Analysis & Computational Complexity Reduction of Monocular and Stereo Depth Estimation Techniques

Rajeev Patwari *
rpatwari@stanford.edu

Varo Ly *
varoly@stanford.edu

Abstract

Accurate depth estimation with lowest compute and energy cost is a crucial requirement for unmanned and battery operated autonomous systems. Robotic applications require real time depth estimation for navigation and decision making under rapidly changing 3D surroundings. A high accuracy algorithm may provide the best depth estimation but may consume tremendous compute and energy resources. A general trade-off is to choose less accurate methods for initial depth estimate and a more accurate yet compute intensive method when needed. Previous work has shown this trade-off can be improved by developing a state-of-the-art method [12] to improve stereo depth estimation.

We studied both the monocular and stereo vision depth estimation methods and investigated methods to reduce computational complexity of these methods. This was our baseline. Consequently, our experiments show reduction of monocular depth estimation model size by ∼75% reduces accuracy by less than 2% (SSIM metric). Our experiments with the novel stereo vision method [12] show that accuracy of depth estimation does not degrade more than 3% (three pixel error metric) in spite of reduction in model size by 20%. We have shown that smaller models can indeed perform competitively. The source code is available at https://github.com/rajeevpatwari/AnyNet and https://github.com/rajeevpatwari/Monocular-Depth

1. Introduction

The current state of the art autonomous driving and real time navigation decision making tasks require visual scene understanding as an addendum to radio or other sensing techniques. Depth estimation in computer vision and robotics is a necessary and crucial step. This is done today using stereo vision depth techniques [5], monocular depth techniques [14], and hybrid approaches [2].

*The authors contributed equally

2. Background and Related Work

In our project, we implemented monocular and stereo depth estimation using supervised machine learning techniques. Our motivation for the models chosen were to choose models that could provide adequate accuracy while also allowing less computation which would enable depth estimation on lower latency and lower power systems in real world applications.

2.1. Monocular Depth

Monocular depth estimation from a single image can be accomplished using classical supervised learning autoencoder techniques as in [3], [14], and [10]. The network in [14] was developed to provide both accuracy and low latency. Improvements to the speed and computational complexity of the these models were accomplished by utlizing an efficient pretrained encoder such as MobileNet and pruning the network using NetAdapt. In our studies we utlized a basic CNN for both the enoder and decoder similar to [3].

2.2. Stereo Depth

In stereo depth estimation using supervised machine learning techniques, images from two cameras and ground truth disparity data are used to train a neural network to estimate the disparity. In [5] many state of the art methods were reviewed. The process of stereo depth estimation using these techniques generally consisted of feature extraction from both images, feature matching between the images, disparity estimation, and final refinement of the disparity map.

3. Technical Approach

3.1. Monocular Depth

3.1.1 Monocular Depth Architecture

For monocular depth estimation, we reproduced a model based on [3]. This consisted of creating a neural network as shown in Fig 1 consisting of four downscaling blocks, a bottleneck block, and 4 upscaling blocks.

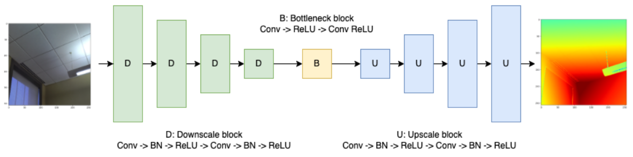
B: Bottleneck block
Conv -> ReLU -> Conv ReLU
D
D
D
D
B
U
U
U
U
D: Downscale block
Conv -> BN -> ReLU -> Conv -> BN -> ReLU
U: Upscale block
Conv -> BN -> ReLU -> Conv -> BN -> ReLU

Figure 1. Monocular Depth Estimation Model

3.1.2 Monocular Evaluation Metric

We used three loss functions in our monocular depth estimation technique. The following equation shows L1 loss used for model optimization. $L1_{loss} = \sum_{i=1}^{n} |d_{i,true} - d_{i,pred}|$ In addition to L1 loss, we also used depth smoothness and SSIM [13] per Tensorflow. The model training involved minimizing all three losses.

We evaluated the accuracy of our generated depth maps by comparing them with the target depth maps using the structural similarity index metric as implemented in our model `https://github.com/rajeevpatwari/Monocular-Depth/blob/main/depth_keras.py#L593`.

3.2. Stereo

3.2.1 Stereo Model Architecture

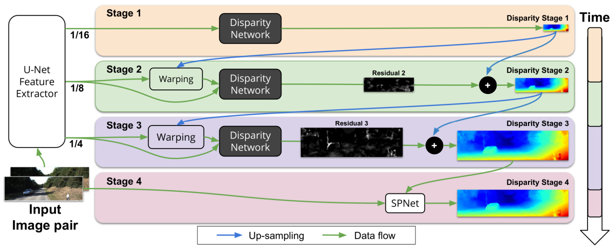

Figure 2. AnyNet [12] Model

Based on our review of [5], we chose to reproduce a stereo vision technique using the AnyNet [12] model. We chose this model as it was fast, accurate, and one of the computationally less intensive models. Figure 2 depicts the architecture of the model which consists of the typical components as previously discussed, except they are combined over 4 different stages.

The U-Net feature extractor is an approximately 20 layer CNN that extracts the features from both the left and right images at various scales (1/16, 1/8, and 1/4) of the images. The warping block matches the features between the left and right images. The disparity network CNN for each stage is approximately 7 layers. For the first stage since the image scale is so small, there is no feature matching before the disparity network CNN. The refinement component is accomplished by an additional CNN called the spatial propagation network, SPNet, which is applied to the predicted disparity map following stage 3 to further improve the results. As part of our studies, we varied the size and complexity of SPNet and assessed it's impact on the accuracy of the results.

3.2.2 Stereo Evaluation Metric

The smooth L1 loss was during training for stereo depth estimation. The smooth L1 loss corresponds to the L2 loss when the difference between the outputs is below a specified parameter and corresponds to the L1 loss when it is above this chosen value. This can be tuned as a hyper parameter, β.

$$L1_{smooth\_loss} =$$

$$\sum_{i=1}^{n} \begin{cases} \frac{1}{2\beta}(d_{i,true} - d_{i,pred})^2 & \text{if } |d_{i,true} - d_{i,pred}| < \beta \\ |d_{i,true} - d_{i,pred}| - .05\beta & \text{otherwise} \end{cases}$$

To assess our stereo model results, we utilized a three pixel error criteria as described in [12] and supported code `https://gist.github.com/MiaoDX/8d5f49c2ccb39d7f2cb8d4e57c3ab752`.

4. Experiments and Results

4.1. Dataset

4.1.1 Monocular Dataset

KITTI [4], NYU-V2 [7], DIODE [11] are three datasets available for monocular depth estimation, but all three datasets are approximately 100 GB. To reduce training time, we have used a validation set from the DIODE dataset which comprises of 325 indoor depth maps and 446 outdoor depth maps. And each data sample comprises of RGB image, depth map, and depth validity mask. Each image is 1024x768, however to reduce the training time we reduced the image size to 256x256 and consequently clipped the depth map as well that is provided as a label to the network. Using this technique each epoch takes approximately 10 minutes to train on the GTX1070 and V100.

In Fig 3, 3 samples are shown from DIODE dataset's test set. The dataset was split between training and validation sets. Validation set was used to evaluate the model's performance on unseen data.

Fig 4 shows a point cloud representation of an input image using its corresponding depth map.

4.1.2 Stereo Dataset

KITTI [4], NYU-V2 [7], Scene Flow [6] are available datasets for stereo depth estimation. Each dataset is on the order of 100GB in size. We picked Monkaa, a subset of the

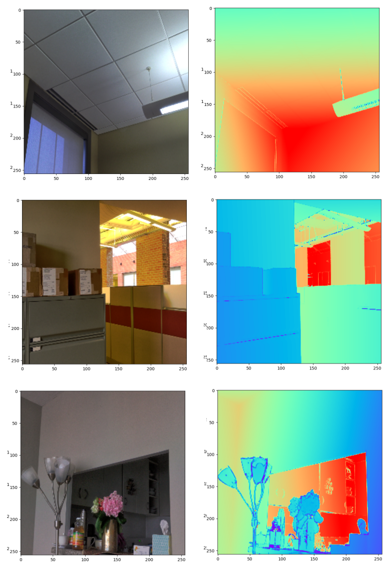
Figure 3. DIODE dataset. Left:input and Right:Depth map

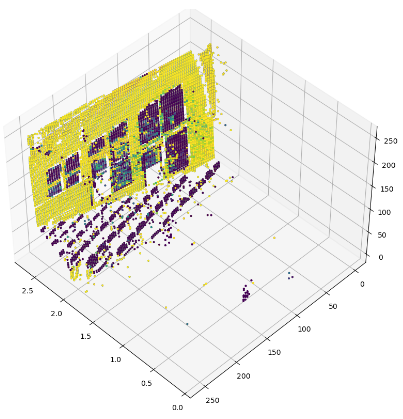
Figure 4. 3D point cloud representation of input

Scene Flow dataset, as the target dataset for our study allowing us enough time to perform multiple ablation studies and model architecture evaluation. It contains 9000 synthetic data samples and corresponding disparity maps. We split the data into train and test sets with a 90-10 split. The figure 5 shows reference example of a single data sample from the dataset.

4.2. Baseline

Our baseline is to reproduce the existing results as a first step. In the subsections below, we describe methods used to reproduce both Monocular and Stereo Vision based depth estimation techniques. We also show some ablation studies done to improve accuracy of predictions.

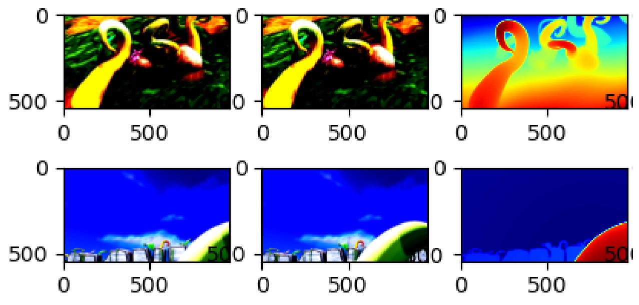
Figure 5. Monkaa Stereo dataset. Left, Right input images and Left Disparity Map

4.2.1 Monocular Depth

We reproduced the chosen monocular depth estimation model and evaluated different types of loss functions. Monocular depth estimation was built using Tensorflow and Keras framework [1]. The reference publication describes using structural similarity (SSIM), depth smoothness and L1 loss as three metrics. We performed training and hyper-parameter search for combination of these loss functions. Fig 6 and Fig 7 below show the convergence over time. We also varied batch sizes from 16, 32, 64.

We used SSIM as an evaluation metric for measuring similarity between target and predicted data. Figure 12 shows results.

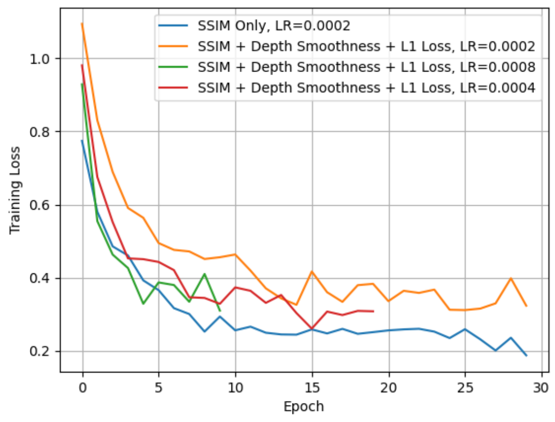

Figure 6. Monocular Depth - Train Loss

We observed that using SSIM as only loss metric resulted in better model convergence, as visualized by training and validation loss curves in figures 6 and 7. However, visually, the results were not on part with using all three losses in optimization. We found this observation to align with our expectation. SSIM is generally lenient on errors in nearest neighbors, however, adding depth smoothness and L1 depth loss would provide a tighter correspondence between targets and predictions.

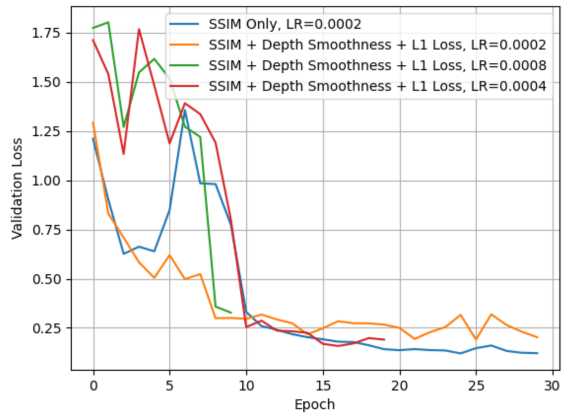

Figure 7. Monocular Depth - Validation Loss

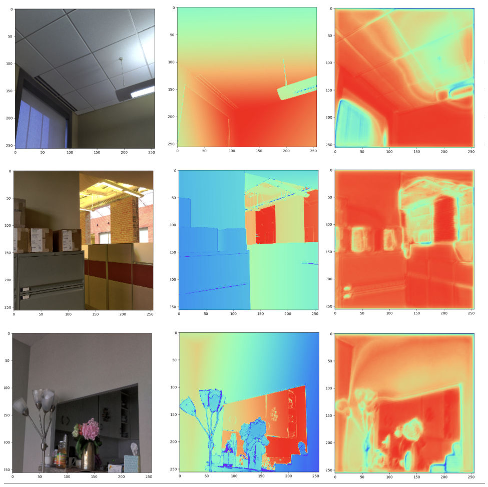

Figure 8. Monocular Depth - Input, Target, Predicted

4.2.2 Stereo Depth - AnyNet

We reproduced the AnyNet learning based stereo vision depth estimation model using Pytorch [8]. Training process is computationally intensive and took 4 hours on GTX1070 and 30min on V100 GPUs. We were also able to use a larger dataset on this model compared to the Monocular depth model. The dataset was split 90%-10% for training and test sets randomly. Figure 9 shows training convergence of 2 variations of the AnyNet model, with and without Spatial Propagation Network (SPNet). The publication [12] describes that the model performs better with SPNet. Figures 10 and 9 shows losses at various stages, described in the model shown in figure 2. The idea is that the user would be able to truncate the model at stage 1, 2, 3 with a slight loss in accuracy but a considerably smaller model size.

The publication [12] uses three pixel error for measuring disparity differences between target and predicted disparities. We have implemented the same metric to evaluate accuracy of the model.

Fig 11 shows predicted disparity map. Three pixel error accuracy is shown in the figure 16 and table 3.

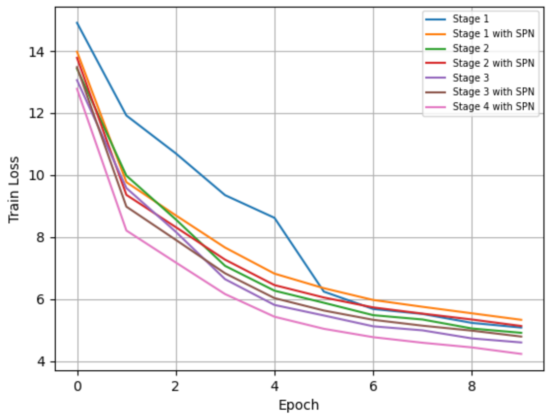

Figure 9. AnyNet - Train Loss

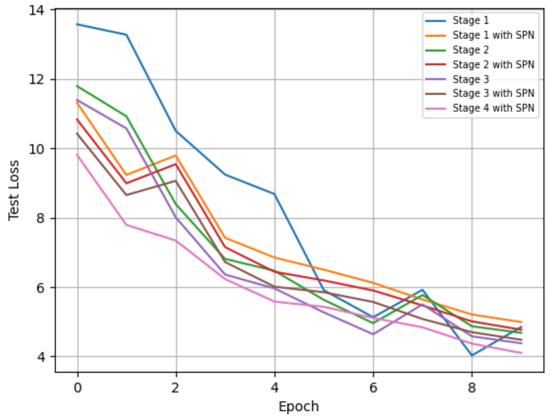

Figure 10. AnyNet - Test Loss

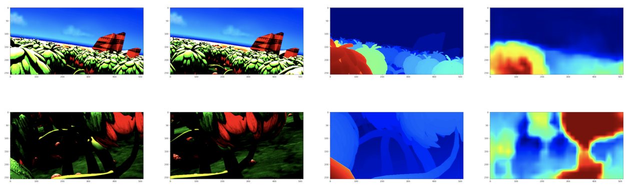

Figure 11. AnyNet - Left, Right inputs; Target & Predicted disparity

4.3. Complexity Reduction

Once we established our baseline with both the models, our subsequent goal was to evaluate computational complexity of the baseline models and investigate methods to

Table 1. Monocular Depth - Model structure

Block	Structure
Down-scale	2x(Conv2D-BN-Act)+Pool
Bottleneck	2x(Conv2D-Act)
Up-scale	2x(Conv2D-BN-Act)

reduce the model's computational complexity. Model size and number of parameters are our metrics to measure reduction in compute usage. On an embedded device mounted on a battery power autonomous system, reduction in model size can boost the efficiency of the whole system. Our evaluation metric remains the same, i.e. lower loss, lower SSIM for Monocular Depth model and lower three pixel error for Stereo Depth.

The subsections below describe experiments and results on reducing computational complexity of the models.

4.3.1 Monocular Depth

Monocular depth model has an autoencoder structure with 4 Down-scale blocks, 1 bottleneck block and 4 Up-scale blocks. Table 1 describes structure of the model (Conv2D is a 2-D convolution; BN is batch normalization; Act is activation function).

The filter sizes for the monocular model described in table 1 are in increased order 16-32-64-128-256. The higher order down-scale and upscale blocks are computationally expensive. Experiments were done in order to reduce the dependency on higher order filters and evaluate model accuracy, with reference to the baseline.

The table below 1 describes three model variants, the first row (Model-1) is the default monocular depth model. The second row (Model-2) is model variant by removing the higher order down scale and upscale blocks. This variation reduces the number of model parameters drastically, as shown in the table. In both Model-1 and Model-2, the activation function used is default LeakyRelU(0.2). Activation function was changed to Swish based on [9] to investigate if it improves accuracy. This variant is shown in last row (Model-2 Swish) of the table 2. The last column of the table shows SSIM evaluation between target depth map and predicted depth map after training for 20 epochs and model convergence was achieved.

The number of model parameters reduces from 1966467 to 489091, which is a 75.17% reduction in model size!

For the given dataset, reducing the model structure from 4-1-4 to 3-1-3 reduced model parameters by more than two thirds and yet maintained relatively acceptable performance with respect to baseline.

The figure 12 shows SSIM comparison of 6 random data samples from test set. The figure 13 shows visual comparison of the depths maps obtained by 3 different models from

Table 2. Monocular Depth - Model variants, sizes & accuracy

Model Structure	Activation	Parameters	Test SSIM
4-1-4	LeakyReLU	1966467	0.9895
3-1-3	LeakyReLU	489091	0.9903
3-1-3	Swish	489091	0.9871

2.

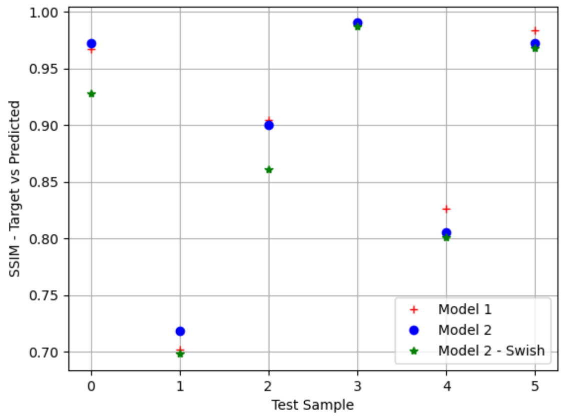

Figure 12. Monocular Depth - Models Comparison

The figure 13 shows visual data for the test samples 0, 2, 3, whose corresponding SSIM is shown in the figure 12.

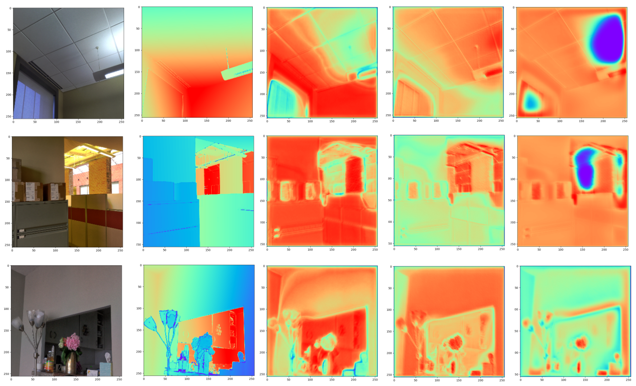

Figure 13. Monocular Depth - Input, Target, Model-1, Model-2, Model-2 Swish

Based on this, we can conclude that the model complexity can be reduced without much impact to predicted depth map as shown by Model-1 vs Model-2. An interesting observation is that replacing LeakyReLU with another advanced activation function Swish did not improve the accuracy of the model.

Another important observation can be made that although SSIM measured on the 3 models, as shown in 1 and

12 is close to 1 and very close to each other, the depth maps shown by figure 13 show huge difference in qualitative perception of depth map. This can explain that SSIM alone cannot be a good loss function for training the Monocular Depth Estimation model. Thus, we can conclude that although the models converged acceptably during initial baseline ablation studies, predicted depth maps did not resemble target.

4.4. Stereo Depth - AnyNet

AnyNet model's structure is mainly comprised of 3 sections as shown in 2. First section is a UNet for feature detection of left and right images separately. Second section is a 4D cost volume reduction. Third section is Spatial Propagation Network (SPN). SPN has larger number of channels in the filters. As UNet is crucial part of the model to detect disparity features, this is not primary focus to reduce complexity of the model. Furthermore, [12] concretely suggests using larger models instead of using UNet if more accuracy is needed at the expense of larger model. Thus, we started by studying effect of SPN on the model accuracy.

Similar to exercise done on previous model, our metric of reducing complexity of AnyNet is to reduce the number of model parameters, which in turn is a direct metric of reduction in compute and memory resources required.

We compared 5 model variants of AnyNet in Table 3. Each model's total number of parameters and average three pixel error across entire test are shown in the table. The lesser the three pixel error, the higher the accuracy of the model. Fig 16 shows distribution of three pixel error across all the samples in test set (800 samples). We do observe some outliers but it is promising to see that all variants of the model have very low error.

Table 3. AnyNet - Model variants, sizes & accuracy

Model Structure	Parameters	Three Pixel Error
No SPN	34629	0.2994
SPN 1 channel	34827	0.3048
SPN 2 channels	35277	0.3193
SPN 4 channels	36933	0.3264
SPN 8 channels	43269	0.3178

The figures 14 and 15 show training and test loss convergence for these 5 model variants.

Figures 17 and 18 showcase visually the disparity maps obtained by training the 5 variants of AnyNet models. As observed, the difference is relatively low on model with 1 channel SPN vs the model with 8 channel SPN, with 20% lesser model size as shown by the distribution of the 25th and 75th quartiles.

From the before mentioned experiments we can conclude that addition of SPN improves accuracy slightly compared to without SPN. As shown in 3 we can choose 1 channel SPN based AnyNet instead of 8 channel SPN with 20% lesser model size.

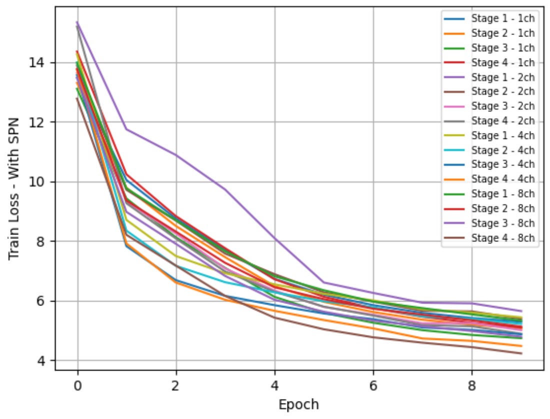

Figure 14. AnyNet - Train Loss for 5 model variants

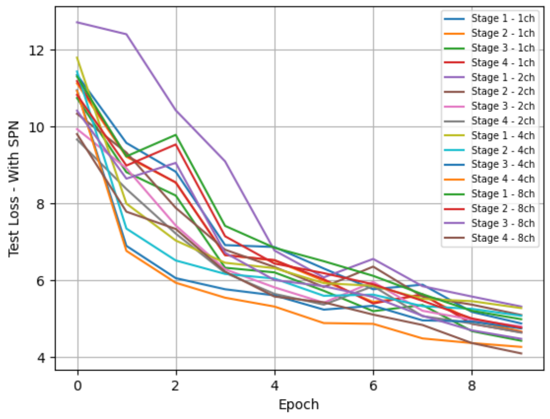

Figure 15. AnyNet - Test Loss for 5 model variants

5. Conclusion

As an initial step, we reproduced learning based methods to estimate depth using both, Monocular and Stereo Vision techniques. We performed ablation studies to tune hyper parameters such as learning rate in order to improve accuracy. We investigated on using various loss functions on monocular depth estimation. After establishing baselines in both methods, we experimented with modifying model architecture by changing the neural network layer sizes. We also changed activation functions to improve accuracy after reducing network size.

For stereo depth estimation, we started from the novel state-of-the-art approach of [12] and studied advantages of reducing the spatial propagation network in reducing model

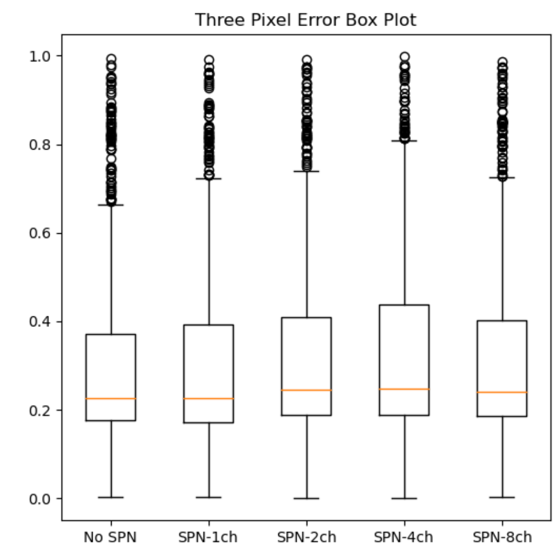

Figure 16. AnyNet - Three Pixel Error on test set for 5 model variants

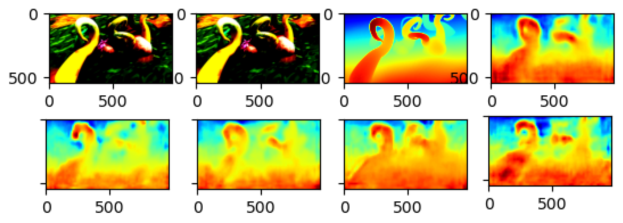

Figure 17. AnyNet - Test Sample 0 - Top Row (Left to Right): Left input, Right input, Target Disparity, Disparity with No SPN Model; Bottom Row (Left to Right): Disparity with 1 channel SPN, Disparity with 2 channel SPN, Disparity with 4 channel SPN, Disparity with 8 channel SPN

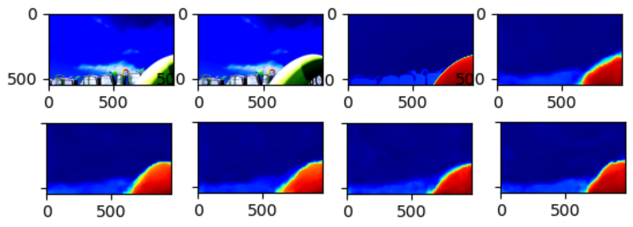

Figure 18. AnyNet - Test Sample 1 - Top Row (Left to Right): Left input, Right input, Target Disparity, Disparity with No SPN Model; Bottom Row (Left to Right): Disparity with 1 channel SPN, Disparity with 2 channel SPN, Disparity with 4 channel SPN, Disparity with 8 channel SPN

complexity.

In both the methods, we established that reducing model size is a viable option to reduce computational complexity, that consequently reduces energy consumption for deploying these techniques in highly constrained navigation systems. To further improve quality of experiments, future work can use same dataset on both monocular and stereo depth techniques for a quantitative and qualitative comparison of both models. However, stereo technique may always outperform monocular depth technique in accuracy.

After reducing the model sizes, we also tried using multi-precision casting in both Pytorch and Tensorflow but were unsuccessful in converting the model from a 64b/32b to a bfloat16 or float16 model. This could be a reasonable future work that can be undertaken as reducing the data type from 32b to 16b further reduces compute and memory needs by 50%. For the most part, this would only need fine-tuning the model. Similarly, the models can be further reduced computationally by pruning and quantizing but this would hurt the accuracy furthermore. Thus, quantizing and pruning may need additional investigative work (FastDepth [14] research used pruning in their method).

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 3

[2] J. S. Ashutosh Saxena and A. Y. Ng. Depth estimation using monocular and stereo cues, 2007. 1

[3] V. Basu. Keras-monocular depth estimation, 2021. 1

[4] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 2

[5] H. Laga, L. V. Jospin, F. Boussaid, and M. Bennamoun. A survey on deep learning techniques for stereo-based depth estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1, 2020. 1, 2

[6] N. Mayer, E. Ilg, P. Häusser, 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 *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. arXiv:1512.02134. 2

[7] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 2

[8] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, edi-

tors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 4

[9] P. Ramachandran, B. Zoph, and Q. V. Le. Searching for activation functions. *CoRR*, abs/1710.05941, 2017. 5

[10] S. T.-V. Thatcher Freeman. Cs231a - training monocular depth estimation models on a budget, 2021. 1

[11] I. Vasiljevic, N. Kolkin, S. Zhang, R. Luo, H. Wang, F. Z. Dai, A. F. Daniele, M. Mostajabi, S. Basart, M. R. Walter, and G. Shakhnarovich. DIODE: A Dense Indoor and Outdoor DEpth Dataset. *CoRR*, abs/1908.00463, 2019. 2

[12] Y. Wang, Z. Lai, G. Huang, B. H. Wang, L. van der Maaten, M. Campbell, and K. Q. Weinberger. Anytime stereo image depth estimation on mobile devices. *CoRR*, abs/1810.11408, 2018. 1, 2, 4, 6

[13] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *Trans. Img. Proc.*, 13(4):600–612, apr 2004. 2

[14] D. Wofk, F. Ma, T.-J. Yang, S. Karaman, and V. Sze. Fastdepth: Fast monocular depth estimation on embedded systems, 2019. 1, 7