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

Vladimir Tankovich, Christian Häne, Sean Fanello, Yinda Zhang, Shahram Izadi, Sofien Bouaziz
Google

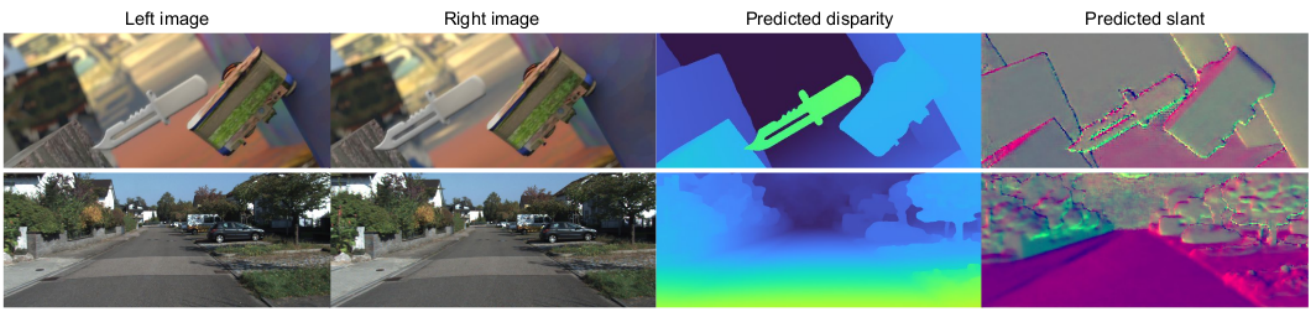

Figure 1: Example result of HITNet on the SceneFlow and KITTI datasets. HITNet ranks 1st-3rd on all the ETH3D metrics and ranks 1st on the KITTI 2012 and 2015 benchmarks among the published methods faster than 100ms.

Abstract

This paper presents HITNet, a novel neural network architecture for real-time stereo matching. Contrary to many recent neural network approaches that operate on a full cost volume and rely on 3D convolutions, our approach does not explicitly build a volume and instead relies on a fast multi-resolution initialization step, differentiable 2D geometric propagation and warping mechanisms to infer disparity hypotheses. To achieve a high level of accuracy, our network not only geometrically reasons about disparities but also infers slanted plane hypotheses allowing to more accurately perform geometric warping and upsampling operations. Our architecture is inherently multi-resolution allowing the propagation of information at different levels. Multiple experiments prove the effectiveness of the proposed approach at a fraction of the computation required by recent state-of-the-art methods. At time of writing, HITNet ranks 1st-3rd on all the metrics published on the ETH3D website for two view stereo and ranks 1st on the popular KITTI 2012 and 2015 benchmarks among the published methods faster than 100ms.

1. Introduction

Recent research in computational binocular stereopsis has largely focused on developing accurate but computationally expensive deep learning approaches. Large convolutional neural networks (CNNs) can often use up to a second or even more to process an image pair and infer a disparity map. For active agents such as mobile robots or self driving cars such a high latency is undesirable and methods which are able to process an image pair in a matter of milliseconds are required instead. Despite this, only 4 out of the top 100 methods on the KITTI 2012 leaderboard are published approaches that take less than 100ms [1].

A common pattern in end-to-end learning based approaches to computational stereo is utilizing a CNN which is largely unaware of the geometric properties of the stereo matching problem. In fact initial end-to-end networks were based on a generic U-Net architecture [36]. Subsequent works have pointed out that incorporating explicit matching cost volumes encoding the cost of assigning a disparity to a pixel, in conjunction with 3D convolutions provides a notable improvement in terms of accuracy but at the cost of significantly increasing the amount of compu-

[1] Additional approaches faster than 100ms are on the leaderboard but the algorithms are unpublished and hence it is unknown how the results were achieved.

tation [23]. Follow up work [24] showed that a downsampled cost volume could provide a reasonable trade-off between speed and accuracy. However, the downsampling of the cost volume comes at the price of sacrificing accuracy. Multiple recent stereo matching methods [49, 11, 27] have increased the efficiency of disparity estimation for active stereo while maintaining a high level of accuracy. These methods are mainly built on three intuitions: ① the use of compact/sparse features for fast high resolution matching cost computation; ② very efficient disparity optimization schemes that do not rely on the full cost volume; ③ iterative image warps using slanted planes to achieve high accuracy by minimizing image dissimilarity. All these design choices are used without explicitly operating on a full 3D cost volume. By doing so these approaches achieve very fast and accurate results for active stereo but they do not directly generalize to passive stereo due to the lack of using a powerful machine learning system. This therefore raises the question if such mechanisms can be integrated into neural network based stereo-matching systems to achieve efficient and accurate results opening up the possibility of using passive stereo based depth sensing in latency critical applications.

We propose HITNet, a framework for neural network based depth estimation which overcomes the computational disadvantages of operating on a 3D volume by integrating image warping, spatial propagation and a fast *high resolution initialization step* into the network architecture, while keeping the flexibility of a learned representation by allowing features to flow through the network. The main idea of our approach is to represent image tiles as planar patches which have a learned compact feature descriptor attached to them. The basic principle of our approach is to fuse information from the high resolution initialization and the current hypotheses using spatial propagation. The propagation is implemented via a convolutional neural network module that updates the estimate of the planar patches and their attached features. In order for the network to iteratively increase the accuracy of the disparity predictions, we provide the network a local cost volume in a narrow band (± 1 disparity) around the planar patch using in-network image warping allowing the network to minimize image dissimilarity. To reconstruct fine details while also capturing large texture-less areas we start at low resolution and hierarchically upsample predictions to higher resolution. A critical feature of our architecture is that at each resolution, matches from the initialization module are provided to facilitate recovery of thin structures that cannot be represented at low resolution. An example output of our method shows how our network recovers very accurate boundaries, fine detail and thin structures in Fig. 1.

To summarize, our main contributions are: ① a fast multi-resolution initialization step that is able to compute *high resolution matches* using learned features; ② an efficient, disparity propagation stage that makes use of slanted support windows with learned descriptors; ③ competitive results in popular benchmarks using a fraction of the computation compared to state-of-the-art methods; ④ very compelling cross-dataset (generalization) results from synthetic data to real-world images.

2. Related Work

Stereo matching has been an active and vibrant field of research for decades [35, 42, 18]. *Traditional methods* utilize hand-crafted schemes to find reliable local correspondences [52, 21, 4, 20] and global optimization algorithms to exploit context when matching [13, 25, 26]. The run-time efficiency of most of these approaches are inherently correlated with the size of the disparity space, which prevents using them for real-time applications.

Efficient algorithms [29, 33, 5, 3] have been proposed that avoid searching the full disparity space by using patchmatch [1] and super-pixel [29] techniques. A family of machine learning based approaches, using random forest and decision trees, have also been developed to quickly establish correspondences [9, 10, 11, 12]. However, these methods require either camera specific learning or post processing.

Recently, *deep learning methods* brought big improvements to stereo matching. Early work trained siamese networks to extract patch-wise features and/or predict matching costs [34, 55, 53, 54]. More recently, end-to-end networks have been proposed to learn these steps jointly, yielding more accurate results [45, 36, 39]. A key component in many modern architectures is the design of a cost volume layer [23] (or correlation layer [22]), allowing the network to run per-pixel feature matching. To speed up computation, cascaded models [38, 7, 31, 15, 50, 56] have been proposed to search in disparity space in a coarse-to-fine fashion. In particular, [38] uses multiple residual blocks to improve the current disparity estimate at the cost of a substantial amount of computation. The recent work of [50] relies on a hierarchical cost volume, allowing the method to be trained on high resolution images and generate different resolutions *on demand*. All these methods rely on expensive cost-volume filtering operations using 3D convolutions [50] or multiple refinement layers [38], which make these methods far from real-time performance, i.e. 30fps. Fast approaches [24, 57] aggressively *downsample* the cost volume in both spatial and disparity space and attempt to recover fine details by multiple edge-aware upsampling layers. Although these methods show real-time performance they sacrifice accuracy especially for thin structures and edges.

Our method is highly inspired by classical stereo matching methods, which aim at propagating good sparse matches [11, 12, 49]. In particular, Tankovich et al. [49] proposed a hierarchical algorithm that makes use of slanted support

windows to amortize the matching cost computation in *tiles*. Inspired by this work, we propose an end-to-end approach that overcomes the issues of the hand-crafted algorithms, while maintaining computational efficiency.

PWC-Net [48], although designed for optical flow estimation, is related to our approach. The method uses a *low resolution* cost volume with multiple refinement stages via image warps and local matching cost computations. Thereby following the classical pyramidal matching approach where a low resolution result gets hierarchically upsampled by initializing the current level with the previous level's solution and gets refined using higher resolution matching locally. In contrast we propose a fast, multi-scale, *high resolution* initialization which is able to recover fine details that cannot be represented at low resolution. Finally, our refinement steps produce local slanted plane approximations, which are used to predict the final disparities, as opposed to standard bilinear warping and interpolation employed in [48].

3. Method

The design of the proposed approach, follows the principles of traditional stereo matching methods [42]. In particular, we observe that recent efficient methods rely on the three following steps: ① compact feature representations are extracted [11, 12]; ② a high resolution disparity initialization step utilizes these features to retrieve feasible hypotheses; ③ an efficient propagation step refines the estimates using slanted support windows [49]. Motivated by these observations, we represent the disparity map as planar tiles at various resolutions and attach a learnable feature vector to each tile hypothesis (Sec. 3.1). This allows our network to learn which information about a small part of the disparity map is relevant to further improvement. This can be interpreted as an efficient, sparse version of the learnable 3D cost volumes that have shown to be beneficial [23].

The overall method is depicted in Fig. 2. Our feature extraction module relies on a very small U-Net [40], where the multi-resolution features of the decoder are used by the rest of the pipelines. These features encode multi-scale details of the image, similar to [7] (Sec. 3.2). Once the features are extracted, we initialize disparity maps as fronto parallel tiles at multiple resolutions. To do so, a matcher evaluates multiple hypotheses and selects the one with the lowest ℓ_1 distance between left and right view feature. Additionally, a compact per-tile descriptor is computed using a small network (Sec. 3.3). The output of the initialization is then passed to a propagation stage, which acts similarly to the approximated Conditional Random Field solution used in [11, 49]. This stage hierarchically refines the tile hypotheses in an iterative fashion (Sec. 3.4).

紧凑的特征表示与提取

高分辨率视差初始化并利用特征检索可行假设

有效的传播步骤refine视差

初始化视差为朝前的平行图

传播阶段类似CRF逐层refine纹理假设

3.1. Tile Hypothesis

We define a tile hypothesis as a planar patch with a learnable feature attached to it. Concretely, it consists of a geometric part describing a slanted plane with the disparity d and the gradient of disparity in x and y directions (d_x, d_y), and a learnable part $\mathbf{p}$ which we call tile feature descriptor. The hypothesis is therefore described as a vector

$$\mathbf{h} = [\underbrace{d, d_x, d_y}_{\text{plane}}, \underbrace{\mathbf{p}}_{\text{descriptor}}]. \tag{1}$$

The tile feature descriptor is a learned representation of the tile which allows the network to attach additional information to the tile. This could for example be matching quality or local surface properties such as how planar the geometry actually is. We do not constrain this information and learned it end-to-end from the data instead.

3.2. Feature Extractor

The feature extractor provides a set of multi-scale feature maps $\mathcal{E} = \{\mathbf{e}_0, \dots \mathbf{e}_M\}$ that are used for initial matching and for warping in the propagation stage. We denote a feature map $\mathbf{e}_l$ and an embedding vector $\mathbf{e}_{l,x,y}$ for locations x, y at resolution $l \in 0, \dots, M$, 0 being the original image resolution and M a $2^M \times 2^M$ downsampled resolution. A single embedding vector $\mathbf{e}_{l,x,y}$ is composed of multiple feature channels. We implement the feature extractor $\mathcal{E} = \mathcal{F}(\mathbf{I}; \boldsymbol{\theta}_\mathcal{F})$ as a U-Net like architecture [40, 32], i.e. an encoder-decoder with skip connections, with learnable parameters $\boldsymbol{\theta}_\mathcal{F}$. The network is composed of strided convolutions and transposed convolutions with leaky ReLUs as non-linearities. The set of feature maps $\mathcal{E}$ that we use in the remainder of the network are the outputs of the upsampling part of the U-Net at all resolutions. This means that even the high resolution features do contain some amount of spatial context. In more details, one down-sampling block of the U-Net has a single 3×3 convolution followed by a 2×2 convolution with stride 2. One up-sampling block applies 2×2 stride 2 transpose convolutions to up-sample results of coarser U-Net resolution. Features are concatenated with skip-connection, and a 1×1 convolution followed by a 3×3 convolution are applied to merge the skipped and upsampled feature for the current resolution. Each upsampling block generates a feature map $\mathbf{e}_l$, which is then used for downstream tasks and also further upsampled in the U-Net to generate a higher resolution feature map. We run the feature extractor on the left and the right image and obtain two multi-scale representations $\mathcal{E}^L$ and $\mathcal{E}^R$.

3.3. Initialization

The goal of the initialization stage is to extract an initial disparity d^{init} and a feature vector $\mathbf{p}^{\text{init}}$ for each tile at various resolutions. The output of the initialization

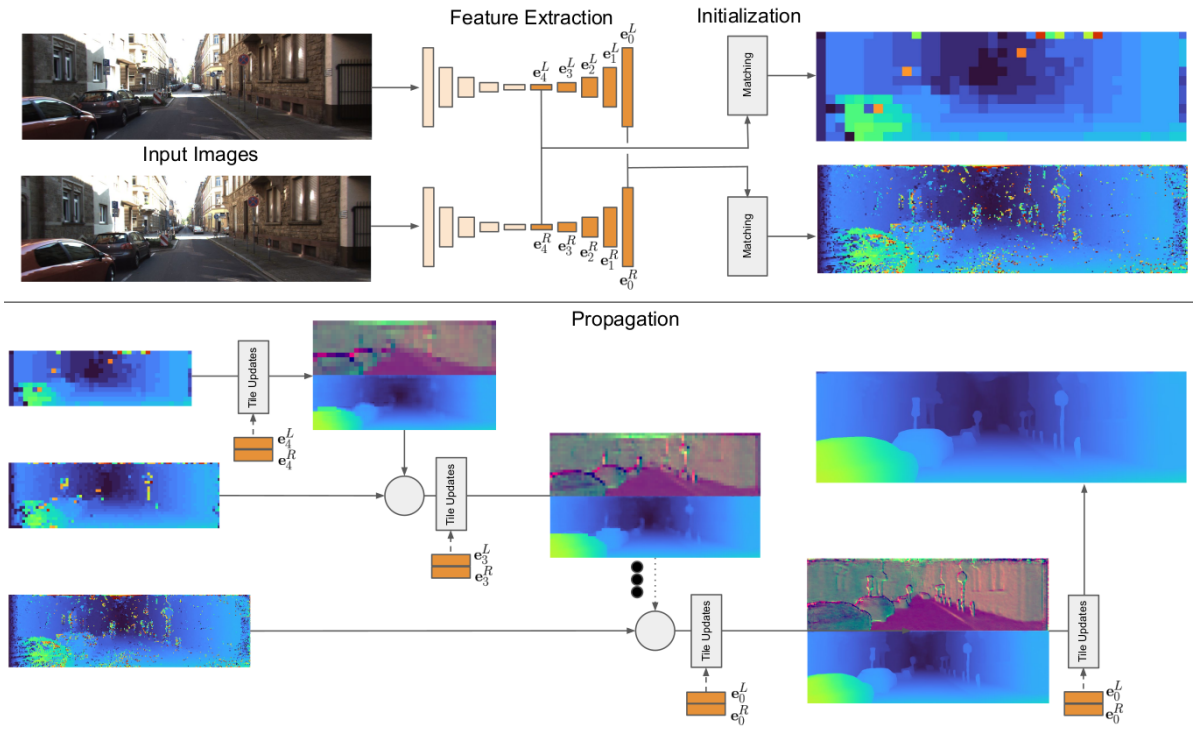

Figure 2: Overview of the proposed framework. (Top) A U-Net architecture is used to extract features at multiple scales from left and right images. An initialization step is run on each scale of the extracted features. This step operates on *tiles* of 4×4 feature regions and it evaluates multiple disparity hypotheses. The disparity with the minimum cost is selected. (Bottom) The output of the initialization is then used at propagation stage to refine the predicted disparity hypotheses using slanted support windows.

stage is fronto-parallel tile hypotheses of the form $\mathbf{h}^{\text{init}} = [d^{\text{init}}, 0, 0, \mathbf{p}^{\text{init}}]$.

Tile Disparity. In order to keep the initial disparity resolution high we use overlapping tiles along the x direction, i.e. the width, in the right (secondary) image but we still use non-overlapping tiles in the left (reference) image for efficient matching. To extract the tile features we run a 4×4 convolution on each extracted feature map $\mathbf{e}_l$. The strides for the left (reference) image and the right (secondary) image are different to facilitate the aforementioned overlapping tiles. For the left image we use strides of 4×4 and for the right image we use strides of 4×1, which is crucial to maintain the full disparity resolution to maximize accuracy. This convolution is followed by a leaky ReLU and a 1×1 convolution.

The output of this step will be a new set of feature maps $\tilde{\mathcal{E}} = \{\tilde{\mathbf{e}}_0, \ldots, \tilde{\mathbf{e}}_M\}$ with per tile features $\tilde{\mathbf{e}}_{l,x,y}$. Note that the width of the feature maps in $\tilde{\mathcal{E}}^L$ and $\tilde{\mathcal{E}}^R$ are now different. The per-tile features are explicitly matched along the scan lines. We define the matching cost ϱ at location (x, y) and resolution l with disparity d as:

$$\varrho(l, x, y, d) = \|\tilde{\mathbf{e}}^L_{l,x,y} - \tilde{\mathbf{e}}^R_{l,4x-d,y}\|_1. \tag{2}$$

The initial disparities are then computed as:

$$d^{\text{init}}_{l,x,y} = \text{argmin}_{d \in [0,D]}\, \varrho(l, x, y, d) \tag{3}$$

for each (x, y) location and resolution l, where D is the maximal disparity that is considered. Note that despite the fact that the initialization stage exhaustively computes matches for all disparities there is no need to ever store the whole cost volume. At test time only the location of the best match needs to be extracted, which can be done very efficiently utilizing fast memory, e.g. shared memory on GPUs and a fused implementation in a single Op. Hence, there is no need to store and process a 3D cost volume.

Tile Feature Descriptor. The initialization stage also predicts a feature description $\mathbf{p}^{\text{init}}_{l,x,y}$ for each (x, y) location and resolution l:

$$\mathbf{p}^{\text{init}}_{l,x,y} = \mathcal{D}(\varrho(d^{\text{init}}_{l,x,y}), \tilde{\mathbf{e}}^L_{l,x,y}; \boldsymbol{\theta}_{\mathcal{D}_l}). \tag{4}$$

The features are based on the embedding vector of the reference image $\tilde{\mathbf{e}}^L_{l,x,y}$ and the costs ϱ of the best matching disparity d_{init}. We utilize a perceptron $\mathcal{D}$, with learnable weights $\boldsymbol{\theta}_{\mathcal{D}}$, which is implemented with a 1×1 convolution

followed by a leaky ReLU. The input to the tile feature descriptor includes the matching costs $\varrho(\cdot)$, which allows the network to get a sense of the confidence of the match.

3.4. Propagation

The propagation step takes tile hypotheses as input and outputs refined tile hypotheses based on spatial propagation of information and fusion of information. It internally warps the features from the feature extraction stage from the right image (secondary) to the left image (reference) in order to predict highly accurate offsets to the input tiles. An additional confidence is predicted which allows for effective fusion between hypotheses coming from earlier propagation layers and from the initialization stage.

Warping. The warping step computes the matching costs between the feature maps $\mathbf{e}_l^L$ and $\mathbf{e}_l^R$ at the feature resolution l associated to the tiles. This step is used to build a local cost volume around the current hypothesis. Each tile hypothesis is converted into a planar patch of size 4×4 that it originally covered in the feature map. We denote the corresponding 4×4 local disparity map as $\mathbf{d}'$ with

$$\mathbf{d'}_{i,j} = d + (i - 1.5)d_x + (j - 1.5)d_y, \tag{5}$$

for patch coordinates $i, j \in \{0, \cdots, 3\}$. The local disparities are then used to warp the features $\mathbf{e}_l^R$ from the right (secondary) image to the left (reference) image using linear interpolation along the scan lines. This results in a warped feature representation $\mathbf{e}_l^{R'}$ which should be very similar to the corresponding features of the left (reference) image $\mathbf{e}^L$ if the local disparity maps $\mathbf{d}'$ are accurate. Comparing the features of the reference (x, y) tile with the warped secondary tile we define the cost vector $\boldsymbol{\phi}(\mathbf{e}, \mathbf{d}') \in \mathbb{R}^{16}$ as:

$$\boldsymbol{\phi}(\mathbf{e}_l, \mathbf{d}') = [c_{0,0}, c_{0,1}, \ldots, c_{0,3}, c_{1,0} \ldots c_{3,3}], \tag{6}$$

where $c_{i,j} = \|\mathbf{e}^L_{l,4x+i,4y+j} - \mathbf{e}^R_{l,4x+i-\mathbf{d'}_{i,j},4y+j}\|_1$.

Tile Update Prediction. This step takes n tile hypotheses as input and predicts deltas for the tile hypotheses plus a scalar value w for each tile indicating how likely this tile is to be correct, i.e. a confidence measure. This mechanism is implemented as a CNN module $\mathcal{U}$, the convolutional architecture allows the network to see the tile hypotheses in a spatial neighborhood and hence is able to spatially propagate information. A key part of this step is that we augment the tile hypothesis with the matching costs $\boldsymbol{\phi}$ from the warping step. By doing this for a small neighborhood in disparity space we build up a local cost volume which allows the network to refine the tile hypotheses effectively. Concretely, we displace all the disparities in a tile by a constant offset of one disparity $\mathbf{1}$ in the positive and negative directions and compute the cost three times. Using this let $\mathbf{a}$ be the augmented tile hypothesis map for input tile map $\mathbf{h}$:

$$\mathbf{a}_{l,x,y} = [\mathbf{h}_{l,x,y}, \underbrace{\boldsymbol{\phi}(\mathbf{e}_l, \mathbf{d}' - \mathbf{1}), \boldsymbol{\phi}(\mathbf{e}_l, \mathbf{d}'), \boldsymbol{\phi}(\mathbf{e}_l, \mathbf{d}' + \mathbf{1})}_{\text{local cost volume}}], \tag{7}$$

for a location (x, y) and resolution l, The CNN module $\mathcal{U}_l$ then predicts updates for each of the n tile hypothesis maps and additionally $w^i \in \mathbb{R}$ which represent the confidence of the tile hypotheses:

$$(\underbrace{\Delta\mathbf{h}_l^1, w^1, \ldots, \Delta\mathbf{h}_l^n, w^n}_{\text{hypotheses updates}}) = \mathcal{U}_l(\mathbf{a}_l^1, \ldots, \mathbf{a}_l^n; \boldsymbol{\theta}_{\mathcal{U}_l}). \tag{8}$$

The architecture of $\mathcal{U}$ is implemented with residual blocks [19] but without batch normalization. Following [24] we use dilated convolutions to increase the receptive field. Before running a sequence of residual blocks with varying dilation factors we run a 1×1 convolution followed by a leaky ReLU to decrease the number of feature channels. The update module is applied in a hierarchical iterative fashion (see Fig. 2). At the lowest resolution $l = M$ we only have 1 tile hypothesis per location from the initialization stage, hence $n = 1$. We apply the tile updates by summing the input tile hypotheses and the deltas and upsample the tiles by a factor of 2 in each direction. Thereby, the disparity d is upsampled using the plane equation of the tile and the remaining parts of the tile hypothesis d_x, d_y and $\mathbf{p}$ are upsampled using nearest neighbor sampling. At the next resolution $M-1$ we now have two hypotheses: the one from the initialization stage and the upsampled hypotheses from the lower resolution, hence $n = 2$. We utilize the w^i to select the updated tile hypothesis with highest confidence for each location. We iterate this procedure until we reach the resolution 0. To further refine the disparity map we decrease the tile size by a factor of 2×2 and assign full resolution features to the tiles. We run the propagation module using $n = 1$ until we reach tile size 1×1, which is our final prediction.

4. Loss Functions

Here we detail the loss functions used to train HITNet. Our losses rely on the ground truth disparities d^{gt}. To compute them at multiple resolutions we maxpool the ground truth disparity maps to downsample them to the required resolution.

4.1. Initialization Loss

Ground truth disparities are given as floating point disparities with subpixel precision, however matching in initialization happens with integer disparities. Therefore we compute the matching cost for subpixel disparities using linear interpolation. The cost for subpixel disparities is then given as

$$\psi(d) = (d - \lfloor d \rfloor)\varrho(\lfloor d \rfloor + 1) + (\lfloor d \rfloor + 1 - d)\varrho(\lfloor d \rfloor), \tag{9}$$

where we dropped the l, x, y subscripts for clarity. We aim at training the features $\mathcal{E}$ to be such that the matching cost ψ is smallest at the ground truth disparity and larger everywhere else. To achieve this, we impose an ℓ_1 contrastive loss [17]

$$L^{\text{init}}(d^{\text{gt}}, d^{\text{nm}}) = \psi(d^{\text{gt}}) + \max(\beta - \psi(d^{\text{nm}}), 0), \quad (10)$$

where $\beta > 0$ is a margin, d^{gt} the ground truth disparity for a specific location and

$$d^{\text{nm}} = \text{argmin}_{d \in [0,D] / \{d : d \in [d^{\text{gt}} - 1.5, d^{\text{gt}} + 1.5]\}} \varrho(d) \quad (11)$$

the disparity of the lowest cost non match for the same location. This cost pushes the ground truth cost toward 0 as well as the lowest cost non match toward a certain margin. In all our experiments we set the margin to $\beta = 1$. Similar contrastive losses have been used to learn the matching score in earlier deep learning based approaches to stereo matching [55, 34]. However, they either used a random non-matching location as negative sample or used all the non matching locations as negative samples, respectively.

4.2. Propagation Loss

During propagation we impose a loss on the tile geometry d, d_x, d_y and the tile confidence w. We use the ground truth disparity d^{gt} and ground truth disparity gradients d_x^{gt} and d_y^{gt}, which we compute by robustly fitting a plane to d^{gt} in a 9×9 window centered at the pixel. In order to apply the loss on the tile geometry we first expand the tiles to a full resolution disparities $\hat{d}$ using the plane equation (d, d_x, d_y) analogously to Eq. 5. We use the general robust loss function $\rho(\cdot)$ from [2] which resembles a smooth ℓ_1 loss, i.e., Huber loss. Additionally, we apply a truncation to the loss with threshold A

$$L^{\text{prop}}(d, d_x, d_y) = \min(\rho(d^{\text{diff}}), A), \quad (12)$$

where $d^{\text{diff}} = d^{\text{gt}} - \hat{d}$. Further we impose a loss on the surface slant, as

$$L^{\text{slant}}(d_x, d_y) = \left\| \begin{array}{c} d_x^{\text{gt}} - d_x \\ d_y^{\text{gt}} - d_y \end{array} \right\|_1 \chi_{|d^{\text{diff}}| < B}, \quad (13)$$

where χ is an indicator function which evaluates to 1 when the condition is satisfied and 0 otherwise. To supervise the confidence w we impose a loss which increases the confidence if the predicted hypothesis is closer than a threshold C_1 from the ground truth and decrease the confidence if the predicted hypothesis is further than a threshold C_2 away from the ground truth.

$$L^{\text{w}}(w) = \max(1 - w, 0)\chi_{|d^{\text{diff}}| < C_1} + \max(w, 0)\chi_{|d^{\text{diff}}| > C_2} \quad (14)$$

4.3. Global Loss

Our network is trained end-to-end utilizing all these losses as a weighted sum over all the scales and pixels: $\sum_{l,x,y} \lambda^{\text{init}} L_l^{\text{init}} + \lambda^{\text{prop}} L_l^{\text{prop}} + \lambda^{\text{slant}} L_l^{\text{slant}} + \lambda^{\text{w}} L_l^{\text{w}}$, with hyperparameters $\lambda^{\cdot}$=1 in our experiments.

5. Experiments

We evaluate the proposed approach on popular benchmarks showing competitive results at a fraction of the computational time compared to other methods. We consider the following datasets: SceneFlow [36], KITTI 2012 [14], KITTI 2015 [37], ETH3D [43]. Following the standard evaluation settings we consider the two popular metrics: the End-Point-Error (EPE), which is the absolute distance in disparity space between the predicted output and the groundtruth; the x-pixels error, which is the percentage of pixels with disparity error greater than x. For the EPE computation on SceneFlow we adopt the same methodology of PSMNet [7], which excludes all the pixel with ground truth disparity bigger than 192 from the evaluation. Unless stated otherwise we use a HITNet with 5 levels, i.e. $M = 4$.

5.1. Training Setup

The SceneFlow dataset 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 considered random crops of 320×960 and a batch size of 1, and a maximum disparity of 320. We trained for 8.1M iterations using the Adam optimizer, starting from a learning rate of $1e^{-4}$, dropping it to $1e^{-5}$ after 5M iterations, and again dropping it to $1e^{-6}$ after 8M iterations.

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 maximal 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 [50], and a small training set is sufficient for our method to achieve good performance. See appendix for a detailed analysis of the training evolution.

The training set for the real world ETH3D stereo dataset [43] contains just a few stereo pairs, so additional data is needed to avoid overfitting. For the benchmark submission

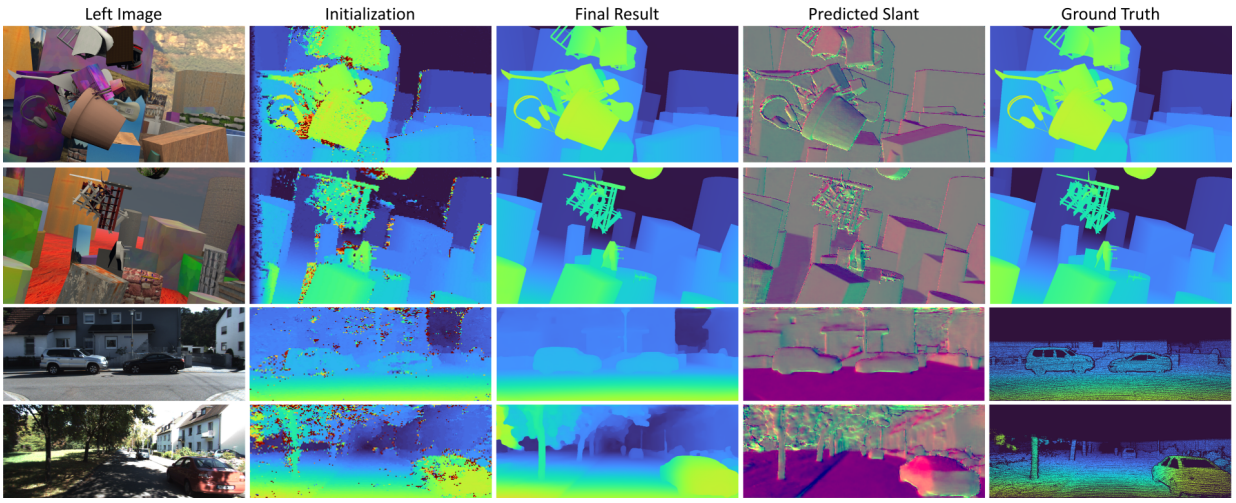

Figure 3: Qualitative results on SceneFlow and KITTI 2012. Note how the model is able to recover fine details, textureless regions and crisp edges.

	KITTI 2012 [14]						KITTI 2015 [37]			
Method	2-noc	2-all	3-noc	3-all	EPE noc	EPE all	D1-bg	D1-fg	D1-all	Runtime
HITNet	2.00	2.65	1.41	1.89	0.4	0.5	1.74	3.20	1.98	0.015s
GANet-deep [56]	1.89	2.50	1.19	1.6	0.4	0.5	1.48	3.46	1.81	1.8s
EdgeStereo-V2 [47]	2.32	2.88	1.46	1.83	0.4	0.5	1.84	3.30	2.08	0.32s
GC-Net [23]	2.71	3.46	1.77	2.30	0.6	0.7	2.21	6.16	2.87	0.9s
SGM-Net [44]	3.60	5.15	2.29	3.50	0.7	0.9	2.66	8.64	3.66	67s
ESMNet [16]	3.65	4.30	2.08	2.53	0.6	0.7	2.57	4.86	2.95	0.06s
MC-CNN-acrt [55]	3.90	5.45	2.09	3.22	0.6	0.7	2.89	8.88	3.89	67s
RTSNet [28]	3.98	4.61	2.43	2.90	0.7	0.7	2.86	6.19	3.41	0.02s
Fast DS-CS [51]	4.54	5.34	2.61	3.20	0.7	0.8	2.83	4.31	3.08	0.02s
StereoNet [24]	4.91	6.02	-	-	0.8	0.9	4.30	7.45	4.83	0.015s

Table 1: Quantitative evaluation on KITTI 2012 and KITTI 2015. For KITTI 2012 we report the percentage of pixels with error bigger than x disparities in both non-occluded (x-noc) and all regions (x-all), as well as the overall EPE in both non occluded (EPE-noc) and all the pixels (EPE-all). For KITTI 2015 We report the percentage of pixels with error bigger than 1 disparity in background regions (bg), foreground areas (fg), and all.

Method	HITNet Large	PSMNet [7]	GA-Net [56]	StereoNet [24]	EdgeStereo [47]
EPE	0.465 px	1.09 px	0.84 px	1.1 px	0.74 px
Run-time	0.04s	0.41s	1.6s	0.015s	0.32s

Table 2: Comparisons with state-of-the-art methods on Scene Flow "finalpass" dataset. The end-point-error (EPE) is reported, lower is better.

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 [41] 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.

5.1.1 Run-time Computational Breakdown

The proposed model architecture runs at 15ms 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 (9 ms) that operate on higher resolutions. The multi-scale propagation steps use downsampled data and contribute less than 3ms. Efficient implementation of initialization using a single fused Op generates initial disparity estimates across all resolutions in 0.25ms, with feature extractor contributing 2.5ms when

run with $16, 16, 24, 24, 32$ channels at corresponding resolutions. Tile descriptor has 13 channels by default, residual blocks use 32 channels, except for the last propagation, that uses 16. Each propagation step uses 2 residual blocks, except for the 4×4 and 2×2 downsampled steps that use 4.

5.1.2 Data Augmentation

The training data available may not be fully representative of the actual test sets for small real world datasets such as KITTI. 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 asymmetric adjustments. We then replace random areas of the right image with random crops taken from another portion of the same image: this helps the network to deal with occluded areas and encourages a better "inpainting".

5.2. Comparisons with State-of-the-art

SceneFlow. On the synthetic dataset SceneFlow "finalpass" we achieve the remarkable End-Point-Error (EPE) of 0.465, which is the lowest reported at time of writing. The EPE error on "cleanpass" version is 0.395. Representative competitors are reported in Tab. 2. The PSMNet algorithm [7] performs multi-scale feature extraction similarly to our method, but in contrast they use a more sophisticated pooling layer. Here we show that our architecture is more effective. Compared to GA-Net [56], we do not need complex message passing steps such as SGM. The results we obtain show that our strategy is also achieving a very similar inference. Finally, a representative fast method, StereoNet [24] is considered, which we consistently outperform. As result our method achieves the lowest EPE while still maintaining real-time performance. In Figure 3 we report qualitative results.

KITTI 2012 and 2015. At time of writing, among the published methods faster than 100ms, HITNet ranks #1 on KITTI 2012 and 2015 benchmarks. Compared to other state-of-the-art stereo matchers (see Tab. 1), our approach compares favorably to GC-Net [23], [38] and many others. Recent methods such as GA-Net [56] and HSM [50] are obtaining slightly better metrics on both the real-world scenarios, although they require 1.8 and 0.15 seconds respectively. Note also that HSM [50] has been trained with additional external high resolution data. Similarly, GA-Net [56] is pre-trained on SceneFlow and fine-tuned on KITTI benchmarks, whereas our approach is fully trained on the small data available on KITTI. Compared to fast methods such as StereoNet [24] and RTSNet [28], our method consistently outperforms them by a considerable margin, showing that it can be employed in latency critical scenarios without sacrificing accuracy.[2]

ETH3D two view stereo. We evaluated our method with multiple state-of-art approaches on the ETH3D dataset, see Tab. 3. At time of writing, HITNet ranks 1^{st}-3^{rd} on all the metrics published on the website. In particular, our method ranks 1^{st} on the following metrics: bad 0.5, bad 4, average error, rms error, 50% quantile, 90% quantile: this shows that HITNet is resilient to the particular measurement chosen, whereas competitive approaches exhibits substantial differences when different metrics are selected, see the submission website for details[3].

Generalization. We finally demonstrate the cross-domain adaptation capabilities of our method. Following the protocol in [47], we trained HITNet on SceneFlow and tested on KITTI 2012 and KITTI 2015 respectively. We also considered multiple competitors as in [47] and report the results in Tab. 4: 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.

5.3. Ablation Study

We analyze the importance of the components of our proposed algorithm. We consider the full HITNet as baseline and compare it with the same model where we removed one of the main features. The ablation study is performed on the SceneFlow "finalpass" data and KITTI 2012.

Low Res Init. Initialization at full disparity resolution provides a compelling starting point to the network, which can focus mostly on refining the prediction. In Table 5 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.

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. 5, removing the slant prediction leads to a substantial drop in precision for both SceneFlow and KITTI

[2] See the KITTI Website at http://www.cvlibs.net/datasets/kitti/eval_stereo.php for the complete metrics.
[3] See the ETH3D Website at https://www.eth3d.net/low_res_two_view for the complete metrics.

Method	EPE px	Bad 1.0	Bad 2.0	Run-time (ms)
HITNet(ours)	0.20	2.79	0.80	15
DN-CSS	0.22	2.69	0.77	310
AdaStereo[46]	0.26	3.41	0.74	400
Deep-Pruner[8]	0.26	3.52	0.86	160
iResNet[31]	0.24	3.68	1.00	200
Stereo-DRNet-Refined[6]	0.27	4.46	0.83	330
PSMNet[7]	0.33	5.02	1.09	540

Table 3: Comparisons with state-of-the-art methods on ETH3D stereo dataset. For all metrics lower is better.

Dataset	HITNet	CRL [38]	iResNet [30]	PSMNet [7]	EdgeStereo [47]
KITTI 2012 EPE	1.00	1.38	1.27	5.54	1.96
KITTI 2012 $> 3px$	5.95	9.07	7.89	27.33	12.27
KITTI 2015 EPE	1.28	1.35	1.21	6.44	2.06
KITTI 2015 $> 3px$	6.29	8.88	7.42	29.86	12.46

Table 4: 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 [36]				KITTI 2012 [14]		
Model	EPE	0.1 px	1 px	3 px	EPE	2 px	3 px
4 Scales	-	-	-	-	0.507 px	3.10 %	2.20 %
No Multi-scale	-	-	-	-	0.747 px	4.76 %	3.62 %
Low Res Init	0.631 px	29.7 %	6.38 %	3.39 %	0.561 px	3.72 %	2.54 %
No Slant Prediction	0.625 px	27.0 %	6.31 %	3.36 %	0.513 px	3.23 %	2.18 %
No Tile Features	0.604 px	26.6 %	6.06 %	3.22 %	0.488 px	3.04 %	2.06 %
No Warping	0.585 px	32.5 %	6.07 %	3.13 %	0.602 px	3.72 %	2.54 %
HITNet (ours)	0.559 px	25.4 %	5.83 %	3.10 %	0.484 px	2.91 %	2.00 %
HITNet Large (ours)	0.465 px	22.0 %	4.99 %	2.71 %	0.490 px	2.98 %	2.13 %

Table 5: Ablation study of the proposed HITNet on SceneFlow [36] and KITTI 2012 [14] datasets. Lower is better.

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 and removing this step hurts the subpixel precision performance as demonstrated in Tab. 5.

Multi Scale Prediction. The multi-scale feature affects both initialization and propagation stages. In Tab. 5, 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.

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 used 32 channels and 6 residual blocks for the last 3 propagation steps, this resorts to a run-time increase to 40ms. As expected this has an improvement on SceneFlow as reported in Tab. 5, HITNet Large; however for the small KITTI datasets this did not improve performance due to overfitting.

6. Discussion

We presented HITNet, a real-time end-to-end architecture for accurate stereo matching. We presented a fast initialization step that is able to compute high resolution matches using learned features very efficiently. These tile initializations are then fused using propagation and fusion steps. The use of slanted support windows with learned descriptors provides additional accuracy. We thoroughly evaluated all our components with an ablation study. Further, we presented state-of-the art accuracy amongst real-time al-

gorithms on two commonly used bench marks. Finally, we demonstrated how our trained networks are able to generalize from synthetic to real data. A limitation of our algorithm is that it needs to be trained on a dataset with ground truth depth. To address this in the future we are planning to investigate self-supervised methods and self-distillation methods to further increase the accuracy and decrease the amount of training data that is required.

References

[1] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (ToG)*, 2009. 2

[2] Jonathan T Barron. A general and adaptive robust loss function. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6

[3] Frederic Besse, Carsten Rother, Andrew Fitzgibbon, and Jan Kautz. Pmbp: Patchmatch belief propagation for correspondence field estimation. *International Journal of Computer Vision (IJCV)*, 2014. 2

[4] Michael Bleyer and Margrit Gelautz. Simple but effective tree structures for dynamic programming-based stereo matching. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2008. 2

[5] Michael Bleyer, Christoph Rhemann, and Carsten Rother. Patchmatch stereo-stereo matching with slanted support windows. In *British Machine Vision Conference (BMVC)*, 2011. 2

[6] 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. 9, 14

[7] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 3, 6, 7, 8, 9, 14

[8] Shivam Duggal, Shenlong Wang, Wei-Chiu Ma, Rui Hu, and Raquel Urtasun. Deeppruner: Learning efficient stereo matching via differentiable patchmatch. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4383–4392, 2019. 9

[9] S. R. Fanello, C. Keskin, S. Izadi, P. Kohli, D. Kim, D. Sweeney, A. Criminisi, J. Shotton, S.B. Kang, and T. Paek. Learning to be a depth camera for close-range human capture and interaction. *Transaction On Graphics (TOG)*, 2014. 2

[10] S. R. Fanello, C. Rhemann, V. Tankovich, A. Kowdle, S. Orts Escolano, D. Kim, and S. Izadi. Hyperdepth: Learning depth from structured light without matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[11] Sean Ryan Fanello, Julien Valentin, Adarsh Kowdle, Christoph Rhemann, Vladimir Tankovich, Carlo Ciliberto, Philip Davidson, and Shahram Izadi. Low compute and fully parallel computer vision with hashmatch. *Inernational Conference on Computer Vision (ICCV)*, 2017. 2, 3

[12] Sean Ryan Fanello, Julien Valentin, Christoph Rhemann, Adarsh Kowdle, Vladimir Tankovich, Philip Davidson, and Shahram Izadi. Ultrastereo: Efficient learning-based matching for active stereo systems. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 3

[13] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient belief propagation for early vision. *International journal of computer vision (IJCV)*, 2006. 2

[14] 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. 6, 7, 9

[15] Spyros Gidaris and Nikos Komodakis. Detect, replace, refine: Deep structured prediction for pixel wise labeling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[16] Chenggang Guo, Dongyi Chen, and Zhiqi Huang. Learning efficient stereo matching network with depth discontinuity aware super-resolution. *IEEE Access*, 2019. 7

[17] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006. 6

[18] Rostam Affendi Hamzah and Haidi Ibrahim. Literature survey on stereo vision disparity map algorithms. *Journal of Sensors*, 2016. 2

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition (CVPR)*, 2016. 5

[20] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence (TPAMI)*, 2008. 2

[21] Asmaa Hosni, Christoph Rhemann, Michael Bleyer, Carsten Rother, and Margrit Gelautz. Fast cost-volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013. 2

[22] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[23] 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, 3, 7, 8, 13, 14

[24] Sameh Khamis, Sean Fanello, Christoph Rhemann, Adarsh Kowdle, Julien Valentin, and Shahram Izadi. StereoNet: Guided hierarchical refinement for edge-aware depth prediction. In *European Conference on Computer Vision (ECCV)*, 2018. 2, 5, 7, 8

[25] Andreas Klaus, Mario Sormann, and Konrad Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *International Conference on Pattern Recognition (ICPR)*, 2006. 2

[26] Vladimir Kolmogorov and Ramin Zabih. Computing visual correspondence with occlusions using graph cuts. In *International Conference on Computer Vision (ICCV)*, 2001. 2

[27] Adarsh Kowdle, Christoph Rhemann, Sean Fanello, Andrea Tagliasacchi, Jon Taylor, Philip Davidson, Mingsong Dou, Kaiwen Guo, Cem Keskin, Sameh Khamis, David Kim, Danhang Tang, Vladimir Tankovich, Julien Valentin, and Shahram Izadi. The need 4 speed in real-time dense visual tracking. *Transaction On Graphics (TOG)*, 2018. 2

[28] H. Lee and Y. Shin. Real-time stereo matching network with high accuracy. In *IEEE International Conference on Image Processing (ICIP)*, 2019. 7, 8, 13

[29] Yu Li, Dongbo Min, Michael S Brown, Minh N Do, and Jiangbo Lu. SPM-BP: Sped-up patchmatch belief propagation for continuous mrfs. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

[30] 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. 9

[31] Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Linbo Qiao, Wei Chen, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. 2017. 2, 9

[32] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE conference on computer vision and pattern recognition (CVPR)*, 2015. 3

[33] Jiangbo Lu, Hongsheng Yang, Dongbo Min, and Minh N Do. Patch match filter: Efficient edge-aware filtering meets randomized search for fast correspondence field estimation. In *IEEE conference on computer vision and pattern recognition (CVPR)*, 2013. 2

[34] Wenjie Luo, Alexander G Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 6

[35] David Marr and Tomaso Poggio. Cooperative computation of stereo disparity. *Science*, 1976. 2

[36] 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. 1, 2, 6, 9

[37] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015. 6, 7

[38] 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. 2, 8, 9

[39] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention (MICCAI)*, 2015. 3

[41] 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. 7

[42] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 2002. 2, 3

[43] 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. 6

[44] A. Seki and M. Pollefeys. SGM-Nets: Semi-global matching with neural networks. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 7

[45] Amit Shaked and Lior Wolf. Improved stereo matching with constant highway networks and reflective confidence learning. In *IEEE conference on computer vision and pattern recognition (CVPR)*, 2017. 2

[46] 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. 9

[47] 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. 7, 8, 9

[48] 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. 3

[49] Vladimir Tankovich, Michael Schoenberg, Sean Ryan Fanello, Adarsh Kowdle, Christoph Rhemann, Max Dzitsiuk, Mirko Schmidt, Julien Valentin, and Shahram Izadi. Sos: Stereo matching in o(1) with slanted support windows. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018. 2, 3

[50] 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. 2, 6, 8

[51] Kyle Yee and Ayan Chakrabarti. Fast deep stereo with 2d convolutional processing of cost signatures. In *Winter Conference on Applications of Computer Vision (WACV)*, 2020. 7

[52] Kuk-Jin Yoon and In-So Kweon. Locally adaptive support-weight approach for visual correspondence search. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 2

[53] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. 2

[54] Jure Zbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. In *IEEE conference on computer vision and pattern recognition (CVPR)*, 2015. 2

[55] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research (JMLR)*, 2016. 2, 6, 7

[56] 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, 8, 13, 14

[57] Yinda Zhang, Sameh Khamis, Christoph Rhemann, Julien Valentin, Adarsh Kowdle, Vladimir Tankovich, Michael Schoenberg, Shahram Izadi, Thomas Funkhouser, and Sean Fanello. ActiveStereoNet: End-to-end self-supervised learning for active stereo systems. *European Conference on Computer Vision (ECCV)*, 2018. 2

A. Training Scheme

In this section we provide additional details regarding our training scheme. 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 5 we show the evolution of the training for more than 200 epochs (learning rate change to $1e^{-5}$ after 200 epochs). 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 in the main paper showing that the method has very good cross-dataset performance.

B. Additional Evaluations

In this section we show additional qualitative results on real-world datasets. In Figure 6 we show comparisons of our method with other approaches. We consider multiple representative competitors such as: GC-Net [23], which uses the full cost volume and 3D convolutions to infer context, RTS-Net [28] that has similar inference time than HITNet, and finally GA-Net [56], 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. GANet [56]. Note how our method retrieves fine structures and crisp edges, while only training on the KITTI datasets, which exhibit significant edge fattening artifacts.

C. Intermediate Outputs

We show intermediate outputs from within our network in Fig 7. 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.

D. 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. 6 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. This has been showed in the main paper where our approach is able to outperform multiple state-of-the-art baselines when trained on synthetic data and tested in real-world scenarios.

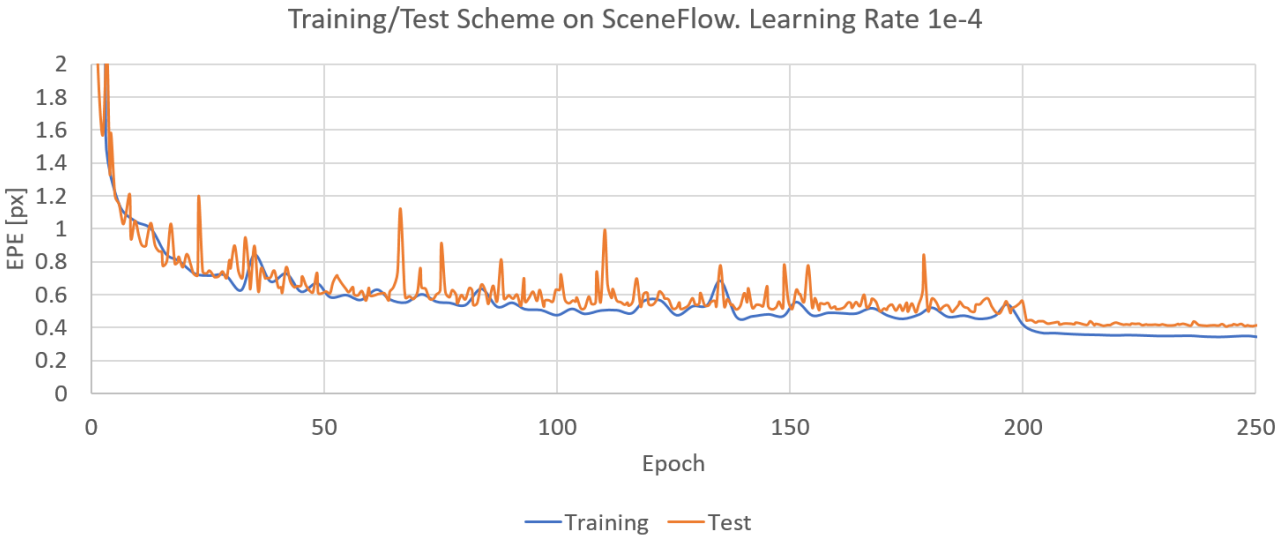

Figure 5: We show the evolution of the training reporting the EPE on training and test set respectively. Note how the scheme reduces the error on both training and test set without showing signs of overfitting.

Model	Param	GMac	EPE	1 Pixel Threshold Error Rates
GC-Net [23]	2.9M [56]	8789 [6]	1.80 [56]	15.6 [56]
PSMNet [7]	3.5M [56]	2594 [6]	1.09 [56]	12.1 [56]
GANet [56]	2.3M [56]	-	0.84 [56]	9.9 [56]
StereoDRNet [6]	-	1410 [6]	0.98 [6]	-
HITNet	0.45M	48	0.57	5.85
HITNet Large	0.96M	136	0.465	4.56

Table 6: 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.

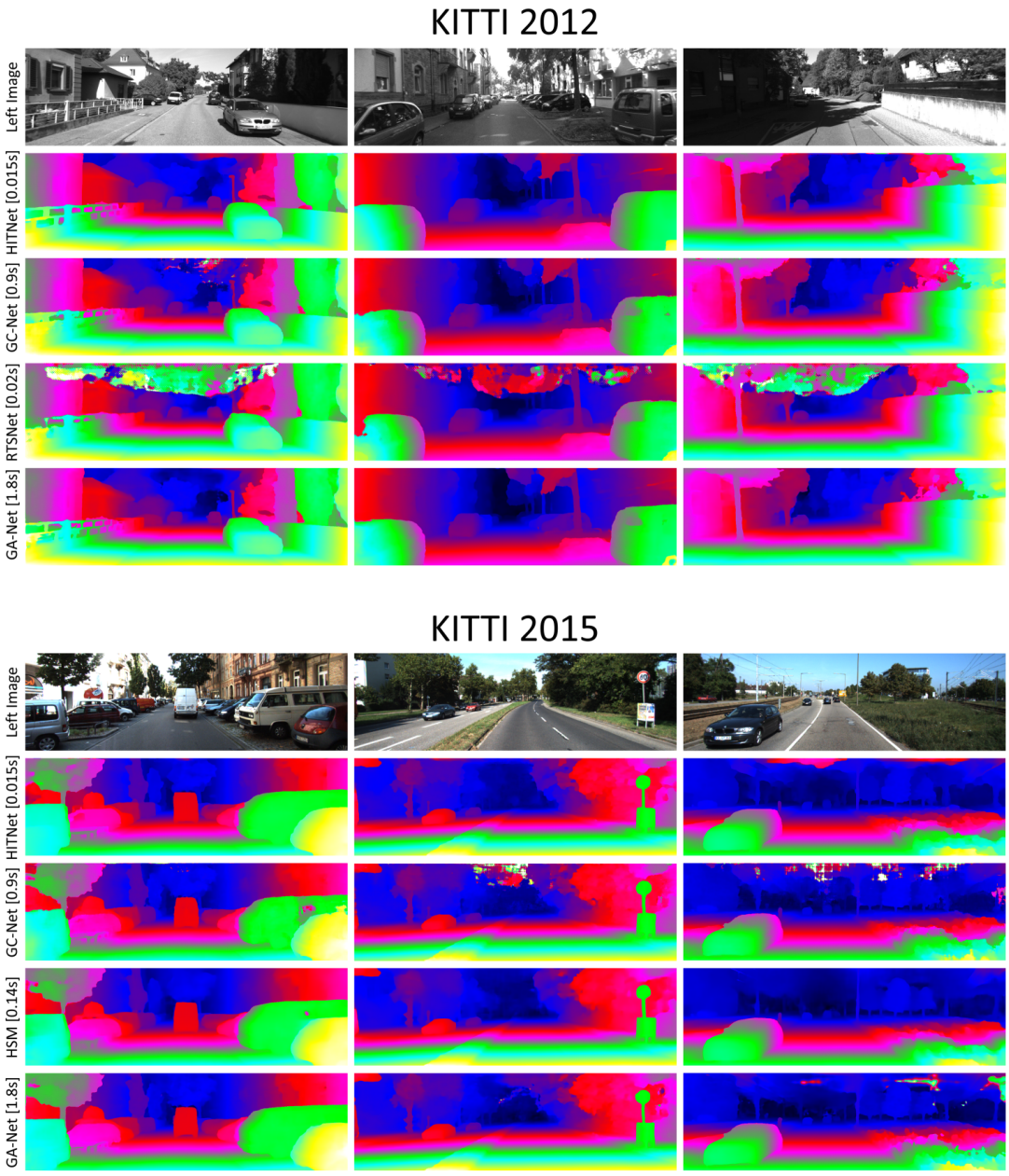

Figure 6: Qualitative Results on KITTI 2012 and 2015. Note how HITNet is able to recover fine structures and crisp edges using a fraction of the computational cost required by other competitors.

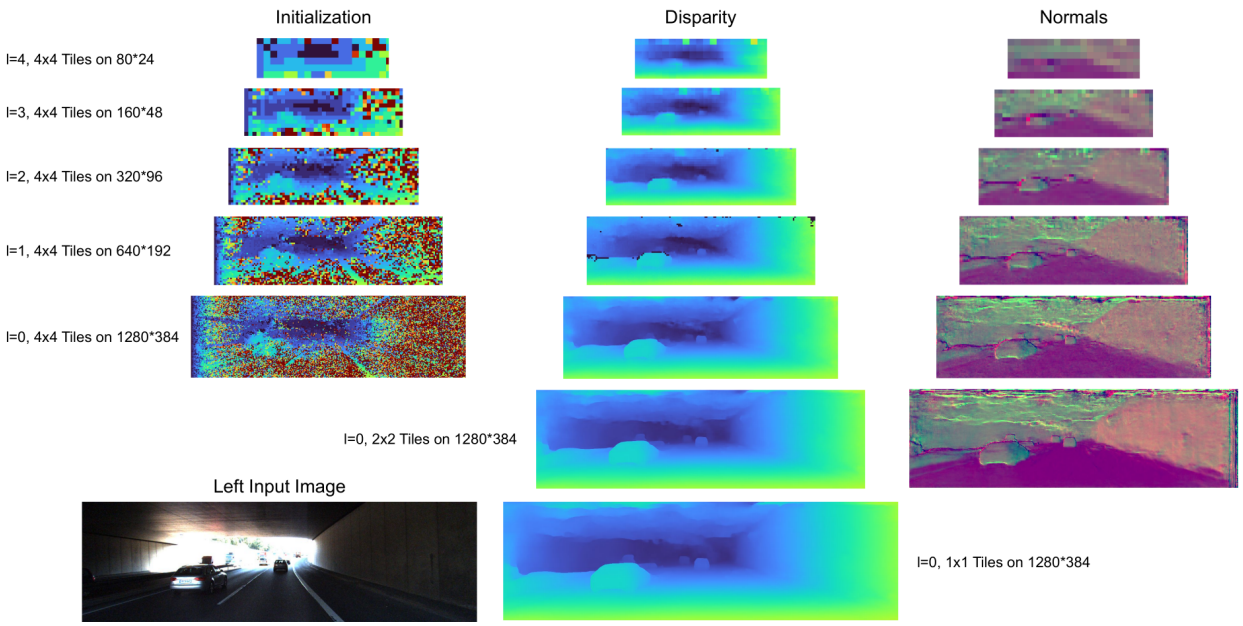

Figure 7: Intermediate results of our network on the left side we show the disparity maps that the matching of the initialization stage provides. On the right hand side we show the final disparity and normals for each resolution. The final two resolutions are 2x2 and 1x1 tiles of the highest resolution feature map, while the initialization is always computed on 4x4 tiles of the feature maps.