

ORStereo: Occlusion-Aware Recurrent Stereo Matching for 4K-Resolution Images

Yaoyu Hu¹, Wenshan Wang¹, Huai Yu¹, Weikun Zhen¹, Sebastian Scherer¹

Abstract—Stereo reconstruction models trained on small images do not generalize well to high-resolution data. Training a model on high-resolution image size faces difficulties of data availability and is often infeasible due to limited computing resources. In this work, we present the Occlusion-aware Recurrent binocular Stereo matching (ORStereo), which deals with these issues by only training on available low disparity range stereo images. ORStereo generalizes to unseen high-resolution images with large disparity ranges by formulating the task as residual updates and refinements of an initial prediction. ORStereo is trained on images with disparity ranges limited to 256 pixels, yet it can operate 4K-resolution input with over 1000 disparities using limited GPU memory. We test the model’s capability on both synthetic and real-world high-resolution images. Experimental results demonstrate that ORStereo achieves comparable performance on 4K-resolution images compared to state-of-the-art methods trained on large disparity ranges. Compared to other methods that are only trained on low-resolution images, our method is 70% more accurate on 4K-resolution images.

I. INTRODUCTION

High-resolution and accurate reconstruction of 3D scenes is critical in many applications. For example, LiDAR scanners with a ~ 5 mm accuracy are typically used to build models for civil engineering analysis. However, this level of accuracy is often not sufficient for detailed inspections and scanners are too expensive and heavy to be carried or flown by drones. Stereo cameras, on the other hand, are compact, and potentially high-resolution source of 3D maps if the data can be effectively processed. Most of the recent high-performance stereo matching models are learning-based, however, a small number of them are focusing on high-resolution images.

Recent high-resolution oriented models need dedicated training data to learn the ability to operate large disparity ranges or resort to a combination of models. Yang *et al.*[1] developed a deep-learning model (HSM) that can handle a disparity range of 768 pixels. They collected a high-resolution (2056×2464) dataset to help the training. HSM has impressive performance on the Middlebury dataset [2] which has a relatively larger image size than other public benchmarks. For higher resolution such as 4K, Hu *et al.*[3] combined the SGBM [4] method with a deep-learning model to deal with the high-resolution input.

Data availability and limited computing resources are the main issues for training a high-resolution model. Typical

binocular stereo datasets have an image size of fewer than 2 million pixels (*e.g.* Scene Flow [5] 540×960 or a recent one [6] with 1762×800) and a limited disparity range (*e.g.* ~ 200 pixels). They are far from what we face when using real-world high-resolution images, *e.g.* 4K-resolution with a disparity range of over 1000 pixels. The next challenge is computing resources. Most deep-learning models consume considerable amounts of GPU memory during training on a small-sized image, *e.g.* the AANet [7] needs 2GB per sample with a crop size of 288×756 and a disparity range of 192 pixels. However, a higher resolution such as 4K resolution may require a crop width of over 2000 pixels to effectively cover a disparity range of 1000 pixels. It is hard to train a model with high-resolution data directly on typical GPUs since the memory consumption scales approximately in cubic with image dimension and disparity range.

To this end, we choose to not rely on any high-resolution data. Our goal is trying to answer this question: **how can a model that is trained on low-resolution data be generalized to high-resolution images?** Our philosophy is learning how to incrementally refine the disparity instead of predicting it directly. Disparity refinement is less dependent on the size of the input data. Also, it may be possible to trade time with accuracy by performing the refinement on a smaller scale but multiple times. We propose to handle high-resolution data in a two-phase fashion. The first phase results in an initial down-sampled disparity map. In the second phase, the same model recurrently refines the full-resolution disparity in a patch-wise manner.

While patch-wise processing is widely used for tasks such as object detection, applying a similar strategy for stereo matching faces two issues. First, the small portion of occluded area gets enlarged in some patches, where occluded regions cover most of those patches. These occlusion regions do not have match in stereo images and disturb the refining process. ORStereo explicitly detects occlusions and stabilizes the recurrent updates. Second, the disparity range in the patch is still as large as it is in the original image, which is on one hand, out of the training distribution, and exceeded the patch size on the other. We find that by utilizing proper normalization techniques, a model can learn the ability to generalize to unseen disparity ranges.

Our model is trained on publicly available datasets with small image sizes. For high-resolution evaluations, we collected a set of 4K-resolution stereo images from both photo-realistic simulations and real-world cameras. The main contributions are summarized as follows.

- We propose a two-phase strategy for high-resolution

¹Yaoyu Hu, Wenshan Wang, Huai Yu, Weikun Zhen, and Sebastian Scherer are with the Robotics Institute, Carnegie Mellon University. {yaoyuh, wenshanw, huaiy, weikunz, basti}@andrew.cmu.edu

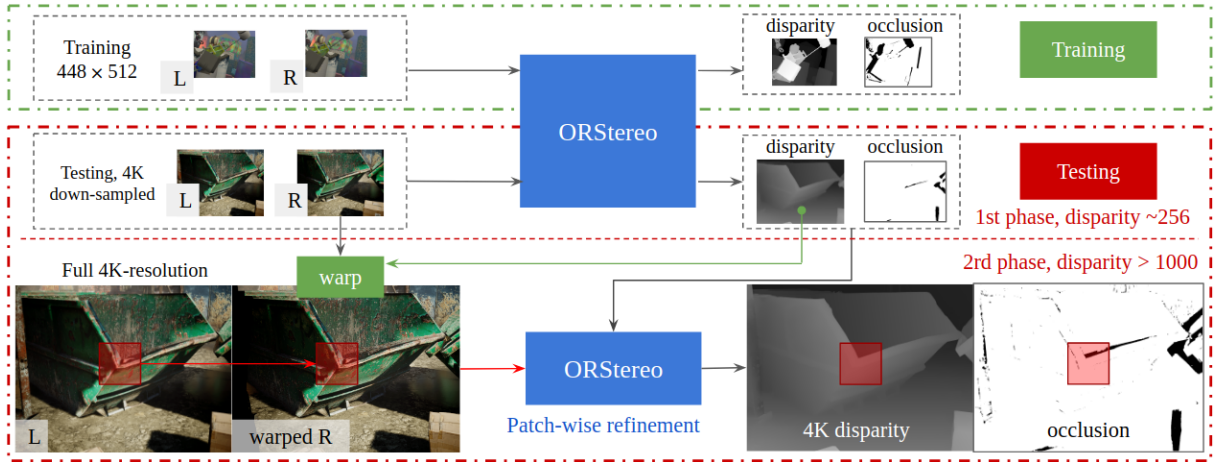


Fig. 1. ORStereo: Occlusion-aware Recurrent binocular Stereo matching. The whole model is trained with low-resolution data. During high-resolution testing, ORStereo is used in two phases. The down-sampled disparity and occlusion are estimated in the first phase. These predictions are up-sampled and patch-wise refined in the second phase.

stereo matching using regional disparity refinements.

- We design a novel structure to recurrently update the disparity and occlusion predictions. The occlusions are explicitly predicted to guide the updates.
- We develop a new local refinement module equipped with special normalization operations.
- We collect a set of 4K-resolution stereo images for evaluation. This dataset is publicly available from the project web page¹.

II. RELATED WORK

Stereo matching is a widely studied topic and there are many works both from geometry-based [8] and learning-based research [9], [10], [11]. Many 3D perception tasks have a similar nature with binocular stereo reconstruction, such as multi-view stereo, monocular depth estimation, and optical flow. We find many valuable inspirations and insights from all those tasks.

Residual prediction and recurrence As stated previously, ORStereo learns to refine the disparity in a patch multiple times to obtain better accuracy. This process is similar to the residual prediction and recurrence. Many existing works train models to predict residuals to refine an initial prediction [12], [13], [14], [15], [16]. However, these models update an initial prediction by a fixed number of steps or dimension scales. In contrast, we introduce disparity improvements with variable steps in a recurrent way. Most of the recurrent models for 3D perception are designed for dealing with sequential data, *e.g.* multi-view stereo models [17], [18]. As for binocular stereo, Jie *et al.* [19] utilized a recurrent neural network (RNN) to keep track of the consistency between the left and right predictions. Recently, RAFT applies an RNN to update optical flow prediction in a way similar to an optimizer, thus the update converges as it proceeds [20]. Inspired by RAFT, we design a model to recurrently update the disparity prediction without deterioration. To handle the

occlusion issue in a small patch from a large image, our model recurrently handles the occlusion.

Occlusion prediction Properly handling occlusion is important for unsupervised learning models, *e.g.* [21], [22]. For supervised models, Ilg *et al.* [23] demonstrated that a CNN (convolutional neural network) can jointly estimate disparity and occlusion. We train ORStereo in a supervised manner. To deal with large occluded areas in small patches, we chose to explicitly identify occlusions and let the model exploit this information. A recent work [24] shares a similar idea where occlusions are used to filter the extracted features before constructing the cost volume.

III. METHOD

The ORStereo is trained on low-resolution images and tested on 4K images (Fig. 1). It consists of 5 components (Fig. 2): a multi-level feature extractor (FE), a base disparity estimator (BDE), a base occlusion mask estimator (BME), a recurrent residual updater (RRU), and a normalized local refinement component (NLR). When training on small images (Fig. 1 Training, Fig. 2), BDE and BME learn to predict an initial disparity image and an occlusion mask, which are feed into the RRU as initial values. The RRU works in a recurrent fashion gradually promotes prediction accuracy. At last, the NLR further refines the RRU’s prediction and outputs the final results. When testing on 4K image, we use the learned model in a two-phase manner. In the first phase, our model estimates down-sampled disparity and occlusion mask. Then in the second phase (Fig. 1 Testing), the RRU and NLR are reused to refine the enlarged patches in the original resolution. The key features of ORStereo is that the RRU and NLR are designed to be generalizable to large disparity ranges. As a result, in testing time, when the 4K images with a large disparity range are presented, the RRU and NLR will be able to improve the estimation accuracy even the disparity range is not seen in the training time.

A. Initial disparity and occlusion estimations

We place two dedicated components, namely the base disparity estimator (BDE) and the base occlusion mask

¹<https://theairlab.org/orstereo>

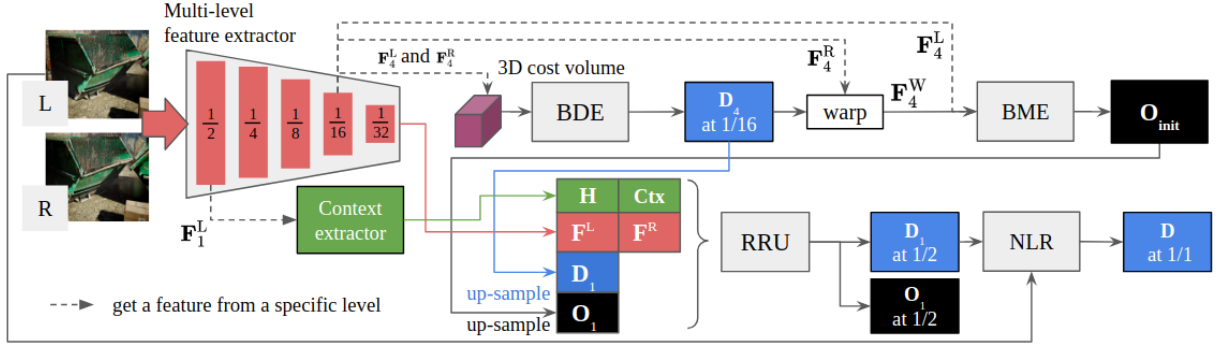


Fig. 2. Structure of ORStereo. We are using the same structure in both the first and second phases. F : feature tensor. D : disparity tensor. O : occlusion mask. Ctx : context information. H : hidden state for the recurrent update. Superscripts: L (left), R (right), and W (warped). Subscripts: the level numbers. BDE: base disparity estimator. BME: base occlusion mask estimator. RRU: recurrent residual updater. NLR: normalized local refinement.

estimator (BME), to calculate initial values for the RRU. The left and right images are first passed through a five-level feature extractor, which has a simplified ResNet [25] structure. The extracted features are denoted as F in Fig. 2. The BDE and BME take the F_4 (Level 4) features at 1/16 of the input size. The BDE is implemented based on the 3D cost volume technique similar to [1]. The BME predicts an initial occlusion mask for the RRU. It is designed as an encoder-decoder structure with skip connections. As shown in Fig. 2, let D_i be the disparity tensor at Level i , we warp F_4^R by D_4 to get F_4^W . By comparing the feature pattern between F_4^L and F_4^W , the BME predicts the initial occlusion mask O_{init} . In testing time, BDE and BME is only used in the first phase. In the second-phase, the initial values for the RRU come from the results of the first phase.

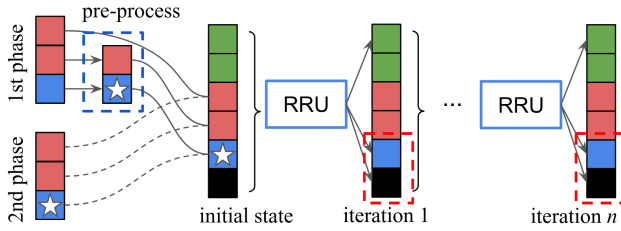


Fig. 3. The RRU pre-processing and iteration. Initial state (from top to bottom): the hidden state H , context Ctx , multi-level features F^L and F^R , disparity D_1 , and occlusion O_1 . O_1 is obtained by resizing the O . Blue dashed box: pre-processing in the first phase. Star: zero disparity. Red dashed box: evaluation of the disparity and occlusion losses. The RRU recurrently updates H , D_1 , and O_1 . The iteration stops when a fixed n is reached or a convergence criterion triggers.

B. Recurrent residual updater

To make the RRU work across two phases, we let the RRU do a pre-process on the disparity and always begin from zero disparity. Also, the RRU needs to properly handle occlusions to stabilize the iteration. Note that the RRU takes all the feature levels (F^L and F^R) and updates disparity and occlusion at Level 1 (D_1 and O_1) with 1/2 image width.

Initial state with zero disparity. The RRU uses multiple input types to start the iteration in both phases, as shown in Fig. 3. In the training time, ORStereo only goes through the first phase. In the second phase as shown in Fig. 1, the full-resolution right image gets warped before going into

ORStereo. Ideally, the warped image should roughly be the same compared with the left image in non-occluded regions. Then patches from the left and the warped images should now correspond to disparity values close to zero at non-occluded pixels. Considering this phenomenon, we add a pre-processing in the first phase to warp F^R and assign an all-zero D_1 to the RRU (Fig. 3). The RRU always has a zero D_1 in the second phase.

The recurrent iterations. We design the RRU by augmenting a GRU (gated recurrent unit) similar to [20]. The detailed model structure is illustrated in Fig. 4. The superscript (i) denotes the iteration step. For the i th iteration, the RRU computes $R_O^{(i)}$ and $R_D^{(i)}$ as the residuals for occlusion $O_1^{(i)}$ and disparity $D_1^{(i)}$ based on the GRU’s hidden variable $H^{(i)}$. Similar to the [20], we keep a context feature, Ctx , as a constant reference to the initial state. The key differences from [20] are that ORStereo recurrently exploits the occlusion information to stabilize the recurrent iteration.

Since the RRU works in a recurrent way, during an evaluation after training, we can apply it like an optimizer by keeping it updating until a pre-defined criterion is satisfied. Here, unsupervised loss functions for similar perception tasks are reasonable criterion candidates. Later in the experiment section, we tested the SSIM[27], which is widely accepted as a robust similarity measure for unsupervised methods. The RRU can keep updating a patch until the SSIM value between the original and warped images goes down. However, we found that the SSIM is not reliable and the RRU simply performs better with fixed and longer iterations.

Residual update of occlusions. Iterations become unstable in occluded regions where little information is useful for stereo matching. To improve robustness, the RRU explicitly handle the occlusion by the occlusion-augmented $H^{(i)}$ (shown in Fig. 4). $O_1^{(i)}$ is represented as a 2-channel tensor in ORStereo and supervised by cross-entropy loss, which does not require a value-bounded $O_1^{(i)}$. However, this unbounded value cause problems for long recurrent updates. Fig. 4 shows that $O_1^{(i-1)}$ first passes through a softmax operator. This ensures an intermediate value-bounded representation, $S_O^{(i)}$, and keeps the gradients of the loss value w.r.t $O_1^{(i)}$ from diminishing. Similar reasoning also explains

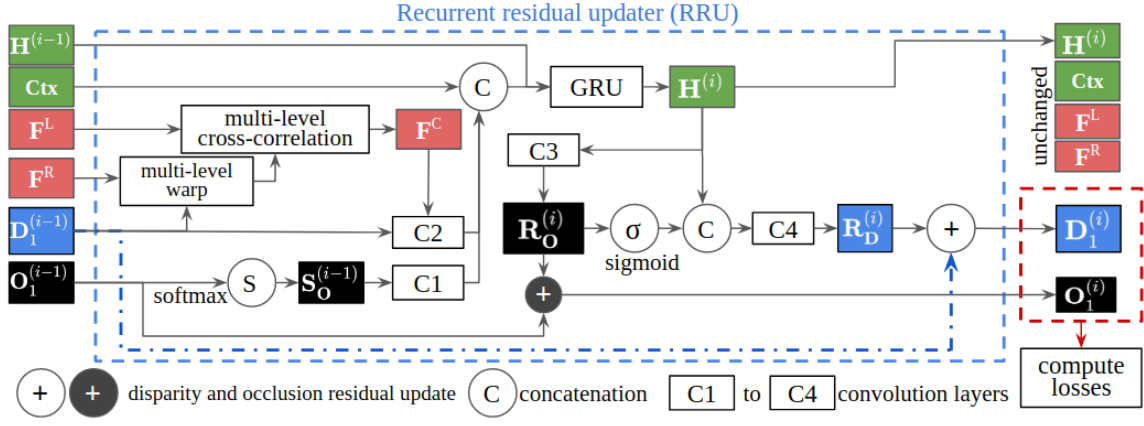


Fig. 4. The structure of the RRU. Residual update for \mathbf{D}_1 is element wise addition, (2). Residual update for \mathbf{O}_1 is defined in (1). The multi-level cross-correlation is implemented by referencing [26].

the purpose of the Sigmoid operator, which bounds the value of $\mathbf{R}_O^{(i)}$. Here, $\mathbf{R}_O^{(i)}$ is a single-channel tensor. $\mathbf{O}_1^{(i-1)}$ gets updated by (1).

$$V(\mathbf{O}_1^{(i)}, j) = V(\mathbf{O}_1^{(i-1)}, j) - (-1)^j \mathbf{R}_O^{(i)} \quad (1)$$

where function $V(\cdot, j)$ takes out the channel at index j from a tensor and $j \in \{0, 1\}$. Equation (1) also makes the iteration more stable. Equation (2) gives the disparity update.

$$\mathbf{D}_1^{(i)} = \mathbf{D}_1^{(i-1)} + \mathbf{R}_D^{(i)} \quad (2)$$

C. Normalized local refinement

Following the RRU, the normalized local refinement module (NLR) adds a final update to the disparity \mathbf{D}_0 , as shown in Fig. 5. The NLR is designed to smooth the object interior and sharpen boundaries by exploiting local consistency between the disparity and the input image. The NLR extracts its own features from the left image and works directly at the image resolution.

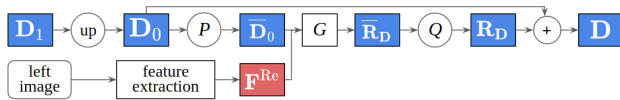


Fig. 5. The normalized local refinement module (NLR). Refer to (3) to (4) for definitions of some symbols.

The NLR also works in both phases. In the second phase the NLR faces a major challenge as disparity values go beyond the training range. The way the NLR handles this challenge is always working under a normalized disparity range. In Fig. 5, the up-sampled disparity, \mathbf{D}_0 , is normalized as $\bar{\mathbf{D}}_0$ by function P . Then module G processes the feature \mathbf{F}^{Re} and $\bar{\mathbf{D}}_0$ to produce residual $\bar{\mathbf{R}}_D$. Then a de-normalization function Q re-scales $\bar{\mathbf{R}}_D$ to \mathbf{R}_D . The final disparity \mathbf{D} is then obtained in the same way of (2). Module G has an encoder-decoder structure. P and Q are defined from (3) to (4).

$$P(\mathbf{D}_0) = (\mathbf{D}_0 - m)/(s + \epsilon) = \bar{\mathbf{D}}_0 \quad (3)$$

$$Q(\bar{\mathbf{R}}_D) = (s + \epsilon)\bar{\mathbf{R}}_D = \mathbf{R}_D \quad (4)$$

where scalar $m = \text{mean}(\mathbf{D}_0)$ and $s = \text{std}(\mathbf{D}_0)$ are the mean and standard deviation of \mathbf{D}_0 . P normalizes the disparity, and Q de-normalizes the residual value. Here, W serves as a local weight which encourages large residuals near local discontinuities. ϵ is a constant hyperparameter. P and Q enable the NLR to generalize to large disparity ranges in the second phase by making the NLR work in a normalized disparity range.

D. Training losses

We adopt a supervised scheme for all the outputs. The total training loss is defined as (5).

$$\begin{aligned} l^{\text{total}} = & \lambda_4^{\text{D}} \text{SL}(\mathbf{D}_4, \mathbf{D}_4^{\text{t}}) + \lambda^{\text{O}} \text{CE}(\mathbf{O}, \mathbf{O}^{\text{t}}) \\ & + \lambda_1^{\text{D}} \sum_{i=1}^n (\gamma^{\text{D}})^{n-i+1} \text{SL}(\mathbf{D}_1^{(i)}, \mathbf{D}_1^{\text{t}}) \\ & + \lambda_1^{\text{O}} \sum_{i=1}^n (\gamma^{\text{O}})^{n-i+1} \text{CE}(\mathbf{O}_1^{(i)}, \mathbf{O}_1^{\text{t}}) \\ & + \lambda^{\text{D}} \text{SL}(\mathbf{D}, \mathbf{D}^{\text{t}}) \end{aligned} \quad (5)$$

where SL and CE are the smooth L1 and cross-entropy loss functions. n is the iteration number and from λ_4^{D} to λ^{D} are the constant weights for different loss values (see Table II).

E. Working in the second phase

We make patches out of four objects for the second phase: the left image, the warped right image, the disparity, and the occlusion mask, all in the full-resolution. Our experiments show that keeping small overlaps among patches achieves better results since accuracy may drop near the patch borders. In the overlap region, the disparity and occlusion predictions are averaged across patches.

IV. EXPERIMENTS

A. Datasets and details of training

Our target is 4K-resolution stereo reconstruction with over 1000 pixels of disparity. ORStereo allows us to train with only small-sized images and a typical disparity range around 200 pixels. We utilize several public datasets for the

TABLE I
EPE METRICS OF ORSTEREO COMPARED WITH THE SOTA MODELS ON THE SCENE FLOW DATASET.

| MCUA[28] | Bi3D[29] | GwcNet[30] | FADNet[31] | GA-Net[32] | WaveletStereo[33] | DeepPruner[34] | SSPCV-Net[35] | AANet[7] | ORStereo (ours) |
|----------|----------|-------------------|------------|------------|-------------------|----------------|---------------|----------|-----------------|
| 0.56* | 0.73 | 0.77 [▲] | 0.83 | 0.84 | 0.84 | 0.86 | 0.87 | 0.87* | 0.74 |

ORStereo only goes through the first phase for this low-resolution test. * Best values from individual works. [▲] Finalpass. ORStereo is trained and tested on the cleanpass subset of the Scene Flow dataset. Some samples are removed as suggested by the Scene Flow dataset.

training, *i.e.*, the Middlebury dataset at 1/4 resolution [2], the Scene Flow [5] dataset ($\sim 35k$ stereo pairs), and the TartanAir [36] datasets ($\sim 18k$ pairs sampled). The Scene Flow and TartanAir datasets do not provide true occlusion labels. We generate them by comparing the left and right true disparities. We will not use the KITTI dataset despite its popularity because it is hard to generate reliable dense occlusion labels from sparse true disparities.

Currently, no public stereo benchmark provides 4K-resolution data. So we collect a set of synthetic 4K-resolution photo-realistic stereo images with ground truth disparity. These images are captured by AirSim [37] in the Unreal Engine. We prepare 100 pairs of stereo images from 7 simulated environments. Some cases may have a disparity range of over 1200 pixels. The evaluation cases in Fig. 7 and 8 are from this dataset. Additional samples are shown in 6. The dataset is only used for evaluation and all images and ground truth disparities are available at the project page.

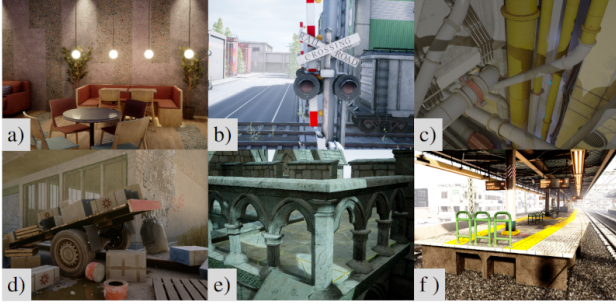


Fig. 6. Sample images from the 100 pairs of 4K-resolution stereo images. 6 of 7 environments are shown: a) restaurant, b) factory district, c) under ground work zone, d) city ruins, e) ancient buildings, f) train station.

Training only happens in the first phase of ORStereo. We use a disparity range of 256 pixels and randomly crop the images to 448×512 pixels. We set the iteration number to 4 for training the RRU. Similar works such as [20], [38] also use fixed number of iterations during training. Other model constants are listed in Table II, where the loss weight constants are chosen to balance the values from different loss components. We train ORStereo with 4 NVIDIA V100 GPUs with a mini-batch of 24 for all experiments. Other training settings varying among different experiments will be shown separately.

TABLE II
THE MODEL CONSTANTS OF ORSTEREO.

| ϵ | λ_1^D | λ^O | λ_1^D | λ_1^O | λ^D | γ^D | γ^O |
|------------|---------------|-------------|---------------|---------------|-------------|------------|------------|
| 1e-6 | 32 | 2 | 2 | 1 | 2 | 0.8 | 0.8 |

The values from λ_1^D to λ^D are selected to make the loss components have similar quantities in the end of a training.

B. Evaluation the first phase on small images

To evaluate the first phase of ORStereo on small-sized images, we train ORStereo on the Scene Flow dataset only and then compare it with the state-of-the-art (SotA) models. We limit the metric computation under 192 pixels, matching the SotA models. Table I lists the results on the testing set of the Scene Flow dataset measured in the average EPE (end point error) metric. ORStereo achieves comparable performance with the SotA models.

C. Evaluation on high-resolution images

For better performance and generalization ability across image sizes and disparity ranges, the Middlebury (at 1/4 resolution) and TartanAir datasets are added to the training. We further augment the data with random color, random flip, and random scale similar to [1]. After training, we apply a 512×512 patch size with an overlap of 32 pixels in the second phase. In Sec. IV-D, after comparing the performance with various iteration numbers, we make the RRU iterate for 10 steps in both the phases for the subsequent evaluations.

Although the training data have small size, we expect ORStereo to learn the ability to refine a patch of disparity at a higher resolution. We first show results on the Middlebury dataset [2] with full resolution. This dataset is still considered as hard for models trained on low-resolution data. We choose three recent models (with available pre-trained weights) that are trained with low-resolution data and have openly evaluated their performance on the Middlebury dataset. Additionally, we include the HSM model[1], which can cover 768 pixels of disparity after training on high-resolution images. The quantitative results are listed in Table III. Trained on smaller image size, we observed that ORStereo delivers accuracy close to the SotA [1] trained on high-resolution images. Later, when the resolution goes up to 4K, ORStereo can still maintain its performance.

TABLE III
COMPARISON ON THE MIDDLEBURY EVALUATION DATASET

| Model & scale | AANet [7] 1/2 | DeepPruner [34] 1/4 | SGBMP [3] full | ORStereo (ours) full | HSM [▲] [1] full |
|---------------|---------------|---------------------|----------------|----------------------|---------------------------|
| EPE | 6.37 | 4.80 | 7.58 | 3.23 | 2.07 |

[▲]This model is trained on high-resolution data. ORStereo achieves near SotA performance without training on high-resolution data. Non-occluded EPE is reported. All EPE values are associated with specific image scales. All values are from the evaluation set and published on the website of Middlebury dataset under the "test dense avgerr nonocc" category.

Some of the models in Table III are unable to operate 4K-resolution images with a limited memory budget or their effective disparity ranges are not enough, we down-sample the input until a model can handle it. Then the results are re-scaled to the original resolution before evaluation. Besides

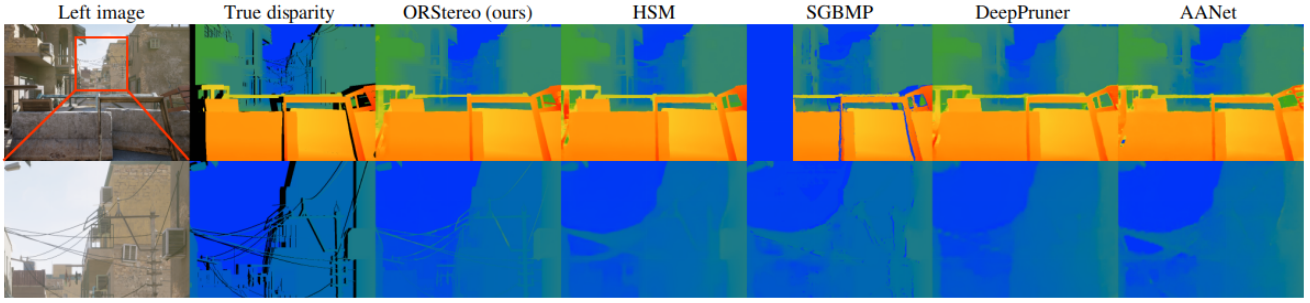


Fig. 7. Results on our 4K synthetic stereo images. The color map is the same as Fig. 8. Columns 3-7 are the disparity estimation of each model. Table IV shows the quantitative statistics over all the 100 evaluation samples. Seeing thin structures is one of the motivation for high-resolution stereo reconstruction. Our model exploits and reconstructs fine-grain details as shown in the zoom-in figures.

the EPE metric. Evaluation is done with all the collected samples. Fig. 8 shows a sample output from ORStereo. In this figure, we can first compare the disparity predictions from the two phases. The EPE of the first phase is 4.10 pixels (Fig. 8 c, up-sampled back to 4K-resolution). ORStereo improves this value to 2.09 pixels after the second phase (Fig. 8 f). As shown in Table IV, although ORStereo is trained on small image size, its EPE is even lower than HSM on these 4K images with a smaller memory consumption. Fig. 7 further provides us with a qualitative comparison. ORStereo has the potential to fully utilize the fine-grained details in high-resolution images and reconstructs thin structures that are better captured by high-resolution images. ORStereo achieves this at a cost of execution time because it needs to refine multiple patches and the RRU iterates several steps for each patch. On average, ORStereo spends about 15s for a single 4K-resolution image.

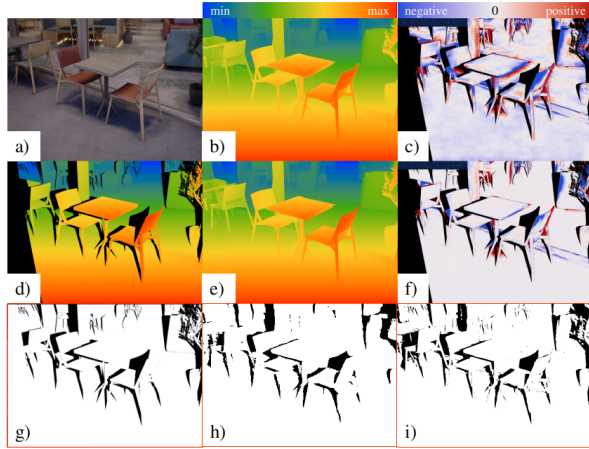


Fig. 8. A 4K sample and ORStereo results. a) 4K Left image. b, c) disparity and error of the first phase. d) ground truth disparity with occlusion. e, f) disparity and error of the second phase. g) true occlusion mask. h, i) occlusion prediction of the first and second phases. c) and f) are masked by the true occlusion from g). The EPE values are 4.10 in c) and 2.09 in f), meaning that the disparity map gets improved in the second phase.

D. Ablation studies

For a better understanding on how various factors affect the performance of ORStereo, we additionally train two models: one that has no explicit occlusion treatments (with all the occlusion update shown in Fig. 4 removed), and

TABLE IV
COMPARISON ON THE SYNTHETIC 4K DATASET.

| Model | Scale | Range | EPE | Mem (MB) |
|-----------------|-------|-------|-------------|-------------|
| AANet[7] | 1/8 | 192 | 9.96 | 8366 |
| DeepPruner[34] | 1/8 | 192 | 8.31 | 4196 |
| SGBMP*[3] | 1 | 256 | 4.21 | 3386 |
| ORStereo (ours) | 1 | 256 | 2.37 | 2059 |
| HSM*[1] | 1/2 | 768 | 2.41 | 3405 |

ORStereo shows the best results among the models trained on small images. There are 100 pairs of stereo images. Scale: the down-sample scale against the original image width. Range: the trained disparity range. EPE: non-occluded EPE. Mem: peak GPU memory (first/subsequent). *SGBMP combines a learning-based and a non-learning-based model, only GPU memory consumption is reported. ▲This model is trained on high-resolution data.

another one with less training data (remove all samples from TartanAir). Furthermore, we test various iteration numbers and disable the RRU or NLR. All the above model variants are evaluated on the same 100 pairs of 4K stereo images used in Table IV. The results are shown in Table V. We first observe that simply making the RRU iterate longer than the training setting (4 steps) improves the overall accuracy. Thus the RRU learns to work like an optimizer, which gradually improve an initial prediction. However, this does not hold with extensively long iterations, *e.g.* model F₂₀. The F_S model shows that monitoring the RRU by the SSIM is not reliable. This may be due to the fact that SSIM fails to distinguish stereo matching in texture-less and repeated texture regions. The model variants (V₁-V₄) regarding to less training data and incomplete model structures all experience a performance drop, meaning that ORStereo does benefit from its special design. Notably, the V₁ model that is trained with less amount of data than the SotA methods in Table IV still achieves the best EPE among the models trained on low-resolution data.

TABLE V
PERFORMANCE COMPARISON BASED ON VARIOUS FACTORS.

| Model | F4 | F10 | F15 | F20 | F _S | V1 | V2 | V3 | V4 |
|-----------|------|-------------|------|------|----------------|------|------|-------|------|
| All data | ✓ | ✓ | ✓ | ✓ | ✓ | - | - | ✓ | ✓ |
| Occlusion | ✓ | ✓ | ✓ | ✓ | ✓ | - | - | ✓ | ✓ |
| RRU Iter. | 4 | 10 | 15 | 20 | SSIM | 10 | 10 | - | 10 |
| NLR | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | - |
| EPE | 2.43 | 2.37 | 2.45 | 2.56 | 2.55 | 3.34 | 2.70 | 11.46 | 3.34 |

F4-F10: Full model with various RRU iteration numbers. F4 is the training setting. F10 is the model used in Table III and IV. F_S utilizes the SSIM to monitor RRU iterations. V1-V4: model variants. V1 is trained without the TartanAir dataset, making the training set have less samples than the other SotA models shown in Table IV.

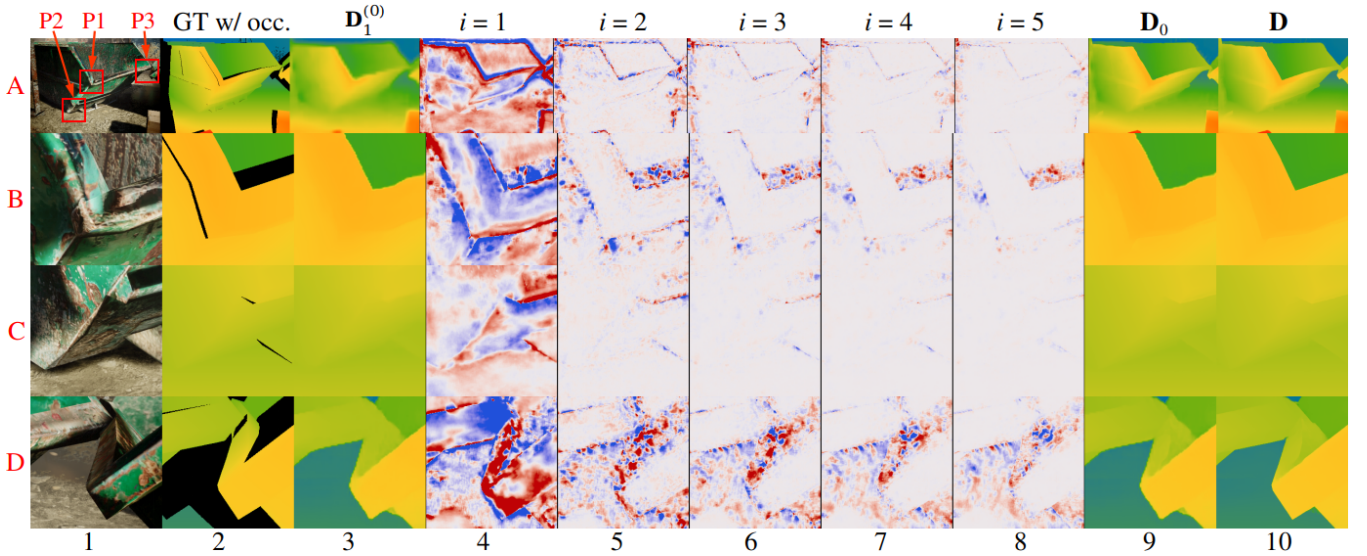


Fig. 9. The two phases of ORStereo. The color maps are the same as Fig. 8. Input: 4K-resolution. Row A: the first phase. 3 patches (512×512, P1-P3) are cropped out for illustrating the second phase (Row B-D). Row B (P1): moderate disparity change and occlusion. Row C (P2): short disparity range, limited occlusion. Row D (P3): large disparity jump, severe occlusion. Column 2, 3, 9, 10 are disparities. Column 4-8 show the first 5 updates made by the RRU. The RRU quickly converges in the non-occluded regions after several iterations and oscillates in the occluded areas. The NLR (Column 10) smooths and sharpens D_0 (Column 9) in both phases. The results also indicate that the NLR can handle unseen disparities beyond the training range.

E. The characteristics of the two-phase procedure

The RRU and NLR are the core components for ORStereo to work across phases. To illustrate the characteristics of the RRU and NLR, in Fig. 9, we show their behaviors in a complete first phase and 3 patches from the second phase. Fig. 9 shows that the RRU delivers stable and convergent updates in the non-occluded areas especially Row D in the figure. In occluded regions, there is hardly any useful information for stereo matching. However, inside a severely occluded patch, ORStereo can use the detected occlusion to guide the residual update, stabilizing the updates in the non-occluded regions.

F. Evaluation on real-world 4K stereo images

Evaluation on real-world high-resolution data is necessary because ORStereo is trained with only synthetic images. We set up a customized stereo camera with two 4K cameras and a LiDAR and capture images around various objects. To evaluate the stereo reconstruction results, we generate dense point clouds as ground truth using a LiDAR-enhanced SfM [39] method. Using point clouds allows us to measure the metric error of the reconstructed model. For each scene, all disparity maps from different viewpoints are converted to point clouds and are registered with the ground truth by the ICP method with an initial pose guess from the SfM results. Both the ground truth and stereo reconstructed point clouds are down-sampled for efficiency. The results are shown in Fig. 10 and Table VI with three different scenes. We use the mean point-to-plane distance ($rmse$) of all the overlapping points between stereo point clouds and ground truth to measure the precision. A point is considered to have a valid overlap if it falls within 0.1m from any ground truth point. Additionally, we use the number of overlapping

points num as a metric to measure the valid overlapped area. Both HSM[1] and ORStereo achieve similar millimeter-level precision. Notably, ORStereo obtains higher num than HSM, which means our model reconstructs more valid areas. ORStereo is trained on small-sized synthetic data and utilizes fewer computation resources but still obtains competitive reconstruction results on real-world 4K stereo images.

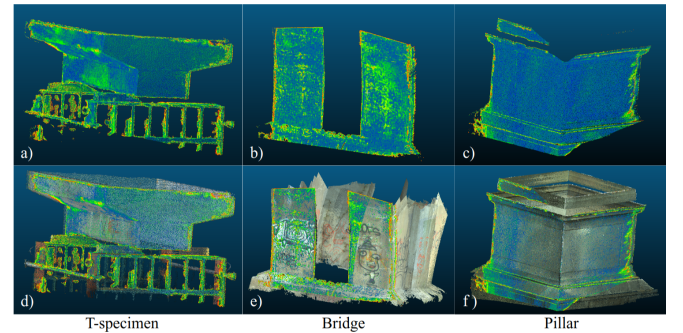


Fig. 10. Reconstruction results aligned with ground truth. Overlapping points are colorized by the point-to-plane distance to the ground truth by blue-green-yellow-red in an increasing manner. a)-c): points from ORStereo. d)-f): the top row with the ground truth.

TABLE VI
STEREO RECONSTRUCTION RESULTS ON REAL-WORLD 4K IMAGES.

| Scene | T-specimen 25 | | Pillar 29 | | Bridge 32 | |
|----------|------------------|-------------|--------------|-------------|--------------|-------------|
| Metrics | $rmse$ | num | $rmse$ | num | $rmse$ | num |
| HSM[1] | 6.56 | 1.30 | 3.94 | 2.43 | 5.82 | 3.50 |
| ORStereo | 6.74 | 1.37 | 3.69 | 2.48 | 5.33 | 3.62 |

Frames: the total number of stereo pairs. $rmse$: point-to-plane distance in millimeters. num : valid overlapping points, unit 10^5 . $rmse$ and num are averaged across all the frames of individual cases. Evaluations are conducted by down-sampling the point clouds by a grid size of 5mm.

V. CONCLUSIONS

We present ORStereo, a model that is trained only on small-sized images but can operate high-resolution stereo images at inference time with limited GPU memory. In testing time, ORStereo refines an initial prediction in a patch-wise manner at full resolution. By jointly predicting the disparity and occlusion, the recurrent residual updater (RRU) can steadily update a disparity patch with severe occlusions. With a special normalization and de-normalization sequence, the normalized local refinement module (NLR) can generalize to unseen large disparity ranges. Our experiments on synthetic and real-world 4K-resolution images validate the effectiveness of ORStereo in both low- and high-resolution stereo reconstruction. ORStereo achieves state-of-the-art performance without any high-resolution training data.

ACKNOWLEDGMENT

This work is supported by Shimizu Corporation. Special thanks to Hayashi Daisuke regarding on-site experiments.

REFERENCES

- [1] G. Yang, J. Manela, M. Happold, and D. Ramanan, "Hierarchical deep stereo matching on high-resolution images," in *CVPR*, 2019.
- [2] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," in *German conference on pattern recognition*. Springer, 2014, pp. 31–42.
- [3] Y. Hu, W. Zhen, and S. Scherer, "Deep-learning assisted high-resolution binocular stereo depth reconstruction," in *ICRA*. IEEE, 2020, pp. 8637–8643.
- [4] "Opencv stereosgbm class reference," https://docs.opencv.org/4.5.0/d2/d85/classcv_1_1StereoSGBM.html, accessed: Nov. 16, 2020.
- [5] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *CVPR*, 2016, pp. 4040–4048.
- [6] G. Yang, X. Song, C. Huang, Z. Deng, J. Shi, and B. Zhou, "Drivingstereo: A large-scale dataset for stereo matching in autonomous driving scenarios," in *CVPR*, 2019.
- [7] H. Xu and J. Zhang, "Aanet: Adaptive aggregation network for efficient stereo matching," in *CVPR*, 2020.
- [8] T. Tani, Y. Matsushita, Y. Sato, and T. Naemura, "Continuous 3d label stereo matching using local expansion moves," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 11, pp. 2725–2739, 2017.
- [9] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry, "End-to-end learning of geometry and context for deep stereo regression," in *ICCV*, Oct 2017.
- [10] J.-R. Chang and Y.-S. Chen, "Pyramid stereo matching network," in *CVPR*, 2018, pp. 5410–5418.
- [11] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "Mvsnet: Depth inference for unstructured multi-view stereo," in *ECCV*, 2018.
- [12] J. Pang, W. Sun, J. S. Ren, C. Yang, and Q. Yan, "Cascade residual learning: A two-stage convolutional neural network for stereo matching," in *ICCV Workshops*, 2017.
- [13] X. Song, X. Zhao, H. Hu, and L. Fang, "Edgestereo: A context integrated residual pyramid network for stereo matching," in *ACCV*. Springer, 2018, pp. 20–35.
- [14] S. Khamis, S. Fanello, C. Rhemann, A. Kowdle, J. Valentin, and S. Izadi, "Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction," in *ECCV*, 2018, pp. 573–590.
- [15] R. Chen, S. Han, J. Xu, and H. Su, "Point-based multi-view stereo network," in *ICCV*, 2019.
- [16] P. L. Dovesi, M. Poggi, L. Andraghetti, M. Martí, H. Kjellström, A. Pieropan, and S. Mattoccia, "Real-time semantic stereo matching," in *ICRA*. IEEE, 2020, pp. 10780–10787.
- [17] Y. Yao, Z. Luo, S. Li, T. Shen, T. Fang, and L. Quan, "Recurrent mvsnet for high-resolution multi-view stereo depth inference," in *CVPR*, 2019.
- [18] J. Liu and S. Ji, "A novel recurrent encoder-decoder structure for large-scale multi-view stereo reconstruction from an open aerial dataset," in *CVPR*, 2020.
- [19] Z. Jie, P. Wang, Y. Ling, B. Zhao, Y. Wei, J. Feng, and W. Liu, "Left-right comparative recurrent model for stereo matching," in *CVPR*, 2018, pp. 3838–3846.
- [20] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," *arXiv preprint arXiv:2003.12039*, 2020.
- [21] Y. Wang, Y. Yang, Z. Yang, L. Zhao, P. Wang, and W. Xu, "Occlusion aware unsupervised learning of optical flow," in *CVPR*, 2018.
- [22] A. Li and Z. Yuan, "Occlusion aware stereo matching via cooperative unsupervised learning," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 197–213.
- [23] E. Ilg, T. Saikia, M. Keuper, and T. Brox, "Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation," in *ECCV*, 2018.
- [24] S. Zhao, Y. Sheng, Y. Dong, E. I.-C. Chang, and Y. Xu, "Maskflownet: Asymmetric feature matching with learnable occlusion mask," in *CVPR*, 2020.
- [25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [26] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *CVPR*, 2018.
- [27] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *CVPR*, 2017.
- [28] G.-Y. Nie, M.-M. Cheng, Y. Liu, Z. Liang, D.-P. Fan, Y. Liu, and Y. Wang, "Multi-level context ultra-aggregation for stereo matching," in *CVPR*, 2019.
- [29] A. Badki, A. Troccoli, K. Kim, J. Kautz, P. Sen, and O. Gallo, "Bi3d: Stereo depth estimation via binary classifications," in *CVPR*, 2020.
- [30] X. Guo, K. Yang, W. Yang, X. Wang, and H. Li, "Group-wise correlation stereo network," in *CVPR*, 2019.
- [31] Q. Wang, S. Shi, S. Zheng, K. Zhao, and X. Chu, "Fadnet: A fast and accurate network for disparity estimation," in *ICRA*. IEEE, 2020, pp. 101–107.
- [32] F. Zhang, V. Prisacariu, R. Yang, and P. H. Torr, "Ga-net: Guided aggregation net for end-to-end stereo matching," in *CVPR*, 2019.
- [33] M. Yang, F. Wu, and W. Li, "Waveletstereo: Learning wavelet coefficients of disparity map in stereo matching," in *CVPR*, 2020.
- [34] S. Duggal, S. Wang, W.-C. Ma, R. Hu, and R. Urtasun, "Deeppruner: Learning efficient stereo matching via differentiable patchmatch," in *ICCV*, 2019.
- [35] Z. Wu, X. Wu, X. Zhang, S. Wang, and L. Ju, "Semantic stereo matching with pyramid cost volumes," in *ICCV*, 2019.
- [36] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," *arXiv preprint arXiv:2003.14338*, 2020.
- [37] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, 2017.
- [38] Z. Bai, Z. Cui, J. A. Rahim, X. Liu, and P. Tan, "Deep facial non-rigid multi-view stereo," in *CVPR*, 2020.
- [39] W. Zhen, Y. Hu, H. Yu, and S. Scherer, "Lidar-enhanced structure-from-motion," in *ICRA*. IEEE, 2020, pp. 6773–6779.