GI_Forum 2019, Issue 1
Page: 119 - 133
Full Paper
Corresponding Author:
lukas.beer@campus.tu-berlin.de
DOI: 10.1553/giscience2019_01_s119

Automatic Generation Of LoD1 City Models And Building Segmentation From Single Aerial Orthographic Images Using Conditional Generative Adversarial Networks

Lukas Beer
TU-Berlin, Germany

Abstract

3D city models play an important role in multiple applications, but creating them still requires effort using various possible techniques. This paper proposes a new machine-learning-based framework for generating 3D city models. With the help of conditional Generative Adversarial Networks and single orthographic images, segmentation and height estimations of buildings are achieved. The height information per pixel and the building coordinates were generalized using a histogram for heights and the Douglas-Peucker algorithm. The framework was evaluated by using variations of the same dataset (for the city of Berlin) to show possible differences due to changes in the image size and representation of the heights. The evaluation reveals that it is possible to generate block models with a mean absolute height error of 5.53m per building, a mean absolute height error for the whole raster of 1.32m, and a Jaccard Index of 0.55 for the segmentation. While the proposed framework for generating LoD1 city models does not attain the accuracy of previous techniques, our work represents a step towards successfully using machine learning for the automatic generation of city models and building segmentation.

Keywords:

city models, generative adversarial networks, LoD1, segmentation

1 Introduction

The use of 3D city models is widespread, with applications in areas that include tourism, disaster management, urban environmental management and the real-estate industry (Singh, Jain & Mandla, 2013). Depending on the application, the models can be created with different levels of detail (LoDs), as shown in Figure 1. LoD1 represents a simple block model of a building without roof shapes; buildings with higher LoDs are more detailed. To create them, there are two principal techniques: photogrammetric 3D reconstruction using stereo imagery, and LiDAR-based laser scanning (Haala & Kada, 2010). Due to the lack of freely available stereo imagery or (LiDAR-based) point clouds of cities, new measurements are often required. Accordingly, it would be an advantage to have open-source data from which height information can be extracted. Nowadays, aerial orthographic images are available online; however, extracting height information from them is still a non-trivial task.

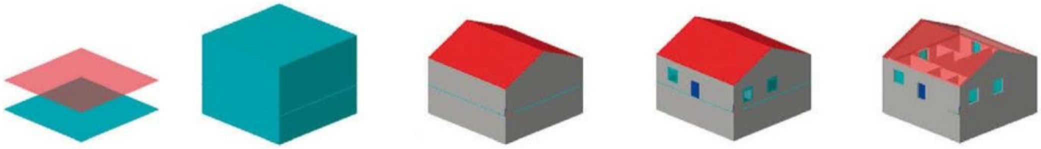


Figure 1: From left to right: LoD0 – LoD4 (Coors, Andrae & Böhm, 2016, p. 70)

This is where machine-learning-based architectures can play an important role. In recent studies, various neural network architectures have been used to tackle the problem of obtaining 3D information from monocular images. The architectures used for solving this problem cover convolutional neural networks (Hu, Ozay, Zhang & Okatani, 2019), multi-scale neural networks (Eigen, Puhrsch & Fergus, 2014), and linear regression (Saxena, Chung,& Ng, 2006). Mou and Zhu (2018) evaluated their fully residual convolutional-deconvolutional network on digital surface models (DSM). Goodfellow et al. (2014) introduced a new algorithm: the Generative Adversarial Network (GAN). With large numbers of target images y_{real}, the GAN learns during the training procedure to map from a random noise vector z to an output image y_{fake}:

$$G: \{z\} \rightarrow y_{fake} \tag{1}$$

GANs are based on two neural networks: a generator G and a discriminator D. While G learns to produce outcomes that are as realistic as possible, D learns to distinguish between the outcomes of G and the real data feeding the information back into G. This zero-sum game was originally introduced for creating artificial images, e.g. handwritten numbers or faces which should be indistinguishable from real data (Goodfellow et al., 2014). Since then, many new GANs with different purposes, additions and improvements have been introduced, such as MLGANs (Metric Learning-based GANs; Dou, 2017), 3D-ED GANs (3D-Encoder-Decoder GANs; Wang, Huang, You, Yang & Neumann, 2017), DCGANs (Deep Convolutional GANs; Radford, Metz & Chintala, 2015), and FC-GANs (Fast-converging Conditional GANs; Li, Wang & Qi, 2018), to name but a few. But for the purpose of the present paper, the use of conditional GANs (cGANs) and their improvements by Isola, Zhu, Zhou and Efros (2017) seems suitable. This so-called 'Pix2Pix GAN' is able to map from a known image x and a random vector z to an output image y_{fake}:

$$G: \{x, z\} \rightarrow y_{fake} \quad (2)$$

Extending this idea, it is also possible to map from an image to a 2.5D raster: every pixel of the outcome represents a height. Something similar has already been done by Ghamisi and Yokoya (2018). In their case, the Pix2Pix GAN learned to estimate a DSM using near-infrared orthographic images. Their work and the present framework have in common the base technique of Isola et al. (2017), but instead of estimating the height of the whole surface, the present paper focuses on performing a building segmentation, a height estimation, and finally the generation of a city model with LoD1. Thus, it is no longer necessary, subsequently, to process whole point clouds in order to obtain city models. Furthermore, the present framework is based on aerial orthographic images, which are easily accessible. In order to create block models, generalized heights are required, as are the conversion of raster data into vector data, and extraction of the corner points. The extraction can be done using any of several algorithms, including the Harris corner detector, the SUSAN corner detector, the Moravec corner detection algorithm (as stated in Patel and Panchal (2014)), or the Douglas-Peucker algorithm (Douglas & Peucker, 1973). In order to achieve interoperability, one of the most common ways of storing city models is to use the City Geography Markup Language (CityGML), as proposed by Kolbe, Gröger and Plümer (2005). The segmentation can be achieved by defining a threshold in the heights.

2 Methods

The present approach for LoD1-generation and building segmentation consists of several steps:

- creating the dataset and training the network to map from the orthographic city images to the 2.5D raster
- generalizing the resulting outcome and transforming the buildings into (CityGML-based) vector data
- using the generalized outcome for the segmentation.

2.1 Network architecture

Isola et al. (2017) came up with a framework that made image-to-image translation simpler, in that this framework is no longer application-specific and consequently can be used for many different tasks. It is therefore suitable for using in our own framework as well. Just like the original GAN (Goodfellow et al., 2014), the basic concept consists of a generator G and a discriminator D. The conditional GAN can map from a random vector z together with an input image x to an output image y_{fake} (2).

The main objective in the Pix2Pix GAN (and thus in the present work) is:

$$\mathrm{G}^* = \arg \underset{min}{G}\ \underset{max}{D}\ \mathcal{L}_{\mathrm{cGAN}}(G, D) + \lambda\ \cdot \mathcal{L}_{\mathrm{L1}}(G) \tag{3}$$

with:

$$\mathcal{L}_{cGAN}\ = \mathbb{E}_{x,y_{fake}}[log(\ D(x, y_{fake}))]\ +\ \mathbb{E}_{x,z}[log\ (1 - D(x, G(x, z)))] \tag{4}$$

and:

$$\mathcal{L}_{L1}(\mathrm{G})\ = \mathbb{E}_{x,y_{fake},z}[||y_{fake}\ -\ G(x, z)\ ||] \tag{5}$$

$\mathcal{L}_{L1}$ and $\mathcal{L}_{cGAN}$ are two different loss functions. Their impacts on the result can be adjusted by altering λ. The functions consist of the expectations $\mathbb{E}_{x,y_{fake}}$, $\mathbb{E}_{x,z}$ and $\mathbb{E}_{x,y_{fake},z}$. The exclusive use of $\mathcal{L}_{cGAN}$ would produce sharp images with more false positives, while the exclusive use of $\mathcal{L}_{L1}$ would produce blurry images with fewer false positives (Ghamisi & Yokoya, 2018; Isola et al., 2017). It should be emphasized that the present framework does not include the random vector *z*. Isola et al. (2017) have shown that omitting z does not result in a decreased quality of the outcome, because G simply learns to ignore the noise vector. This is congruent with Mathieu, Couprie and LeCun (2015).

D tries to maximize the objective and G tries to minimize it. In addition, the structure of D and G is an outstanding part of the Pix2Pix GAN. G consists of an encoder-decoder network, as used in earlier solutions (Pathak, Krähenbühl, Donahue, Darrell & Efros, 2016; Wang & Gupta, 2016; Zhou & Berg, 2016), but additionally there are skip connections between layers. By concatenating different layers, this addition aims to shuttle information across the network in order to enhance the sharing of low-level information. Another improvement of the Pix2Pix framework is D, which is based on a so-called 'PatchGAN': per image, this architecture evaluates $N \times N$ patches, and decides per patch whether it is real or fake. These probabilities are stored in the final layer. Figures 2 and 3 show a more detailed description of G and D from the Pix2Pix GAN (Isola et al., 2017) which is used in the framework that we propose here. The processes between the layers are represented as coloured arrows. In what follows, the elements of the network will be explained briefly. For more detailed understanding, the reader is referred to Isola et al. (2017).

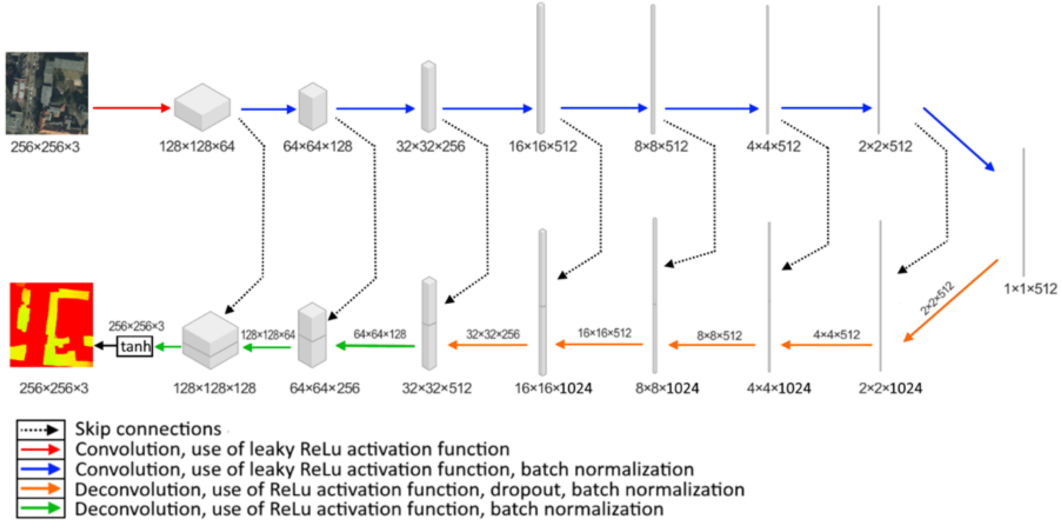

256×256×3
128×128×64
64×64×128
32×32×256
16×16×512
8×8×512
4×4×512
2×2×512
1×1×512
2×2×512
4×4×512
8×8×512
16×16×512
32×32×256
64×64×128
128×128×64
256×256×3
tanh
256×256×3
128×128×128
64×64×256
32×32×512
16×16×1024
8×8×1024
4×4×1024
2×2×1024
Skip connections
Convolution, use of leaky ReLu activation function
Convolution, use of leaky ReLu activation function, batch normalization
Deconvolution, use of ReLu activation function, dropout, batch normalization
Deconvolution, use of ReLu activation function, batch normalization

Figure 2: Generator architecture

Adding dropout means that each layer has a predefined probability (dropout rate) of not being considered for a training iteration (Srivastava, Hinton, Krizhevsky, Sutskever & Salakhutdinov, 2014). Additionally, batch normalization (Ioffe & Szegedy, 2015) regularizes and normalizes the network. This and the dropout prevent the network from overfitting (Ioffe & Szegedy, 2015). A convolution with stride *n* means that the filter kernel ignores each layer with a stepsize of *n*.

ReLU and leaky ReLU activation functions (Maas, Hannun & Ng, 2013) are used. For the leaky ReLU, the following function is applied to each element of the layers:

$$f(x) = max\{\alpha \cdot x,\ x\} \tag{6}$$

If the ReLU is not leaky, the slope α is set to zero. In order to train the network, a minibatch stochastic gradient descent is used, and the Adam solver (Kingma & Ba, 2015) is applied in order to optimize the network.

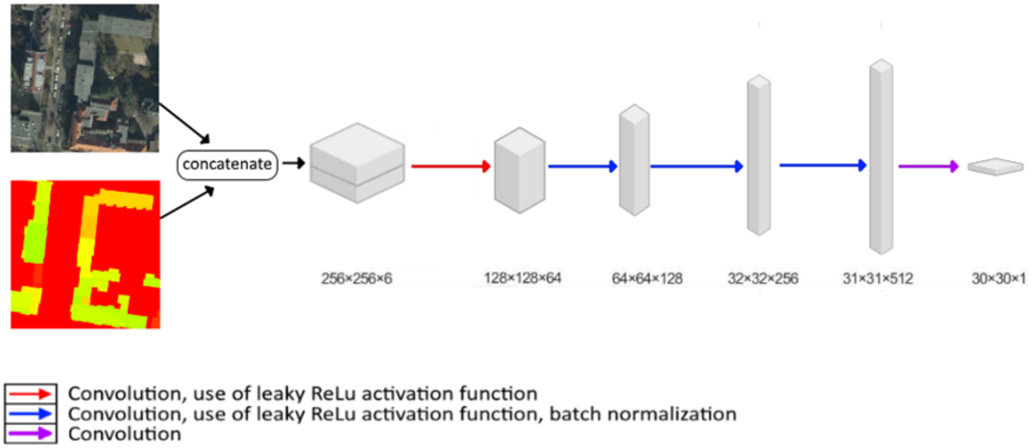

concatenate
256×256×6
128×128×64
64×64×128
32×32×256
31×31×512
30×30×1
Convolution, use of leaky ReLu activation function
Convolution, use of leaky ReLu activation function, batch normalization
Convolution

Figure 3: Discriminator architecture

2.2 Training details

For training, a batch size of 1, an initial learning rate of 0.0002, and λ = 100 were used. With λ=100, the network encourages both the sharpness from $\mathcal{L}_{cGAN}$ and the correctness (with fewer false positives) from $\mathcal{L}_{L1}$. The initial learning rate adjusts by how much the weights need to be updated, while the batch size refers to the number of samples which are used in one training iteration. To control the exponential decay rates of the Adam solver, the momentum parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$ were used. The dropout rate was set to 0.5; the slope α was set to 0.2. In addition to the experience gained during training tests, the hyperparameters chosen for the present framework adhere to Isola et al. (2017), Ghamisi and Yokoya (2018), and Shi, Li and Zhu (2019). The network was trained for 19 hours per set (as defined in section 2.3), using an nVidia GTX1060 with 6GB of VRAM. The present framework was implemented in Python.

2.3 Dataset

For many machine-learning-based applications, a lot of training data is required, especially when it comes to complex problems. Due to the specific aim of this paper, a new dataset was created using freely available data from the City of Berlin. The Senate for Urban Development and Housing of Berlin (2019) provides digital orthographic images with a 20cm ground resolution and a 3D city model with LoD1. In their city model, the building footprints are extracted from the cadastre; the heights are calculated from a DSM with a 5m ground resolution. In order for the height information inside the Pix2Pix GAN to be exploitable, the orthographic images and the city model had to be matched. Therefore, the height and positions were transferred from vector into raster data. In total, 1,064km^2 could have been used for the present approach. However, an area of 4km^2 was excluded from the dataset for visualizing the framework results of a complete district afterwards. The training was done using three different representations of the same training data, called 'sets'. Images were removed from these sets if they did not contain any height information. The individual sets are described in detail in the following sections. Because of the network architecture, the input images measure 256pixels × 256pixels, or 512pixels × 512pixels. This means that the strided convolutions result in bisecting the input size, until the last layer is reached.

Set 1

Each pixel directly stores height information in the pixel intensity, resulting in a greyscale image. Due to this conversion, the height information became discrete. The 256pixel × 256pixel resolution of the images represents 51.2m × 51.2m on the ground. In total, 152,393 images were created, of which 137,154 were used for training, 7,619 for testing during the training, and 7,620 for the evaluation.

Set 2

As in set 1, each pixel contains height information directly in its intensity, but compared to set 1, the resolution differs: the input images measure 512pixels × 512pixels, which represent 102.4m × 102.4m on the ground. 46,383 images were created in total, of which 41,742 were used for training, 2,318 for testing during the training, and 2,319 for the evaluation.

Set 3

Instead of representing the height *h* directly in the pixel intensity, an extended conversion into the HSV (hue, saturation, value) colour space was used. The lightness and the saturation were set to 100%, and the height was converted into hue. The RGB (red, green, blue) values can be calculated, as e.g. in Kaur and Banga (2013). Because of larger differences in the colour space for small height differences and the small number of buildings with a height of over 100m in Berlin (compared to the total dataset), the maximum height was set to 100m:

$$h = \begin{cases} h, & h < 100 \\ 100, & h \geq 100 \end{cases} \tag{7}$$

And then:

$$hue = h \cdot 3.6 \tag{8}$$

Thus, a height of 0m would represent [255,0,0] in the RGB colour space, while a height of 33.3m would result in [0,255,0]. As in set 2, the resolution of the dataset is 512pixels × 512pixels. Therefore, 46,383 images were created, of which 41,742 were used for training, 2,318 for testing during the training, and 2,319 for the evaluation.

The two ways of representing height (height → intensity, and height → hue) can be seen in Figure 4.

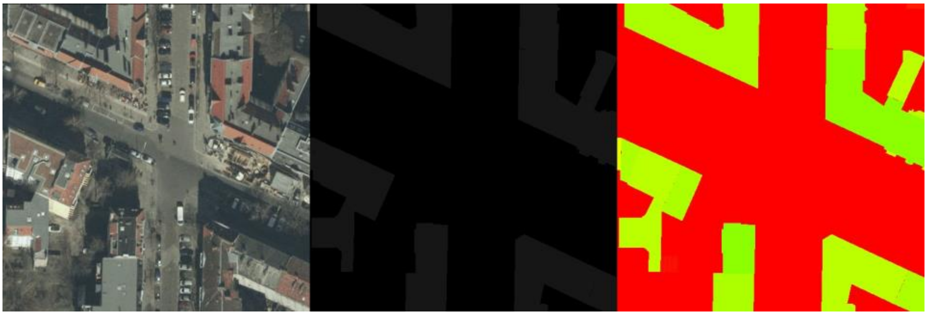


Figure 4: From left to right: Orthographic image, greyscale representation and hue representation of the height

2.4 Generalization of the outcome

After training has been completed, what the generator learned can be applied: to map from an input image to a 2.5D raster. The next task was to process the height information in order to obtain generalized buildings. First, the height information from the raster had to be extracted. Thanks to the direct conversion of height into intensity, the height information can be extracted directly from sets 1 and 2. Set 3 was transformed from the RGB colour space back into the HSV colour space, and hue was used for the height.

In order to enhance the accuracy of the result, two morphological filters were applied. First, a morphological opening was used in order to erase small image artefacts which were not part of any building. In other words, a threshold for a minimum building size was created. Second, a morphological closing was used to fill in small holes in the border or inside a polygon (Maragos, 1987). Thus, the shape had a continuous border. The kernel measured 5pixels × 5pixels, which represents 1m × 1m for set 1, and 2m × 2m for sets 2 and 3.

By extracting the contours of the resulting polygons, a 2.5D representation of the building would already have been possible, but with one coordinate per pixel on the boundary, there would have been too many coordinates. Therefore, only the most important vertices for each boundary were extracted. For some cases, a convex hull algorithm would have been sufficient, but due to the presence of some concave-shaped buildings, this generalization would not have fitted all our needs. Therefore, the Douglas-Peucker algorithm was used, which is a simple but effective recursive algorithm (van Kreveld, Löffler & Wiratma, 2018). In our case, the algorithm adds vertices from the building contours to a list of points, as long as the computed line-segment between two points lies within a specific distance of a vertex. This list of points represents the most important points of the buildings. If the global coordinates of one pixel and the ground resolution are given, the transformation from image to world coordinates can be made for every pixel. Subsequently, just one height per building was needed to generate LoD1 city models. This single height was calculated with the help of a histogram of all heights inside one building polygon: the height which occurs most often was chosen for the whole building. For simplification reasons, only one height per building was used, even if a building consisted of several parts with different heights. The conversion into CityGML was done automatically by storing the vertices and the height in the correct format, as in Kolbe et al. (2005).

2.5 Segmentation

Since the present framework was intended to estimate a height only if there was a building, the segmentation was achieved by applying a threshold to the generalization: if the estimated height per pixel was greater than 0, it was classified as 'building'. A minimum number of pixels per building is not necessary, due to the morphological opening, as described in section 2.4.

3 Evaluation metrics

In order to evaluate the results of the proposed framework with the present dataset, the mean absolute error (MAE) and the root mean squared error (RMSE) were calculated 3 times per set: once for the pure raster, which is the direct result of the Pix2Pix GAN (referred to below as error type 1); once for the whole raster after generalizing the buildings (referred to as error type 2), and once for the estimated building heights only (referred to as error type 3).

In order to evaluate the segmentation results, the Jaccard-Index (JI; also called Intersection over Union) was calculated for each of the 3 sets, for error types 1 and 2.

3.1 Estimating the overall height error

The MAE and RMSE were calculated once for the unprocessed outcome of the Pix2Pix GAN and once for the generalized building heights. To compute the overall error, every single pixel was used:

$$\forall h_{real}, h_{fake}, \qquad MAE = \frac{1}{n}\sum |h_{real} - h_{fake}| \tag{9}$$

$$\forall h_{real}, h_{fake}, \qquad RMSE = \sqrt{\frac{1}{n}\sum(h_{real} - h_{fake})^2} \tag{10}$$

3.2 Estimating the building height error

The same evaluation metrics were used again, but only for those pixels for which the MAE and RMSE had been calculated, where a building was estimated by our framework or really existed. Thus, areas without buildings are ignored:

$$\forall h_{real}, h_{fake} \mid h_{real} + h_{fake} > 0 \qquad MAE = \frac{1}{n}\sum |h_{real} - h_{fake}| \tag{11}$$

$$\forall h_{real}, h_{fake} \mid h_{real} + h_{fake} > 0 \qquad RMSE = \sqrt{\frac{1}{n}\sum (h_{real} - h_{fake})^2} \tag{12}$$

3.3 Estimating the segmentation accuracy

The JI values are between 0 and 1, where 1 represents identical images and therefore perfect segmentation. The JI is given by:

$$JI = \frac{y_{real} \cap y_{fake}}{y_{real} \cup y_{fake}} \tag{13}$$

4 Results

In Table 1, the accuracies for the different sets and types of error can be seen. Independent of the type, it shows that set 3 has the lowest accuracy levels, followed by set 2. The training using set 1 reached the highest accuracy levels. The results show that the image size of the training data seems important: even though sets 1 and 2 represent height in the same way, set 1 has 20–35% more errors than set 2. This high error difference might be due to it possibly being harder to reach conclusions because of the smaller surface area. This could be reinforced by the architecture of the discriminator: the patches, which D tries to distinguish, have little information about the structural components of buildings. The results show that converting the height into the HSV colour space (set 3) might lead to greater accuracy as well: even though the error differences between sets 2 and 3 are small, set 3 has smaller errors in each of the three types.

Table 1: Errors and JI for different sets and different error types

	Type 1			Type 2			Type 3		
	Set 1	Set 2	Set 3	Set 1	Set 2	Set 3	Set 1	Set 2	Set 3
MAE [m]	1.98	1.42	1.35	1.97	1.46	1.32	6.66	5.83	5.53
RMSE [m]	3.66	3.04	2.94	3.65	3.08	2.96	6.72	6.20	6.02
JI	0.54	0.55	0.55	0.54	0.55	0.55	/	/	/

Thus, the Douglas-Peucker algorithm in combination with a height generalization was not responsible for the decrease in accuracy, no matter which dataset was used. With regards to the problems which might be causing the error, a closer look at the images shows that high errors occurred mainly at the borders of a building. Figure 5d shows the absolute difference between the image actually produced (Figure 5b) and the target image (Figure 5c) of set 3, and explains this behaviour: blurred and unclear edges and boundaries in the generated image result in high errors at the boundaries (shown in yellow in Figure 5d). In addition, false negatives as well as false positives influence the error. Figure 5a shows the orthographic input image.

If we look at the results of the segmentation, we find the best results in sets 2 and 3, with almost no differences between the sets. Within the error types, there were no differences for the JI.

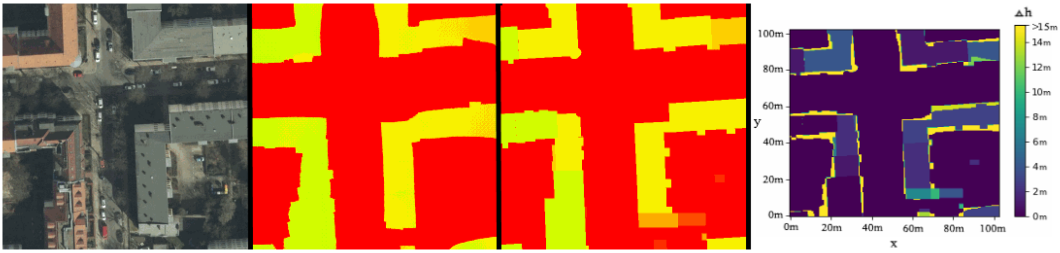
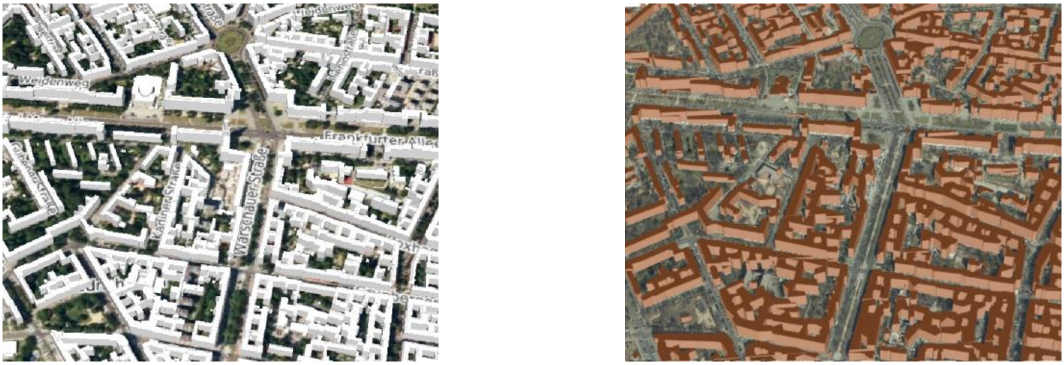


Figure 5: Left to right: (a) orthographic image, (b) produced image, (c) target image, (d) absolute difference between produced and target image

The results in height and segmentation are congruent with recent studies. Ghamisi and Yokoya (2018) postulate an RMSE of 2.56 – 3.89m over the whole DSM for various cities, also using Pix2Pix GANs. Due to the lack of studies in this field comparable to our own, further comparison with other metrics is not discussed here. Nevertheless, the segmentation with the training data is comparable to, and even exceeds, recent results for building segmentation using cGANs: Shi et al. (2019) postulate a JI of 0.52. However, compared to other architectures, there is still room for considerable improvement: Bischke, Helber, Folz, Borth and Dengel (2017) report a JI of 0.70 when using a SegNet, while the multi-constraint fully-convolutional network of Wu et al. (2018) resulted in a mean JI of 0.83. In their work, U-Nets reached a score of 0.81 and fully-convolutional networks achieved a JI of 0.52. Using HOG-ADA resulted in a JI of 0.31 (Wu et al., 2018).

5 Empirical evaluation of larger areas of cities

For an empirical evaluation, one entire district of Berlin (Friedrichshain) was removed before training and processed completely afterwards. In order to process the whole district, it was divided into 104.2m × 104.2m tiles. The result of the LoD1 generation can be seen in Figure 6b, while the segmentation result can be found in Figure 7. To allow the quality of the present framework to be judged better, Figure 6a shows a conventional 3D city model.



(a) conventional 3D city model (virtualcitySYSTEMS, 2017)

(b) present framework

Figure 6: 3D city models using underlying orthographic images

The outcome of the LoD1 generation (Figure 6b) looks realistic at first sight, with few artefacts. As well as looking at errors due to false positives and false negatives, the network analysed each 104.2m × 104.2m tile individually. These tiles were concatenated later. Therefore, the heights at the tile borders do have small differences. Compared to the conventional LoD1 in Figure 6a, the artificially generated city model in Figure 6b seems to be less accurate: borders and connections between buildings are drawn more roughly. The JI is an indicator for that, too. Furthermore, the heights of atypical buildings like towers were not estimated correctly. Nevertheless, distinguishing between the conventional and the generated city models is difficult at first glance. As shown in Figure 7, most of the buildings are located correctly. It can thus be concluded that the segmentation was quite often successful, even though some buildings were misinterpreted: atypical buildings with a unique architecture, such as museums, might cause problems.

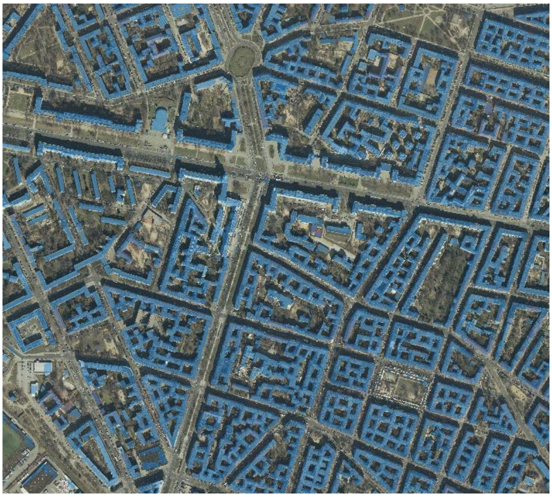


Figure 7: Segmented buildings using underlying orthographic image

6 Conclusion

A new technique for automatic city model generation has been presented. It shows one of the possibilities that the improvements of machine-learning-based architectures can contribute to the field of geoinformation science. With this new framework, it is possible using aerial orthographic images to create acceptably accurate city models from single images, and to realize building segmentation. It should, however, be emphasized that modifying the height representation alters the accuracy of the height estimation. Other methods that use different colourmaps and add multiple channels for the height representation might reduce the errors even more and should be investigated in the future. Testing and evaluating our method on orthographic images of other cities should be considered, as should using datasets with different LoDs. Even though there is considerable room for improving the accuracy of the height estimation and the segmentation, this work represents the first step in a new direction for meeting the need for city models.

References

Bischke, B., Helber, P., Folz, J., Borth, D., & Dengel, A. (2017). Multi-Task Learning for Segmentation of Building Footprints with Deep Neural Networks. *CoRR, abs/1709.05932.* Retrieved from http://arxiv.org/abs/1709.05932

Coors, V., Andrae, C., & Böhm, K.-H. (2016). *3D-Stadtmodelle - Konzepte und Anwendungen mit CityGML.* Berlin, Offenbach: Vde Verlag GmbH.

Dou, Z. (2017). Metric learning-based generative adversarial network. *CoRR, abs/1711.02792.* Retrieved from http://arxiv.org/abs/1711.02792

Douglas, D. H., & Peucker, T. K. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, *10*(2), 112–122. doi:10.3138/FM57-6770-U75U-7727

Eigen, D., Puhrsch, C., & Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 2366-2374. Retrieved from http://papers.nips.cc/paper/5539-depth-map-prediction-from-a-single-image-using-a-multi-scale-deep-network.pdf

Ghamisi, P., & Yokoya, N. (2018). IMG2dsm: Height simulation from single imagery using conditional generative adversarial net. *IEEE Geoscience and Remote Sensing Letters*, *15*(5), 794–798. doi:10.1109/lgrs.2018.2806945

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems,* 2672 - 2680. Retrieved from http://papers.nips.cc/paper/5423-generative-adversarial-nets

Haala, N., & Kada, M. (2010). An update on automatic 3d building reconstruction. *ISPRS Journal of Photogrammetry and Remote Sensing*, *65*(6), 570–580. doi:10.1016/j.isprsjprs.2010.09.006

Hu, J., Ozay, M., Zhang, Y., & Okatani, T. (2019). Revisiting single image depth estimation: toward higher resolution maps with accurate object boundaries. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1043-1051. Retrieved from https://arxiv.org/abs/1803.08673

Ioffe S. and Szegedy C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML.* Retrieved from https://arxiv.org/pdf/1502.03167.pdf

Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1125-1134. Retrieved from http://arxiv.org/abs/1611.07004

Kaur, S., & Banga, D. V. K. (2013). Content based image retrieval: Survey and comparison between rgb and hsv model. *International Journal of Engineering Trends and Technology*, *4/2013*. Retrieved from http://ijettjournal.org/volume-4/issue-4/IJETTV4I4P215.pdf

Kingma, D., & Ba, J. (2015). Adam: A method for stochastic optimization. *ICLR*. Retrieved from https://arxiv.org/abs/1412.6980

Kolbe, T., Gröger, G., & Plümer, L. (2005). CityGML - interoperable access to 3d city models. *Geo-information for Disaster Management*. doi:10.1007/3-540-27468-5_63

Li, C., Wang, Z., & Qi, H. (2018). Fast-converging conditional generative adversarial networks for image synthesis. *IEEE International Conference on Image Processing (ICIP),* 2132-2136. doi:10.1109/ICIP.2018.8451161

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In ICML Workshop on Deep Learning for Audio, Speech, and Language Processing.

Maragos, P. (1987). Tutorial on advances in morphological image processing and analysis. *Optical Engineering*, *26*(7), 623 – 632. doi:10.1117/12.7974127

Mathieu, M., Couprie, C., & LeCun, Y. (2015). Deep multi-scale video prediction beyond mean square error. *International Conference on Learning Representations (ICLR)*, Retrieved from: ttps://arxiv.org/pdf/1511.05440.pdf

Mou, L., & Zhu, X. X. (2018). Im2height: Height estimation from single monocular imagery via fully residual convolutional-deconvolutional network. *CoRR, abs/1802.10249* Retrieved from http://arxiv.org/abs/1802.10249

Patel, T. P., & Panchal, S. R. (2014). Corner detection techniques: An introductory survey. *International Journal of Engineering Development and Research*, *2*. Retrieved from https://www.ijedr.org/papers/IJEDR1404047.pdf

Pathak, D., Krähenbühl, P., Donahue, J., Darrell, T., & Efros, A. A. (2016). Context encoders: Feature learning by inpainting. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2536–2544. doi:10.1109/CVPR.2016.278

Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *CorRR, abs/1511.06434*. Retrieved from http://arxiv.org/abs/1511.06434

Saxena, A., Chung, S. H., & Ng, A. Y. (2006). Learning depth from single monocular images. *Advances in neural information processing systems,* 1161-1168. Retrieved from http://papers.nips.cc/paper/2921-learning-depth-fromsingle-monocular-images.pdf

Senate for Urban Development and Housing of Berlin. (2019). Geoportal Berlin. *berlin.de*. Retrieved January, 30, 2019 from https://www.stadtentwicklung.berlin.de/ geoinformation/

Shi, Y., Li, Q., & Zhu, X. X. (2019). Building Footprint Generation Using Improved Generative Adversarial Networks. *IEEE Geoscience and Remote Sensing Letters*, 16(4), 603-607. Retrieved from http://arxiv.org/abs/1810.11224

Singh, S. P., Jain, K., & Mandla, V. R. (2013). VIRTUAL 3D CITY MODELING: TECHNIQUES AND APPLICATIONS. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XL-2/W2*, 73–91. doi:10.5194/isprsarchives-xl-2-w2-73-2013

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), 1929-1958. Retrieved from http://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf?utm_content=buffer79b43&utm_medium=social&utm_source=twitter.com&utm_campaign=buffer

van Kreveld, M., Löffler, M., & Wiratma, L. (2018). On optimal polyline simplification using the hausdorff and fréchet distance. *arXiv preprint arXiv:1803.03550*. Retrieved from http://arxiv.org/abs/1803.03550

virtualcitySYSTEMS. (2017). Virtualcitymap - 3d-stadtmodelle im browser. *virtualcitysystems.de* Retrieved January, 30, 2019 from https://berlin.virtualcitymap.de/#/

Wang, W., Huang, Q., You, S., Yang, C., & Neumann, U. (2017). Shape inpainting using 3d generative adversarial network and recurrent convolutional networks. *Proceedings of the IEEE International Conference on Computer Vision*, 2298-2306. doi: 10.1109/ICCV.2017.252

Wang, X., & Gupta, A. (2016). Generative image modeling using style and structure adversarial networks. *European Conference on Computer Vision*, 318-335. doi:10.1007/978-3-319-46493-0_20

Wu, G., Shao, X., Guo, Z., Chen, Q., Yuan, W., Shi, X., Xu, Y., & Shibasaki, R. (2018). Automatic building segmentation of aerial imagery using multi-constraint fully convolutional networks. R*emote Sensing*, *10*(3), 407 - 426. doi:10.3390/rs10030407

Zhou, Y., & Berg, T. L. (2016). Learning temporal transformations from time-lapse videos. *European conference on computer vision*, 262-277. doi:10.1007/978-3-319-46484-8_16