# Surface Normals: Always-On Perception for Vision-Based Robots

Gwangbin Bae[1] and Andrew J. Davison[1]

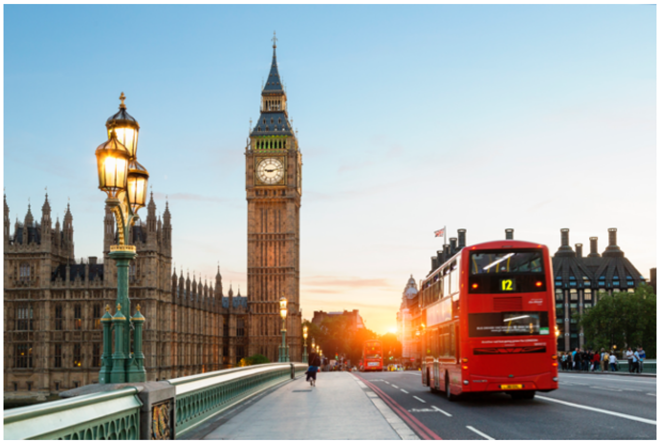

Fig. 1. Big Ben, the clock tower of the Palace of Westminster in London.

***Abstract*— In recent years, the usefulness of surface normal estimation methods has been demonstrated in various areas of robotics and computer vision. State-of-the-art methods show strong generalization capability and are highly efficient, running in real-time even on laptop computers. This makes them a strong candidate for being an ”always-on” perception for vision-based robots. Using the extracted surface normal cues as a foundation, task-(and domain-)specific functionalities can be built and called ”on-demand”.**

**In this paper, we push the limits of single-image surface normal estimation by rethinking the inductive biases needed for the task. Specifically, we propose to (1) utilize the per-pixel ray direction and (2) encode the relationship between neighboring surface normals by learning their relative rotation. The proposed method can generate crisp — yet, piecewise smooth — predictions for challenging in-the-wild images of arbitrary resolution and aspect ratio. Compared to a recent ViT-based state-of-the-art model, our method shows a stronger generalization ability, despite being trained on an orders of magnitude smaller dataset. The code is available at https://github.com/baegwangbin/DSINE.**

**At the workshop, we will show a real-time demo of the proposed method using a laptop computer.**

## I. INTRODUCTION

What do you see in Fig. 1? When you see the street, you know — without even trying — that it is a flat surface. When you see the double-decker bus, you would instantly associate it with some geometric shape, which was probably developed from your previous experience. When we see an image, we do not perceive it as an array of RGB values, but instead as a set of *geometric* objects, and such geometric perception seems to be *always-on*.

Then someone asks, ”How tall do you think the Big Ben is?”. This now requires more reasoning. We need to identify the objects for which we have a strong prior on their physical size and also estimate their relative distance to the camera, so our 3D reconstruction of the scene can be scaled properly. Similarly, when performing complex object manipulation tasks (e.g. building a Lego set), we need to obtain a more precise and fine-grained geometry of the objects. Such inference requires more compute, but we can stay efficient by only doing it *on-demand*.

Recent advances in single-image *surface normal estimation* show that the state-of-the-art models can be used as an always-on perception for vision-based robots. This task, unlike monocular depth estimation, is not affected by scale ambiguity and has a compact output space (a unit sphere vs. positive real value), making it feasible to collect data that densely covers the output space. As a result, learning-based surface normal estimation methods show strong generalization capability for out-of-distribution images, despite being trained on relatively small datasets [1]. They are also efficient, running in real-time even on laptop computers.

Despite their essentially local property, predicted surface normals contain rich information about scene geometry. In recent years, their usefulness has been demonstrated in various areas of robotics and computer vision, including object grasping [2], multi-task learning [3], image generation [4], depth estimation [5], [6], simultaneous localization and mapping [7], human body shape estimation [8], [9], [10], and CAD model alignment [11]. The goal of this paper is to improve upon the existing methods by rethinking the inductive biases needed for the task.

## II. INDUCTIVE BIAS FOR SURFACE NORMAL ESTIMATION

### A. *Encoding per-pixel ray direction*

Under perspective projection, each pixel is associated with a ray that passes through the camera center and intersects the image plane at the pixel. Assuming a pinhole camera, a ray of unit depth for a pixel at $(u, v)$ can be written as

$$\mathbf{r}(u,v) = \begin{bmatrix} \frac{u-c_u}{f_u} & \frac{v-c_v}{f_v} & 1 \end{bmatrix}^T, \quad (1)$$

where $f_u$ and $f_v$ are the focal lengths and $(c_u, c_v)$ are the pixel coordinates of the principal point.

Per-pixel ray direction is essential for surface normal estimation. For rectangular structures (e.g. buildings), we can identify sets of parallel lines and their respective vanishing points. The ray direction at the vanishing point then gives

*This work was supported by Dyson Technology Ltd.

[1] Gwangbin Bae and Andrew J. Davison are with the Dyson Robotics Lab and the Department of Computing, Imperial College London [g.bae, a.davison]@imperial.ac.uk

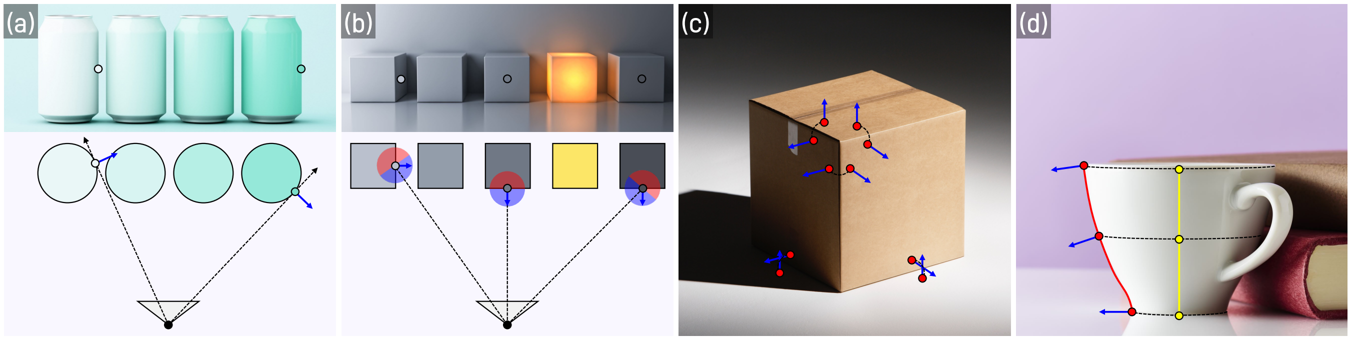

Fig. 2. **Motivation.** In this paper, we propose to utilize the *per-pixel ray direction* and estimate the surface normals by learning the *relative rotation between nearby pixels*. **(a)** Ray direction serves as a useful cue for pixels near occluding boundaries as the normal should be perpendicular to the ray. **(b)** It also gives us the range of normals that would be visible, effectively halving the output space. **(c)** The surface normals of certain scene elements (in this case, the floor) may be difficult to estimate due to the lack of visual cues. Nonetheless, we can infer their normals by learning the pairwise relationship between nearby normals (e.g. which surfaces should be perpendicular). **(d)** The relative angle between the normals of the yellow pixels can be inferred from that of the red pixels by assuming circular symmetry.

us the 3D orientation of the lines and hence the surface normals [12]. Early works on single-image 3D reconstruction [13], [14], [15], [16] made explicit use of such cues.

Now consider an occluding boundary created by a smooth (i.e. infinitely differentiable) surface. The surface normals at an occluding boundary can be determined uniquely by forming a generalized cone [17] (whose apex is at the camera center) that intersects the image plane at the boundary. In other words, the normals at the boundary should be perpendicular to the ray direction (see Fig. 2-a). To this end, we compute the per-pixel ray direction — i.e. Equation 1 — and provide this as an additional input to the network.

### B. Ray ReLU activation

The ray direction also decides the range of normals that would be *visible* in that pixel, effectively halving the output space (see Fig. 2-b). To incorporate such a bias, we propose a ray direction-based activation function analogous to ReLU. Given the estimated normal $\mathbf{n}$ and ray direction $\mathbf{r}$ (both are normalized), the activation can be written as

$$\sigma_{\text{ray}}(\mathbf{n},\mathbf{r}) := \frac{\mathbf{n} + (\min(0,\mathbf{n}\cdot\mathbf{r}) - \mathbf{n}\cdot\mathbf{r})\,\mathbf{r}}{\|\mathbf{n} + (\min(0,\mathbf{n}\cdot\mathbf{r}) - \mathbf{n}\cdot\mathbf{r})\,\mathbf{r}\|}. \tag{2}$$

Equation 2 ensures that $\mathbf{n}\cdot\mathbf{r} = \cos\theta$ (i.e. the magnitude of $\mathbf{n}$ along $\mathbf{r}$) is less than or equal to zero. The rectified normal is then re-normalized to have a unit length.

### C. Modeling inter-pixel constraints

The surface normals of two pixels are related by a 3D rotation matrix, $R$. We first represent $R$ using the *axis-angle* representation, $\boldsymbol{\theta} = \theta\mathbf{e}$, where a unit vector $\mathbf{e}$ represents the axis of rotation and $\theta$ is the angle of rotation.

Within flat surfaces (which are prevalent in man-made scenes/objects), $\theta$ would be zero and $R$ would simply be the identity. In a typical indoor scene, the surfaces of objects are often perpendicular or parallel to the ground plane, creating lines across which the normals should rotate by 90° (see Fig. 2-c). For a curved surface, the relative angle between the pixels can be inferred from the occluding boundaries by assuming a certain level of symmetry (see Fig. 2-d).

We encode such inter-pixel constraints by learning the relative rotation with respect to the neighboring pixels. For pixel $i$, we can define its local neighborhood $\mathcal{N}_i = \{j : |u_i - u_j| \le \beta \text{ and } |v_i - v_j| \le \beta\}$ ($\beta$ is set to 2). We can then learn the pairwise relationship between the surface normals $\mathbf{n}_i$ and $\mathbf{n}_j$ in the form of a rotation matrix $R_{ij}$.

For each pair of pixels, three quantities should be estimated: First is the angle $\theta_{ij}$ between the two normals. This is easy to learn as $\theta_{ij}$ is independent of the viewing direction and is $0^\circ$ or $90^\circ$ for many pixel pairs. Secondly, we need to estimate the axis of rotation $\mathbf{e}_{ij}$ (i.e. a 3D unit vector around which the normals rotate). While directly learning $\mathbf{e}_{ij}$ requires complicated 3D reasoning, we propose to learn the *2D projection* of $\mathbf{e}_{ij}$ on the image plane. As image intensity tends to change sharply near the intersection between two locally flat surfaces, this task can be as simple as edge detection. Then, we recover its 3D orientation by ensuring that $\mathbf{e}_{ij}$ is perpendicular to $\mathbf{n}_j$.

After rotating the neighboring pixels' normal vectors, we fuse them using a learnable set of weights $\{w_{ij}\}$. The updated normal of pixel $i$ is thus written as

$$\begin{aligned}\mathbf{n}_i^{t+1} &= \frac{\sum_j w_{ij}\sigma_{\text{ray}}(R_{ij}\mathbf{n}_j^t, \mathbf{r}_i)}{\|\sum_j w_{ij}\sigma_{\text{ray}}(R_{ij}\mathbf{n}_j^t, \mathbf{r}_i)\|} \\ R_{ij} &= \exp\left(\theta_{ij}[\mathbf{e}_{ij}]_\times\right).\end{aligned} \tag{3}$$

where the proposed ray-ReLU activation is used to ensure that the rotated normals are in the visible range for the target pixel $i$. We also added a superscript for the normals to represent an iterative update.

### D. Network architecture

We use a light-weight CNN with a bottleneck recurrent unit. The architecture is the same as that of [6] except for the quantities that are estimated from the updated hidden state. The number of surface normal updates $N_{\text{iter}}$ is set to 5, as it gave a good balance between accuracy and computational efficiency. As a result, each forward pass returns $N_{\text{iter}} + 1$ predictions (initial prediction obtained via direct regression + $N_{\text{iter}}$ updates). We then apply convex upsampling [19] to

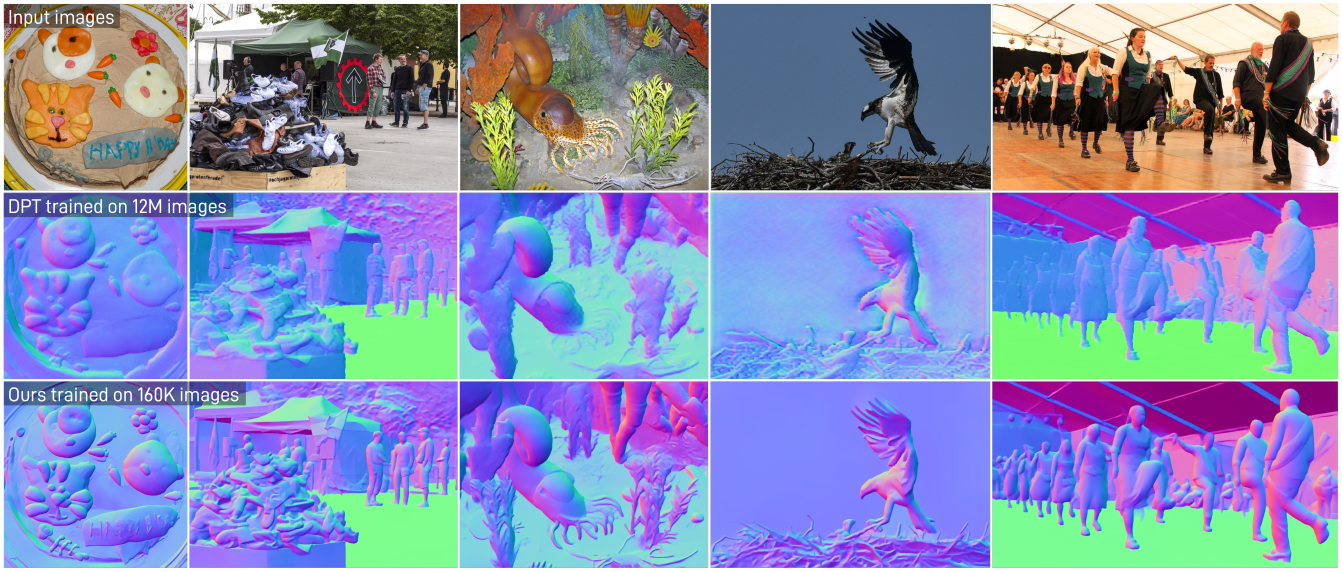

Fig. 3. **Comparison to Omnidata v2 [18].** Our method shows a stronger generalization capability for challenging in-the-wild objects. For textureless regions (e.g. sky in the fourth column), our model resolves any inconsistency in the prediction and outputs a flat surface, while preserving sharp boundaries around other objects.

recover full-resolution outputs. The network is trained by minimizing the weighted sum of their angular losses. The loss for pixel $i$ can be written as

$$\mathcal{L}_i = \sum_{t=0}^{N_{\text{iter}}} \gamma^{N_{\text{iter}}-t} \cos^{-1}(\mathbf{n}_i^{\text{gt}} \cdot \mathbf{n}_i^t) \tag{4}$$

where $0 < \gamma < 1$ puts a bigger emphasis on the final prediction. We set $\gamma = 0.8$ following RAFT [19].

### E. Dataset and training

The model is trained on a meta-dataset, consisting of 160K images sampled from a set of RGB-D datasets [20], [21], [22], [23], [24], [25], [26], [27], [28], [18]. Our dataset, compared to Omnidata [18], has a similar number of scenes (1655 vs. 1905) but a significantly smaller number of images (160K vs. 12M).

The network is trained for 5 epochs. We use the AdamW optimizer [29] and schedule the learning rate using 1cycle policy [30] with $lr_{\text{max}} = 3.5 \times 10^{-4}$. The batch size is set to 4 and the gradients are accumulated every 4 batches. The training approximately takes 12 hours on a single NVIDIA 4090 GPU.

## III. Results

In Figure 3, we compare the generalization performance of our method to that of Omnidata v2 [18], [31], on challenging in-the-wild images from OASIS [32]. Omnidata v2 is a transformer architecture [33] trained on 12 million images. Despite being trained on an orders of magnitude smaller dataset, our method shows stronger generalization performance with a significantly higher level of detail.

One notable advantage of our method over ViT-based models (e.g. [31]) lies in the simplicity and efficiency of network training. For example, Omnidata v2 [31] was trained for 2 weeks on four NVIDIA V100 GPUs. A set of sophisticated 3D data augmentation functions [31] were used to improve the generalization performance and cross-task consistency [34] was enforced by utilizing other ground truth labels. On the contrary, our model can be trained in just 12 hours on a single NVIDIA 4090 GPU, does not require geometry-aware 3D augmentations, and does not require any additional supervisory signal. Our model also has 40% fewer parameters compared to [31] (72M vs 123M).

## IV. Conclusion

In this paper, we discussed the inductive biases needed for surface normal estimation and introduced how per-pixel ray direction and the relative rotational relationship between neighboring pixels can be encoded in the output. Per-pixel ray direction allows camera intrinsics-aware inference and thus improves the generalization ability, especially when tested on images taken with out-of-distribution cameras. Explicit modeling of inter-pixel constraints — implemented in the form of rotation estimation — leads to piece-wise smooth predictions that are crisp near surface boundaries.

Compared to a recent transformer-based state-of-the-art method, our method shows stronger generalization capability and a significantly higher level of detail in the prediction, despite being trained on an orders of magnitude smaller dataset. Thanks to its fully convolutional architecture, our model can be applied to images of arbitrary resolution and aspect ratio, without the need for image resizing or positional encoding inter/extrapolation. We believe that the domain- and camera-agnostic generalization capability of our method makes it a strong front-end perception that can benefit many downstream 3D computer vision tasks.

# References

[1] Gwangbin Bae, Ignas Budvytis, and Roberto Cipolla. Estimating and exploiting the aleatoric uncertainty in surface normal estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[2] Guangyao Zhai, Dianye Huang, Shun-Cheng Wu, HyunJun Jung, Yan Di, Fabian Manhardt, Federico Tombari, Nassir Navab, and Benjamin Busam. Monograspnet: 6-dof grasping with a single rgb image. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[3] Shikun Liu, Linxi Fan, Edward Johns, Zhiding Yu, Chaowei Xiao, and Anima Anandkumar. Prismer: A vision-language model with multi-task experts. *Transactions on Machine Learning Research*, 2024.

[4] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[5] Xiaojuan Qi, Renjie Liao, Zhengzhe Liu, Raquel Urtasun, and Jiaya Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[6] Gwangbin Bae, Ignas Budvytis, and Roberto Cipolla. Irondepth: Iterative refinement of single-view depth using surface normal and its uncertainty. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2022.

[7] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. In *International Conference on 3D Vision (3DV)*, 2024.

[8] Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J Black. Icon: Implicit clothed humans obtained from normals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[9] Yuliang Xiu, Jinlong Yang, Xu Cao, Dimitrios Tzionas, and Michael J Black. Econ: Explicit clothed humans optimized via normal integration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[10] Oliver Boyne, Gwangbin Bae, James Charles, and Roberto Cipolla. Found: Foot optimization with uncertain normals for surface deformation using synthetic data. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024.

[11] Florian Langer, Gwangbin Bae, Ignas Budvytis, and Roberto Cipolla. Sparc: Sparse render-and-compare for cad model alignment in a single rgb image. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2022.

[12] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[13] Youichi Horry, Ken-Ichi Anjyo, and Kiyoshi Arai. Tour into the picture: using a spidery mesh interface to make animation from a single image. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 225–232, 1997.

[14] Jana Košecká and Wei Zhang. Extraction, matching, and pose recovery based on dominant rectangular structures. *Computer Vision and Image Understanding*, 100(3):274–293, 2005.

[15] David C Lee, Martial Hebert, and Takeo Kanade. Geometric reasoning for single image structure recovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[16] David Ford Fouhey, Abhinav Gupta, and Martial Hebert. Unfolding an indoor origami world. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[17] David Marr. Analysis of occluding contour. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 197(1129):441–475, 1977.

[18] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[19] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[20] Shreeyak Sajjan, Matthew Moore, Mike Pan, Ganesh Nagaraja, Johnny Lee, Andy Zeng, and Shuran Song. Clear grasp: 3d shape estimation of transparent objects for manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[21] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019.

[22] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[23] Yuan-Ting Hu, Jiahong Wang, Raymond A Yeh, and Alexander G Schwing. Sail-vos 3d: A synthetic dataset and baselines for object detection and 3d mesh reconstruction from video data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[24] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[25] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[26] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[27] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[28] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[29] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[30] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of residual networks using large learning rates. *arXiv preprint arXiv:1708.07120*, 2018.

[31] Oğuzhan Fatih Kar, Teresa Yeo, Andrei Atanov, and Amir Zamir. 3d common corruptions and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[32] Weifeng Chen, Shengyi Qian, David Fan, Noriyuki Kojima, Max Hamilton, and Jia Deng. Oasis: A large-scale dataset for single image 3d in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[33] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[34] Amir R Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J Guibas. Robust learning through cross-task consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.