Unsupervised Learning of Monocular Depth Estimation

Binoy Thomas
ECE Department
North Carolina State University
Raleigh, US
bthomas7@ncsu.edu

Pranav Jonnalagadda
ECE Department
North Carolina State University
Raleigh, US
pjonnal@ncsu.edu

Sameer Watve
ECE Department
North Carolina State University
Raleigh, US
sswatve@ncsu.edu

***Abstract*—There have been significant research in the field of depth estimation of single images and videos. However most of the existing methodologies approach depth estimation as a supervised learning problem, which require a large ground truth depth map data for training. The main challenges to that is recording high quality depth data in a wide range of environments. We implemented a unsupervised learning technique to tackle the problem of depth estimation. The method we implemented made use of a single image to estimate the depth by calculating the disparity between the left and right poses of the same image.**

Depth estimation finds applications in a large number of applications. It is used for robotics and for visual odometry where we need to predict the movement of a robotic system based of the depth of the image. It is also used in augmented reality to know exactly where objects are in the view of the person.

We show how a convolutional neural network can be used to learn to perform single image depth estimation with no available ground truth depth data. We tried to find disparity between the left and the right poses of an image from the KITTI 2015 dataset. We obtained considerably better results compared to other supervised learning methods. We found out that in order to obtain better results, it is not only important to consider the image reconstruction loss but also the training loss. We were able to achieve very good results by training the system with a large dataset and using a large number of epochs to train the system.

I. Introduction

Conventional displays are two dimensional. A picture or a video of the three dimensional world is encoded to be stored in two dimensions. Needless to say, we lose information corresponding to the third dimension which has depth information. Getting 3D information from 2D images has been a fundamental question bugging researchers since years. Many solutions for obtaining the depth of the image have been published and the results have been really good. Most of these techniques rely on the assumption that many observations of the same image are available. They use supervised learning techniques which use ground truth labels to obtain the depth of an image. The supervised learning methods propose geometric solutions to solve the problem. These solutions are expensive since collecting annotations for the ground truth labels is not easy since we would need expensive laser or depth cameras.

Depth estimation has applications in a large number of robotic navigation fields where it is important for the robot for path planning and to avoid obstacles [8]. Consider an example

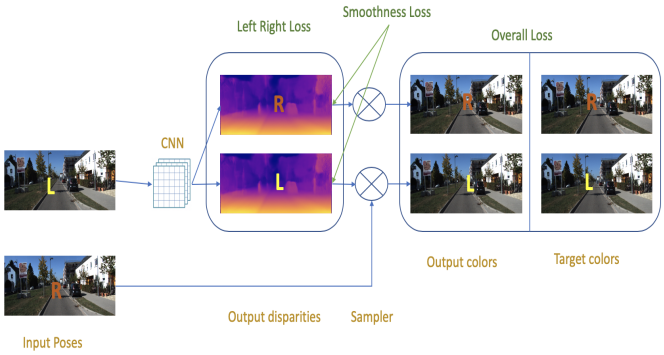

Fig. 1: System Model

for a robotic arm to grasp an object. Depth information would be important to know where the object is. Depth estimation finds applications in the field of augmented reality as well where users can accurately perceive the information around them using depth information. There are many applications such as synthetic object insertion in computer graphics, synthetic depth of field in computational photography , using depth as a cue in human body pose estimation, robot assisted surgery , and automatic 2D to 3D conversion in film . One of the key applications of depth estimation is Augmented Reality (AR). A fundamental problem in AR is to place an object in 3D space such that its orientation, scale and perspective are properly calibrated. Depth information is vital for such processes.

Humans are great at monocular depth estimation. They use cues such as perspective, scaling and they use lighting information to estimate where the object is in the image. They use both top-down and bottom-up cues to link the scene information to estimate depth. Our eyes estimate depth by comparing the image obtained by our left and right eye. The minor displacement between both viewpoints is enough to calculate an approximate depth map. We call the pair of images obtained by our eyes a stereo pair. This, combined with our lens with variable focal length, and general experience of seeing things, allows us to have seamless 3D vision.

There are many hardware approaches to predict depth from

the image such as dual camera technology used in smart phones, LIDAR sensors that provide z coordinate information of the scenes or other expensive sensors. The advent of Neural Networks eliminated need of using such sensors.

One of the important software approach to predict depth uses multiple images of the scene. In this approach we take multiple images of the same scene with slight displacements. By matching key points that are common with each image, we can reconstruct a 3D model of the scene. Algorithms such as Scale-Invariant Feature Transform (SIFT) are excellent at this task.

For single image depth estimation, there are supervised as well as unsupervised methods. The method in this paper uses pose estimation by which it finds the disparity between the left and the right poses of an image to obtain and puts it through a CNN to train the system to get depth information. The Convolution Neural Network implemented in this paper does not require any depth data and is trained to synthesize depth in between. It learns depth information by processing the pixel level information and finding the correspondence between pairs of the Left and Right images which are available in the KITTI2015 dataset.

For the experiments we did not run the model on the whole KITTI dataset. We used 12600 images from the dataset. The model was complex and the it took over 24 hours to train it for 150 epochs. We started experimenting with 20 epochs and slowly increased the number of epochs. Exceptional results were obtained above 100 epochs.

The model creates a dense depth map for the images which are 256*512 images. We ran the model using ResNet18 and ResNet50 and compared the results obtained from both. Our method proposed to create a network architecture that performs end-to-end unsupervised learning quickly. It also evaluates many types of training losses and this explains the effectiveness of our system. It obtains exceptional results on the complex KITTI dataset which contains videos of different types of urban and rural streets. It used novel feature reconstruction loss which improves accuracy if the depth estimation better.

II. Related work

Most of the work related to depth estimation in the field of depth estimation used one of many constraints on the dataset. Hence, they are not robust and tend to become very expensive. Some methods uses stereo matching to obtain pairs of images for estimation. Others used several overlapping images from various viewpoints to establish a ground truth to estimate depth[]. Some other methods used a fixed camera and different lighting schemes to observe the change in the HSV values of the images and use this information to estimate depth. These methods are only applicable when there is a lot of information available on the scene. Hence, our paper is focused on only monocular depth estimation which takes into consideration only a single stereo image and makes no assumption on the scenes geometry or the types of objects present in the image.

A. Supervised methods

Deep learning based depth estimation related work was first published by Eigen et al. [9] which used ConvNets for depth estiamtion. The type of loss used for estimating the accuracy of the depth was the scale-invariant loss. They used multiscale neural networks to estimate the depth. Liu et al [15] viewed depth estimation as a random variable and used a continuous conditional random field learning model to find the depth of the image. Next was the research done by Laina et al. [14] who used a CNN to model a network for mapping between a monocular image and a depth map.

Make3D is a model proposed by Saxena et al. This method uses a method of patch estimation which first converts the images into patches and then estimates the 3D location and orientation of the patches in the image. This method among other methods are inefficient because they have difficulty with thin objects as the patches would not be able to observe these. Another problem with planar based methods is that they lack global references and they make predictions locally. Their results are not very good due to these reasons.

Ladicky et al. used semantics to improve the accuracy of the depth estiamtion. Karsh et al. [13] used another way of predicting the depth in which they made image predictions by copying whole depth images into the testing set. This has the drawback that the whole training set must be available during testing.

All these methods rely on high quality images during training time. It is not only important that these images are high quality but these images must also be highly aligned and they must have good ground truth depth during training time. Our method uses a single image to predict the depth of the image. It uses binocular views of the same image to predict depth of the image.

B. Unsupervised methods

Unsupervised methods rely on learning depth of images by calculating the photometric warp loss. This type of loss replaces the loss based on the ground truth labels. Research into this method was first done by Garg et al. They used binocular stereo pair images to train a network which minimized the photometric losses by reducing the difference between the left and the right images viewpoint using the prediction of the depth. Godard et al. used the left-right consistency to improve the accuracy of the depth estimation. This method significantly improved the estimation of depth and helped make significant progress in the field of unsupervised learning for estimating depth. Flynn et al. used a method of estimating the depth by creating new views of the same image by selecting pixels from nearby pixels.

The next major research in unsupervised learning methods was done by Xie et al. [9] They created a model known as Deep3D. This method tries to create the right view images from the left view images to emulate binocular images. They took into consideration the disparities between the left and teh right images to obtain results which reduce the image reconstruction loss. This method looks at every pixel and its

corresponding pixels and stores the disparity values for each of them. Hence, the memory consumption of this method is very high and using this method was not practical. This method is very close to the results of Garg et al. [11] who created a supervised learning method for monocular depth estimation to reduce the reconstruction losses.

In our project, we have used a fully convolutional neural network which is completely unsupervised in a way that there is no ground truth labels required to generate appropriate results. But we have used two views i.e. poses to generate a binocular view of a single image. This method is monocular because it just finds disparities between the left and the right view of the same image. The aim of this method is to reduce the photometric loss but this can affect the depth estimation.

III. Methodology

This section describes our depth prediction network. We introduce a depth estimation training loss which does a left-right consistency check and enables us to train on image pairs without requirement of ground truth depth data.

A. Dataset

KITTI dataset is acquired using a standard station wagon with two high-resolution color and grayscale video cameras. The dataset consists of video sequences in various environments like city, residental areas, roads, campus etc. Every image data in KITTI consists of both left and right view images of a particular object. There are over 60 different sub classes of data available. The raw dataset is about 175GB. We have used this to train, validate and test our unsupervised depth network.

B. Depth Estimation

For any given image *I*, our goal is to learn a function f that can predict scene depth, $\hat{d} = f(I)$. Most of the supervised learning methods have input images and their corresponding depth values at training. We do a pose depth estimation as an image reconstruction problem during training.The technique is, Given a calibrated (left-right) pair of monocular camera images of a scene, we try to learn a function which can reconstruct one image from the other that means we have learned something about the 3D shape of the scene. At training time, We have access to two images $I^l$and I^r, which are the corresponding left and right view images of a scene, captured at the same moment in time. Instead of predicting the depth of the scene, we find the dense correspondence d^T, which when applied on the left image will help us to reconstruct the right image. Lets refer to the reconstructed right image as $\tilde{I}^r$. Similarly we repeat the process by inputting the right image and estimating the left image $\tilde{I}^l = I^r(d^l)$.

d corresponds to image disparity- a scalar value, which helps us in calculating the depth. Let b be the baseline distance between the two cameras and f be the focal length of the camera, The relationship between depth $\hat{d}$ and disparity is given by $\hat{d} = bf/d$.

C. Depth Network

Our Network calculates the depth of a scene by inferring the disparities that help us reconstruct the right image from the left image and vice versa. We can simultaneously infer both disparities (left-right and right-left) using only the left image as input and try to obtain better depths by enforcing them to be consistent with one another.

We generate a fully differential image using a bilinear sampler. The figure below shows learning to generate right image by sampling the left image, will produce disparities aligned with our target right image. We want our output disparity to map with our input left image, which means that the network has to sample from the right image as well. We could instead use our network to generate left view by sampling the right view.

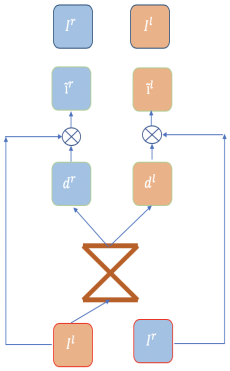

Fig. 2: Sampling Strategies for backward mapping

This generates a left view aligned disparity map. This always does not give us the proper exact depth map prediction. So we solve the problem by training the network to predict disparity maps for both left and right views by sampling from opposite input images. Here left image is still used as the input image, while right image is also used as an input **only during training**. By doing this, we enforce consistency between both disparity maps using left-right consistency which leads to better accurate results compared to only inputting either the left or the right view images during training. Our Convolutional network architecture is inspired from DispNet [2], but has a lot of important modifications which help us train the model without the ground truth depth data. Our network consists of an encoder and a decoder. The decoder used skip connections [3] from the encoder's activation network which helps it to resolve higher resolution details. We output disparity predictions at four different scales, Our Network takes a single image as input and predicts two disparity maps at each output scale (left-to-right and right-to-left).

The table 1 and 2 below shows our model architecture

D. Loss Calculations

We define a loss C_s for each output scale s, leading to a total loss as a sum $C = \sum_{s=1}^{4} C_s$. The loss module computes

layer	k	s	channels	in	out	input
conv1	7	2	3/32	1	2	left
conv1b	7	1	32/32	2	2	conv1
conv2	5	2	32/64	2	4	conv1b
conv2b	5	1	64/64	4	4	conv2
conv3	3	2	64/128	4	8	conv2b
conv3b	3	1	128/128	8	*	conv2
conv4	3	2	128/256	8	16	conv3b
conv4b	3	1	256/256	16	16	conv4
conv5	3	2	256/512	16	32	conv4b
conv5b	3	1	512/512	32	32	conv5
conv6	3	2	512/512	32	64	conv5b
conv6b	3	1	512/512	64	64	conv6
conv7	3	2	512/512	64	128	conv6b
conv7b	3	1	512/512	128	128	conv7

TABLE I: Encoder

layer	k	s	channels	in	out	input
upconv7	3	2	512/512	128	64	conv7b
iconv7	3	1	1024/512	64	64	upconv7b+conv6b
upconv6	3	2	512/512	64	32	iconv7
iconv6	3	1	1024/512	32	32	upconv6+conv5b
upconv5	3	2	512/256	32	16	iconv6
iconv5	3	1	512/256	16	16	upconv5+conv4b
upconv4	3	2	256/128	16	8	iconv5
iconv4	3	1	128/128	8	8	upconv4+conv3b
disp4	3	1	128/2	8	8	iconv4
upconv3	3	2	128/64	8	4	iconv4
iconv3	3	1	130/64	4	4	upconv3+conv2b+disp4
disp3	3	1	64/2	4	4	iconv3
upconv2	3	2	64/32	4	2	iconv3
iconv2	3	1	66/32	2	2	upconv2+conv1b+disp3
disp2	3	1	32/2	2	2	iconv2
upconv1	3	2	32/16	2	1	iconv2
iconv1	3	1	18/16	1	1	upconv1+disp2
disp1	3	1	16/2	1	1	iconv1

TABLE II: Decoder

C_s as a combination of three terms,

$$C_s = \alpha_{ap}(C^l_{ap}+C^r_{ap})+\alpha_{ds}(C^l_{ds}+C^r_{ds})+\alpha_{lr}(C^l_{lr}+C^r_{lr}) \quad (1)$$

where C_{ap} brings the reconstructed image similar to the corresponding training input, C_{ds} shows smooth disparities and C_{lr} shows the consistency between the predicted left and right images. All these parameters contain both left and right image variant, only the left image component is fed through the conv layers.

We also calculate **Appearence Matching Loss**, While training, the network learns to generate a image from sampling pixels from the opposite images (for the left image, the right one and for the right image, left one). Our depth image formation model uses a bilinear sampler which is sampler from the spatial transformer network (STN) [4] to sample the input image with the disparity map. STN uses bilinear sampling where the output image pixel is the weighted sum of 4 input image pixels. In some other papers [5], [6] they use the bilinear sampler to locally fully diferentiable and intergrates into the convolutional architecture.

We use a combination of L1 and single scale SSIM [7] term as our photometric image reconstruction cost C_{ap}, This compares the input image I^l_{ij} and its reconstruction $\tilde{I}^l_{ij}$, N is the number of pixels.

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

We use a simplified version of SSIM, 3x3 block filter instead of gaussian and $\alpha = 0.85$.

Smoothness Disparity loss We want the disparities to be L1 smooth because as depth disparities occur quite often at image gradients, so we weigh this cost with an aware term using image gradients ∂I.

$$C^l_{ds} = \sum_{ij} |\partial_x d^l_{ij}| e^{-||\partial_x I^l_{ij}||} + |\partial_y d^l_{ij}| e^{-||\partial_y I^l_{ij}||} \quad (3)$$

Left Right Disparity Consistency Loss Inorder to produce more accurate disparity maps, we train our model to predict both left and right disparity maps. To ensure that both maps are coherent, we introduc L1 left-right disparity penalty. This makes sure that our left-view disparity matches our projected right-view disparity map.

$$C^l_{lr} = \frac{1}{N} \sum_{ij} |d^l_{ij} - d^r_{ij+d^l_{ij}}| \quad (4)$$

All the other terms in equation (1) are obtained by mirroring for the right-viewed disparity maps and evaluated on all 4 output scales.

During testing, Our model predicts the depth map for the input left image, has the same resolution as the input images. Using the camera focal length and baseline distance from the training images, we then convert the disparity map into final depth map.

IV. Results

A. Model training

The Pytorch platform is used to implement this project. The training set consisted 12 video sets from KITTI, in total 12600 left and corresponding right view RGB images. The validation set has one video set and 800 left right images. The GPU used for training on models is as per following details: Windows 10, 64 bit, Intel 2.20GHZ Xeon processor, 128GB RAM
We used two versions of models for depth estimation.
1. ResNet50 as encoder and similar decoder
2. ResNet18 encoder and similar decoder
The second version was having less trainable parameters and producing results as good as first.
Down sampling of the frames have been used to accelerate training and testing. In the kitti dataset each frame size is of the resolution 1392 x 512. Down sampled input frame size is also treated as hyperparameter.
Grid search method used for hyperparameter selection. The following set of hyperparameters were used:

The following is the best set of hyperparameters we used at the end to generate results from test set:
Learning rate: $1e-4$
Batch size: 8

Hyperparameter	A	B	C
Learning rate	1e-3	1e-4	1e-5
Batch size	8	12	16
Epochs	50	150	200
Input size	1024 x 512	512 x 256	256 x 256

TABLE III: Hyperparameter Selection

Epochs: 200
Input frame size: 512 x 256
The plots for training and validation accuracy vs number of epochs which is used for the grid search method are as follows:

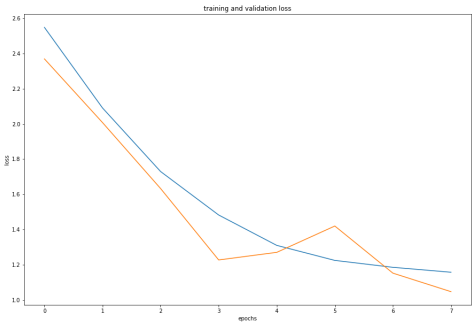

Fig. 3: batch size: 8, learning Rate: 1e-4, epochs: 8

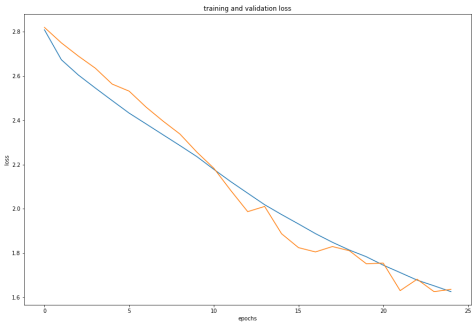

Fig. 4: batch size: 8, learning Rate: 1e-4, epochs: 25

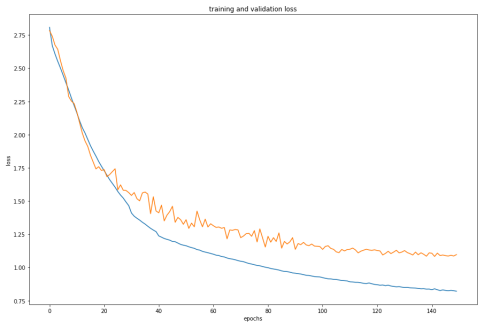

Fig. 5: batch size: 8, learning Rate: 1e-4, epochs: 150

B. Model testing

The test set has one video set from kitti consisting 40 left images. The predicted depth maps and corresponding input image are as shown below:

Results with ResNet18 model trained for 50 epochs:

Fig. 6: input image 1

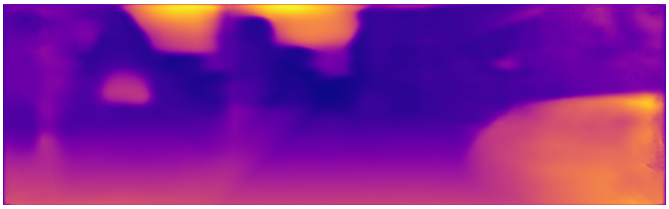

Fig. 7: depth map 1

We observed that as we increase epochs results are fine tuned. Results with ResNet18 trained for 150 epochs:

Fig. 8: input image 1

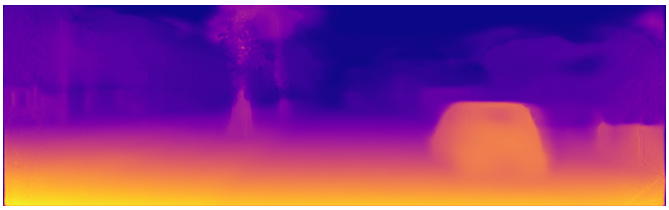

Fig. 9: depth map 1

Fig. 10: input image 2

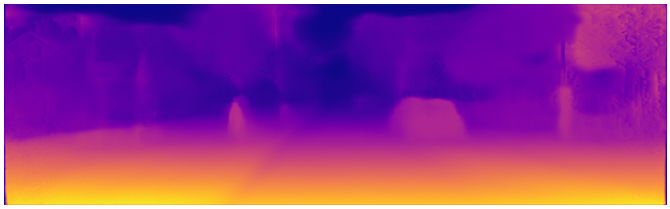

Fig. 11: depth map 2

Fig. 12: input image 3

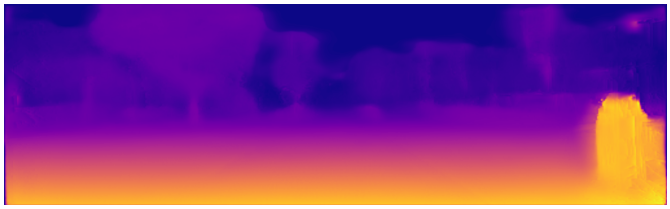

Fig. 13: depth map 3

V. Conclusion

Depth Estimation is a challenging problem with numerous applications. Autoencoders are among the simplest type of networks used to extract depth information. The novel loss function in this approach enforces consistency between the predicted depth maps from each camera view during training, improving predictions. We trained model presented here only on half of the kitti dataset and still it performs better than most of the supervised methods that require for vast amounts of annotated training data. We found that the ResNet18 encoder decoder network is easy to train compared to ResNet50 and predicted depth maps are similar by both.

References

[1] Scharstein, D. Szeliski, R. International Journal of Computer Vision (2002) 47: 7. https://doi.org/10.1023/A:1014573219977

[2] Mayer, E. Ilg, P. Hausser, 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 CVPR,2016. 2, 3, 4, 5.

[3] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. PAMI, 2016. 2, 4.

[4] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In NIPS, 2015. 3, 4.

[5] R. Garg, V. Kumar BG, and I. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In ECCV,2016. 1, 3, 4, 6, 7.

[6] J. Xie, R. Girshick, and A. Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In ECCV, 2016. 1, 2, 3, 4, 5, 6, 8.

[7] Z.Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. Transactions on Image Processing, 2004. 4.

[8] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In ICCV, 2015.

[9] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. Deepstereo: Learning to predict new views from the worlds imagery. In CVPR, 2016

[10] R. Garg, V. Kumar BG, and I. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In ECCV, 2016

[11] C. Godard, P. Hedman, W. Li, and G. J. Brostow. Multi-view reconstruction of highly specular surfaces in uncontrolled environments. In 3DV, 2015.

[12] K. Karsch, K. Sunkavalli, S. Hadap, N. Carr, H. Jin, R. Fonte, M. Sittig, and D. Forsyth. Automatic scene inference for 3d object compositing. TOG, 2014

[13] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In 3DV, 2016.

[14] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields. PAMI, 2015

[15] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. Distill, 2016.