Learning a River Network Extractor Using an Adaptive Loss Function

Furkan Isikdogan, Alan Bovik, and Paola Passalacqua

Abstract—**We have created a deep-learning-based river network extraction model, called DeepRiver, that learns the characteristics of rivers from synthetic data and generalizes them to natural data. To train this model, we created a very large database of exemplary synthetic local channel segments, including channel intersections. Our model uses a special loss function that automatically shifts the focus to the hardest-to-learn parts of an input image. This adaptive loss function makes it possible to learn to detect river centerlines, including the centerlines at junctions and bifurcations. DeepRiver learns to separate between rivers and oceans, and therefore, it is able to reliably extract rivers in coastal regions. The model produces maps of river centerlines, which have the potential to be quite useful for analyzing the properties of river networks.**

Index Terms—**Coastal systems, convolutional neural networks, deep learning, river network extraction.**

I. Introduction

ACCURATELY mapping rivers from remotely sensed imagery is a challenging task, particularly in deltaic systems. Existing methods [1], [2] supply limited power to map river deltas, although they have been successfully applied to remotely sensed data in noncoastal regions. To address this problem, we proposed an automated channel extraction algorithm [3], [4] that uses a unique set of optimized hand-crafted filters to extract complex river networks in coastal systems from multispectral remotely sensed images. This tool, which is called RivaMap, uses the multiscale singularity index [5], which is a powerful tool for finding continuous curvilinear structures in images, where the index has a strong response. We used this new tool to successfully delineate rivers and to separate rivers from oceans.

One drawback of the singularity index is that it does not have a strong response at river junctions and bifurcations, resulting in gaps between centerlines. Obtaining the centerline connectivity at these points is a difficult problem to solve but is needed to quantitatively analyze the structure of river networks and their dynamical changes in response to environmental forcing. This is particularly relevant in coastal systems, where densely populated landscapes are at risk due to natural and anthropogenic factors [6]. The structure of deltaic networks has not been as studied as their upstream counterpart due to a lack of appropriate tools [7].

In this letter, we describe an alternative data-driven approach to river centerline detection. Our learned model, called DeepRiver, is based on the assumption that, given a sufficient number of examples, a deep convolutional neural network can learn image filters that respond strongly to centerlines. We first feed a neural network with a large volume of synthetic data covering a wide variety of possible configurations of channels and their corresponding centerlines. We then apply the trained model on real remotely sensed data, including simple water index responses and surface water maps generated by a separate learned water extraction algorithm called DeepWaterMap [8]. We thin the outputs of DeepRiver to a single-pixel-width skeleton to obtain centerlines, and we deploy the distance transform to estimate the channel width at every centerline point. Finally, we validate the centerline detection and width estimation performance on hand-cleaned ground truth data. While there has been some research on the applications of convolutional neural networks on similar problems, including models trained to extract roads from remotely sensed images [9], [10], the potential of using large-scale synthetic data and adaptive loss rerouting to extract centerlines remains unexplored. To the best of our knowledge, this is the first model that uses a specialized convolutional neural network with a synthetic data generator and an adaptive loss function to extract river centerlines from remotely sensed images. Our code and the trained models are freely available at https://github.com/isikdogan/deepriver/.

II. Data Generation

Our data generation process produces an unlimited stream of randomly generated synthetic input images with corresponding labels. The synthetic images consist of random configurations of local channel segments. Although a single synthetic sample is linear in nature, collectively they allow for learning to approximate complex curvatures as locally linear structures.

The input images are $P \times Q$ binary matrices that are designed to simulate water masks. Each pixel in the image is labeled as *centerline*, *noncenterline water*, or *land*.

The process starts with an empty mask where all pixels are labeled as land. Then, it draws a main channel centerline having a random orientation using Bresenham's algorithm [11] and labels the drawn pixels as *centerline*. The algorithm then expands the centerline by a random width and labels the expanded regions as *noncenterline water*. The widths

Manuscript received September 19, 2017; revised December 26, 2017 and January 31, 2018; accepted February 28, 2018. Date of publication March 19, 2018; date of current version May 21, 2018. The work of P. Passalacqua was supported by the National Science Foundation under Grant CAREER/EAR1350336, Grant FESD/EAR1135427, and Grant SEES/OCE-1600222. *(Corresponding author: Furkan Isikdogan.)*

The authors are with the University of Texas at Austin, Austin, TX 78112 USA (e-mail: isikdogan@utexas.edu; bovik@ece.utexas.edu; paola@austin.utexas.edu).

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LGRS.2018.2811754

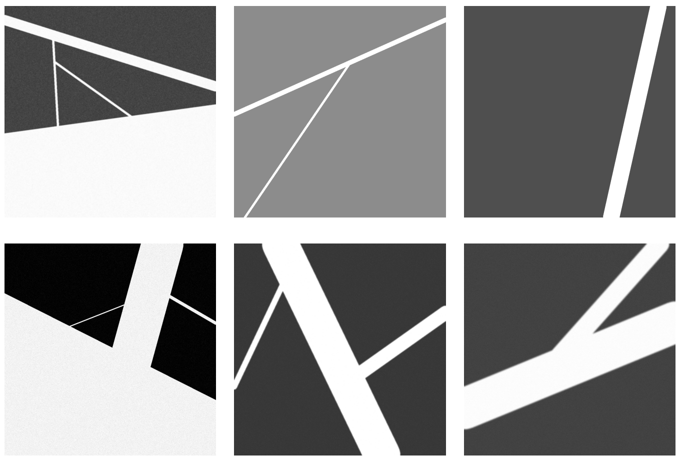

Fig. 1. Examples of randomly generated channels and branches.

are modeled as a uniform random variable between 1 and $N/4$ pixels, where $N = \min(P, Q)$ is the smaller matrix dimension. The width is bounded by $N/4$, since the banks of wider channels might otherwise be indistinguishable from shorelines. A random number of branches are then added to the existing centerlines, by iteratively drawing tributary centerlines having random orientations and widths, starting at randomly selected centerline pixels. The number of branches is selected randomly from the set of integers {0,1,2,3}, while the width of a tributary channel is a number lying between 1 and the width of the main channel, so that the tributary channels are always narrower than the main channel. A shoreline is added with probability 0.5 by dividing the image into two randomly partitioned regions and then labeling all pixels in one partition as *noncenterline water*. Finally, the images are distorted with additive noise, brightness and contrast shift, and Gaussian blur, in random order and magnitude to emulate imperfect imaging conditions (Fig. 1).

III. Training a Centerline Extractor

A. Model

We used a deep neural network and designed a special loss function to learn to extract river centerlines. The model consists of 10-scale convolutional encoder–decoder modules (Fig. 2). Each convolutional module contains three convolutional blocks, which consist of a convolutional layer followed by batch normalization and rectified linear unit activation layers [i.e., max $(0, x)$]. The model uses skip connections [12] to reuse previous layer features and to fuse multiscale features efficiently. We have shown in our earlier work [8] that this architecture worked well on remotely sensed images for surface water mapping.

We trained this model to extract river centerlines from a large number of single-band synthetically generated and labeled water masks. The model accepts as inputs single-band water maps and outputs pseudoprobabilities for each class label. The pseudoprobabilities are simply computed by passing the output layer activations through a softmax function.

The synthetic input data were generated in run time during training until the average training loss is converged. Therefore,

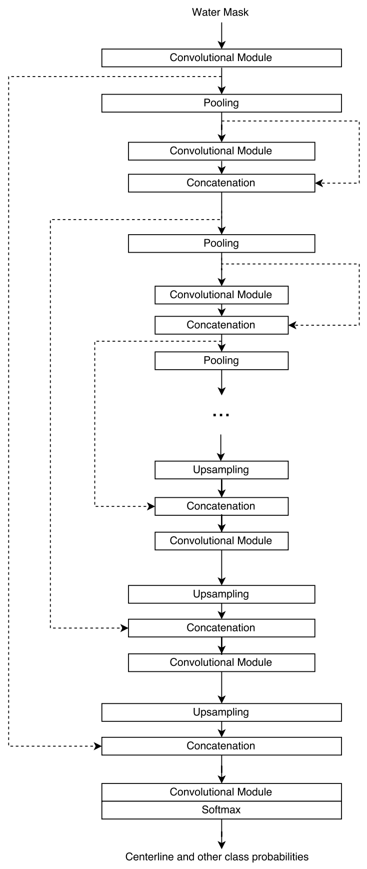

Fig. 2. Overall architecture of the model. The skip connections (dashed lines) on the left fuse multiscale features, whereas the skip connections on the right help to reuse previous layer activations.

the model was not trained on the same data samples over several epochs as is often done. We were able to do this because of the simplicity and ease of computation of the training matrices. Overall, more than 500 000 input matrices were generated and used during training.

B. Adaptive Loss Function

The centerlines have a relatively small number of pixels as compared with the land and noncenterline water pixels. Therefore, minimizing a uniform cost function during training would cause centerlines to be minimally adapted to.

We addressed this problem by designing an adaptive loss function that works well on imbalanced classes without explicit class rebalancing. Our adaptive loss function first computes the pixelwise cross entropy between the predicted and target label matrices. Then, instead of averaging the loss over all pixels, it applies a spatial max pooling to the pixelwise loss values (Fig. 3). This max pooling operation adaptively reroutes the pixelwise loss values, effectively shifting the attention toward the pixels with the highest loss value within a local neighborhood. This forces the model to work harder toward learning the harder-to-learn pixels without explicitly assigning a cost weight to each class.

The size of the pooling window determines the number of pixels over which the maximum loss is evaluated. A pooling

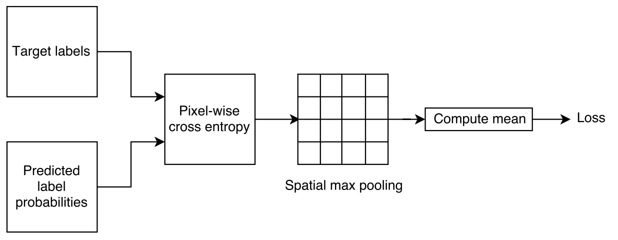

Fig. 3. Computation of the adaptive loss.

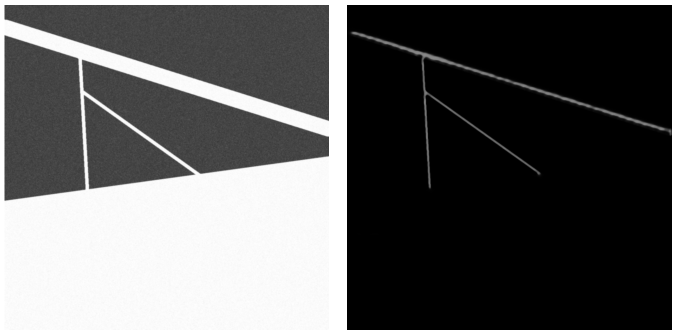

Fig. 4. Channel extraction results on synthetically generated channels. (Left) Input image has a main channel, two random branches, and a shoreline. (Right) Model produces a strong response at the centerlines, successfully connects the centerlines at the junctions, and separates between channels and the ocean.

size of 1×1 would disable the loss rerouting, whereas a pooling window as large as the input image would make the training highly sensitive to noise, since the model would focus only on a single pixel at every iteration. We empirically choose the pooling size as 8×8 and found that the results are not highly sensitive to the pooling size except when it approaches the extreme sizes.

This approach has two major benefits over explicit class rebalancing. The first one is that it adapts to the input data automatically. Therefore, a change in the input data distribution does not require recomputing the class frequencies and changing the class weights accordingly in the loss function. The second one is that since the loss is essentially class-agnostic, it learns to compensate for within-class imbalance as well as between-class imbalance. For example, centerlines are hard to learn as compared with other classes, but intersections are even harder to learn within the centerlines class. This simple adaptive loss function drove the model to efficiently learn centerlines and intersections on river networks.

C. Generalization to Natural Input Images

After training the model on synthetic images using adaptive loss, the model learned to extract centerlines from both synthetic (Fig. 4) and natural images (Fig. 5). The model is sensor-agnostic and robust against noise, detail loss, and brightness and contrast shifts as a result of being trained on randomly distorted images. Therefore, the input can be any type of remotely sensed image as long as there is a contrast between water and nonwater pixels. For example, the model can be used to extract rivers from Landsat-based water maps generated by DeepWaterMap [8] or even a simple water index, such as the modified normalized difference water index (MNDWI) [13]) (Fig. 5). Of course, the quality of the end result will depend to some degree on the efficacy of the water index that supplies input to the model.

The output layer activations of DeepRiver are remarkably similar to the multiscale singularity index responses generated by RivaMap [4], both responding strongly to curvilinear structures in their inputs. However, DeepRiver better handles intersections, while also yielding thinner channel responses that are more concentrated around the centerlines (Fig. 5). Furthermore, RivaMap occasionally produces spurious responses to river and ocean boundaries as a result of using an index that has some edge response [4]. Unlike RivaMap, DeepRiver has virtually no response to steplike edges (Figs. 4 and 5). This improved property allows for separating rivers from oceans and reliably extracting rivers in coastal regions. By using a learned model that detects rivers and that more successfully separates them from oceans, DeepRiver delivers accurate results under a wider variety of conditions.

IV. Centerline Delineation and Width Estimation

DeepRiver delineates the rivers in a given input image. However, the centerlines in the output are usually greater than one pixel wide. DeepRiver does the hard part of thinning by capturing the structure. Then, a simple thinning algorithm is used as a postprocessing step to ensure that the produced skeleton is single-pixel wide. We use a fast thinning algorithm [14] which iteratively removes noncenterline pixels until no further pixels can be removed. The outputs of the centerline extraction model do not require excessive thinning, since the centerlines extracted by the model are already thin. Therefore, the thinning procedure produces stable and reliable centerline results.

To estimate the channel width at each centerline pixel, our algorithm computes the closest distance between the centerline and the banks of rivers. Doubling this distance yields the river widths. We used Borgefors' distance transform algorithm [15] to find the shortest path to the nearest land pixel from each centerline point location.

V. Evaluation

We evaluated the centerline detection and width estimation performance of DeepRiver by measuring how well it could reconstruct a river map from its centerline and width estimates. We compared the results of our learned model with the results generated by a nonlearning-based model RivaMap. We used as input a manually annotated water mask of the Ganges–Brahmaputra–Jamuna delta [16], which covers the delta with 30-m resolution. We manually delineated the shorelines and separated the channels from the ocean on the input water mask.

We reconstructed the channels in the input water mask by automatically extracting the centerlines, by determining where the centerline probability was higher than the other classes and regrowing the centerline pixels by amounts equal to their widths (Fig. 6). Then, we compared the reconstructed channels against the ground truth using precision [TP / (TP + FP)], recall [TP / (TP + FN)], and the corresponding commission [FP / (TP + FP)] and omission [FN / (TP + FN)] errors. As an

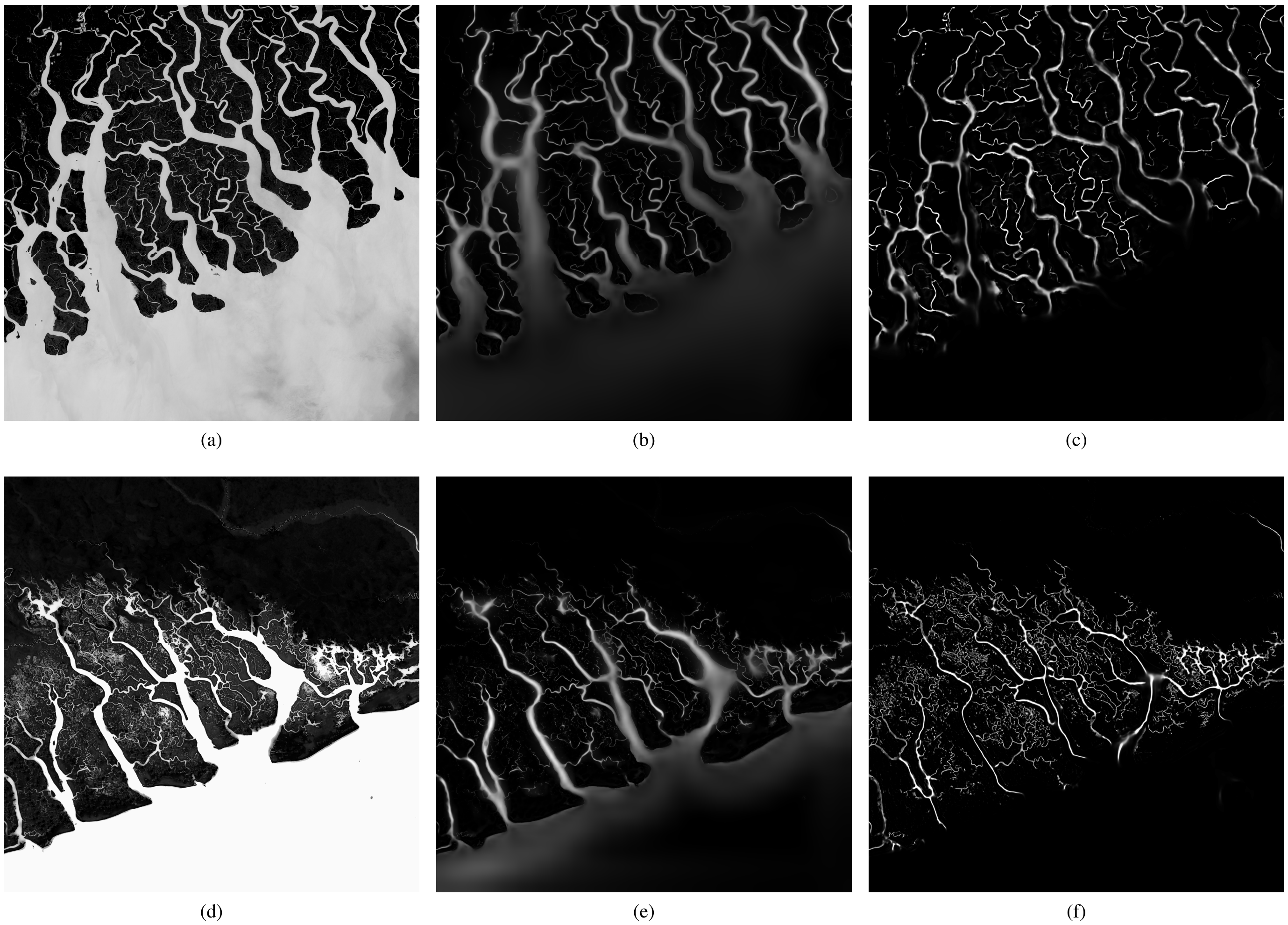

Fig. 5. Channel extraction results on different types of natural images. (a) and (d) Input images. (b) and (e) Channels extracted by RivaMap [4]. (c) and (f) Channels extracted by DeepRiver. Input (a) was generated by applying a water index (MNDWI) to a Landsat 8 image that shows a portion of the Ganges–Brahmaputra river delta (WRS-2 path/row: 138/45). Input (d) was generated by running DeepWaterMap on a Landsat 8 image that shows a portion of the Niger Delta (WRS-2 path/row: 188/57). Both RivaMap and DeepRiver are robust against contrast changes and can operate on water maps generated by different algorithms. DeepRiver yields thinner centerlines reducing the dependence on a thinning algorithm and provides a better separation between rivers and oceans.

TABLE I

QUANTITATIVE EVALUATION OF DEEPRIVER AND RIVAMAP

Metric	**RivaMap**	**DeepRiver**
Precision	0.87	0.97
Recall	0.85	0.88
Commission Error	0.13	0.03
Omission Error	0.15	0.12
F1 Score	0.86	0.92
Width Correlation	0.83	0.92

overall performance measure, we used the F1-score, which is the harmonic mean of precision and recall. To evaluate width estimation separately, we computed the correlation coefficient between the width estimates and the ground truth widths. To evaluate width estimation separately, we computed the correlation coefficient between the width estimates and the ground truth widths (Table I).

The results of our experiments showed that both RivaMap and DeepRiver are reliable river extractors. Overall, DeepRiver performed better, particularly in corner cases, such as channels near shorelines and intersections. Although DeepRiver is able to extract rivers reliably from noisy water masks, its centerline extraction ability, no doubt, will be adversely affected by poor quality input water masks. The interplay between river

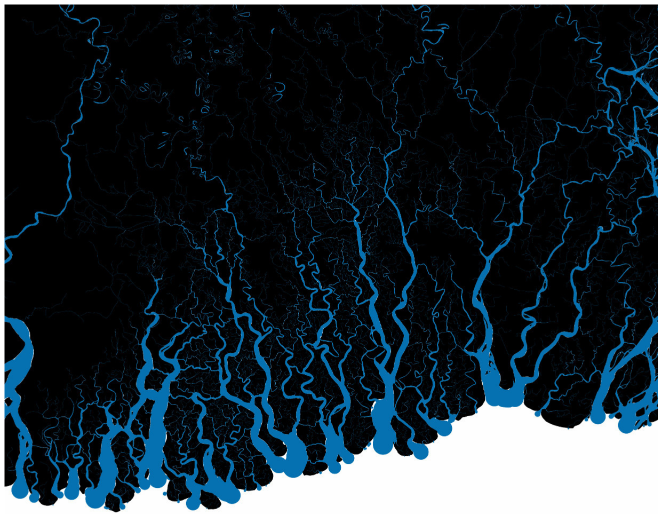

Fig. 6. Visualization of the automatically extracted channels. Reconstructed channels (shown in blue) overlaid on the input image.

extraction performance and water masking, as it affects our model and other models, is a broad topic that bears further study.

VI. Conclusion and Future Work

We have described a new deep-learning-based river centerline extraction model, named DeepRiver. Our model is robust against noise, detail loss, and brightness and contrast shifts. Therefore, it can extract river centerlines from surface water images, regardless of the types of sensors and algorithms used to generate them. DeepRiver uses a very lightweight model architecture and is amenable for efficient deployment on large geographic data sets.

We have proposed an adaptive loss function that makes learning harder-to-learn parts in the inputs easier without explicit class balancing. This adaptive loss function allowed for better handling the channel intersections and shorelines, leading to an improvement in the performance in challenging cases, such as extracting river networks on coastal regions. Although we found this type of loss function to be very useful for pixelwise classification of remotely sensed images, its use cases are not limited to remotely sensed images. Other types of image segmentation and transformation tasks can also greatly benefit from this adaptive loss rerouting approach. Future work would involve experimenting with using this loss function with other types of input data that involve between-class and within-class imbalance.

References

[1] T. M. Pavelsky and L. C. Smith, "RivWidth: A software tool for the calculation of river widths from remotely sensed imagery," *IEEE Geosci. Remote Sens. Lett.*, vol. 5, no. 1, pp. 70–73, Jan. 2008.

[2] B. Lehner, K. Verdin, and A. Jarvis, "New global hydrography derived from spaceborne elevation data," *EOS, Trans. Amer. Geophys. Union*, vol. 89, no. 10, pp. 93–94, 2008.

[3] F. Isikdogan, A. Bovik, and P. Passalacqua, "Automatic channel network extraction from remotely sensed images by singularity analysis," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 11, pp. 2218–2221, Nov. 2015.

[4] F. Isikdogan, A. Bovik, and P. Passalacqua, "RivaMap: An automated river analysis and mapping engine," *Remote Sens. Environ.*, vol. 202, pp. 88–97, Dec. 2017.

[5] G. S. Muralidhar, A. C. Bovik, and M. K. Markey, "A steerable, multiscale singularity index," *IEEE Signal Process. Lett.*, vol. 20, no. 1, pp. 7–10, Jan. 2013.

[6] J. P. M. Syvitski *et al.*, "Sinking deltas due to human activities," *Nature Geosci.*, vol. 2, no. 10, pp. 681–686, 2009.

[7] P. Passalacqua, "The Delta Connectome: A network-based framework for studying connectivity in river deltas," *Geomorphology*, vol. 277, pp. 50–62, Jan. 2016.

[8] F. Isikdogan, A. C. Bovik, and P. Passalacqua, "Surface water mapping by deep learning," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 11, pp. 4909–4918, Nov. 2017.

[9] S. Saito, T. Yamashita, and Y. Aoki, "Multiple object extraction from aerial imagery with convolutional neural networks," *J. Imag. Sci. Technol.*, vol. 60, no. 1, pp. 10402-1–10402-9, 2016.

[10] Y. Wei, Z. Wang, and M. Xu, "Road structure refined CNN for road extraction in aerial image," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 5, pp. 709–713, May 2017.

[11] J. E. Bresenham, "Algorithm for computer control of a digital plotter," *IBM Syst. J.*, vol. 4, no. 1, pp. 25–30, Mar. 1965.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[13] H. Xu, "Modification of normalised difference water index (NDWI) to enhance open water features in remotely sensed imagery," *Int. J. Remote Sens.*, vol. 27, no. 14, pp. 3025–3033, 2006.

[14] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Commun. ACM*, vol. 27, no. 3, pp. 236–239, 1984.

[15] G. Borgefors, "Distance transformations in digital images," *Comput. Vis., Graph., Image Process.*, vol. 34, no. 3, pp. 344–371, 1986.

[16] P. Passalacqua, S. Lanzoni, C. Paola, and A. Rinaldo, "Geomorphic signatures of deltaic processes and vegetation: The Ganges-Brahmaputra-Jamuna case study," *J. Geophys. Res., Earth Surf.*, vol. 118, no. 3, pp. 1838–1849, 2013.