ISSN 2278-3091
Volume 10, No.1, January - February 2021
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse021012021.pdf
https://doi.org/10.30534/ijatcse/2021/021012021

A Comparative Study of Models for Monocular Depth Estimation in 2D Images

Saksham Khatod[1],Aadesh Mallya[2], Rohan Bhardwaj[3], Abhishek Vichare[4]
[1]NMIMS' Mukesh Patel School of Technology Management & Engineering, Mumbai, saksham.khatod92@gmail.com
[2]NMIMS' Mukesh Patel School of Technology Management & Engineering, Mumbai, aadeshmallya99@gmail.com
[3]NMIMS' Mukesh Patel School of Technology Management & Engineering, Mumbai,rohanbhardwaj111@gmail.com
[4]NMIMS' Mukesh Patel School of Technology Management & Engineering, Mumbai,vichare1@gmail.com

ABSTRACT

Monocular depth estimation has been a challenging topic in the field on computer vision. There have been multiple approaches based on stereo and geometrical concepts to try and estimate depth of objects in a two-dimensional field such as that of a plain photograph. While stereo and lidar based approaches have their own merits, there is one issue that seems recurrent in them, the vanishing point problem. An improvised approach to solve this issue involves using deep neural networks to train a model to estimate depth. Even this solution has multiple approaches to it. The general supervised approach, an unsupervised approach (using autoencoders) and a semi-supervised approach (using the concept of transfer learning). This paper presents a comparative account of the three different learning models and their performance evaluation.

Key words: Depth estimation, Autoencoders, Transfer learning, Supervised learning, Unsupervised learning, Semi-Supervised learning.

1. INTRODUCTION

We as humans, have the brain to construct and perceive 3D space around us. The parietal lobe of the brain specifically, is responsible for perceiving space around us, and we understand the distance and the coordinates of an object that we see, through the visual cortex. Monocular Depth Estimation is like providing a machine with a visual cortex and a parietal lobe. It is what combines computer vision with deep learning to reconstruct 3D space from one single 2D glance. In theory, this is much more difficult than what the human brain does. The brain combines images taken from the left eye and the right eye, combines them to locate the actual position of a point in space. Monocular Depth Estimation however involves teaching the computer to estimate the depth of objects in a single 2D image taken from a single lens.

Introduction of cognitive and logical reasoning into a conventional machine can potentially solve many problems by example mapping, like essential attributes recognition, pattern recognition, clustering, classification and predictions. Deep neural networks provide such models. These are essentially mathematically derived equations and functions, but are associated with a memory and correction element to emulate the humane way of understanding the knowledge being fed to us. Neural networks depend on three important factors –(1) property and features of the interconnection between the neural layers, (2) on the application basis (classification models, prediction models, generative models, and optimization models), or (3) on the basis of the learning approach followed (supervised, unsupervised or semi-supervised).

Every single neural network model is unique in nature, and each of them, have their own merits. The vast theoretical and practical essence of deep neural networks have diverse and unending applications to them. Among these, we will be focusing on the regression and estimation tasks. Neural networks can be used for estimation tasks obviously and conveniently.

2. STUDIED ARCHITECTURES

2.1 High Quality Monocular Depth Estimation via Transfer Learning by IbraheemAlhashim and Peter Wonka [1]

Reconstructing a 3D Scene from an image can be done using multiple methods. We can use a stereo pair to estimate the coordinate system. We can also use Hardware Technologies for e.g. LiDAR to measure real-time depth information. One of the more recent approaches to the above problem is to use the power of Convolutional Neural Networks. We've been able to produce reasonable depth maps using nothing but an RGB image and a CNN Model. The Paper employs a quite straightforward architecture that uses encoder-decoder layers with skip connections.

Encoder-Decoder networks are typically employed in image restoration, optical flow estimation and image segmentation tasks. The Idea of Transfer Learning helps the authors make use of these encoders in a different domain than they were originally used in. The Authors mention that the encoders do not end up aggressively reducing the resolution of the input which enables more precise depth maps. Recent methods use Encoder-Decoder networks as a small part of a much larger network. The Authors in this paper are proposing a technique that can generate depth maps that are comparable to those obtained by state-of-the-art architectures using just a simple encoder-decoder architecture.

i) Architecture

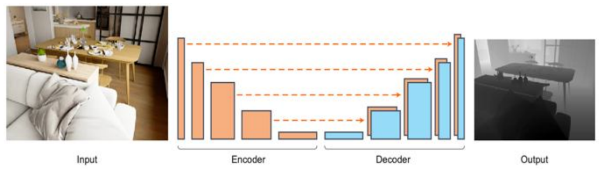


Figure 1: Architecture showing the encoders and the decoders [1]

This Paper makes use of Transfer Learning to apply the DenseNet-169 Model [2] to create a feature vector which can be used later for generating the depth map. DenseNet-169 Model was originally trained on ImageNet [3]. The Authors removed the top layers from the model which were related to ImageNet Classification. This Resultant Vector is then supplied to a consecutive series of up-sampling layers [4] which form the decoder. Each Up-sampling block except the last one is followed by a Rectified Linear Unit Activation Function. The Decoders are used to construct the depth map at half the input resolution.

ii) Loss Function

The Authors also propose a new loss function. The Loss Function calculates the difference between the ground-truth depth map and the prediction of the network [5]. The Loss Function can have a remarkable impact on the training speed and the overall performance of the network. The Authors propose a loss function which while minimizing the depth differences also penalizesmisrepresentations of high frequency details in the image section of the depth map. The Final Loss Function is a weighted sum of three loss functions.

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

The First loss term is the standard point-wise loss. The Second loss term calculates the difference between x and y gradients between the depth maps. The Last term uses the Structural Similarity term [6] which is usually used for image reconstruction tasks. These loss terms increase when the depth values increase. To solve this, the authors consider the reciprocal of the original depth map and multiply it by m(maximum depth in the scene) to generate the target depth map which will be later used as a ground-truth depth map.

iii) Data Augmentation

For data augmentation, the authors consider horizontal flipping to make the algorithm learn features like floors and ceilings. They also consider photo-metric transformations for e.g. interchanging the red and green image matrices in the input which increases performance while still being efficient.

iv) Result

The Algorithm proves to be quite reliable. Even after using a simple architecture, the authors manage to get a Root Mean Square Error of 0.390. The Color Scheme used to train the model is inverse of that of other algorithms mentioned in this paper. However, it is still easy to compare its output to other algorithm's output by approximating the gradient of color changes in the various parts of the image.

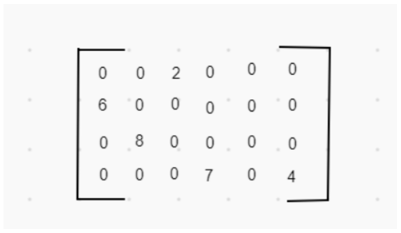
2.2 Visualization of Convolutional Neural Networks for Monocular Depth Estimation [7]

As talked about earlier, the human brain uses many cues to create a sub-conscious depth map around us – bilinear perspective, relative and absolute size estimation (based on previous memories), interposition, gradient, roughness, texture, light and shades etc, to name a few. A question arises at this point; how exactly does a neural network utilize them? Understanding this is of paramount importance to us as the model we design needs to work in every case irrespective of where an image has been clicked or under what conditions.

This paper we referred to, attempts to understand how a convolutional neural network can successfully identify minimum number of relevant pixels from an input frame to estimate depth. The authors have approached it as an optimization problem whereby the challenge was to use minimize computation energy while simultaneously and accurately creating depth maps for the input images. We will be exploring some important concepts used in this paper, the validation of their approach and testing the efficiency of their model.

i) Sparse Matrix

Consider a matrix which represents an image of an object in a dark background (0 indicates a black pixel while any other value >=0 will represent a feature)



$$\begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 0 & 4 \end{bmatrix}$$

Figure 2: Sample Image Matrix of an Object [7]

Clearly, there are multiple black/dead pixels in this [4 X 6] image matrix. These remote black pixels will not contribute in the feature extraction process.

Sending this image into the convolutional neural network would mean wastage of resources, as the image is denoted as NumPy array of the size of the dimension of the image. Passing a 640 x 480 image would take the neural network a humongous amount of time trying to detect the pixels which actually matter.

The authors have therefore decided to take the approach of sparse and dense matrix in this model. The authors thought of creating an array of triplets. Each triplet containing a pointer[row, column] to the non-zero value and the value itself.

Table 1: Sample Sparse Matrix [7]

ROW	COL	VALUE
4	**6**	**5**
0	2	2
1	0	6
2	1	8
3	3	7
3	5	4

As you can see the table 1 above,

1. The first row denotes – the number of rows, the number of columns in the input array and the number of non-zero values
2. The remaining rows contain the position and the value of the non-zero values

This newly-generated matrix, will be passed to the model as the input image and is called the sparse matrix. The matrix which was input earlier is called the dense matrix.

Let us consider another example – let us take two dense matrices :

V1 = [4 , 3, 1, 6]

V2 = [2 , 0, 4, 0]

SPARSE-VECTOR form - (4, [0,2] , [2,4])

The optimization may look pretty miniscule but when taking a 640 X 480 image, these tiny improvements coalesce. Sparse matrix tackles the issue of storage, computing time and disconnections in pixels. The sparse matrix relates every non-zero pixel and thus, makes it easier for the CNN to extract a pattern. The authors have used different sparseness parameters to test the model.

ii) Mask Prediction Network

The model has two parts – A mask prediction network, and a trained depth estimation net which bottlenecks the feature detection pre-trained network with the mask prediction i.e. the array of pixels generated by the mask prediction network and multiplicated element-by-element with the feature detection matrix generated by the pre-trained model.

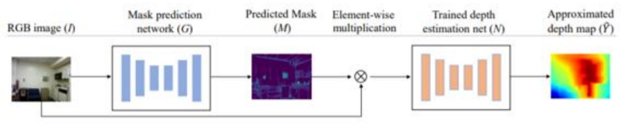

RGB image (I)
Mask prediction network (G)
Predicted Mask (M)
Element-wise multiplication
Trained depth estimation net (N)
Approximated depth map ($\hat{Y}$)

Figure 3: Diagram of the proposed approach in the paper [7]

The model is self-explanatory and straightforward. The entire elaborate architecture has been studied using PyTorchsummary library to summarize the entire model. The highlight of this approach is the mask prediction network.

The mask prediction network (G) - this network represents the unsupervised approach in the architecture. The authors have used an encoder-decoder structure for the same. The encoder is a simple dilated residual network(DRN) which preserves the attributes and local structure of the image due to less down-sampling. This DRN consist of 22 layers pre-trained on ImageNet for feature extraction. It will output a feature map with 512 channels and about 0.125 times the resolution of the input image. The decoder consists of three up-projection blocks which outputs a map with 64 channels and the same dimensions as that of input image, which is followed by a [3 X 3] convolutional layer finally giving us the mask. The encoder-decoder makes the network G, which consists of a total of 23.5 million parameters.

The mask generated in sent into a bottleneck network which applies this mask on the original image. The disparity is calculated and the loss is sent back to the network G to allow the network to backpropagate the information and adjust its weight. The process continues till the perfect(or at least, the models considers it to be) mask is generated. This mask is now sent into a trained depth estimation net, a simple network trained on a supervised set. The idea was simple. Estimating depth from scratch is difficult, but if a pre-calculated mask which has already detected important features of the image and their depth, if passed into the network; the work of the network is much more simplified.

2.3 Unsupervised Monocular Depth Estimation with Left-Right Consistency [8]

The proposed approach works on monocular images, without ground truth depth data. Given an arbitrary point X, the left projection is XL the projection of it on right view is XR. The magnitude of the vector between XL and XR gives us disparity. This is done for every point and a Disparity Image is created. The approach attempts to project corresponding points, given a single part of stereo image i.e. the left image or the right image. The network is trained to account for the losses during image reconstruction, as only reconstruction alone can cause a low-quality depth map. There should be consistency in the disparity produced by right and left image. The authors ensure that their algorithm does the same, providing a better performance when compared to supervised approach which has been trained with ground truth data.

i) Image Reconstruction

The proposed method is image reconstruction. They want to learn a function which can recreate the other part of image when one part of the image is given. In the training phase, the network has access to both of the color images, i.e. left and right. They try to find a function called Dense Correspondence Field, which, when applied to left image, gives them right image and vice versa. The depth estimation is done by using the left and right image, along with their disparities and epipolar geometry constraints. They simultaneously infer both disparities using only left input image. They use backward mapping using a bilinear sampler to predict the image. The following equation helps in obtaining that.

$\tilde{I}^r = I^l(d^r)$.

Here, $\tilde{I}^r$= *right reconstructed color image*

I^l= *left color image*

d^r = *Dense Correspondence Field*

When the distance between cameras is b, the depth can be predicted using the disparity that is obtained above with the equation:-

$\hat{d} = bf/d$

ii) Loss

The overall loss is a sum of all the losses combined. All the losses have a left and right image variant, since during training both left and right color images are used.

$C_s=\alpha_{ap}(C^l_{ap}+C^r_{ap})+\alpha_{ds}(C^l_{ds}+C^r_{ds})+\alpha_{lr}(C^l_{lr}+C^r_{lr})$

C_{ap} is the loss for appearance matching.

C_{ds} is the loss for disparity smoothness

C_{lr} is the left-right disparity consistency loss.

Appearance Matching Loss: The projection of right image that is obtained from applying the Dense Correspondence Field to the left image might not be 100% accurate, for this they need to take into account the loss. The application of this Appearance Matching Loss gives a more accurate result. It compares the Input left image with the obtained output image, which is the right image. Due to the usage of the bilinear sampler, they do not require any simplification or approximation of this cost function.

$$C^l_{ap}=\frac{1}{N}\sum_{i,j}\alpha\frac{1-SSIM\left(I^l_{ij},\tilde{I}^l_{ij}\right)}{2}+(1-\alpha)\left\|I^l_{ij}-\tilde{I}^l_{ij}\right\|$$

Here an L1 penalty and Simplified Single Scale SSIM are used and α=0.85.

Disparity Smoothness Loss: Since a gradient is produced by a depth map with several shades of a color, there can be inaccuracy at the edges where 2 colors meet. This producesdepth an inaccurate result. To compensate for this, the authors introduce a Disparity Smoothness Loss, which takes into account the said smoothness disparities. This weightedfunction, when applied to image, gives a more accurate depth map.

$$C^l_{ds}=\frac{1}{N}\sum_{i,j}\left|\partial x d^l_{ij}\right|e^{-\left\|\partial x I^l_{ij}\right\|}+\left|\partial y d^l_{ij}\right|e^{-\left\|\partial y I^l_{ij}\right\|}$$

Even here, an L1 penalty is applied on disparity gradients ∂d.

Left-Right Disparity Consistency Loss: As the authors only have a single image i.e. the left image as input, and theycreate both the left and right disparities with only one image, they need to be sure that the maps are accurate. They add a penalty for this which tries to make the left disparity equal to the projection formed by the right disparity. When the left disparity, and the projection are same, an accurate result is obtained. They call this the "Left-Right Disparity Consistency Loss".

$$C^l_{lr}=\frac{1}{N}\sum_{i,j}\left|d^l_{ij}-d^r_{ij+d^l_{ij}}\right|$$

iii) Result

The resulting image has an RMS error of 0.209 which is less than many supervised models currently in use. But since this is an unsupervised approach, the complexity and implementation of the code is high. The current model works on individual frames, but they propose a future work which can expand this to video usage as well.

3. COMPARING PERFORMANCE OF DIFFERENT ARCHITECTURES ON SAMPLE IMAGES

Firstly, we need a dark image to challenge the feature extraction capabilities of the architecture. We found this image online. It comprises of bright spots in an otherwise dark sky

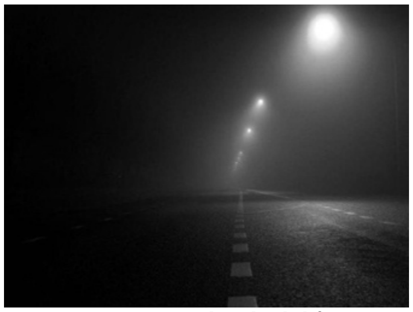


Figure 4: Image downloaded from https://images.unsplash.com/34/WyVMN1W6Tves4NUkaXwh_14.JPG?ixlib=rb-1.2.1&ixid=eyJhcHBfaWQiOjEyMDd9

Secondly, we need an image where the background is dusty making the subject difficult to see. This image is perfect for this scenario.

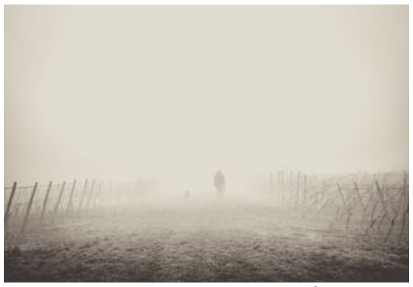


Figure 5: Image downloaded from https://i1.pickpik.com/photos/261/590/374/596cb9d56c5cc-preview.jpg

Thirdly, we need an image where everything is clearly visible and there’s no direct light source making feature extraction difficult. This is the easiest image we’ll feed into the architectures.

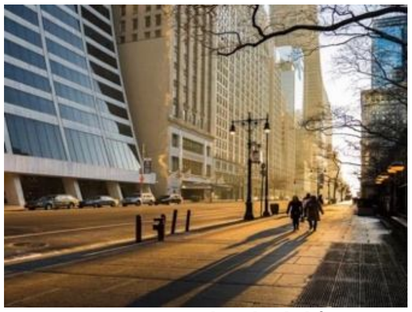


Figure 6:. Image downloaded from https://live.staticflickr.com/7841/46265194244_5eb4f6ed27_b.jpg

Fourthly, we need an image where we do have a direct light source which illuminates the surrounding and also makes the subject clearly visible. The Light source should be coupled with a dark background to make the rest of the image challenging for the architectures.

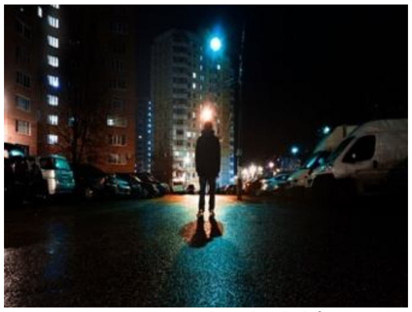


Figure 7: Image downloaded from https://cdn.governmentnews.com.au/wp-content/uploads/2019/03/31100208/iStock-895143304.jpg

Fifthly, we need an image where the majority of the image is bright white. White Images are some of the more difficult ones to predict because they make feature extraction more arduous.

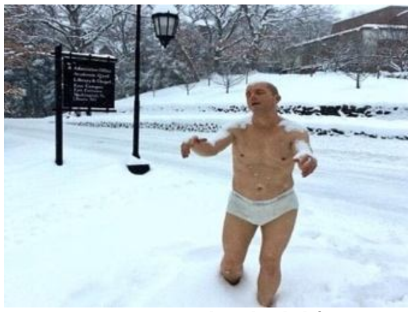
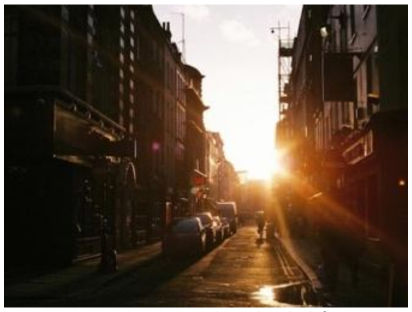


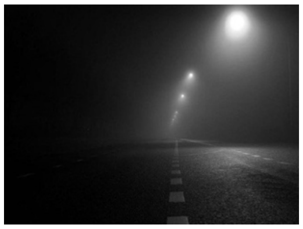
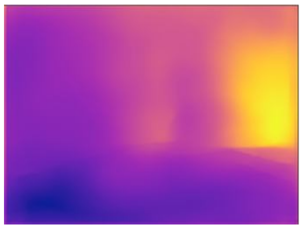
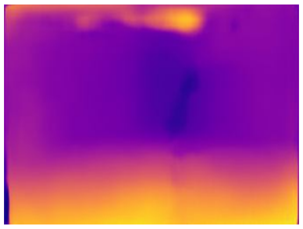
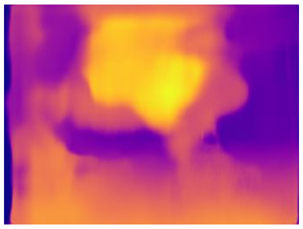
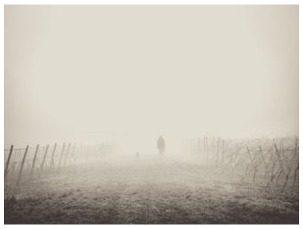
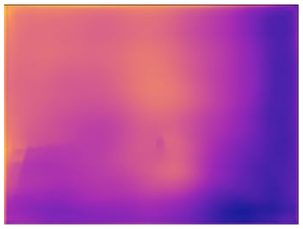
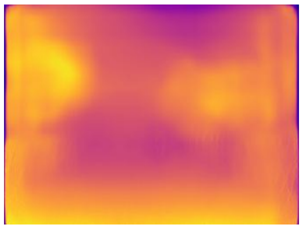
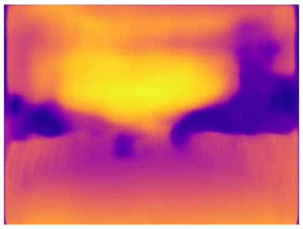
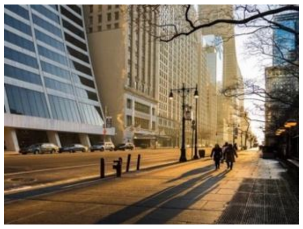
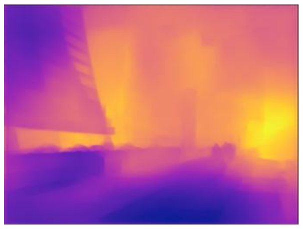
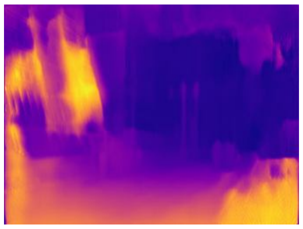
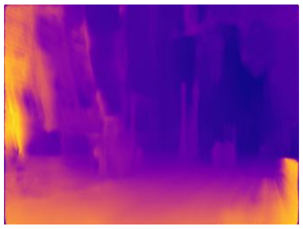
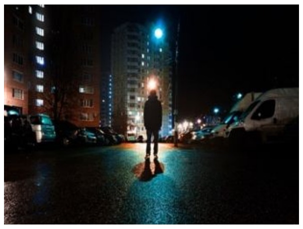
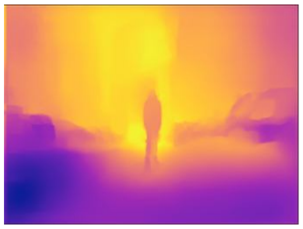
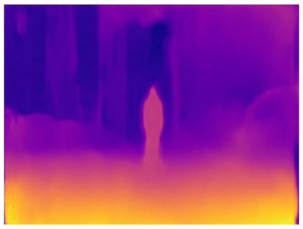
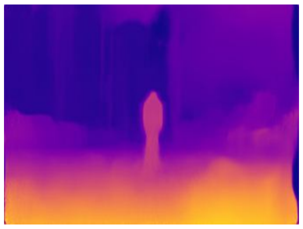
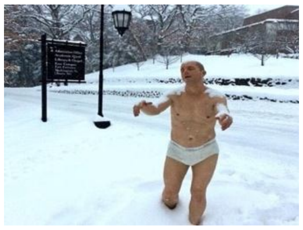
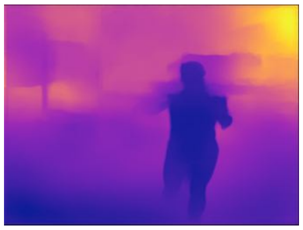
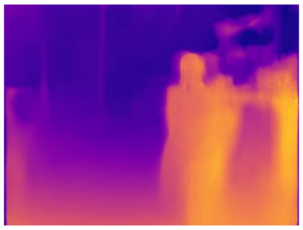
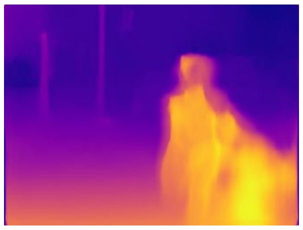
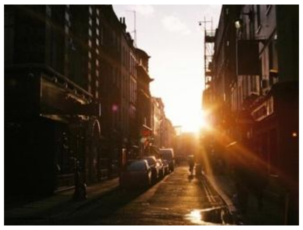
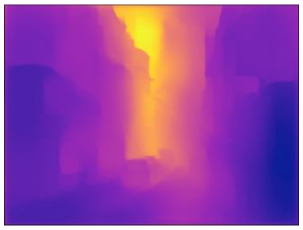
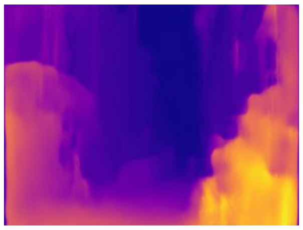
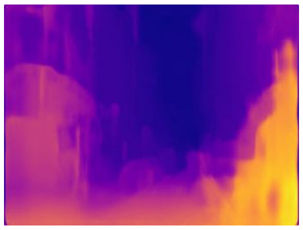
Figure 8: Image downloaded from https://the-hollywood-gossip-res.cloudinary.com/iu/s--Kkkq86rV--/t_full/cs_srgb,f_auto,fl_strip_profile.lossy,q_auto:420/v1391691259/sleepwalker-statue.jpg

Sixthly, we need an image where a bright light source comes straight into the sensor of the camera and makes the nearby area difficult to discern.



Figure\.9:. Image downloaded from https://i.pinimg.com/originals/3c/d6/af/3cd6afd3841e6128ea8bc03e3daa956d.jpg

Table 2: Comparing Depth Maps of Several Architectures

Images	**Semi-Supervised 1**	**Semi-Supervised 2**	**Unsupervised**
			
			
			
			
			
			

4. CONCLUSION

After our extensive study of the three algorithms, we've ended up concluding that we do not require complex methods to generate an relatively accurate depth map. Even models with the simplest of the architectures can output depth maps that can match those produced by state-of-the-art algorithms. All 3 Methods take >30 hours to train on a high-end GPU. For our tests, we've used pre-trained models provided by the authors. Surprisingly, the Unsupervised Approach performed best in our tests. The Semi-supervised approaches also performed relatively well on the test dataset. We think a semi-supervised approach provides a great balance of complexity and ease-of-use.

5. CONFLICT OF INTEREST

On behalf of all authors, the corresponding author states that there is no conflict of interest.

REFERENCES

1. IbraheemAlhashim, Peter Wonka, High Quality Monocular Depth Estimation via Transfer Learning, arXiv:1812.11941
2. G. Huang, Z. Liu, L. van der Maaten, and K. Q. Wein- berger. Densely connected convolutional networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2261–2269, 2017.
3. 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.
4. J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Kar-ras, M. Aittala, and T. Aila. Noise2Noise: Learning image restoration without clean data. In J. Dy and A. Krause, editors, Proceedings of the 35th International Conference on Machine Learning, volume 80 of Proceedings of Machine Learning Research, pages 2965–2974, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR.
5. D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In NIPS,2014.
6. 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, 13:600–612, 2004.
7. Junjie Hu, Yan Zhang, Takayuki Okatani, Visualization of Convolutional Neural Networks for Monocular Depth Estimation, arXiv:1904.03380
8. Clément Godard, Oisin Mac Aodha, Gabriel J. Brostow, Unsupervised Monocular Depth Estimation with Left-Right Consistency, arXiv:1609.03677