Solving Occlusion in Terrain Mapping with Neural Networks

Maximilian Stölzle[1,2,3], **Takahiro Miki**[1], **Levin Gerdes**[2], **Martin Azkarate**[2], **and Marco Hutter**[1*]

Abstract

Accurate and complete terrain maps enhance the awareness of autonomous robots and enable safe and optimal path planning. Rocks and topography often create occlusions and lead to missing elevation information in Digital Elevation Maps (DEMs). Currently, mostly traditional inpainting techniques based on diffusion or patch-matching are used by autonomous mobile robots to fill-in incomplete DEMs. These methods cannot leverage the high-level terrain characteristics and the geometric constraints of line of sight we humans use intuitively to predict occluded areas. We propose to use neural networks to reconstruct the occluded areas in DEMs. We introduce a self-supervised learning approach capable of training on real-world data without a need for ground-truth information. We accomplish this by adding artificial occlusion to the incomplete elevation maps constructed on a real robot by performing ray casting. We evaluate our self-supervised learning approach on several real-world datasets which were recorded during autonomous exploration of both structured and unstructured terrain with a legged robot, and additionally in a planetary scenario on Lunar analogue terrain. We state a significant improvement compared to the Telea and Navier-Stokes baseline.

1 Introduction

As we empower mobile robots to autonomously navigate to their target, they rely on maps of the surrounding environment for motion planning. 2.5D Digital Elevation Maps (DEMs) represent a memory-efficient approximation as they project the 3D structure into two-dimensional grid cells. To enable safe and optimal path planning, we strive for complete and accurate elevation maps. In practice, we often encounter occlusions caused by terrain discontinuities such as rocks, obstacles or convex terrain characteristics which hide an area patch from the sensor's viewpoint. Further, depth measurements can be degraded due to reflections, stereo matching failures, dust, or textureless surfaces which frequently lead to additional missing elevation information in the DEM.

Existing implementations to estimate the elevation of the occluded areas rely on traditional inpainting methods for photo-realistic images such as Navier-Stokes [1], Telea [2] or on searching terrain patches with close resemblance in an offline library [3]. Although these methods are able to reconstruct meaningful information about the occluded areas as we show in Section 3, they were developed and tuned with applications for inpainting of RGB camera images in mind. Assumptions such as that the unobserved data is missing randomly do not apply for terrain maps [3].

We are inspired by the application of data-driven methods for inpainting of photo-realistic images [4–7] even for irregular holes [4] and aim to tailor these methods for the application of filling occlusion in terrain maps. Our method allows us to exploit prior information about the deployment area such as

*This work was partly supported by the PERASPERA activity of the European Commission.
[1]Robotic Systems Lab, ETH Zürich, 8092 Zürich, Switzerland {tamiki, mahutter}@ethz.ch [2]Planetary Robotics Lab, European Space Agency, Keplerlaan 1, 2201 AZ Noordwijk, Netherlands {levin.gerdes, martin.azkarate}@esa.int [3]Cognitive Robotics, Delft University of Technology, Mekelweg 2, 2628 CD Delft, Netherlands M.W.Stolzle@tudelft.nl

NeurIPS 2021 Workshop on Robot Learning: Self-Supervised and Lifelong Learning, Virtual, Virtual

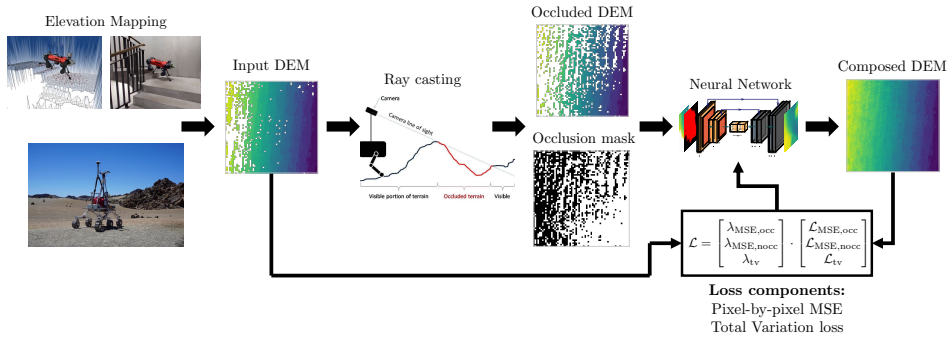

Figure 1: Partially occluded DEM are sampled via robot-centric elevation mapping. Artificial occlusion is added by performing ray casting from a random vantage point which allows a neural network to be trained to reconstruct the DEM without occlusion. The composed DEM consists of patching the original occluded DEM with the reconstruction in the occluded area. The ray casting graphic is adapted from Kolter et al. [3].

known terrain characteristics as well as the geometric constraints of line of sight for the reconstruction of missing elevation information using a neural network. To the best of our knowledge, this paper is the first to propose the use of neural networks for inpainting of robot-generated DEMs.

Self-supervised learning is essential to enable training on real-world datasets for which complete ground-truth information is challenging to acquire. Zhan et al. [8] propose a self-supervised approach for de-occlusion in the framework of scene understanding by overlaying and image with an object and then training the neural network to remove the object from the scene. Dai et al. [9] uses a self-supervised learning approach for scene completion of real-world, incomplete RGB-D scans using a Generative Adversarial Network [10] (GAN) by randomly removing some scans.

We propose a new strategy for self-supervised learning for the inpainting of occlusion in terrain maps based on adding artificial occlusion with ray casting. As non-occluded ground-truth data is very difficult to acquire, this enables us to use incomplete real-world elevation maps for training our neural network. We would like to point out that this self-supervised learning approach breaks ground towards a future where robots can actively improve their performance while they explore their environment in the line of the connected research on lifelong learning [11].

We would like to emphasize the following contributions of this paper: 1) We adapt state-of-the art neural network-based inpainting methods for filling-in missing elevation information in occluded DEMs and show significant improvements in both qualitative and quantitative results on a variety of real-world datasets compared with the traditional baseline methods. 2) We propose a novel self-supervised learning method for filling-in occlusion in terrain maps by adding artificial occlusion with ray casting enabling training on partially occluded real-world datasets. We propose an iterative algorithm for realistic occluded map data generation.

2 Methodology

First, we give a brief overview of our method as visualized in Fig. 1: we consider partially occluded 2.5D elevation maps and strive to fill-in the missing elevation information to match the less-occluded, ground-truth DEM as accurately as possible. We propose to use an U-Net [12]-like neural network to inpaint the occluded DEM with the inputs consisting of the 2.5 DEM and a binary occlusion mask. Supervised learning requires us to know ground-truth data to compute a training loss and after back-propagation optimize the neural network weights. As complete and accurate ground-truth information is rarely available for real-world datasets, we introduce a self-supervised method which leverages ray casting to further occlude the input DEM. This allows us to compute a Mean Squared Error (MSE) and Total Variation (TV) loss between the occluded DEM and the less occluded input DEM.

2.1 Problem statement

We consider a 2.5D DEM $\mathbf{m}_{\text{occ}} \in \mathcal{R}^{n \times m}$ of a terrain patch with missing values primarily caused by occlusion. We create a binary occlusion mask $\mathbf{M}_{\text{occ}} \in [0; 1]^{n \times m}$ which assigns a value of 1 to grid

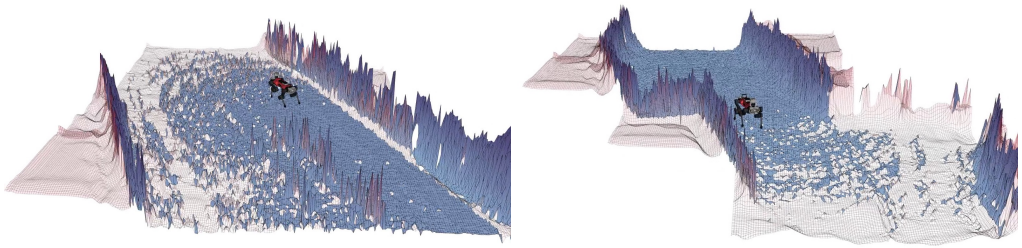

Figure 2: Snapshots of inference using the trained neural network on the Gonzen mine dataset recorded with the ANYmal legged robot. The occluded DEM is marked in color, while the reconstruction is visualized in grey with a slight opacity. To improve the clarity of presentation, we only inpaint subgrids which are less than $85\,\%$ occluded as the neural network requires sufficient input elevation information.

cells with missing elevation information and 0 to grid cells with known elevations. We define our problem as such, that we want to estimate a reconstructed DEM $\mathbf{m}_{\text{rec}}$ which matches the ground-truth DEM $\mathbf{m}_{\text{gt}}$ as closely as possible.

2.2 Self-supervised learning

We propose a self-supervised learning methodology to train on real-world datasets without ground-truth to learn to fill-in missing elevation information of DEMs. We reformulate the training setup by using our partly occluded real-world elevation map as a target and creating additional artificial occlusion which is used as an input into the neural network. We decided to add artificial occlusion by ray casting from a randomly sampled vantage point. We employ an iterative algorithm to generate useful artificial occlusion masks with occlusion ratios between $0.1\,\%$ and $50\,\%$ and choose the elevation offset of the vantage point accordingly.

2.3 Model, training and loss

We adopt an U-Net [12] as the architecture of our neural network which is often used in the literature for image and video inpainting [4, 6]. The input is composed of two channels: the occluded elevation map $\mathbf{m}_{\text{occ}}$ and the binary occlusion mask $\mathbf{M}_{\text{occ}}$. As the occluded DEM contains missing elevation values represented as NaNs computationally, we replace them similarly to [6] with the floating point number 0.0 chosen after a selection study. The output $\mathbf{m}_{\text{comp}}$ is composed between the original input $\mathbf{m}_{\text{occ}}$ for the non-occluded grid cells and the reconstruction $\mathbf{m}_{\text{rec}}$ of the neural network for the occluded pixels. We also implement input and output normalization by subtracting and adding back the mean of DEM respectively. After every training epoch on the training set, a pixel-by-pixel MSE loss is evaluated for the occluded area as our validation loss. The training is stopped if either a maximum number of epochs is reached (100) or the validation loss did not improve during a specified number of epochs (50). We use the Adam [13] optimizer with a learning rate of 0.0001, a weight decay of 0.001, and the beta coefficients $(0.9, 0.999)$. For training, we enforce a MSE loss between the ground-truth DEM $\mathbf{m}_{\text{gt}}$ and the reconstructed DEM $\mathbf{m}_{\text{rec}}$ as our pixel-by-pixel reconstruction loss. We separate the MSE loss for the occluded region of the DEM MSE_{occ} and the non-occluded region of the DEM MSE_{nocc}. A TV loss [14] has shown to be valuable as a smoothing penalty on the reconstruction of the occluded area. We compute the final loss as a weighted sum of all loss components with weights of 1 and 10 for the MSE loss of the non-occluded and occluded area respectively and finally we scale the TV loss with 0.1. More implementation details can be found in the Appendix.

3 Experiments and Results

3.1 Datasets

We evaluate our proposed self-supervised learning approach on several real-world datasets. As complete ground-truth DEM are hard to acquire in the real world, we run inference on the test set with artificial occlusion created using ray casting. In the following section, we go into more details about each of the datasets. **ANYmal datasets:** We consider multiple datasets recorded using the ANYmal [15] legged robot in three different terrains. The ANYmal robot is equipped with a

Table 1: Results for real-world datasets evaluated using artificial occlusion generated with ray casting on the test set averaged over five random seeds. The chosen unit of elevation is meters.

Method	**Terrain**	$\ell_{1,\text{occ}}$	MSE_{occ}	$PSNR_{occ}$
Telea [2]	ETH Stairs	0.09581	0.04980	26.45
Navier-Stokes [23]		0.09563	0.05339	26.14
Our method		$\mathbf{0.071 \pm 0.004}$	$\mathbf{0.018 \pm 0.001}$	$\mathbf{30.9 \pm 0.3}$
Telea [2]	Obstacle course	0.28733	0.35048	23.72
Navier-Stokes [23]		0.28005	0.34751	23.76
Our method		$\mathbf{0.173 \pm 0.002}$	$\mathbf{0.166 \pm 0.003}$	$\mathbf{26.97 \pm 0.07}$
Telea [2]	Gonzen mine	0.20346	0.16535	14.29
Navier-Stokes [23]		0.20203	0.16787	14.23
Our method		$\mathbf{0.081 \pm 0.002}$	$\mathbf{0.026 \pm 0.001}$	$\mathbf{22.3 \pm 0.2}$
Telea [2]	Tenerife Lunar	0.08978	0.02714	40.58
Navier-Stokes [23]		0.08812	0.02705	40.60
Our method		$\mathbf{0.0502 \pm 0.0009}$	$\mathbf{0.0110 \pm 0.0002}$	$\mathbf{44.51 \pm 0.09}$

dome LiDAR sensor and an IMU to perceive 3D point clouds of its environment and odometry information respectively. 2.5D DEMs of size $300 \times 300\,\text{px}$ are derived using robot-centric elevation mapping [16, 17]. The datasets encapsulate a staircase and an obstacle course at ETH Zürich, and the subterranean exploration of the Gonzen [18] mine in Switzerland. For the ANYmal datasets, we use approximately $80\,\%$ of the subgrid DEMs for the training set and approximately $10\,\%$ each respectively for the validation and test sets. The ETH stairs, ETH obstacles course, and Gonzen mine datasets contain in total 26,233, 37,274 and 16,459 samples. We divide the $300 \times 300\,\text{px}$ DEMs into 16 subgrids that is $75 \times 75\,\text{px}$ each, to increase generalization capabilities, reduce the GPU memory requirements and exclude empty parts. We uploaded a video of the inference of our method on the Gonzen mine dataset [18] to YouTube[2]. **Tenerife Lunar Analogue dataset:** We evaluate our methods on a lunar analogue dataset which was collected during a field test campaign in June 2017 at Minas de San José on the Teide Volcano on the island of Tenerife using the HDPR [19], a lab rover testbed with resemblance to the Rosalind Franklin rover [20] used in the ExoMars mission. The dataset contains GNSS recordings, the rovers raw inertial data, and images from three stereo cameras during several traverses over the duration of several days and under different lighting conditions on different pre-planned paths. We apply the GA SLAM [21] technique to extract DEMs from the raw dataset. We divide the $600 \times 600\,\text{px}$ DEMs into 64 subgrids à $75 \times 75\,\text{px}$. We exclude subgrids if they are more than $50\,\%$ occluded. This policy results in 42,600 samples for the training set, 1,000 samples for the validation set and 7,950 samples for the test set.

3.2 Results

Similar to other publications on the topic of image inpainting such as [4, 22], we state the ℓ_1 loss, the MSE, and the Peak signal-to-noise ratio (PSNR). The PSNR is a function of the total MSE loss of the occluded area and the maximum dynamic range of all DEMs. We only report the evaluation metrics for the occluded area as we argue that a composed DEM incorporating the non-occluded parts of the input DEM can be easily created. Evaluating quantitatively on real-world datasets containing artificial occlusion generated with ray casting, we state a decrease of $37\,\%$, $38\,\%$, $40\,\%$, and $46\,\%$ in ℓ_1 error compared to the Telea [2] baseline approach using self-supervised learning as listed in Table 1. Similarily, hhe MSE and derived PSNR metrics also show consistent and significant improvements using our method instead of the baseline methods.

4 Conclusion

This work proposes a self-supervised learning approach for filling occlusion in 2.5D terrain maps. The method leverages artificial occlusion generated with an iterative ray casting algorithm to train a neural network on real-world data with an incomplete ground-truth. Evaluating quantitatively on real-world datasets, we state a decrease of between $37\,\%$ and $46\,\%$ in ℓ_1 error compared to the Telea [2] baseline approach. It is crucial to explore uncertainty estimation methods such as dropout for model uncertainty in future work to enable the passing of elevation variance estimates to downstream navigation tasks such as motion planning.

[2] `https://youtu.be/2Khxeto62LQ`

References

[1] M. Ebrahimi and E. Lunasin, "The navier–stokes–voight model for image inpainting," *The IMA Journal of Applied Mathematics*, vol. 78, no. 5, pp. 869–894, 2013.

[2] A. Telea, "An image inpainting technique based on the fast marching method," *Journal of graphics tools*, vol. 9, no. 1, pp. 23–34, 2004.

[3] J. Z. Kolter, Y. Kim, and A. Y. Ng, "Stereo vision and terrain modeling for quadruped robots," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 1557–1564.

[4] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 85–100.

[5] U. Demir and G. Unal, "Patch-based image inpainting with generative adversarial networks," *arXiv preprint arXiv:1803.07422*, 2018.

[6] N. Wang, S. Ma, J. Li, Y. Zhang, and L. Zhang, "Multistage attention network for image inpainting," *Pattern Recognition*, vol. 106, p. 107448, 2020.

[7] O. Elharrouss, N. Almaadeed, S. Al-Maadeed, and Y. Akbari, "Image inpainting: A review," *Neural Processing Letters*, vol. 51, no. 2, pp. 2007–2028, 2020.

[8] X. Zhan, X. Pan, B. Dai, Z. Liu, D. Lin, and C. C. Loy, "Self-supervised scene de-occlusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[9] A. Dai, C. Diller, and M. Nießner, "Sg-nn: Sparse generative neural networks for self-supervised scene completion of rgb-d scans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 849–858.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[11] S. Thrun and T. M. Mitchell, "Lifelong robot learning," *Robotics and autonomous systems*, vol. 15, no. 1-2, pp. 25–46, 1995.

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

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

[14] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*. Springer, 2016, pp. 694–711.

[15] M. Hutter, C. Gehring, A. Lauber, F. Gunther, C. D. Bellicoso, V. Tsounis, P. Fankhauser, R. Diethelm, S. Bachmann, M. Blösch, *et al.*, "Anymal-toward legged robots for harsh environments," *Advanced Robotics*, vol. 31, no. 17, pp. 918–931, 2017.

[16] P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, and R. Siegwart, "Robot-centric elevation mapping with uncertainty estimates," in *International Conference on Climbing and Walking Robots (CLAWAR)*, 2014.

[17] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 4, pp. 3019–3026, 2018.

[18] T. Dang, M. Tranzatto, S. Khattak, F. Mascarich, K. Alexis, and M. Hutter, "Graph-based subterranean exploration path planning using aerial and legged robots," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1363–1388, 2020.

[19] E. Boukas, R. A. Hewitt, M. Pagnamenta, R. Nelen, M. Azkarate, J. A. Marshall, A. Gasteratos, and G. Visentin, "Hdpr: A mobile testbed testbed for current and future rover technologies," in *Artificial Intelligence, Robotics, and Automation in Space (iSAIRAS 2016), 13th International Symposium on*, 2016, pp. 1–6.

[20] J. L. Vago, F. Westall, A. J. Coates, R. Jaumann, O. Korablev, V. Ciarletti, I. Mitrofanov, J.-L. Josset, M. C. De Sanctis, J.-P. Bibring, *et al.*, "Habitability on early mars and the search for biosignatures with the exomars rover," *Astrobiology*, vol. 17, no. 6-7, pp. 471–510, 2017.

[21] D. Geromichalos, M. Azkarate, E. Tsardoulias, L. Gerdes, L. Petrou, and C. Perez Del Pulgar, "Slam for autonomous planetary rovers with global localization," *Journal of Field Robotics*, vol. 37, no. 5, pp. 830–847, 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21943

[22] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5505–5514.

[23] M. Bertalmio, A. L. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1. IEEE, 2001, pp. I–I.

[24] K. Yi, Y. Guo, Y. Fan, J. Hamann, and Y. G. Wang, "Cosmo vae: Variational autoencoder for cmb image inpainting," *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, 2020.

[25] H. Zhang, Z. Hu, C. Luo, W. Zuo, and M. Wang, "Semantic image inpainting with progressive generative networks," in *Proceedings of the 26th ACM international conference on Multimedia*, 2018, pp. 1939–1947.

[26] J. Li, F. He, L. Zhang, B. Du, and D. Tao, "Progressive reconstruction of visual structure for image inpainting," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[27] Y.-L. Chang, Z. Y. Liu, K.-Y. Lee, and W. Hsu, "Free-form video inpainting with 3d gated convolution and temporal patchgan," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

APPENDIX

4.1 Ray casting

We rely on ray casting to generate artificial occlusion in the framework of self-supervised learning. We developed a lightweight C++ component with Python bindings to perform fast ray casting of an entire grid map from a given vantage point[3]. The ray casting algorithm (see Algorithm 1) takes a grid map (e.g. DEM) and a vantage point $\mathbf{x}_v$ in Cartesian coordinates as inputs. The vantage point is specified relative to the center of the DEM (e.g. image coordinates $(\frac{u}{2}, \frac{v}{2})$). It iterates with a nested *for* loop through every cell in the grid and subsequently checks whether the cell is visible from the vantage point. This is done by tracing a ray in 3D from the vantage point into direction $\mathbf{d}_{ray}$ of the cell $\mathbf{x}_{gc}$. We initialise the grid cell as not occluded. We step along the ray with a step length of half of the grid resolution $\mathbf{r}_g \in \mathbb{R}^2$. After each step, we evaluate the current corresponding pixel and extract the elevation of the pixel from the DEM. If we have stored an elevation information for the pixel in the DEM, we check if the elevation is higher than the current vertical offset of the ray. If that is the case, we break the loop and designate the grid cell, which we are ray casting, as occluded. Otherwise, we continue stepping along the ray until we reach the target grid cell where we break and designate the grid cell as not occluded. We also break when we are past the grid cell by checking if the distance of the vantage point to the current ray position is larger or equal to the distance from the vantage point to the grid cell as we sometimes do not directly reach the grid cell because of numeric inaccuracies.

4.2 Self-supervised learning

We propose a self-supervised learning methodology to train on real-world datasets without ground-truth to learn to fill-in missing elevation information of DEMs. We reformulate the training setup by using our partly occluded real-world elevation map as a target and creating additional artificial occlusion which is used as an input into the neural network. We considered a dilation of the already occluded area in addition to randomly occluding pixels. However, this approach does not render realistic and diverse occlusion masks. We decided to add artificial occlusion by ray casting from a randomly sampled vantage point.

We employ an iterative algorithm to generate useful artificial occlusion masks with occlusion ratios between $0.1\,\%$ and $50\,\%$ and choose the elevation offset o of the vantage point accordingly. First, we sample a random vantage point from the grid with a uniform distribution. Then, we sample a random elevation offset for the vantage point $o \sim \mathcal{U}\,(o_{\text{min}}, o_{\text{max}})$, perform ray casting for the chosen vantage point and evaluate the resulting occlusion ratio (e.g., number of occluded grid cells over number of total grid cells).

We require that the occlusion ratio r_{occ} lies within the interval $[r_{\text{occ,min}}, r_{\text{occ,max}}]$. If that is not the case, we sample a new elevation offset. This time, we adjust the range of the uniform distribution: if $r_{\text{occ}} > r_{\text{occ,max}}$, we increase o_{min} to the previously sampled elevation offset o, and if $r_{\text{occ}} < r_{\text{occ,min}}$, we decrease o_{max} to the previously sampled elevation offset o. If $\|o_{\text{max}} - o_{\text{min}}\| < 0.05\,\text{m}$, we enforce a minimal sampling range by respectively decreasing o_{min} by $0.05\,\text{m}$ or increasing o_{max} by $0.05\,\text{m}$. We repeat this algorithm until either we have satisfied our occlusion ratio constraint $[r_{\text{occ,min}}, r_{\text{occ,max}}]$ or we have reached the maximum number of iterations (15). We reformulate our loss function to only consider the areas of artificial occlusion and ignore the already occluded areas in the target DEM.

4.3 Model architecture

We adopt an U-Net [12] for our neural network architecture as used by many publications in the past few years on the topic of image and video inpainting [4, 6, 24–27]. The input is composed of two channels: the occluded elevation map $\mathbf{m}_{\text{occ}}$ and the binary occlusion mask $\mathbf{M}_{\text{occ}}$. As the occluded elevation map contains missing values represented as NaNs for the grid cells which are occluded, we need to replace those NaNs with a floating point number. We treat this replacement value as a hyperparameter and select 0.0 after a selection study. We also implement input and output normalization: we compute the mean of the non-occluded elevation values of a DEM and subtract

[3] `https://github.com/mstoelzle/grid-map-raycasting`

Algorithm 1: Perform ray casting from vantage point $\mathbf{x}_{vp}$ and with grid resolution $\mathbf{r}_g$ for DEM $\mathbf{m}_{\text{gt}}$ and return occluded DEM $\mathbf{m}_{\text{occ}}$

Result: $\mathbf{m}_{\text{occ}}$
$u \leftarrow 0$;
$v \leftarrow 0$;
while $u < n$ **do**
  $\mathbf{x}_{gc,x} \leftarrow (-\frac{n}{2} + u) * \mathbf{r}_{g,x}$;
  while $v < m$ **do**
    $\mathbf{x}_{gc,y} \leftarrow (-\frac{m}{2} + v) * \mathbf{r}_{g,y}$;
    $\mathbf{x}_{gc,z} \leftarrow \mathbf{m}_{\text{gt}}$;
    if *isNan*$(\mathbf{m}_{gt})$ **then**
      $\mathbf{m}_{\text{occ,uv}} = True$;
      continue
    end
    $\mathbf{d}_{\text{ray}} \leftarrow \frac{\mathbf{x}_{gc} - \mathbf{x}_{vp}}{\|\mathbf{x}_{gc} - \mathbf{x}_{vp}\|_2}$;
    occ $\leftarrow$ False;
    $\mathbf{x}_{rc} \leftarrow \mathbf{x}_{vp}$;
    while *occ* == *False* **do**
      $\mathbf{x}_{rc} \leftarrow \mathbf{x}_{rc} + 0.5 \cdot \mathbf{d}_{ray} \cdot \min(\mathbf{r}_{g,x}, \mathbf{r}_{g,y})$;
      $u_{rc} \leftarrow round(\frac{n}{2} + \frac{\mathbf{x}_{rc,x}}{\mathbf{r}_{g,x}})$;
      $v_{rc} \leftarrow round(\frac{m}{2} + \frac{\mathbf{x}_{rc,y}}{\mathbf{r}_{g,y}})$;
      if $u == u_{rc}$ *and* $v == v_{rc}$ **then**
        break
      end
      if $\|\mathbf{x}_{rc} - \mathbf{x}_{vp}\|_2 > \|\mathbf{x}_{gc} - \mathbf{x}_{vp}\|_2$ **then**
        break
      end
      if *isNotNan(*$\mathbf{m}_{gt,rc}$*)* **then**
        if $\mathbf{m}_{\text{gt},rc} > \mathbf{x}_{rc,z}$ **then**
          occ $\leftarrow$ True;
        end
      end
    end
    $\mathbf{m}_{\text{occ,uv}} \leftarrow$ occ;
    $v \leftarrow v + 1$;
  end
  $u \leftarrow u + 1$;
end

this mean from each elevation value in the grid. After receiving the output of the model, we add this mean back to each elevation value before computing the loss. We adapt our model from the vanilla U-Net [12] architecture and make slight adjustments to the number of max-pooling steps because we are dealing with smaller input images (64×64 px instead of 572×572 px) than in the original paper [12] and we want to keep the network as lightweight as possible. Thus, we limit the number of max-pooling operations to 3 (instead of 5 in the original paper) and treat the number of channels in each hidden dimension as a hyperparameter for which we select 64, 128, and 256 channels for our hidden dimensions. Analogue to the original paper [12], we use double convolutions with kernel 3 and padding 1 at every hidden dimension for the contracting path. Every convolution is followed by 2D batch norm and a Rectified Linear Unit (ReLU). After the encoder, we directly enter the expansive path without a latent space. Bilinear up-sampling with a scale factor of two is used for every decoding step with subsequent concatenation with the skip connection data. Analogue to the encoder, every double convolution is followed by a 2D batch norm and a ReLU. Ablation study experiments showed that skip connections are absolutely essential for decent reconstruction performance. We visualize the adapted U-Net architecture in Figure 3.

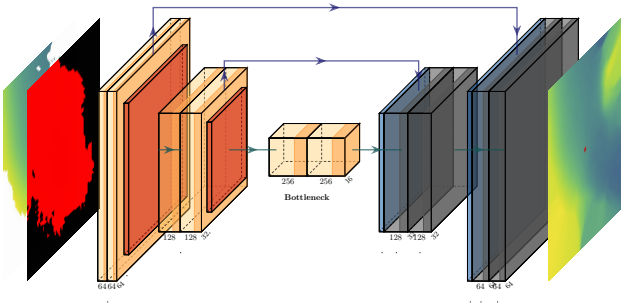

Figure 3: U-Net [12] as implemented in this work. We input the occluded DEM $\mathbf{m}_{\text{occ}}$ and a binary occlusion mask $\mathbf{M}_{\text{occ}}$ and output the reconstructed DEM. $\mathbf{m}_{\text{rec}}$

4.4 Evaluation metrics

Similar to other publications on the topic of image inpainting such as [4, 22], we state the ℓ_1 loss, the MSE, and the PSNR. We use the separated test set for all evaluation results.

$$\ell_{1,\text{occ}} = \frac{1}{N_{\text{occ}}} \sum_{(i,j)\in R_{\text{occ}}} \|\mathbf{m}_{\text{gt},ij} - \mathbf{m}_{\text{rec},ij}\|_1 \tag{1}$$

The PSNR is a function of the total MSE loss of the occluded area and the maximum dynamic range of the pixel-values L determined by computing the delta between the highest and lowest elevation of the ground-truth DEMs $\mathbf{m}_{\text{gt}}^k$ of the entire test dataset.

$$\begin{aligned} \text{PSNR}_{\text{occ}} &= 20 \cdot \log_{10}(L) - 10 \cdot \log_{10}(\mathcal{L}_{\text{MSE,occ}}) \\ L &= \max(\mathbf{m}_{\text{gt}}^k) - \min(\mathbf{m}_{\text{gt}}^k) \end{aligned} \tag{2}$$

It is important to mention that the PSNR is not linearly accumulated over mini-batches, but rather computed as a function of the MSE loss over the entire dataset. We only report the evaluation metrics for the occluded area as we argue that a composed DEM incorporating the non-occluded parts of the input DEM $\mathbf{m}_{\text{comp}}$ can be easily created and thus the reconstruction capability of the non-occluded area by the model is not essential.