The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLII-2/W17, 2019
6th International Workshop LowCost 3D – Sensors, Algorithms, Applications, 2–3 December 2019, Strasbourg, France

# INPAINTING OCCLUSION HOLES IN 3D BUILT ENVIRONMENT POINT CLOUDS

P. Väänänen[1,*], V. Lehtola[2]

[1] Umbra Software Ltd., Helsinki, Finland, pekka@umbra3d.com
[2] University of Twente, ITC faculty, Enschede, the Netherlands, ville.lehtola@iki.fi

**Commission II**

**KEY WORDS:** inpainting, point cloud, occlusion, indoor, built environment

**ABSTRACT:**

Point clouds obtained from mobile and terrestrial laser scanning are imperfect as data is typically missing due to occlusions. This problem is often encountered in 3D reconstruction and is especially troublesome for 3D visualization applications. The missing data may be recovered by intensifying the scanning mission, which may be expensive, or to some extent, by computational means. Here, we present an inpainting technique that covers these occlusion holes in 3D built environment point clouds. The proposed technique uses two neural networks with an identical architecture, applied separately for geometry and colors.

## 1. INTRODUCTION

Mobile laser scanning (MLS) and terrestrial laser scanning (TLS) are used to measure forest, urban, roadway, and indoor environments. The data captured from these is used for various applications such as surveying, visualization, or simulation (Lehtola et al., 2017). The strength of MLS and TLS, compared to airborne laser scanning for instance, is that the point clouds are very dense and very accurate. However, the exploitation of these data is hindered by the presence of occlusion that appear as shadows in the measured 3D point clouds, see e.g. (Ochmann et al., 2016, Nikoohemat et al., 2018). In simulation for machine learning, for example, this could hinder the training of agents for autonomous vehicles, because missing data can cause problems with generalization.

There are some techniques to prevent occlusions from happening. A straightforward way is to plan the scanning mission so that all relevant surfaces are covered by at least one scanning location. Unfortunately in built environments some areas will be out of limits for the scanner, such as rooftops of surrounding buildings when scanning at the street level. After a reasonable physical effort to improve the scanning geometry and trajectories, the remaining problem can be addressed computationally. One suitable technique for this is inpainting.

Inpainting is a well-known technique for reconstructing blank spots in 2D images (Bertalmio et al., 2000). In 3D, inpainting has been employed for patching ancient mutilated statues and other objects (Perez et al., 2012, Setty, Mudenagudi, 2018, Hu et al., 2019). Usually the scans represent single objects that have clearly defined interior and exterior volumes, so the depth images can be rendered from an outside perspective. In general, formulating the 3D inpainting problem in 2D allows a wide range existing techniques to be used. However, applying 2D inpainting methods to scans from built environment is not straightforward.

Built environment data is different from 3D objects because they represent disjoint surface regions with varying point densities. Poor scanning geometry, such as a long distance between the scanner and the target surface combined with a slope in the target surface, results in an inadequate sampling of some areas of the scan. Built environment data can also have holes because of windows and other structures that are not present in e.g. sculptures. These difficulties come on top of the normal occlusion effect, i.e., when something in front is casting a shadow on something in the background. Since built environment data are composed of multiple distinct regions, it could be helpful to classify the point cloud before processing it.

Classification of 3D point cloud data from MLS and TLS is a well-known problem (Weinmann et al., 2015, Grilli et al., 2017). Novel approaches in classification include machine learning (Qi et al., 2017) and direct processing of the unregistered LIDAR data (Lehtola et al., 2019). Hence, it is reasonable to assume that the point cloud has been semantically classified before the inpainting process.

In this paper, we propose a novel inpainting technique suitable for 3D built environment point clouds. Our technique incrementally fills surface holes one small round surface patch at a time. Each patch is reduced into a pair of depth and color images, which are then inpainted by a Convolutional Neural Network (CNN). We exploit the fact that large 3D scans have multiple well-sampled homogeneous regions that can be used to train the network, so the common problem of costly data acquisition can be avoided. The dataset used in this work is shown in Figure 1.

The paper is organized as follows. In Section 2 we briefly review the literature relevant to inferring missing areas in 2D and 3D data. Section 3 describes our inpainting procedure and the neural network architecture, followed by Section 4 where we explain our dataset construction in more detail. The results are visually examined in Section 5, concluded by a short summary in Section 6.

## 2. RELATED WORK

Disocclusion, i.e. repairing the occlusions, is a well known inpainting problem for 2D images (Tauber et al., 2007). When using multiple cameras to reconstruct 3D geometry, a typical

*Corresponding author

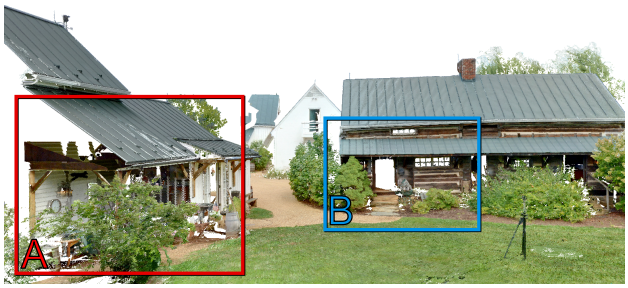


Figure 1. The 106 million point subset of the Pharsalia TLS dataset used for evaluation in this work. Segments Backyard (A) and Stairs (B) are annotated in the image.

cause for the occlusions is the movement of objects (or the camera) in which case the inpainting can be aided with depth information (Daribo, Pesquet-Popescu, 2010). Disocclusion in MLS or TLS is also related to the positioning and movement of objects and the sensor platform, but the occlusions manifest in 3D point clouds instead of 2D images.

However, since 2D data format is easier to handle, many inpainting techniques for 3D point clouds begin with taking a 2D range image snapshot from the 3D point cloud and then formulate the inpainting problem as if it was 2D. Example works include individual object hole filling with image inpainting of range images (Wang, Oliveira, 2003, Salamanca et al., 2008) and manifold triangle mesh inpainting using range images (Perez et al., 2012).

In addition to inpainting, small holes can be filled by using a smoothing operator and point resampling (Alexa et al., 2003), or by using radial basis functions (Ohtake et al., 2003). A review of 34 hole filling methods for triangle meshes can be found in (Pérez et al., 2016). More recently, machine learning techniques have gained interest. Deep learning is applicable for geometric inpainting e.g. by using a combination of discretized signed distance fields and range images (Han et al., 2017).

Even if the 2D inpainting problem is well-known, the formulation of the problem for 3D point clouds yet has some specific challenges. The first one is the 3D to 2D transformation that must be applied before inpainting. It is straightforward to render a range image by projecting the points to a plane, but how is the plane of projection chosen? In (Salamanca et al., 2008) the reference system of a 3D scanner is used, and in (Perez et al., 2012) the direction that maximizes the projected area of a hole is chosen. In (Roveri et al., 2018) the plane is computed by a neural network. If oriented surface normals are available then a tangent plane of a smoothed normal field can be used. That is the approach in our method.

The second challenge is hole identification, a more complex task for 3D point clouds than 2D images, because the edge of a point cloud cannot be determined from pixel coordinates. Therefore, distinguishing between the boundaries of the point cloud and the holes within the point cloud can be complex, especially if an occlusion is close to a boundary. For example in (Hu et al., 2019) a 3D point cloud is first projected to a single range image for hole detection, and image edges are used to distinguish between holes and the scan boundaries.

There is a strong tradition that only individual objects, e.g. human faces, antique vases, traffic signs etc. are used as an input to a hole filling algorithm (Pérez et al., 2016) or a machine learning model (Wu et al., 2015, Yang et al., 2017). This approach in an adapted form would make sense also in built environment, where the problem of point classification and segmentation is well known (Qi et al., 2017, Lehtola et al., 2019). That is, patches for the learning data would be sampled from specific pools of already labelled data.

## 3. METHOD

We propose a Generative Adversarial Network (GAN) (Goodfellow et al., 2014) based inpainting network to fill point cloud occlusion holes. The network operates on images of circular surface patches with missing areas, and generates inpainted images that are back-projected to the point cloud. We use uncorrupted scan regions as training data by performing artificial corruption operations on them. Therefore the method can be considered to store the patterns of the surrounding scan into the network weights, which are then used to infer the missing areas. We train separate models, $\mathbf{G}_{depth}$ and $\mathbf{G}_{rgb}$, for depth and color images, respectively, using the same architecture. See Figure 2.

We run the inpainting network only in the vicinity of surface boundaries. Surface boundary points are found using the angle criterion of (Bendels et al., 2006). The inpainted patches are clipped to a 2D convex hull of the original patch to avoid extending the surface outwards. The radius of each patch is 15 cm.

We implemented the inpainting model with the PyTorch library (Paszke et al., 2017) and exposed it as a plugin for the open source CloudCompare point cloud processing application[1].

### 3.1 Network architecture

The network architecture is similar to U-Net (Ronneberger et al., 2015), and it is identical for both color ($\mathbf{G}_{rgb}$) and depth ($\mathbf{G}_{depth}$) image models. We train the network with an adversarial technique similar to (Iizuka et al., 2017) using a simple five layer convolutional network as a discriminator. The loss function of the inpainting network is a combination of a per-pixel $L_1$ reconstruction loss and an adversarial hinge loss (Lim, Ye, 2017). We pre-train the network using only the $L_1$ loss, and add the adversarial loss once $L_1$ loss has converged. The reconstruction loss gives larger weight to pixels surrounding the hole, which has been noted to increase the output quality (Pathak et al., 2016).

## 4. DATA

We apply our method on the TLS Pharsalia dataset, a courtesy of Trimble Inc. The dataset was captured with a Trimble TX8 scanner and it represents a historic plantation house in Virginia, USA. The scan contains both built structures and flora as can be seen in Figure 1. Since the point cloud is very large, we crop a 106 million point subset for our evaluation. From this subset, we extract two smaller subsets that we name Stairs and Backyard. These subsets have large amounts of occlusion, and are therefore used for evaluation.

We extract 200,000 patches from the uncorrupted regions as $64 \times 64$ pixel depth and color images. The patches are used as training data for the neural network model. Similar to (Roveri et al., 2018), each point is interpolated with a truncated Gaussian.

[1] http://www.cloudcompare.org

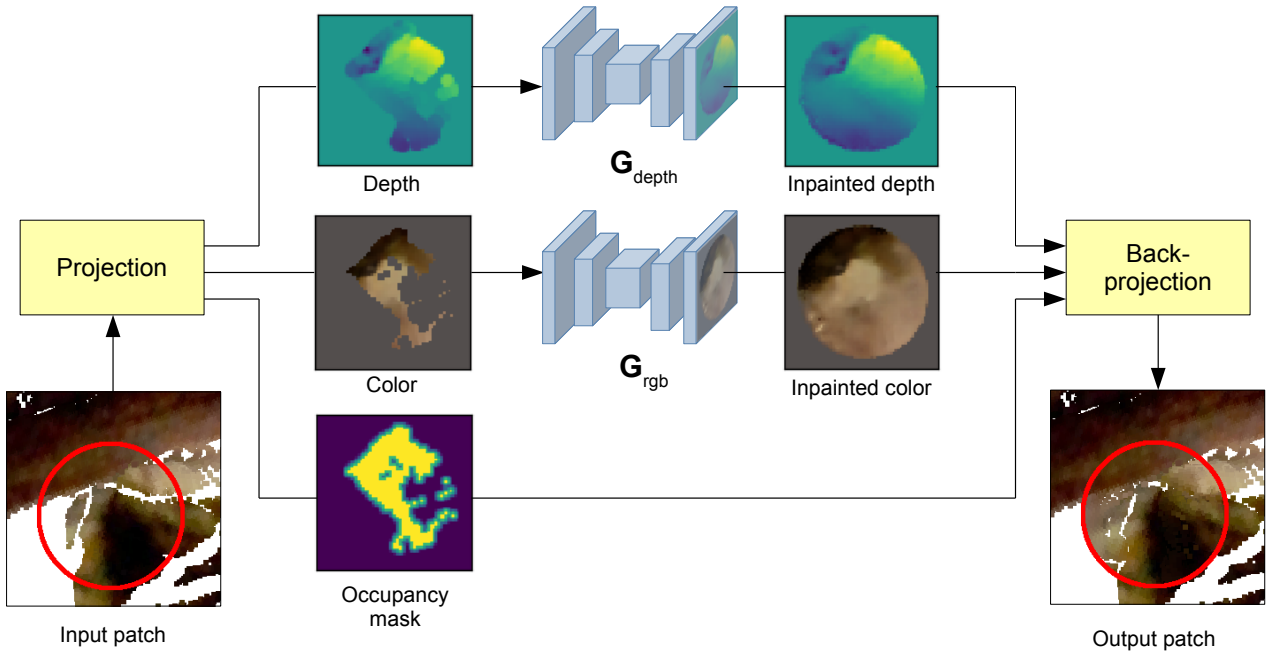


Figure 2. Processing a single surface patch. The points of a boundary region are first projected to into three images: a depth image, a color image, and an occupancy mask. The depth and color images are processed by the respective $\mathbf{G}_{depth}$ and $\mathbf{G}_{rgb}$ inpainting networks. The inpainted images are of the same dimension as the input. Finally, the pixels corresponding to the empty areas of the occupancy mask are back-projected to the point cloud.

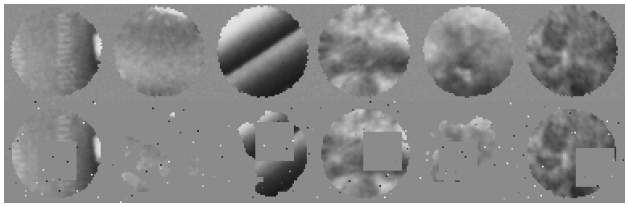

Figure 3. Examples of ground truth height maps (top row) and artificially corrupted images (bottom row) used as input data during training.

We synthesize the training data by adding artificial missing regions and noise to the ground truth patches. An example set of height map images is shown in Figure 3.

The data is assumed to have oriented normals, which in this case were computed using Principal Component Analysis (PCA) and oriented using scanner locations. We also assume a rudimentary classification has been performed, so that points belonging to the ground plane can be separated from individual objects.

Training the color and geometry models are done in two separate passes using the 200,000 patches. The patches, i.e. the training data, consists of four classes (Walls, Roofs, Flora, and Sand). We balance the classes during training by giving a smaller weight to errors in larger classes. This is equivalent in the expectation to resampling the dataset so that each class is of the same size. This weighting is done solely to balance the skew in the dataset, since the dataset contains large grass areas that would otherwise cause the network output to be biased (see Figure 1).

## 5. RESULTS

We treat the visual quality of the results as the main indicator of the validity of the proposed method. For thoroughness, we show results on two segments that have different properties. The first segment Stairs has smooth surface shapes and clear color textures, and the second segment Backyard is more complex and has more occlusion.

The input scan and the output result from our method are both shown, side by side, in Figures 4, 5 and 6, and in 3D at web[2].

Our method can reconstruct a plausible surface shape even in highly occluded regions, as can been seen in Figure 4. This is also shown in Figure 6 where the boundary of the lid is correctly extrapolated. The overall shape of the missing corner of a step in Figure 4 is reconstructed correctly, but the generated surface has varying point density. This is explored more in Section 5.1.

The neural network model produces plausible colors with no visible seams between the input and output surface areas. We hypothesize this is behavior is encouraged by the boundary weighted loss used during network training. We also find that the adversarial loss yields more detailed inpainting results than using just the per-pixel $L_1$ reconstruction loss. However, in smooth regions the difference is small, which indicates higher-level structure was already learned in the pre-training phase.

If the surrounding point colors are blurry, the model output degenerates into smooth interpolation as can be seen in Figure 7. Note how the hole boundary is not visible in the inpainting result, which would not have been the case if the model had hallucinated high frequency details.

Sometimes the lighting conditions vary in between the individual scans, which causes incompatible colors for neighboring

[2]Interactive point cloud visualizations are available through a weblink https://blog.umbra3d.com/blog/fixing-laser-scans-with-deep-learning

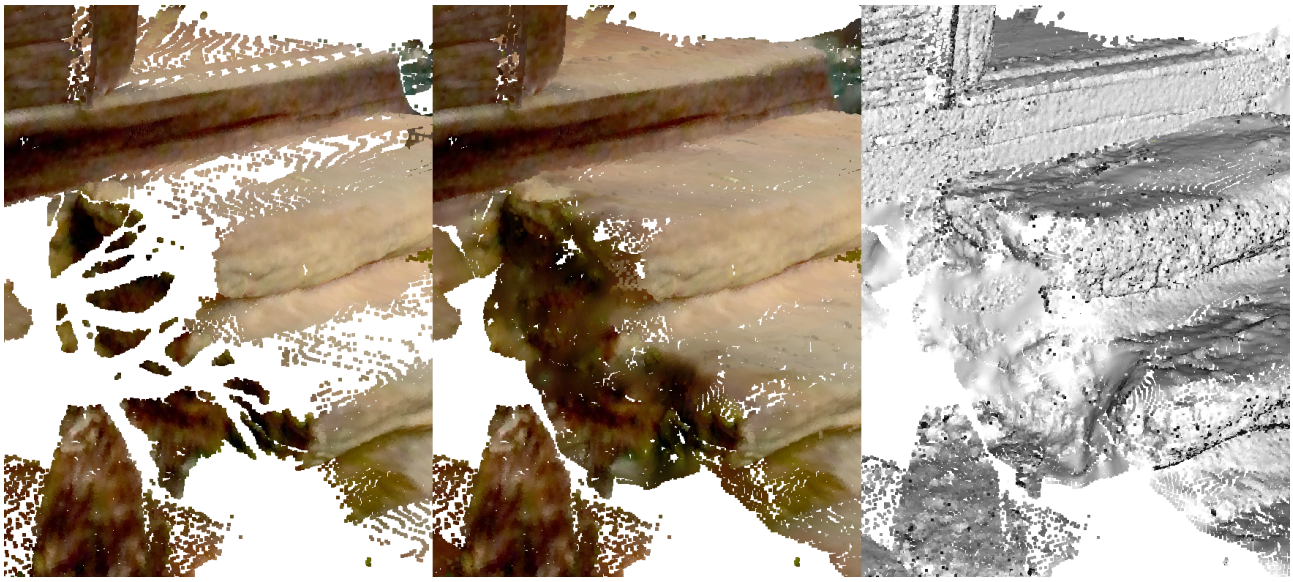

Figure 4. A close up of the Stairs segment. From left to right: the input scan, the inpainted colors, and estimated PCA normals of the inpainted surface. The dark outlier points were caused by the normal estimation. The visible cracks in the inpainted surface (middle) happen when the pixel grid of a back-projected depth image doesn't have sufficient resolution to capture sharp elevation changes.

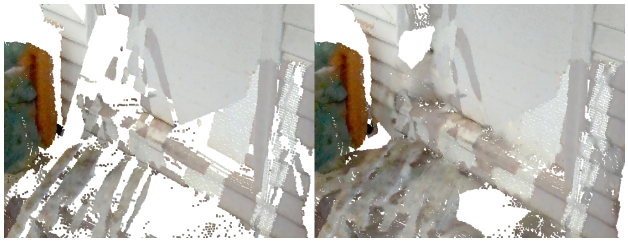

Figure 5. Inpainting the Backyard segment. Even though the input data (left) is of low quality, a plausible surface can be inferred in the output (right) point cloud.

Figure 6. A close-up view of input (left) and output (right) point cloud of a wooden barrel in the Backyard segment. Note how both the shape and colors of the generated surfaces match the surrounding scan.

Figure 7. An example of a large occlusion hole surrounded by poorly sampled colors in the Backyard segment. The colors of the input (left) are smoothly interpolated in the inpainting result (right). The bust visible in the image was processed separately. Points in the background have been removed for visual clarity.

points on the same surface. The input in Figure 5 is an example of such a case, but our method still produces continuous colors that are broken up into individual regions.

Processing each patch takes 20 ms on a system with NVIDIA GTX 1070 GPU and an Intel i7 7700K processor. A kd-tree lookup to gather the patch points was not included in the measurement. The inpainting step is straightforward to parallelize on the GPU, but was not pursued here since we processed patches serially in order to back-project the points between each inpainting step.

### 5.1 Limitations

While our method gives plausible results in many scenarios, some limitations remain. First of all, one assumption is that all blank pixels in the input image must be inpainted. This becomes problematic when some holes in the surface must be preserved. For example windows or pipes should not be modified by the inpainting network.

Also, the inpainting network operates only locally one surface patch at a time. This means all context must be inferred from

Figure 8. Color leaks in the Stairs segment. The input point cloud (left) has greenish tint around the occluded region because of projected colors captured by the RGB camera. In the inpainting result (right) the same tint has been propagated to the generated regions.

the patch image itself which in many cases is impossible. This could be partially remedied by passing class information directly to the network e.g. as a separate image.

The presence of noise can disturb normal estimation, which may cause some patches to be projected from a grazing angle. This produces varying point densities seen in Figure 4. In addition, our implementation back-projects each pixel as a single point which sometimes causes small cracks in the output. This is also visible in Figure 4.

Since input colors are assumed correct, the network will interpolate even erroneous colors. This is apparent in Figure 8 where colors of occluding shrubs have been projected to a wall.

Finally, training the neural network can be time consuming. This is made worse by the adversarial loss which, despite good results, introduces instability to the training process. In our implementation a combination of low learning rates, instance noise, and spectral normalization provided the required stabilization, but we expect advances in GAN training to solve this problem.

## 6. CONCLUSIONS

We have shown that a 3D point cloud inpainting system can be implemented as two neural networks with identical architectures trained on a large scan. Notably, the training does not require corrupted data, as it is done by artificially corrupting this single large scan. The scan is incrementally processed in small circular surface patches, making it possible to extract a large number of training samples. Given oriented normals, the method is local and can be parallelized.

The patch-based formulation of the inpainting problem allows us to leverage well-known deep learning methods, and results in visually plausible results. The proposed method is likely to be useful in 3D reconstruction, where the problem of missing data is encountered often. Unfortunately our formulation also limits the context available to the network. The limitations become apparent when the patch contains points from multiple distinct classes, which result in blurred colors and incorrect geometry. Future work may explore remedying these shortcomings with a more accurate point classification pass and more diverse training data.

## REFERENCES

Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., T. Silva, C., 2003. Computing and Rendering Point Set Surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1), 3–15. https://doi.org/10.1109/TVCG.2003.1175093.

Bendels, G. H., Schnabel, R., Klein, R., 2006. Detecting Holes in Point Set Surfaces. *Journal of WSCG*, 14.

Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C., 2000. Image inpainting. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 417–424.

Daribo, I., Pesquet-Popescu, B., 2010. Depth-aided image inpainting for novel view synthesis. *2010 IEEE International Workshop on Multimedia Signal Processing*, IEEE, 167–170.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative Adversarial Nets. Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger (eds), *Advances in Neural Information Processing Systems 27*, Curran Associates, Inc., 2672–2680.

Grilli, E., Menna, F., Remondino, F., 2017. A review of point clouds segmentation and classification algorithms. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42, 339.

Han, X., Li, Z., Huang, H., Kalogerakis, E., Yu, Y., 2017. High-Resolution Shape Completion Using Deep Neural Networks for Global Structure and Local Geometry Inference. *IEEE International Conference on Computer Vision (ICCV)*.

Hu, W., Fu, Z., Guo, Z., 2019. Local Frequency Interpretation and Non-Local Self-Similarity on Graph for Point Cloud Inpainting. *IEEE Transactions on Image Processing*.

Iizuka, S., Simo-Serra, E., Ishikawa, H., 2017. Globally and Locally Consistent Image Completion. *ACM Transactions on Graphics (Proc. of SIGGRAPH 2017)*, 36(4), 107.

Lehtola, V., Kaartinen, H., Nüchter, A., Kaijaluoto, R., Kukko, A., Litkey, P., Honkavaara, E., Rosnell, T., Vaaja, M., Virtanen, J.-P. et al., 2017. Comparison of the selected state-of-the-art 3D indoor scanning and point cloud generation methods. *Remote sensing*, 9(8), 796.

Lehtola, V. V., Lehtomäki, M., Hyyti, H., Kaijaluoto, R., Kukko, A., Kaartinen, H., Hyyppä, J., 2019. Preregistration Classification of Mobile LIDAR Data Using Spatial Correlations. *IEEE Transactions on Geoscience and Remote Sensing*.

Lim, J. H., Ye, J. C., 2017. Geometric GAN. arXiv pre-print arXiv:1705.02894 [stat.ML].

Nikoohemat, S., Peter, M., Oude Elberink, S., Vosselman, G., 2018. Semantic Interpretation of Mobile Laser Scanner Point Clouds in Indoor Scenes Using Trajectories. *Remote sensing*, 10(11), 1754.

Ochmann, S., Vock, R., Wessel, R., Klein, R., 2016. Automatic reconstruction of parametric building models from indoor point clouds. *Computers & Graphics*, 54, 94–103.

Ohtake, Y., Belyaev, A., Seidel, H. P., 2003. A multi-scale approach to 3d scattered data interpolation with compactly supported basis functions. *2003 Shape Modeling International.*, 153–161.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A., 2017. Automatic differentiation in PyTorch. *NIPS Autodiff Workshop*.

Pathak, D., Krähenbühl, P., Donahue, J., Darrell, T., Efros, A., 2016. Context encoders: Feature learning by inpainting. *CVPR*.

Perez, E., Salamanca, S., Cerrada, C., Merchn, P., Adan, A., 2012. Filling holes in manifold digitized 3d meshes using image restoration algorithms. *Proceedings of the IEEE Intelligent Vehicles Symposium Workshops*.

Pérez, E., Salamanca, S., Merchán, P., Adán, A., 2016. A Comparison of Hole-filling Methods in 3D. *Int. J. Appl. Math. Comput. Sci.*, 26(4), 885–903. https://doi.org/10.1515/amcs-2016-0063.

Qi, C. R., Su, H., Mo, K., Guibas, L. J., 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 652–660.

Ronneberger, O., P.Fischer, Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, LNCS, 9351, Springer, 234–241. (available on arXiv:1505.04597 [cs.CV]).

Roveri, R., Öztireli, A. C., Pandele, I., Gross, M., 2018. PointProNets: Consolidation of Point Clouds with Convolutional Neural Networks. *Computer Graphics Forum*, 37(2), 87-99. https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13344.

Salamanca, S., Industriales, E. D. I., Informatica, E. S. D. I., Cerrada, C., 2008. Filling holes in 3d meshes using image restoration algorithms. *Proceedings of the Fourth International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT'08)*.

Setty, S., Mudenagudi, U., 2018. Example-based 3D inpainting of point clouds using metric tensor and Christoffel symbols. *Machine Vision and Applications*, 29(2), 329–343.

Tauber, Z., Li, Z.-N., Drew, M. S., 2007. Review and preview: Disocclusion by inpainting for image-based rendering. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(4), 527–540.

Wang, J., Oliveira, M. M., 2003. A hole-filling strategy for reconstruction of smooth surfaces in range images. *16th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2003)*, 11–18.

Weinmann, M., Jutzi, B., Hinz, S., Mallet, C., 2015. Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. *ISPRS Journal of Photogrammetry and Remote Sensing*, 105, 286–304.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J., 2015. 3d ShapeNets: A deep representation for volumetric shapes. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1912–1920.

Yang, B., Wen, H., Wang, S., Clark, R., Markham, A., Trigoni, N., 2017. 3d Object Reconstruction from a Single Depth View with Adversarial Learning. *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, 679–688.