

Pushing the Efficiency of StereoNet: Exploiting Spatial Sparsity

Georgios Zampokas^{1,2}^a, Christos-Savvas Bouganis¹^b and Dimitrios Tzovaras²^c

¹Imperial College London, London, UK

²Information Technologies Institute, Centre for Research & Technology - Hellas, Thessaloniki, Greece
{g.zampokas19, christos-savvas.bouganis}@imperial.ac.uk, dimitrios.tzovaras@iti.gr

Keywords: deep-learning, stereo-matching, sparsity

Abstract: Current CNN-based stereo matching methods have demonstrated superior performance compared to traditional stereo matching methods. However, mapping these algorithms into embedded devices, which exhibit limited compute resources, and achieving high performance is a challenging task due to the high computational complexity of the CNN-based methods. The recently proposed StereoNet network, achieves disparity estimation with reduced complexity, whereas performance does not greatly deteriorate. Towards pushing this performance to complexity trade-off further, we propose an optimization applied to StereoNet that adapts the computations to the input data, steering the computations to the regions of the input that would benefit from the application of the CNN-based stereo matching algorithm, where the rest of the input is processed by a traditional, less computationally demanding method. Key to the proposed methodology is the introduction of a lightweight CNN that predicts the importance of refining a region of the input to the quality of the final disparity map, allowing the system to trade-off computational complexity for disparity error on-demand, enabling the application of these methods to embedded systems with real-time requirements.


1 INTRODUCTION


The aim of stereo matching is the estimation of a disparity map between a rectified image pair; an essential part in applications such as autonomous driving, robotics navigation and aerial surveying. The field is now dominated by powerful deep learning-based approaches, with state-of-the-art accuracy in the popular stereo-matching datasets (KITTI, SceneFlow). However, in most practical real-world applications, latency is a critical factor alongside the quality of the disparity estimation. Until recently, methods mostly focused on the accuracy aspect of stereo-matching, often resulting in large networks, which require devices with significant computational power. However, in many cases, the aforementioned applications are deployed on mobile or embedded devices, such as drones, mobile phones, or autonomous vehicles. Therefore, an important and timely direction of research is towards increasing the computational efficiency of deep-learning methods, allowing to target less powerful hardware and reduced energy consumption.


CNN-based stereo matching methods operate on a dense manner, processing all image locations equally. While this results in increased receptive fields, which produce strong features, it also introduces redundant computations in flat image regions. In specific CNN parts, typically towards the end of the network, such as high resolution upsamplings and refinements in stereo matching, semantic segmentation and super-resolution, image regions have a varying impact on performance. We argue that low-frequency areas often result in minimal gains when processed by those parts, while areas with high-frequency details like edges contribute more on the final result.

An efficient implementation should be able to either allocate less processing time for those areas, or even skip computations on them completely. However, such areas are not known a priori and have to be identified during runtime as they are input dependent. Therefore, the input image needs to be used as a guide to estimate which areas can contribute most towards the final result. This constitutes a data-driven optimization approach, which decides how to efficiently allocate the available computational budget given the input data.

Considering that, this work focuses on pushing the efficiency of StereoNet (Khamis et al., 2018), using

^a <https://orcid.org/0000-0002-3536-5697>

^b <https://orcid.org/0000-0002-4906-4510>

^c <https://orcid.org/0000-0001-6915-6722>

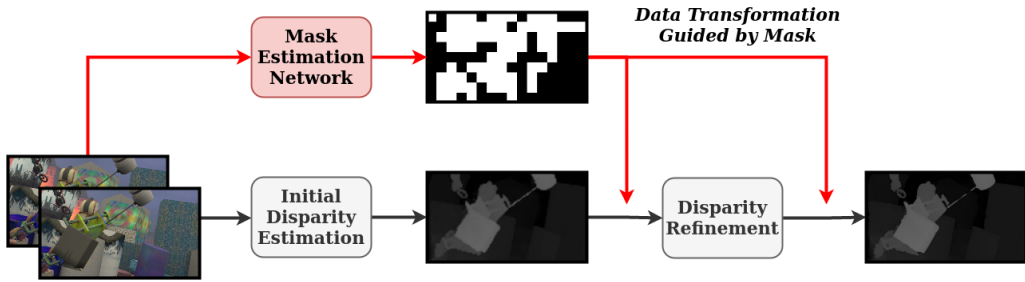


Figure 1: The architecture and flow of StereoNet marked in black color. Our proposed methodology (in red color) guides the the computations to more meaningful regions, according to the output of a parallel Mask Estimation Network.

a block-sparse approach. StereoNet is a lightweight architecture for stereo disparity estimation, which achieves a good trade-off between disparity error and computational complexity, holding a prominent position in the performance/complexity pareto front. Low complexity is achieved by aggressive downsampling of the 4D cost volume, whereas the disparity refinement part -accounting for almost 60% of the total computations-, is responsible for refining high frequency details in the final disparity map.

Given a bound on computational resources, which dictates the necessity of reducing processing in image regions, we aim at the minimization of error in the disparity estimation, considering the input image as a guide. Our key contributions can be summarized as:

1. A block-sparse methodology for increasing the efficiency of StereoNet, by skipping low importance computations.
2. We develop a data driven importance mask estimation network to identify image areas, according to their contribution on final disparity map estimation.
3. A controllable trade-off between disparity error and computational cost, by introducing a sparsity factor, which controls the amount of computational power to be used.

2 RELATED WORK

The work is placed between the field of real-time stereo matching and CNN optimization. Research on real-time stereo has gained a lot of traction lately, due to the importance of execution latency in real-world scenarios.

The first attempt for low latency disparity estimation is the work of (Mayer et al., 2016), where they introduced an architecture based on an encoder-decoder scheme. (Duggal et al., 2019) proposed a method to prune the disparity search space based on the seminal

work of (Barnes et al., 2009). They map the Patch-Match algorithm into a fully differentiable recurrent neural network layer, where each step represents an iteration of the original algorithm. To create a more efficient pipeline, (Yee and Chakrabarti, 2020) replaced the feature extraction layers of stereo matching networks with faster traditional matching costs, to construct the cost volume. Using a multi-branch architecture with depthwise separable convolutions with various dilation rates, (Xing et al., 2020) proposed two efficient architectures. By using multiple 2D correlation layers for speed, along with a multi-scale residual learning technique to tackle vanishing gradients, (Wang et al., 2020) achieve state-of-the-art performance preserving fast inference time. (Wang et al., 2019) proposed an "anytime" approach for disparity estimation, providing disparity estimates at different stages, with increasing accuracy.

The most computationally expensive part of stereo matching CNNs is the filtering of a 4D cost volume, which contains feature correspondences between the left and right images. It mostly relies in 3D convolutions with complexity of $O(K^3 D H W C_{in} C_{out})$, where H, W represent image dimensions, D is the disparity levels of cost volume, K is the kernel size and C_{in}, C_{out} are the number of input and output channels of each convolution layer respectively. (Zhang et al., 2019) substitute the 3D convolutional blocks with local and global aggregation layers, inspired from traditional stereo matching literature on cost aggregation. A popular technique to decrease complexity, is the reduction of the resolution of cost volume. (Khamis et al., 2018) aggressively downsample the input images, constructing a very low resolution volume and then refine the disparity estimate using image-guided filters. Building on top of that, (Aguilera et al., 2020) proposed to replace the heavy 3D convolutional layers with a lighter U-Net like network to filter the cost volume, proving its effectiveness on embedded devices. Towards improving generalization, (Shen et al., 2020) proposed a multi-scale and multi-dimension approach to cost volume construction. They build cost vol-

umes at different scales and refine the estimate using a wrapped 3D volume. Recently, (Tankovich et al., 2020) achieved remarkable accuracy by incorporating image slant to guide geometric wrapping and upsampling, in a multi resolution approach.

There exist multiple directions in optimizing the computational efficiency of a CNN model, either in a static or a dynamic way. Decompositions of convolutional kernels and connection pruning, have already been standardized in successful CNN implementations ((Szegedy et al., 2015), (He et al., 2016)), demonstrating the success of static optimizations. As it is more relevant given our data-driven approach, we focus on dynamic optimization methods. (Figurnov et al., 2016) attempted to tackle the spatial redundancy in convolutional layers, by skipping computations on feature maps guided by various perforation configurations. Later a ResNet based model with a spatially varying computation time was proposed in (Figurnov et al., 2017). Computational blocks and spatial locations within them can be skipped once they do not contribute to performance, according to a halting policy. (Teerapittayanon et al., 2016) exploit the capability of a CNN to predict some samples with high confidence at early layers, proposing an architecture with early exits for samples that can be inferred accurately, avoiding executing the full network for minimal gains. (Shomron et al., 2020) proposed a method to dynamically predict whether CNN output feature maps activations are zero or not according to their neighboring activation values, skipping the zero-valued ones to save computations.

Our method draws inspiration from (Figurnov et al., 2017), on estimating the importance of processing for various image regions. However, instead of early-stopping the execution for the whole image once the network is confident for prediction, we propose to customize the processing of different parts given as input. Also, the proposed method departs from the CNN stereo matching methods, as it proposed a data-driven approach to maximize the quality of the disparity estimation, given a computational budget. Most attempts that focus on spatial sparsity have targeted tasks without dense output such as Image Classification and Human Pose Estimation, thus to our knowledge this is the first attempt to exploit spatial sparsity in an inherently dense output computer vision task.

3 IDEA SKETCHING

Current CNN-based algorithms for disparity estimation perform redundant computation in low-frequency

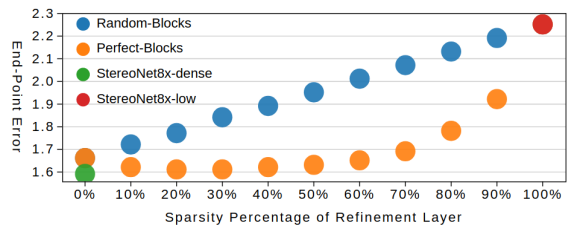


Figure 2: Figure demonstrating the change in EPE with respect to sparsity using block format. Blue dots represent network operation states where image regions are skipped by random selection, while orange dots are states which skip regions in an error-aware manner.

regions, since these regions are going through the same pipeline as edge and texture regions, without significant impact on the disparity estimation. This is also the case in StereoNet and to demonstrate this, we investigate the impact of the computationally heavy refinement part of Stereonet-8x-Single (Figure 1). The full architecture consists of a 2D feature extractor and 3D convolution blocks, which regress a low-resolution disparity map. Then this map is bilinearly upsampled to full resolution and further processed to produce a high-resolution disparity estimate. We train the network for 11 epochs with 192 disparity levels, on the *sceneflow-finalpass* dataset (Mayer et al., 2016), as described in (Khamis et al., 2018), reaching an accuracy of 2.24 pixel End-Point-Error (EPE) for the coarse and 1.59 pixel EPE for the refined disparity estimation over the test set.

On Accuracy. In our analysis, we examine the contribution of refining different image regions to the final result, by skipping the refinement of image areas. We gradually negate the effect of refinement in steps of 10% of total pixels, or sparsity levels. Pixel selection takes place in two ways, in a random and in an impact-aware manner. The latter selects the ones where error reduction after refinement is the largest. The results indicate that, on average, an increase in sparsity, results in error increase (Figure 2). In random selection cases, error and sparsity are linearly correlated, whereas by using the actual refinement error difference as a sparsity mask, the error moves in a more optimal curve, outperforming random selection, defining the potential gain using an oracle.

On computational cost. In the stereo matching literature, inference time is utilized as a metric of performance. Aiming to remove the dependency of the performance from hardware specifications and capabilities and provide a rigid metric, the analysis below focuses on the FLOPs as a description of the computational complexity of the algorithm. Moreover, convolutions over floating point tensors represent the bulk of computations, which further justifies the reference to FLOPs. Similar to the previous

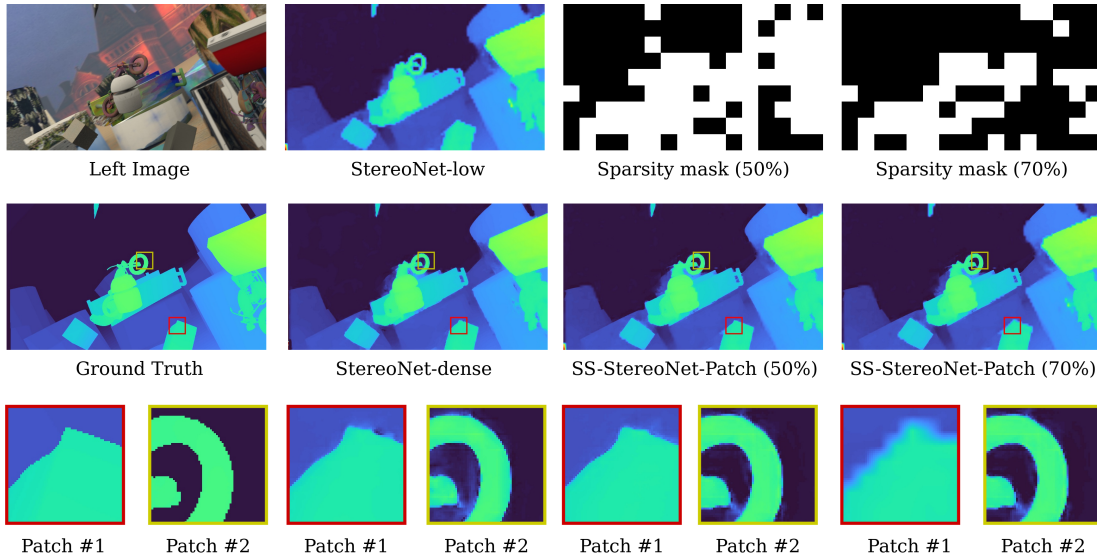


Figure 3: Qualitative evaluation of a sample from scene flow dataset. Sparsity of SS-StereoNet is denoted in parentheses. Third row contains disparity estimate of selected patch samples from the disparity map above them. Note how Patch #2 is refined even at 70% sparsity, whereas refinement of Patch #1 is skipped at higher sparsity.

analysis, we calculate the FLOPs for 10 sparsity levels as $F(s) = F_{\text{Low}} + (1 - s) * F_{\text{Ref}}$, where s denotes the sparsity of tensor entering the refinement part and F_{Low} , F_{Ref} the FLOPs required by the low resolution regression and the refinement part respectively.

Conclusion. Considering the above, a set of operation states exist for this network by adjusting the sparsity of the refinement part as seen in Figure 2. In this work, we attempt to estimate the best performing sparse states and emerge with an improved performance/cost ratio. The above analysis considers an ideal scenario regarding computational cost, since the estimation of a selection mask and the data transformation introduce additional computational overheads. Nevertheless, it provides an ideal performance-cost curve, and an upper bound on the expected performance improvements.

4 Method Overview

Focusing on StereoNet, the proposed Spatially Sparse architecture (SS-StereoNet) extends StereoNet by including a parallel CNN mask estimation network along with the main network. Before executing the disparity refinement part, the input is divided in active and non-active blocks according to the output of mask estimation network. Only active blocks enter the refinement pipeline, whereas the rest are not processed. Finally, after refinement, the output is assembled by combining the refined active blocks together with the

unchanged non-active blocks.

4.1 Block Format Convention

The StereoNet network contains a series of convolutional layers. An important factor for our approach is the receptive field of the processing block. To replicate the performance of the original network, an active pixel would have to carry information from all pixels that belong in its receptive field, resulting in large memory and calculation overhead. Given an input $U [H, W, C]$ entering a set of convolutional blocks, we would ideally like to isolate each element, and decide whether to process it or not. To replicate the base network performance for that input, each element must carry its receptive field r during processing, which would result in $[sHW, 2r + 1, 2r + 1, C]$, where s the targeted sparsity.

Thus, for any tensor entering a set of convolutional blocks with a total receptive field of r , all elements would require $2r + 1$ times the tensor size for individual processing. The memory requirement is also augmented with the use of popular dilated convolutions, which further enlarge the receptive field, resulting in significant memory overheads. In an attempt to reduce the memory required, we propose to handle the data blockwise. This way, the input tensor is divided in blocks and the block element carries the receptive field overhead of border pixels only, resulting in a more efficient implementation with size of $[s, H/b_W + 2r, W/b_H + 2r, C]$, where b_W , b_H denote block dimensions. Additionally, popular Graph-

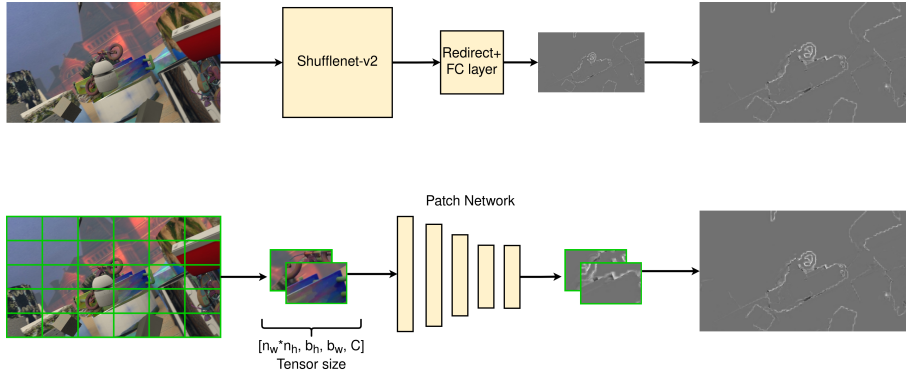


Figure 4: The architectures of the two proposed networks for Mask Estimation: *Full* architecture (top) and *Patch* architecture (bottom).

ical Processing Units (GPUs) are optimized for dense data workloads, which leads to a better performance as more structured the data are.

The approach of conveying full receptive fields, not only increases required memory, but also introduces additional computations. Towards a more optimized approach, we opt to discard the block receptive fields completely and use the block as is extracted from the input tensor. Therefore, both memory and computational overheads are eliminated, at the cost of a reduced receptive field.

4.2 Ground Truth Construction

In order to train the mask estimation network, ground truth mask targets must be extracted. Introducing a pre-training step, a set of *perfect* masks $M_p = \{m_p \in \mathcal{R}^{n_w \times n_h}\}$, where n_w, n_h are the number of blocks in the horizontal and vertical dimension, is calculated for every sample in the dataset and is stored as an additional temporary dataset. The set of masks is extracted for a specific blocksize and represent the mean refinement upside for each image block (x, y) , using results from dense execution of the network,

$$U_{\text{ref}} = |d_{\text{low}} - d_{\text{gt}}| - |d_{\text{dense}} - d_{\text{gt}}|, \quad (1)$$

$$m_p(x, y) = \frac{1}{b_w b_h} \sum_{i=x b_w}^{b_w(x+1)} \sum_{j=y b_h}^{b_h(y+1)} U_{\text{ref}}(i, j), \quad (2)$$

where d_{low} , d_{dense} and d_{gt} represent the disparity from low resolution regression, after refinement and the ground truth. Consequently, the training can be performed in an end-to-end supervised manner, using M_p as ground truth targets.

4.3 Mask Estimation Sub-network

A key component of the proposed approach is the mask estimation network, with the goal of identify-

| Layer | Output size | Kernel | Channels | Stride |
|-------|------------------|---|----------|--------|
| conv1 | 30×30 | 7×7 | 32 | 2 |
| conv2 | 15×15 | 3×3 | 64 | 2 |
| conv3 | 8×8 | 3×3 | 64 | 2 |
| conv4 | 4×4 | 3×3 | 64 | 2 |
| conv5 | 4×4 | 3×3 | 32 | 1 |
| FC | $n_w \times n_h$ | $(32 \times n_w \times n_h) - \text{dim } fc$ | | |

Table 1: Patch Architecture for $[60, 60]$ patch.

ing “important” regions in the feature map. The problem is formulated as a classification task, using binary targets $m_{\text{bin}}(s) \in \{0, 1\}$, where s is the given sparsity factor. For a given s , we sort m_p and select a threshold t , as the value of the n^{th} element of m_p . We construct a binary sparsity mask $m_{\text{bin}}(k)$ by activating “important” tensor locations and zeroing the rest as,

$$m_{\text{bin}}(k) = \begin{cases} 1, & \text{if } m_p(k) > t \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

The Binary Cross Entropy loss is used to supervise training. While binary classification training suppresses the processing upside magnitude, it generates stronger distinction of active and non-active blocks, due to the hard labelling.

The Mask Estimation network is aimed to be computationally lightweight, to minimize the overhead which will be added to the base network. We propose two different architecture approaches with respect to the input image. In the first approach, the left RGB image is used as input to the network (SS-StereoNet-Full). We select a pretrained Shufflenet-v2 (Ma et al., 2018) network to serve as the feature extraction backbone, providing a good starting point for training. An additional convolution is added to the end of the network, which redirects the backbone output to a 4×4 tensor with 32 features, along with a final Fully-Connected layer to aggregate information and produce a $m_{\text{est}} \in \mathcal{R}^2$ of size $[n_w, n_h]$.

| Method | Parameters (M) | FLOPs (G) | EPE (px) | Time (ms) |
|--|----------------|-----------|-------------|---------------|
| PSMNet (Chang and Chen, 2018) | 5.25 | 620.0 | 1.09 | Out of Memory |
| DeepPruner-Best (Duggal et al., 2019) | 7.39 | 368.0 | 0.86 | 2043 |
| HITNet XL (Duggal et al., 2019) | 2.07 | 187.0 | 0.34 | 1679 |
| HITNet L (Duggal et al., 2019) | 0.97 | 78.0 | 0.43 | 795 |
| DeepPruner-Fast (Duggal et al., 2019) | 7.47 | 135.0 | 0.97 | 605 |
| AANet (Xu and Zhang, 2020) | 3.93 | 115.0 | 0.87 | 589 |
| StereoNet-8x-Multi (Khamis et al., 2018) | 0.40 | 131.2 | 1.10 | 221 |
| DispNetC (Mayer et al., 2016) | 38.00 | 75.0 | 1.67 | 191 |
| StereoNet-8x-dense | 0.40 | 113.2 | 1.59 | 187 |
| StereoNet-8x-Perforated-2(L _{5,6}) (Figurnov et al., 2016) | 0.40 | 98.6 | 1.88 | 164 |
| StereoNet-8x-Perforated-2(L ₀₋₆) (Figurnov et al., 2016) | 0.40 | 69.9 | 2.08 | 129 |
| StereoNet-8x-low | 0.28 | 54.4 | 2.26 | 85 |
| AAFS (Chang et al., 2020) | 0.02 | 0.7 | 3.90 | 72 |
| AnyNet (Wang et al., 2019) | 0.04 | 1.4 | 3.30 | 29 |
| SS-StereoNet-Full (50%) | 1.02 | 86.8 | 1.70 | 178 |
| SS-StereoNet-Full (70%) | 1.02 | 75.0 | 1.78 | 151 |
| SS-StereoNet-Patch (50%) | 0.44 | 87.0 | 1.66 | 165 |
| SS-StereoNet-Patch (70%) | 0.44 | 75.2 | 1.78 | 138 |

Table 2: Evaluation on Sceneflow-finalpass dataset. Codes of cited methods are used from their public repositories. Sparsity in our approaches is denoted with the sparsity percentage in parentheses. Execution times are measured on a Jetson AGX under MAXN power profile.

Alternatively, we also propose a network that operates on a regions of the whole input (SS-StereoNet-Patch). Inspired by the patch architecture proposed in (Cao and Zhang, 2017), we propose a lightweight patch classification architecture, presented in Table 1, where each layer contains a series of 2D Convolution, Batch Normalization and Leaky ReLU activation function. Patches of the RGB image corresponding to the spatial locations of U , are provided as input to the mask estimation subnetwork. Therefore, calculations on each patch are executed independently from other image areas. Network diagrams for both architectures are presented in Figure 4.

Since *Full* architecture operates on the whole image, it uses a larger receptive field than *Patch*, thus is able to estimate smoother masks even when the input does not include many high frequency regions. On the other hand, *Patch* is limited on the local receptive field of each block, requiring the presence of multiple high frequency regions in the image, to estimate a reliable mask.

4.4 Accuracy Restoration

With a selection method in place, only the set of important blocks are refined, while skipping the rest. However, the refinement weights of the base network correspond to dense image processing and are not optimized for the new topology. Therefore, an additional accuracy restoration step is added, to tailor the refinement weights to perform better with block format introduced. Training is performed similar to the original StereoNet, by minimizing Huber loss, be-

tween disparity estimates and ground truth, although the dense disparity estimate is produced by the sparse implementation, by activating all blocks.

4.5 Inference Stage

The ultimate goal of our methodology is to eliminate the low impact computations during inference. As mentioned in 4.1, the input tensor U , containing N total blocks, should be divided in $n_1 = (1 - s) \times N$ active and $n_0 = s \times N$ non-active blocks according to the input sparsity factor s . During inference, the mask estimation network provides a continuous output $m_{\text{est}}(x)$, which is sorted and then binarized using Eq. 3, similarly to ground truth. Using the binary mask as a guide, we extract two separate tensors T_1, T_0 from the input tensor U , corresponding to active and non-active blocks. Given $[H, W, C]$ is the size of U , tensors T_1, T_0 have sizes of $[n_1, b_H, b_W, C]$ and $[n_0, b_H, b_W, C]$ respectively. T_1 is then used as input to the processing part, resulting in T'_1 whereas T_0 is left as is. Once the processing is done, both T'_1 and T_0 are combined into the refined U' .

5 EXPERIMENTAL RESULTS

We use the same StereoNet-8x-Single model from the Section 3, which was trained for 11 epochs in the *sceneflow-finalpass* dataset (Mayer et al., 2016). We further finetune the network for 300 more epochs on either KITTI 2012 (Menze et al., 2015) or KITTI 2015 (Menze et al., 2018). For internal evaluation, a

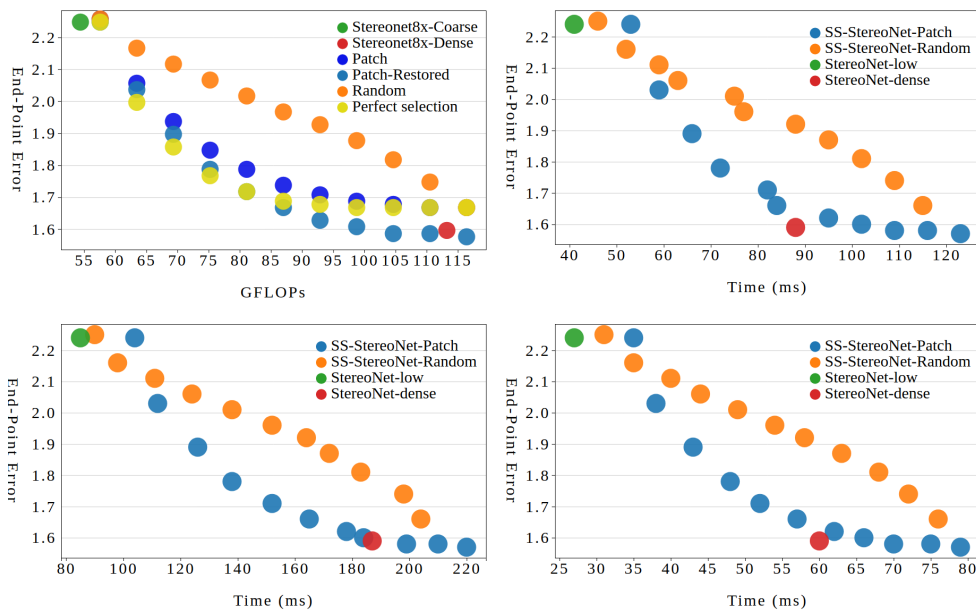


Figure 5: Figure presenting the results of our optimized method in sceneflow dataset. The top-left graph depicts the SS-StereoNet sparse operation states with respect to the theoretical FLOPs. The actual execution time of those operation states is presented, by implementing them on the following devices: Nvidia GTX 1060 (top-right), Jetson AGX under MAXN profile (bottom-left) and Nvidia Titan X Maxwell (bottom-right).

subset of 160 image pairs from the train set are used for training, while the rest of them (34 for KITTI 2012, and 40 for KITTI 2015) are used for testing. The learning rate is set at 0.001 and halves after 200 epochs. This serves as a performance reference point for our experiments.

The Mask Estimation sub-network is also trained on the sceneflow dataset, following the described methodology for 7 epochs, using Adam optimizer, along with 0.001 learning rate and batch size of 64. Similar procedure is followed for KITTI datasets, but with a reduced batch size of 1. We work with block size of 60×60 for all the experiments. Finally, both networks are combined and an additional 3 epochs are required for accuracy restoration, where only the weights of the refinement part are updated. We use the Pytorch framework to implement our networks, while time benchmarks are performed with a Nvidia Jetson AGX embedded GPU device. Perforated layers are denoted as Perforated- $x(L_i)$, where x denotes the subsampling factor and i is the i -th resblock in StereoNet refinement pipeline.

5.1 Discussion

We perform an internal evaluation of the proposed SS-StereoNet on discussed datasets, compared against their ideal and dense/coarse counterparts. Samples from KITTI datasets contain images, which include a

large region of irrelevant information with respect to driving, such as the sky or distant locations. Therefore they are suitable for the proposed optimization, since by identifying such regions and skipping the computations, there is minimal impact in accuracy with reduced FLOPs, as demonstrated in Table 3.

Our SS-StereoNet clearly takes advantage of better area selection than random mask, almost reaching the ideal target (Figure 5). However, with the additional accuracy restoration step, it outperforms the ideal target, pushing the sparse performance closer to StereoNet-8x-dense implementation. Our SS-StereoNet with 50% sparsity performs close to the dense version, with the minimal loss of 4.4% accuracy, while requiring 23% less FLOPs.

An important aspect is the selection of the Mask Estimation Network architecture, between *Full* and *Patch*. The former incorporates larger context resulting in improved mask estimation on data with large textureless areas (KITTI), whereas the latter requires high frequency information on a large number of extracted patches. Therefore, it is deemed more suitable for data with crisp details, like the sceneflow dataset, as compared in Figure 8.

To investigate practical gains, we also benchmark the sparse operation points of SS-StereoNet and their original architectures on three devices with different characteristics, shown in Figure 5. Across all devices, a linear decline of execution time is observed, as spar-

| Sparsity | Architecture | FLOPs (G) | KITTI2012 | | KITTI2015 | |
|----------|-------------------|-----------|-----------|---------|-----------|---------|
| | | | Out-All | Avg-All | Out-All | Avg-All |
| 100% | StereoNet8x-low | 50.00 | 6.67 | 1.29 | 4.35 | 1.14 |
| | StereoNet8x-dense | 104.04 | 4.99 | 1.05 | 2.56 | 0.78 |
| 50% | Perfect mask | - | 5.60 | 1.11 | 4.52 | 1.04 |
| | Random mask | 77.22 | 5.90 | 1.15 | 4.52 | 1.04 |
| | SS-StereoNet-Full | 77.22 | 5.64 | 1.11 | 4.04 | 0.99 |
| 75% | Perfect mask | - | 6.10 | 1.17 | 4.51 | 1.04 |
| | Random mask | 70.45 | 6.25 | 1.19 | 4.90 | 1.08 |
| | SS-StereoNet-Full | 70.45 | 6.08 | 1.17 | 4.57 | 1.04 |

Table 3: Evaluation on KITTI 2012 and KITTI 2015 datasets. The percentage of pixels with error bigger than 3 is reported, along with the overall EPE.

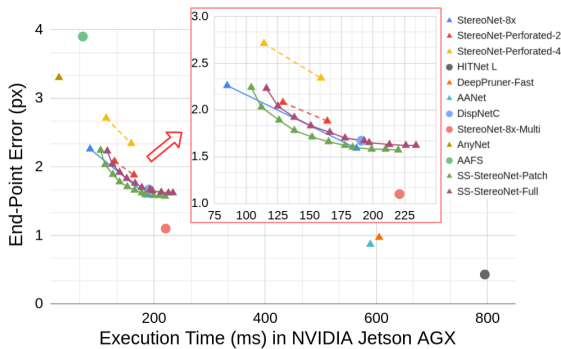


Figure 6: Evaluation on Sceneflow-finalpass dataset. Codes of cited methods are used from their public repositories. Execution times are measured on a Jetson AGX under MAXN power profile.

sity is increased. Actual practical gains, compared to the StereoNet-dense implementation, are achieved at around 50% sparsity for the desktop GPUs, while 20% sparsity is enough in Jetson AGX device, which is close to the ideal implementation. The reason is that the embedded device is kept more busy during the execution, compared to the desktop devices and its runtime is not affected significantly by data transformation overheads.

Compared to the other optimization method, ours outperforms basic Perforated CNN optimization in both original and restored version, as seen in Figure 6, achieving a more efficient application of spatial sparsity. Regarding the stereo matching literature, there exist a set of methods which focus on good accuracy, whereas others provide very low latency approaches. The original StereoNet is already well placed together with the best performing real-time methods. Therefore, our approach creates a competitive pareto front for stereo matching accuracy, which can be adjusted according to the desired FLOPs. Our SS-StereoNet achieves similar performance to DispNetC (Mayer et al., 2016), albeit it has with less parameters and faster execution time. Essentially, we provide an ap-

proach to bridge the gap between lightweight and ultra-lightweight stereo matching methods, under a controlled application of spatial sparsity.

6 CONCLUSIONS

We enhance StereoNet to enable a performance-accuracy trade-off, by skipping image regions, with small contribution to final output. Our method creates a pareto front of possible operation points, controlled by a sparsity factor, which represents the available computational resources for disparity estimation. The proposed optimization methodology is suitable for any part of a CNN which contains convolutions operating at high resolution where computation redundancy is more evident. This methodology is a *non-invasive/vertical* to the network, meaning it can be used on top of a suitable trained network to produce a more efficient version. It can be used in combination with other optimizations, since it does not interfere with the network flow in a global level.

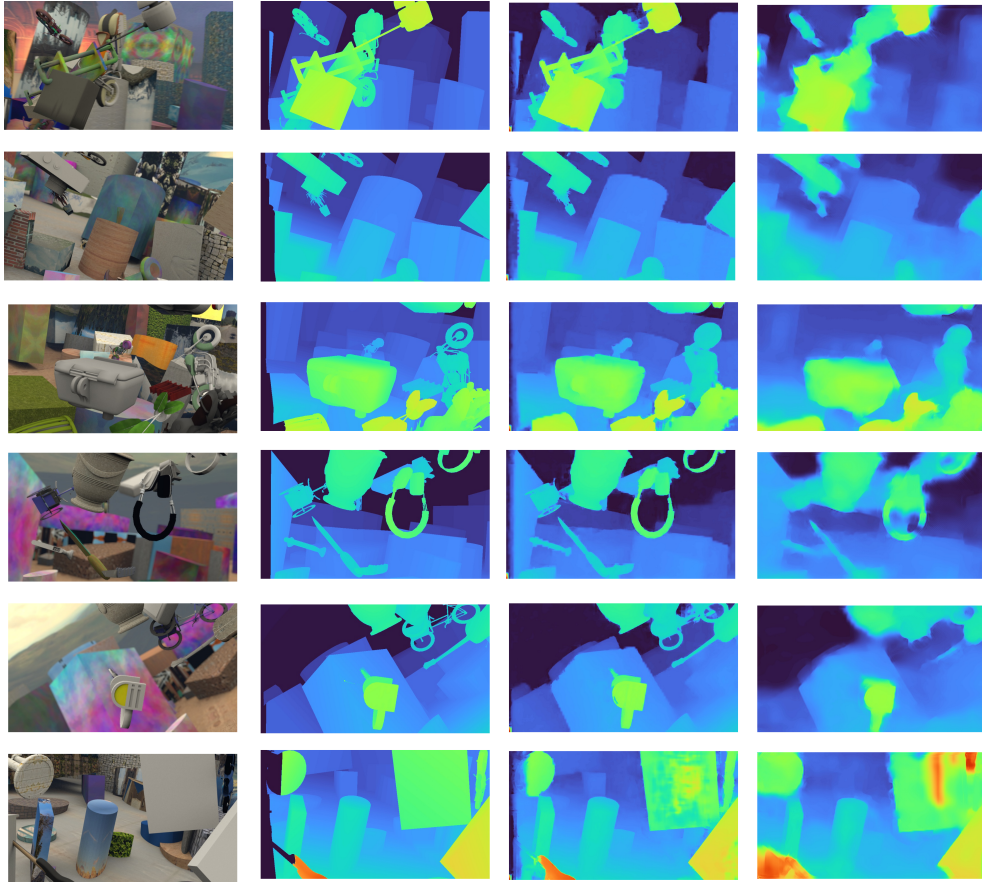


Figure 7: Qualitative evaluation of scene flow dataset samples. Columns from left to right include: RGB input, Ground Truth, SS-StereoNet-Patch at 50% sparsity and disparity estimate from AnyNet (Wang et al., 2019) method.

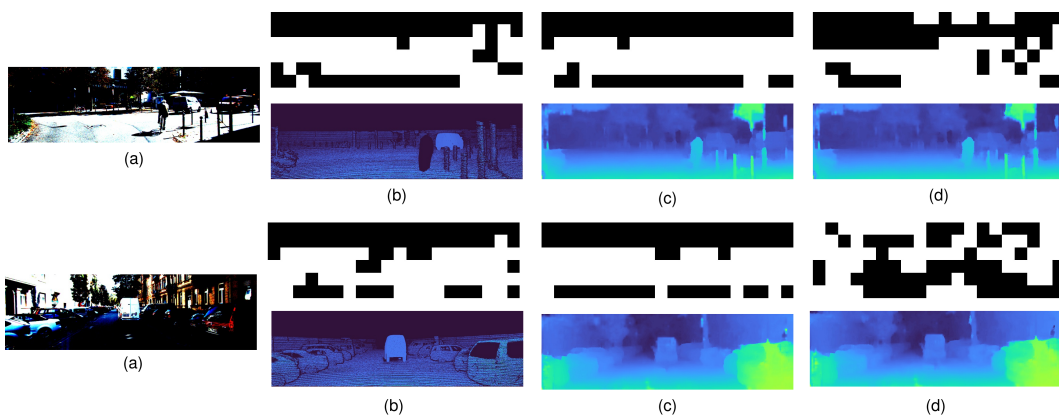


Figure 8: Evaluation on KITTI 2015 samples. Columns contain: input image (a), ground truth (b), SS-StereoNet-Full with 50% sparsity (c), SS-StereoNet-Patch with 50% sparsity (d). Top rows of each sample contain sparsity masks, while bottom rows contain disparity estimates.

REFERENCES

- Aguilera, C. A., Aguilera, C., Navarro, C. A., and Sappa, Á. D. (2020). Fast CNN stereo depth estimation through embedded GPU devices. *Sensors*, 20(11):3249.
- Barnes, C., Shechtman, E., Finkelstein, A., and Goldman, D. B. (2009). PatchMatch: A randomized correspondence algorithm for structural image editing. In *ACM Transactions on Graphics*.
- Cao, Q. and Zhang, H. (2017). Combined holistic and local patches for recovering 6d object pose. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2219–2227.
- Chang, J. and Chen, Y. (2018). Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, abs/1803.08669.
- Chang, J.-R., Chang, P.-C., and Chen, Y.-S. (2020). Attention-aware feature aggregation for real-time stereo matching on edge devices. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*.
- Duggal, S., Wang, S., Ma, W.-C., Hu, R., and Urtasun, R. (2019). Deeppruner: Learning efficient stereo matching via differentiable patchmatch. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.
- Figurnov, M., Collins, M. D., Zhu, Y., Zhang, L., Huang, J., Vetrov, D., and Salakhutdinov, R. (2017). Spatially adaptive computation time for residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Figurnov, M., Ibraimova, A., Vetrov, D. P., and Kohli, P. (2016). Perforatedcnns: Acceleration through elimination of redundant convolutions. In *Advances in Neural Information Processing Systems*, volume 29.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Khamis, S., Fanello, S., Rhemann, C., Kowdle, A., Valentin, J., and Izadi, S. (2018). StereoNet: Guided hierarchical refinement for real-time edge-aware depth prediction. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11219 LNCS:596–613.
- Ma, N., Zhang, X., Zheng, H.-T., and Sun, J. (2018). Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
- Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048.
- Menze, M., Heipke, C., and Geiger, A. (2015). Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*.
- Menze, M., Heipke, C., and Geiger, A. (2018). Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*.
- Shen, Z., Dai, Y., and Rao, Z. (2020). Msmd-net: Deep stereo matching with multi-scale and multi-dimension cost volume. *CoRR*, abs/2006.12797.
- Shomron, G., Banner, R., Shkolnik, M., and Weiser, U. (2020). Thanks for nothing: Predicting zero-valued activations with lightweight convolutional neural networks. In *Computer Vision – ECCV 2020*, pages 234–250.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Tankovich, V., Häne, C., Fanello, S. R., Zhang, Y., Izadi, S., and Bouaziz, S. (2020). Hitnet: Hierarchical iterative tile refinement network for real-time stereo matching. *CoRR*, abs/2007.12140.
- Teerapittayanon, S., McDanel, B., and Kung, H. (2016). Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2464–2469.
- Wang, Q., Shi, S., Zheng, S., Zhao, K., and Chu, X. (2020). Fadnet: A fast and accurate network for disparity estimation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 101–107.
- Wang, Y., Lai, Z., Huang, G., Wang, B. H., van der Maaten, L., Campbell, M., and Weinberger, K. Q. (2019). Anytime stereo image depth estimation on mobile devices. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5893–5900.
- Xing, J., Qi, Z., Dong, J., Cai, J., and Liu, H. (2020). Mabnet: A lightweight stereo network based on multi-branch adjustable bottleneck module. In *Computer Vision – ECCV 2020*, pages 340–356.
- Xu, H. and Zhang, J. (2020). Aanet: Adaptive aggregation network for efficient stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Yee, K. and Chakrabarti, A. (2020). Fast deep stereo with 2d convolutional processing of cost signatures. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.
- Zhang, F., Prisacariu, V., Yang, R., and Torr, P. H. (2019). Ga-net: Guided aggregation net for end-to-end stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.