# Void Filling of Digital Elevation Models with Deep Generative Models

Konstantinos Gavriil, Georg Muntingh and Oliver J. D. Barrowclough

***Abstract*—In recent years, advances in machine learning algorithms, cheap computational resources, and the availability of big data have spurred the deep learning revolution in various application domains. In particular, supervised learning techniques in image analysis have led to superhuman performance in various tasks, such as classification, localization, and segmentation, while unsupervised learning techniques based on increasingly advanced generative models have been applied to generate high-resolution synthetic images indistinguishable from real images.**

**In this paper we consider a state-of-the-art machine learning model for image inpainting, namely a Wasserstein Generative Adversarial Network based on a fully convolutional architecture with a contextual attention mechanism. We show that this model can successfully be transferred to the setting of digital elevation models (DEMs) for the purpose of generating semantically plausible data for filling voids. Training, testing and experimentation is done on GeoTIFF data from various regions in Norway, made openly available by the Norwegian Mapping Authority.**

***Index Terms*—Digital elevation models; unsupervised learning; predictive models; remote sensing**

## I. Introduction

IN THE field of remote sensing and data capture, a common issue is that certain areas are not completely or adequately covered, resulting in regions of 'missing data'. The reasons behind this issue vary depending in the data capture technique applied. For example, NASA's Shuttle Radar Topography Mission (SRTM) from the early 2000s attempted to provide a complete digital elevation model (DEM) of most of the globe. However, issues with missing data arose in regions of high gradient, such as mountainous regions, meaning very rugged terrain was often not well captured [1]. In stereophotogrammetry, where pairs of aerial or satellite images are matched to create digital elevation models, failures can occur when there are differences between the content of two images (e.g. variable cloud cover at different capture times) or in regions where there are not enough features to perform a successful matching. In light detection and ranging (LIDAR) capture, the sensors are typically positioned together with the source of illumination. This means that data is only captured on the 'visible' surface and no data is captured on the 'back side' of objects unless the sensor is moved.

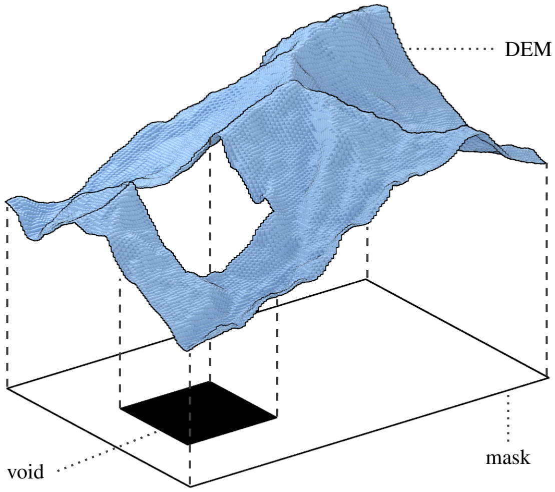

Fig. 1. Visual representation of a digital elevation model, and the respective binary mask of the known and unknown values.

Traditional methods to rectify the issue of missing or conflicting data include interpolation using spline surfaces [2], kriging [3], inverse distance weighting (IDW) [4] and triangular irregular networks (TINs) [1]. These methods perform differently with respect to the type of terrain they are used to fill; smooth, sharp or containing irregular patterns.

In this letter we apply transfer learning techniques to train a model to be able to recover general features that are found in digital elevation/surface models. In this way we avoid the need to apply different methods to different terrain types. The need for human input is also limited to post-processing.

Our results are obtained by transferring to DEMs the recent successes of generative modeling techniques in the research field of *image inpainting*, meaning the problem of filling missing regions of an image with data that appear plausible, in the sense of human interpretation. In the context of DEMs, plausibility of the filled data is not necessarily sufficient. In many cases, one would wish to accurately reproduce the missing part of the elevation model. However, in many cases this is an unrealistic goal due to various limitations. We therefore restrict our attention in this letter to providing a satisfactory fill for the various regions considered.

## II. Background in Generative Models

*Generative models* form a branch of unsupervised machine learning techniques that estimate the (joint) probability distri-

K. Gavriil is with Evolute GmbH and the Institute of Discrete Mathematics and Geometry, Vienna University of Technology, Wiedner Hauptstrasse 8-10/104, A-1040 Vienna, Austria (email: gavriil@evolute.at).

G. Muntingh and O.J.D. Barrowclough are with SINTEF Digital, Forskningsveien 1, 0373 Oslo, Norway (email: georg.muntingh@sintef.no and oliver.barrowclough@sintef.no)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 675789. This projected was also supported by an IKTPLUSS grant, project number 270922, from the Research Council of Norway.

Manuscript received November 30, 2018; revised February 4, 2019; accepted February 22, 2019.

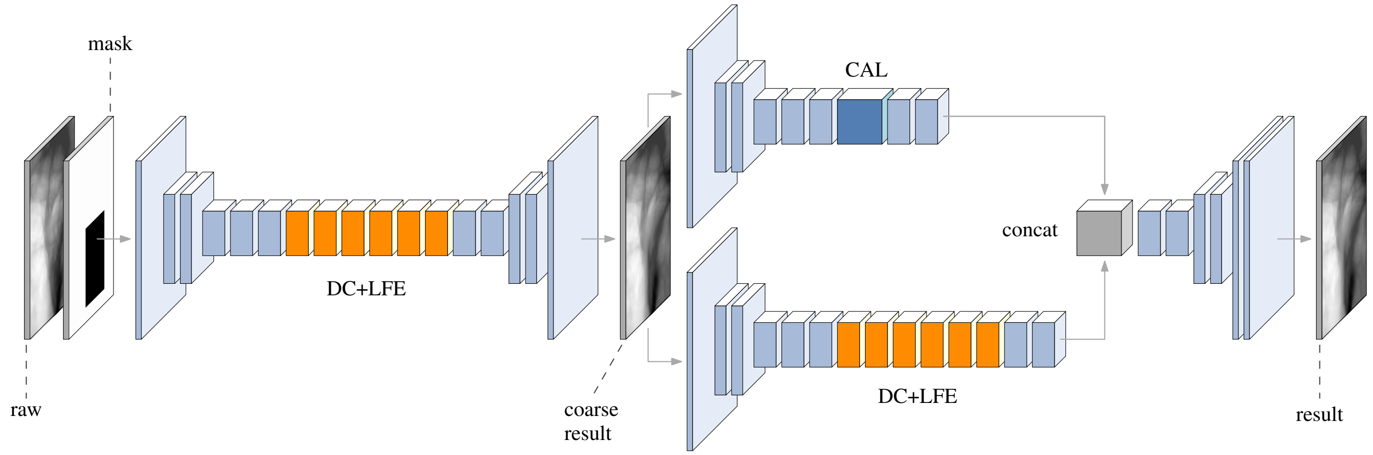

Fig. 2. The generative void filling model utilized for missing DEM value completion. We highlight the dilated convolutions (DC), local feature extraction (LFE) modules, and the contextual attention layer (CAL).

bution underlying given data. For complex high-dimensional probability distributions, such as for DEM data generation, it is not feasible to learn this distribution explicitly. It is however possible to obtain an implicit description through a model capable of generating samples from an estimated distribution.

*Generative adversarial networks* (GANs) [5] form a highly promising framework for training a model that generates such samples. One reason for this is that the adversarial loss in GANs is known to catch highly recognizable salient features not picked up by mean squared error (MSE) [6]. At the core of a GANs are two adversaries attempting to outwit one another: A *generator* learns to generate fake samples that are supposed to look real, and a *discriminator* learns to distinguish real data from fake samples. When these adversaries are carefully balanced, both become better over training time.

Initially GANs were difficult to train due to this balancing act. This situation is remedied to some extent by the recently proposed *Wasserstein GAN* (WGAN) [7], which capitalizes on better theoretical properties of the Wasserstein-1 distance — also known as Earth Mover's (EM) distance, as it measures the effort necessary for moving the estimated probability density to the true density.

Although GANs at their inception showed great results for small images, this success was initially difficult to scale up to high-resolution data. Constraining the network architecture to only convolutional layers, the *Deep Convolutional GAN* (DCGAN) [8] brought simplicity, deeper models, flexibility in image resolution, and insight into what GANs learn. Adding dilated convolutions [9] takes this one step further, bringing back an enhanced receptive field previously handled by fully connected layers.

These techniques form the foundation for a recent wave of deep generative inpainting networks [10]–[12]. Further improvements to training stability and speed are obtained by adding to the loss function a spatially discounted reconstruction loss, as well as local and global critics to ensure local consistency and global coherence, and a contextual attention mechanism to capture relevant features at a distance [13].

## III. Methodology

### A. Problem formulation and notation

We will consider preprocessed digital elevation/surface models in GeoTIFF format, in which the data forms a grid with a single height value for every position $(i, j)$. Let $\mathbf{D} = (d_p) \in \mathbb{R}^{m\times n}$ be a partial digital elevation model, where $p$ is an abbreviation for pixel referring to the coordinates $(i, j)$ of a point on the DEM grid and $d_p$ is the corresponding height value. Partial means that some pixel values are considered void. A binary matrix $\mathbf{M} = (m_p) \in \{0, 1\}^{m\times n}$ acts as a mask representing the void regions of $\mathbf{D}$. We refer to pixels $p$ for which $m_p = 0$ as *known*, and *unknown* otherwise.

**Problem 1.** *Given an initial partial elevation model* $\mathbf{D}^0$ *and the corresponding mask* $\mathbf{M}$*, construct a complete elevation model* $\mathbf{D}$ *with semantically plausible generated values for the masked regions.*

**Algorithm 1** DEM Void Filling

```
Input: partial DEM $\mathbf{D}^0 = (d^0_p)$, mask $\mathbf{M} = (m_p)$
       blending weight function $\beta$, blending width $w$
       paraboloid fitting window radius $r$
Output: predicted DEM $\mathbf{D}$
$\mathbf{D} \leftarrow G(\mathbf{D}^0, \mathbf{M})$    ▷ initial result
partition unknown pixels to sets $R_k$ of pixels
  with $L_1$-distance of $k$ from a known pixel
for $k \leftarrow 1, \dots, w$ do
    for $p = (i, j) \in R_k$ do
        compute the set $X$ of known pixels of $\mathbf{D}^0$
          in subgrid $[i - r, i + r] \times [j - r, j + r]$
        perform least squares paraboloid fitting to X
          $f^* \leftarrow \operatorname*{argmin}_f \sum_{q\in X} [f(q) - d_q]^2$
        $d^0_p \leftarrow f^*(p)$    ▷ approximate $C^2$ extension of $\mathbf{D}^0$
        $\alpha \leftarrow \beta(\frac{k-1}{w})$    ▷ blending weight
        $d_p \leftarrow (1 - \alpha)\, d^0_p + \alpha d_p$    ▷ blend
    end for
    $m_{R_k} \leftarrow 0$    ▷ label pixels in $R_k$ as known
end for
```

### B. Main algorithm

Our method solves Problem 1 in two steps. Initially, we get a complete elevation model $\mathbf{D}$ by employing a deep generative network $G$, which has been trained to complete missing data values while respecting relevant features of the surrounding regions. The second step involves optional post-processing of the result of Step 1 for obtaining a smooth transition between the initial known region and the prediction provided by $G$.

Algorithm 1 is a complete description of the proposed solution which we will analyze in detail in the sequel.

### C. Model Architecture

The proposed DEM void filling generative model $G$ is an adaptation of the generative image inpainting model presented in [13]; see Figure 2. This model demonstrates promising results for texture-like images, which we leverage to transferring topographic patterns in our setting.

The input consists of the two arrays $\mathbf{D}^0$ and $\mathbf{M}$. We focus on size $256 \times 256$ arrays for our implementation. The training set is generated by artificially removing randomly generated rectangular regions from the ground truth provided by complete GeoTIFFs from the Norwegian Mapping Authority. This data source was chosen for two reasons. First of all it is openly available, facilitating reproducible research. Secondly, we hypothesize that the variety in Norwegian topography makes it well-suited for generalization to other regions of the world.

Following the coarse-to-fine network approach of [13], the missing region is at first filled with a coarse prediction, which is fed to a second network for refinement, before the end result $\mathbf{D} = G(\mathbf{D}^0, \mathbf{M})$. The coarse prediction stage is a dilated deep convolutional encoder-decoder network trained with reconstruction loss, generating a smooth initial guess for the contents of the hole. The refinement stage contains two parallel encoders, one implementing the contextual attention mechanism and the other a dilated deep convolutional encoder, merged as input to a single decoder generating a prediction for the completed grid.

For improved local feature aggregation—necessary for remote sensing applications—both dilation components of the model utilize the recent *Local Feature Extraction* (LFE) module [14], which consists of convolutional layers masks of size $3 \times 3$ and with first increasing and then decreasing dilations 2-4-8-8-4-2.

The composed network is trained end-to-end with $\ell_1$ reconstruction loss, global and local Wasserstein GAN Gradient-Penalty (WGAN-GP) adversarial loss [15]. The reconstruction loss is spatially discounted, in the sense that hallucination is stronger the further away one is from known data. For further network specifics, hyperparameters, and examples, see [16].

### D. Boundary post-processing

We propose an optional post-processing step to remedy any boundary artifacts between the known edges and the generated elevation values. The intuition behind the procedure is that we compute the approximate $C^2$-continuous extension of the known region boundary and blend it appropriately with the resulting DEM. Figure 3 demonstrates an example of the post-processing step.

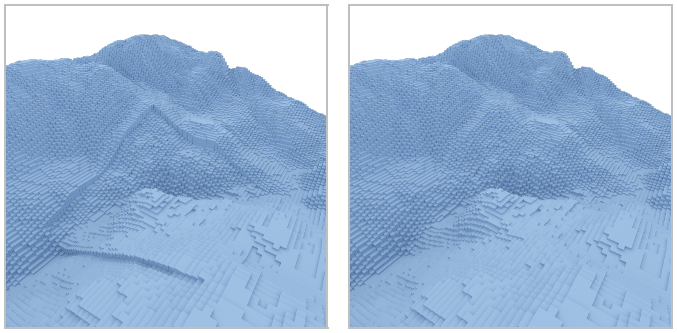

Fig. 3. Detail from a landscape DEM with exaggerated boundary discontinuity (left) and the result of the post-processing boundary blending step (right).

We partition the unknown pixels to sets $R_k$, each containing pixels with $L_1$-distance of $k$ from a known pixel. For $k = 1, \ldots, w$, where $w \in \mathbb{N}$ is the chosen width of the blending region, we update the current boundary $R_k$ with the following procedure. For each boundary pixel $p = (p_i, p_j) \in R_k$, let $X$ be the set of known pixels in $[p_i - r, p_i + r] \times [p_j - r, p_j + r]$, i.e., the known entries of the submatrix of size $(2r+1) \times (2r+1)$ centered at $p$. The value of $r$ is chosen by the user (we use 3 or 4). The best fitting paraboloid

$$f(i,j) = Ai^2 + Bij + Cj^2 + Di + Ej + F$$

in the least squares sense is given by the solution

$$f^* = \operatorname*{argmin}_f \sum_{q \in X} \left[ f(q) - d_q \right]^2$$

and approximates the local curvature of the DEM. We assign $f^*(p)$ to the value of $p$, and repeat the process for all pixels of $R_k$. The boundary $R_k$ is labeled as known and we move to the next boundary $R_{k+1}$. The entire extension procedure is repeated $w$ times to achieve an extension of width $w$.

The $C^2$ extension is then blended with the predicted result in the following manner. We choose a strictly increasing bijective blending function $\beta : [0,1] \to [0,1]$. For our experiments, we use a sigmoid blending function. The final value $d_p^f$ at pixel $p \in R_k$, $k = 1, \ldots, w$ is the linear blend of the value $d_p^0$ of the $C^2$-extended initial elevation model and the value $d_p$ of the result from the generative network $G$, that is

$$d_p^f = (1 - \alpha_k) d_p^0 + \alpha_k d_p,$$

where $\alpha_k = \beta(\frac{k-1}{w})$ is the blending weight.

## IV. Experiments and Results

We trained two separate models for our experiments. Model $G_1$ was trained on rectangular 2m-resolution DEMs of the three largest cities in Norway, namely Oslo, Trondheim, and Bergen, while model $G_2$ was trained on 10m-resolution DEM of Western and Eastern Norway.

We compare our generators $G_1$, $G_2$ with two traditional methods to void filling; inverse distance weighting interpolation (IDW) and splines. For a fair comparison, no post-processing was used. The implementation of IDW is taken from GDAL [17] with the option of two $3 \times 3$ smoothing filter passes. For the spline approach we utilize locally refined (LR) B-splines [2], [18]. This letter is too short to contain a complete description of LR B-splines, but for the purpose of our application we expect them to perform at least as well as tensor-product B-splines [3].

Figure 4 contains a carefully selected collection of representative scenarios. These include large missing regions (Figures 4{a,d}), multiple voids (Figures 4{b,c,e}), and non-axis-aligned voids (Figure 4e). The strength of the spline method is to smoothly interpolate the boundary of the missing regions. The IDW method gives good results on small gaps, but shows axis-aligned and diagonal artifacts on larger grids. Our approach typically achieves the expected geometric continuation and respects surrounding features.

The generator $G$, IDW, and LR B-spline methods were also applied to randomly selected urban and rural DEMs, 50 of each. The results were compared to the ground truth in the EM distance applied to histograms of intensities, and the MSE, as summarized in Table I. Complete recovery of the ground truth is an unrealistic goal, so these results provide limited insight. However, quantitative measures, while being less subjective, may not directly correspond to how humans perceive and judge generated images [19], which is reflected in the table.

Table I. Average MSE and EM errors for the various void filling methods, with the generator $G_1$, $G_2$ trained on the urban, rural datasets respectively.

| | | $G_1$ | $G_2$ | IDW | LR B-spline |
|---|---|---|---|---|---|
| Urban | MSE | 28.76 | - | 22.10 | **17.08** |
| | EM | 10.28 | - | **8.21** | 12.99 |
| Rural | MSE | - | **809.09** | 1079.01 | 868.13 |
| | EM | - | **8.55** | 9.37 | 11.32 |

## V. Conclusion

In this letter, we adapt an existing methodology for image inpainting to filling voids in a DEM. We present results from multiple usage scenarios and showcase the advantages and the drawbacks of our approach.

We consider this work as a generic proof of concept, establishing viability of using deep generative models in the context of DEMs. As such we have limited the scope of the presented methodology to the task of filling missing regions in DEMs. However, we identify the wider applicability of our pre-trained model to other types of remote sensing data (domain adaptation) and related tasks (transfer learning), such as manipulating existing data. By making the model and other resources publicly available [16], we encourage the reader to transfer these results to related applications domains.

In the future we would like to extend this methodology to targeted applications, such as superresolution and generating prescribed structures by replacing input noise vectors by interpretable code vectors. The latter can be achieved by disentangling the individual entries of the code vector by introducing a component that minimizes mutual information [20]. Another natural next step is multi-view learning for consolidating various data sources, for instance by stacking these views as separate input layers. Multi-task learning holds a high potential for extracting more generic features that are more suitable for transfer learning to specific tasks. Finally we wish to investigate using other evaluation metrics [19] more suitable to this use case.

## Acknowledgments

We adapted a publicly available GitHub repository for our experiments [13]. The open source C++ library GoTools was also used for generating the LR B-spline data [21]. Data provided courtesy Norwegian Mapping Authorities (www.hoydedata.no), copyright Kartverket (CC BY 4.0).

## References

[1] E. Luedeling, S. Siebert, and A. Buerkert, "Filling the voids in the SRTM elevation model - a TIN-based delta surface approach," *ISPRS Journal of Photogrammetry and Remote Sensing*, 2007.

[2] V. Skytt, O. J. D. Barrowclough, and T. Dokken, "Locally refined spline surfaces for representation of terrain data," *Computers & Graphics*, vol. 49, pp. 58–68, 2015.

[3] H. I. Reuter, A. Nelson, and A. Jarvis, "An evaluation of void-filling interpolation methods for SRTM data," *International Journal of Geographical Information Science*, 2007.

[4] D. Shepard, "A two-dimensional interpolation function for irregularly-spaced data," in *Proceedings of the 1968 23rd ACM National Conference*, ser. ACM '68, 1968, pp. 517–524.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., pp. 2672–2680.

[6] W. Lotter, G. Kreiman, and D. D. Cox, "Unsupervised learning of visual structure using predictive generative networks," *CoRR*, vol. abs/1511.06380, 2015.

[7] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial network," in *International Conference on Machine Learning (ICML)*, 2017.

[8] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015.

[9] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *CoRR*, vol. abs/1511.07122, 2015.

[10] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Transactions on Graphics*, 2017.

[11] Y. Li, S. Liu, J. Yang, and M. H. Yang, "Generative face completion," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017.

[12] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros, "Context Encoders: Feature Learning by Inpainting." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[13] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[14] R. Hamaguchi, A. Fujita, K. Nemoto, T. Imaizumi, and S. Hikosaka, "Effective Use of Dilated Convolutions for Segmenting Small Object Instances in Remote Sensing Imagery," in *Proceedings - IEEE Winter Conference on Applications of Computer Vision, WACV 2018*, 2018.

[15] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," Dec. 2017, pp. 5769–5779, arxiv: 1704.00028.

[16] K. Gavriil, G. Muntingh, and O. J. D. Barrowclough, "Digital Elevation Model – Fill," https://github.com/konstantg/dem-fill, 2018.

[17] GDAL Development Team, *GDAL - Geospatial Data Abstraction Library, Version 2.2.2*, Open Source Geospatial Foundation, 2018. [Online]. Available: http://www.gdal.org

[18] T. Dokken, T. Lyche, and K. F. Pettersen, "Polynomial splines over locally refined box-partitions." *Computer Aided Geometric Design*, vol. 30, no. 3, pp. 331–356, 2013.

[19] A. Borji, "Pros and cons of GAN evaluation measures," *Computer Vision and Image Understanding*, 2018.

[20] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *NIPS*, 2016, pp. 2172–2180.

[21] SINTEF Digital, "GoTools Geometry Toolkit," https://github.com/SINTEF-Geometry/GoTools, 2018.

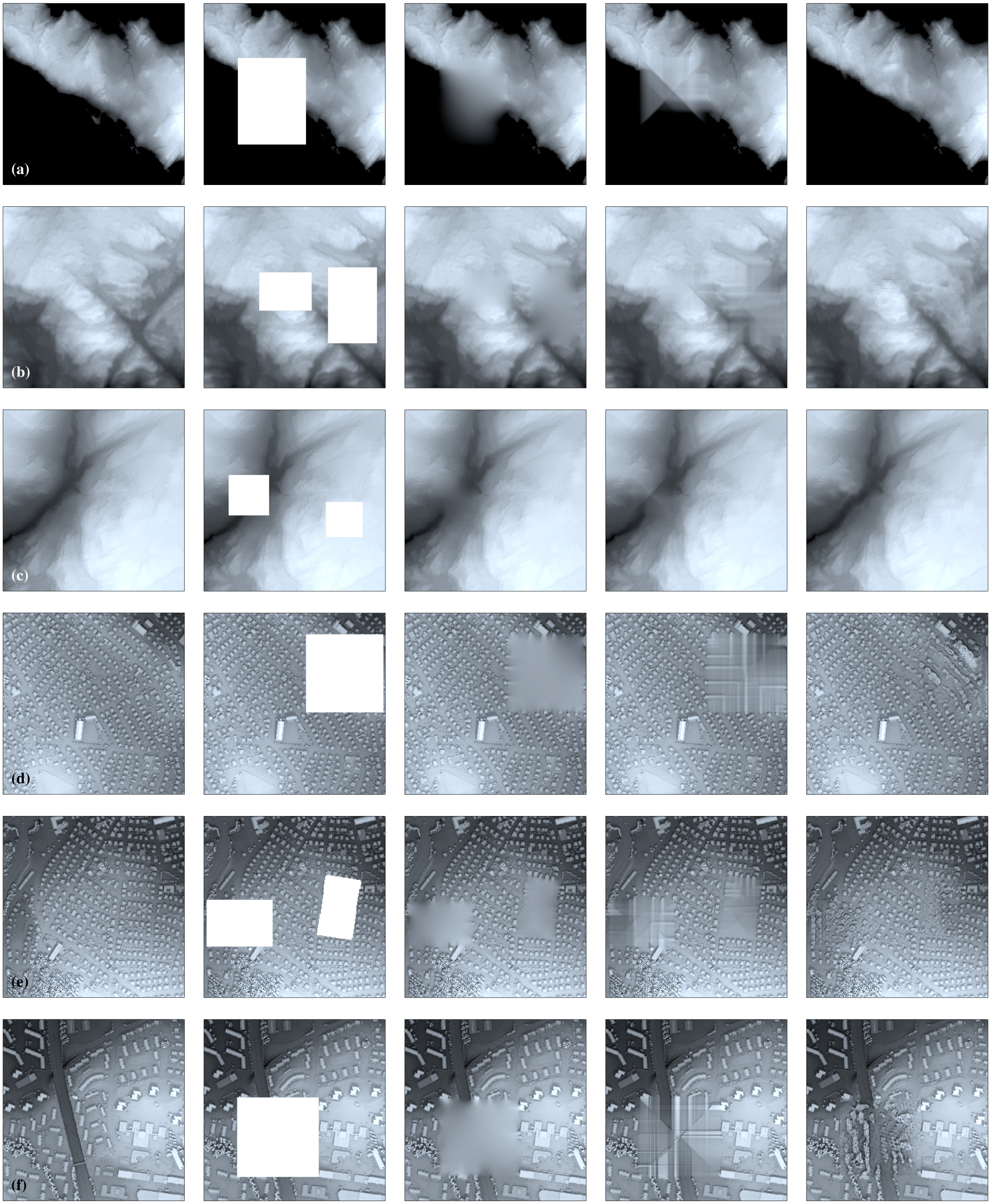

Fig. 4. A selection of results of our application to (a)–(c) rural and (d)–(f) urban data, rendered such that shadows emphasize any artifacts in the image. From left to right: original DEM, mask, LR B-spline approximation, IDW, our generator $G$. Row (f) shows a failure case, in that it fails to reconstruct the road forming the most prominent feature.