Augmented Reality for Users with Low Vision

Yannick Schrempp
Karlsruher Institute of Technology
Kaiserstraße 12, 76131 Karlsruhe
yannick.schrempp@student.kit.edu

Ting Zhu
Karlsruher Institute of Technology
Kaiserstraße 12, 76131 Karlsruhe
ting.zhu@student.kit.edu

Eike Heinz
Karlsruher Institute of Technology
Kaiserstraße 12, 76131 Karlsruhe
eike.heinz@student.kit.edu

Abstract

In this paper we use an AR device, specifically the HoloLens 2, to generate and display objects from the real world as 3D holograms. Our approach outsources computationally intensive workloads to reduce the overall runtime, keep a high accuracy and thus improve the user experience. To achieve this, we use Unity with the Mixed Reality Toolkit of Microsoft to create the AR app and a server software to perform instance segmentation and shape estimation using Mask R-CNN and Mesh R-CNN respectively.

1. Introduction

Augmented Reality (AR) is a growing field of research and is also finding more and more use in industrial applications. An exemplary use case of AR is the prototyping and visualization of buildings and industrial products. Even in everyday life AR might become a useful supporting tool due to the recent rise of Artificial Intelligence (AI) and neural networks which might be a supporting technology. The increasing speed of hardware resources might enable AR glasses like the HoloLens to calculate complex transfer functions of Convolutional Neural Networks (CNNs) in the future. This can be the entry of AR in everyday life and a supporting tool to investigate scenes for people with low vision.

2. Related work

Since our project contains separate work from instance segmentation and 3D reconstruction topics, we will start to provide related work to those two topics in the following section. In the end we will a short overview of the relevant references to AR projects with respect to those topics.
When it comes to the evaluation of instance segmentation models, the most common benchmark dataset that is used in the community is the MS COCO dataset [16]. While Mask R-CNN [11] was one of the best performing models for a long time since 2017 in the recent years some architectures were developed which increased the mean Average Precision (mAP) slightly on MS COCO. Some architectures to be mentioned regarding this are SOLOv2 [22], PANet [17] and GCNet [3]. The YOLACT [2] architecture significantly improved inference time and was among the first architectures to achieve real time inference for instance segmentation on the GPU Titan XP with 33 frames per second [9].
Using RGBD data is a less explored field of research in instance segmentation. Nevertheless publications that use synthetic RGBD data for custom datasets or special applications exist [24], [8], [10]. Song et. al. [20] provided the first object detection and semantic segmentation benchmark on the SUN dataset which contains depth information. Further work on 3D bounding box estimation or semantic segmentation on this dataset is available [25], [7]. But until writing this paper there was no instance segmentation work based on this dataset. The biggest datasets for instance segmentation like MSCOCO do not contain depth information and a clear statement regarding quantitative improvements when using the depth channel cannot be made.

Compared with the maturity in 2D object recognition, 3D shape prediction is still in the exploratory and most of the state-of-the-art methods rely on deep neural networks. These methods have different classification criteria. According to 3D shape representation, the methods include voxels, point clouds and meshes. In terms of the input data type, the methods can be roughly divided into the following types: Single-View Shape Prediction uses a single image as input for 3D reconstruction, and one state-of-the-art method is Mesh R-CNN [6]. The input of Multi-View Shape Prediction methods are multiple images or a video of an object, a representative method for it is Recurrent MVSNet [27]. If the dataset contains depth information in addition to RGB information, 3D input methods such as KinectFusion [12] can be used.

Research using AR is still a growing field of research. There are several publications tackling the problem of augmenting the real world using holograms in various ways and scenarios. Joachimczak et. al. [14] perform real-time 3D reconstruction of persons and objects, by transmitting the completed frames as polygonial mesh to the HoloLens. An older approach by Yang et. al. [26] uses classic non-deep-learning approaches of image analysis to create, display and track holograms in AR.

Much research has been done to aid surgeons in various ways. Pratt et. al. [18] use HoloLens to display additional information obtained via computed tomography angiography during extremity reconstruction surgery. Whereas [5] aims to help surgeons navigate via visualizations during aortic repair by simultaneously reducing side effects from radiation exposure and contrast agent administration.

3. Instance segmentation for preprocessing

The goal of training an instance segmentation model was to preprocess the input data from the Hololens so the 3D reconstruction model only needs to consider a single segmented object. Furthermore the 3D reconstruction task already has the information on which class the reconstruction process shall be done which is a big advantage during the training process. The quality of the segmentation should be as high as possible to avoid providing bad input data for the 3D reconstruction model.
Our approach to leverage the depth channel is to simply stack the depth channel onto the RGB channel in the input dimension of Mask R-CNN. Using this technique, we were still able to do transfer learning on the pretrained weights from MS COCO to achieve fast convergence.

3.1. Data and dimensionality reduction

Since the Hololens is able to provide RGBD images we decided to use a dataset which contains a depth channel. We choose to use the SUN dataset which contains the NYU depth v2 [19], Berkeley B3DO [13] and SUN3D [23] datasets. In total this dataset contains 10335 RGBD images and we trained models with the RGBD images and others with RGB only to compare the two approaches.
Since the dataset is composed of several small datasets, the number of different classes is very huge and the labels do not have a common origin what resulted in over 14000 different classes on the raw labels. Since the goal of our application is to do reconstruction of indoor objects, we decided to focus on a common set of indoor classes for the

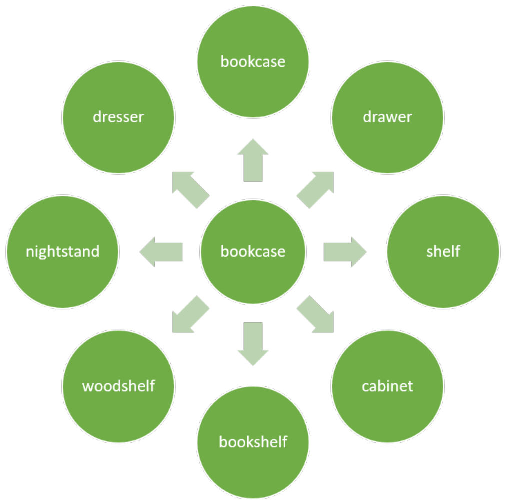

Figure 1. Results from the Word2Vec matching for the bookcase class.

task of the 3D reconstruction part and the instance segmentation part. The data which was used for the 3D reconstruction part is further examined in section 4.1. Since the data that was chosen for the object reconstruction task contained only 5 different classes, we used a Word2Vec approach to merge very similar classes into one class. Using a pretrained model on linguistic features[1] we were able to transform the classes into a vector and measure the distance between different classes which was considered as similarity between those. After calculating the similarity between all predefined indoor classes and every class in the SUN dataset we applied an empirical threshold to filter the good matches. Some qualitative results are shown in Figure 1. With respect to the existent classes in the dataset we used in section 4, the following classes remained in the dataset: bookcase, table, chair, sofa and bed.

3.2. Instance segmentation results

The implementation of Mask R-CNN we used to solve the instance segmentation task is the widely used Keras Mask R-CNN implementation [1]. We decided to go with this architecture since it is well documented and a lot of information is available about altering the architecture to train on RGBD data. The overall performance of Mask R-CNN is still not outdated and if performance improvements are necessary the modular structure of our application allows to exchange Mask R-CNN for another architecture in the future.
Some qualitative results of the instance segmentation model trained on RGBD data can be seen in Figure 3.2. The mAP and Intersection over Union (IoU) scores per class for the best RGB and RGBD model can be seen in table 3.2. The best weighted mAP when using the depth channel is 4.4%

[1]From https://spacy.io/models

class name	mean mAP (best RGB)	mean mAP (best RGBD)	mean IoU (best RGB)	mean IoU (best RGBD)
bed	0.331	0.285	0.351	0.295
chair	0.367	0.40	0.453	0.475
table	0.258	0.335	0.316	0.377
sofa	0.132	0.147	0.183	0.2
bookcase	0.100	0.143	0.141	0.189

Table 1. mAP and mean IoU values for every class in the SUN-RGBD dataset for RGB and RGBD images.

higher than the best model without using the depth channel. The significant difference might be even higher if the training would be done from scratch since the loaded weights for transfer learning were not optimized using the depth channel and the first CNN layer for the depth channel had to be initialized randomly. If there were bigger datasets providing a depth channel it might be an interesting approach to train a model with depth channel from scratch to even achieve better results.

Augmentation

To reduce overfitting we applied the augmentation strategy of Jung et. al. [15] which offers to provide strength and number of augmentations per image. We managed to apply the augmentation to RGB as well as RGBD data. The results for both types of input images for different augmentation configurations can be seen in table 2.

Augm. number	Augm. strength	mAP (RGB)	mAP (RGBD)
0	0	0.252	0.264
2	3	0.237	0.296
3	7	0.226	0.288

Table 2. mAP values for input data with different augmentation strategies.

4. Object reconstruction from 2D images

The goal of shape estimation is to convert the output of instance segmentation to a 3D model and convert it into a transferable format.

After comparing different 3D reconstruction methods, we choose Mesh R-CNN to implement shape estimation for the following reasons: (1) No dataset contains both 3D annotations and corresponding depth information. For this reason, methods requiring 3D input are not considered. (2) Due to the uncertainty about the functionality and performance of Microsoft HoloLens 2 and Unity with the Mixed Reality Toolkit (MRTK), we tend to choose the most straightforward scenario.

4.1. Dataset

In order to predict 3D shapes from 2D images using supervised learning methods, the dataset should contain 3D annotations and images of the objects.

Compared to the richness and diversity of 2D image datasets, few datasets contain 3D annotation. The only datasets that meet our requirements are ShapeNet [4] and Pix3D [21].

ShapeNet is a huge dataset and its subset ShapeNetCore covers 55 common object categories with about 51,300 unique 3D models. Despite the huge amount of annotated data, we could not use it to train the model since all samples are synthetic and all objects are in isolation. These reasons lead to a strong contrast to the indoor scenes (natural, possibly cluttered) that we deal with.

Pix3D provides image-shape pairs with precise pixel-level 2D to 3D alignment. It has 3D information on only 395 objects, but they are obtained from the real world and with a natural background. Most of the items are IKEA furniture, which also meets the needs of this project focused on indoor scenes.

In the Pix3D dataset, the amount of training samples for several classes is too small (less than 20), so we remove ‘tool’, ‘misc’, and ‘wardrobe’ to reduce noise. We also merge ‘desk’ into ‘table’, as they are similar in appearance. Table 3 shows the amount of processed data.

Class name	Number of images	Number of models
Bed	994	20
Bookcase	361	17
Chair	3839	221
Sofa	1947	20
Table	2570	86
Total	9711	364

Table 3. Number of images in the Pix3D dataset.

4.2. Architecture

An overview of Mesh R-CNN is shown in Figure 3.

Mesh R-CNN extends Mask R-CNN [11] with a mesh prediction branch that outputs triangle meshes. This feature also allows it to be better integrated with instance Segmentation as described above. The core of Mesh R-CNN is a Voxel branch which is used to predict 3D meshes compared to the Mask branch of Mask R-CNN.

The first step of the Voxel Branch is to predict a grid of voxel occupancy probabilities giving the coarse 3D shape of the object. Similar to Mask R-CNN, which predicts $M \times M$

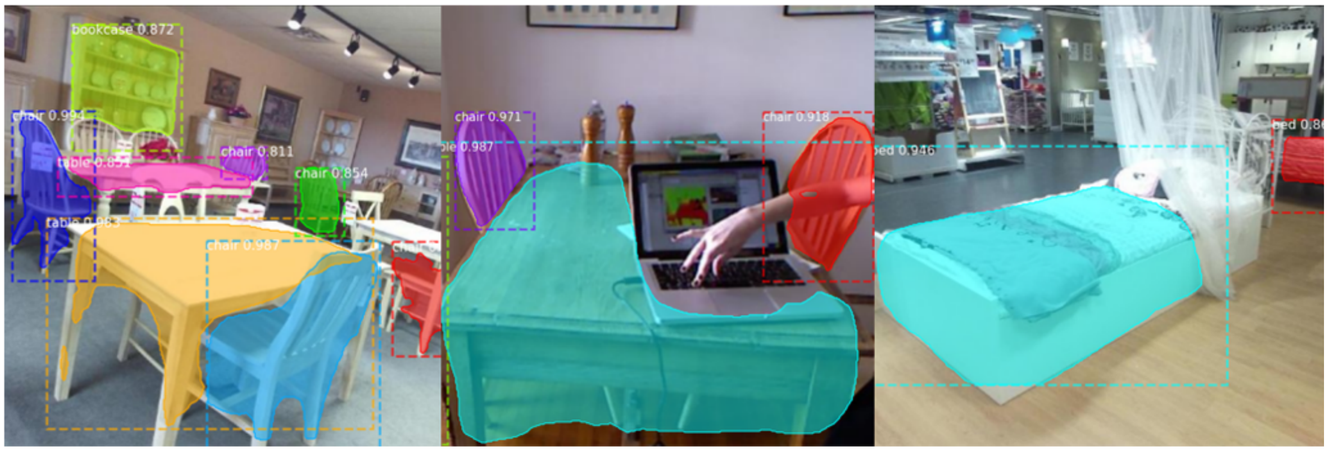

Figure 2. Sample detections of best Mask R-CNN model with depth input data

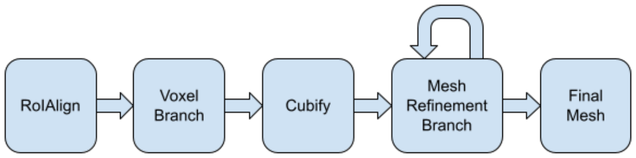

Figure 3. System overview of Mesh R-CNN.

grid 2D shape, a $G \times G \times G$ grid 3D shape is predicted in Mesh R-CNN.

After predicting coarse 3D voxels of an object, an operation called Cubify is required to convert the voxel to triangles. Each cuboid triangle contains 8 vertices, 18 edges and 12 faces and shared vertices and edges are merged. This step is used to prepare for the refinement of the mesh.

The last step is Mesh Refinement, which consists of three sub-steps:

- Vertex alignment extracts image features for vertices.
- Graph convolution propagates information along mesh edges.
- Vertex refinement then updates vertex positions.

In order to get a more refined 3D model, Mesh Refinement needs to be repeated three times.

4.3. Shape Estimation Results

The implementation of Mesh R-CNN we used to solve the shape estimation task is the paper's original implementation. We made some modifications: avoiding disk reads and writes to fit our in-memory pipeline; using the RoI output from the Instance Segmentation task as one of the input parameters so that the shape estimation task only needs to predict a single object at a time.

After removing 'misc', 'tool' and 'wardrobe' and merging 'desk' into 'table', we randomly divide the remaining Pix3D data into two parts: 7264 images for training and 2445 images for testing.

Class name	AP^{mesh}
Bed	0.506
Bookcase	0.681
Chair	0.456
Sofa	0.730
Table	0.586
Total	0.592

Table 4. AP^{mesh} for every class in the Pix3D dataset.

We train for 12 epochs with a batch size of 64 per image on 8 K80 GPUs. The results are shown in Table 4. We use the metric called AP^{mesh} from the Mesh R-CNN paper to evaluate the result. AP^{mesh} is the mean area under the per-category precision-recall curves for $F1^{0.3}$ at 0.5.

For a more visual presentation, an example is shown in Figure 4.

5. Combination of the models and results

Although there is a so-called *Research Mode* available on the HoloLens 2 which allows direct access to the raw depth sensor and camera data, we opted to use the other approach and implement the task using a Universal Windows Platform (UWP) App built with Unity and take advantage of the Microsoft Mixed Reality Toolkit (MRTK).

5.1. Unity App

Using the AirTap gesture available with the MRTK we trigger our whole Pipeline. Upon recognizing this gesture, we take a picture of the surrounding area using the integrated webcam and also retrieve the camera transformation and projection matrices from the image. We can calculate the location of the pointer in the image space by applying the transformation and projection matrices accordingly.

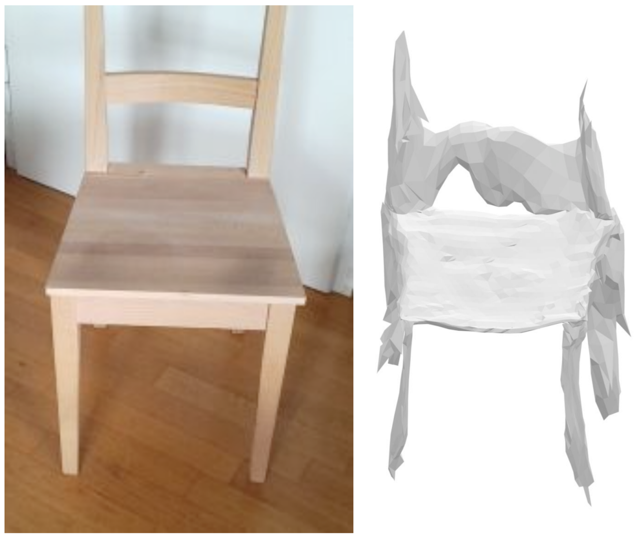
Figure 4. Sample result of Shape Estimation

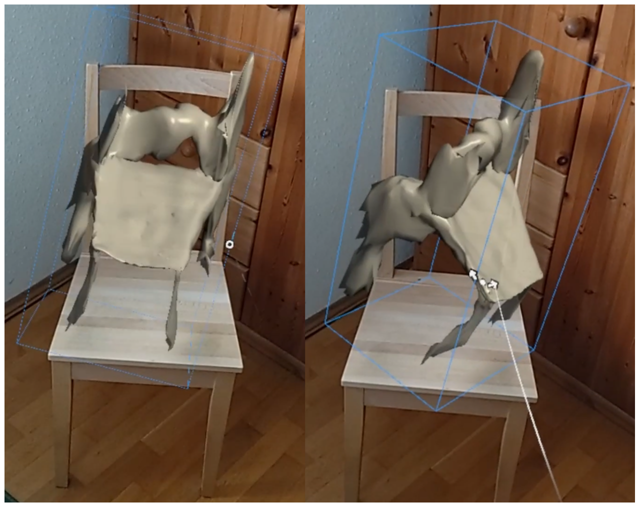
Figure 5. Screenshot from the Unity App during interaction with a generated 3D model of a chair.

Since we have to reconstruct the image on the server, we have to introduce delimiters to separate the individual pixels. This is implemented in parallel using the Unity Jobs System to speed up the calculations.

The app receives the generated 3D model as OBJ data and reconstructs the mesh using the vertices and face elements describing the object. Afterwards we attach MRTK scripts that allow the user to interact with the hologram. An example of the interaction with a 3D model generated by our pipeline can be seen in Figure 5.

5.2. Server

Since the hardware of the HoloLens 2 is not powerful enough to enable running the deep learning models inside the UWP App, we had to extract the computationally expensive instance segmentation and shape estimation. Therefore we send the preprocessed image along with the pointer location in the image space via WiFi to a server. There we reconstruct the image and perform the instance segmentation and estimate the shape of the objects using the methods described in section 3 and 4. We can then select the model to display using the pointer location received with the image. The selected model is then sent back to the Unity App for further processing.

Task	Time
Preprocessing on HoloLens 2	$\sim$ 13 seconds
Instance Segmentation (CPU)	$\sim$ 3 seconds
Shape Estimation (GPU)	$\sim$ 10 seconds
Postprocessing on HoloLens 2	$\sim$ 3 min
Total	$\sim$ 3 min 26 seconds

Table 5. Performance evaluation of our pipeline.

5.3. Performance measurements

Our pipeline does not allow real time interaction, since there is a significant processing delay introduced by several factors as we can see in Table 5. The preprocessing we had to implement requires 13 seconds, even with multithreading. Segmenting the image and estimating the 3d shape of the objects is mostly limited by the computational power of the server. In our experiments we ran the server software on a system with an AMD Ryzen 2700X, 64GB RAM and a Nvidia GeForce GTX 1080ti. Due to library version conflicts we were only able to run the Mesh R-CNN model on the GPU.

Most of the time is currently needed for the postprocessing on the HoloLens 2 which contains generating the mesh from the OBJ data received from the server. We are positive that performance improvements can be achieved here, by using a different approach to generate a mesh using OBJ data.

6. Conclusion and future work

In this paper we show that Augmented Reality wearables like the HoloLens 2 can be used to create and display generated 3D models obtained from the surrounding area. Due to current limitations performing complex calculations like instance segmentation and 3D shape estimation using neural networks is not feasible on the wearable itself. By offloading the computationally intensive workloads and improving the creation of meshes from OBJ data we believe a pipeline runtime of less than a minute is achievable.
We have also shown that by additionally using depth information to perform instance segmentation the accuracy of the detections can be significantly increased. Since HoloLens 2 does not allow access to the depth sensors unless using Research mode, an interesting approach might

be to use the Spatial Awareness Layer created by the MRTK.

Code available at:
Instance Segmentation:
`https://github.com/Yannick947/Mask_RCNN`
SUNRGBD preprocessing:
`https://github.com/Yannick947/data_generator_sunrgbd`
Shape Estimation:
`https://github.com/merlinz165/meshrcnn`
Unity App:
`https://github.com/EikeHeinz/CVHCI-HoloLens-app`
Server:
`https://github.com/EikeHeinz/CVHCI-HoloLens-server`

References

[1] Waleed Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow. `https://github.com/matterport/Mask_RCNN`, 2017.

[2] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation, 2019.

[3] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond, 2019.

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[5] Verónica García-Vázquez, Felix von Haxthausen, Sonja Jäckle, Christian Schumann, Ivo Kuhlemann, Juljan Bouchagiar, Anna-Catharina Höfer, Florian Matysiak, Gereon Hüttmann, Jan Peter Goltz, Markus Kleemann, Floris Ernst, and Marco Horn. Navigation and visualisation with hololens in endovascular aortic repair. *Innovative Surgical Sciences*, 3(3):167–177, 2018.

[6] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9785–9795, 2019.

[7] Joris Guerry, Alexandre Boulch, Bertrand Le Saux, Julien Moras, Aurelien Plyer, and David Filliat. Snapnet-r: Consistent 3d multi-view semantic labeling for robotics. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[8] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation, 2014.

[9] Abdul Mueed Hafiz and Ghulam Mohiuddin Bhat. A survey on instance segmentation: State of the art, 2020. Int J Multimed Info Retr (2020).

[10] Jungong Han, Ling Shao, Dong Xu, and Jamie Shotton. Enhanced computer vision with microsoft kinect sensor: a review. 43(5):1318–1334, 2013. Journal Article Research Support, Non-U.S. Gov't.

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[12] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568, 2011.

[13] Allison Janoch, Sergey Karayev, Yangqing Jia, Jonathan T. Barron, Mario Fritz, Kate Saenko, and Trevor Darrell. *A Category-Level 3D Object Dataset: Putting the Kinect to Work*, pages 141–165. Springer London, London, 2013.

[14] Michal Joachimczak, Juan Liu, and Hiroshi Ando. Real-time mixed-reality telepresence via 3d reconstruction with hololens and commodity depth sensors. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ICMI '17, page 514–515, New York, NY, USA, 2017. Association for Computing Machinery.

[15] Alexander B. Jung, Kentaro Wada, Jon Crall, Satoshi Tanaka, Jake Graving, Christoph Reinders, Sarthak Yadav, Joy Banerjee, Gábor Vecsei, Adam Kraft, Zheng Rui, Jirka Borovec, Christian Vallentin, Semen Zhydenko, Kilian Pfeiffer, Ben Cook, Ismael Fernández, François-Michel De Rainville, Chi-Hung Weng, Abner Ayala-Acevedo, Raphael Meudec, Matias Laporte, et al. imgaug. `https://github.com/aleju/imgaug`, 2020. Online; accessed 05-Jan-2021.

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 01.05.2014. 1) updated annotation pipeline description and figures; 2) added new section describing datasets splits; 3) updated author list.

[17] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. *CoRR*, abs/1803.01534, 2018.

[18] Philip Pratt, Matthew Ives, Graham Lawton, Jonathan Simmons, Nasko Radev, Liana Spyropoulou, and Dimitri Amiras. Through the HoloLens™ looking glass: augmented reality for extremity reconstruction surgery using 3d vascular models with perforating vessels. *European Radiology Experimental*, 2(1), Jan. 2018.

[19] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, volume 7576 of *Lecture Notes in Computer Science*, pages 746–760. Springer Berlin Heidelberg, 2012.

[20] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 567–576, 2015.

[21] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018.

[22] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation, 2020.

[23] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. pages 1625–1632, 12 2013.

[24] Christopher Xie, Yu Xiang, Arsalan Mousavian, and Dieter Fox. The best of both modes: Separately leveraging rgb and depth for unseen object instance segmentation, 30.07.2019.

[25] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. Pointfusion: Deep sensor fusion for 3d bounding box estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[26] Ming-Der Yang, Chih-Fan Chao, Kai-Siang Huang, Liang-You Lu, and Yi-Ping Chen. Image-based 3d scene reconstruction and exploration in augmented reality. *Automation in Construction*, 33:48–60, 2013. Augmented Reality in Architecture, Engineering, and Construction.

[27] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent mvsnet for high-resolution multi-view stereo depth inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5525–5534, 2019.