CS482/682 Final Project Report Group 14

Lightning Fast Monocular Depth Estimation

Maxwell Fleming, Justin Sech, Jacob Whitaker, Abhinav Chinta
(mflemi21, jsech1, jwhita14, achinta3)

1 Introduction

Background Our eyes observe our world from two different perspectives, and our brains automatically construct a third dimension, depth, from two flat images. However, when you close your right eye, you still retain significant intuition about depth. You may lose absolute precision, but our brains are able to "fill in" for the missing right eye, giving us an accurate depth estimation. The goal of this project is precisely that: teach a single eye to learn what the world looks like in 3-dimensions. Specifically, we implement self-supervised, monocular depth estimation using a deep convolution encoder-decoder architecture to produce a pixel-level depth assignment comparable to ground-truth depth maps in the KITTI dataset [2].

Related Work Early research in depth estimation focused on directly mapping depth from motion or stereo images. [1] introduced one of the earliest methods for creating pixel-level depth estimates from a *single* image; however, it requires depth-map supervision. Imprecise and expensive depth-sensor technology makes it technically and economically challenging to produce large, diverse datasets of ground truth depth maps, especially for a complex variety of scenes. [3] introduces a novel monocular approach that advantages left-right stereo image pairs for self-supervision without ground-truth depth.

2 Methods

Dataset The model was trained and evaluated on the KITTI dataset [2] containing 42,382 rectified stereo pairs from 61 scenes, with a typical image size of (375 × 1242) pixels. The train split of 29,000 images was down-sampled to (256 × 512), normalized, and augmented with horizontal flips, rotations, and random adjustments to brightness, contrast, and saturation.

Model Architecture Our model is a re-implementation of [3]. Our model is a deep-convolution encoder-decoder network that uses skip-connections to retain higher-resolution information in its deeper layers. While [3] found best results with a custom built ResNet50 encoder (48M params), we leveraged transfer learning by using a significantly smaller, pre-trained ResNet18 [4] with just 11M parameters. Our decoder structure consists of 6 "deconvolution" blocks. Each block spatially doubles the input with a transposed convolution, joins its output with the skip-connection saved from the spatially-matched encoder layer, and then passes the result through an additional convolution layer. Each transposed and standard convolution layer is followed by batch normalization and an exponential linear units (ELU) activation to avoid clipping small disparity values with standard ReLU.

Training At train time the model was fed stereo images with the goal to learn a *disparity mapping* that encodes a pixel-level transformation from left image to right image (and vice-versa). The major learning assumption is that learning this transformation will effectively teach our model fundamental information about depth. Our model learns to compute a disparity mapping at 4 stages of resolution with a separate convolution layer and sigmoid activation.

Self-Supervised Loss We use each of the 4 disparity resolutions to compute loss to enforce consistency in our disparity at each resolution. At each resolution, we compute loss as a weighted sum of three terms: appearance matching loss (AML), disparity smoothness loss (DSL) and left-right disparity consistency loss (LRDCL). AML measures how well the disparity and the opposite image and can reconstruct the original image. It is a weighted sum of L1 loss and SSIM loss [5] between the original and reconstructed image. DSL enforces smooth disparity gradients by applying L1 loss to the disparity gradients while still allowing true edges by using the exponential norm of image gradients as weights. Finally LRDCL is a measure of how well a disparity can reconstruct itself when applied to the opposite disparity. An essential part of these loss functions is that they never use ground truth depth, only the left and right images.

Evaluation Our model's final output is a left disparity map. We convert this to a depth map by inverting it and multiplying by a calibration factor from properties of the KITTI stereo camera. Afterwards we can compute error metrics by comparing our predicted depth map to the ground truth depth maps. While visualizing our disparities, we noticed they "looked" correct but produced poor depth values. We corrected this by scaling our predicted depth maps by the ratio of the median depth of ground truth and the median depth our prediction. This is justifiable as we only care about the relative depth of objects (depth distribution), and we scale for the sole purpose of evaluation. We display the effect of this scaling in Figure 1.

3 Results

We evaluated our model on the Eigen splits of the KITTI dataset. Table 1 summarizes how we compare to [1] and [3]. As represented by our relatively high Abs Rel, Sq Rel, RMSE, and RMSE log scores, our model sometimes predicts a very wrong depth value on a given pixel, however, our $\delta < 1.25$, $\delta < 1.25^2$ and $\delta < 1.25^3$ scores show our model correctly predicts depth within each threshold for the majority of pixels. We visualize this in Figure 2.

	Abs Rel ↓	Sq Rel ↓	RMSE ↓	RMSE log ↓	$\delta < 1.25$ ↑	$\delta < 1.25^2$ ↑	$\delta < 1.25^3$ ↑
Ours	0.283	3.259	9.222	0.490	62.9%	85.7%	94.3%
Eigen [1]	0.203	1.548	6.307	0.282	70.2%	89.0%	95.8%
Godard [3]	0.148	1.344	5.927	0.247	80.3%	92.2%	96.4%

Table 1: For each metric ↓ means lower is better and ↑ means higher is better

One of the highlights of our model is that our training time is 125x faster when compared to the original implementation [3]. Our implementation trains in approximately 10 minutes using a single RTX 3080 on a dataset of 30 thousand images for 4 epochs using SGD with learn rate of 10^{-2}, momentum of 0.9, and batch size of 22.

4 Discussion

We feel the best way to showcase our understanding of the problem and our model is to describe steps we made debugging two major errors. Our initial model's disparity outputs were poor. As seen in Figure 3, half of each disparity map contained some information, while the other half was completely zeroed. Disparity represents the shift required to map a pixel to its stereo-pair. Our model was learning that half of the image should not be shifted whatsoever. We traced this to the AML loss function and found that, while we correctly applied disparity to generate an image, we compared it to the incorrect image (ex. generating right and comparing to left). This confirmed our suspicion, and we could now learn full-dimensioned depth estimates. Visualizing our next model's outputs in Figure 4 indicated we were only learning edge detection. This told us our loss was incorrectly encouraging sharp edges. We traced this to the DSL component of our loss which should have been rewarding the exact opposite behavior. We found we were regressing raw gradients rather than their absolute values, which in turn was pushing our model towards edges rather than away from them. Incorporating this fix immediately corrected our models behavior to successfully learn much smoother, interpretable disparity visualizations.

References

[1] David Eigen, Christian Puhrsch, and Rob Fergus. “Depth Map Prediction from a Single Image using a Multi-Scale Deep Network”. In: *CoRR* abs/1406.2283 (2014). arXiv: 1406.2283. URL: http://arxiv.org/abs/1406.2283.

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun. “Are we ready for autonomous driving? The KITTI Vision Benchmark Suite”. In: May 2012, pp. 3354–3361. ISBN: 978-1-4673-1226-4. DOI: 10.1109/CVPR.2012.6248074.

[3] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. *Unsupervised Monocular Depth Estimation with Left-Right Consistency*. 2017. arXiv: 1609.03677 [cs.CV].

[4] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[5] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861.

Supplemental Materials: Lightning Fast Monocular Depth Estimation

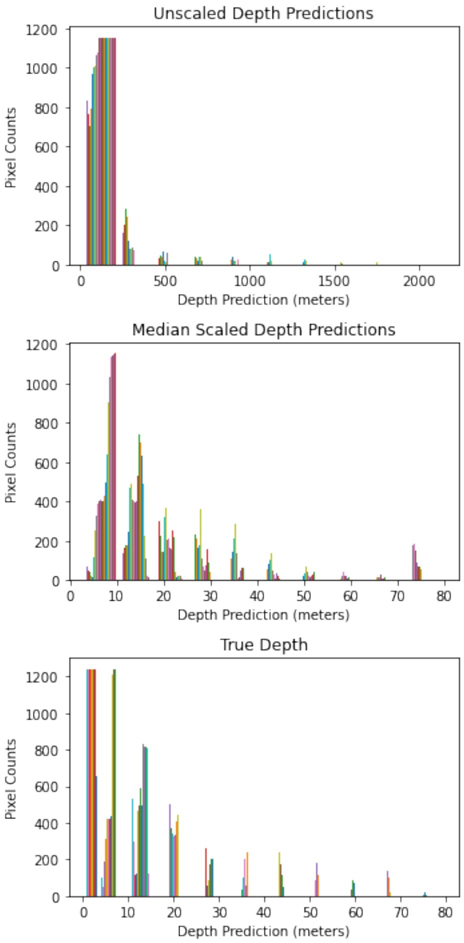

Figure 1: These histograms represent the different pixel depths for our unscaled depth predictions, median scaled depth predictions, and true depth. The unscaled depth values are clearly not correct as they predict depths that are far too deep. To fix this, we scale our depth predictions within the range of the ground-truth so we can properly evaluate depth distribution. The values from this histogram come from the same images visualized in Figure 2

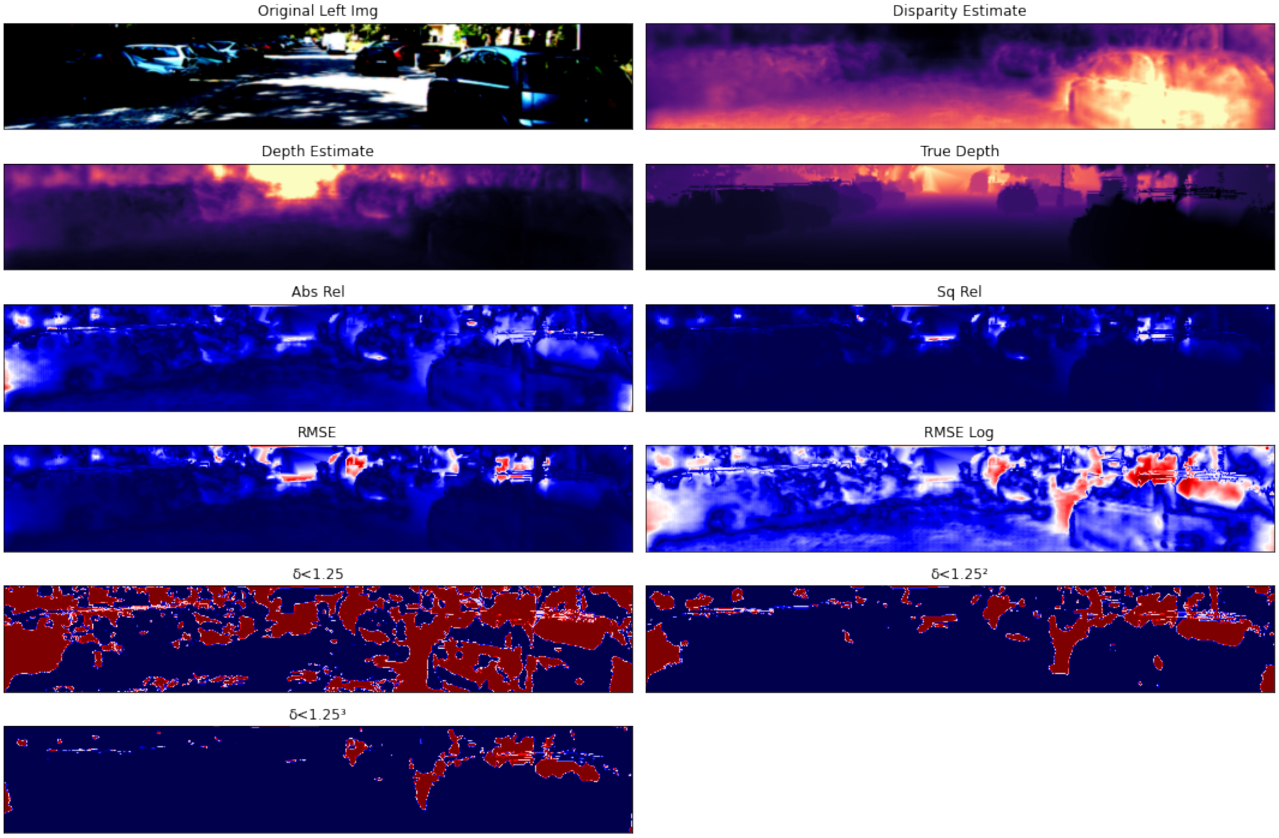

Figure 2: This is an example output from out model, including our disparity and depth estimates. We also include visualizations of the error for each pixel in the image where blue represent areas of low error and red represents areas of high error. Notice, the disparity map visualization illustrates logical information as the depth gradually increases towards the back of the image. The disparity appropriately captures the shallowest car but without especially fine-grain precision and detail. Clearly there is still higher-resolution information to be learned, but overall our implementation captures the majority of depth information.

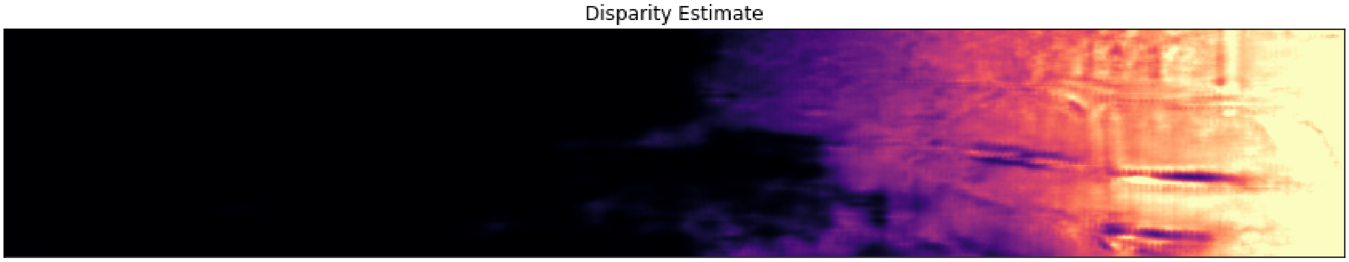

Figure 3: This is the disparity output from one of our early buggy models. As shown, the model is only predicting half of the disparity

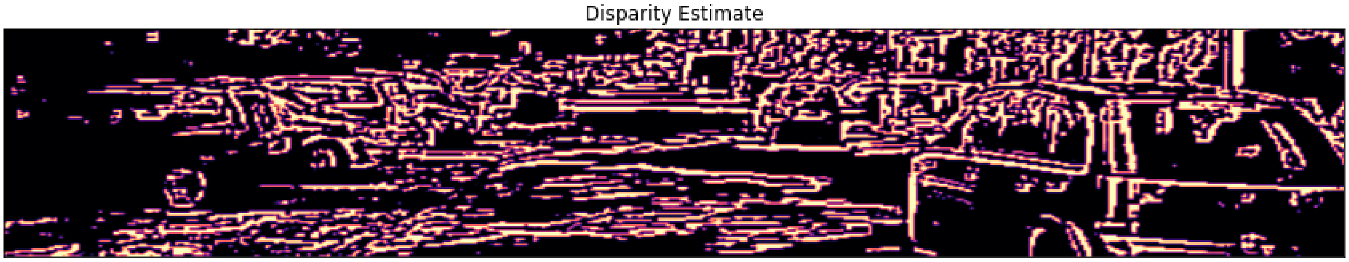

Figure 4: This is the disparity output from another buggy models. As shown, the predicted disparity is very sharp and look a lot like edge detection.