Monocular Depth Estimation using Deep Learning

Gopal Krishna
Northeastern University
Boston, MA
krishna.g@northeastern.edu

Gourav Beura
Northeastern University
Boston, MA
beura.g@northeastern.edu

***Abstract*—Predicting depth is an essential component in understanding the 3D geometry of a scene. While for stereo images local correspondence suffices for estimation, finding depth relations from a single image is less straightforward, requiring integration of both global and local information from various cues. Accurate depth estimation from images is a fundamental task in many applications including scene understanding and reconstruction. Monocular depth estimation i.e. depth estimation from 2D images is a fundamental task in scene understanding and reconstruction. Having a dense depth map of the real-world can be very useful in applications including navigation, augmented reality, image refocusing, and segmentation. Recent developments in monocular depth estimation are focusing on using convolutional neural networks (CNNs) to perform 2D to 3D reconstruction. While the performance of these methods has been steadily increasing, there are still major problems in both the quality and the resolution of these estimated depth maps. We attempt to experiment with different techniques in deep learning to evaluate how they fare with the task of monocular depth estimation.**

***Index Terms*—deep learning, depth estimation, stereo, transfer learning, deep supervision, augmentation**

I. Introduction

Depth estimation plays a crucial role in contemporary computer vision tasks, such as autonomous driving systems, 3D face recognition, mixed reality etc. At present, there are two main ways for depth estimation- the most frequently used active approach to get the depth are using sensors and structured light cameras which are part of illuminating coded signals technique. The other approaches follow a passive way, which is usually based on multi-view geometry, such as stereo depth estimation. Stereo depth estimation is an important area of research in computer vision and is commonly achieved by employing stereo vision, in which images from two cameras are used to triangulate and estimate distances. Over the past few decades, researchers have developed outstanding stereo vision based systems. Even though these systems work well in many environments, stereo vision is essentially restricted by the baseline distance between the two cameras because the depth estimates tend to be inaccurate when the distances considered are large (i.e. even very minute camera calibration estimation errors translate to very large errors in distances). Further, stereo vision also tends to fail for regions with any texture in images, because correspondences cannot be reliably found between the image pairs.

However, every image frame also contains numerous monocular visual cues — such as texture variations and gradients, de-focus, color/haze, etc. that have not been utilized properly in stereo systems. Some of these cues apply even in regions without texture, where stereo would work well. Humans perceive depth by seamlessly combining many of these stereo and monocular cues, yet most work on depth estimation has focused on stereo vision, and on other algorithms that require multiple images such as structure from motion or depth from de-focus. Monocular depth estimation is a difficult task, which requires taking into account the global structure of the image. There is much prior work on estimating depth based on stereo images or motion, yet the monocular case often arises in practice with potential applications including better understanding of the many images distributed on the web, social media outlets, real estate listings, and shopping sites. These include many examples of both indoor and outdoor scenes.

There are likely several reasons why the monocular case has not yet been tackled to the same degree as the stereo one. Provided accurate image correspondences, depth can be recovered deterministically in the stereo case. Thus, stereo depth estimation can be reduced to developing robust image point correspondences — which can often be found using local appearance features. By contrast, estimating depth from a single image requires the use of monocular depth cues such as line angles and perspective, object sizes, image position, and atmospheric effects. Furthermore, a global view of the scene may be needed to relate these effectively, whereas local disparity is sufficient for stereo.

In this paper we present existing approaches for monocular depth estimation from a single image. We directly regress on the depth using convolutional neural network with two components- an encoder that first estimates the global structure of the scene, then a decoder that refines it using local information. The network is trained using a loss that explicitly accounts for depth relations between pixel locations and the difference between edge predictions in addition to the pointwise error. We follow Alhashim et. al's proposed transfer-learning based technique as baseline and employ a encoder-decoder based neural network structure to learn image to image transformation from RGB images to depth maps. We then proceed to try out different techniques like augmentation, deep supervision and improved loss functions to improve upon the performance of the network.

II. Related Work

There are certain complications associated with 3D scene reconstruction from RGB images that have not been tackled to the degree as Stereo based approaches. Limitations like lack of scene coverage, translucent, scale ambiguities, or reflective materials lead to obscure cases where extracting geometry from appearance becomes a very complicated task. In current real-world practices, the more successful approaches for capturing depth of a scene rely on hardware assisted methods, such as using sensors, or require a large number of scenes captured using high quality cameras along with expensive offline reconstruction process. Recently, Deep neural networks learning based methods are able to produce reasonable depth maps from a single or couple of RGB images at real-time input. There are multiple existing approaches that directly related to our work of estimating depth from a single image.

In recent work, Alhashim et al. [1] proposed a simple transfer learning-based network architecture that produces depth estimations of higher accuracy and quality. The output depth maps capture object boundaries more reliably than those generated by existing methods with fewer parameters and less training iterations. Second, defined a corresponding loss function, learning strategy and simple data augmentation policy for faster learning.

Eigen et al. [2] presented a new approach for estimating depth from a single image.They directly regressed on the depth using a neural network with two components: one that first estimates the global structure of the scene, then a second that refines it using local information. Their network was trained using a loss that explicitly accounts for depth relations between pixel locations, in addition to pointwise error.

Ladicky et al. [3] showed the process of monocular depth estimation by integrating semantic object labels with features to improve performance. But, they used handcrafted features and superpixels to create image segmentation.

Karsch et al. [4] used a a transfer mechanism with k-Nearest Neighbours on SIFT Flow [8] to estimate depths of static backgrounds from single RBG images, which they augmented with motion information to improve the estimation of moving foreground subjects in videos.

There have been several Machine learning techniques applied in the stereo vision case, often obtaining better results while relaxing the need for careful camera alignment [7] [9] [10] [11]. Most relevant to this work is Konda et al. [7], they trained a factored auto-encoder on image batched to predict depth from stereo sequences; however, this relied on the lotion difference provided by stereo.

Nassir et. al utilized depth sensors, like RGBD cameras and LIDAR, to get the depth information of the corresponding image to get the pixel-level dense depth map directly, but their expriment suffered a limition in measurement range and outdoor sunlight sensitivity [12]. Although LIDAR is widely used in unmanned driving industry for depth measurement [13], it only generated the sparse 3D map. Due to the low cost, small size and wide applications of monocular cameras, estimating the dense depth map from a single image has received more attention, and it has been well researched based on deep learning in an end-to-end manner recently.

III. Proposed Method

We describe a method for estimating depth map from a single RGB. We first describe the U-Net [5] like encoder-decoder architecture that is employed by Alhashim et. al [1]. We then discuss our observations on the varieties of both encoder and decoder and their impact on the performance. Next, we describe an appropriate loss function for the task at hand. We also define guidelines for image augmentation policies that may further improve the model's training procedure.

A. *Network Architecture*

We experiment with multiple architectures following encoder-decoder architecture generally with different encoders. For our encoder, the RGB image is encoded into feature vectors either using U-Net [5] encoder or state-of-the-art convolutional neural network architectures pre-trained on ImageNet [15]. We perform most our experiments with the DenseNet-169 [14] network that is pre-trained on the aforementioned ImageNet dataset. The feature vector encoded by the encoder is then input into successive upsampling layers, using either transpose convolutions or bilinear upsampling followed by convolutional layers, in order to construct the final output depth map at the original input resolution. The upsampling layers and associated skip-connections together form the decoder. The decoder contains Batch Normalization [16] and the ability to turn it on or off. Further details about the network architecture and its layers are discussed in further sub-sections. The complexity and various components of our architecture makes it interesting to analyze which components contribute most towards achieving quality depth maps. As mentioned earlier in the section we have experimented with different state-of-the-art encoder backbones of more or less complexity than the DenseNet-169. We also looked at different versions of the decoders and empirically found that the setting of an encoder-decoder architecture with more convolutional blocks exhibiting more complexity do not necessarily help the performance. Our experiments also show that a simple decoder made of bilinear upsampling step followed by two standard convolutional layers performs adequately well.

B. *Learning and Inference*

Standard loss function for depth regression problems consider the difference between the ground truth depth map y and prediction of the depth regression network $\hat{y}$. Loss function can have significant impact on the training speed and overall depth estimation performance. Many variations of standard loss functions employed for optimizing the neural network can be used in the depth estimation problems.

In our method we aim to define a loss function which balances the reconstruction of depth images along with minimizing the difference in depth values while also penalizing high frequency details distortion in the image domain for the

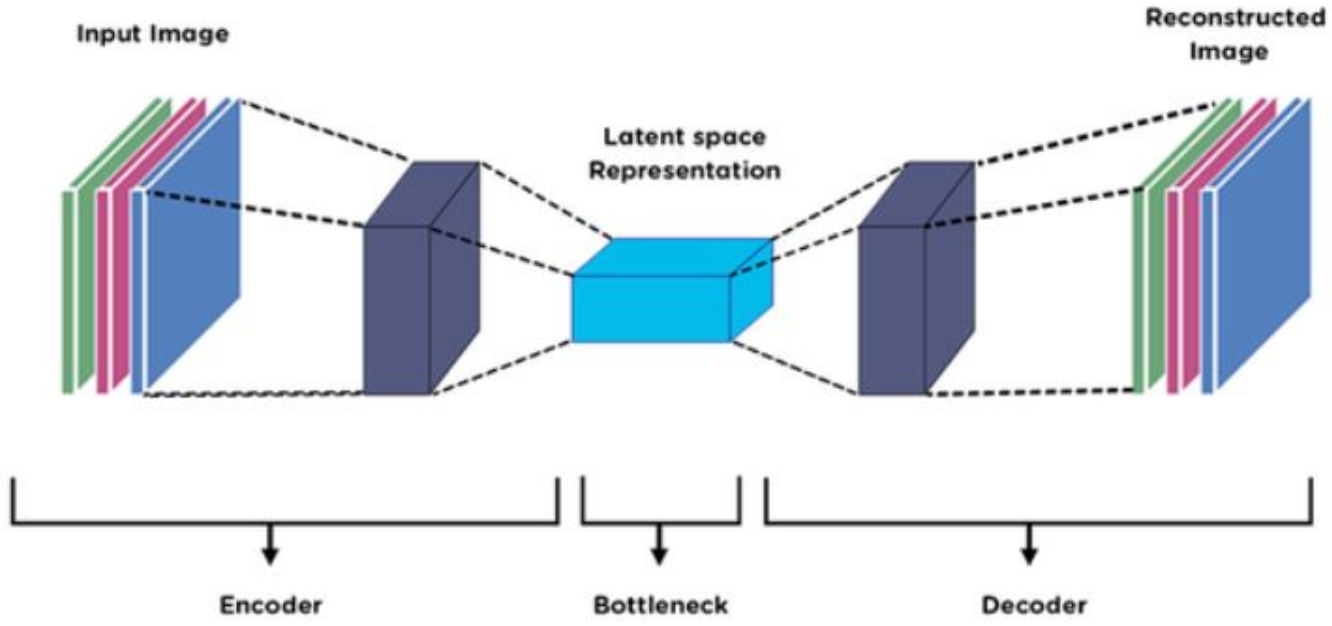

Fig. 1. Representation of Network Architecture

depth map (i.e. details corresponding to the boundaries of objects in the scene). To this end, we found the loss function setup used by Alhashim et. al [1] to be very well suited to the task.

We define the loss used L between y and $\hat{y}$ as the weighted sum of three loss functions:

$$L(y, \hat{y}) = \lambda L_{depth}(y, \hat{y}) + L_{grad}(y, \hat{y}) + L_{SSIM}(y, \hat{y}) \quad (1)$$

The first loss term L_{depth} is the point-wise L1 loss defined on the depth values:

$$L_{depth}(y, \hat{y}) = \frac{1}{n} \sum_{p}^{n} |y_p - \hat{y_p}| \quad (2)$$

The second loss term L_{grad} is the L1 loss defined over the image gradient g of the depth image:

$$L_{grad}(y, \hat{y}) = \frac{1}{n} \sum_{p}^{n} |g_x(y_p - \hat{y_p})| + |g_y(y_p - \hat{y_p})| \quad (3)$$

where g_x and g_y, respectively, compute the differences in the x and y components for the depth image gradients of y and $\hat{y}$.

The third loss term L_{SSIM} uses Structural Similarity (SSIM) [17] term which is commonly-used metric for image reconstruction tasks. It has been recently shown to be a good loss term for depth estimating convolutional neural networks. SSIM has an upper bound of 1, hence we define the loss L_{SSIM} as follows:

$$L_{SSIM}(y, \hat{y}) = \frac{1 - SSIM(y, \hat{y})}{2} \quad (4)$$

We define a weight parameter λ for the loss term L_depth. We set $\lambda = 0.1$ as a reasonable weight for this term.

We found that the above loss term values were prone to being larger when the ground-truth depth values are bigger. In order to compensate for this issue, we consider the reciprocal of the depth where for the original depth map y_{orig} we define the target depth map y as $y = m/y_{orig}$ where m is the maximum depth of the scene (e.g. $m = 10$ meters for the NYU Depth v2 dataset).

C. *Augmentation Policy*

The size of data in our experiments was smaller than what is expected to train deep learning models and hence we relied heavily on data augmentation techniques to counter the effects of small dataset. Image augmentation by geometric and photo-metric transformations is a standard practice to reduce over-fitting leading to a better generalization performance. Not all geometric transformations are appropriate since distortions on the image domain do not always translate meaningfully on the ground truth-depth. Applying a vertical flip to an image capturing an indoor scene may not contribute to the learning of expected statistical properties (e.g. geometry of the floors and ceilings). Therefore, we only consider horizontal flipping of images at a probability of 0.5. Image rotation introduces invalid data for the corresponding ground-truth depth, hence we do not include it. For photo-metric transformations we followed color channel permutations as described by Alhasim et. al. This resulted in increased performance while also

being extremely efficient. We set the probability for this color channel augmentation to 0.5.

D. *Deep Supervision*

Deep Supervision [22] is adding companion objective functions at each hidden layer of a network and then compute the final loss as the output loss plus the sum of the companion losses. This motivates the hidden layers of network to learn independently of the output layer such that the output layer is not tasked with learning the most informative feature transformations. For small training data and relatively shallower networks, deep supervision functions as a strong regularization for classification accuracy and learned features. We have briefly experimented with deep supervision in our project by using outputs from the last three layers of decoder and upsampling each to match the size of original ground truth depth map.

IV. Experimental Results

In this section we describe our experimental results and compare the performance of our network to existing stateof-the-art methods. Furthermore, we perform ablation studies to analyze the influence of the different parts of our proposed method. Finally, we compare our results on a newly proposed dataset of high quality depth maps in order to better test the generalization and robustness of our trained model.

A. Datasets

NYU Depth v2 dataset provides images and depth maps of resolution of 640 × 480 for different indoor scenes captured [18]. The dataset is a collection of 120K training samples and 654 testing samples. We train our method on a subset of 50k images. The depth maps have an upper bound of 10 meters and missing depth values are filled using the inpainting method. Our implementation of "Densenet-169" network produces images that are half the input resolution, i.e. have a resolution of 320 × 240. But for UNet implementation we augment the images and model produces image of resolution 320x320. During training phase in "DenseNet-169", we take the input images at their original resolution and downsample the ground truth depths to 320×240. For our baseline model we downsample both the input and output image. During test time, we compute the depth map prediction of the subset of test image.Sample of dataset shown in Fig. 2.

KITTI dataset is a collection of stereo images and corresponding 3D laser scans of outdoor scenes which are captured with the help of a image capturing equipment mounted on a moving vehicle [19]. The RGB images have a resolution of approximately 1241 × 376 and their respective depth maps are of very low density with lots of missing data. Due to computational limitations associated with processing high resolution RGB images and difficulty in finding out the ground truth depth images, we plan on training out network with this dataset as a part of our future implementation.

Fig. 2. NYU-V2-Depth Dataset. RGB images are the input images, and plasma colored images are ground truth depth images

B. Implementation Details

We implemented our proposed depth estimation network using PyTorch [20] and trained it on one NVIDIA GTX 1060 GPU with 6GB memory. We also used Google Colab's GPU environment for running few experiments.

Our encoder is primarily a DenseNet-169 pretrained on ImageNet. The weights for the decoder are randomly initialized. We use the Adam [21] optimizer with learning rate 0.01 and parameter values $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The batch size is set to 2. The total number of trainable parameters are approximately 45M. Training is performed for 20 epochs on the NYU Depth v2 dataset and takes 4 hours to finish for each experiment.

C. Experiments

We started the experiments with establishing a baseline performance for the task using U-Net [5] which was trained from scratch without pre-trained weights. The intermediate layer outputs from decoder path of the network was analyzed early in the training process to observe if the model was learning meaningful transformations in the layers before the output. This is shown in Fig. 4.

Once this baseline was established we added image augmentations as discussed in previous sections to try and improve the performance of the baseline. We did some experiments with deep supervision in this setting to calculate loss on multiple decoder layers instead of just the output layer.

We built on this architecture by using encoder-decoder networks having ImageNet pretrained state-of-the-art architectures like DenseNet, ResNet etc. as the encoder. We call such encoder networks as backbones. We performed most of our experimentation with DenseNet-169 which closely follows Alhashim et. al [1]. We tried the same image augmentation methods as with original U-Net architecture. We also trained each backbone U-Net with encoder layers in two different settings- encoder freezed (i.e. the pretrained paramaters on

ImageNet encoder are not trainable) and encoder trainable. Training the network in these settings yielded some interesting observations that are discussed in the following section.

D. Performance and observations

We evaluated the model using train-test-validation split by taking first 1000 frames of NYU Depth v2 as training and rest of the data evenly split between testing and validation. This is only a very small subset of the original dataset, having dense ground truth depth maps. The original dataset consisted of around 120K images having sparse depth maps which need to be filled using inpainting techniques. We decided to proceed with only 1449 images because trying to complete sparse depth maps would be a computer vision problem on its own aas well as would make the iterations of training and evaluation considerably slower.

For quantitative analysis we found RMSE to be the widely used error metric as defined below:

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{n}\sum_{p}^{n}(y_p - \hat{y_p})^2} \tag{5}$$

where y_p is a pixel in depth image y, $\hat{y_p}$ is a pixel in the predicted depth image $\hat{y}$, n is the total number of pixels for each depth image.

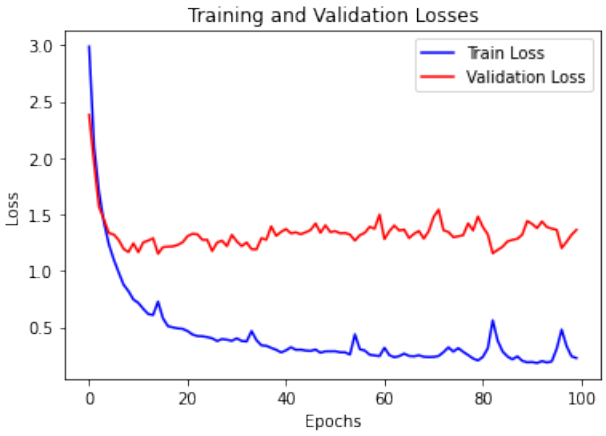

Fig. 3. Loss curves for baseline training

We couldn't get a meaningful quantitative or comparative analysis using RMSE as due to small amount of data and limited iterations of hyperparameters search our model fails to converge and learn as well as expected. We provide the loss curves from one of our training loops in Fig. 3. It's easy to observe how the model fails to generalize after a point and starts overfitting.

We do observe some interesting observations about the model's performance on depth estimation task. One of those was that the model tends to not converge at all, given limited data, if the encoder layers in the backbone network are trainable. On the contrary, the model starts converging if the encoder layers are freezed and only decoder path is trained on the task. We believe that this is because the trainable

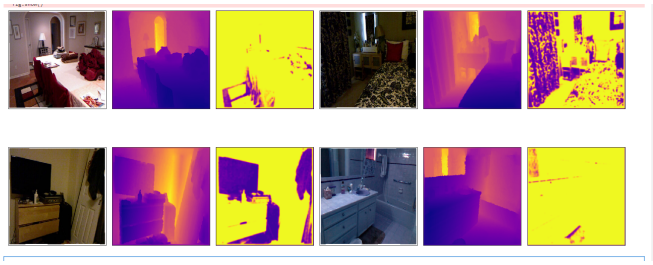

Fig. 4. This figure shows comparisions between input RGB image, ground truth depth map and one of intermediate layer output.

encoder layers do not generalize that well on image to image transformations if not given enough data to learn. On the other hand, once can clearly observe the model starting to learn parameters if the encoder part is freezed with ImageNet weights and only decoder is trainable. Our experiments with data augmentations also proved to be successful in improving the model's convergence and performance in general.

The choice of backbone network also had an impact on the performance of the network but any ImageNet based backbone with similar depth had comparable performances whereas the encoder depth increase led to gains in the performance at the expense of slower training time.

Fig. 5 shows a gallery of depth estimation results that are predicted using our methods along with corresponding ground truth. As can be seen, our approach doesn't produce state-of-the-art results due to sub-optimal model parameters. Yet we can see that our model has started estimating relative depth in most large regions successfully and is also identifying sharp edges where the depth changes arbitrarily.

V. Conclusion

In this paper, we propose to solve monocular depth estimation in a supervised learning problem setup. We loosely follow the convolutional neural network architecture proposed by Alhashim et al. [1] for generating depth map estimation from single RGB images. We leverage pre-trained deep neural network architecture to design an depth prediction models. Through our raw results from experiments, we demonstrate that having a well constructed encoder, with meaningful weights initialization, can partially match the performance of state-of-the-art methods with small training cycle given enough data. Our method neither relies on expensive multi-stage depth estimation networks nor requires designing and combining multiple feature encoding layers. Our method is partially generating depth maps on the NYU Depth v2 dataset. Our aim in the future is to explore the existing deep learning literature for generating higher quality depth maps that capture object boundaries with high accuracy, and we have successfully shown that it's possible using existing architectures. Finding improved data augmentation policies and their probability values for the problem of depth estimation is an interesting topic for future work. Therefore, there are interesting possibilities we plan to pursue for future work. Once we get a combination of network architectures that

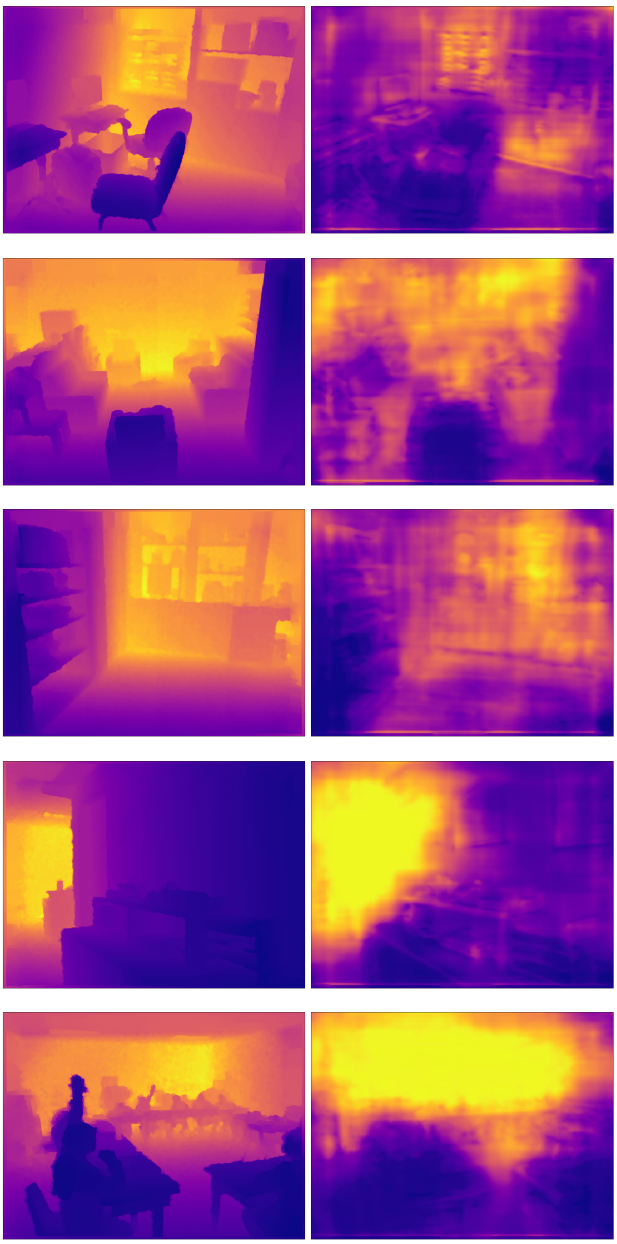

Fig. 5. Depth map ground truth (left) and predictions (right) from DenseNet-169 backbone as encoder.

provide precise depth maps our future scope is to feed live video to generate real-time depth estimation. There are many unexplored questions on limitations of our proposed network and exploring the model's performance through combinations of different encoders, augmentations, and learning strategies that leave the door open for future work in this problem.

ACKNOWLEDGMENT

We would like to express our very great appreciation to Professor Bruce Maxwell for his course lectures, valuable and constructive suggestions during the planning and development of this project work. His willingness to give his time so generously has been very much appreciated.

REFERENCES

[1] Ibraheem Alhashim, and Peter Wonka, "High Quality Monocular Depth Estimation via Transfer Learning",2018
[2] David Eigen, Christian Puhrsch, Rob Fergus, "Depth Map Prediction from a Single Image using a Multi-Scale Deep Network", Jun 2014.
[3] M. P. Lubor Ladicky, Jianbo Shi. Pulling things out of perspective. In CVPR, 2014.
[4] K. Karsch, C. Liu, S. B. Kang, and N. England. Depth extraction from video using nonparametric sampling. In TPAMI, 2014.
[5] Ronneberger, Olaf and Fischer, Philipp and Brox, Thomas,"U-Net: Convolutional Networks for Biomedical Image Segmentation",2015.
[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
[7] R. Memisevic and C. Conrad. Stereopsis via deep learning. In NIPS Workshop on Deep Learning, 2011..
[8] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. Freeman. Sift flow: dense correspondence across difference scenes. 2008
[9] K. Konda and R. Memisevic. Unsupervised learning of depth and motion. In arXiv:1312.3429v2, 2013.
[10] F. H. Sinz, J. Q. Candela, G. H. Bakır, C. E. Rasmussen, and M. O. Franz. Learning depth from stereo. In Pattern Recognition, pages 245–252. Springer, 2004.
[11] K. Yamaguchi, T. Hazan, D. Mcallester, and R. Urtasun. Continuous markov random fields for robust stereo estimation. In arXiv:1204.1393v1, 2012.
[12] "Cnn-slam, keisuke and tombari, federico and laina, iro and navab, nassir," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6243–6252.
[13] K. Yoneda, H. Tehrani, T. Ogawa, N. Hukuyama, and S. Mita, "Lidar scan feature for localization with highly precise 3-d map," in 2014 IEEE Intelligent Vehicles Symposium Proceedings. IEEE, 2014, pp. 1345–1350
[14] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2261–2269, 2017.
[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255, 2009.
[16] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In ICML, 2015.
[17] J. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing, 13:600612, 2004.
[18] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In Computer Vision– ECCV 2012, pages 746–760, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
[19] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. I. J. Robotics Res., 32:1231–1237, 2013.
[20] Paszke, Adam and Gross, Sam and Massa, Francisco and Lerer, Adam and Bradbury, James and Chanan, Gregory and Killeen, Trevor and Lin, Zeming and Gimelshein, Natalia and Antiga, Luca and Desmaison, Alban and Kopf, Andreas and Yang, Edward and DeVito, Zachary and Raison, Martin and Tejani, Alykhan and Chilamkurthy, Sasank and Steiner, Benoit and Fang, Lu and Bai, Junjie and Chintala, Soumith,"PyTorch: An Imperative Style, High-Performance Deep Learning Library", *Advances in Neural Information Processing Systems 32*,2019
[21] Kingma, Diederik P. and Ba, Jimmy,"Adam: A Method for Stochastic Optimization",2014
[22] Lee, Chen-Yu and Xie, Saining and Gallagher, Patrick and Zhang, Zhengyou and Tu, Zhuowen,"Deeply-Supervised Nets",2014