Practical Stereo Matching via Cascaded Recurrent Network with Adaptive Correlation

Jiankun Li[1] Peisen Wang[1*] Pengfei Xiong[2*] Tao Cai[1] Ziwei Yan[1] Lei Yang[1]
Jiangyu Liu[1] Haoqiang Fan[1] Shuaicheng Liu[3,1†]
[1]Megvii Research [2]Tencent
[3]University of Electronic Science and Technology of China
https://github.com/megvii-research/CREStereo

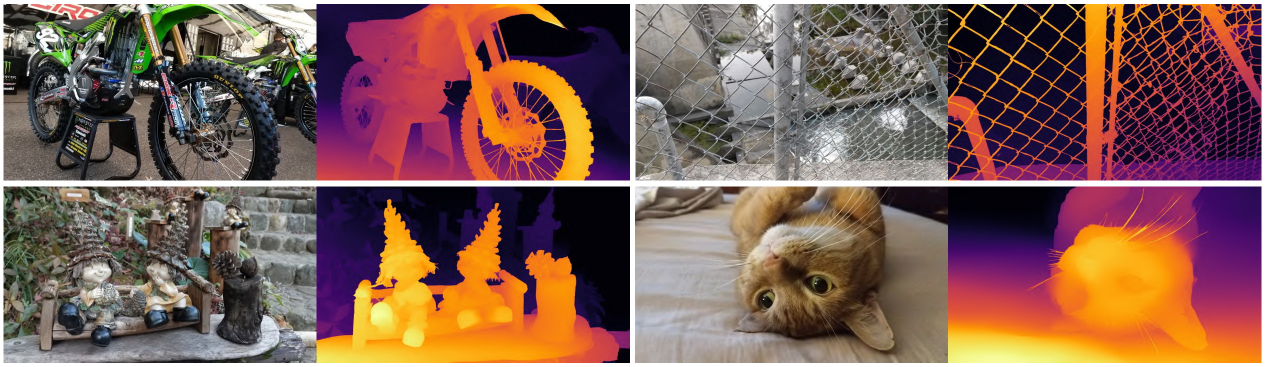


Figure 1. Examples of our predictions on images from Holopix50K [16] dataset. We show left images of the stereo pairs and their corresponding predicted disparities. Our results achieve high accuracy and exhibit high-quality details for fine-structured objects.

Abstract

With the advent of convolutional neural networks, stereo matching algorithms have recently gained tremendous progress. However, it remains a great challenge to accurately extract disparities from real-world image pairs taken by consumer-level devices like smartphones, due to practical complicating factors such as thin structures, non-ideal rectification, camera module inconsistencies and various hard-case scenes. In this paper, we propose a set of innovative designs to tackle the problem of practical stereo matching: 1) to better recover fine depth details, we design a hierarchical network with recurrent refinement to update disparities in a coarse-to-fine manner, as well as a stacked cascaded architecture for inference; 2) we propose an adaptive group correlation layer to mitigate the impact of erroneous rectification; 3) we introduce a new synthetic dataset with special attention to difficult cases for better generalizing to real-world scenes. Our results not only rank 1st on both Middlebury and ETH3D benchmarks, outperforming existing state-of-the-art methods by a notable margin, but also exhibit high-quality details for real-life photos, which clearly demonstrates the efficacy of our contributions.

1. Introduction

Stereo matching is a classical research topic of computer vision, the goal of which, given a pair of rectified images, is to compute the displacement between two corresponding pixels, namely "disparity" [34]. It plays an important role in many applications, including autonomous driving, augmented reality, simulated bokeh rendering and so forth.

Recently, with the support of large synthetic datasets [5, 27, 46], convolutional neural network (CNN) based stereo matching methods have taken the accuracy of disparity estimation to a new height [8, 23, 44]. However, to make the algorithm truly practical in the scenario of everyday consumer photography, we are still faced with three major obstacles.

Firstly, it remains a complicated issue for most existing algorithms to precisely recover the disparity of fine image details, or thin structures such as nets and wire frames. The fact that consumer photos are being produced in higher resolutions only serves to worsen the problem. In computational bokeh, for instance, disparity error around fine details would result in degraded rendering results that are unpleasing to human perception [32]. Secondly, perfect rectification [24, 56] is hard to obtain for real-world stereo image pairs, as they are often produced by camera modules with

[*]Equal contribution. † Corresponding author.

different traits. For example, most current smartphones capture the stereo pair with a wide-angle and a telephoto lens, which have distinct characteristics like focal length and distortion parameters, inevitably resulting in non-ideal rectifications. Therefore existing methods assuming that the stereo pair is perfectly rectified are likely to fail under such adversarial conditions. In addition, the image pair produced by inconsistent cameras modules may vary in illumination, white balancing, image quality, etc., making the estimation task even harder. Finally, though it has been shown that models trained from large enough synthetic datasets can generalize well to real-world scenes [10, 27], disparity estimation in typical hard cases, like non-texture or repetitive-texture regions, continues to be difficult, which requires special attention be paid in covering relevant scenes in the training dataset.

In this paper, we propose CREStereo, namely Cascaded REcurrent Stereo matching network, which comprises a set of novel designs, to tackle the problem of practical stereo matching. To better recover intricate image details, we design a hierarchical network to update disparities recurrently in a coarse-to-fine manner; in addition, we adopt a stacked cascaded architecture for high-resolution inference. To alleviate the negative influence of rectification error, we design an adaptive group local correlation layer for feature matching. Furthermore, we introduce a new synthetic dataset with richer variations in lighting, texture and shapes, in order to better generalize to real-world scenes.

So far, CREStereo ranks 1st on both ETH3D two-view stereo [36] and Middlebury [35] benchmarks, and achieves competitive performance on KITTI 2012/2015 [11] among published methods. Additionally, our network demonstrates superior performance for arbitrary real-world scenes, well proving the effectiveness of our designs.

Our main contributions can thus be summarized as follows: 1) we propose a cascaded recurrent network for practical stereo matching and a stacked cascaded architecture for high-resolution inference; 2) we design an adaptive group correlation layer to handle non-ideal rectification; 3) we create a new synthetic dataset to better generalize to real-world scenes; 4) our method outperforms existing methods on public benchmarks such as Middlebury and ETH3D by a significant margin, and considerably increases the accuracy of recovered disparities for real-world stereo images.

2. Related Work

Traditional algorithms. Stereo matching is a challenging problem and has been studied for a long time. Traditional algorithms can be categorized into local and global methods. Local methods [2, 15, 47] compute matching cost using a support window centered at pixels along the epipolar line. Global methods treat stereo matching as an optimization problem, where an explicit cost function is formulated and optimized by *belief propagation* [20,42,52] or *graph cut* [4] algorithms. A semi-global matching (SGM) method is later proposed [14] using mutual information instead of intensity based on dynamic programming.

Learning-based algorithms. Deep neural network was first introduced in stereo matching task only for matching cost computation. Zbontar and LeCun [54] proposed to train a CNN to initialize the matching cost between patches, which is refined by cross-based aggregation and semi-global optimization as in SGM [14]. In recent years, end-to-end network has become mainstream in stereo matching. One line of networks [12, 21, 22, 27, 29, 44, 49] only uses 2D convolutions. Mayer *et al*. [27] introduced the first end-to-end network named DispNet and its correlation version DispNetC for disparity estimation. Pang *et al*. [29] proposed a two-stage framework called CRL with multi-scale residual learning. Guo *et al*. [12] proposed GwcNet with group-wise correlation to improve similarity measurement. AANet [49] introduced a novel aggregation method using sparse points and multi-scale interaction. A very recent method, RAFT-Stereo [23], takes advantage of the iterative refinement in the optical flow network RAFT [45] to design a network adapted for stereo matching. Another line of networks [7, 17, 18, 51, 55] uses 3D convolutions to perform cost volume construction and aggregation in traditional methods. GCNet [17] and PSMNet [7] proposed to construct a 4D cost volume with 3D hourglass aggregation networks. For high-res images, Yang *et al*. [51] proposed a coarse-to-fine hierarchical network to address memory and speed issues. Lately, neural architecture search has also been introduced into deep stereo networks [8].

Practical stereo matching. Stereo matching oriented toward real-world images is a less explored problem. Pang *et al*. [30] proposed a self-adaptation approach for generalizing CNN to target domain without ground truth disparity. Luo *et al*. [25] proposed a wavelet synthesis network to produce better results for bokeh applications on smartphones. Song *et al*. [39] introduced a domain adaptation pipeline for networks to narrow down the gap between synthetic and real-world domains.

Synthetic datasets. Sufficient training data is essential for deep stereo models, but it is hard to obtain accurate disparity in real world. Synthetic datasets [5,26,27,46] provide high-accuracy and dense ground truth. Recently, He *et al*. [13] built a data generation pipeline for stereo matching using Blender [3], with textures from real images of common datasets. Autoflow [40] introduced a simple method to render randomized polygons with motion for optical flow training. Despite the effectiveness of these datasets, they still have limited variations of object shapes, and a restricted distribution of the disparity/optical flow values, which weakens the generalizing ability from synthetic to real world.

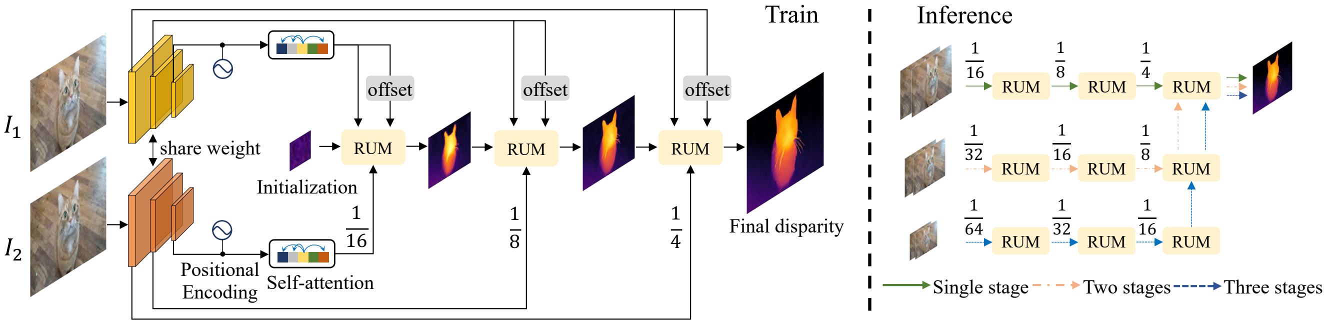


Figure 2. An overview of our proposed network. Left: A pair of stereo images I_1 and I_2 are fed into two shared-weight feature extraction networks to produce a 3-level feature pyramid, which is used to compute different scales of correlations in the 3 stages of cascaded recurrent networks. The feature pyramid of I_1 also provides context information for latter update blocks and offsets computation. In each stage of the cascades, the features and the predicted disparities are refined iteratively using the Recurrent Update Module (RUM, Sec. 3.2), and the final output disparity of the former stage is fed to the next as an initialization. For each iteration in RUM, we apply Adaptive Group Correlation Layer (AGCL, Sec. 3.1) to compute the correlation. Right: Our proposed stacked cascaded architecture in inference phase, which takes an image pyramid as input, taking advantage of multi-level context, as detailed in Sec. 3.3 .

3. Methods

In this section we present the key components of the proposed Cascaded REcurrent Stereo matching network (CREStereo) and our new synthetic dataset.

3.1. Adaptive Group Correlation Layer

We observe that it is difficult to implement perfect calibration for real-world stereo cameras. For instance, the two cameras may not be strictly placed on the horizontal epipolar line, resulting in slight rotations in 3D space; or images from camera lenses usually have residual distortion even after they are rectified. As a result, for a stereo image pair, the corresponding points may not locate on the same scanline. We thus propose an Adaptive Group Correlation Layer (AGCL) to reduce the matching ambiguity in this situation, achieving better performance compared to all-pairs matching [23, 45] while only local correlation is computed.

Local Feature Attention. Instead of computing global correlation for every pair of pixels, we only match points in a local window to avoid large memory consumption and computation cost. In light of LoFTR [41] for sparse feature matching, we add an attention module before correlation computation in the first stage of cascades in order to aggregate global context information in single or cross feature maps. Following [41], we add positional encoding to the backbone output, which enhances positional dependence of the feature maps. The self and cross attention is computed alternately, where a linear attention layer is used to reduce computation complexity.

2D-1D Alternate Local Search. Different from the flow estimation network RAFT [45] and its stereo version [23] where all-pairs correlation is computed by a matrix multiplication of two $C \times H \times W$ feature maps, which outputs a 4D $H \times W \times H \times W$ or 3D $H \times W \times W$ cost volume, we only compute correlation in a local search window that outputs a much smaller volume of $H \times W \times D$ to save the memory and computation cost. H and W denote the height and width of the feature maps, and D is the number of correlation pairs much smaller than W. Our correlation computation is also distinct from cost volume based stereo networks like [7, 18, 49, 51] where the search range is related to the maximum displacement of foreground objects. This fixed range is much larger than the number of local correlation pairs we use, which leads to more noisy interference. Furthermore, we don't need to preset the range when the model generalizes to stereo pairs with different baselines.

Given two resampled and attended feature maps $\mathbf{F}_1$ and $\mathbf{F}_2$, the local correlation at position (x, y) can be denoted as

$$\mathrm{Corr}(x, y, d) = \frac{1}{C} \sum_{i=1}^{C} \mathbf{F}_1(i, x, y) \mathbf{F}_2(i, x', y'), \quad (1)$$

where $x' = x + f(d)$, $y' = y + g(d)$, $\mathrm{Corr}(x, y, d) \in \mathbb{R}^{H \times W \times D}$ is the matching cost of d-th ($d \in [0, D-1]$) correlation pair, C is the number of feature channels, $f(d)$ and $g(d)$ denote the fixed offset of current pixels in horizontal and vertical directions.

Traditionally, search direction between two rectified images only lies on the epipolar line in stereo matching. To deal with non-ideal stereo rectification cases, we adopt a 2D-1D alternate local search strategy to improve the matching accuracy. In 1D search mode, we set $g(d) = 0$ and $f(d) \in [-r, r]$, where $r = 4$. Positive displacement value of $f(d)$ is reserved to adjust inaccurate results after every iterative sampling. The results computed by Eq. 1 are stacked and concatenated at channel dimension for the final correlation volume. In 2D search mode, a $k \times k$ grid with dilation l similar to dilated convolution [53] is used for correlation computation. We set $k = \sqrt{2r+1}$ to make sure the out-

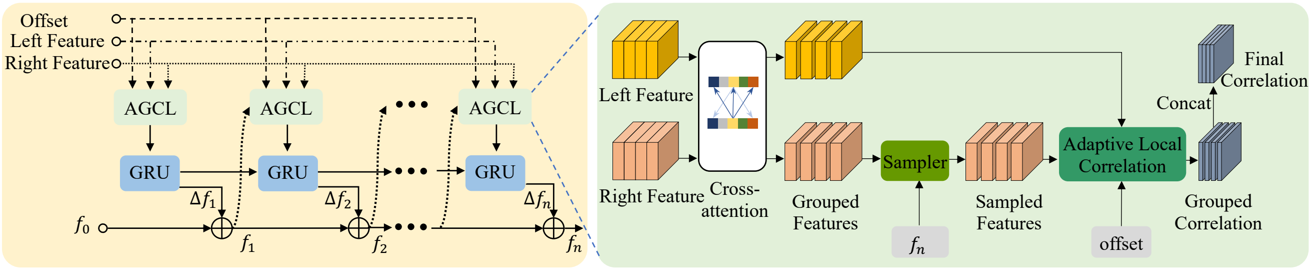


Figure 3. The architecture of proposed modules. Left: Recurrent Update Module (RUM). Right: Adaptive Group Correlation Layer (AGCL). Details are described in Sec. 3.2 and Sec. 3.1, respectively.

put features have the same number of channels so that they can be fed to a shared-weight update block. Cooperating with iterative resampling, alternate local search also acts as a propagation module for recurrent refinement, where the network learns to replace the biased prediction on current location with its more accurate neighbors.

Deformable search window. Stereo matching often suffers from ambiguity in occlusion or textureless areas. Correlation computed in a fixed-shape local search window tends to be vulnerable to those cases. Extending deformable convolution [57] to correlation computation, we use a content adaptive search window for correlation pairs generation, which is different from AANet [49] where a similar strategy is adopted only in cost aggregation. With the learned additional offset $\mathrm{d}x$ and $\mathrm{d}y$, the new correlation can be computed as

$$\mathrm{Corr}(x, y, d) = \frac{1}{C}\sum_{i=1}^{C} \mathbf{F}_1(i, x, y)\mathbf{F}_2(i, x'', y'') \qquad (2)$$

where $x'' = x + f(d) + \mathrm{d}x$, $y'' = y + g(d) + \mathrm{d}y$. Fig. 4 shows how offsets change the formation of a conventional search window.

Group-wise correlation. Inspired by [12] which introduces the group-wise 4D cost volume, we split the feature map into $\mathcal{G}$ groups to compute local correlation individually. Finally, we concatenate $\mathcal{G}$ correlation volumes of $D \times H \times W$ in channel dimension to get the $\mathcal{G}D \times H \times W$ output volume. The procedure is shown in Fig. 3.

3.2. Cascaded Recurrent Network

For non-texture or repetitive-texture areas, matching is more robust using low-res and high-level feature maps due to large receptive field and sufficient semantic information. However the details of fine structures may be lost in such feature maps. In order to maintain robustness and preserve the details in high-res input simultaneously, we propose cascaded recurrent refinement for correlation computing and disparity updating.

Recurrent Update Module. We build a Recurrent Update Module (RUM) based on GRU blocks and our Adaptive Group Correlation Layer (AGCL). Unlike in RAFT [45] where the feature pyramid is constructed in a single correlation layer with the output being merged into one volume, we compute correlations for every feature map respectively in different cascade levels and refine the disparities for several iterations independently. As is shown in Fig. 3, the "sampler" samples locations of grouped feature taking coordinate grid derived from f_n as input. $\{f_1, ..., f_n\}$ are intermediate predictions of n iterations with initialization f_0. Current correlation volume is constructed with learned offsets $o \in \mathbb{R}^{2\times(2r+1)\times h\times w}$. The GRU blocks update current prediction and feed it to the AGCL in next iteration.

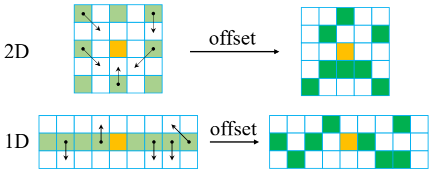


Figure 4. Illustration of the adaptive local correlation. The top and the bottom are 2D and 1D situations respectively, which share the same number of searched neighbors to produce correlation maps in the same shape.

Cascaded Refinement. Except for the first level of cascades, which starts at 1/16 of the input resolution with disparity initialized to all zeros, other levels take the upsampled version of prediction from previous level as initialization. Though handling different levels of refinement, all RUMs share the same weights. After the last refinement level, convex upsampling [45] is conducted to get the final prediction at input resolution.

3.3. Stacked Cascades for Inference

As is discussed in previous sections, during training we use a three-level feature pyramid at fixed resolutions to do hierarchical refinement. However, for images of higher resolution as input, more downsampling should be done in order to enlarge the receptive field for extracted features and correlation computation. But for small objects with large displacement in high resolution images, features in these regions may suffer from deterioration with direct downsampling. To solve this problem, we designed a stacked cascaded architecture with shortcuts for inference. Specifi-

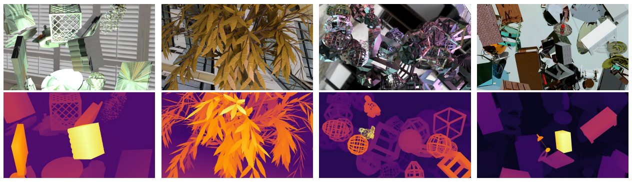


Figure 5. Example image-disparity pairs of our synthetic data featuring various shapes and textures (repetitive-texture, reflective non-texture surface, etc.)

cally, we downsample the image pair in advance constructing an image pyramid and feed them into the same trained feature extraction network to take advantage of multi-level context. An overview of the stacked cascade architecture is shown on the right of Fig. 2, where skip connections in the same stage are not displayed for brevity. For a specific stage of the stacked cascades (denoted as rows in Fig. 2), all the RUMs in that stage will be used together with the last RUM in the stage of higher resolution. All stages of the stacked cascades share the same weight during training, so no fine-tuning is needed.

3.4. Loss Function

For each stage $s \in \{\frac{1}{16}, \frac{1}{8}, \frac{1}{4}\}$ of our feature pyramid, we resize the sequence of output $\{\mathbf{f}_i^s, \cdots, \mathbf{f}_n^s\}$ to the full prediction resolution with the upsampling operator μ_s, and use the exponentially weighted l_1 distance similar to RAFT [45] as the loss function (with γ set to 0.9). Given ground truth disparity $\mathbf{d}_{\text{gt}}$, the total loss is defined as:

$$\mathcal{L} = \sum_s \sum_{i=1}^{n} \gamma^{n-i} ||\mathbf{d}_{\text{gt}} - \mu_s(\mathbf{f}_i^s)||_1 \tag{3}$$

3.5. Synthetic Training Data

Compared to previous synthetic datasets, our data generating pipeline devotes extra attention to challenging cases in real-world scenes, and features various enhancements. We make use of Blender [3] to generate our synthetic training data. Each scene consists of left-right image pairs and the corresponding pixel-accurate dense disparity map, captured with dual virtual cameras and customarily positioned objects. Our major design considerations are described as below, with some examples shown in Fig. 5.

Shape. We diversify the shapes of the models used as the main scene content with multiple sources: 1) The ShapeNet [6] dataset with over 40,000 3D models of common objects with varied shapes, forming our basic source of content. 2) Blender's sapling tree gen add-on, providing fine-detailed and cluttered disparity maps. 3) We use blender's internal basic shapes combined with the wireframe modifier to generate models for challenging scenes featuring holes and open-work structures.

Lighting and texture. We place different types of lights with random color and luminance at random position inside the scene, resulting in a complex lighting environment. Real world images are used as textures for objects and scene background, particularly hard scenes containing repeated patterns or lacking visible features. Additionally, we exploit the light tracing ability of Blender's Cycles renderer and randomly set objects as transparent or with metallic reflection, in order to cover real-world scenes with similar attributes.

Disparity distribution. To cover different baseline settings, we make efforts to ensure the disparity of the generated data distributes smoothly within a wide range. We put objects within a frustum-shaped space formed by the cameras' field of view and a max distance. The exact position of each object is randomly chosen from a probability distribution, then the object is scaled according to its distance to prevent blocking the view. This practice results in a randomized but controllable disparity distribution.

4. Experiments

4.1. Datasets and Evaluation Metrics

We evaluate our method on three popular public benchmarks. Middlebury 2014 [33] provides 23 high-resolution image pairs under different lighting environments. Captured with large-baseline stereo cameras, the maximum disparity in Middlebury can exceed 600 pixels. ETH3D [36] consists of 27 monochrome stereo image pairs with disparity sampled by a laser scanner, covering both indoor and outdoor scenes. KITTI 2012/2015 [28] consists of 200 wide-angle stereo image pairs of street views, with lidar-sampled sparse disparity ground truth.

In addition to our rendered dataset, we collect major public datasets for training, including Sceneflow [27], Sintel [5] and Falling Things [46]. Sceneflow contains 39k training pairs of multiple synthetic scene setups. Falling things contains a large amount of images from scenes of household object models. Sintel provides 1.2k stereo pairs from various synthetic sequences. The other data sources we utilize are InStereo2K [1], Carla [9] and AirSim [37].

For evaluation, we follow the popular metrics including AvgErr (average error), Bad2.0 (percentage of pixels with disparity error larger than 2 pixels) [35, 36], D1-all (percentage of disparity outlier pixels in left image) [11], etc.

4.2. Implementation Details

Training. Our network is implemented with Pytorch [31] framework. The model is trained on 8 NVIDIA GTX 2080Ti GPUs, with a batch size of 16. The whole training process is set to 300,000 iterations. We use the Adam [19] optimizer with a standard learning rate of 0.0004. We perform a warm-up process of 6,000 iterations at the beginning of the training where the learning rate is linearly increased from 5% to 100% of the standard value. After 180,000 iter-

Method	Middlebury		ETH3D	
	Bad 2.0	AvgErr	Bad 1.0	Avgerr
2D all-pairs [45]	47.38	5.62	6.17	0.38
1D all-pairs [23]	44.41	4.93	6.03	0.38
1D local	19.87	3.03	3.13	0.28
2D local	20.70	2.99	3.33	0.29
1D+2D local	19.23	3.01	3.05	0.28
1D local, 2 levels	13.84	2.24	2.35	0.23
2D local, 2 levels	14.07	2.15	2.09	0.23
1D+2D local, 2 levels	**12.48**	1.99	2.20	0.22
1D+2D local, 3 levels	12.67	**1.80**	**2.01**	**0.21**
w/o def. & group. & atten.	6.86	1.11	1.26	0.19
w/o deformable search	6.84	1.08	1.22	0.19
w/o group-wise correlation	6.82	1.08	1.20	0.18
w/o attention	6.49	1.07	1.22	0.18
full method	**6.46**	**1.05**	**1.03**	**0.17**

Table 1. Ablation study for RUMs. The top half is comparisons for different forms of correlation layers and different levels of cascades, trained on public datasets except Middlebury and ETH3D. And the bottom half is evaluation for key components in AGCL, trained on full datasets.

ations, the learning rate is linearly decreased down to 5% of the standard value towards the end of the training process. The model is trained with an input size of 384 × 512. All training samples undergo a set of augmentation operations before getting fed into the model.

Augmentation. To imitate the camera module inconsistencies and non-ideal rectification, we employ multiple data augmentation techniques for training. Firstly, we apply asymmetric chromatic augmentations for the two inputs respectively, including shifts in brightness, contrast and gamma. To further enhance the robustness for rectification error in real-world images, we conduct spatial augmentation applied only to the right image: slightly random homography transformation and vertical shift at a very small range (< 2 pixels). To avoid mismatching in ill-posed regions, we use random rectangle occlusion patches with height and width between 50 and 100 pixels. Finally, to fit input data from various sources into the network's training input size, the group of stereo images and disparity undergoes random resize and crop operations.

4.3. Ablation Study

In this section we evaluate our model on different settings to prove the effectiveness of the network components. Besides the ablation study for the stacked cascades, all evaluation resolutions are 768 × 1024.

Correlation types. In order to compare the effect of different types of correlations, we replace our correlation layers with other forms. As shown in Tab. 1, both 2D and 1D all-pairs correlation used in [45] and [23] lead to a substantial drop in accuracy compared with their local forms.

Method	Input size	Middlebury	
		Bad 2.0	AvgErr
single cascade	768 × 1024	6.46	1.05
single cascade	1024 × 1536	6.00	1.61
2 stacked cascades	1024 × 1536	5.30	0.94
2 stacked cascades	1536 × 2048	**4.53**	0.93
3 stacked cascades	1536 × 2048	4.58	**0.92**

Table 2. Ablation study for stacked cascaded architecture during inference.

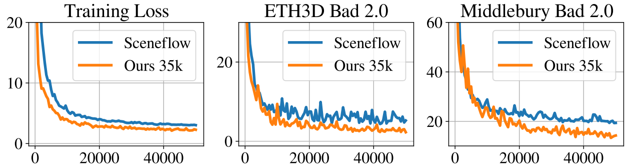


Figure 6. Training loss and ETH3D / Middlebury validation error of models trained with Sceneflow and our synthetic dataset.

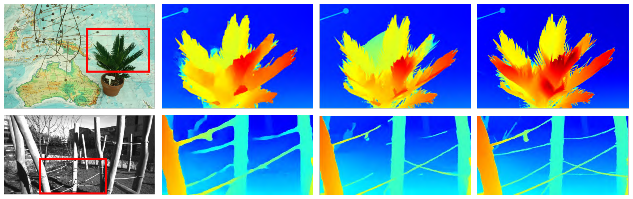
When we replace the alternate local correlation with a single 2D or 1D correlation, it harms the final precision, which is more evident when the network contains more than 1 level of cascades because the rectification error increases with the resolution.

Components in AGCL. As shown in the bottom half of Tab. 1, using a fixed correlation window without learned offsets degrades the accuracy which demonstrates the effectiveness of the adaptive mechanism. Replacing group correlation with single form and removing the local feature attention modules both deteriorate the accuracy.

Cascaded RUMs. We compare the performance of different numbers of cascade stages. As is shown in Tab. 1, using a single RUM without cascades leads to a substantial drop in precision. When changing the number of cascades, the prediction error decreases evidently when more levels of cascades are used while the correlation type keeps the same. This demonstrates the importance of our cascaded architecture.

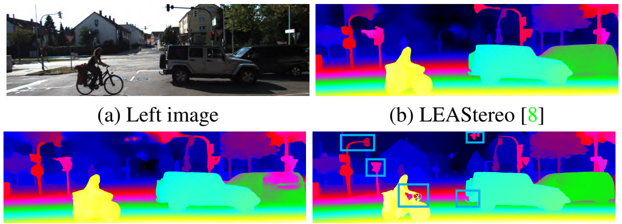
Stacked cascades. During inference, we feed the cascades using different levels of image pyramid as input while sharing the same trained parameters. We compare the performance of different stages of cascades with various of resolutions on Middlebury. As shown in Tab. 2, the prediction error increases with the input size when only a single cascade is used. Multi-level input helps to reduce the error substantially, which demonstrates that our stacked cascades scheme enjoys a great improvement for disparity accuracy.

New synthetic data. To analyze the effectiveness of our proposed synthetic data, we sample 35,000 pairs of images from our training dataset and compare with similar-sized Sceneflow [27]. Both datasets are used to train our model with the same augmentation for 50,000 iterations. As shown in Fig. 6, our synthetic data results in lower training loss and



(a) Left image (b) HITNet (c) RAFT-Stereo (d) Ours

Figure 7. Visual comparisons on Middelbury and ETH3D with HITNet [44] and RAFT-Stereo [23].



(a) Left image (b) LEAStereo [8]

(c) AANet [49] (d) Ours

Figure 8. Visual comparisons with other methods on one case of KITTI 2015 test set. Our method preserves more details.

better performance in both ETH3D and Middlebury validation data. This demonstrate that our dataset is more advantageous in domain generalization.

4.4. Comparisons with State-of-the-art

Middlebury. We train our network on 23 pairs of images (including 13 additional pairs with ground truth) from Middlebury 2014 dataset together with our full training set without fine-tuning. The proportion of Middlebury training set is augmented to 2% of the full training set. We evaluate the test set at 1536 × 2048 using resized full-resolution images where 2-stage inference is adopted, and the results are submitted to the online leaderboard . We achieve the 1st place on the majority of the metrics among more than 120 other methods, surpassing the published state-of-the-art by 21.73% on the bad 2.0 metric, 31.00% on the A95 metric. The quantitative comparison results with other methods are shown in Tab. 3.

ETH3D. We train our network on the whole training set with a proportion of 2% augmented training data from ETH3D low-res two-view stereo dataset. Without fine-tuning, we evaluate the test set at the size of 768 × 1024 where 2-stage inference is adopted. At the time of writing, we achieve state-of-the-art among published methods on the online benchmark for all metrics. Our method surpasses the published state-of-the-art by 59.84% on the bad 1.0 metric. Quantitative comparisons are shown in Tab. 4.

Method	Bad 2.0	Bad 1.0	AvgErr	RMS	A95
CREStereo (Ours)	**3.71**^{1}	**8.25**^{1}	**1.15**^{1}	**7.70**^{1}	**1.58**^{1}
RAFT-Stereo [23]	4.74^{2}	9.37^{2}	1.27^{2}	8.41^{3}	2.29^{2}
LocalExp [43]	5.43^{5}	13.9^{10}	2.24^{13}	13.4^{23}	4.81^{17}
HITNet [44]	6.46^{14}	13.3^{4}	1.71^{4}	9.97^{5}	4.26^{9}
LEAStereo [8]	7.15^{18}	20.8^{40}	1.43^{3}	8.11^{2}	2.65^{3}
SDR [50]	7.69^{24}	18.8^{32}	2.94^{32}	15.4^{43}	7.13^{30}
MC-CNN-acrt [54]	8.08^{27}	17.1^{23}	3.82^{58}	21.3^{86}	14.1^{55}
CFNet [38]	10.1^{37}	19.6^{33}	3.49^{46}	15.4^{44}	16.4^{58}
HSMNet [48]	10.2^{38}	24.6^{48}	2.07^{5}	10.3^{8}	4.32^{10}
AdaStereo [39]	13.7^{59}	29.5^{61}	2.22^{10}	10.2^{7}	5.67^{25}
AANet++ [49]	15.4^{66}	25.5^{51}	6.37^{94}	23.5^{103}	48.8^{112}

Table 3. Quantitative results on Middlebury benchmark.

Method	Bad 1.0	Bad 0.5	AvgErr	RMSE
CREStereo (Ours)	**0.98**^{1}	**3.58**^{1}	**0.13**^{1}	**0.28**^{1}
RAFT-Stereo [23]	2.44^{5}	7.04^{4}	0.18^{3}	0.36^{3}
HITNet [44]	2.79^{9}	7.83^{6}	0.20^{6}	0.46^{10}
AdaStereo [39]	3.09^{12}	10.22^{18}	0.24^{14}	0.44^{7}
CFNet [38]	3.31^{17}	9.87^{15}	0.24^{14}	0.51^{19}
GwcNet [12]	3.66^{25}	12.04^{37}	0.29^{40}	0.67^{52}
iResNet [22]	3.68^{26}	10.26^{19}	0.24^{14}	0.51^{19}
HSMNet [48]	4.00^{36}	11.33^{28}	0.28^{36}	0.62^{43}
AANet [49]	5.01^{52}	13.16^{45}	0.31^{45}	0.68^{57}
GANet [55]	6.56^{67}	25.41^{108}	0.43^{83}	0.75^{73}

Table 4. Quantitative results on ETH3D benchmark.

KITTI. Different from the training procedure for Middlebury and ETH3D, we fine-tune the model pre-trained on the full training set for another 50K iterations on KITTI 2012 and 2015 training sets. The initial learning rate is set to 0.0001. We augment the proportion of KITTI datasets to 75% with the rest part randomly sampled from the whole training set. During evaluation, we pad the input to 384×1248 before feeding to the network and single stage inference is adopted. We achieve competitive performance on both datasets , surpassing LEAStereo [8] in KITTI 2012 by 9.47% on Out-Noc under 2 pixels error threshold. We show a visual comparison of KITTI 2015 in Fig. 8.

4.5. Practical Performance

Compared with the real-world images from standard stereo datasets which is limited in numbers and scenes, images taken from consumer-level devices pose greater challenges to stereo matching. For fair comparisons, we trained all other stereo networks with author-released code and recommended settings on our full training set.

Holopix50K. Fig. 9 shows the qualitative comparison results of our network with several published stereo matching on Holopix50K [16] dataset in varied scenes. Prerectification were performed to eliminate possible negative disparity. The visual results show that our method has a significant advantage in thin objects like cat whiskers and wire meshes. We also achieve better performance on textureless areas like walls and windows.

Disturbed ETH3D. We simulate common disturbances in practical scenes on ETH3D dataset to test the robustness of our proposed method and list the quantitative results in

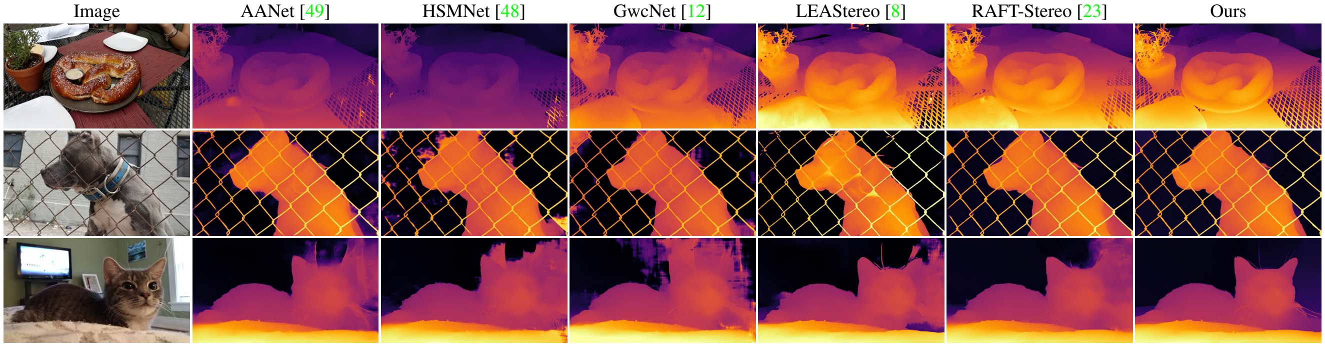


Figure 9. Comparison of results from different methods on Holopix50K [16] dataset. Zoom in for best view.

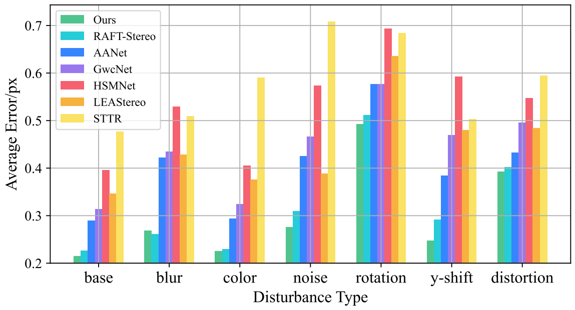


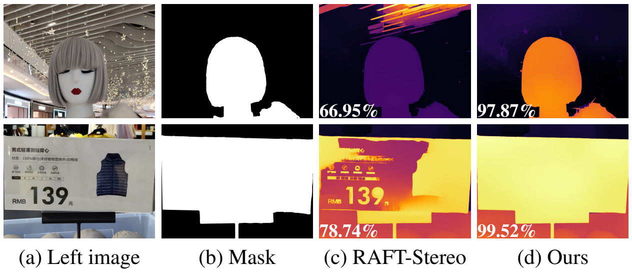
Figure 10. Comparisons among methods on the ETH3D training dataset with different types of disturbances.

Method	mxIoU	mxIoUbd
Ours	**97.50%**	**72.61%**
RAFT-Stereo [23]	94.58%	69.26%
HSMNet [48]	91.70%	60.17%
AANet [49]	91.02%	63.70%
GwcNet [12]	90.77%	64.26%
STTR [21]	90.82%	62.12%
LEAStereo [8]	92.38%	58.06%

Table 5. Quantitative results on 400 smartphone captured scenes. We choose the resolution with best performance for every method.

Fig. 10. The disturbances here include image blur, color transform, chromatic noise, image perspective transform, vertical shift and spatial distortion. The results demonstrate that our method is less vulnerable to these disturbances.

Smartphone photos. Because it is difficult to obtain ground truth disparity in real-world scenes, an empirical way is to manually label a foreground mask M_f for evaluating the disparity quality [25]. The metric of IoU (intersection over union) is commonly used in segmentation tasks. For a disparity map, we can place a threshold t to obtain a foreground mask M_t, where the disparity values of the foreground are larger than t. The "mxIoU" means the maximum IoU between M_f and M_t by changing t. Similarly, "mxIoUbd" means mxIoU in a banded area within p (we set $p = 4$) pixels from the boundary of M_f. The quantitative and qualitative comparison results are shown in Tab. 5 and Fig. 11 respectively.



(a) Left image (b) Mask (c) RAFT-Stereo (d) Ours

Figure 11. Comparison of predicted disparities of repetitive-texture and non-texture cases in smartphone photos with RAFT-Stereo [23]. The mxIoU scores are marked in the figure.

5. Conclusion

Despite unprecedented success of deep stereo networks, obstacles remain for accurately recovering disparities in real-world scenes. In this paper, we have presented CREStereo, a novel stereo matching network that attains state-of-the-art results on both public benchmarks and real-world scenes. Our key message here is that, both network architecture and training data deserve rigorous thoughts in order for an algorithm to truly work in the real world. Via cascaded recurrent network with adaptive correlation, we are able to recover delicate depth details better than existing methods; and we manage to better handle hard-case scenes like non-texture or repetitive-texture areas through careful design of our synthetic dataset. A limitation of our method is that the model is not yet efficient enough to run in current mobile applications. Future improvements could be made to adapt our network for various portable devices, preferably in real-time.

Acknowledgement

We thank all the reviewers for their valuable comments. This work was supported by the National Natural Science Foundation of China (NSFC) under grants No.61872067.

References

[1] Wei Bao, Wei Wang, Yuhua Xu, Yulan Guo, Siyu Hong, and Xiaohu Zhang. Instereo2k: a large real dataset for stereo matching in indoor scenes. *Science China Information Sciences*, 63(11):1–11, 2020. 5

[2] Stan Birchfield and Carlo Tomasi. Depth discontinuities by pixel-to-pixel stereo. *International Journal of Computer Vision*, 35(3):269–293, 1999. 2

[3] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam, 2021. 2, 5

[4] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001. 2

[5] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *Proc. ECCV*, pages 611–625, 2012. 1, 2, 5

[6] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 5

[7] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proc. CVPR*, pages 5410–5418, 2018. 2, 3

[8] 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. *arXiv preprint arXiv:2010.13501*, 2020. 1, 2, 7, 8

[9] Jean-Emmanuel Deschaud. Kitti-carla: a kitti-like dataset generated by carla simulator. *arXiv preprint arXiv:2109.00892*, 2021. 5

[10] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2758–2766, 2015. 2

[11] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc. CVPR*, pages 3354–3361, 2012. 2, 5

[12] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. In *Proc. CVPR*, pages 3273–3282, 2019. 2, 4, 7, 8

[13] Ju He, Enyu Zhou, Liusheng Sun, Fei Lei, Chenyang Liu, and Wenxiu Sun. Semi-synthesis: A fast way to produce effective datasets for stereo matching. In *Proc. CVPR*, pages 2884–2893, 2021. 2

[14] Heiko Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *Proc. CVPR*, volume 2, pages 807–814, 2005. 2

[15] Heiko Hirschmüller, Peter R Innocent, and Jon Garibaldi. Real-time correlation-based stereo vision with reduced border errors. *International Journal of Computer Vision*, 47(1):229–246, 2002. 2

[16] Yiwen Hua, Puneet Kohli, Pritish Uplavikar, Anand Ravi, Saravana Gunaseelan, Jason Orozco, and Edward Li. Holopix50k: A large-scale in-the-wild stereo image dataset. In *Proc. CVPRW*, June 2020. 1, 7, 8

[17] 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 *Proc. CVPR*, pages 66–75, 2017. 2

[18] Sameh Khamis, Sean Fanello, Christoph Rhemann, Adarsh Kowdle, Julien Valentin, and Shahram Izadi. Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. In *Proc. ECCV*, pages 573–590, 2018. 2, 3

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[20] Andreas Klaus, Mario Sormann, and Konrad Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *Proc. ICPR*, volume 3, pages 15–18, 2006. 2

[21] Zhaoshuo Li, Xingtong Liu, Nathan Drenkow, Andy Ding, Francis X Creighton, Russell H Taylor, and Mathias Unberath. Revisiting stereo depth estimation from a sequence-to-sequence perspective with transformers. *arXiv preprint arXiv:2011.02910*, 2020. 2, 8

[22] Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Wei Chen, Linbo Qiao, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. In *Proc. CVPR*, pages 2811–2820, 2018. 2, 7

[23] Lahav Lipson, Zachary Teed, and Jia Deng. Raft-stereo: Multilevel recurrent field transforms for stereo matching. *arXiv preprint arXiv:2109.07547*, 2021. 1, 2, 3, 6, 7, 8

[24] Charles Loop and Zhengyou Zhang. Computing rectifying homographies for stereo vision. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 1, pages 125–131. IEEE, 1999. 1

[25] Chenchi Luo, Yingmao Li, Kaimo Lin, George Chen, Seok-Jun Lee, Jihwan Choi, Youngjun Francis Yoo, and Michael O Polley. Wavelet synthesis net for disparity estimation to synthesize dslr calibre bokeh effect on smartphones. In *Proc. CVPR*, pages 2407–2415, 2020. 2, 8

[26] Nikolaus Mayer, Eddy Ilg, Philipp Fischer, Caner Hazirbas, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. What makes good synthetic training data for learning disparity and optical flow estimation? *International Journal of Computer Vision*, 126(9):942–960, 2018. 2

[27] 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 *Proc. CVPR*, pages 4040–4048, 2016. 1, 2, 5, 6

[28] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proc. CVPR*, pages 3061–3070, 2015. 5

[29] Jiahao Pang, Wenxiu Sun, Jimmy SJ Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *Proc. CVPRW*, pages 887–895, 2017. 2

[30] Jiahao Pang, Wenxiu Sun, Chengxi Yang, Jimmy Ren, Ruichao Xiao, Jin Zeng, and Liang Lin. Zoom and learn: Generalizing deep stereo matching to novel domains. In *Proc. CVPR*, pages 2070–2079, 2018. 2

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 5

[32] Lars Rehm. Evaluating computational bokeh: How we test smartphone portrait modes. https://www.dxomark.com/evaluating-computational-bokeh-test-smartphone-portrait-modes/, 2018. Accessed: 2021-11-15. 1

[33] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German Conference on Pattern Recognition*, pages 31–42, 2014. 5

[34] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, 2002. 1

[35] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, 2002. 2, 5

[36] Thomas Schops, Johannes L Schonberger, 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 *Proc. CVPR*, pages 3260–3269, 2017. 2, 5

[37] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017. 5

[38] Zhelun Shen, Yuchao Dai, and Zhibo Rao. Cfnet: Cascade and fused cost volume for robust stereo matching. In *Proc. CVPR*, pages 13906–13915, 2021. 7

[39] Xiao Song, Guorun Yang, Xinge Zhu, Hui Zhou, Zhe Wang, and Jianping Shi. Adastereo: a simple and efficient approach for adaptive stereo matching. In *Proc. CVPR*, pages 10328–10337, 2021. 2, 7

[40] Deqing Sun, Daniel Vlasic, Charles Herrmann, Varun Jampani, Michael Krainin, Huiwen Chang, Ramin Zabih, William T Freeman, and Ce Liu. Autoflow: Learning a better training set for optical flow. In *Proc. CVPR*, pages 10093–10102, 2021. 2

[41] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proc. CVPR*, pages 8922–8931, 2021. 3

[42] Jian Sun, Nan-Ning Zheng, and Heung-Yeung Shum. Stereo matching using belief propagation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(7):787–800, 2003. 2

[43] Tatsunori Taniai, Yasuyuki Matsushita, Yoichi Sato, and Takeshi Naemura. Continuous 3d label stereo matching using local expansion moves. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 40(11):2725–2739, 2017. 7

[44] Vladimir Tankovich, Christian Hane, Yinda Zhang, Adarsh Kowdle, Sean Fanello, and Sofien Bouaziz. Hitnet: Hierarchical iterative tile refinement network for real-time stereo matching. In *Proc. CVPR*, pages 14362–14372, 2021. 1, 2, 7

[45] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Proc. ECCV*, pages 402–419, 2020. 2, 3, 4, 5, 6

[46] Jonathan Tremblay, Thang To, and Stan Birchfield. Falling things: A synthetic dataset for 3d object detection and pose estimation. In *Proc. CVPRW*, pages 2038–2041, 2018. 1, 2, 5

[47] Geert Van Meerbergen, Maarten Vergauwen, Marc Pollefeys, and Luc Van Gool. A hierarchical symmetric stereo algorithm using dynamic programming. *International Journal of Computer Vision*, 47(1):275–285, 2002. 2

[48] Jialiang Wang, Varun Jampani, Deqing Sun, Charles Loop, Stan Birchfield, and Jan Kautz. Improving deep stereo network generalization with geometric priors. *arXiv preprint arXiv:2008.11098*, 2020. 7, 8

[49] Haofei Xu and Juyong Zhang. Aanet: Adaptive aggregation network for efficient stereo matching. In *Proc. CVPR*, pages 1959–1968, 2020. 2, 3, 4, 7, 8

[50] Tingman Yan, Yangzhou Gan, Zeyang Xia, and Qunfei Zhao. Segment-based disparity refinement with occlusion handling for stereo matching. *IEEE Trans. on Image Processing*, 28(8):3885–3897, 2019. 7

[51] Gengshan Yang, Joshua Manela, Michael Happold, and Deva Ramanan. Hierarchical deep stereo matching on high-resolution images. In *Proc. CVPR*, pages 5515–5524, 2019. 2, 3

[52] Qingxiong Yang, Liang Wang, Ruigang Yang, Henrik Stewénius, and David Nistér. Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(3):492–504, 2008. 2

[53] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *Proc. CVPR*, pages 472–480, 2017. 3

[54] Jure Zbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. In *Proc. CVPR*, pages 1592–1599, 2015. 2, 7

[55] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *Proc. CVPR*, pages 185–194, 2019. 2, 7

[56] Zhengyou Zhang. Determining the epipolar geometry and its uncertainty: A review. *International journal of computer vision*, 27(2):161–195, 1998. 1

[57] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proc. CVPR*, pages 9308–9316, 2019. 4