Monocular Depth Estimation using Synthetic Data for an Augmented Reality Training System in Laparoscopic Surgery*

Andre M. Schreiber, Minsik Hong, and Jerzy W. Rozenblit

Abstract— **Depth estimation is an important challenge in the field of augmented reality. Supervised deep learning methods of depth estimation can be difficult to use in novel settings due to the need for labeled training data. The work presented in this paper overcomes the challenge in a laparoscopic surgical simulation environment by using synthetic data generation for RGB-D training data. We also provide a neural network architecture that can generate real-time 448x448 depth map outputs suitable for use in AR applications. Our approach shows satisfactory performance when tested on a non-synthetic test dataset with an RMSE of 2.50 cm, MAE of 1.04 cm, and $\delta < 1.25$ of 0.987.**

I. Introduction

Depth estimation from camera images has various applications in fields such as robotics and augmented reality (AR). In AR, depth estimation provides a method for obtaining the depth at various pixels in a camera image. Such depth information can be used to provide scene understanding for the AR software and to produce occlusive interactions between computer-generated geometry and geometry in the real world.

Various approaches exist for estimating the depth map of a scene. Sensor-based techniques such as LIDAR or stereo depth cameras can provide depth data in a variety of environments [1]. However, use of such sensors requires additional hardware. Additionally, LIDAR can only provide sparse depth maps and does not produce dense, per-pixel depth measurements [1], while other methods like IR-based stereo cameras may produce outputs with regions of unknown depth (depth holes). Sensor-based approaches also may not be applicable when the setting of interest involves short distances (such as distances less than 20 cm).

With the increased interest in deep learning in recent years, much research has been conducted on using neural networks to solve challenging computer vision problems such as depth estimation. Several deep learning approaches exist for the task of depth estimation from monocular camera images. Many of these approaches are summarized by [1], and include supervised, unsupervised, and semi-supervised techniques. Supervised methods require labeled ground-truth data for training, whereas unsupervised depth estimation does not rely on ground-truth depth data during training. Semi-supervised approaches use a hybrid approach that adopts characteristics of unsupervised and supervised learning [2]. Supervised approaches have the limitation of requiring large quantities of labeled training data to perform well [2], whereas unsupervised methods may suffer from erroneous or ill-defined scale [1]. In addition, many supervised depth estimation algorithms, such as those described by [3] produce outputs of resolutions that would yield significant artifacts if used for depth maps in AR applications.

Deep learning algorithms for depth estimation are typically trained and evaluated on large datasets such as KITTI [4] or NYU Depth [5]. However, applying algorithms trained on such datasets may produce poor results when applied to datasets that show large differences from the original training dataset. This poses a problem for using a model trained on these datasets for applications where the input data differ significantly from the data on which the algorithm was initially trained.

The laparoscopic (minimally invasive) surgical training environment poses numerous challenges for AR depth estimation. The training setup involves a stationary camera and depths shorter than seen in typical indoor datasets like NYU Depth [5]. For example, the minimum distance between a camera and a surgical instrument is less than 15 cm as shown in Figure 1-(a). Techniques based on sensors suffer in this environment due to minimum depth limitations, sparsity of depth data, or depth holes—all of which create large, noticeable artifacts if used as a depth map for AR rendering. The lack of camera motion also precludes the possibility of using scene reconstruction techniques that rely on a moving camera to rectify issues like depth holes. Definitive scale is required for this application, increasing the difficulty of applying unsupervised deep learning approaches. Supervised approaches are attractive but require labeled ground-truth

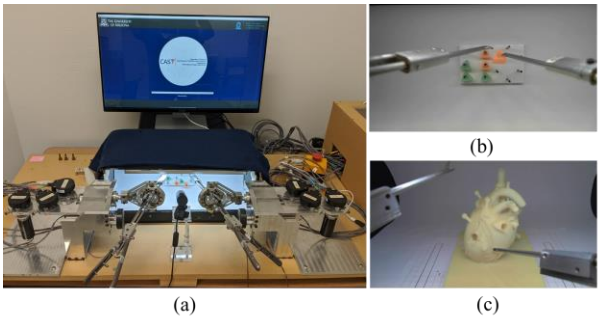

Figure 1. Example of laparoscopic surgery training environment: (a) Computer-Assisted Surgical Trainer (CAST) [27], [28], (b) a peg board setup, (c) a 3D printed heart setup.

*Research supported by National Science Foundation under Grant Number 1622589 "Computer Guided Laparoscopy Training".

Andre M. Schreiber is with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721 USA (corresponding author; e-mail: aschreiber1@email.arizona.edu).

Minsik Hong is with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721 USA (e-mail: mshong@email.arizona.edu).

Jerzy W. Rozenblit is with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721 USA (e-mail: jerzyr@arizona.edu).

depth data, which is difficult to obtain due to issues with sensor-based techniques as previously described. Furthermore, models trained on datasets different from those on which they are ultimately applied often suffer from issues with generalization, showing poor performance on datasets that are different from those on which they are trained on. This issue with generalization engenders difficulty with regards to using publicly available depth datasets like NYU Depth [5] for training a supervised model for the laparoscopic surgery training environment.

The laparoscopic surgical training environment (e.g., a peg board for the Fundamentals of Laparoscopic Surgery [6] as shown in Figure 1-(b)) for which this model is to be applied has known parameters, such as the size of the environment and the computer-aided design (CAD) models for the objects within the scene (such as training geometry and instruments). Similarly, the environment does not vary nearly as appreciably as those featured in datasets such as KITTI [4] or NYU Depth [5]. Most variations in the laparoscopic training environment result from minor changes in camera positioning, instrument pose, lighting conditions, or training environment geometry (e.g., switching a peg board (Figure 1-(b)) for an object manipulation task to a 3D printed heart (Figure 1-(c)) for an instrument navigation task). Such regulation of the laparoscopic training environment enables the creation of synthetic data using computer graphics software, from which high-resolution depth maps can be generated along with color renders of the environment.

In this paper, we introduce and demonstrate the effectiveness of a method of supervised monocular depth estimation using synthetic data. The approach of using synthetic data circumvents the challenges associated with collecting ground-truth data in the laparoscopic environment, such as minimum depth limitations of depth cameras and holes produced by such depth sensors. In addition, we present a neural network architecture that enables real-time depth estimation on a GPU, while producing 448x448 output resolution depth maps suitable for use in AR. Benchmarking the neural network against RGB-D data collected in a laparoscopic training setup for which ground-truth depth data could be collected demonstrates the viability of the approach.

II. Related Work

Extensive research has been conducted on depth estimation using deep learning approaches, as well as the use of synthetic data for training machine learning and deep learning models for computer vision applications.

Zhao *et al.* [1] provide a review of numerous deep learning-based approaches for depth estimation, including supervised, unsupervised, and semi-supervised approaches. Eigen *et al.* [3] provide a supervised regression approach, which utilizes a two-network model including a coarse depth estimation network that produces a global-scale estimation of depth and a local-scale network that improves the predictions provided by the coarse network. Laina *et al*. [7] provide an approach to supervised monocular depth estimation showing improved performance over that of [3] by using a fully convolutional architecture, up-projection blocks, and reverse Huber loss. Laina *et al.* also utilize a pretrained ResNet-50 [8] architecture for the initial part of their network. Wofk *et al.* [9] introduce the FastDepth network, which utilizes an encoder-decoder convolutional architecture with an optimized decoder to provide improved performance on embedded platforms.

Garg *et al*. [10] provide an approach to depth estimation which does not require ground-truth data, utilizing pairs of images with known camera deviations and training a neural network to construct one image from the other using predicted depth information. Chen *et al*. [11] provide a new dataset where each image provides a pair of points with labeled relative depth and presents a neural network for estimating relative depth for each pixel. However, the method presented in [11] only provides relative depth measurements, rather than measurements with known units like meters.

Tremblay *et al.* [12] successfully utilize synthetic data and domain randomization to train a neural network on an object detection task. Handa *et al.* [13] utilize synthetic data in order to attain high performance on per-pixel labelling for scene understanding. Sankaranarayanan *et al*. [14] provide a method of reducing the domain gap between synthetic and real datasets using a method employing the generative adversarial network (GAN) [15]. Synthetic data has also been successfully applied to the problem of crowd counting, with fine tuning by using real data or domain adaptation improving the performance of a network trained on synthetic data [16]. Wang *et al.* [17] utilize data generated in Unreal Engine 4 [18] to accomplish a vehicle detection task and use transfer learning with real data to improve performance on non-synthetic data.

III. Data Generation and Collection

A. Synthetic Data Generation

Synthetic data for a laparoscopic surgical training setup were created in order to train a supervised monocular depth estimation neural network. CAD models for the components of the simulation setup were either known or created from measurements. Unity [19] was used to render the synthetic environment. The texture (UV) coordinates used for applying image texture details to the surfaces of the CAD objects were obtained using Blender [20], and the CAD objects were provided textures and materials within Unity. For rendering within Unity, the High-Definition Render Pipeline (HDRP) was used to maximize visual fidelity.

To show the viability of this approach, the results from the neural network trained on synthetic data were to be benchmarked against data collected using an Intel RealSense D435 camera [21]. The intrinsic matrix of the camera used for benchmarking was recorded and its parameters were applied to the Camera component in Unity to ensure that aspects such as field of view remained consistent between the synthetic data and real data that the network was to be evaluated on.

Two main environments—a blocks world style 3D printed training environment and a 3D printed heart environment—were used for the synthetic data scene. These environment objects were given multiple different poses to provide more variety in the synthetic dataset. In addition, combinations of these objects were used, such as only one environment being present, both being present, or none being present. Animations of the surgical instruments were created

to ensure variety in instrument poses, while also preventing issues such as clipping of the instruments and the environment that could occur if the instruments were to be randomly placed. Multiple different texture and material variants of the scene geometry were used to increase variety for the dataset. Different lighting conditions were also used to maximize dataset variety, with different light intensities, color temperatures, and positions.

In Unity, the depth buffer of a rendered scene is necessary for occlusive interactions between objects. Thus, the depth of each pixel is already known. However, depth buffers in computer graphics applications such as Unity are non-linear, providing greater depth resolution at closer distances. Therefore, the depth buffer was linearized and converted to meters before being recorded as a synthetic depth map.

Data were collected in Unity using a script that played the animation for each environment configuration, recording data at regular intervals as the animation played. After the animation for a given configuration was finished, the environment would change to a new configuration and data collection would start after a short delay to ensure stable lighting conditions after the environment change. Before each image was taken, a small random position and rotation offset was applied to the camera to increase the variations in camera posture within the training dataset.

Data were recorded by saving the 8-bit per channel color buffer and linearized 16-bit depth buffer contents to two 640x480 pixel EXR [22] files. While data were recorded, the timescale in Unity was set to 0 to pause the program and ensure that object motion would not cause differences between the color and depth data for a given synthetic data sample. 3248 synthetic training samples (plus an additional 836 validation samples to optimize training) and 622 synthetic testing samples were created with this approach. An example RGB-D sample from the synthetic training dataset is shown in Figure 2.

B. Benchmark Data Collection

In addition to synthetic data, RGB-D images were captured using a RealSense D435 camera to allow for benchmarking of the neural network trained on synthetic data. The RealSense D435 camera has a minimum depth of approximately 28 cm at maximum resolution [23]. Due to the minimum depth limitation of the RealSense camera, a laparoscopic training setup different from the one shown in Figure 1 was constructed. This new setup is shown as the color image in Figure 3. Such a setup was utilized to ensure that the distance of objects from the camera was greater than the minimum depth supported by the depth camera in order to minimize issues such as depth holes. Using this new setup, six separate camera sequences were recorded using the RealSense camera. Each camera sequence involved changes such as different camera pose and/or different scene geometry. For each camera sequence, instruments were moved throughout the sequence to mimic the actions of a trainee using the laparoscopic training setup.

The depth maps collected from the RealSense camera were post-processed to align them with their corresponding color images. Each camera sequence was recorded at a 640x480 pixel resolution and 15 FPS frame rate for both depth and color. Images and aligned depth maps were extracted from the videos, where an image was extracted to create a sample every 20 frames. Samples were collected every 20 frames instead of every frame in order to reduce the amount of nearly identical samples in the dataset. A test set of 547 RGB-D samples was created from the camera data. An example RGB-D sample from the benchmark dataset is shown in Figure 3.

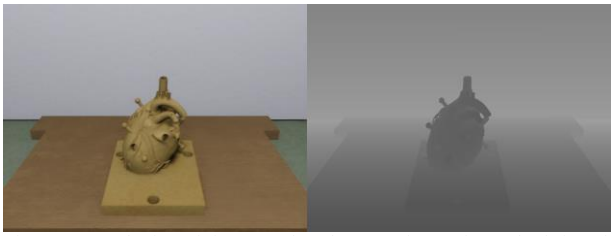

Figure 2. Synthetic training sample. The RGB image is on the left and the depth map is on the right.

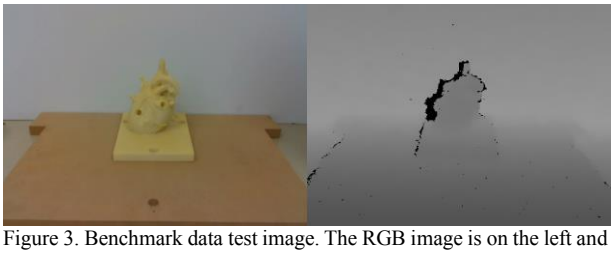

Figure 3. Benchmark data test image. The RGB image is on the left and depth map is on the right. Black areas of the depth map are due to depth holes.

IV. Network Architecture and Training

A. Architecture

The network adopted for depth estimation utilizes an encoder-decoder architecture. In this encoder-decoder architecture, the input image is processed by the encoder, producing a latent representation of the image. The latent representation produced by the encoder is then processed and upscaled by the decoder in order to yield a depth map.

Similar to the network presented in [7], a ResNet-50 network [8] is used for the encoder stage of the network. The original ResNet architecture [8] is used for image recognition and features a fully connected layer at the end of the network in order to produce outputs for image classification. In our architecture, this fully connected layer is replaced by a decoder that utilizes the latent representation of the RGB image created by the ResNet encoder to construct a depth map. In addition, the beginning of the ResNet-50 network involves a convolutional layer with 64 filters, a kernel size of 7x7, and a stride of 2x2, followed by batch normalization and ReLU activation [8]. This initial ResNet layer was replaced by two similar layers, using the same kernel size, padding, and stride, but with the first layer having only 32 filters and the second layer having 64 filters. These layers were not pretrained, but the rest of the ResNet encoder used weights pretrained on the ImageNet dataset [24]. The change to the initial layer was implemented to improve performance with an input size image of 448x448x3.

The decoder section of our network is used for creating a depth map output from the latent representation produced by the ResNet-50 encoder. Two approaches for the decoder were tested. One variant (which we will call ConvT5) involved utilizing blocks of transpose convolutional layers of kernel

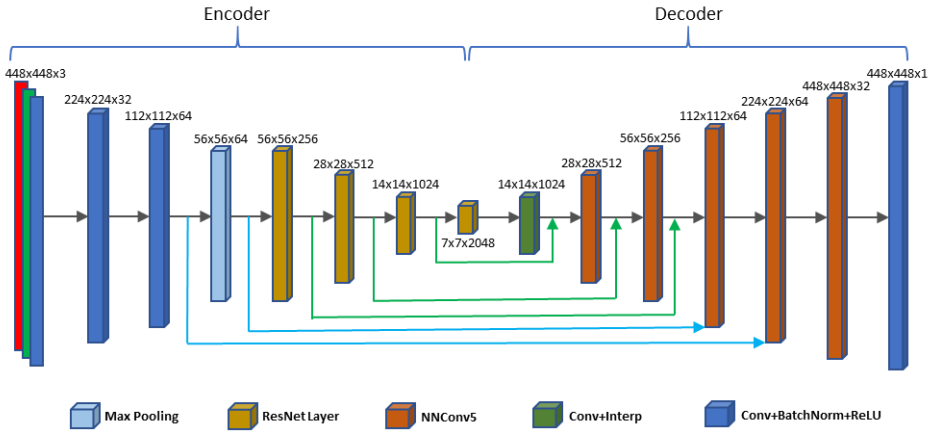

Figure 4. Neural network architecture with NNConv5-based decoder

size 5x5 and stride 2x2, where each transpose convolutional operation was followed by batch normalization and ReLU activation. The transpose convolution approach is referred to as DeConv5 in [9]. The second approach involved using NNConv5 blocks as given by [9]. The NNConv5 blocks apply convolution with a kernel of size 5x5, followed by 2 times nearest neighbor interpolation. Similar accuracies with each decoder were observed with slightly improved performance given by the decoder based on NNConv5, so the NNConv5-based decoder was ultimately used for the network. Additive skip connections are used between the encoder and decoder.

The network architecture for the NNConv5-based decoder is shown in Figure 4. In Figure 4, light blue arrows into NNConv5 blocks indicate additive skip connections within the interpolation of the NNConv5 block, while green arrows into other arrows indicate additive skip connections into the input of the NNConv5 block. The dark blue blocks in Figure 4 represent convolutional blocks (which include a convolutional layer, batch normalization, and ReLU activation), the green block represents pointwise convolution followed by 2 times nearest neighbor interpolation, the light blue block is a max pooling layer, the yellow blocks are layers from the ResNet-50 network, and the orange blocks are NNConv5 blocks.

B. *Training Procedure*

The network was trained using the synthetic dataset of RGB-D images. For the final networks that were tested on real data, training was conducted on the union of the training and validation synthetic datasets (total of 4084 images before perturbation).

Data augmentation was performed to improve performance. The dataset size was doubled by horizontally reflecting each image. In addition, random perturbation to the images with probability 0.3 was performed, with the following perturbations equally likely for an image selected for perturbation: (1) clipping color values between random minimum and maximum values; (2) making the image greyscale; (3) adding uniform noise; (4) and multiplying hue, saturation, and value by random numbers. RGB Noise was added after perturbation with probability 0.45.

The network was trained for 20 epochs with a batch size of 4 on the training dataset, which ultimately consisted of 8168 RGB-D samples after data augmentation. Each training RGB image input and depth map target were resized to a resolution of 448x448. Reverse Huber loss (berHu loss), as presented in [7], was used for the training loss, and Adam [25] was used as the optimizer with α=0.001, β_1=0.9, and β_2=0.999. The network was implemented using PyTorch [26].

V. Results

A. *Methodology*

The network was tested by using the synthetic data test set and the test set of data captured by the depth camera. The metrics used are root mean square error (RMSE), mean absolute error (MAE), and $\delta < 1.25$. The formulas for the RMSE and $\delta < 1.25$ error metrics are provided in [3]. The color images used as inputs to the network were resized to a resolution of 448x448 before being provided to the network, and the outputs from the neural network are 448x448 resolution depth maps. The original color image data and ground-truth depth data has resolution 640x480. The depth maps from the neural network were thus upscaled to 640x480 before being compared against the ground-truth. Unmeasured areas (depth holes) in the ground-truth images were ignored when computing the error metrics.

B. *Quantitative Results*

The error metrics on both the synthetic and non-synthetic testing datasets are shown in Table 1. The RMSE and $\delta < 1.25$

Table 1. Performance metrics

Network Variant	**Synthetic Test Set**			**Non-Synthetic Test Set**			GPU Time (ms)
	RMSE (cm)	MAE (cm)	$\delta < 1.25$	RMSE (cm)	MAE (cm)	$\delta < 1.25$	
ConvT5 Decoder	1.66	0.731	0.990	2.52	**0.973**	0.986	13.4
NNConv5 Decoder	**1.50**	**0.633**	**0.993**	**2.50**	1.04	**0.987**	**12.8**

metrics are common methods of measuring the performance of depth estimation neural networks, as described in [3] and [9]. MAE is given as an error metric in order to provide insight on the average prediction error in metric units with all errors weighted equally. The synthetic test set is comprised of synthetic images that the network was not trained on, while the non-synthetic test set is composed of the non-synthetic images collected by the RealSense camera. The error metrics are shown for two network architectures; both used the ResNet-50 encoder but utilized different decoders (one used the NNConv5-based decoder and the other used the ConvT5-based decoder).

For the results shown in Table 1, lower values of RMSE and MAE are better and higher values of $\delta < 1.25$ are better. The RMSE and MAE both measure the magnitude of error in predictions, with RMSE giving higher weight to large prediction errors and MAE representing the average absolute difference between the ground-truth and the prediction. $\delta < 1.25$ represents the average fraction of pixels in which the maximum of the predicted value divided by measured value and the measured value divided by the predicted value is less than a threshold of 1.25. The performance of the neural network on the synthetic test set is higher than that of the non-synthetic test set, which is expected as the network was only trained on synthetic data. The results suggest that the network produces accurate depth estimations, with the average absolute difference between predicted depth and ground-truth being 1.04 cm for the network with the NNConv5-based decoder on the non-synthetic dataset. The RealSense camera used for ground-truth measurements has an absolute error of 2% or less at up to 2 meters [23]. Given the dimensions of the laparoscopic environment (with objects approximately 0.3 m to 0.7 m away from the camera), our predictions provide satisfactory accuracy when compared to the RealSense camera.

The GPU run times per sample are also presented. The computer used for this benchmark had an Nvidia RTX 2070 Super GPU, 32 GB of RAM, and an Intel i9-9900K CPU. The time taken to process each image shows that the network can be used for real-time depth estimation.

C. *Qualitative Results*

Qualitative predictions on both a synthetic test image and a non-synthetic test image using the final network are shown in Figures 5 and 6, respectively. The depth map predictions shown in these figures have been rescaled to 640x480 from their original 448x448 resolution to be compared against the ground truth depth maps, which are natively of resolution 640x480.

(a) (b) (c)

Figure 5. Predictions of neural network on synthetic test data, showing (a) color, (b) true depth, and (c) predicted depth.

(a) (b) (c)

Figure 6. Predictions of neural network on non-synthetic data, showing (a) color, (b) true depth, and (c) predicted depth.

Figure 5 shows a color image, ground-truth depth map, and predicted depth map for a synthetic image that was part of the synthetic test set. The images shown in Figure 6 are a color image and ground-truth depth image collected by the RealSense camera, along with the predictions made by the neural network using the color image. In Figures 5 and 6, lighter areas of the depth maps indicate higher depth values. As seen in Figures 5 and 6, the predictions produced by the neural network show high similarity with the ground-truth. In addition, Figure 6-(b) shows significant artifacts due to depth holes from the RealSense camera, whereas the predictions from our network produce a depth map without depth holes.

VI. Conclusion and Future Work

The results shown in this work suggest the effectiveness of using synthetic data for supervised depth estimation in applications like computer-assisted laparoscopic surgical simulation. Our findings show that the results of a depth estimation neural network trained only on synthetic data are accurate in our laparoscopic setting if the synthetic environment is constructed with accurate dimensions and reasonable visual fidelity.

This approach has numerous advantages over alternatives for obtaining depth information. As seen in Figure 3, large depth holes can be produced by depth cameras. Such depth holes greatly diminish the utility of depth cameras for AR applications by introducing regions of unknown depth that would lead to implausible AR renderings. The supervised neural network approach presented here does not suffer from this issue, as shown by Figure 6. In addition, the network presented here provides real-time, reasonably high resolution 448x448 depth maps relative to works like [3]. Thus, the depth maps from our network can be upscaled to produce only minor artifacts for AR rendering at 480p or 720p resolutions.

The accuracy that can be achieved by this approach suggests that it can be used to generate depth maps in laparoscopic training (and other) setups in which one could not ordinarily obtain high quality data using depth cameras. The data collected for the experiment presented in this paper was collected with a setup providing sufficient depths to be measured using a depth camera, but more practical laparoscopic training setups are smaller and will have depths that cannot be measured using depth cameras like [21]. Thus, our approach suggests that synthetic data can be employed for training supervised neural networks for use in environments in which densely labeled depth data cannot be collected from depth cameras.

Further work involves integrating this network within the AR rendering pipeline of a laparoscopic surgical simulation setup. The integration of this network into the AR rendering

pipeline of our CAST training system [27], [28] involves using the neural network outputs to produce occlusion interactions between physical objects in the simulator scene and computer-generated geometry. Such further work will involve retraining the network to be applied for depth map generation in an environment in which ground-truth depth maps cannot be measured by a depth camera due to minimum depth limitations. Further work could also involve modifying the network architecture to utilize video sequences instead of distinct images, thereby allowing the network to utilize temporal consistency between video frames to reduce error and increase stability in the generated depth maps by using methods similar to those described in [29].

The ability to generate depth maps from monocular camera images within the surgical simulator provides numerous advantages. The current occlusion system utilized in CAST [27], [28] involves rendering physical objects with known poses and shapes into a secondary depth buffer, which is then used in addition to the native depth buffer of the rendering API (i.e., OpenGL) to test occlusion for fragments of computer-generated geometry. This approach requires accurate knowledge of the shapes, positions, and poses of physical objects in the simulator scene at any given time; for example, such an approach requires knowing the position, rotation, and CAD models of the environment geometry, as well as of the instruments present in the simulator scene. Such an approach produces reasonable results for stationary geometry but can pose problems for depth map generation of dynamic objects such as instruments (where small errors in predicted pose can lead to significant visual inconsistency). Similarly, the prior method introduces limitations in introducing dynamic or unexpected events (e.g., simulated bleeding) with accurate occlusive interactions since it requires accurate knowledge of poses and shapes of all relevant physical objects. By utilizing a neural network, these challenges can be overcome as the proposed neural network produces depth maps in real-time using only camera footage, rather than requiring extensive knowledge of the scene setup and geometry.

REFERENCES

[1] C. Zhao, Q. S. Sun, C. Zhang, Y. Tang, and F. Qian, "Monocular depth estimation based on deep learning: An overview," *Sci. China Technol. Sci.*, vol. 63, no. 9, pp. 1612–1627, 2020.

[2] Y. Kuznietsov, J. Stuckler, and B. Leib, "Semi-Supervised Deep Learning for Monocular Depth Map Prediction," in *IEEE conference on computer vision and pattern recognition*, 2017, pp. 6647–6655.

[3] D. Eigen, C. Puhrsch, and R. Fergus, "Depth Map Prediction from a Single Image using a Multi-Scale Deep Network," in *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2*, 2014, pp. 2366–2374.

[4] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics : The KITTI dataset," *Int. J. Rob. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[5] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor Segmentation and Support Inference from RGBD Images," in *European conference on computer vision*, 2021, pp. 746–760.

[6] "Fundamentals of Laparoscopic Surgery." [Online]. Available: http//www.flsprogram.org/. [Accessed: 13-Apr-2021].

[7] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper Depth Prediction with Fully Convolutional Residual Networks," in *2016 Fourth international conference on 3D vision (3DV)*, 2016, pp. 239–248.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[9] D. Wofk, F. Ma, T. Yang, S. Karaman, and V. Sze, "FastDepth : Fast Monocular Depth Estimation on Embedded Systems," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6101–6108.

[10] R. Garg, V. Kumar B G, G. Carneiro, and I. Reid, "Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue," in *European conference on computer vision*, 2016, pp. 740–756.

[11] W. Chen, Z. Fu, D. Yang, and J. Deng, "Single-Image Depth Perception in the Wild," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 730–738.

[12] J. Tremblay *et al.*, "Training Deep Networks with Synthetic Data : Bridging the Reality Gap by Domain Randomization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 969–977.

[13] A. Handa, P. Viorica, V. Badrinarayanan, S. Stent, and R. Cipolla, "Understanding Real World Indoor Scenes With Synthetic Data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4077–4085.

[14] S. Sankaranarayanan, Y. Balaji, A. Jain, S. N. Lim, and R. Chellappa, "Learning from Synthetic Data : Addressing Domain Shift for Semantic Segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3752–3761.

[15] I. J. Goodfellow *et al.*, "Generative adversarial networks," in *arXiv preprint arXiv:1406.2661*, 2014.

[16] Q. Wang, J. Gao, W. Lin, and Y. Yuan, "Learning from Synthetic Data for Crowd Counting in the Wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8198–8207.

[17] Y. Wang, W. Deng, Z. Liu, and J. Wang, "Deep learning-based vehicle detection with synthetic image data," *IET Intell. Transp. Syst.*, vol. 13, no. 7, pp. 1097–1105, 2019.

[18] "Unreal Engine 4." [Online]. Available: https://www.unrealengine.com/en-US/. [Accessed: 13-Apr-2021].

[19] "Unity." [Online]. Available: https://unity.com/. [Accessed: 13-Apr-2021].

[20] L. Flavell, "UV Mapping," in *Beginning Blender*, Apress, 2010, pp. 97–122.

[21] L. Keselman *et al.*, "Intel RealSense Stereoscopic Depth Cameras," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1–10.

[22] "Openexr." [Online]. Available: https://www.openexr.com/. [Accessed: 13-Apr-2021].

[23] "Intel RealSense D400 Series Product Family Datasheet." [Online]. Available: https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/Intel-RealSense-D400-Series-Datasheet.pdf. [Accessed: 18-Apr-2021].

[24] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[25] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *arXiv preprint arXiv:1412.6980*, 2014.

[26] A. Paszke *et al.*, "PyTorch : An Imperative Style , High-Performance Deep Learning Library," 2019.

[27] M. Hong and J. W. Rozenblit, "An Adaptive Force Guidance System for Computer-guided Laparoscopy Training," *IEEE Trans. Cybern.*, 2021, doi: 10.1109/TCYB.2021.3051837.

[28] M. Hong, K. Meisner, S. Lee, A. M. Schreiber, and J. W. Rozenblit, "A Fuzzy Reasoning System for Computer-Guided Laparoscopy Training," in *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2020, pp. 1712–1717.

[29] X. Luo, J. Huang, R. Szeliski, K. Matzen, and J. Kopf, "Consistent Video Depth Estimation," *ACM Trans. Graph.*, vol. 39, no. 4, pp. 71:1-71:13, 2020.