*remote sensing*

MDPI

*Article*

# Void Filling of Digital Elevation Models with a Terrain Texture Learning Model Based on Generative Adversarial Networks

**Zhonghang Qiu, Linwei Yue * and Xiuguo Liu**

School of Geography and Information Engineering, China University of Geosciences, Wuhan 430074, China; qzh@cug.edu.cn (Z.Q.); liuxg@cug.edu.cn (X.L.)

* Correspondence: yuelw@cug.edu.cn

Received: 26 October 2019; Accepted: 25 November 2019; Published: 28 November 2019


**Abstract:** Digital elevation models (DEMs) are an important information source for spatial modeling. However, data voids, which commonly exist in regions with rugged topography, result in incomplete DEM products, and thus significantly degrade DEM data quality. Interpolation methods are commonly used to fill voids of small sizes. For large-scale voids, multi-source fusion is an effective solution. Nevertheless, high-quality auxiliary source information is always difficult to retrieve in rugged mountainous areas. Thus, the void filling task is still a challenge. In this paper, we proposed a method based on a deep convolutional generative adversarial network (DCGAN) to address the problem of DEM void filling. A terrain texture generation model (TTGM) was constructed based on the DCGAN framework. Elevation, terrain slope, and relief degree composed the samples in the training set to better depict the terrain textural features of the DEM data. Moreover, the resize-convolution was utilized to replace the traditional deconvolution process to overcome the staircase in the generated data. The TTGM was trained on non-void SRTM (Shuttle Radar Topography Mission) 1-arc-second data patches in mountainous regions collected across the globe. Then, information neighboring the voids was involved in order to infer the latent encoding for the missing areas approximated to the distribution of training data. This was implemented with a loss function composed of pixel-wise, contextual, and perceptual constraints during the reconstruction process. The most appropriate fill surface generated by the TTGM was then employed to fill the voids, and Poisson blending was performed as a postprocessing step. Two models with different input sizes ($64 \times 64$ and $128 \times 128$ pixels) were trained, so the proposed method can efficiently adapt to different sizes of voids. The experimental results indicate that the proposed method can obtain results with good visual perception and reconstruction accuracy, and is superior to classical interpolation methods.

**Keywords:** digital elevation models; void filling; terrain texture; deep learning

## 1. Introduction

Digital elevation models (DEMs) are important sources of fundamental geospatial data [1,2] and play a significant role in research fields as diverse as water resources and hydrology [3], agriculture [4], geomorphology [5,6], and glaciology [7]. With the rapid development of remote sensing and photogrammetry technologies, the ability to obtain DEM data has been increasingly enhanced [8,9]. Currently, there are abundant public DEM data products covering most of the world's land areas, and they provide important topographical information for geoscience-related research [10–14]. However, the quality of DEM data is inevitably affected by the limitations of observation technology and terrain conditions involved in the process of DEM acquisition and production. The quality of commonly used DEM products, such as Shuttle Radar Topography Mission (SRTM) [12], Advanced Spaceborne

Thermal Emission and Reflectance Radiometer Global Digital Elevation Model (ASTER GDEM) [14], and TerraSAR-X add-on for Digital Elevation Measurement (TanDEM-X) [15] are affected by a large number of voids [16], which need to be reconstructed before the DEMs are used in further applications. Therefore, it is of great significance to fill the voids in DEM to meet the high demands of various geoscience applied research.

Many studies have addressed the DEM void filling tasks. In the early stage of this field, interpolation algorithms were constantly employed to fill the voids in DEM data. Commonly employed interpolation methods include inverse distance weighting (IDW), kriging, and spline [17–19]. Reuter et al. [20] evaluated the void filling performance of multiple interpolation algorithms. Interpolation methods only use the neighboring elevation values to fill voids; thus, the performance is significantly limited by the size of missing areas. Generally, it is difficult for interpolation methods to obtain reconstruction results with accurate geomorphological features.

Multi-source fusion is another effective solution for the void filling of DEM data. With the supplementary information existing in the multi-source elevation measurements involved in this method, even large-scale voids can be filled effectively. Vertical biases in the void areas in different elevation models can be eliminated by using the fill and feather (FF) technique, in which a constant is added and a feather process is applied to mitigate the abrupt change at the void edge. The delta surface fill (DSF) method proposed by Grohman et al. [21] constructs the delta surface by computing the difference in the points bordering the voids from both the original DEM data and auxiliary data, and it can account for the vertical bias of the alternative surface. Luedeling et al. [22] proposed a more promising method based on triangular irregular networks (TINs), which was an improvement of the DSF method and could better handle the varying altitude biases between different DEM datasets. Scholars have also tried to introduce the concepts of image processing and geostatistics into DEM void filling tasks. Yue et al. [23] constrained the surface modeling by the geometric characteristics of the basic quantities of the surface and constructed a high-precision surface modeling model (HASM) that could be used to fill the voids in DEM data with auxiliary data. However, the main concern is that a reliable source of information is difficult to acquire in areas with rugged topography or indistinct textures.

Recently, the reconstruction of geospatial data using feature mining and spatial prediction by neural networks has been explored. The conditional generative adversarial network (CGAN) model was applied to the spatial processes of terrain data in the work of [22] and [23]; this model can guide the direction of data generation using auxiliary information. Zhu et al. [24] designed a CGAN-based deep learning architecture that could capture features of sampled spatial data for spatial interpolation. Dong et al. [25] filled SRTM void data using a conditional adversarial network model. Gavriil et al. [26] also employed generative adversarial networks (GANs) for the task of void filling, and generative models were trained with respect to different terrain types. Their results demonstrated the ability to utilize generative networks in the context of DEM. The previous works verified the feasibility of using neural networks to understand the spatial process. However, only simulated experiments were conducted in the majority of these works; thus, the characteristics of real-data voids were not sufficiently tested. Furthermore, the topographic information is not fully utilized in the existing deep learning-based methods.

To this end, this study aimed to establish a learning mechanism of the terrain textural features to fill voids in DEM. To obtain a void filling result that has similar spatial structural patterns to those of the real DEM data, the terrain texture generation model (TTGM) is constructed considering the homogeneity and heterogeneity of terrain textures. Trained by a large number of DEM patches with complex terrain textures in mountainous areas, the TTGM was able to capture deep features of the DEM data and generate data with terrain structural patterns. The terrain texture details of the generated data were enhanced by training the model involving topographic information, such as the slope and relief degree of land surface (RDLS), together with DEM data. Additionally, a loss function composed of pixel-wise, contextual, and perceptual loss was designed to constrain data generation during the reconstruction process. The proposed method was then employed to generate a satisfactory fill surface

to fill in the void, followed by a postprocessing procedure to remove boundary artifacts. In addition, two models trained on different input sizes (64 × 64 and 128 × 128 pixels) were combined to efficiently fill the voids of arbitrary shapes.

## 2. Methodology

In this work, the voids in DEM data were filled using the proposed terrain texture generation model (TTGM) using the framework of a deep convolutional generative adversarial network. The auxiliary information contained in the global DEM tiles over a rugged topography was incorporated into the deep neural network to simulate the terrain features. The void data and the generated fill surface were then merged, and the neighboring information bordering the voids was used as a constraint. The workflow of the void filling method is given in Figure 1.

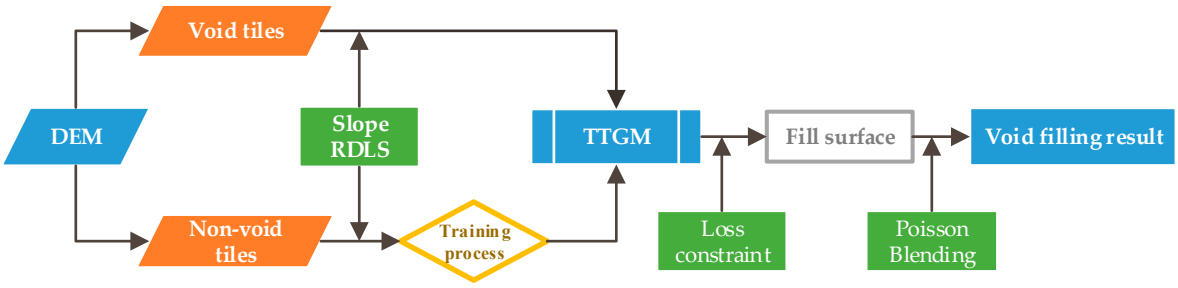



**Figure 1.** Void filling processing scheme used for the present study.

### *2.1. Terrain Texture Generation model*

#### 2.1.1. Generative Adversarial Networks

Generative adversarial networks (GANs), proposed by Goodfellow Ian in 2014 [27], consist of two networks: a generator G and a discriminator D. G takes noise as input and generates samples, whose distribution is hoped to be indistinguishable from the training distribution, while D receives both the generated fake samples and real training data and is trained as a classifier to distinguish data types. The training of G and D can be realized by min-max optimization:

$$\min_G \max_D V(G,D) = E_{h:P_{data}(h)}[\log(D(h))] + E_{z:P_z(z)}[\log(1 - D(G(z)))], \quad (1)$$

where h is a sample following the real data distribution *Pdata*, and z represents a random code of the latent space. The Adam algorithm [28] is usually the optimization method used to solve Equation (1).

GANs have made great progress in image generation and are widely applied in many fields, including image translation [29–31], text generation [32], and image inpainting [33,34]. However, in geospatial applications, the images generated by GANs usually suffer from problems such as coarse details, staircase effects, and low fidelity.

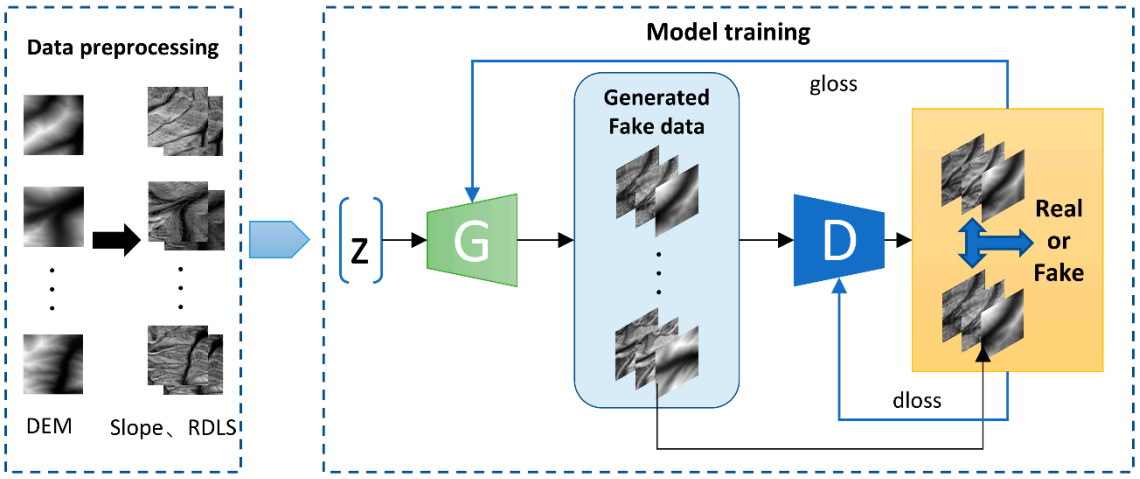
Deep convolutional GAN (DCGAN) [35] is an improvement of the GAN, and it can be used for feature discrimination and texture generation. The DCGAN model consists of a generator and a discriminator, both of which use the convolutional neural networks (CNNs) architecture. The main improvement in the DCGAN is its network structure. Some works [36,37] (e.g., Wasserstein GAN (WGAN), Least Squares GAN (LSGAN)) have mainly modified the objective function of the GAN, which also improved the stability of the training process and the quality of the generated results.

#### 2.1.2. The Framework of the Terrain Texture Generation Model

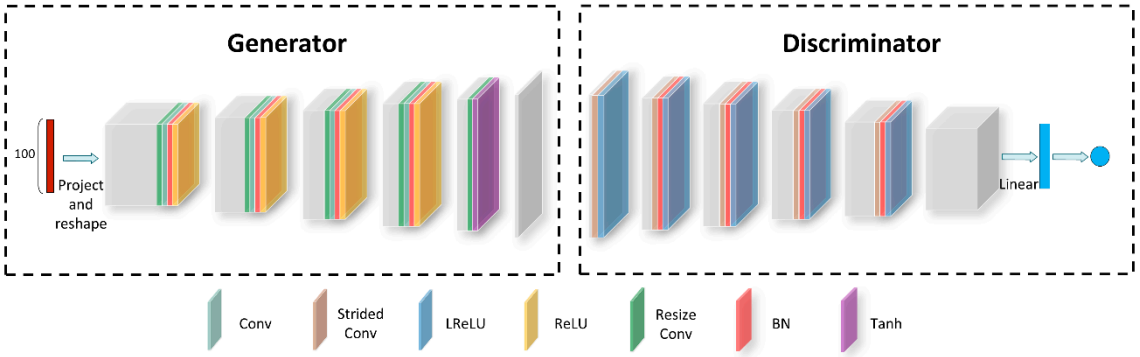
Our primary goal was to obtain a model that can simulate the natural terrain textures of DEM data and retain the intrinsic three-dimensional topographic information. Therefore, we propose a terrain texture generation model (TTGM) based on the DCGAN to address the problem of terrain

texture generation. The TTGM was trained with the WGAN adversarial loss because the WGAN value function improves the stability of training and almost solves the mode collapse problem [36].

The main architecture of the TTGM is shown in Figure 2a. To fully use the terrain features, we calculated the slope and RDLS by using elevation values and put them all into the discriminator. The generator aims to capture the potential spatial feature distribution and generate terrain texture with similar patterns to those in the training data, while the discriminator aims to extract the features from the generated data and compare them with the training data.




(a) Architecture of TTGM




(b) Detailed structure of G and D

**Figure 2.** The framework of the proposed method. (**a**) is the architecture of the terrain texture generation model (TTGM), while (**b**) is the detailed structure of the generator (G) and the discriminator (D) in the model. In the figure, $z$ indicates the random code of the latent space. The slope and relief degree of land surface (RDLS), together with the elevation values, composed the training set.

The proposed TTGM was constructed on the basis of DCGAN with a symmetrically structured generator and discriminator. The visualization of the detailed structure is displayed in Figure 2b. A 100-dimension vector z, which is sampled from a Gaussian distribution, is the input of the generator. All sample data $G(z)$ generated by the generator and the data to be input into the discriminator have three channels, which correspond to the DEM, slope, and RDLS. The last convolution layer in the discriminator is flattened and fed into a single output.

The multiple layers of deconvolution in the generator cause a checkerboard pattern of artifacts in the generated image, as shown in the first row of Figure 3. To mitigate this, we replaced the standard deconvolutional layers with resize-convolution, which is a nearest-neighbor interpolation followed by a convolutional layer. This natural approach has worked well in the removal of checkerboard artifacts. Some examples are given in Figure 3 for visual comparison. The results in the first row have heavy

checkerboard artifacts, which were generated using deconvolution, whereas nearly no checkerboard artifacts can be seen in the results obtained by resize-convolution in the second row.

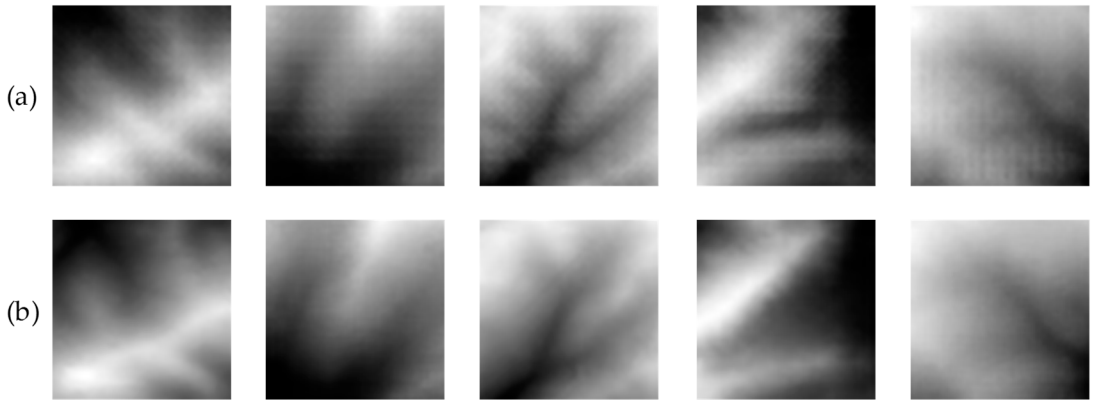



**Figure 3.** Visual comparison of digital elevation model (DEM) patches generated by the standard approach with deconvolution (**a**) and by the proposed approach with resize-convolution (**b**).

Batch normalization (BN) [38] can greatly improve the training speed and accelerate the convergence process. BN is applied to all layers except for the output layer of the generator and the input/output layer of the discriminator. Rectified linear units (ReLU) activation [39] is used in the generator for all layers except for the output layer, which uses the Tanh function. The Leaky ReLU [40], whose slope of the leak is set to 0.2, is used for all layers in the discriminator.

### *2.2. DEM Void Filling by Constrained Image Generation*

In Section 2.1, we define the structure of the model. After the TTGM is trained with a large amount of DEM data with complex terrain textures, it can generate DEM data patches that contain terrain structural patterns to be applied to fill the voids in DEM data.

In addition to the external terrain data, the neighboring information is also significant for the reconstruction of a seamless DEM. We define $M$ as a binary mask that has a value of 0 or 1. A value of 0 represents the parts of the image within the void region, while a value of 1 represents the parts of the non-void region. Here, $z$ represents the latent code, $G(z)$ denotes the corresponding generated DEM patch, and $y$ denotes the original DEM data. Filling the voids in DEM data can be simply represented by the following formula:

$$DEM_{\text{reconstructed}} = M \odot y + (1 - M) \odot G(z), \tag{2}$$

where $\odot$ denotes the element-wise product, $M \odot y$ is the uncorrupted part of the data, while $(1 - M) \odot G(z)$ represents the completed part that is used to fill the void. $M$, $y$ and the generator $G$ are all determined. The quality of the void filling result is determined by $G(z)$, which depends on $z$. Therefore, the process of obtaining a high-quality void filling result can be transformed into an optimization problem. Considering the context of DEM data and the useful data existing in void data, we designed a loss function to constrain the generation process. The most appropriate latent code $\hat{z}$ can be formulated by

$$\hat{z} = \underset{\mathbf{z}}{argmin}\left\{L_{pixel}\left(\mathbf{z}\middle|\mathbf{y}, \mathbf{M}\right) + L_{context}\left(\mathbf{z}\middle|\mathbf{y}, \mathbf{M}\right) + \lambda L_{perceptual}(\mathbf{z})\right\}, \tag{3}$$

where $L_{pixel}$, $L_{context}$, $L_{perceptual}$ denote the pixel-wise loss, context loss, and perceptual loss. $\lambda$ is the weight of the respective loss term, and it was set to 0.1 in all experiments. The smaller the value of these loss functions, the more effective $G(\hat{z})$ is in filling the voids in DEM data.

#### 2.2.1. Design of Loss Functions

Pixel-Wise Loss

The pixel-wise loss [41] emphasizes the matching of each corresponding pixel between the two images. $L_{pixel}$ is defined as

$$L_{pixel} = \|M \odot (y - G(z))\|_2. \tag{4}$$

The mean value of the pixels of the generated data close to the original data cannot account for the accuracy of the individual pixel value. Hence, the spatial patterns of the generated data may be blurry or smooth when only the pixel-wise loss is adopted. Therefore, context loss and perceptual loss were also utilized to improve the quality of the generated data.

Context Loss

For DEM data of a fixed size, the larger the void, the less the remaining available data; thus, it is worth considering how to make full use of the remaining data in the non-void region. The pixel-wise loss calculates the $l2$-norm between the uncorrupted portion of $G(z)$ and $y$; however, treating each pixel equally may not be the most appropriate approach. A void filling result with more detailed textures can be obtained by paying more attention to the data close to the void region. If the weight of a pixel in an uncorrupted region is positively correlated with the number of void pixels around it, then the farther from the void region, the smaller the pixel weight. A $9 \times 9$ window is used to traverse the DEM data. Then, the weight matrix W can be expressed as follows:

$$\mathbf{W}_i = \begin{cases} \sum_{j \in P(i)} \frac{(1-\mathbf{M}_j)}{|P(i)|} & \text{if } \mathbf{M}_i \neq 0 \\ 0 & \text{if } \mathbf{M}_i = 0 \end{cases}, \tag{5}$$

where $\mathbf{W}_i$ denotes the weight at pixel location $i$, $P(i)$ refers to the set of neighbors of pixel $i$ in a local window, and $\left|P(i)\right|$ represents the number of all pixels on it. $\mathbf{M}_i$ is the value of pixel $i$ in the binary mask $\mathbf{M}$.

Given the robustness of the $l1$-norm, a spatial-weighted context loss function with respect to the relative features of the surrounding uncorrupted regions is constructed and defined as follows:

$$L_{contextual} = \|\mathbf{W} \odot (G(\mathbf{z}) - \mathbf{y})\|_1. \tag{6}$$

Perceptual Loss

The pixel-wise loss and context loss quantify the difference between the generated data and original data. However, the generated data may not guarantee continuity with the original DEM data within the void region when the loss value is minimal. The discriminator is trained to distinguish the true data from generated fake data to obtain generated data in which the qualitative result is close to the original DEM data. We designed a perceptual loss to penalize implausible generated data:

$$L_{percetual} = \log(1 - D(G(\mathbf{z}))). \tag{7}$$

#### 2.2.2. Seamless Blending

Backpropagation on the defined total loss function in Equation (3) is used to update $\mathbf{z}$, and the closest $\hat{z}$ can eventually be found and applied to generate $G(\hat{z})$. Although the generated DEM data patch has similar spatial structural patterns to those in the original DEM data, there is still a vertical deviation between them, in which case, filling the void directly with $G(z)$ forms a breaking trace at the edge of the seam line. In this study, we adopted the Poisson fusion algorithm [42] for the seamless blending of the data to be filled and the generated filling source. A smooth transition

of the reconstruction result at the void boundary was ensured by constructing a gradient Poisson correction equation:

$$\hat{x} = arg\ \min_{x} \|\nabla x - \nabla G(\hat{z})\|_2^2,\ s.t.\ x_i = y_i \quad for \quad M_i = 1 \quad , \tag{8}$$

where $\hat{x}$ represents the final solution, $x_i$ equals $y_i$ at the uncorrupted region of the original DEM data, and $\nabla$ denotes the gradient operator. Two void filling results are shown in Figure 4. It is clear that the merged result (Figure 4d) shows smoother transitions over the boundary of the voids than the directly filled data (Figure 4c).

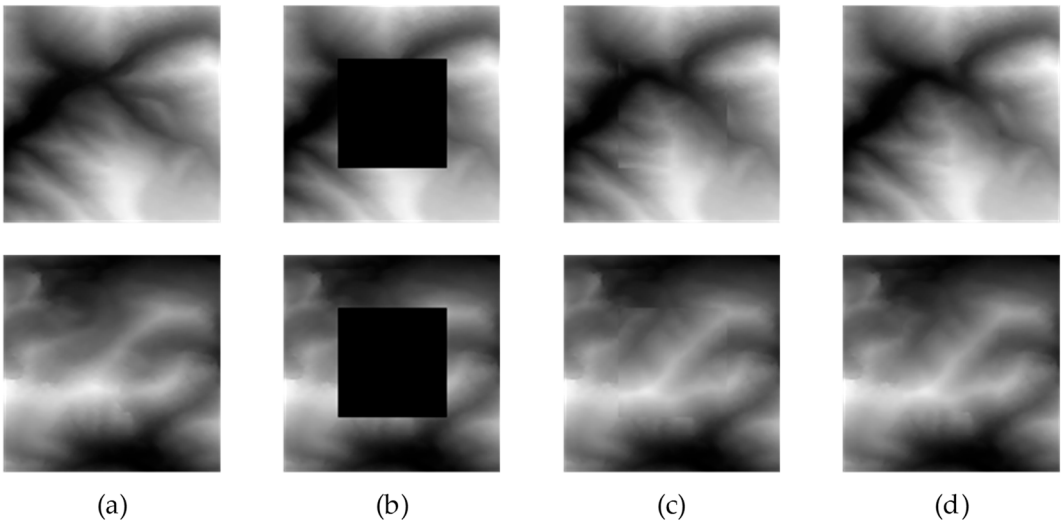



**Figure 4.** Void filling results with and without blending. (**a**) Original DEM data; (**b**) void data; (**c**) void filling results without blending; (**d**) void filling results with blending.

## 3. Experiments

### *3.1. Data Description and Evaluation Metrics*

We selected the 30 m resolution SRTM-1 global data to conduct exploratory experiments on the proposed method. In order to train the model to capture complex terrain textures and to prevent the model from over-memorizing and overfitting, we selected the training data from four representative mountainous areas around the world: the Rocky Mountains, the Andes, the Atlas Mountains, and the Kunlun Mountains, as shown in Figure 5. The trained model was then applied to the reconstruction of global void tiles. A total of 40 1° × 1° SRTM-1 DEM tiles were selected in these areas, with the terrain elevations ranging from 104 to 8583 m. Each tile is composed of 3601 × 3601 pixels.

The selected DEM tiles were divided into 70,000 64 × 64 pixel patches and 50,000 128 × 128 pixel patches. The pixel patches were distributed so that 70% formed the training set, and 30% formed the test set. For each piece of data, a moving 3 × 3 window (Figure 6a) was employed to traverse it. The slope value at the central pixel $e$ can be calculated by the algorithm proposed in [43]:

$$\text{Slope} = arctan\sqrt{Slope_{we}^2 + Slope_{sn}^2}, \tag{9}$$

$$\text{Slope}_{we} = \frac{(e_8 + 2e_1 + e_5) - (e_7 + 2e_3 + e_6)}{8 \times \text{Gridsize}}, \tag{10}$$

$$\text{Slope}_{sn} = \frac{(e_7 + 2e_4 + e_8) - (e_6 + 2e_2 + e_5)}{8 \times \text{Gridsize}}, \tag{11}$$

where Gridsize denotes the spatial resolution of the DEM data, $Slope_{we}$ and $Slope_{sn}$ are the calculation results of the slope in the horizontal direction and vertical direction, respectively, and Slope refers to the final slope value at the local point $e$.

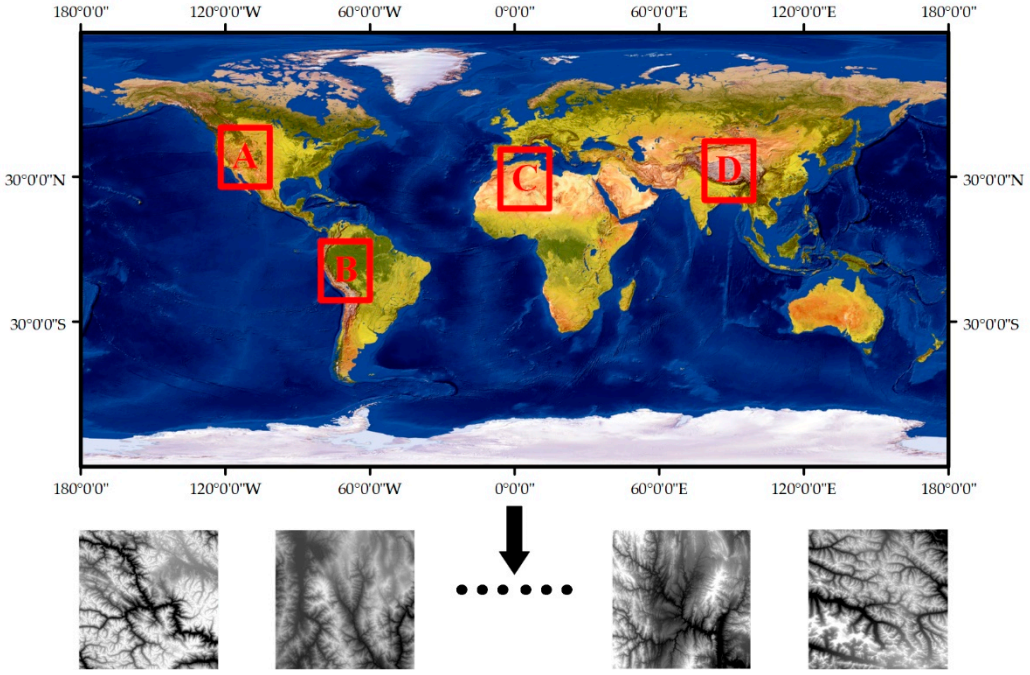



**Figure 5.** Study areas for the DEM void filling method; Subregion A is the Rocky Mountains; Subregion B is the Andes; Subregion C is the Atlas Mountains; Subregion D is the Kunlun Mountains.

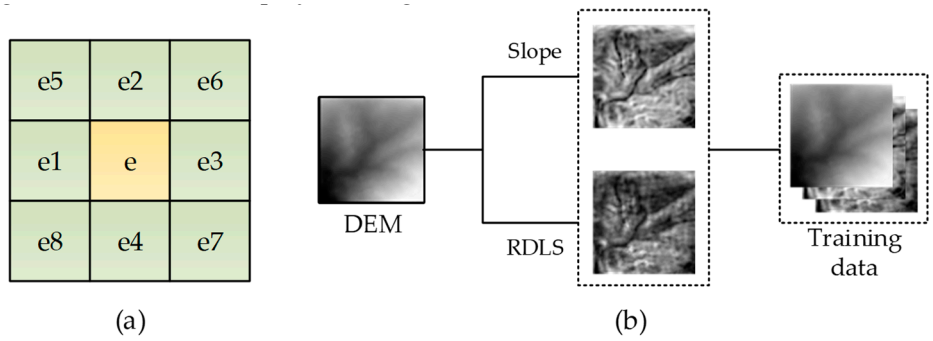



**Figure 6.** Construction of training dataset. (**a**) The 3 × 3 moving window; (**b**) calculation results of the slope and RDLS.

The value of RDLS at the central pixel $e$ is calculated by the following formula:

$$R = e_{max} - e_{min}, \tag{12}$$

where $e_{max}$ and $e_{min}$ are the maximum and minimum elevation values in the window, respectively. In addition to the DEM patches, the slope and RDLS were also incorporated to construct the training dataset, which is displayed in Figure 6b.

For a thorough evaluation of the performance of the proposed method, four types of artificially created voids were used to construct the test data in the simulated experiments. An example is shown in Figure 7.

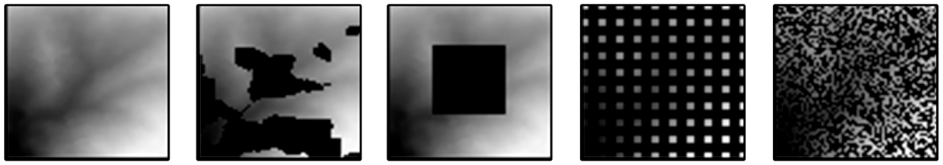



**Figure 7.** Test with different types of voids. (**a**) The original DEM data; (**b**) real pattern voids; (**c**) central block voids; (**d**) sample missing voids; (**e**) 80% missing random voids.

The accuracy of the void filling results was evaluated by comparing the resulting data with the original DEM data. Several metric indexes, namely, mean error (ME), standard deviation (SD), mean absolute error (MAE), and root-mean-square error (RMSE), were used to evaluate the differences between them. Let $y_i$ and $\hat{y}_i$ denote the void filling result and the original DEM within the void area, respectively. Then, the metric indexes can be calculated by the following function:

$$\mathrm{ME} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i), \tag{13}$$

$$\mathrm{SD} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}((y_i - \hat{y}_i) - u)^2}, \tag{14}$$

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|(y_i - \hat{y}_i)\right|, \tag{15}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}, \tag{16}$$

where N is the number of void pixels within the data tile, and $u$ is equal to the mean error. In practice, these metric indexes (ME, SD, MAE, RMSE) only reflect the statistics of the elevation error. Because the topographic texture is critical for delineating the terrain features, we also used the structural similarity index (SSIM) [44] to evaluate the structural similarity between the generated DEM patch and the original DEM data:

$$\mathrm{SSIM}(x, y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_1\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right) + \left(\sigma_x^2 + \sigma_y^2 + c_2\right)}, \tag{17}$$

where x and y are the DEM data generated by the neural network and the original DEM data used for reference; $\mu_x$ and $\mu_y$ are the mean of x and y; $\sigma_x^2$ and $\sigma_y^2$ are the variance of x and y; $\sigma_{xy}$ is the covariance of x and y; $c_1 = (K_1 L)^2$ and $c_2 = (K_2 L)^2$; $K_1 = 0.01$, $K_2 = 0.03$; and $L$ is the dynamic range of pixel values.

### 3.2. Simulated Experiments

#### 3.2.1. Void Filling of Simulated Real Pattern Voids

Given the varying sizes and shapes of the real data voids, the TTGMs were trained with two sizes of input patches (64 × 64 and 128 × 128); the two models are designated TTGM64 and TTGM128. The two models were applied to randomly selected void data patches of two sizes, 150 of each, and the quantitative results are given in Table 1. The visual results of 16 void-filled data patches are displayed in Figure 8. The original DEM data were used as a reference for accuracy evaluation. The quantitative accuracy of TTGM128 is relatively low because there is generally a larger number of missing pixels in the 128 × 128 data. Both models obtained results with high SSIM values, which indicate that the

proposed method is capable of generating natural terrain textures. It can be clearly seen that the generated data (in Figure 8c) are highly similar to the original DEM when the void area is relatively small. Even if the void area accounts for more than 50%, the generated data remain highly consistent with the original DEM in terms of the texture structural patterns. The seamless void-filled results show good visual fidelity, as illustrated in Figure 8d.

**Table 1.** Quantitative results of TTGM64 and TTGM128.

| Unit: m | ME | SD | MAE | RMSE | SSIM |
|---|---|---|---|---|---|
| TTGM64 | 0.86 | 18.81 | 15.22 | 20.21 | 0.92 |
| TTGM128 | 0.12 | 22.99 | 17.18 | 24.02 | 0.90 |

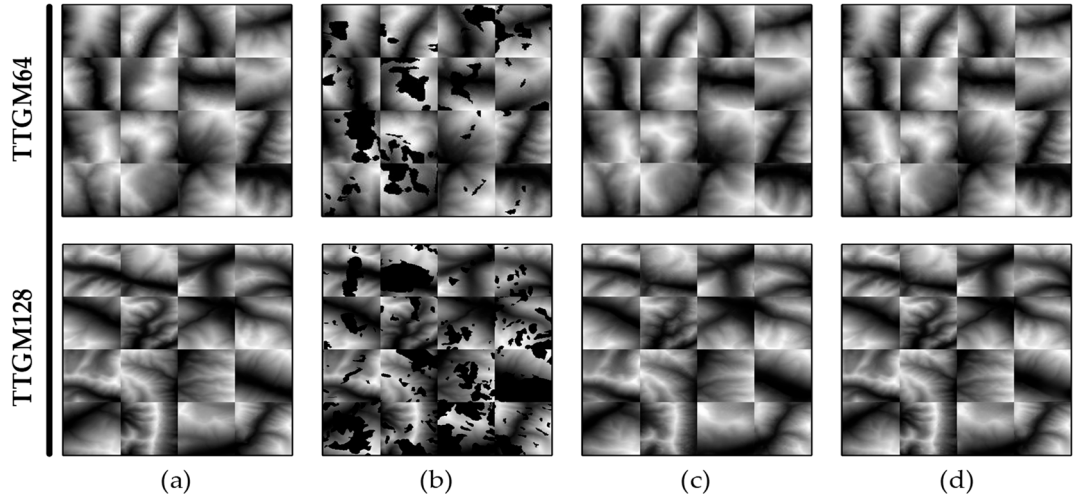



**Figure 8.** Void filling results of TTGM64 and TTGM128, indicating the trained TTGMs with two sizes of input patches (64 × 64 and 128 × 128). (**a**) The original DEM data; (**b**) test data with real type voids; (**c**) generated DEM patches; (**d**) void filling results.

In this section, we also investigate the relationship between the void area ratio and the accuracy of the void filling results. It is clear from Figure 9 that the correlations between the void area ratio and RMSEs are all positive in the two models, namely, TTGM64 (Figure 9a; $r = 0.74$, $p < 0.01$) and TTGM128 (Figure 9b; $r = 0.65$, $p < 0.01$), indicating that the accuracy of the void filling result decreases as the void area grows larger. The reason might be that as the void area increases, the information in the image is utilized less effectively; thus, the accuracy of the void filling results is significantly affected. Suppose that a void area accounts for 50% in a 64 × 64 data patch and that the number of void pixels only accounts for 12.5% in a 128 × 128 patch. In this case, it is more advantageous to choose TTGM128 for the void filling task. However, with a smaller size of input patch, it is easier for the model to learn the terrain texture structure during the training process. Therefore, in this set of experiments, the overall accuracy with which TTGM64 fills the void data is better. In this regard, we propose a multiscale void filling strategy in which the void tiles are divided into differently sized patches (64 × 64 or 128 × 128), depending on the sizes of the voids. In real cases, the constructed test patches are input to the corresponding model, and the obtained results are then merged into a seamless DEM.

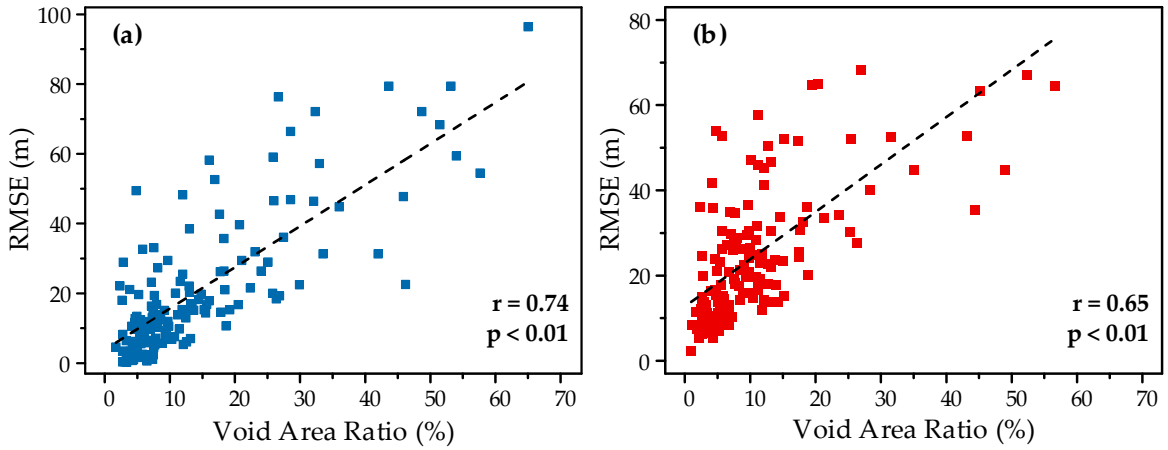



**Figure 9.** Correlation scatterplot of RMSEs and the void area ratio. (**a**) The plot of TTGM64; (**b**) the plot of TTGM128.

### 3.2.2. Application of the Proposed Method on Simulated SRTM Void Data

As described in this section, an experiment was conducted on simulated SRTM void data to demonstrate the utility of our void filling strategy. We clipped a 1000 × 1000 region from the SRTM1 data (Figure 10a) and added artificially created voids to simulate void data (Figure 10b). First, data voids were detected, and the number of void pixels for each segmented void was determined. If the number was less than 1000, then the void and its surrounding area were cropped to a 64 × 64 patch and processed by TTGM64; otherwise, a 128 × 128 patch was cropped and input to TTGM128. If the void region could not be covered by a 128 × 128 patch, several patches were used to cover the area in which the void is located. The patches were then used as the input to the proposed model for void filling.

The generated data were then merged without any processing, so void areas were only replaced with the generated data, as shown in Figure 10c. The results indicate that the proposed TTGM effectively learned the topographic structural patterns and inferred the spatial pattern of the DEM data using the neighboring information bordering the voids. The final void filling results are shown in Figure 10d. The results indicate that the generated data are highly consistent with the original data. Moreover, the statistical information for both the original DEM data and the void filling result within the void area shows a very close distribution, as reported in Table 2. All the results indicate that the accuracy of the generated data and the feasibility of this void filling strategy are validated.

**Table 2.** Statistical information for the original DEM data and void filling result in the void area.

| | Min | Max | Mean | STD |
|---|---|---|---|---|
| Original DEM | 4090 | 5537 | 4928 | 226 |
| Void filling result | 4096 | 5529 | 4931 | 225 |

In addition, we extracted the topographic feature of the DEM to judge the reliability of the repaired DEM data. DEM data can be used to extract topographic factors (e.g., slope and aspect), topographic features (e.g., ridges, valleys, and peaks), and river networks, among other features [45,46]. By analyzing the derived features of the DEM data, we can explore their potential applications.

In this regard, we compared the river networks extracted from the original DEM data and the void filling results. In Figure 11, the drainage lines extracted from the void filling result are displayed in blue, while the red lines denote the reference river network extracted from the original DEM. The extracted river networks are superimposed and displayed. The void pixels are displayed in black. Figure 11a shows that the two river networks are highly consistent in the position of the flow and branches. In order to better compare the extraction results, we provide zoomed views of four subregions in

Figure 11b–e. Overall, the void filling results obtained show extracted hydrological features that are continuous and topologically consistent with the reference river network.

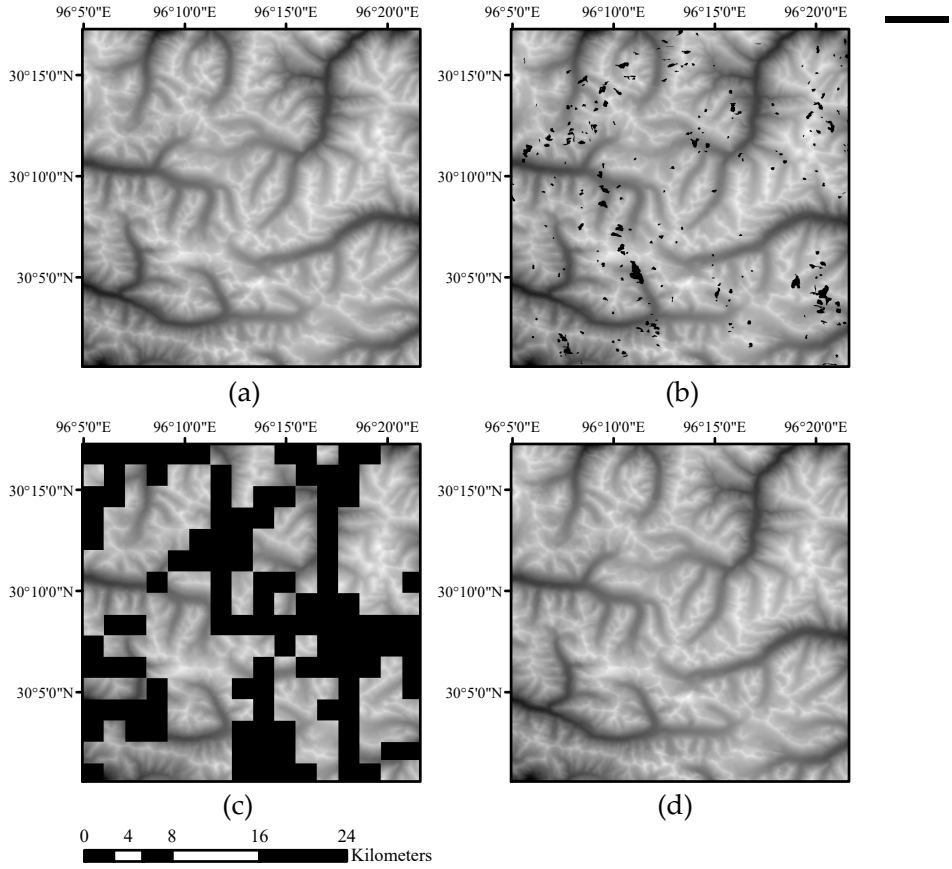



**Figure 10.** Simulated experiment on SRTM-1 data. (**a**) The original DEM data; (**b**) test data; (**c**) generated data patch after being mosaicked; (**d**) void filling result.

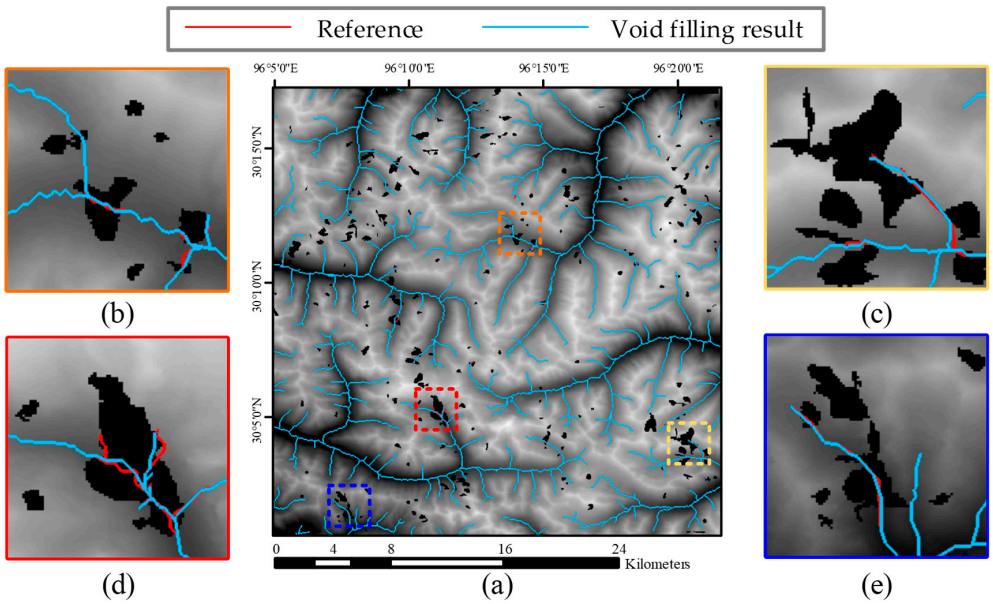



**Figure 11.** Comparison of river networks. (**a**) The extraction results of the original DEM data and void filling result; (**b**–**e**) zoomed views of the four subregions.

### *3.3. Comparison with Interpolation Methods*

The voids in DEM data can be partly filled and reconstructed by combining the multi-source elevation information [47,48]. However, obtaining a reliable information source for significantly rugged topography is difficult. For those areas, interpolation algorithms are commonly used to fill the voids. In this section, we compare our method with three classical algorithms, namely, kriging, spline, and IDW.

The interpolation algorithms were implemented using the 3D analyst module in ArcGIS Toolbox (Esri, Redlands, USA) with all the parameters set to default values. To fully validate the effectiveness of the proposed method in the general cases, several tests were conducted with DEM tiles randomly selected all over the globe, and artificial voids were added to simulate test data, as shown in Figure 12. The accuracy curves of the six DEM tiles are shown in Figure 13. The curves of the MAE and RMSE show the similar trends, and the green curves that represent results of the proposed method contain the highest accuracy.

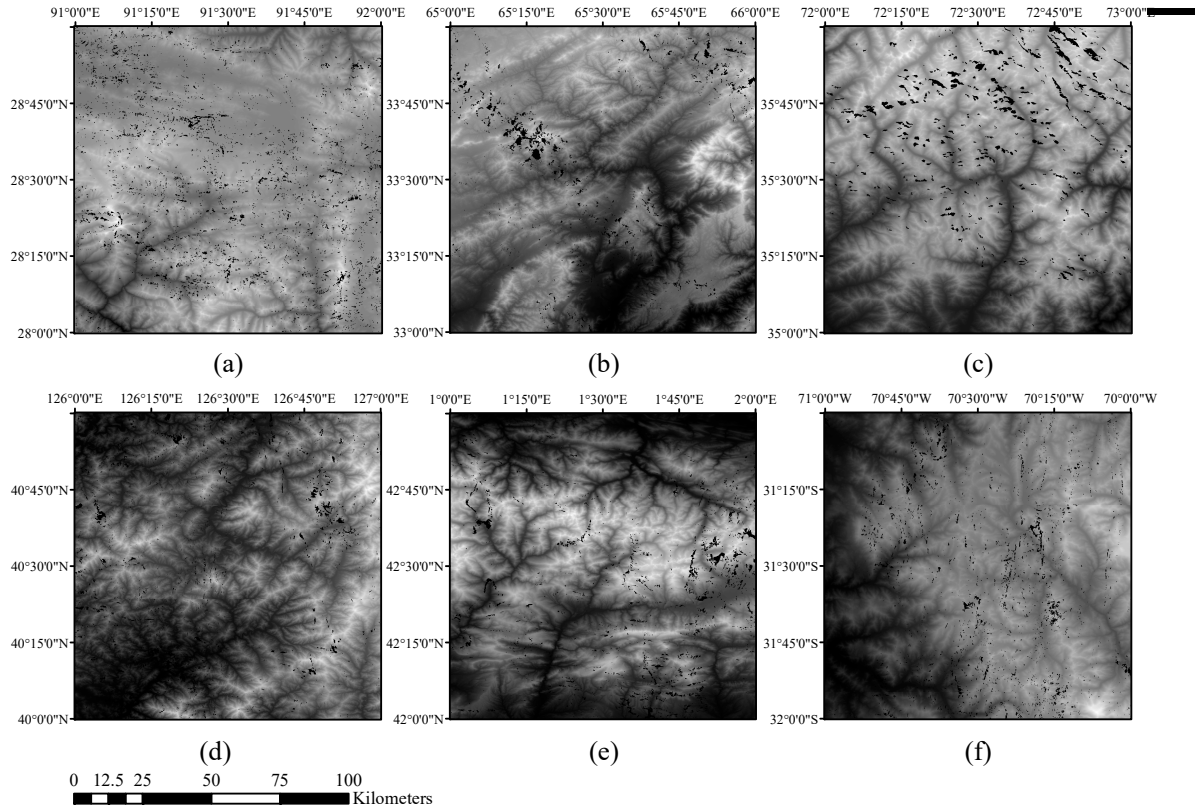



**Figure 12.** Simulate voids in SRTM-1 data. (**a**–**f**) Six DEM tiles selected from all over the globe.

As shown in Figure 14a, the tile numbered N35_E072 (Figure 14a) and its simulated test data (Figure 14b) were selected for further illustration of the performance of these methods. There were 480 voids in the test data, with a total number of 374,486 pixels within the void areas. Table 3 shows the accuracy statistics of the void filling results, and the visual results are shown in Figure 14c–f. For a better comparison of the results, we provide zoomed views within the red rectangular region in Figure 15.

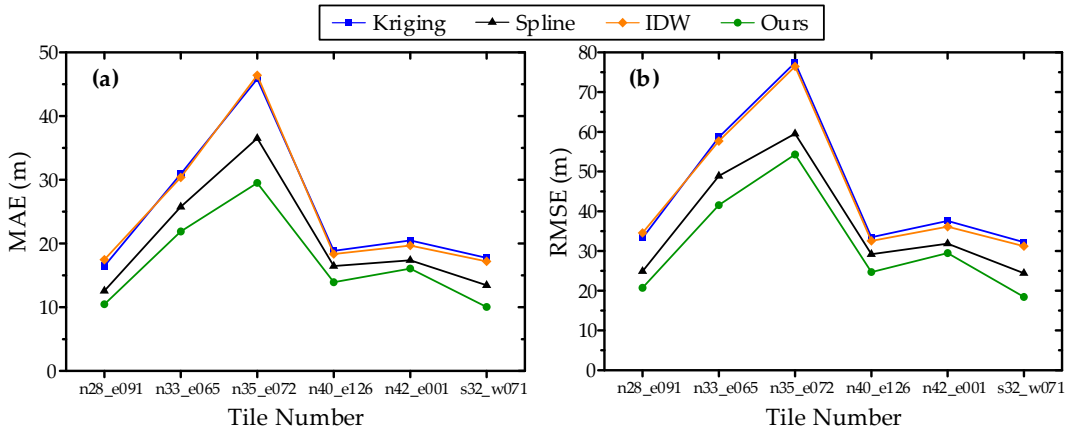



**Figure 13.** The accuracy assessment of kriging, spline, and IDW (inverse distance weighted), and the proposed method for filling the voids in simulated test data. (**a**) and (**b**) show the mean absolute error (MAE) and the root mean square error (RMSE), respectively.

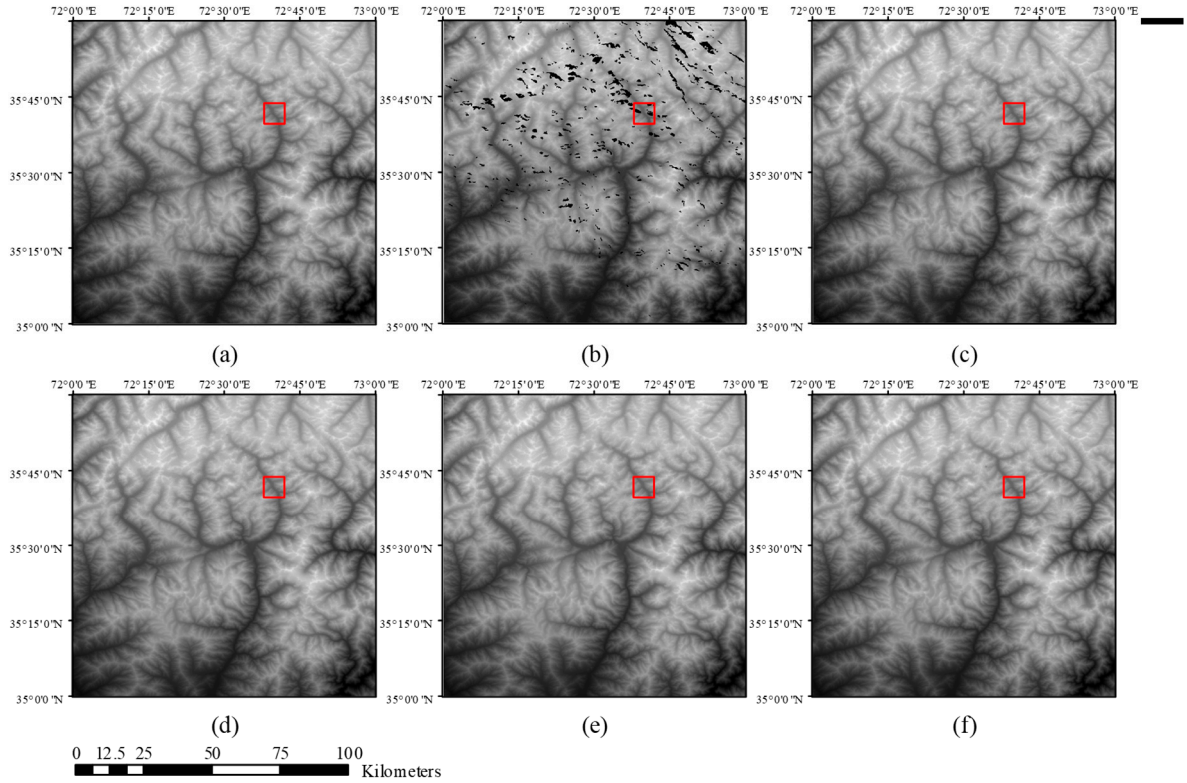



**Figure 14.** Simulated experiment on SRTM-1 data. (**a**) The original DEM data; (**b**) simulated test data; (**c**–**f**) the void filling results of Kriging, Spline, IDW, and the proposed method.

**Table 3.** Quantitative results of kriging, spline, and IDW, and the proposed method for filling the voids in the tile numbered 'N35_E072'.

| | Kriging | Spline | IDW | Ours |
|---|---|---|---|---|
| ME | 0.10 | 0.73 | −0.46 | −0.59 |
| SD | 77.42 | 59.51 | 76.44 | 54.28 |
| MAE | 45.84 | 35.52 | 46.41 | 29.51 |
| RMSE | 77.42 | 59.51 | 76.44 | 54.29 |

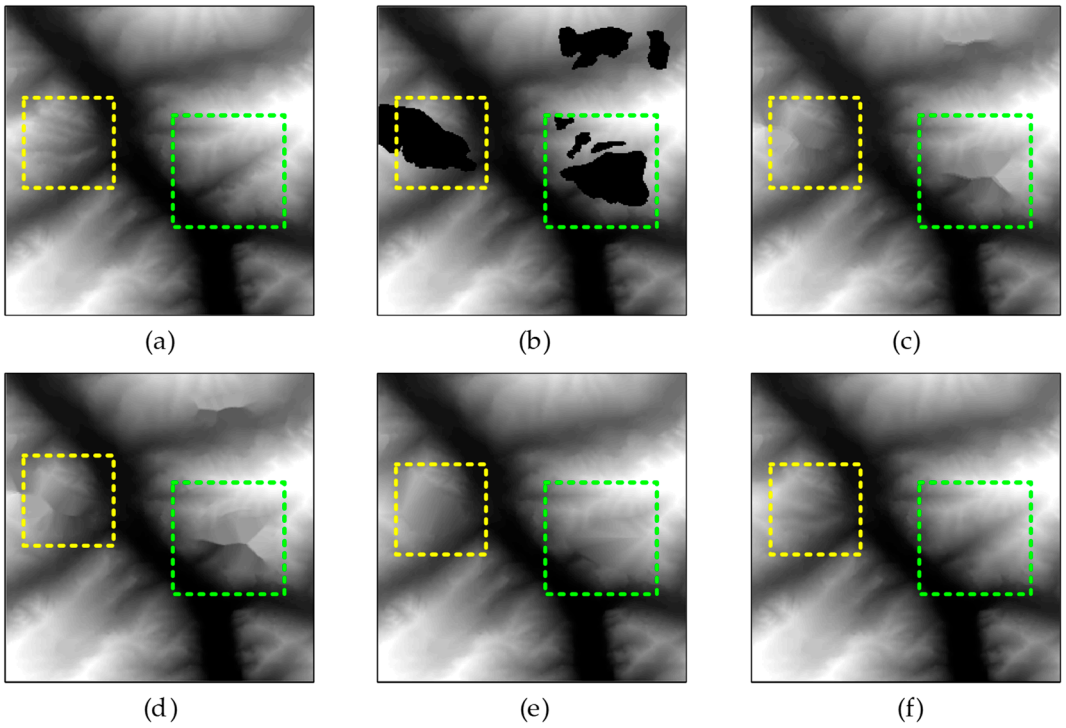



**Figure 15.** Zoomed displays of the red rectangular region in Figure 14. (**a**) The original DEM data, (**b**) simulated test data; (**c**–**f**) the void filling results of Kriging, Spline, IDW, and the proposed method.

Combined with the visual and quantitative results, it is obvious that the proposed method obtained much better results than all three interpolation methods. Although the result of the spline interpolation algorithm has better statistics than IDW and kriging, the visual results (Figure 15c–e) show that it is over-smooth, so it lost most of the effective terrain information within the void regions. Furthermore, 20% of the voids contain more than 1000 missing pixels, with the largest voids containing 17131 missing pixels; thus, the RMSE and SD of the results are relatively high. The spline interpolation algorithm adapts to small and medium-sized voids in dissected terrain and high altitude, while the kriging and inverse distance weighting interpolation algorithms are more suitable for large voids in flat areas. Overall, the interpolation methods were not capable of dealing with the large voids in the experiments, in which only limited neighboring spatial information was involved. The proposed method applies deep learning to learn the statistical structural pattern from terrain data collected at a global scale and thus obtains good visual results in cases with voids in rugged regions.

### *3.4. Real-Data Experiments*

This section presents real-data experiments that were conducted to further verify the proposed method. Given that the spline method outperformed the other two interpolation methods in Section 3.3, we only compared our method with spline in the real-data experiments. The qualitative results are given in Figure 16, in which the two rows represent a set of real-data experiments. In terms of the overall visual perspective of the void filling results, both methods successfully filled the voids in the DEM data. More detailed visual results within the red and green rectangular regions are displayed in the lower-right corner of each image. It is clear that the void filling result of spline interpolation tends to be excessively smooth. On the contrary, the proposed method can make full use of the existing information in the void data and produce data with terrain textures. However, because of the lack of ideal reference data for quantitative evaluation, statistical results cannot be provided. Good visual results do not necessarily translate to higher quantitative results; thus, we aim to further improve our method to obtain reconstruction results with higher vertical accuracy.

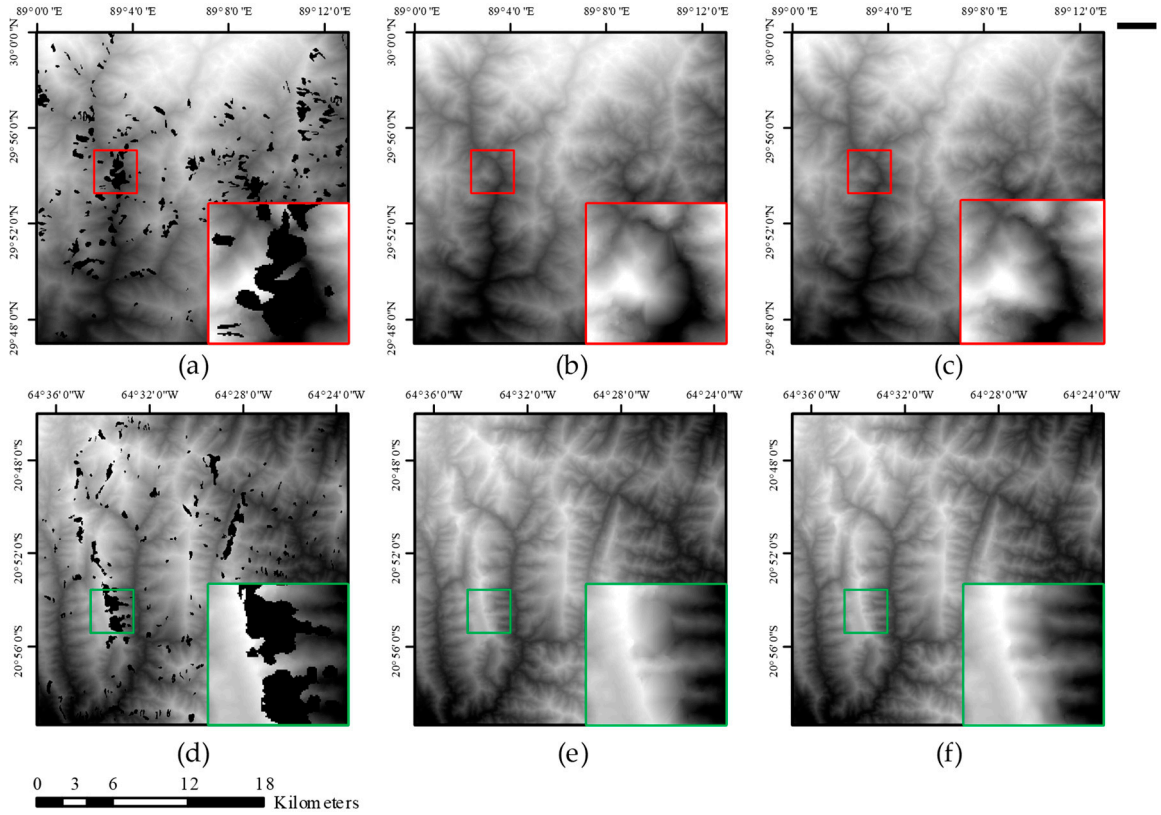



**Figure 16.** Real-data experiments. (**a**)–(**c**) are the partial results of data tile N29E089, and (**d**)–(**f**) are the results of data tile S21W065. From left to right: void data, void filling result from the spline method, and void filling result from our proposed method.

## 4. Discussions

### *4.1. Evaluation of the Void Filling Method with Respect to Input*

In this study, the proposed TTGM was trained on a large number of samples, which were constructed with elevation, slope, and RDLS, to learn the texture structure in the DEM and generate data with terrain texture features. To evaluate the influence of incorporating the topographic information on the performance of our proposed method, we trained two models with and without the slope and RDLS, referred to as TTGM-D and TTGM-DSR, respectively. The models were employed to fill the test data with real and center-type voids, and the accuracy evaluation results are compared in Table 4. The metrics indicate that TTGM-DSR performed better in the simulation experiments, in which MAE and RMSE mainly reflect the vertical accuracy and SSIM corresponds to the structural similarity.

**Table 4.** Comparison of quantitative results of TTGM-D and TTGM-DSR.

| Void Type | Model | ME | SD | MAE | RMSE | SSIM |
|---|---|---|---|---|---|---|
| Real | TTGM-D | 1.39 | 18.58 | 14.75 | 20.67 | 0.89 |
| | TTGM-DSR | 2.01 | 18.42 | 14.85 | 19.78 | 0.91 |
| Center | TTGM-D | 1.48 | 27.53 | 23.35 | 29.55 | 0.88 |
| | TTGM-DSR | −2.98 | 26.51 | 21.39 | 28.14 | 0.90 |

Mini-batches of the original DEM and the data generated by TTGM-DSR and TTGM-D are displayed in Figure 17 for visual comparison. The generated DEM data are generally similar to the original data. The apparent differences between the results obtained by the two models can be observed in the red rectangular regions. It can be seen that TTGM-D generated some false terrain textures that

are quite different from the original DEM data, while the terrain structural patterns resulting from TTGM-DSR are similar to those in the original DEM data as a whole. The qualitative and quantitative results prove that the addition of auxiliary data has a certain enhancement effect on the ability of the model to learn the terrain textures in DEM data.

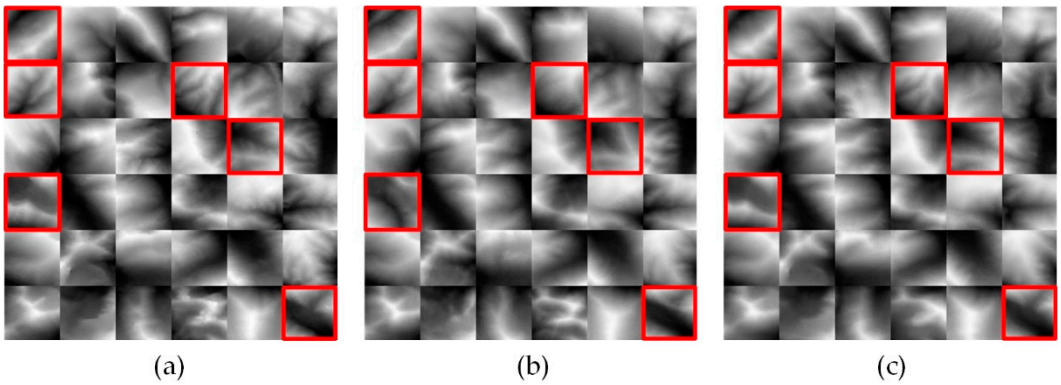



**Figure 17.** Visual comparison of generated DEM patches. (**a**) The original DEM data; (**b**) TTGM-D; (**c**) TTGM-DSR.

### *4.2. Validation of the Model's Ability to Simulate Terrain Texture in the DEM*

The ability of our model to generate data with terrain texture information was demonstrated by applying the model to mini-batches of DEM data with different types of voids that were chosen from the test dataset. We invoked the generator every 50 iterations during the void filling process, inputted the test data into the generator, and calculated the values of MAE and SSIM to compare the output texture-like DEM patches and the original DEM data. The evolution of model generations can be identified in the plots of Figure 18, in which the green error curve refers to MAE, and the blue curve refers to SSIM.

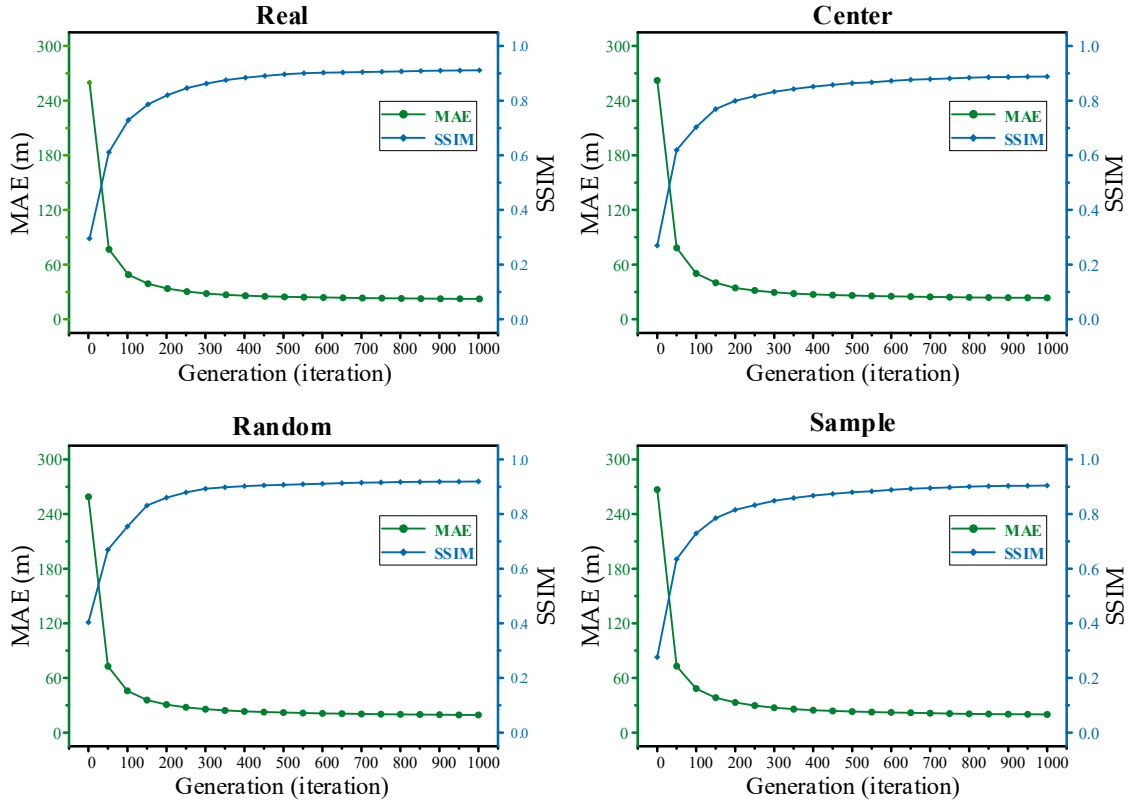



**Figure 18.** Generation process based on the test data with four different types of voids.

As indicated in Figure 18, the curves in the four plots all have similar trends, and all of them achieve the convergent status by the end of the generation. If we take the result in the first plot as an example, it is obvious that the first 50 iterations experience a sharp decrease in the value of MAE. However, after that, the value of MAE decreases slowly and levels off at around 20 m. The trend of the blue curve is the opposite of that of the green curve, and the value of SSIM reaches around 0.9 in the final generation.

A comprehensible visualization of the evolution of the generated DEM patches regarding visual fidelity is shown in Figure 19. We enumerated the generated data by iterations (1, 50, 100, 500, and 1000) between the original DEM data and the input void data as an example. It is obvious that the data generated by the generator are cluttered at the very beginning. After 50 iterations, coarse terrain structural patterns can be seen in the generated results. Then, as the number of iterations increases, the loss function constrains the generation of images during backpropagation, while terrain details such as peaks and valleys begin to appear during the evolution. By iteration 1000, high-quality generated DEM data, which are almost visually indistinguishable from the original DEM data, are obtained.

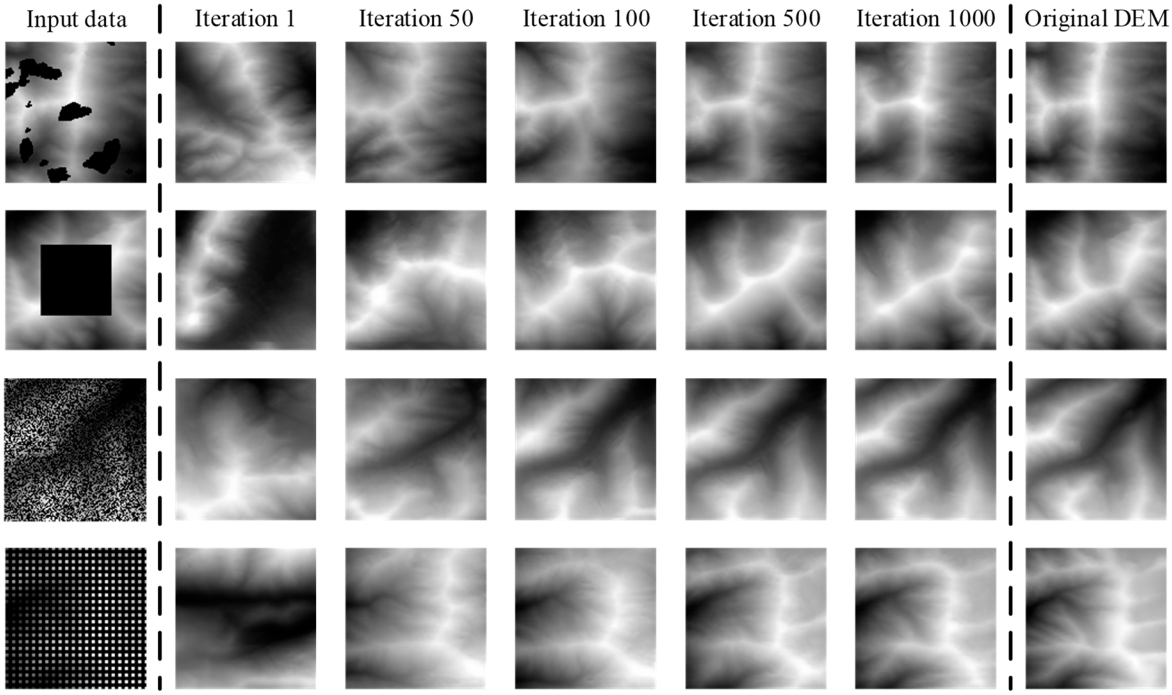



**Figure 19.** Visual evolution of the generated DEM patches. From left to right: input test data, generated DEM patches at different iterations, the original DEM data.

Moreover, as shown in the second row of the results, there is a slight difference in the structure of the peaks between the generated data and original DEM data because it is difficult for the model to effectively use the information around us to accurately fill such a large void in the center. Comparatively, the void ratios in the test data in the third and fourth rows are above 70%, and the generated DEM patches have better visual fidelity. Although the void area ratio is large, the effective pixels are distributed throughout the image. In those cases, the proposed model can generate data with better terrain structural patterns. Overall, the model performs well on test data with four types of voids. These results prove that our model is capable of simulating the terrain spatial texture features of the DEM and is not generating data by simply memorizing training samples.

### *4.3. Influence of Terrain Factors on Generated Images*

The relief and slope are topographic factors that reflect changes in terrain. In this section, we study the influences of these terrain factors on the quality of the generated data. A batch of 200 128 × 128

data was selected for the experiment. For each data patch, we calculated the average absolute error of the original data and the generated image. The mean value of the relief and slope were calculated to characterize the topographic features of the whole image.

A correlation analysis between the relief, slope, and MAE was conducted, and the correlation scatterplots are given in Figure 20. It is obvious that the correlations between the topographical factors and MAE are all positive: relief and MAE (Figure 20a; $r = 0.74$, $p < 0.01$) and slope and MAE (Figure 20b; $r = 0.70$, $p < 0.01$). Generally, vertical accuracy of most generated DEM patches in which the slope range in 20°–40° are relatively concentrated and high, for the TTGM was trained based on a large amount of data in mountainous areas. The generated DEM patches can acquire higher accuracy in flat areas with small slope and RDLS values. Although the proposed model has a good performance in simulating the terrain textures in the rugged terrain, which is significant for the extraction of the terrain features, e.g., river networks and ridge lines. However, the accuracy inevitably tends to decrease for the data tiles with large RDLS and slope values.

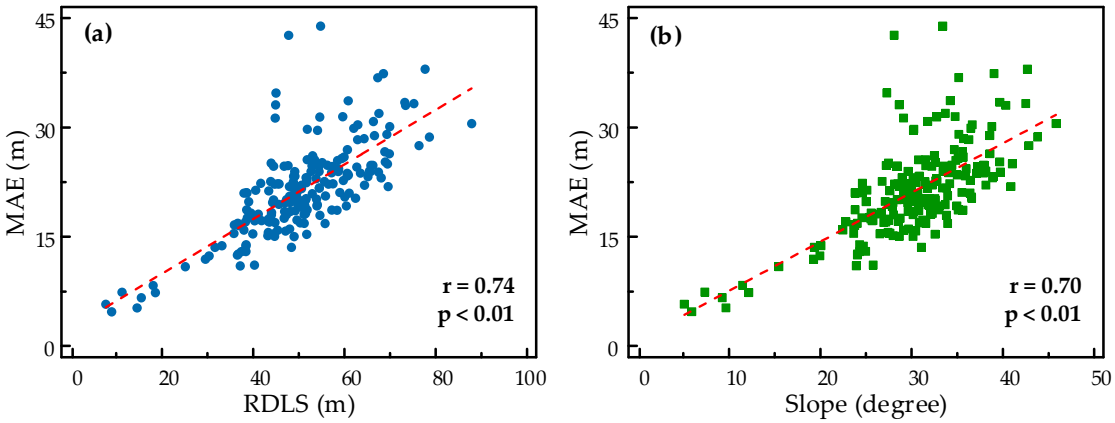



**Figure 20.** The relationships between (**a**) RDLS and MAE, (**b**) slope and MAE.

## 5. Conclusions

In this paper, we present a void filling method that uses a TTGM for the reconstruction of areas in which data are missing in DEM. The TTGM was constructed to capture deep geospatial features and generate good-quality DEM patches. In addition to the elevation information, the slope and RDLS were incorporated as inputs to be fed into the neural network during the training process. The proposed method utilizes the TTGM to obtain an ideal fill surface under the constraint of the loss function, and it uses a postprocessing step to seamlessly merge the fill surface with the void data. Experiments were conducted on SRTM DEM data with artificial voids. Compared with classical interpolation methods, our method obtained void filling results with better evaluation accuracy and more texture details that look much more realistic. The visual results and comparison of the derived topographic features in the experiments with simulated data and real DEM tiles demonstrate the effectiveness of our void filling strategy. Moreover, we present a detailed analysis of the TTGM for varying cases with different ratios of missing pixels and model parameters.

**Author Contributions:** Conceptualization, L.Y.; methodology, L.Y. and Z.Q.; investigation, L.Y. and Z.Q.; writing—original draft preparation, Z.Q.; writing—review and editing, L.Y. and X.L.

**Funding:** This research was funded by National Natural Science Foundation of China (41801263), Open Fund of Key Laboratory of Geographic Information Science (Ministry of Education), East China Normal University (Grant No. KLGIS2017A03) and the Fundamental Research Funds for the Central Universities, China University of Geosciences (Wuhan) (CUG170663).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References


1. Xiong, L.; Tang, G.; Yan, S.; Zhu, S.; Sun, Y. Landform-oriented flow-routing algorithm for the dual-structure loess terrain based on digital elevation models. *Hydrol. Process.* **2014**, *28*, 1756–1766. [CrossRef]
2. Zhou, Q.; Liu, X. Analysis of errors of derived slope and aspect related to DEM data properties. *Comput. Geosci.* **2004**, *30*, 369–378. [CrossRef]
3. Wechsler, S. Uncertainties associated with digital elevation models for hydrologic applications: A review. *Hydrol. Earth Syst. Sci. Discuss.* **2007**, *11*, 1481–1500. [CrossRef]
4. Fu, P.; Rich, P.M. A geometric solar radiation model with applications in agriculture and forestry. *Comput. Electron. Agric.* **2002**, *37*, 25–35. [CrossRef]
5. Ehsani, A.H.; Quiel, F. Application of Self Organizing Map and SRTM data to characterize yardangs in the Lut desert, Iran. *Remote Sens. Environ.* **2008**, *112*, 3284–3294. [CrossRef]
6. Zhang, Z.; Wang, C.; Zhang, H.; Tang, Y.; Liu, X. Analysis of Permafrost Region Coherence Variation in the Qinghai–Tibet Plateau with a High-Resolution TerraSAR-X Image. *Remote Sens.* **2018**, *10*, 298. [CrossRef]
7. Yue, L.; Shen, H.; Yu, W.; Zhang, L. Monitoring of Historical Glacier Recession in Yulong Mountain by the Integration of Multisource Remote Sensing Data. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2018**, *11*, 388–400. [CrossRef]
8. Li, X.; Shen, H.; Feng, R.; Jie, L.; Zhang, L. DEM generation from contours and a low-resolution DEM. *ISPRS J. Photogramm. Remote Sens.* **2017**, *134*, 135–147. [CrossRef]
9. Karkee, M.; Steward, B.L.; Aziz, S.A. Improving quality of public domain digital elevation models through data fusion. *Biosyst. Eng.* **2008**, *101*, 293–305. [CrossRef]
10. Mukherjee, S.; Joshi, P.K.; Mukherjee, S.; Ghosh, A.; Garg, R.D.; Mukhopadhyay, A. Evaluation of vertical accuracy of open source Digital Elevation Model; (DEM). *Int. J. Appl. Earth Obs. Geoinf.* **2013**, *21*, 205–217. [CrossRef]
11. Yamazaki, D.; Ikeshima, D.; Tawatari, R.; Yamaguchi, T.; O'Loughlin, F.; Neal, J.C.; Sampson, C.C.; Kanae, S.; Bates, P.D. A high-accuracy map of global terrain elevations. *Geophys. Res. Lett.* **2017**, *44*, 5844–5853. [CrossRef]
12. Meng, X. SRTM DEM and its application advances. *Int. J. Remote Sens.* **2011**, *32*, 3875–3896.
13. Zhang, G.; Xie, H.; Kang, S.; Yi, D.; Ackley, S.F. Monitoring lake level changes on the Tibetan Plateau using ICESat altimetry data (2003–2009). *Remote Sens. Environ.* **2011**, *115*, 1733–1742. [CrossRef]
14. Toutin, T. ASTER DEMs for geomatic and geoscientific applications: A review. *Int. J. Remote Sens.* **2008**, *29*, 1855–1875. [CrossRef]
15. Krieger, G.; Moreira, A.; Fiedler, H.; Hajnsek, I.; Werner, M.; Younis, M.; Zink, M. TanDEM-X: A satellite formation for high-resolution SAR interferometry. *IEEE Trans. Geosci. Remote Sens.* **2007**, *45*, 3317–3341. [CrossRef]
16. Farr, T.G.; Rosen, P.A.; Caro, E.; Crippen, R.; Duren, R.; Hensley, S.; Kobrick, M.; Paller, M.; Rodriguez, E.; Roth, L.; et al. The Shuttle Radar Topography Mission. *Rev. Geophys.* **2007**, *45*, 361. [CrossRef]
17. Blomgren, S. A digital elevation model for estimating flooding scenarios at the Falsterbo Peninsula. *Environ. Model. Softw.* **1999**, *14*, 579–587. [CrossRef]
18. Chen, C.; Yue, T. A method of DEM construction and related error analysis. *Comput. Geosci.* **2010**, *36*, 717–725. [CrossRef]
19. Robinson, T.P.; Metternicht, G. Testing the performance of spatial interpolation techniques for mapping soil properties. *Comput. Electron. Agric.* **2006**, *50*, 97–108. [CrossRef]
20. Reuter, H.I.; Nelson, A.; Jarvis, A. An evaluation of void-filling interpolation methods for SRTM data. *Int. J. Geogr. Inf. Sci.* **2007**, *21*, 983–1008. [CrossRef]
21. Grohman, G.; Kroenung, G.; Strebeck, J. Filling SRTM Voids: The Delta Surface Fill Method. *Photogramm. Eng. Remote Sens.* **2006**, *72*, 213–216.
22. Luedeling, E.; Siebert, S.; Buerkert, A. Filling the voids in the SRTM elevation model—A TIN-based delta surface approach. *ISPRS J. Photogramm. Remote Sens.* **2007**, *62*, 283–294. [CrossRef]
23. Yue, T.-X.; Chen, C.-F.; Li, B.-L. A high-accuracy method for filling voids and its verification. *Int. J. Remote Sens.* **2012**, *33*, 2815–2830. [CrossRef]
24. Zhu, D.; Cheng, X.; Zhang, F.; Yao, X.; Gao, Y.; Liu, Y. Spatial interpolation using conditional generative adversarial neural networks. *Int. J. Geogr. Inf. Sci.* **2019**. [CrossRef]

25. Dong, G.; Chen, F.; Ren, P. Filling SRTM Void Data Via Conditional Adversarial Networks. In Proceedings of the International Geoscience and Remote Sensing Symposium (IGARSS), Valencia, Spain, 22–27 July 2018; pp. 7441–7443.
26. Gavriil, K.; Muntingh, G.; Barrowclough, O.J.D. Void Filling of Digital Elevation Models with Deep Generative Models. *IEEE Geosci. Remote Sens. Lett.* **2019**. [CrossRef]
27. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Bing, X.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the International Conference on Neural Information Processing Systems, Kuching, Malaysia, 3–6 November 2014.
28. Kingma, D.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference for Learning Representations, San Diego, CA, USA, 7–9 May 2015.
29. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
30. Wang, T.C.; Liu, M.Y.; Zhu, J.Y.; Tao, A.; Catanzaro, B. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 8798–8807.
31. Liu, M.Y.; Breuel, T.; Kautz, J. Unsupervised Image-to-Image Translation Networks. *arXiv* **2017**, arXiv:1703.00848. Available online: https://arxiv.org/abs/1703.00848 (accessed on 2 March 2017).
32. Guo, J.; Lu, S.; Han, C.; Zhang, W.; Wang, J. Long Text Generation via Adversarial Training with Leaked Information. *arXiv* **2017**, arXiv:1709.08624. Available online: https://arxiv.org/abs/1709.08624 (accessed on 24 September 2017).
33. Iizuka, S.; Simo-Serra, E.; Ishikawa, H. Globally and locally consistent image completion. *ACM Trans. Graph.* **2017**, *36*, 107. [CrossRef]
34. Yeh, R.A.; Chen, C.; Lim, T.Y.; Schwing, A.G.; Hasegawa-Johnson, M.; Do, M.N. Semantic Image Inpainting with Deep Generative Models. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 22–25 July 2017; pp. 5485–5493.
35. Radford, A.; Metz, L.; Chintala, S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv* **2015**, arXiv:1511.06434. Available online: https://arxiv.org/abs/1511.06434 (accessed on 19 November 2015).
36. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein GAN. *arXiv* **2017**, arXiv:1701.07875. Available online: https://arxiv.org/abs/1701.07875 (accessed on 26 Janurary 2017) .
37. Mao, X.; Li, Q.; Xie, H.; Lau, R.Y.K.; Zhen, W.; Smolley, S.P. Least Squares Generative Adversarial Networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2794–2802.
38. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv* **2015**, arXiv:1502.03167. Available online: https://arxiv.org/abs/1502.03167 (accessed on 11 February 2015).
39. Nair, V.; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the International Conference on International Conference on Machine Learning (ICML), Haifa, Israel, 21–24 June 2010; pp. 807–814.
40. He, K.; Zhang, X.; Ren, S.; Jian, S. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 13–16 December 2015; pp. 1026–1034.
41. Sajjadi, M.S.M.; Schölkopf, B.; Hirsch, M. EnhanceNet: Single Image Super-Resolution through Automated Texture Synthesis. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 4491–4500.
42. Rez, P.; Gangnet, M.; Blake, A. Poisson image editing. *ACM Trans. Graph.* **2003**, *22*, 313–318.
43. Horn, B.K.P. Hill shading and the reflectance map. *Proc. IEEE* **1981**, *69*, 14–47. [CrossRef]
44. Zhou, W.; Alan Conrad, B.; Hamid Rahim, S.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans Image Process* **2004**, *13*, 600–612.
45. Ma, L.; Li, Y. Evaluation of SRTM DEM over China. In Proceedings of the IEEE International Conference on Geoscience & Remote Sensing Symposium (IGARSS), Denver, CO, USA, 31 July–4 August 2006.

46. Hancock, G.R.; Martinez, C.; Evans, K.G.; Moliere, D.R. A comparison of SRTM and high-resolution digital elevation models and their use in catchment geomorphology and hydrology: Australian examples. *Earth Surf. Process. Landf.* **2010**, *31*, 1394–1412. [CrossRef]
47. Robinson, N.; Regetz, J.; Guralnick, R.P. EarthEnv-DEM90: A nearly-global, void-free, multi-scale smoothed, 90 m digital elevation model from fused ASTER and SRTM data. *ISPRS J. Photogramm. Remote Sens.* **2014**, *87*, 57–67. [CrossRef]
48. Jhee, H.; Cho, H.-C.; Kahng, H.-K.; Cheung, S. Multiscale quadtree model fusion with super-resolution for blocky artefact removal. *Remote Sens. Lett.* **2013**, *4*, 325–334. [CrossRef]