UAV Depth Perception from Visual, Images using a Deep Convolutional Neural Network

Kyle Julian
Stanford University
476 Lomita Mall
kjulian3@stanford.edu

John Mern
Stanford University
476 Lomita Mall
jmern91@stanford.edu

Rachael Tompa
Stanford University
476 Lomita Mall
rtompa2@stanford.edu

Abstract

Recent proliferation of Unmanned Aerial Vehicles (UAVs) into the commercial space has been accompanied by a similar growth in aerial imagery . While useful in many applications, the utility of this visual data is limited in comparison with the total range of desired airborne missions. In this work, we extract depth of field information from monocular images from UAV on-board cameras using a single frame of data per-mapping. Several methods have been previously used with varying degrees of success for similar spatial inferencing tasks, however we sought to take a different approach by framing this as an augmented style-transfer problem.

In this work, we sought to adapt two of the state-of-the-art style transfer methods to the problem of depth mapping. The first method adapted was based on the unsupervised Pix2Pix approach. The second was developed using a cyclic generative adversarial network (cycle GAN). In addition to these two approaches, we also implemented a baseline algorithm previously used for depth map extraction on indoor scenes, the multi-scale deep network. Using the insights gained from these implementations, we then developed a new methodology to overcome the shortcomings observed that was inspired by recent work in perceptual feature-based style transfer. These networks were trained on matched UAV perspective visual image, depth-map pairs generated using Microsoft's AirSim high-fidelity UAV simulation engine and environment. The performance of each network was tested using a reserved test set at the end of training and the effectiveness evaluated using against three metrics. While our new network was not able to outperform any of the other approaches but cycle GANs, we believe that the intuition behind the approach was demonstrated to be valid and that it may be successfully refined with future work.

1. Introduction

Recent advancements in Unmanned Aerial Vehicle (UAV) technology have led to a proliferation of vehicles for hobbyist and low-end commercial use. While these vehicles are able to gather image data with relatively inexpensive on-board cameras, the high cost of specialized sensors limits the range of missions they can effectively conduct. Extracting and inferring data from the available visual imagery provides an effective method to provide some of those "high-end" functions at a reasonable cost. We will address one such area of interest by extracting relative distance or "depth" in the field of view to enable 3D scene mapping inexpensively and reliably. This data may then be used for a variety of purposes both on-board, such as augmented simultaneous localization and mapping (SLAM) and obstacle avoidance, and off-board such as 3D map construction.

2. Related Work

Prior attempts to solve this problem have employed a wide variety of approaches. Several successful approaches take advantage of the added information of stereo imagery [1, 2]. Other techniques much like our own leverage single images. Some researchers have used multi-scale Markov Random Fields to model the relationship between features and depth [3]. Others have used image classification to make course depth map images [4]. Previous work has used segmentation to aid their depth map creation [5] or vanishing lines/points [6]. A recent approach explores training deep convolution neural networks for depth estimation [7, 8]. While this approach, the multi-scale deep network, was reported to be successful, the dataset on which it was trained and evaluated, being exclusively indoor images, is considerably different than the dataset considered here. It was found that this approach did not perform as well on our dataset, likely due to the less orderly geometry present in the outdoor scenes. Another approach uses residual networks to learn a feature mapped up-sampled from the original scene along with a reverse Huber loss, and promising

results show that little training data is required to achieve good performance [9]. Our use of style-transfer as a mapping approach draws on previous work in the field, specifically adapting [10, 11]. Insights critical to our novel style-transfer CNN approach were gained from work on perceptual feature-based transfer [12] An unsupervised approach based on feature-loss was proposed by [13]. Because it is unsupervised, this approach suffers from reduced as the accuracy loss was not trained on. Neural network based methods have also been employed for multi-image based estimation, specifically stereo-camera images [14, 15]. While benefiting from image disparity, these approaches require more expensive cameras to function.

3. Technical Approach

Our work was begun by exploring the effectiveness of existing style transfer methods in depth map generation, then adapting these methods to improve their performance. In addition, we implemented an existing deep CNN-based method for depth map extraction as a baseline. Finally, we developed a novel approach utilizing feature-based transfer. In total we implemented four techniques able to extract depth with varying success: Pix2Pix, Cycle GAN, multi-scale deep network, and the novel style transfer CNN.

3.1. Dataset

We trained all of the networks on simulated data gathered from Microsoft's AirSim, a sophisticated UAV simulation environment specifically designed to generate UAV images for use in deep learning [16]. Microsoft's AirSim is a hardware in the loop simulator that produces raw images and depth maps from a simulated photo-realistic environments. We used a neighborhood environment to gather our training data. A sample of the simulation setup and collected images are shown in Figure 1. The simulator was flown to collect 1,963 pairs of images. The pairs were randomized to break temporal similarities and divided into a train, validation, and test set. 70% is used for training, 20% are used for validation, and 10% are used for testing.

3.2. Pix2Pix

A growing approach for image-to-image translation problems is to use Generative Additive Networks (GANs) which attempts to create output that is indistinguishable from reality. Pix2Pix is a conditional GAN that is not application specific. The network is conditional because instead learning the mapping from a random noise vector Z to an output image Y it learns the mapping from input image X and random noise vector Z, to an output image Y. This added input is what enables the algorithm to be successful at image-to-image translation [10]. This network is trained

Figure 1: Sample from AirSim

with the loss function

$$\arg\min_G \max_D L_{cGAN}(G, D) + \lambda_{L1}(G) \tag{1}$$

where G is the generator, D is the discriminator, λ_{L1} is a hyperparameter, and L_{cGAN} is

$$\mathbb{E}_{x,y\sim p_{data(x,y)}}[logD(x,y)]+ \\ \mathbb{E}_{x\sim p_{data(x),z\sim p_z(z)}}[log(1-D(x,G(x,z)))] \tag{2}$$

3.3. CycleGAN

An extension to Pix2Pix is CycleGAN which not only learns to map input image X to an output image Y but ensures that the output image Y can be used to recreate input image X. CycleGAN requires two sets of images with different characteristics and does not require paired images. If the training data is not provided in pairs, there is ambiguity in the objective function which is mediated by extending the translation problem to be cyclic. For instance, if we take the input and output from before, X and Y, the objective function is augmented with the cycle consistency loss based on the results of $X \rightarrow Y \rightarrow X$. Therefore, this network is trained with the loss function

$$L_{GAN}(G, D_Y, X, Y)+L_{GAN}(F, D_X, Y, X)+\lambda L_{cyc}(G, F) \tag{3}$$

where λ is a hyperparameter, $L_{GAN}(G, D_Y, X, Y)$ is

$$\mathbb{E}_{y\sim p_{data(y)}}[logD_Y(y)]+ \\ \mathbb{E}_{x\sim p_{data(x)}}[log(1-D_Y(G(x)))] \tag{4}$$

$L_{GAN}(F, D_X, Y, X)$ is

$$\mathbb{E}_{x\sim p_{data(x)}}[logD_Y(x)]+ \\ \mathbb{E}_{y\sim p_{data(y)}}[log(1-D_Y(F(y)))] \tag{5}$$

and $L_{cyc}(G, F)$ is

$$\mathbb{E}_{x\sim p_{data(x)}}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y\sim p_{data(y)}}[\|G(F(y)) - y\|_1] \quad (6)$$

The repository for both networks describe above is: https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix.

3.4. Multi-Scale Deep Network

The multi-scale deep network consists of convolutional layers and fully connected layers to produce both a coarse depth map and a refined depth map [8]. The network architecture contains two separate branches. The first branch takes the input image and applies a series of convolutional layers and 2×2 max pooling layers. After five convolutional layers, the output is flattened and fed into two more fully connected layers. The output of this branch is reshaped to be an image of size roughly a quarter of the height and width of the original image. This network uses the max pooling and smaller spatial size to produce a more global approximation of the depth map during training.

The second branch of the network takes the original scene image and applies 63 9×9 convolutional filters with stride 2 and 2×2 max-pooling layers. The output of this layer is the same size spatially as the output image of the first branch, so the output of the first layer is concatenated to produce a depth 64 tensor. The second branch finishes with two more 5×5 convolutional layers before producing the final depth image. This approach combines the coarse depth map with the features from the original image, allowing the network to produce more detailed depths maps by the end of the second branch.

The loss function for this network is based on the difference in the depth values logarithm, which makes the loss invariant to the scale of the depth [17]. Given true depth y_i and estimated depth $\hat{y_i}$, for $i \in \{1, ..., n\}$, where n is the number of pixels, the loss is defined as

$$d_i = \log(y_i) - \log(\hat{y_i}) \quad (7)$$

$$L(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n} d_i^2 - \frac{\lambda}{n^2}\big(\sum_{i=1}^{n} d_i\big)^2 \quad (8)$$

Using logarithms can have instabilities if the input value is zero, and our data set has pixel values of zero. To overcome this challenge, we added a small epsilon to the depth values before taking the logarithm and computing the loss. In addition, we used another loss function based on $d_i = y_i - \hat{y_i}$ to explore how removing the logarithm affects performance, even though the loss is no longer scale invariant. Furthermore, the dataset is relatively small for supervised learning, so we added more dropout after the coarse fully connected layer to try to minimize over-fitting.

3.5. Style Transfer CNN

In addition to the approaches described above, we also developed a new approach to solve the depth mapping problem. This method takes its inspiration from recent work in CNN based style transfer [12]. The main insight we drew on was that the network should not be trained on pixel-to-pixel L2 loss alone, as this typically results in overly blurry images, but should instead be set up in such a way as to ensure the perceptual features of interest in the image are retained. In order to identify these features, we first trained an auto-encoder network on the depth maps that reduces the representations to minimum size at the middle layer. These minimal features are taken to represent the critical features to retain. The intuition is that these features contain all the information necessary for reconstruction of the depth maps. The auto-encoder network uses multiple stride-2 convolution layers to reduce the 2D size of the image while increasing the filter depth. After the encoder layers, the decoder up-sizes the image via transpose convolution. Initially, the decoders up-sized directly to the original image size. This, however, resulted in significant "checker-boarding" in the final image. This issue was solved by up-converting to an image twice the 2D dimension of the original and then performing a stride-2 convolution to produce the final output image. Since the output image is grayscale, the network output only had a single channel, which was duplicated three times for conversion back to an RGB image.

The auto-encoder was initially trained only on L2 loss with respect to the original depth map image. While the auto-encoder was able to effectively reconstruct the image with little loss after this training, the overall performance of the transfer network was found to be poor. It was discovered that the transfer network was not able to recover the same features from the RGB images as it was from the depth maps. The auto-encoder was then re-trained using the composite loss function

$$L_T = L_{L2} + \lambda_a L_{\text{features}} \quad (9)$$

where λ_a is a hyperparameter, L_{L2} is the L2 loss, and L_{features} is

$$L2(f(X) - f(Y)) \quad (10)$$

where $f(X)$ and $f(Y)$ are the features encoded from the visual images and depth maps respectively. This significantly helped the performance of the transfer network, though additional performance gains may be realized through tailoring of the network design and tuning of the training parameters. Once the auto-encoder is trained, a second network with an identical architecture to the auto-encoder network is setup as the transfer network. The transfer network decoder layers are initialized with the weights from the auto-encoder decoder network. The transfer network is then trained with the visual scenes as inputs and the depth maps as targets.

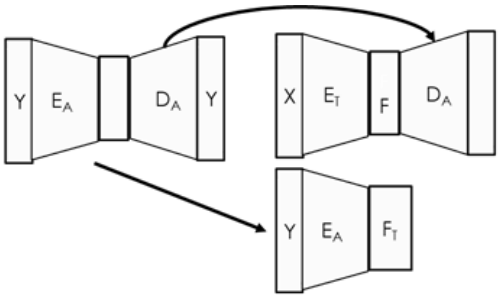


Figure 2: Style Transfer CNN Architecture

The loss function is a composite loss function

$$L_i = L_{L2} + \lambda * L_f \tag{11}$$

where λ is a hyperparameter, L_{L2} is

$$L2(Y_{\text{est}} - Y) \tag{12}$$

and L_e is

$$L2(features_{\text{est}}(X) - features(Y)) \tag{13}$$

4. Evaluation

After each network is trained, we pass the test set through each network and evaluate the output with average absolute error (AE), mean square error (MSE), and root mean squared logarithmic error (RMSLE). The AE quantifies the average pixel error, while MSE gives greater penalty for large errors. RMSLE metric is scale invariant since the logarithmic difference is equivalent to the log of the ratio. As a result, RMSLE penalizes the ratio between the true and predicted depth values, so small values are penalized more for the same errors as pixels with larger values. We want this metric to penalize errors where objects are nearby, since it is more important to get the near-field depths correct than those very far away. All of the errors are an average over the individual pixels of the real and network generated depth images, labeled d_{real} and d_{network} respectively. Let n be the total number of pixels in the test set, and let $d^{(i)}_{\text{real}}$ and $d^{(i)}_{\text{network}}$ be the i^{th} pixel of the real and network generated depth maps with $i \in 1, ..., n$ The AE is defined as

$$\frac{1}{n}\sum_{i=1}^{n}\left|d^{(i)}_{\text{real}} - d^{(i)}_{\text{network}}\right| \tag{14}$$

The MSE is defined as

$$\frac{1}{n}\sum_{i=1}^{n}\left(d^{(i)}_{\text{real}} - d^{(i)}_{\text{network}}\right)^2 \tag{15}$$

The RMSLE is defined as

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\log(256 - d^{(i)}_{\text{real}}) - \log(256 - d^{(i)}_{\text{network}})\right)^2} \tag{16}$$

Table 1: Results of each network.

Network	AE	MSE	RMSLE
Pix2Pix	15.7	676.5	0.357
CycleGAN	22.6	1165.7	0.348
Multi-Scale Deep Network	15.9	659.9	0.399
Style Transfer CNN	22.4	1080.5	0.434

Table 2: Results of CycleGAN weight sweep.

$X \to Y \to X$ Weight A	$Y \to X \to Y$ Weight B	AE	MSE	RMSLE
1	1	37.6	2219.6	0.484
5	5	23.7	1310.4	0.489
10	**10**	**22.6**	**1165.7**	**0.348**
20	20	23.9	1315.7	0.450
5	10	24.4	1367.5	0.450
1	5	28.1	1660.6	0.479
5	1	26.8	1301.3	0.450

5. Results

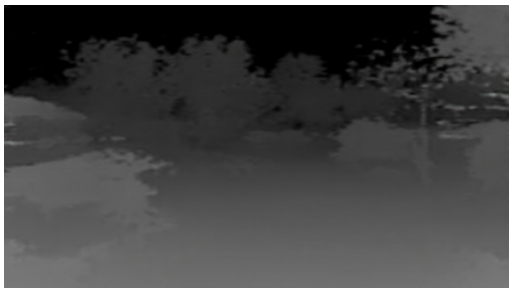
The final training approach used for each network is briefly described below. The comparative results of the networks' performance on the test set is then shown in terms of the evaluation metrics previously defined. A summary of these results are presented in Table 1 and the depth map from each network for the scene in Figure 1 is shown in Figure 3. This image was selected because it contained many of the most common features representative of the suburban environment - road, structure, trees, and other vegetation.

Pix2Pix was run with no changes from the original paper [10] and proved to be a well preforming network for this problem. This network provided the results with the minimum AE and near minimum MSE and RSMLE.

CycleGAN was initially run with no changes from the original paper [11] and preformed the best on RMSLE but it was expected that it could also preform better on AE and MSE. It was hypothesized that putting too high of a weight on being able to cycle the image could potentially cause the network to learn incorrect features so a weight sweep was conducted. Weights and results from this sweep are presented in Section 5 where the bold results are the CycleGAN defualts and depth maps from the various networks are shown in Section 5. Originally, it was anticipated that lowering both weights would improve performance but the results show that the default values actually preform the best.

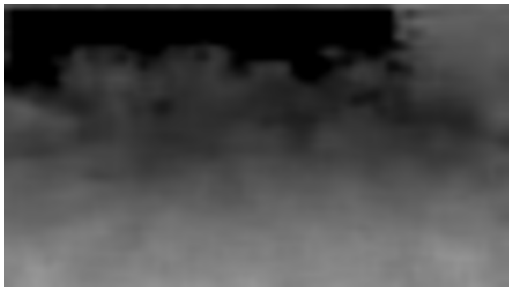
For the multi-scale deep network, the loss function was

(a) Actual depth map

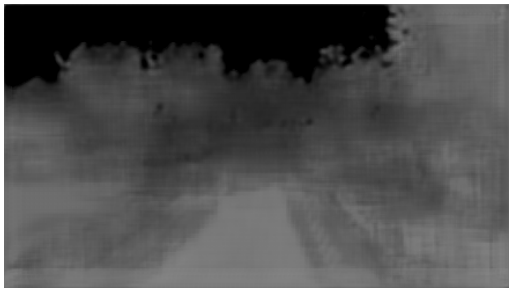


(b) Pix2Pix

(c) CycleGAN

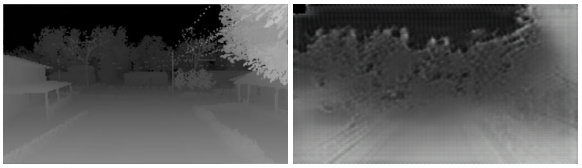


(d) Multi-Scale Deep Network

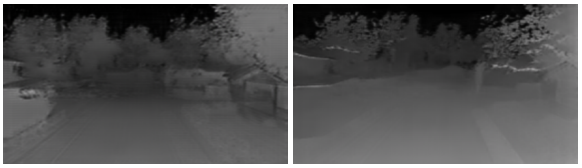


(e) Style Transfer CNN

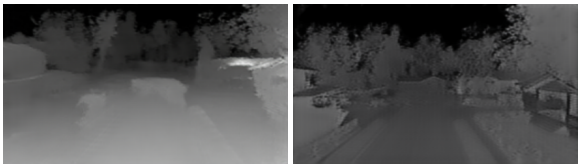
Figure 3: The various network generated depth maps for the scene presented in Figure 1.



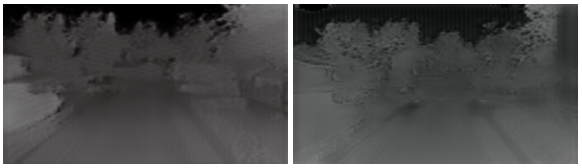
(a) Actual depth map (b) A = 1, B = 1



(c) A = 5, B = 5 (d) A = 10, B = 10



(e) A = 20, B = 20 (f) A = 5, B = 10



(g) A = 1, B = 5 (h) A = 5, B = 1

Figure 4: The various network generated depth maps for the scene presented in Figure 1.

Table 3: Results of each network.

Loss Type	AE	MSE	RMSLE
Logarithmic Difference	42.2	3246.7	0.632
Difference	15.9	659.9	0.399

first investigated due to instability issues. Previous approaches base the loss on the difference between the logarithm of the pixel values, but as **??** shows, the trained networks perform poorly. One possible reason is that the dataset we use is normalized to have pixel values of zero in far away places, leading to very large loss terms. Inspecting images reveals that network learns to produce very conservative, dark depth maps rather than good depth map estimates.

When the difference between pixel values is used in the loss function instead, the performance is much improved, as shown in Table 3. As a result, we used this modification for the loss function for the multi-scale deep network.

One problem with using the mutli-scale deep network for our problem stems from our small dataset. As Figure 5

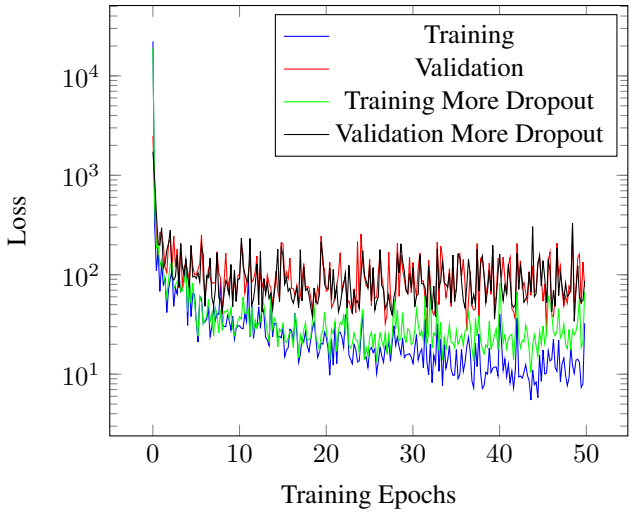


Figure 5: Multi-scale depth map training and validation loss

shows, the loss of the training set is much less than the loss of the validation set (note the logarithmic y-axis scaling). In an attempt to reduce the impact of over-fitting, dropout was added after the convolutional layers in the refined network with probability of keeping the activation of 0.5. The green and black lines in Figure 5 show the impact of dropout on the loss. Although the training loss is greater, the validation loss has no improvement. To overcome these challenges, a larger dataset might be required. Given that AirSim is new and powerful tool, and many people are looking to collaborate on UAV perception projects, the future might yield larger open source datasets for people to train UAV perception algorithms via supervised learning.

The style-transfer CNN was trained using Adam as the optimizer. The initial learning rate was set to 10^-3, and the default beta values were used throughout training. In the later stages of training, when the loss descent plateaued, the learning rate was slightly increased to allow the system to escape local optima. The mini-batches used during training were limited to five images. This was due to a memory allocation constraint on the GPU on which the network was being trained. The main hyperparmeters that were investigated were then the weighting terms, λandλ_a in the composite loss functions for the transfer network and auto-encoder respectively. The transfer-function weight parameter was selected using hyperparameter sweeps over 10 epochs of data and two-fold validation, the data in each validation fold was randomly selected from the test data to total 20 percent of the total data. The final value was set to 0.01. While this seems to indicate significantly less emphasis was placed on the feature loss in the total loss, inspection of both loss terms shows that the unweighted absolute value of the feature loss was an order of magnitude higher than the L2 loss in early training, so that the relative contribution to the loss term of either term is nearly equal. It should be noted that this parameter tuning took place before the composite loss term was included in the auto-encoder training, and was instead based on performance observed training on the L2-loss trained auto-encoder. The weighting term on the auto-encoder was similarly selected via hyperparameter tuning sweeps, using only single-fold validation. This term was set to 0.1 for final training. The auto-encoder network was trained over 50 epochs of the data, and the transfer network was trained over 100 epochs. The resulting performance of the trained transfer network was unable to surpass the performance of the baseline network in any of the reported metrics. This is likely due to the features represented at the central layer of the auto-encoder not being extractable from both the RGB image and the depth map. Qualitatively, however, the features are much more clearly retained in the style-transfer CNN than in either Pix2Pix or the baseline. Additionally, while the features in CycleGAN appear more sharply defined, the values over these features can be seen to be highly inaccurate at points. This suggests that our novel method has an ability to both retain feature representation and map accurately across average values. With future development of the network architecture, the performance may be further improved.

6. Conclusions and Future Work

As can be seen in the above section, the baseline method, multi-scale deep network, outperformed most other methods in nearly every metric. Inspection of the output maps, however, shows that the images produced are extremely blurry. So while they are able to achieve low average error, their utility for practical depth mapping applications is limited. CycleGAN is able to best retain the image features with clear definition, but often with high error in the depth-space representation. This was mitigated somewhat by the adapted loss function introduced in training. The insight to design a network and corresponding loss function to intentionally retain features while still minimizing error was demonstrated to be a viable approach in the novel style-transfer CNN approach developed here. Further tuning of this approach may allow it to achieve improved performance.

Future work on this topic should seek to identify jointly extractable features of interest for use in feature-based style transfer approaches, such as the one presented here. This investigation should inform the design of appropriate auto-encoder networks for perceptual feature extraction. Additionally, the metrics used, while standard for evaluation of this transfer problem, are incomplete. Additional metrics evaluating other characteristics of depth map accuracy, such as maximum error, should be investigated. Inclusion of ad-

ditional feature-space representations, such as image optical flow, could help greatly improve performance in near-field accuracy. Extending this work beyond single images to include processing of video-streams through a recurrent system may also be a valuable approach.

Eventually we would like to fly an effective scene to depth map network on a real aircraft, the Intel Aero Ready-to-Fly Drone. This aircraft is equipped with an Intel RealSense depth camera, which will allow us to compare our network in real environments against an accurate depth sensor.

Acknowledgements

The authors would like to thank Shital Shah for his help with Airsim, Mykel Kochenderfer for his continuous support and enthusiasm about our project, Jeremy Morton for his great conversations and input on CNN techniques, Louis Dressel for his great simulated pilot skills, and the CS231N course staff!

References

[1] Y. S. Heo, K. M. Lee, and S. U. Lee. Joint depth map and color consistency estimation for stereo images with different illuminations and cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35, 2013.

[2] Y. M. Tsai, Y. L. Chang, and L. G. Chen. Block-based vanishing line and vanishing point detection for 3d scene reconstruction. In *2006 International Symposium on Intelligent Signal Processing and Communications*, 2006.

[3] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. 3-d depth reconstruction from a single still image. *International journal of computer vision*, (1), 2008.

[4] Sebastiano Battiato, Salvatore Curti, Marco La Cascia, Marcello Tortora, and Emiliano Scordato. Depth map generation by image classification, 2004.

[5] Y. L. Chang, C. Y. Fang, L. F. Ding, S. Y. Chen, and L. G. Chen. Depth map generation for 2d-to-3d conversion by short-term motion assisted color segmentation. In *2007 IEEE International Conference on Multimedia and Expo*, 2007.

[6] S. Battiato, A. Capra, S. Curti, and M. La Cascia. 3d stereoscopic image pairs by depth-map generation. In *Proceedings of the 3D Data Processing, Visualization, and Transmission, 2Nd International Symposium*, 2004.

[7] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[8] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, 2014.

[9] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE, 2016.

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, 2016.

[11] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.

[12] Justin Johnson, Alexandre Alahi, and Fei-Fei Li. Perceptual losses for real-time style transfer and super-resolution. *CoRR*, 2016.

[13] Ravi Garg, Vijay Kumar B. G, and Ian D. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. *CoRR*, 2016.

[14] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *CoRR*, 2015.

[15] Wenjie Luo, Alexander G. Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[16] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles, 2017.

[17] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems 27*. 2014.