

Soft Cross Entropy Loss & Bottleneck Tri-Cost Volume For Efficient Stereo Depth Prediction

Appendix: An Overview of Small Simple Training (SST) Net Experiments

Tyler Nuanes^{†,‡}

Matt Elsey[‡]

Aswin Sankaranarayanan[†]

John Shen[†]

[†]Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA

[‡]Light, 725 Shasta St, Redwood City, CA

Contact: tnuanes@andrew.cmu.edu

We detail the Small Simple Net (SST) data pipeline, network implementation, training details, and experimental results in this appendix. First, we describe our data batching and augmentation pipeline; we then go further to describe the splits used from each dataset for training, validation, and testing. Second, we describe the details of Small Simple Training (SST) Network’s implementation. Third, we go over training details, which covers the training and evaluation pipelines. Finally, we describe experimental results and confidence studies.

1. Data Pipeline

1.1. Data Processing and Augmentation

Our data pipeline is implemented in TensorFlow 2.2 with Keras. We encode data in Tensorflow’s TFRecord format, which stores binary sequences of structured data for fast reads. During training, data is interleaved from TFRecords nondeterministically and shuffled with a buffer size of 40 images into a batch size of 4. During validation and testing, data is interleaved deterministically. Due to the large size of the validation set, we limit the validation set to the first 1600 frames read in deterministic interleaved manner. Inference is reported over all images in the inference set. Unless specified, we do not fine-tune on each dataset or separate the results at inference in order to demonstrate the generalization and capacity of each approach.

In order, our data augmentation pipeline performs the following operations during training: random crop to 256×512 ($H \times W$), vertical flip with 50% chance, horizontal flip with 50% chance, offset hue $\pm 5\%$, scale saturation $\pm 5\%$, scale gamma randomly between $[0.9, 1.2]$, scale contrast $\pm 10\%$, and offset brightness $\pm 10\%$. During validation, only a center crop is taken without additional augmentation. During testing on SST Net, only a center crop of size 320×768 is taken.

1.2. Dataset Details

We selected four popular datasets for our evaluation. Flying Things 3D Clean is a 26,760-frame synthetic dataset composed of random shapes and textures placed into a scene [1]. Virtual Kitti 2 is a 17,008-frame synthetic dataset designed to mimic Kitti 2015 [2]. Driving Stereo is a 174,437-frame real-world dataset with densified lidar ground truth [3]. Kitti 2015 is a 200-frame real-world dataset with densified lidar ground truth [4]. From each dataset, data is roughly split into 70/15/15 splits. This is done systematically to keep scenes exclusive to a single split. Kitti 2015 is used exclusively in testing. When running initial experiments, the Driving Stereo test set ground truth was not released, so we used a subset of the training data as our test set.

We used Tensorflow’s TFRecord format to store data prior to training, which stores sequences of binary data for fast reads. For FlyingThings3D (FT3D), TFRecords were generated separately for the *A*, *B*, and *C* splits. For VirtualKitti2 (VK2), TFRecords were generated separately for each scene (Scene01, Scene18, etc.) and its unique conditions, (15_deg_left, overcast, etc.) For DrivingStereo (DS), TFRecords were generated separately for each collection event, differentiated by start date and start time. Kitti 2015 (K15) was placed in a single TFRecord of 200 frames. Each TFRecord holds a maximum of 500 stereo pairs. Additional frames fill subsequent TFRecords. Sequences are recorded to the TFRecords in order. During training, data is interleaved from each TFRecord in the dataset. For training, there are 39 TFRecords for FT3D, 32 for VK2, and 261 for DS.

For FT3D, the TEST split was used as the test set. The validation set was generated from the training set, including the final 960 images from the *A* split, the final 960 images from the *B* split, and the final 970 images from the *C* split. A few frames may be from the same sequences in the vali-

dation and training set.

For VK2, Scene20 was used for validation and testing. The first 500 frames from each sequence (total of 8 sequences) were used for testing. The remaining 337 frames were used for validation. The fog and rain sequences for all scenes were not used in training due to the ambiguous correspondences between images and ground truth disparity.

For DS, the following sequences, identified by (start date; start time) were used for validation: (July 18, 2018 10:16), (July 24, 2018 14:31), (Oct. 10, 2018 07:51), (Oct. 16, 2018 11:43), and (Oct. 23, 2018 08:34). The following sequences were used for testing: (July 31, 2018 11:22), (Aug. 13, 2018 15:32), (Oct. 17, 2018 15:38), (Oct. 24, 2018 11:01), (Oct. 25, 2018 07:37), (Oct. 26, 2018 15:24). The remaining sequences were in the training set. We began our experiments before the DS test set ground truth was released, so we used our own test set in our evaluations.

2. Small Simple Training (SST) Network

When using the bottleneck tri-cost volume, $FE_o = 34$, with 32 features used for SAD and the final 2 reference features concatenated to each disparity step; in this case, $C_{CV} = 3$. SST Net can be seen as a lightweight version of PSM Net [5] and was designed for rapid testing rather than achieving state of the art.

As left-right consistency is a popular regularization technique in unsupervised training [6], our network predicts disparity for both views during training and back-propagates on both with respect to ground truth. When using left-right consistency, the cost aggregation network is Siamese, with shared weights processing the cost volume for each view.

SST Net is designed to be lightweight to allow for a large amount of training at the cost of quality, so metrics reported from SST Net are not comparable to state of the art. SST Net is a simplified version of PSM Net [5] with ResNet-34 style residual blocks [7]. The feature extractor network is a Siamese U-Net with 12 residual blocks truncated on the decoder at 1/4 resolution. The cost aggregator uses GWC Net’s proposed skip connections [8] on a single U-Net with most conv3D replaced by conv2D in order to increase training speed by roughly 40%. Only upsampling and downsampling convolutions are 3D.

Notably, instead of trilinearly upsampling from 1/4 resolution, we only bilinearly upsample the results to $[\frac{1}{4}D, H, W, 1]$. This allows the network to precisely place weights at each disparity step and is important to achieve good results with Soft CCE loss and local MAP.

Feature Extractor Stage			Cost Volume Stage		
res	layer	C_o	res	C_i	C_o
1	conv2D	32	1/4	$2 \times FE_o$	C_{CV}
dwn	conv2D	32			
1/2	res-34(1)	32			
dwn	conv2D	64			
1/4	res-34(1)	64			
dwn	conv2D	128			
1/8	res-34(1)	128			
1/8	res-34(2)	128			
up	conv2D	64			
1/4	concat	128			
1/4	conv2D	FE_o			
ResNet-34(dil) Style			res	layer	C_o
$x = \text{relu}(x_{in})$			1/4	conv2D	32
$x = \text{conv2D}(x, C_o, \text{dil})$			dwn	conv3D	32
$x = \text{relu}(x)$			1/8	conv2D	32
$x = \text{batchnorm}(x)$			dwn	conv3D	64
$x = \text{conv2D}(x, C_o, 1)$			1/16	conv2D	64
$x_{out} = x_{in} + x$			1/16	res-34(2)	64
			1/16	res-34(4)	64
			up	conv3D	32
			1/8	add	32
			1/8	conv2D	32
			up	conv3D	32
			1/4	add	32
			1/4	conv2D	32
			1/4	conv2D	32
			1/4	conv2D	1

Table 1: Small Simple Training Network. res is the resolution, C_i is channels in, C_o is the channels out, and dil is the dilation rate. Nonresidual conv are followed by relu and batchnorm, except for the final conv. Up (up) & down (dwn) sampling kernel size and stride are $[2, 2]$ when 2D and $[2, 2, 2]$ when 3D. Other conv have kernel size $[3, 3]$. Each res-34 in the feature extractor is repeated 3 times.

3. Experimental Details

3.1. Training & Evaluation

SST Net predictions occur for disparity values from 0 px to 300 px, so metrics and losses are calculated only over ground truth within this range. Pixels with disparities outside of this range or without ground truth due to sparse data are excluded from calculations. Metrics and losses are accumulated and scaled by the relative number of unmasked pixels in each image. In other words, each unmasked pixel contributes equally, but each image will not contribute equally. Metrics accumulate all the pixel data and report results after processing all inference frames.

All losses are weighted by the inverse of the maximum masked disparity, 300 px. Cost volume experiments are all

trained with both Huber and Soft CCE losses as these experiments were conducted in parallel with our loss study. All cost volume results are reported with local MAP $\delta = \infty$ (ie, simple expected value). Similarly, all loss experiments are trained with a 96-channel cost volume generated by concatenation and per-channel multiplication.

We report EPE (px), D1 (%), and D $\frac{1}{2}$ (%). For select experiments, we additionally report the 95% confidence intervals, calculated using the Student T distribution.

Training each model took place on a single 32 GB NVIDIA GPU. Each model is trained to 85 epochs and the best validation weights are used to evaluate the test set. Each epoch has 2000 batches (not the full training set), and batch size is 4.

3.2. Parameter Sweep Studies

Our parameter sweep studies support the following three hypotheses:

- H1. Disparity networks benefit from more robust losses, which are less sensitive to challenging outliers.
- H2. Regression losses, which have one-to-many solutions in probability distribution space, learn sub-optimal probability distributions.
- H3. Stereo-based dense disparity prediction benefits from purely monocular cues, likely in occluded or low-texture regions.

3.2.1 Loss Functions

We evaluate SST Net with the losses in Table 2 on our test set with local MAP $\delta = \infty, 3, 2, 1, \frac{1}{2}$. All experiments are trained with concatenation and multiplication cost volume (96 channels). We find that all networks show minor improvements, particularly in D1 and D $\frac{1}{2}$. CCE-trained networks improved substantially more with narrower δ compared to networks trained with regression losses. Soft CCE outperforms regression losses by at least 10% in D1, supporting our claim that stereo-based depth prediction networks should learn probability distributions directly. Based on the performance gain in D $\frac{1}{2}$, $\delta = 1$ seems to be the best choice to running local MAP on the Soft CCE network.

Notably, Gaussian, Hard, and Soft CCE have comparable D1 performance. However the D $\frac{1}{2}$ metric reveals that Soft CCE outperforms all other CCE losses by close to a factor of two. It also makes 15% improvements to EPE and 20% improvement in D1 and D $\frac{1}{2}$ over the best regression loss, ℓ_1 . Notably, despite the wide popularity of Huber losses in training stereo disparity networks, we demonstrate that ℓ_1 consistently outperforms Huber, particularly in the D $\frac{1}{2}$ metric, where ℓ_1 performs over 15% better. In addition, ℓ_2 , which most heavily penalizes large errors, shows the worst results. These experiments support our hypothesis that disparity networks learn better under more robust

Loss	∞	3	2	1	$\frac{1}{2}$
EPE					
MSE ₁	2.78	2.72	2.72	3.02	3.02
Huber ₅	1.39	1.37	1.37	1.53	1.54
MAE ₃	<u>1.29</u>	1.29	1.29	1.29	1.31
Laplacian ₁	2.55	1.70	1.62	1.58	1.62
Hard ₁	1.51	1.46	1.46	1.47	1.47
Gaussian ₃	1.32	1.21	1.21	1.25	1.40
Soft ₅	<u>1.24</u>	1.12	1.12	1.12	1.13
Soft & Huber ₁	<u>1.26</u>	1.25	1.25	1.26	1.26
Soft & MAE ₁	1.23	1.22	1.22	1.23	1.23
D1					
MSE ₁	19.9	18.5	18.5	23.2	23.2
Huber ₅	5.5	5.0	5.0	6.3	6.4
MAE ₃	5.2	4.8	4.8	4.8	4.9
Laplacian ₁	23.7	12.3	5.4	5.3	5.5
Hard ₁	<u>5.0</u>	<u>4.0</u>	<u>3.9</u>	<u>4.0</u>	<u>4.0</u>
Gaussian ₃	5.3	<u>4.0</u>	<u>3.9</u>	<u>3.9</u>	<u>4.0</u>
Soft ₅	5.4	3.8	3.8	3.8	3.8
Soft & Huber ₁	<u>5.0</u>	4.5	4.5	4.5	4.5
Soft & MAE ₁	4.9	4.4	4.4	4.4	4.4
D $\frac{1}{2}$					
MSE ₁	47.6	46.6	46.6	55.1	55.1
Huber ₅	18.2	17.3	17.2	22.6	22.8
MAE ₃	15.0	14.4	14.3	14.3	14.7
Laplacian ₁	46.8	30.8	30.9	33.0	34.2
Hard ₁	32.0	31.5	31.4	31.4	31.4
Gaussian ₃	22.4	20.8	20.6	21.0	27.5
Soft ₅	13.7	11.3	11.2	10.9	11.0
Soft & Huber ₁	<u>14.4</u>	13.7	13.6	13.5	13.5
Soft & MAE ₁	<u>14.3</u>	13.6	13.5	13.4	13.5

Table 2: Lightweight SST Net inference results under local MAP at different δ . MAE is ℓ_1 and MSE is ℓ_2 . Subscripts indicate the number of times each experiment was run & averaged. There was an outlier in one Huber run for $\delta = \{1, \frac{1}{2}\}$; excluding this outlier, performance would be similar to $\delta = 2$. Minimum values per column are bold. Values within 5% of the minimum of each column are underlined.

losses. Combining losses with Soft CCE reduces the quality of the learned distribution, as local MAP offers only minor improvements to a network trained by Soft & Huber or Soft & MAE.

To demonstrate that these improvements apply generally to both synthetic and LIDAR-based data, we break out inference performance on each dataset in Table 3 for the Huber, MAE, Gaussian, and Soft losses. No finetuning was performed for these results. We consistently find 10% - 40% improvement in the D1 and D $\frac{1}{2}$ metrics for Soft CCE over Huber loss. Notably, large improvements are not restricted

Loss	FT3D Clean			FT3D Final			Virtual Kitti 2			Driving Stereo			Kitti 2015 Train		
	EPE	D1	D $\frac{1}{2}$	EPE	D1	D $\frac{1}{2}$	EPE	D1	D $\frac{1}{2}$	EPE	D1	D $\frac{1}{2}$	EPE	D1	D $\frac{1}{2}$
Huber	2.57	9.9	15.0	2.71	10.4	15.8	1.09	5.1	18.1	0.73	1.6	15.1	0.97	4.6	18.4
MAE	2.50	9.9	14.4	2.60	10.3	15.1	0.95	4.9	15.1	0.69	1.5	13.5	0.97	4.4	18.3
Gaussian	2.13	8.2	16.7	2.32	8.8	17.8	1.20	4.0	41.0	0.62	1.0	12.1	0.87	3.7	16.7
Soft	2.18	8.2	11.1	2.37	8.8	12.2	0.80	3.6	12.1	0.60	1.0	10.0	0.83	3.6	15.4

Table 3: Lightweight SST Net inference results under local MAP $\delta = 2$ on test sets. Model weights with best D1 metric were selected. There was no finetuning. The best metric on each dataset is bolded. FT3D is FlyingThings3D. MAE is ℓ_1 .

to either synthetic or LIDAR datasets.

3.2.2 Cost Volumes

We investigated distance metrics, multi-cost volumes, and compact representations of the cost volume in SST Net and document results in Table 4. All experiments are carried out with Huber & Soft CCE loss. We do not consider the number of features fed into the cost volume, but an optimal number likely depends on network architecture. Our experiments demonstrate:

1. AD tends to outperform other distance functions.
2. Single costs, such as SAD, degrade from per-channel metrics, such as AD, by 6% – 20%.
3. Adding two reference features to SAD prevents most degradation, showing monocular cues benefit prediction.
4. SAD with two reference features is comparable to a 32-channel AD or 64-channel concatenation.
5. Up to four groups of SAD with two reference features, performance does not significantly improve.
6. Grouping four different costs and concatenating 32 reference features improves performance by 3% – 6%.

SAD plus two reference features provides a baseline of performance that is difficult to meaningfully surmount. Alone, single-metric SAD performance is limited, but adding monocular cues improves metrics, demonstrating their usefulness for depth prediction. However, it appears there is some promise in providing multiple costs with different distance functions and many reference features, as Multi Groups & CRF performs the best by about 6% in D $\frac{1}{2}$.

3.3. Experimental Confidence

In the cost-volume study, the concatenation network is run 5 times to determine standard deviation for metrics in Table 5. In the loss study, the Huber loss was run 5 times for the same purpose in Table 5. Several additional losses were run multiple times to determine 95% confidence intervals, as listed in in Tables 6, 7, 8, & 9. Notably, there was an outlier in the Huber local MAP at $\delta = \{1, \frac{1}{2}\}$, which demonstrates a suboptimal learned distribution, being more accurate under the $\delta = \infty$ expected value.

Architecture	GFLOP	C_{CV}	EPE	D1	D $\frac{1}{2}$
Concat	636	64	1.30	5.3	15.8
AD	571	32	<u>1.29</u>	5.3	15.4
Mul	571	32	1.46	5.8	16.7
Var	571	32	1.36	6.0	16.3
EUD	509	1	1.55	6.8	19.1
SAD	509	1	1.44	6.1	17.4
Corr	509	1	1.55	6.1	17.0
Sum Var	509	1	1.46	6.1	17.8
Concat & AD	700	96	<u>1.25</u>	<u>5.1</u>	<u>14.7</u>
Concat & Mul	700	96	<u>1.26</u>	<u>5.0</u>	14.4
Concat & Var	701	96	<u>1.25</u>	<u>5.0</u>	<u>14.5</u>
SAD & Corr	<u>511</u>	2	1.42	5.7	15.9
AD & Mul	636	64	<u>1.25</u>	5.2	<u>14.7</u>
AD & Mul & Var	701	96	<u>1.29</u>	5.3	15.2
8 SAD Groups	523	8	1.32	5.7	16.2
4 SAD Groups	<u>515</u>	4	1.33	5.4	15.8
4 SAD Groups + 2 RF	<u>519</u>	6	<u>1.25</u>	5.2	15.4
Multi & CRF	580	36	<u>1.27</u>	<u>5.1</u>	<u>15.1</u>
Multi Groups & CRF	579	36	1.23	4.9	14.4
Multi Groups + 2 RF	<u>519</u>	6	<u>1.29</u>	<u>5.1</u>	15.2
SAD	509	1	1.44	6.1	17.4
SAD + 1 RF	<u>511</u>	2	1.36	5.5	16.2
SAD + 2 RF	<u>513</u>	3	<u>1.27</u>	<u>5.1</u>	15.3
SAD + 4 RF	<u>518</u>	5	<u>1.28</u>	<u>5.1</u>	15.2
SAD & CRF	573	33	<u>1.28</u>	5.2	<u>15.1</u>

Table 4: Lightweight SST Net inference on cost volume experiments. Concat is concatenation. RF are separate, additional reference features. CRF is concatenation of all 32 reference features. Per-channel costs: AD is absolute difference, Mul is multiplication, and Var is variance. Single costs: SAD is the sum of AD, EUD is euclidean distance, Corr is correlation, and SumVar is the sum of Var. Groups signifies grouping over input channels. Multi includes SAD, EUD, Corr, and SumVar. & means the same 32 features are used. Standard deviation is approximately 0.03 px for EPE, 0.2% for D1, & 0.5% for D $\frac{1}{2}$, based on running Concat 5 times. Minimum values are bold. Values within 5% of the minimum are underlined.

Experiment	Metric	EPE	D1	D½
Cost – CC	Mean	1.30	5.3	15.6
	Std Dev	0.03	0.2	0.5
	95% Confidence	± 0.04	± 0.2	± 0.5
Loss – Huber	Mean	1.39	5.5	18.2
	Std Dev	0.05	0.1	2.6
	95% Confidence	± 0.05	± 0.2	± 2.9

Table 5: Experimental Confidence. Intervals generated by Student T distribution for 5 experiments. CC is concatenation

δ	EPE	D1	D½
∞	1.39 ± 0.05	5.5 ± 0.17	18.2 ± 2.9
3	1.37 ± 0.05	5.0 ± 0.12	17.3 ± 3.0
2	1.37 ± 0.05	5.0 ± 0.13	17.2 ± 3.1
1	1.53 ± 0.44	6.3 ± 3.47	22.6 ± 17.1
$\frac{1}{2}$	1.54 ± 0.43	6.4 ± 3.45	22.8 ± 17.0

Table 6: Huber₅

One experiment was an outlier with EPE = 2.21, D1 = 11.7, and D½= 49.2 for $\delta = \{1, \frac{1}{2}\}$.

δ	EPE	D1	D½
∞	1.29 ± 0.04	5.2 ± 0.13	15.0 ± 0.5
3	1.29 ± 0.03	4.79 ± 0.05	14.4 ± 0.4
2	1.29 ± 0.03	4.77 ± 0.04	14.3 ± 0.4
1	1.29 ± 0.03	4.82 ± 0.03	14.3 ± 0.3
$\frac{1}{2}$	1.31 ± 0.03	4.86 ± 0.03	14.7 ± 0.3

Table 7: MAE₃

δ	EPE	D1	D½
∞	1.32 ± 0.05	5.3 ± 0.25	22.4 ± 1.2
3	1.21 ± 0.02	4.0 ± 0.10	20.8 ± 0.6
2	1.21 ± 0.02	3.9 ± 0.10	20.6 ± 0.6
1	1.25 ± 0.02	3.9 ± 0.08	21.0 ± 0.9
$\frac{1}{2}$	1.40 ± 0.02	4.0 ± 0.07	27.5 ± 0.9

Table 8: Gaussian₃

δ	EPE	D1	D½
∞	1.24 ± 0.02	5.40 ± 0.14	13.7 ± 0.3
3	1.12 ± 0.02	3.84 ± 0.06	11.3 ± 0.1
2	1.12 ± 0.02	3.79 ± 0.05	11.2 ± 0.1
1	1.12 ± 0.02	3.81 ± 0.06	10.9 ± 0.1
$\frac{1}{2}$	1.12 ± 0.02	3.83 ± 0.05	11.0 ± 0.1

Table 9: Soft₅

large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. *CoRR*, abs/1512.02134, 2015.

- [2] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual KITTI 2. *CoRR*, abs/2001.10773, 2020.
- [3] Guorun Yang, Xiao Song, Chaoqin Huang, Zhidong Deng, Jianping Shi, and Bolei Zhou. Drivingstereo: A large-scale dataset for stereo matching in autonomous driving scenarios. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [4] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018.
- [5] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. *CoRR*, abs/1803.08669, 2018.
- [6] Clément Godard, Oisín Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *CoRR*, abs/1609.03677, 2016.
- [7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.
- [8] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. *CoRR*, abs/1903.04025, 2019.

References

- [1] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A