce crisp edges, correct occlusions and thin structures in all the considered cases.

Model Size. Finally, we tested if an increase in the model size is beneficial or not. In particular we double the channels in the feature extractor, and use 32 channels and 6 residual blocks for the last 3 propagation steps, this resorts to a run-time increase to 54ms. As expected this has an improvement on SceneFlow as reported in Tab. 2, HITNet Large; however for the small KITTI datasets this did not improve performance due to over-fitting. Further increasing model size by using 64 channels for the last 3 propagation steps improved SceneFlow results, increased runtime to 114ms, and increased over-fitting on a smaller dataset. We don't see a reason to explore larger model sizes on a synthetic dataset as it will add to over-fitting on smaller real datasets that are publicly available. The metrics on "clean-pass" for XL version are: 0.31 epe, 15.6 bad 0.1, 3.67 bad 1.0, 1.99 bad 3.0.

3. Model Architecture Details

By default, the HITNet architecture is implemented with a 5-scale feature extractor with $16, 16, 24, 24, 32$ channels at corresponding resolutions. During Initialization step the first convolution over 4×4 tiles outputs 16 channels, followed by 2-layer MLP with 32 and 16 channels and ReLU non-linearities. Tile descriptor has 13 channels by default, residual blocks use 32 channels, unless mentioned otherwise. Each intermediate propagation steps use 2 residual blocks without dilations. At each spatial resolutions, the propagation module uses feature maps from appropriate scale: full-resolution feature maps for 4×4 tiles, 2X downsampled for coarser tiles that have size 8×8 in full resolution, but sample 4×4 pixels in coarser feature map, etc till 16X downsampled and 64×64 tiles in original resolution. The last 3 levels of propagation start at 4×4 tiles and progressively in-paint and refine strong correct disparity at the edges over larger regions. To achieve that, they operate on coarse feature maps: the 4×4 tiles use 4X downsampled features for warping, the 2×2 tiles use 2X downsampled features for warping, the 1×1 tiles use full-resolution features for warping.

In HITNet model used in KITTI and ETH3d experiments last 3 propagation steps use $4, 4, 2$ residual blocks with $32, 32, 16$ channels and $1, 3, 1, 1$; $1, 3, 1, 1$; $1, 1$ dilations.

HITNet model used in Sceneflow experiments uses $16, 16, 24, 24, 32$ channels for feature extractor. A single initialization at 4x4 tiles. Last 3 propagation steps use $6, 6, 6$ residual blocks with $32, 32, 16$ channels and $1, 2, 4, 8, 1, 1$ dilations.

HITNetL model used in Sceneflow experiments uses $32, 40, 48, 56, 64$ channels for feature extractor. A single initialization at 4x4 tiles. Last 3 propagation steps use $6, 6, 6$ residual blocks with $32, 32, 32$ channels and $1, 2, 4, 8, 1, 1$ dilations.

HITNetXL model used in Sceneflow experiments uses $32, 40, 48, 56, 64$ channels for feature extractor. A single initialization at 4x4 tiles. Last 3 propagation steps use $6, 6, 6$ residual blocks with $64, 64, 64$ channels and $1, 2, 4, 8, 1, 1$ dilations.

HITNet model used in Middlebury experiments uses $32, 40, 48, 56, 64$ channels for feature extractor. Last 3 propagation steps use $6, 6, 6$ residual blocks with $32, 32, 32$ channels and $1, 2, 4, 8, 1, 1$ dilations.

The models used for submission for benchmarks and scripts to run them are available at
https://github.com/google-research/google-research/tree/master/hitnet

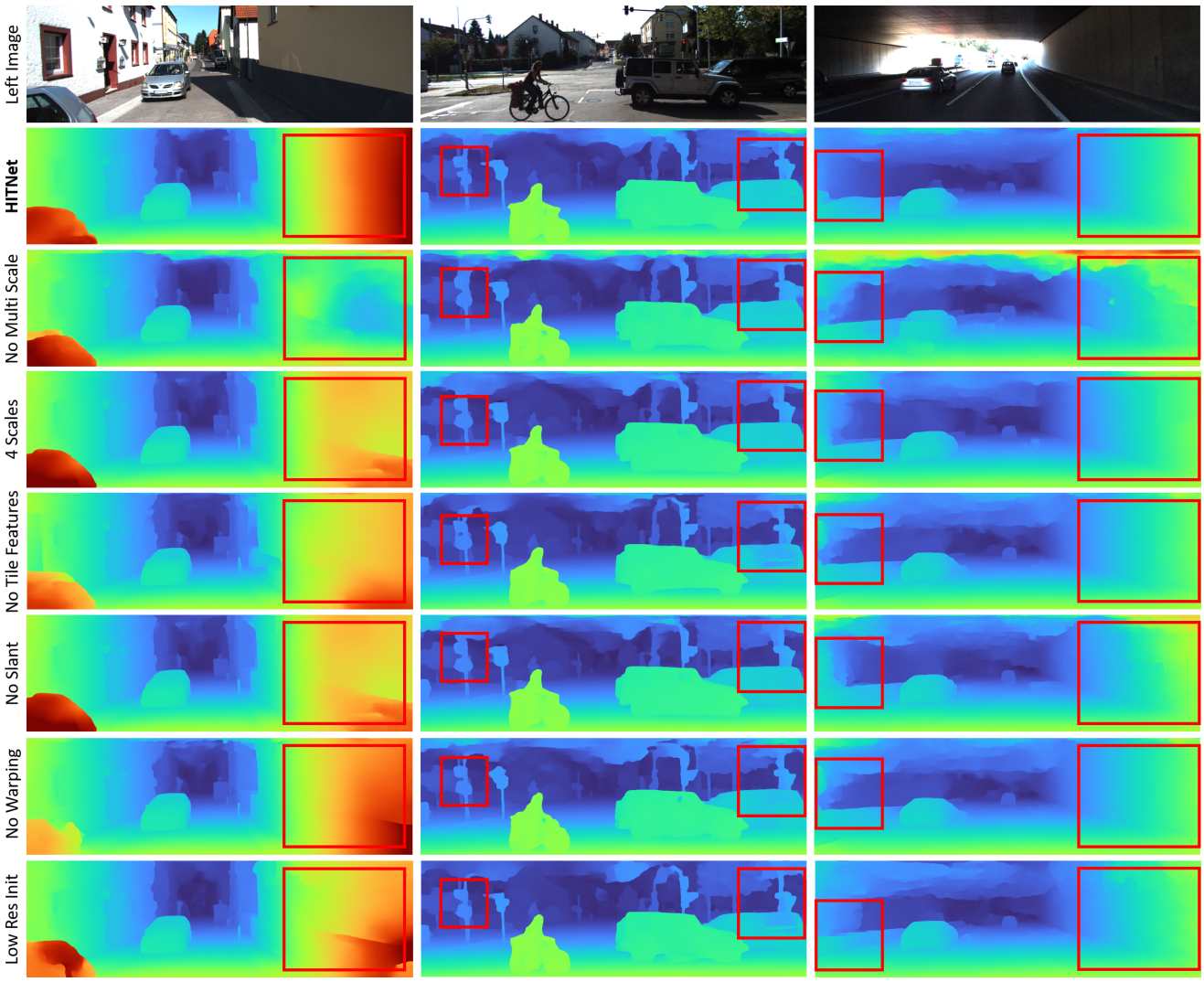

Figure 5: Ablation study, qualitative evaluation. Note how our HITNet model relies on all proposed design choices in order to achieve the best results on fine details, edges and occluded regions.

Fig 6 provides more details for the initialization module described in the main paper. Similarly Fig 7 and Fig 8 show how resblocks are integrated into propagation logic. Finally, Fig 9 depicts differences between propagation steps when a single hypothesis or multiple hypotheses are used.

4. Runing Time Details

The HITNet architecture used for ETH3d and KITTI experiments runs at 19ms per frame on a Titan V GPU for 0.5Mpixel (KITTI resolution) input images. The majority of the time is spent during the last 3 propagation steps (7.5 ms) that operate on higher resolutions. The multi-scale propagation steps use down-sampled data and contribute less than 5ms. Efficient implementation of initialization using a single fused Op generates initial disparity estimates across all resolutions in 0.25ms, with feature extractor contributing 6ms. For Middlebury experiments the model has a run-time of approximately 107.5ms per Mpix of input resolution using custom CUDA operations for initialization and warping, when maximum disparity is 160 and the run-time scales linearly with resolution. The run-time has a small increase with disparity range, and is about 109ms per Mpix for a maximum disparity of 1024. Without custom CUDA operations the run-time is increased by a factor of 3, as a single warping operation contains more than a hundred simple operations over large tensors, and while it's trivial to fuse them together, not doing so results in most of the time spent on global memory access. When tested on an 18-core Xeon 6154 CPU, the default tensorflow runtime runs 3.3s per Mpix, which would translate to about $60s$ for a single threaded runtime, which compares favourably to other CPU

Model	Param	GMac	EPE	1 Pixel Threshold Error Rates
GC-Net [5]	2.9M [17]	8789 [1]	1.80 [17]	15.6 [17]
PSMNet [2]	3.5M [17]	2594 [1]	1.09 [17]	12.1 [17]
GANet [17]	2.3M [17]	-	0.84 [17]	9.9 [17]
StereoDRNet [1]	-	1410 [1]	0.98 [1]	-
LEAStereo [3]	1.81M [3]	782 [3]	0.78 [3]	7.82 [3]
HITNet	0.45M	48	0.53	5.52
HITNetL	0.97M	146	0.43	4.56
HITNetMiddlebury	1.62M	187(450 for 1.57Mpix input)	-	-
HITNetXL	2.07M	375	0.36	4.09

Table 3: Comparisons of number of parameters and GMacs (Giga Multiply-accumulate operations) with other methods on Scene Flow "finalpass" dataset. The numbers were partially adopted from the papers cited in the table. The lower the better.

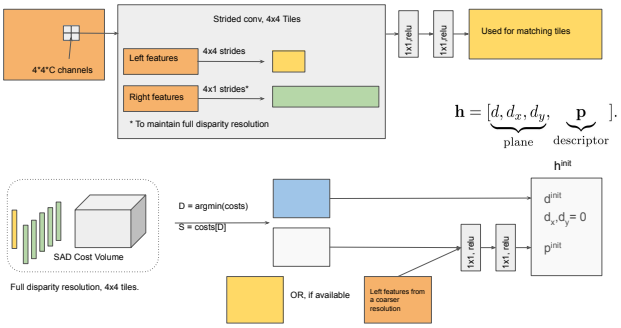

Figure 6: Initialization: The features extracted by the feature extractor are matched and inital tile features are computed. The number of feature channels C depends on feature extractor architecture and the current level (see Sec. 3)

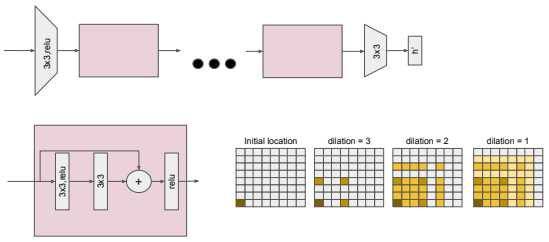

Figure 8: ResNet blocks. First 3x3 convolution generates a local state for the tile, which is incrementally updated by each block. The final state generates tile updates. Each block may use dilated convolutions to increase the speed of diffusion.

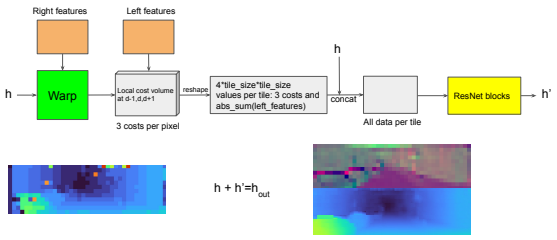

Figure 7: Propagation, for the single hypothesis case.

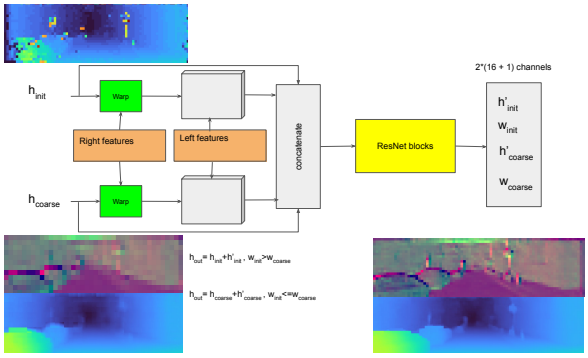

Figure 9: Propagation with multiple hypotheses.

methods. The CPU tensorflow runtime does make use of SIMD instruction set, which other methods may not utilize.

5. Number of Parameters

An important aspect of efficient neural network architectures is the number of parameters they have. This will influence the amount of compute required and the amount of memory needed to store them. Moreover, being able to achieve good performance with fewer numbers of parameters makes the network less susceptible to over-fitting. In Tab. 3 we show that our network is able to achieve better results than other approaches with a significantly lower number of parameters and compute.

Having less parameters also increases the generalization capabilities of the proposed method: indeed less learnable weights implies that the network is less prone to overfitting - our approach is able to outperform multiple state-of-the-art baselines when trained on synthetic data and tested in real-world scenarios.

References

[1] Rohan Chabra, Julian Straub, Christopher Sweeney, Richard Newcombe, and Henry Fuchs. StereoDRNet: Dilated residual stereonet. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 7

[2] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4, 7

[3] Xuelian Cheng, Yiran Zhong, Mehrtash Harandi, Yuchao Dai, Xiaojun Chang, Tom Drummond, Hongdong Li, and Zongyuan Ge. Hierarchical neural architecture search for deep stereo matching. In *NIPS*, 2020. 7

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 4

[5] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 2, 7

[6] H. Lee and Y. Shin. Real-time stereo matching network with high accuracy. In *IEEE International Conference on Image Processing (ICIP)*, 2019. 2

[7] Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Wei Chen, Linbo Qiao, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. 2018. 4

[8] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4

[9] Jiahao Pang, Wenxiu Sun, JS Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *International Conference on Computer Vision-Workshop on Geometry Meets Deep Learning (ICCVW 2017)*, 2017. 4

[10] Daniel Scharstein, Heiko Hirschmuller, York Kitajima, Greg Krathwohl, Nera Nesic, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German Conference on Pattern Recognition (GCPR)*, 2014. 1, 2

[11] Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[12] Xiao Song, Guorun Yang, Xinge Zhu, Hui Zhou, Zhe Wang, and Jianping Shi. Adastereo: A simple and efficient approach for adaptive stereo matching. Technical report, arXiv preprint arXiv:2004.04627, 2020. 2

[13] Xiao Song, Xu Zhao, Liangji Fang, Hanwen Hu, and Yizhou Yu. Edgestereo: An effective multi-task learning network for stereo matching and edge detection. *International Journal of Computer Vision (IJCV)*, 2020. 3, 4

[14] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: Cnns for optical flow using pyramid, warping, and cost volume. *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 4

[15] Gengshan Yang, Joshua Manela, Michael Happold, and Deva Ramanan. Hierarchical deep stereo matching on high-resolution images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[16] Gengshan Yang, Joshua Manela, Michael Happold, and Deva Ramanan. Hierarchical deep stereo matching on high-resolution images. In *CVPR*, pages 5510–5519, 06 2019. 2

[17] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 7