Monocular Depth Estimation using Transfer Learning

Pratyaksh P. Rao, Tharun V.P., Kaustubh Joshi, Mukund Mitra, Abhra Roy Chowdhury

July 9, 2021

Group 9A

1 Introduction

Depth estimation using vision sensors is usually done using stereo cameras so that the two cameras can be used to triangulate the position of a certain point in the environment. However using monocular camera offers benefits like low power consumption, light weight and low cost. Moreover, monocular cameras can also be used for estimating the depth using various monocular cues like defocus, texture variations and gradients, cold/haze, etc. [2]. In this project we focus on the monocular depth estimation problem and its potential applications in deformable Simultaneous Localization and Mapping (SLAM). We estimate the dense depth map of each pixel using a single monocular RGB image using a deep learning-based transfer learning approach. Additionally, we implement a transfer learning-based method and an encoder-decoder structure to solve the monocular depth estimation task. The NYU dataset V2 is used for training the depth estimation models.

As a part of the group project, we have successfully implemented the following -

- Building the deep learning network
- Improving the loss function
- Adding data augmentation
- Designing a new encoder and decoder network

2 Methodology

This section illustrates the framework we have used in defining various loss functions used to decrease the training loss, different data augmentation techniques, and network architecture of a new encoder-decoder structure. The project was implemented by referring to the paper by Allahsim *et. al*[1].

2.1 Network Architecture

The encoder is designed by extracting features from the RGB image using the DenseNet-169, which is a network pretrained on the ImageNet dataset.The top layers related to the ImageNet classification task are removed for encoding the extracted features into a feature vector.

For the decoder, we implement a 1×1 convolutional layer where the number of output channels are equal to the output of the truncated encoder. Subsequently, we add upsampling blocks, each composed of a bilinear upsampling followed by two 3×3 convolutional layers. The first convolutional layer of the two is applied on the concatenation of the output of the previous layer and the pooling layer from the encoder having the same spatial dimension. Except the last layer, every upsampling block is followed by a leaky ReLU activation function with parameter $\alpha = 0.2$.

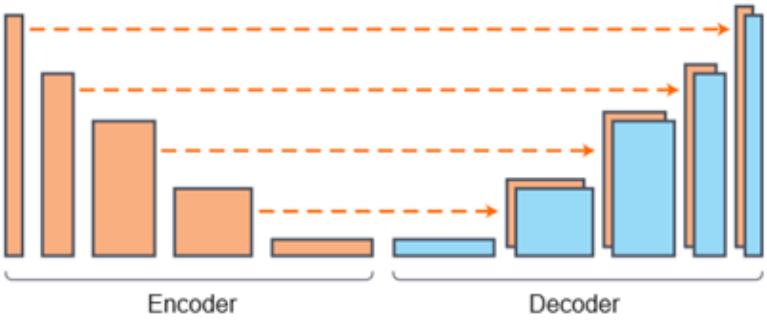


Figure 1: Network Architecture [1]

2.2 Loss function

In depth regression problems, a loss function is defined as a function of difference between the ground truth depth map y and the prediction of depth regression network $\hat{y}$. To train our network, we incorporate different loss functions and compare them on an individual basis as well as their weighted sums. We attempt to find a combination and subsequently a loss function which minimizes the difference of depth values while also penalizing distortions of high frequency details in the image domain of the depth map [1].

2.2.1 Point-wise L1 loss

The L1-loss is defined as the mean absolute error between the ground truth and predicted depth map. It is given by the following equation -

$$L_{\text{depth}}(y, \hat{y}) = \frac{1}{n} \sum_{p}^{n} |y_p - \hat{y}_p|$$

2.2.2 L_{grad} for image gradient

L_{grad} is the L1 loss defined over image gradient g of the depth images. Moreover, it penalizes distortions of high frequency details in the image. The loss function is given as follows

$$L_{\text{grad}}(y, \hat{y}) = \frac{1}{n} \sum_{p}^{n} |g_x(y_p, \hat{y}_p| + |g_y(y_p, \hat{y}_p|$$

2.2.3 Structural Singularity

L_{SSIM} is a common metric used for image reconstruction tasks. It is mostly used to improve the quality of image reconstruction. It is defined as follows -

$$L_{SSIM}(y, \hat{y}) = \frac{1 - SSIM(y, \hat{y})}{2}$$

2.2.4 Huber Loss:

$$L_{\delta}(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{for} |y - f(x)| \leq \delta \\ \delta |y - f(x)| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases}$$

2.2.5 Reverse Huber Loss (BerHu Loss)

The BerHu loss is equal to the $L_1(x) = |x|$ norm when $x \in [c, c]$ and equal to L_2 outside this range. The version used here is continuous and first order differentiable at the point c where the switch from L_1 to L_2 occurs. In every gradient descent step, when we compute $B(\hat{y} - y)$ we set $c = \frac{1}{5} max_i(|\hat{y}_i - y_i|)$, where

i indexes all pixels over each image in the current batch, that is 20% of the maximal per-batch error. Empirically, BerHu shows a good balance between the two norms in the given problem.

$$B(x) = \begin{cases} |x| & |x| \leq c \\ \dfrac{x^2 + c^2}{2c} & |x| > c \end{cases}$$

2.2.6 Weighted Loss

Since each loss function has its own functionality, we take a weighted sum. The weights are tuned for optimal performance (w1 = 0.1, w2=w3=1).

$$L(y, \hat{y}) = w_1 * L_{depth}(y, \hat{y}) + w_2 * L_{grad}(y, \hat{y}) + w_3 * L_{SSIM}$$

Note: An inherit problem with such loss terms is that they tend to be larger when the ground-truth depth values are bigger. In order to compensate for this issue, we consider the reciprocal of the depth.

2.3 Data Augmentation

Data augmentation is the process through which individual samples in the dataset are manipulated while preserving their innate characteristics which leads to a better generalization thereby reducing overfitting. The original image is as shown in Fig. 2. We have performed the operations as shown in Fig 3. and Fig 4.

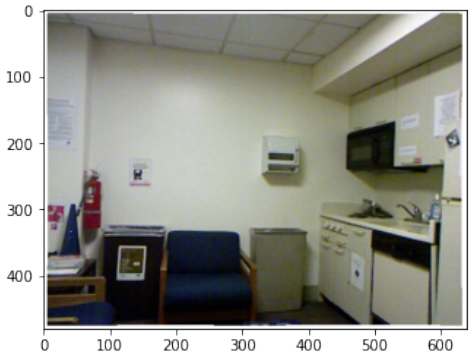
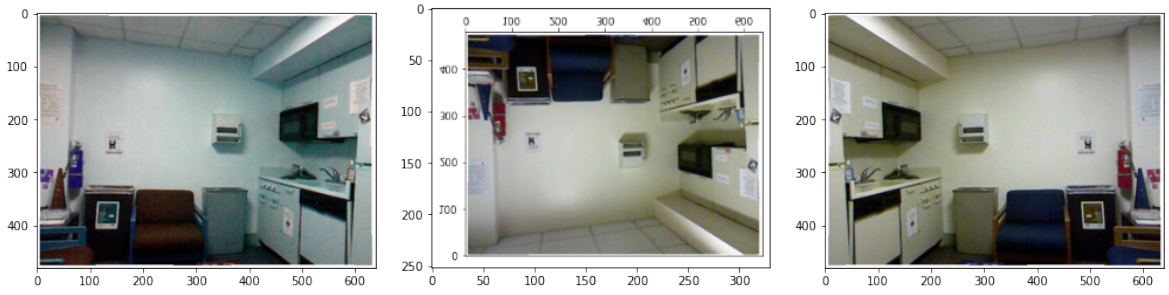
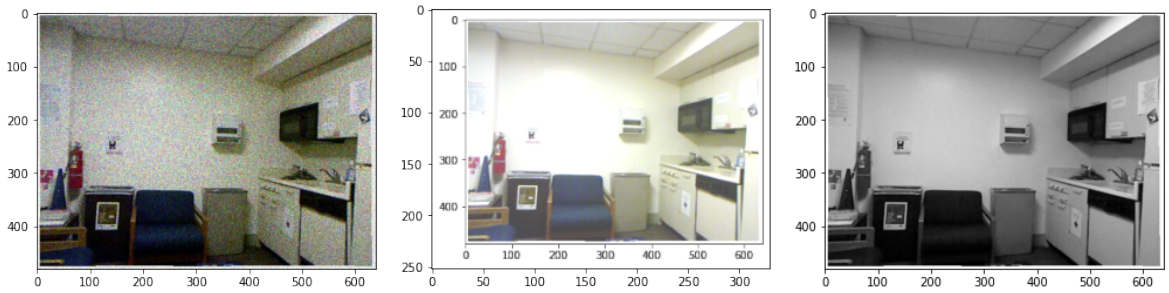


Figure 2: Original Image



(a) (b) (c)

Figure 3: (a) RGB-BGR (b) Vertical Flip (c) Horizontal Flip

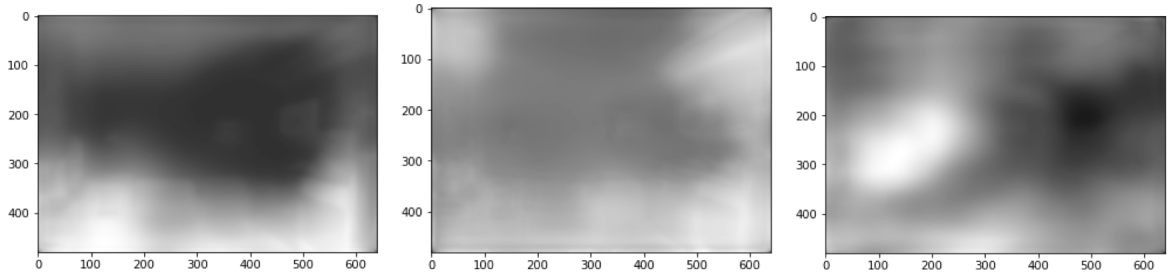


(a) (b) (c)

Figure 4: (a) Salt & Pepper (b) Brightness (c) Saturation

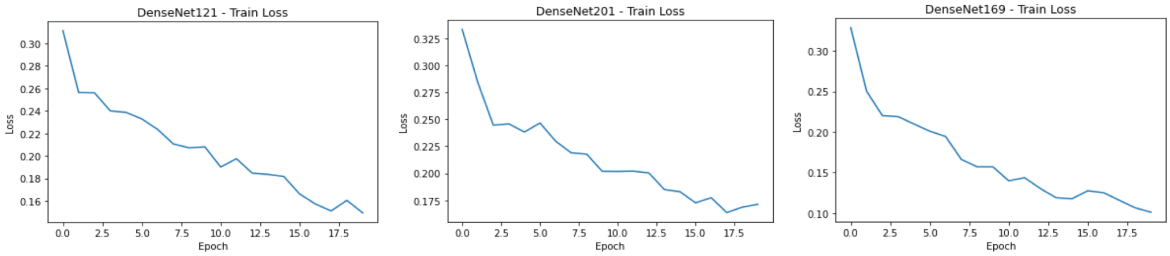
3 Results

Due to lack of suitable processing capabilities, we trained the model using only 1000 RGB images. Moreover, we trained the model on just 20 epochs for faster computation and timely completion of the project.



(a) (b) (c)

Figure 5: Depth Image:(a) DenseNet121 (b) DenseNet201 (c) DenseNet169



(a) (b) (c)

Figure 6: (a) DenseNet121 (b) DenseNet201 (c) DenseNet169

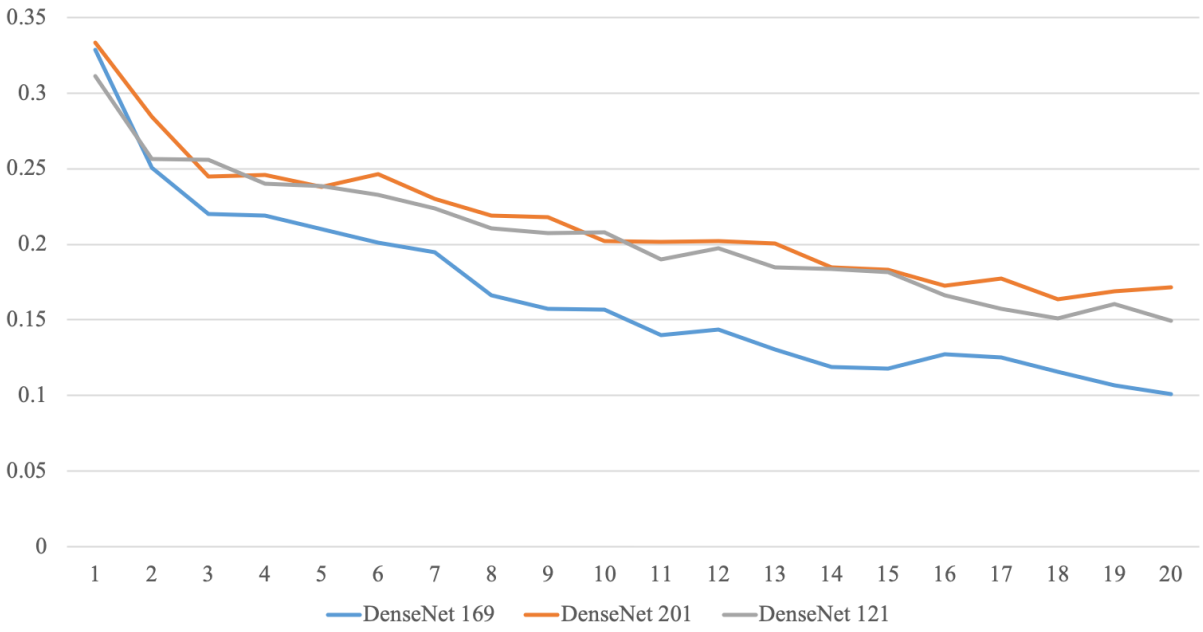


Figure 7: Comparison of Model Loss vs Epochs

4 Conclusion and Future Work

In this project, we have implemented DenseNet169 that is taken as a baseline for validating the performance of two modified encoder-decoder architectures. The NYU dataset V2 is used for training the above mentioned depth estimation models. Using an extensive analysis of different loss functions and data augmentation techniques, the performance of the three models were compared. It was observed that DenseNet169 outperformed the custom encoder-decoder architectures. Moreover, the weighted sum of the loss functions decreased the model loss. Lastly, by incorporating various data augmentation techniques, a significant amount of variance was brought into the dataset that subsequently helped in enhancing the overall performance of the model.

Monocular depth estimation is an useful tool in multiple fields where there are constraints on hardware size, cost and power consumption. A significant application of this work is in space-constrained dynamically deformable environments in the human body for functions like endoscopy or colonoscopy. Additionally, it can be a useful tool in the search and rescue operations for examination while navigating through small cracks for an extended duration of time.

This method for depth estimation can be improved upon in the future by devising new and better loss functions. Additionally, an improvised weighted combination of the same or better loss functions can be devised to get a better estimate of the depth.

References

[1] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *arXiv preprint arXiv:1812.11941*, 2018.

[2] Ashutosh Saxena, Jamie Schulte, Andrew Y Ng, et al. Depth estimation using monocular and stereo cues. In *IJCAI*, volume 7, pages 2197–2203, 2007.