Cross-Domain Synthetic-to-Real In-the-Wild Depth and Normal Estimation for 3D Scene Understanding

Anonymous CVPR submission

Paper ID 17351

1. Supplementary Material

In this supplementary material, we discuss our approach on generating the OmniHorizon dataset in Unreal Engine 4. We elaborate on the factors and certain assumptions that we made in order to render the dataset. Additionally, we discuss about training the UBotNet on indoor datasets and architecture choices. Finally, we demonstrate additional results for depth and normal estimation from real-world images in the wild.

1.1. Depth clamping

Rendering engines such as Unreal Engine 4 work with a larger depth range compared to that captured by physical sensors. However, we were interested in exploring the range of depth information that can be used for covering a wide range of objects in outdoor scenarios. This motivated us to simulate the limitations of the physical sensors and restrict the depth range to 150 m, similar to the Fukuoka dataset [4]. The engine places the far plane at infinity, which results in depth values being generated for extremely distant objects. To avoid this, we modify the depth material to visualise the impact of constraining the depth to a maximum specified value. We show the results for the clamping of depth at a range of 10m, 75m and 150m in Figure 2. At a depth of 10 m, only the truck is visible. When the depth range is raised to 75 m, cars and building start to appear in the background. At 150 m, the trees and most of the background are visible. By limiting the depth in outdoor environments, it is possible to focus solely on nearby items, or, depending on the application, on distant objects as well.

1.2. View-space vs world-space normals

The view space normals are calculated relative to the camera orientation, whereas the world space normals are calculated with respect to the global axes of the scene. The normals in view space are desired when using a perspective camera as they are tied to the camera pose (extrinsic parameters). However, the panoramic image is obtained by rotating the camera on both the horizontal and vertical axis in increments of fixed angle steps (5°), followed by merging the multiple views.

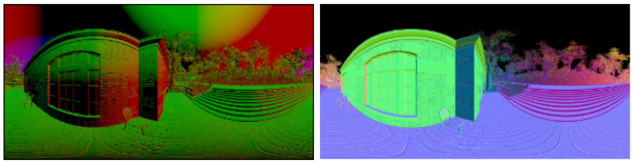

(a) view-space normals (b) world-space normals

Figure 1. *Comparison between view-space and world-space normals.* The normals captured in view-space appear as gradient with lack of clear distinction between the basis vectors. Normal maps recorded in world-space follow a consistent coordinate system.

Since the coordinate system is relative to the camera in view space, it also gets modified with the rotation. This results in a gradient of normals with no basis vectors. The normals obtained in world space are absolute and independent of camera pose. Figure 1 shows the difference between the view-space and world-space normals. Therefore, we captured the normals in world space as it was consistent for both within and between the scenes. We show the convention used for the world-space normals in Figure 3.

1.3. Virtual Avatars

As discussed in main paper, we utilised Metahumans [3] for the virtual avatars in the scene. We have used premade MetaHumans available in the Quixel bridge. It allowed us to bring in highly detailed characters and more diversity in the pedestrians. But there were certain challenges while using the Metahumans for the dataset. They are generated with multiple level of details (LODs) for perfomance optimisation. As a result, there would be sudden popups and other artifacts when the camera is approaching a character. Figure 4 illustrates how the character hair and details change when the camera is approaching the character. Lower LOD level (LOD 8) indicates lowest detailed polygon mesh with no advanced features such as detail normal maps or hairs. The higher LOD level (Level 0/1) has higher polygons with extra detail maps for the skin and hair grooming system.

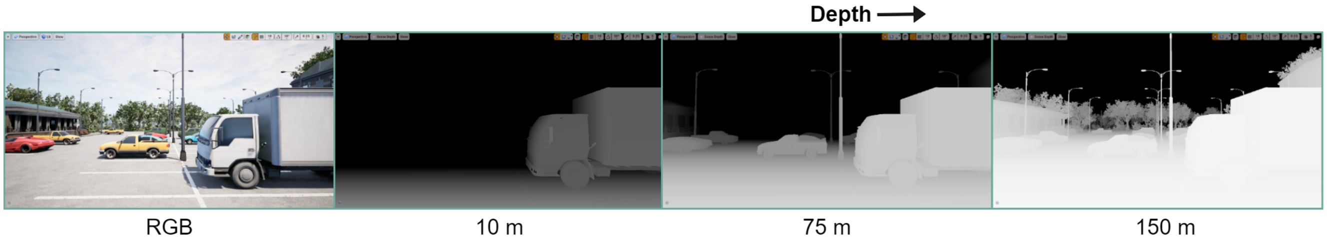


Figure 2. *Depth clamping experiment.* Comparison between various depth ranges after clamping to a specific range: 10 m, 75 m and 150 m. Inverted depth maps are shown for better visualization.

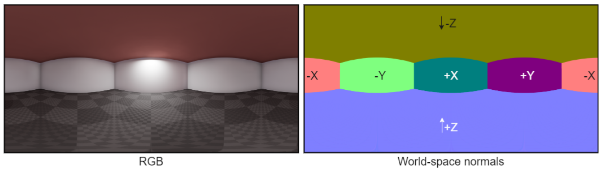


Figure 3. *Convention for the world-space normals.*

Additionally, we also observed artifacts in the normal maps for the characters with detailed grooming such as facial hair. Figure 5 shows the issues with the normal maps of a character in the region with facial hair. For such characters we used LOD 1 or LOD 2 to resolve the problems.

1.4. Assumptions in the Dataset

Our dataset renders several realistic outdoor and indoor environments with dynamic scene components. While curating this dataset, we made certain assumptions especially about the outdoor scenes which we list below:

1. The sky is assumed to be situated at infinitely large distance from the camera, and is represented as a spherical mesh of large radius encompassing the entire scene. Additionally, normals are not rendered for the sky region. It is represent using black which indicates invalid normal values. This allows us to distinguish sky from other regions in the scene.

2. Transparent and transluscent materials such as water, windows of the buildings and windshields of vehicles are replaced with fully reflective materials. We observed that inferring depth of such materials from color images is challenging and this limitation, for example, also applies to real-world datasets captured using lidars [6]. Figure 6 depicts the limitation of using transparent and translucent materials in the dataset. The original water shader in the scene was designed in such a way that it acted as a see-through material in case of depth. As a result, the depth map captures the terrain hidden underneath the water surface. We modified the the water shader to a reflective surface and thus depth is correctly rendered as a planar surface. We observed a similar case for the glass shader used for windows in the vehicles. The vehicles indeed have detailed indoors but due to reflections on the glass, the inside is not clearly visible. However, the depth map has much cleaner view of the indoors. To avoid this conflict of information, we use fully opaque and reflective materials for the windows.

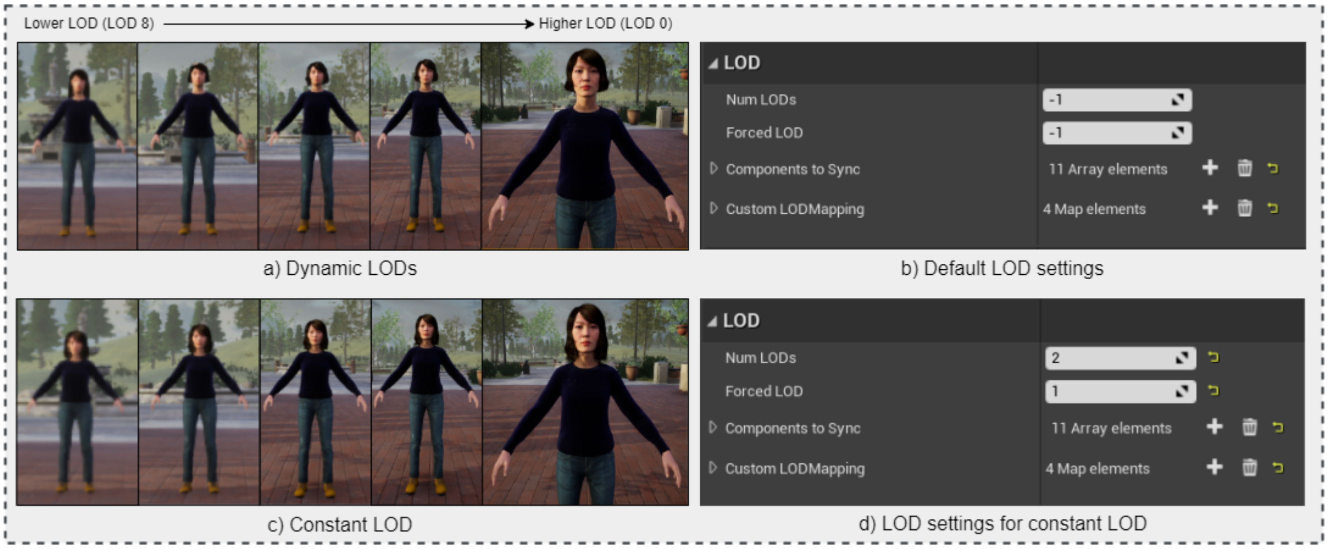
2. UBotNet

UBotNet for Indoor datasets. In the main paper, we discussed about the UBotNet architecture and the results from training on the OmniHorizon dataset. We additonally trained UBotNet on real-world indoor dataset Pano3D [1] to validate the performance of the network on other datasets. Pano3D is proposed as a modification of Matteport3D [2] and Gibson3D [7]. We used the official splits provided by the authors for Matterport3D for training and validation. For, Gibson, we used the *GibsonV2 Full Low Resolution* for training and validated on Matterport. All the images used for training were of 512 x 256 resolution. We used the loss function and training parameters outlined in our main paper. We trained UBotNet Lite on the both the datasets for 60 epochs.

Table 1. *Quantitative results for depth estimation using UBotNet Lite validated on indoor dataset - Matterport3D.*

	Depth Error ↓			Depth Accuracy ↑		
Dataset	RMSE	MRE	RMSE log	$\delta1$	$\delta2$	$\delta3$
Matterport3D	0.639	0.142	0.064	0.817	0.952	0.981
Gibson 3D	0.591	0.154	0.061	0.830	0.965	0.986

Table 1 shows the quantitative results for the task of depth estimation by UBotNet Lite evaluated on Matterport3D. We also show the qualitative results for the validation task in Figure 8. We observed better performance in overall metrics and the visual results when the network is trained on the Gibson3D.



LOD

Num LODs	-1
Forced LOD	-1
Components to Sync	11 Array elements
Custom LODMapping	4 Map elements

a) Dynamic LODs b) Default LOD settings

LOD

Num LODs	2
Forced LOD	1
Components to Sync	11 Array elements
Custom LODMapping	4 Map elements

c) Constant LOD d) LOD settings for constant LOD

Figure 4. *Dynamic LODs vs Constant LOD.* a) The Dynamic LOD system loads different meshes with various level of details based on the proximity to camera. This however results in sudden poping up of the meshes which generates artefacts in the data. b) Default LOD settings used by the engine. c) The modified LOD system is used to maintain LODs at a fixed LOD so that the avatar's appearance is unaffected by distance. d) The LOD of the character is locked to 1 using Forced LOD.

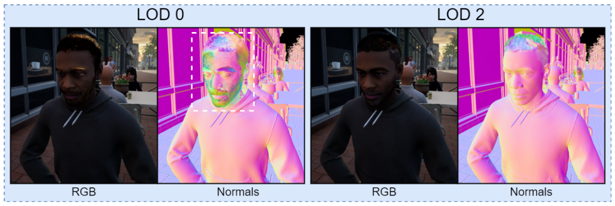


Figure 5. *Artefacts in normal maps for facial hairs.* When the camera is very close to the characters, the engine uses additional detail meshes for characters with facial hair at the highest LOD level (LOD 0). As a result, artefacts appear in the normal maps.We use LOD 1 or 2 for such characters.

Absolute vs Relative positional encoding. We utilised relative positional encoding [5] for self-attention in our proposed UBotNet architecture. We compare it against the absolute positional embeddings and show the quantitative results in Table 2. The absolute positional embeddings perform inferior to the relative positional embeddings used for self-attention. Moreover, the differences are more prominent in case of normal estimation. This is reaffirmed by the visual differences shown in Figure 7. The network loses the context required for learning the consistent representation of the normals. It behaves similar to the UNet$_{128}$ network discussed in the main paper.

3. Addition Results

We show additional results on the real-world images in the Figure 9 and Figure 10. The networks used were trained purely on OmniHorizon.

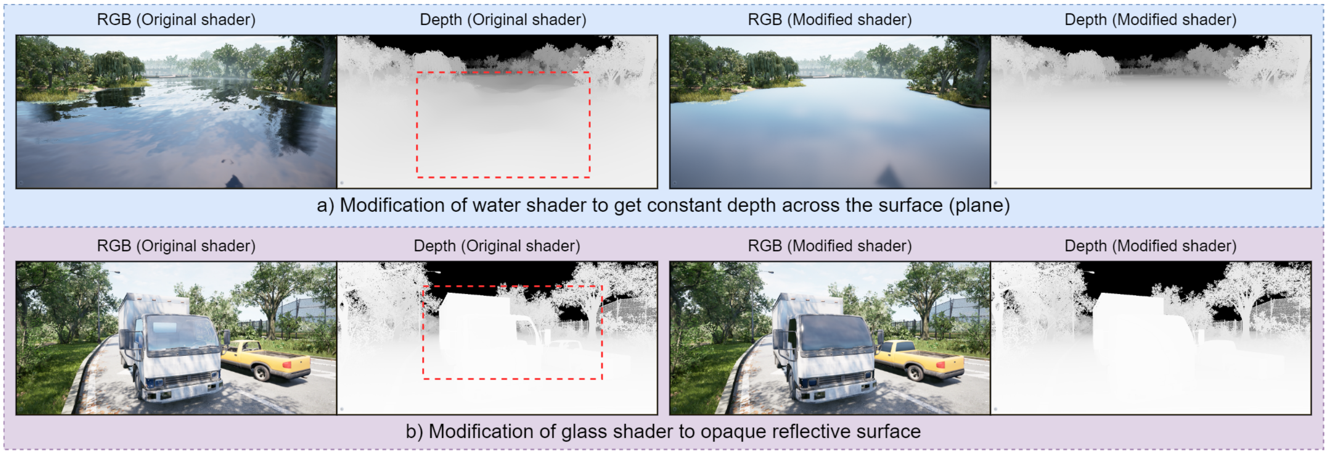


Figure 6. *Assumptions for the dataset.* a) Modification of water shader to achieve constant depth across the surface of the water. b) Modification of glass shader into opaque reflective surface which hides the interior parts of the vehicles.

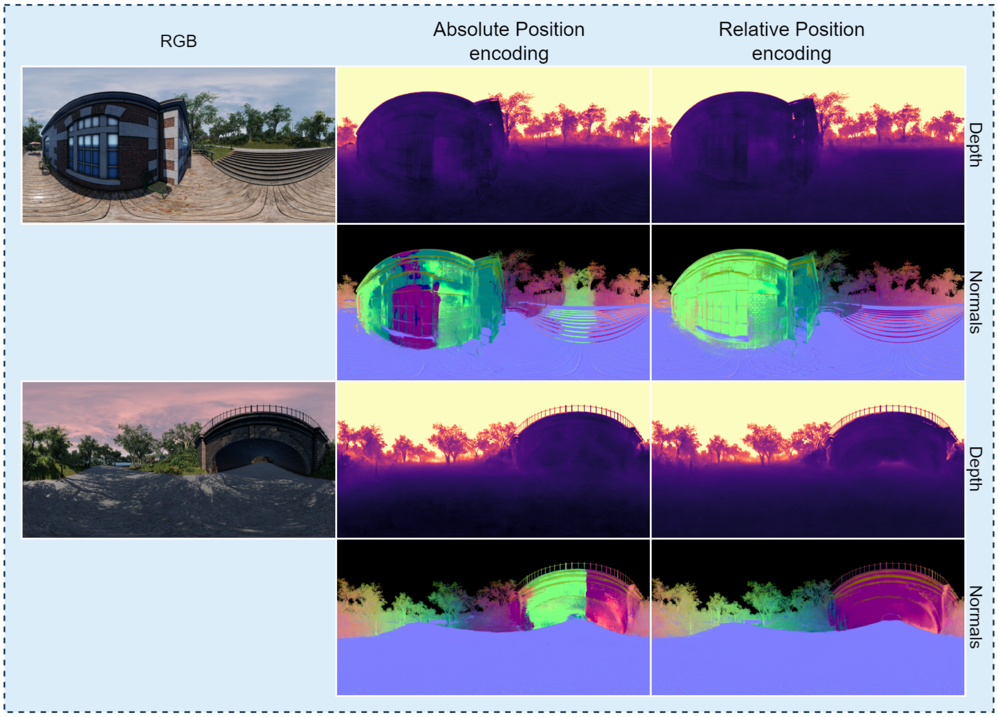


Figure 7. *Comparison between Abs. and Rel. positional embedding.* Absolute positional embedding loses the context required for learning the normals when used for self-attention.

Table 2. *Quantitative results for the comparison between the positional embedding used in the UBotNet architecture for self-attention.* The results for the Relative Positional Embedding are repeated from our main paper for the comparison.

	Depth Error ↓			Depth Accuracy ↑			Normal Error ↓			Normal Accuracy ↑		
Method	RMSE	MRE	RMSE log	$\delta1 < 1.25$	$\delta2 < 1.25^2$	$\delta3 < 1.25^3$	Mean	Median	RMSE	5.0°	7.5°	11.25°
Absolute Pos. Emb.	**0.053**	0.290	0.152	0.691	0.871	0.925	8.65	3.98	13.99	54.26	63.00	73.23
Relative Pos. Emb.	0.054	**0.271**	**0.151**	**0.712**	**0.875**	**0.926**	**7.44**	**3.61**	**12.12**	**56.80**	**67.28**	**78.52**

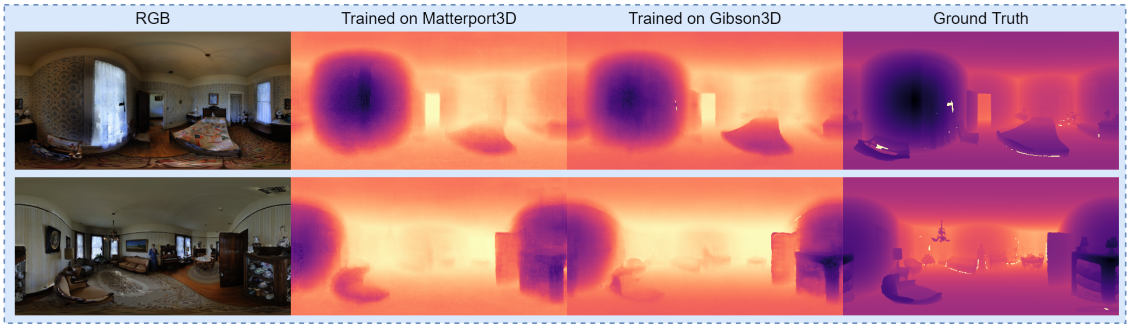


Figure 8. *Qualitative results for UBotNet Lite trained on Indoor datasets - Matterport3D and Gibson3D.*

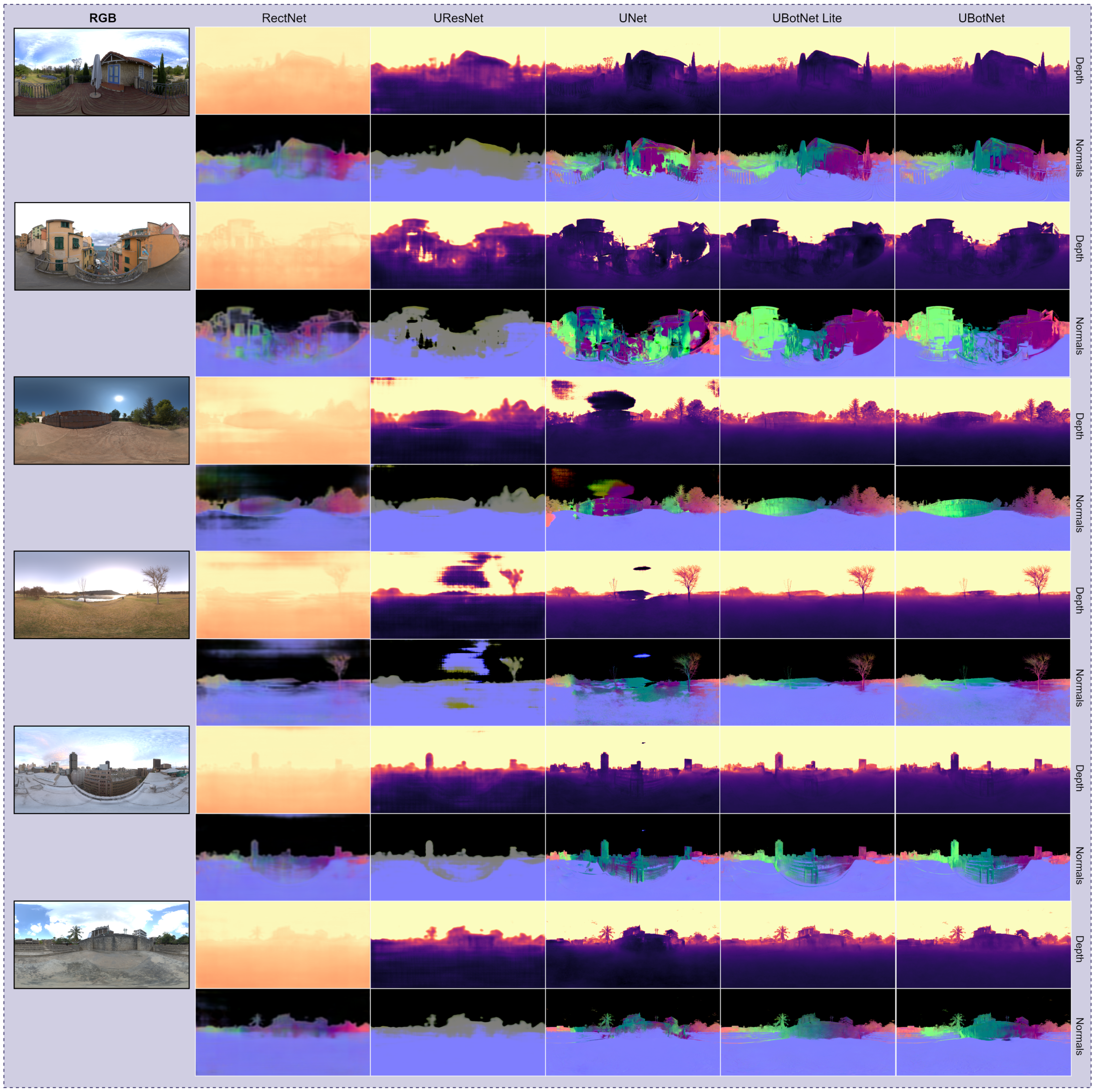


Figure 9. *Depth and Normal estimation on real-world images in the wild.* Comparison between all the networks discussed in main paper for depth and normal estimation on real world images.

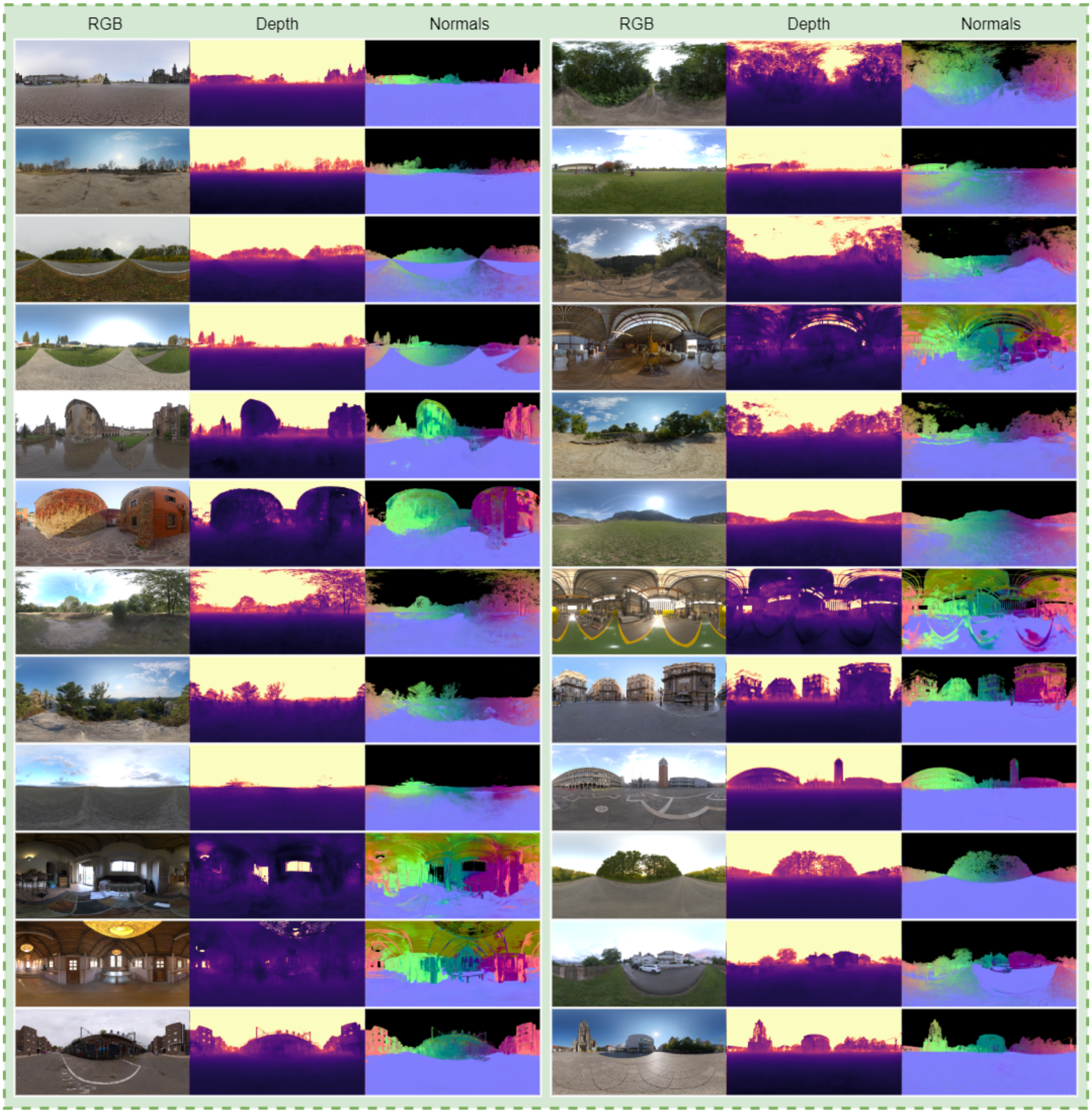


Figure 10. *Examples of depth and normal estimation using UBotNet on real-world images in the wild.*

References

[1] Georgios Albanis, Nikolaos Zioulis, Petros Drakoulis, Vasileios Gkitsas, Vladimiros Sterzentsenko, Federico Alvarez, Dimitrios Zarpalas, and Petros Daras. Pano3d: A holistic benchmark and a solid baseline for 360° depth estimation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3722–3732, 2021. 2

[2] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017. 2

[3] Epic Games. Metahumans, 2022. https://www.unrealengine.com/en-US/metahuman. 1

[4] Oscar Martinez Mozos, Kazuto Nakashima, Hojung Jung,

Yumi Iwashita, and Ryo Kurazume. Fukuoka datasets for place categorization. *The International Journal of Robotics Research*, 38(5):507–517, 2019. 1

[5] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16519–16529, June 2021. 3

[6] Igor Vasiljevic, Nick Kolkin, Shanyi Zhang, Ruotian Luo, Haochen Wang, Falcon Z. Dai, Andrea F. Daniele, Mohammadreza Mostajabi, Steven Basart, Matthew R. Walter, and Gregory Shakhnarovich. DIODE: A Dense Indoor and Outdoor DEpth Dataset. *CoRR*, abs/1908.00463, 2019. 2

[7] Fei Xia, Amir R. Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2