SONDERPUBLIKATION Nr. 002

Bathymetry from multispectral aerial images via Convolutional Neural Networks

Hannes Nübel

DHyG-Sonderpublikation
Nr. 002
Juni 2020

Herausgeber:
Deutsche Hydrographische Gesellschaft e. V. (DHyG)

DOI: 10.23784/DHyG-SP_002

DHyG-SONDERPUBLIKATION

Nr. 002

Bathymetry from multispectral aerial images via Convolutional Neural Networks

Hannes Nübel

Abstract

Recently, optical approaches were applied more often to derive the depth of waterbodies. In shallow areas, the depth can be deduced mainly by modeling the signal attenuation in different bands.
In this approach, it is examined how well a Convolutional Neural Network is able to estimate water depths from multispectral aerial images. To train on the actually observed slanted water distances, the net is trained with the original images rather than the orthophoto. The utilized dataset contained, apart from RGB images, also panchromatic images with a Coastal Blue filter, which were captured synchronously. As a further step, the value of the Coastal Blue band in the CNN-based regression is analyzed.
The trained CNN is showing a standard deviation of 3 to 4 decimeters. It is able to recognize trends for varying depths and ground covers. Problems mainly occurred when facing sunglint or shaded areas. The inclusion of the Coastal Blue band added value with respect to the distribution of depths in the test area.

Contents

List of Figures

1 Introduction

1.1 Motivation

Reconstructing the surface of the earth by means of photogrammetry is an established method. Coordinates of object points can be computed via forward intersection when the respective point is observed in two or more images. However, applying this procedure to water surfaces is more complex. Nevertheless charting water depths is necessary, especially in shallow water areas, for example when considering safe routing of ships, or when determining the volume of a lake which is needed for extinguishing fires.
The complexity involves that measuring of identical points is rather complicated due to the specular and dynamic nature of the water surface. Furthermore, there is refraction on the water surface because of transition of the image ray between two media. For generating an orthophoto this particularly means that every pixel in each image has its unique refracted ray corresponding to the water surface which also may show local dynamics.
Thus, to find the corresponding ground points of each pixel, this ray has to be traced from the respective image position, with its direction given by the orientation of the image, also considering the refraction on the watersurface. Another point is that even if the direction of each ray is known, enough

identical points have to be detected to calculate their coordinates with help of the intersecting rays from the images. That is also demanding, because the submerged ground is often homogeneous and in addition there is attenuation in the water. Also, because of reflection and other factors the same points can appear differently when taken from different perspectives.
Because of different magnitudes of absorption of light for various spectral bands in the water column, it is also possible to fit a linear or higher dimensional regression model to band ratios, approximating the relation from radiometry to depth. But as soon as the scene contains different types of vegetation on the ground of the water basin, a more complex regression model is needed. Furthermore, spectrally based bathymetry estimation is commonly carried out based on orthophotos. Not only are orthophotos of waters prone to geometric errors due to neglection of ray refraction at the water surface, but most also ignore the fact that only pixel values from the image center (nadir direction) directly relate to water depth whereas pixels from the edge of an image rather show the slanted water distance. Each pixel of an aerial image, in turn, stores radiometric information which is mainly related to the potentially slanted under water distance of the respective image ray. Especially for aerial images taken with wide-angle lenses it is therefore beneficial to perform the bathymetry estimation based on the (oriented) images rather than the orthophoto.

To extend the linear regression approach, a Convolutional Neural Network (CNN) can be used to cope with variations in bottom reflectance. Pixel-wise depth estimation based on the oriented aerial images require the slanted water distance for the image pixels for training. This information can e.g. be derived from bathymetric LiDAR (Light Detection And Ranging), especially when carried out concurrently with the image capture.

The CNN based approach has the advantage that spatial context information is taken into account. The reliability of the net is therefore increased, since proximity often implies similar depths.

1.2 Related Work

Originally bathymetric data was captured with sound navigation and ranging (SoNAR) (Masnadi-Shirazi et al., 1992). Because the instruments are mounted underneath a ship in the water, in comparison to sensors above the water surface there is no transition between two media and it is not dependent to dynamics of the water surface. Besides, the water does not have to be as clear as for optical sensors.
Currently, another increasingly applied technique is to derive water depths via LiDAR from airborne platforms (Irish and White, 1998). Instead of sound waves as with SoNAR, laser pulses are emitted, from which afterwards as well the echos are measured. Because of the transition of the laser pulse from the atmosphere into the water, a change in the speed of the light has to be modeled according to the runtime measurements. A comparison to the SoNAR approach was made by Costa et al. (2009). The main advantage was the increased efficiency especially for large areas, because with a plane or drone the same area can be assimilated in less time than by ship, which of course also is a cost determining factor. Other than that, it is also possible to acquire data in remote areas, as well as in peripheral areas of a waterbody, which are not accessible with SoNAR.
For topographic applications stereo photogrammetry is a common strategy. Identical points are detected in multiple images and their position is calculated

in a global system via bundle adjustment. The challenges that arise for this process in bathymetric applications are discussed by Mulsow et al. (2019) in comparison to data acquisition with an airborne laser scanner. The achievable accuracies are similar and aerial images can be captured cheaper and more flexible. But this method is dependent on the texture of the ground to find identical points. Especially in deep or homogeneous areas this is not given and therefore larger discrepancies are occurring.

The idea of spectral bathymetry, is to create a mathematical or physical model that builds a connection between reflectance and water depth. In theory it is thus possible to estimate the water depth for each pixel. A common assumption is that the bottom reflectance is acting negatively exponential referring to the water depth (Lyzenga et al., 2006). Furthermore an offset has to be included, modeling the reflectance for an infinite water depth.
Even though this behavior is helping to derive depths, it is also a limiting factor. Meaning that this approach can only be applied for shallow waterbodies, depending on the attenuation of light in the water column. Since reflectance varies strongly for different ground covers it can be useful to include multiple spectral bands (Legleiter et al., 2009), because the radiance for particular ground types is similar in related bands. Especially logarithmic ratios between bands seem to be able to approximate depth well. Also a combination of the red channel, which is most affected by signal absorption and a better penetrating spectral band seems to have high informational content (Mandlburger et al., 2018).
Developing a bathymetric model including spectral and spatial information with help of a neural network has also be done by Wang et al. (2019). In that approach the spatial information is given by the coordinates X and Y. Together with four multispectral band ratios they form a feature vector which is the

input layer of the net, which after the input layer consists of three hidden layers and one output layer building a multilayer perceptron (MLP).
The approach applied here mainly differs in two points. First, the spatial information is included by using a CNN, which is learning weights for small kernels covering a certain area around a pixel, rather than depending on the actual X and Y coordinate of a point. Because of this more general assumption the net can also be transferred to other areas. Second, apart from the net being a CNN, the architecture is deeper and instead of deducing the depth for single points with certain features, a semantic segmentation of complete images is done. So the net is able to learn high-level feature extracting convolution kernels because of having more hidden layers.

1.3 Aim and Structure of this work

In this thesis, the approach of training a convolutional network to predict the slanted distances from image rays inside a waterbody, will be examined. Next to quality assessment and critical discussion, it will also be discussed to which extent the Coastal Blue channel has an influence on the network.
In Section 2 the underlying dataset and its preparation is introduced. The methods including preprocessing the reference data and training the net is then outlined in Section 3. Subsequent Section 4 is presenting and discussing the results of the trained net and finally Section 5 is summarizing and concluding the work and further addressing the possible topics for proceeding after the thesis.

2 Dataset

2.1 Multi View Stereo

In the following, the acquisition of the investigated data is addressed. The images used for the processing were taken at the "Autobahnsee" in Augsburg (Figure 2.1), which is approximately up to 5 meters deep and has a small isle as well as multiple vegetation patches and a complex elevation profile. For data acquisition two IGI DigiCAM 100 cameras are used, which are based on PhaseOne iXU-RS 1000 cameras with 11608 by 8708 pixels each, one equipped with an RGB sensor and the other with a pan-chromatic sensor and a filter for the Coastal Blue wavelength (Mandlburger et al., 2018).
Using the information from both images, the same position and orientation is required. But for practical reasons the cameras had to be mounted side by side 2.2. Therefore, the Coastal Blue image is transformed into the RGB datum using a homography with the Software MATLAB (2018). To transform images via homography, the images need to be either only rotated, or the object in the images should be planar (Agarwal et al., 2005). This is not the case, but the area is rather flat and the cameras are very close to each other being triggered synchronously. Furthermore, the height variation of the terrain is comparatively low to the flying height, so the error is neglectable. Because

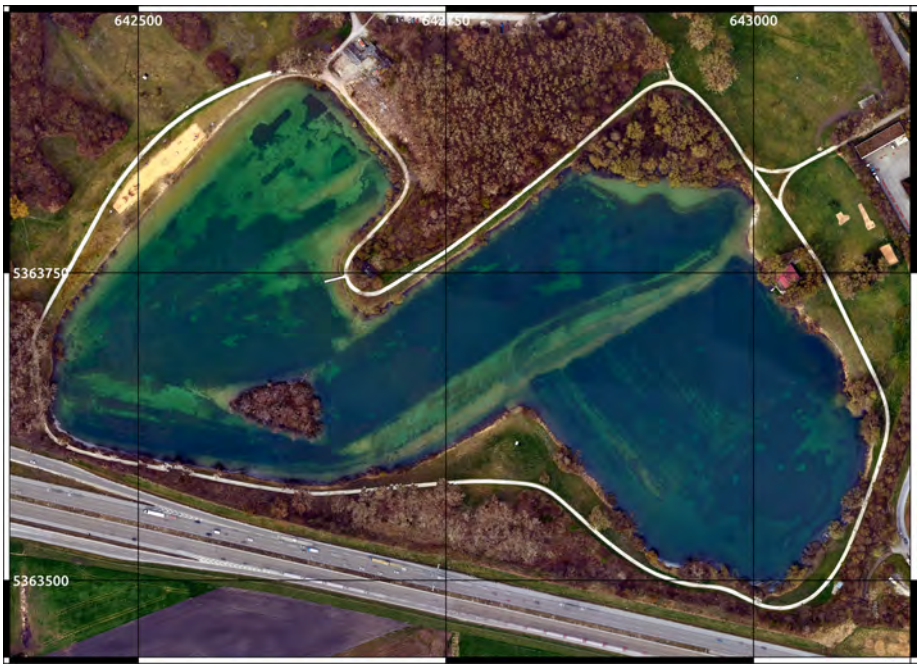


Figure 2.1: Orthophoto Autobahnsee with different ground covers.

Figure 2.2: Sensor configuration (left: RGB, right: pan chromatic with Coastal Blue filter) (taken from Mandlburger et al., 2018).

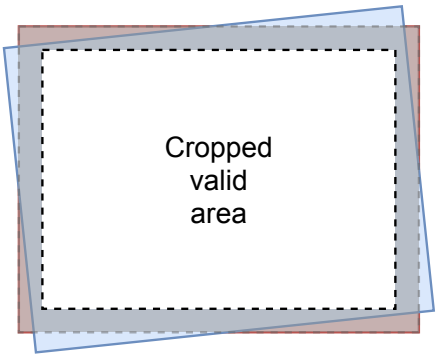


Figure 2.3: Schematic view of applied no-data-frame for merged images. RGB image (red), Coastal Blue image transformed into RGB datum (blue) and applied no data frame for merged images (transparent gray).

of the transformation into the RGB image datum, there will be pixels in the new Coastal Blue image with no data in the overlapping area with the RGB image. To ensure having an image with pixels valid for all bands, a small no data frame at the boundaries of each image is applied for the merged images in the RGB image datum. All pixels in this margin area will be ignored later on. That procedure is shown schematically in Figure 2.3.

2.1.1 Error estimation of homography

The identical points, which were used to estimate the homography do have a maximum height variation of about 40 meters. For the utilized points the mean error after applying the transformation is about 0.6 pixels in the image space. The maximum error of 1.5 pixels has an effect of about 11 cm in the object space. When propagating the error from the mean terrain height to the deepest point in the lake (Figure 2.4) the difference is merely within a few

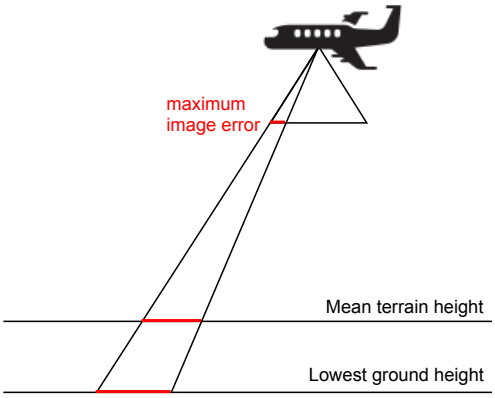


Figure 2.4: Sketch of error propagation from mean terrain height to lowest ground point within the lake.

millimeters. By considering that this is the maximum error at the corner of the image, where its impact is maximum, the errors in the water should be mostly smaller than 11 cm (i.e. less than 2 pixels). In addition, the refraction at the water surface is reducing the error, because rays towards the edges of the image are increasingly deflected towards the nadir direction.
By assuming that mostly similar regions imply similar depths, the potential errors are accepted, to be able to evaluate the advantage of including the Coastal Blue band.

2.2 LiDAR

Moreover, the employed hybrid sensor system also integrates a RIEGL VQ-880-G topo-bathymetric laser scanner (Riegl, 2019) to obtain a point cloud, from which the water surface model and ground model can be extracted. In Figure

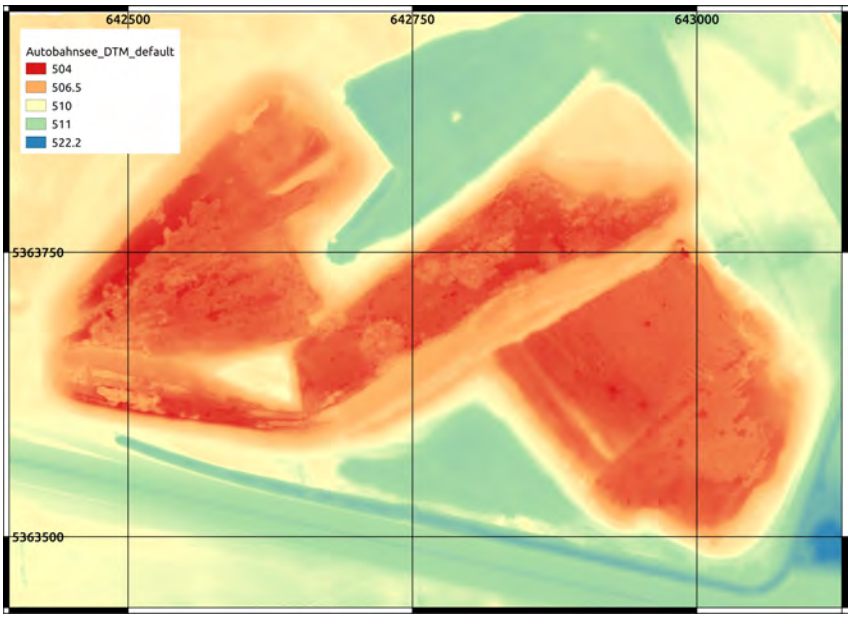


Figure 2.5: Topo-bathymetric LiDAR-derived Digital Terrain Model Autobahnsee.

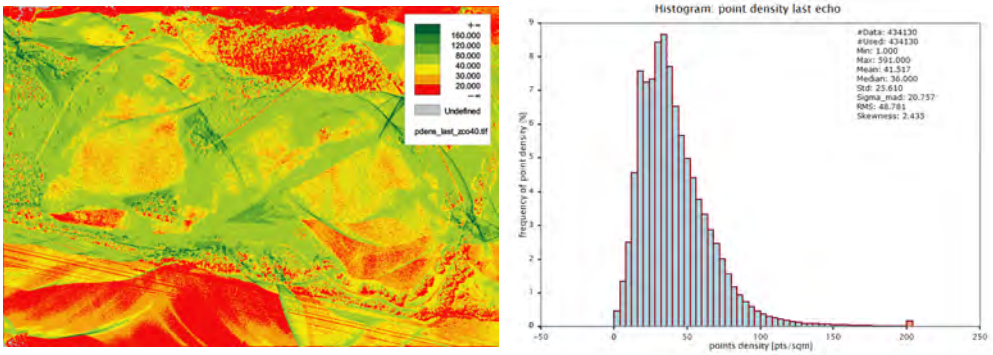


Figure 2.6: Last-echo point density map (same extent as in Figure 2.5).

2.5 the ground model for the observed area is depicted. It is noted that there are complex structures at the ground of the lake caused by the distribution of soil and vegetation. These will be used to extract the reference data, being the slanted distances of the images rays in the water. The scanner is designed for shallow water mapping. Therefore, a green laser with wavelength 532 nm is used, because of its capability to penetrate water for measuring the ground of a waterbody and available high energy laser sources (Doneus et al., 2015). The mean point density of the obtained point cloud is about 40 points per square meter to get a dense model. In Figure 2.6 it becomes apparent that the points are less dense in deep water regions, because of attenuation of the laser and refraction at the water surface.

3 Methods

3.1 Preprocessing reference data

The following Section is discussing the applied methodology to derive the reference data the applied CNN is to be trained with. It is given by the respective slanted distances of the rays of every pixel in the water. To obtain them, the orientations of the camera and a water surface model (WSM), as well as a ground model are used to trace the path of rays from the camera to the corresponding ground point with consideration of refraction at the water surface. The WSM is estimated from the first echos of the laser scanner, while the last echos constitute the basis for filtering the ground points and, finally, calculating the Digital Terrain Model (DTM) the ground model.

3.1.1 Raytracing

In order to get the slanted distances, the rays corresponding to the individual pixels can be calculated in the local camera coordinate system using the interior orientation of the camera. They can then be transformed into a global coordinate system with help of the pixel coordinates, as well as positions and orientations of the camera at the time of exposure (Kraus and Waldhäusl, 1996).

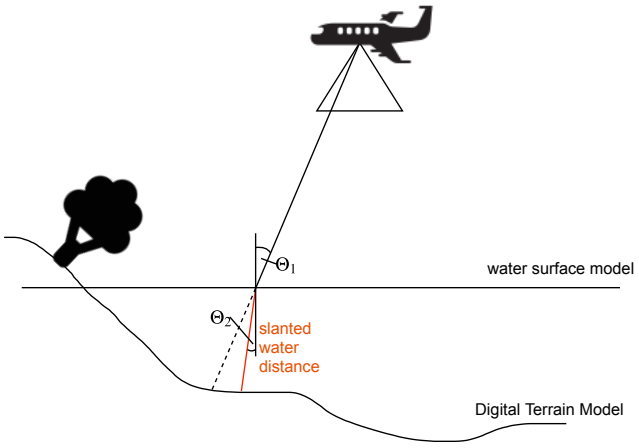


Figure 3.1: Refraction of image ray on the water surface.

Those steps are implemented in python, using the orientation file containing the interior and exterior orientations. After the file is loaded, an array with the size of one image can be created, containing row and column for each pixel. Using the interior orientation of the camera, the two dimensional pixel coordinates can be transformed into three dimensional image coordinates or rays.

$$\mathbf{x}_{cam} = \begin{bmatrix} m_x \cdot c & 0 & x_0 \\ 0 & -m_y \cdot c & y_0 \\ 0 & 0 & -1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} c_{pix} \\ r_{pix} \\ 1 \end{bmatrix}$$

By applying the rotation matrices which are part of the exterior orientation of each image, the direction of rays in the global coordinate system can be computed for each image. Having those and the coordinates of the starting point (i.e. the camera positions), simulated ground points are needed, that

have to be underneath the reference ground model, to which the intersections are to be calculated afterwards.

$$\mathbf{X} = \mathbf{X}_0 + \lambda \cdot \mathbf{R} \cdot \mathbf{x}_{cam}$$

Therefore, the rays multiplied by a factor are added to the corresponding camera position. These points and the rays are then stored for every image together with column and row of the pixel as additional information, so that later the slanted distances can be stored as a raster image. The next step then is to intersect the rays with the WSM, which is done using the Software OPALS (Pfeifer et al., 2014).

3.1.2 Refraction

After the intersection points of the rays with the water surface are known, for their further propagation they have to be corrected due to refraction following Snell's law (Kotowski, 1988). That results in a change of direction for the ray depending on the incidence angle (Figure 3.1). This relation may be described with the following formula, in which Θ_1 and Θ_2 describe the incidence and refraction angle and n_1 and n_2 are the refractive indices in the atmosphere and water.

$$\frac{\sin\Theta_1}{\sin\Theta_2} = \frac{n_2}{n_1}$$

The refracted ray starting from the particular intersection point with the water surface, is afterwards intersected with the ground model, which gives the observed ground point in the respective pixel.
By knowing the two intersection points, the euclidean distance can be calculated, constituting the slanted distance through the waterbody. For the refraction correction the module opalsSnellius (OPALS, 2019) can be used.

Consequently it is possible to export the slanted distances together with the previously passed columns and rows. With that information a reference raster for every image can be created. An example of that can be seen in Figure 4.2a. The last step of preprocessing the data, is to mask the multispectral images, so that only pixels with valid reference depths are included.

3.1.3 Water surface and ground model with vegetation masking

In order to calculate intersections of image rays with the WSM, the WSM needs to be modified, as the used WSM is bigger than the actual water area. To crop it to the size of the lake, the difference between the WSM and the ground model is calculated. Only the area, where the WSM is above the ground model is used afterwards to ensure having only intersections for water observing rays.
Besides, vegetation on the border of the lake is not taken into account because the ground model is a Digital Terrain Model (DTM). In that case the network would falsely be trained on water depths, when there is actually only vegetation visible in the image.
To prevent - or at least reduce - this problem, the Digital Surface Model (DSM) may be used to mask the WSM if the DSM is higher. Because the images were taken in nadir view, the DSM correction is only applied once on the WSM and not individually for every image. To create the DSM, a grid size of 25 cm and a search radius of 1.5 m is set, within which either the highest point is set, or if all points are within a height difference threshold, it is interpolated. If there are not enough points in the search radius, the respective grid point is omitted. This was done with the module opalsDSM (OPALS, 2019). The problem in this case is, that especially above the waterbody, there are reflections also in

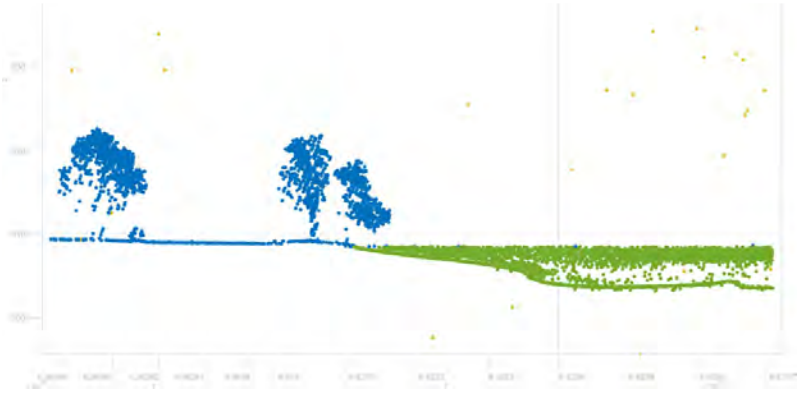


Figure 3.2: Section view of LiDAR point cloud of Autobahnsee (western shoreline). Points colored by intermediate classification into: point above (blue) and below (green) the water surface and isolated points (brown) (taken from Mandlburger et al., 2018)

the atmosphere, which can be seen in Figure 3.2. So, to not accidently mask the WSM because of such points, a rough polygon is defined, so that only the shore line is part of the algorithm. After the masking, to get a smoother output without isolated pixels and to reduce errors, an morphological opening can be applied with the module opalsMorph (OPALS, 2019). The outcome of that is the cropped WSM, shown in context with the orthophoto in Figure 3.3. This model constitutes the basis for raytracing (Section 3.1.1).

3.2 Deep learning

3.2.1 Neural Networks

The human brain is able to learn and recognize certain features and characteristics very fast and reliably with help of a massive amount of neurons, with

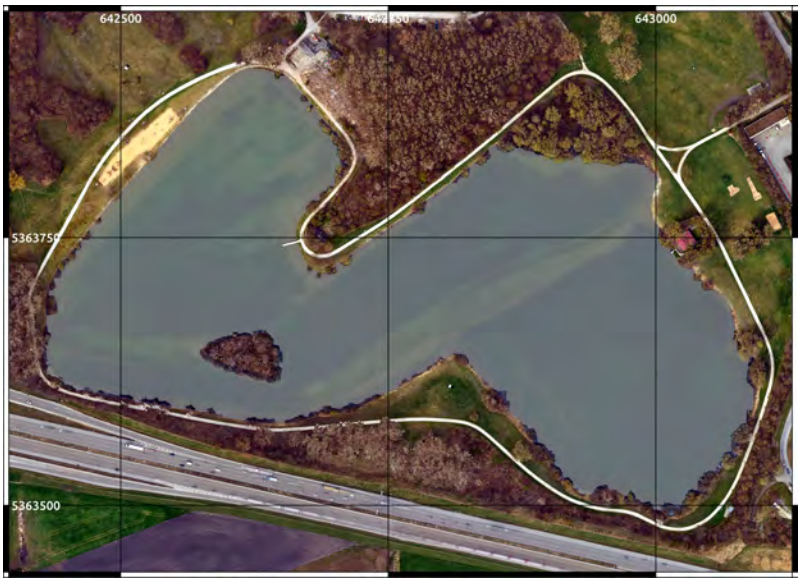


Figure 3.3: Cropped water surface model with vegetation masking.

even more connections (Nielsen, 2015). This concept of deriving information from a vast amount of connections is transferred to machines in the context of so called neural networks to build some kind of artificial intelligence. Organized normally in layers, we have neurons connected with neurons from previous layers. Their values are weighted, added and charged with a bias (Nielsen, 2015). Those weights and biases are to be learned within the training process of the net through backpropagation, which can be explained as the partial derivative of the loss function with respect to all weights (Nielsen, 2015). The output of every neuron is then computed with help of an activation function, which transforms the output to a certain range of values (Nwankpa et al., 2018). This is necessary because deep learning is primarily used for complex issues. In order to learn complex characteristics, a complex activation function should be applied to the output of a neuron. Without, the Neural

Network would be just a linear regression model.
To train the net, a large amount of training data is needed, with different appearances of the target quantity which the net is supposed to recognize. Furthermore, validation data is required to see how well it performs during training. For both, also reference data called ground truth is needed. With the ground truth and the output from the net, a loss function can be used to calculate the error of the model.
As mentioned before, the neurons are arranged in layers with the first layer being the input and the last one being the output, which mainly contains class scores. For neural networks the hidden layers in between are fully-connected layers, meaning that every neuron has a connection to every other neuron in the previous layer (Stanford University, 2019).

3.2.2 Convolutional Neural Networks (CNNs)

For applications relating to images, the input layer consists of as many neurons as there are pixels in one image multiplied by the number of bands. Especially for larger images and a deeper net with many layers, neural networks with fully connected layers accumulate and are taking a long time to be trained. That results from each connection representing a weight, and a vast amount of training data to learn those weights. CNNs are a more specific kind of neural networks with which especially images can be processed accurately and efficiently by benefiting from their spatial structure (Nielsen, 2015). This is realized by performing convolutions with the layers. Depending on the size of the kernels, small regions of, e.g., 3 by 3 pixels provide information for a neuron in a deeper layer. Because effective feature extracting convolution kernels might be useful not only at one position but everywhere in the image,

only one kernel is trained for the whole layer rather than training kernels relating to specific regions in the image. So instead of training weights for every neuron in each layer, only the weights for one kernel have to be trained. As a result the architecture of the model can be much deeper for accessing high-level features but still being efficient in context of training duration and required training data.

3.2.3 U-Net

The U-Net (Ronneberger et al., 2015), which is being used as a basis for the applied net, is a convolutional network that was designed for biomedical image segmentation. It is a fully convolutional network, meaning that the output image size is equal to the input size.
The name results from the architecture of the net, because it has a compressing and a symmetric expanding path, shown in Figure 3.4. It mainly consists of convolutional layers followed by a rectified linear unit (ReLU) and max-pooling layers in the compressing and up-convolution layers in the expanding path. The ReLU activation function, defined by $f(x) = \max(0, x)$, is commonly used in modern neural networks. In contrast to functions like sigmoid, it is not having the issue of a vanishing gradient, so even for high values the function is sensitive to changes in the input (Goodfellow et al., 2016). In the compressing path the dimension of the respective image tile is bisected in each pooling step. After each step, the number of feature maps calculated with the convolutional layers is doubled. The downsampling of the image tiles is done by max pooling, meaning that for the respective values of the sliding kernel, the maximum value is taken as output (Goodfellow et al., 2016). To restore the input dimension the tiles need to be upsampled afterwards, which

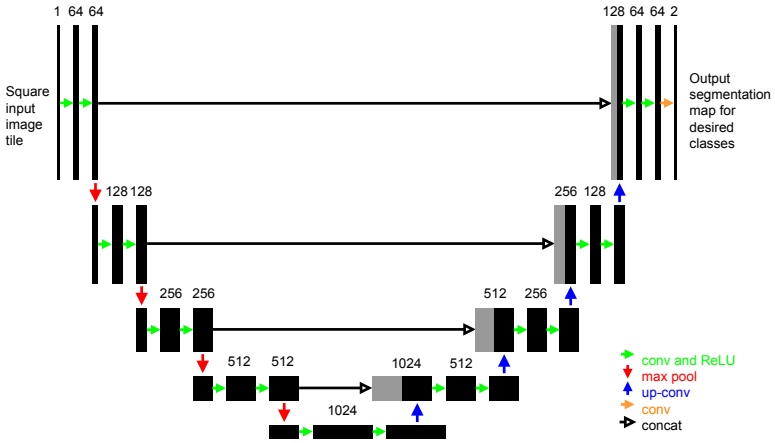


Figure 3.4: U-Net architecture. Each black box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. Gray boxes represent copied feature maps. The arrows denote the different operations. (Ronneberger et al., 2015)

is done via transposed convolution (Dumoulin and Visin, 2016). Instead of defining an interpolating kernel for the upsampling, it is as well learned and optimized as the net is trained.

The upcoming outputs in the U-Net are concatenated with the facing layers from the compressing path to preserve spatial information. Finally, there is a last 1x1 convolution at the end that is mapping each feature vector to the needed number of classes. To be able to predict values for pixels at the image border using convolutional strategies, meaningful pixel values beyond the edge of the image need to be provided. One of the commonly used strategies is to mirror pixels at the image boundary. Convolutions refer to a subset of pixels, processing the respective output for the center pixel. Without expanding the image at the edges, it would shrink and therefore it would not be possible to process a segmentation for the entire image.

Adapting the U-Net architecture

The implementation of the CNN is realized in python using the deep learning library keras (Chollet et al., 2015) with tensorflow backend (Abadi et al., 2015) and the net is trained on a Graphics Processing Unit (GPU) beacuse of the efficiency compared with a Central Processing Unit (CPU).
As mentioned in section 3.2.3, the U-Net (Ronneberger et al., 2015) serves as basis for the CNN. The main difference is, that the net is not being used for segmentation as in most approaches, but for fitting a regression model. So instead of having multiple classes with a normalized output for every pixel, only one quantity is trained, containing floating point numbers for the water distance of each pixel. Therefore, the output from the last convolutional layer has only a depth of one and instead of an activation function like sigmoid, by which the class scores can be received, once again ReLU is used. As loss function, the root mean squared error between the reference and the predicted values is calculated.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

Padding is used, to keep the size from the input image when performing a convolution. Consequently, the output image at the last layer will have the same size as the original image from the data set.
As in the original architecture of the U-Net, the width and height are also divided by 2 in every pooling step, to be able to have twice as much filters determining the size along the third dimension.
Overfitting is a common issue with deep nets, meaning that it is learning too specific characteristics that only appear in the training data set and do not represent a trend that can be transferred to likewise data. To restrict

this behavior, an additional dropout layer, which is randomly ignoring a percentage of the output, is added after the pooling steps.
Because the images are too big to train one epoch on a complete image, patches have to be generated. The size of the patch has to be divisible by two to the power of 4, to have a natural number of pixels in width and height after the fourth and last pooling layer. In this case, 480 by 480 pixel patches were used. This choice has been made, because if smaller patches were used, the net might have problems to learn the characteristics in shore areas, while bigger patches might lead to having trust in the structure of the lake, more than the spectral bands.
Image patches from the interior of the lake are fully covered while the number of valid pixels varies for patches in the shore area. The following strategy is applied to ensure a representative number of patches in the shore and interior area while at the same time excluding patches entirely on the dry part of the image. The number of patches which are picked randomly within the image, is set relatively to the valid pixels. If, for one randomly picked patch, the percentage of valid pixels is lower than 25 percent, the patch is rejected. This ensures, that every patch is containing enough data and also shore patches are taken into account, so not only the center of the lake is being trained. The patches are also augmented randomly by rotating the image or transposing dimensions. Data augmentation is very useful for enlarging training data, because known data is taken but modified to appear differently.
Last but not least to train a net, the images have to be separated into training images, which also contain a percentage of validation data, and test images which are not used at the training. For that, the lake area is split into two parts, which are marked in Figure 3.5. To make sure that the test data is completely new to the net, the images containing both areas were neither used in training,

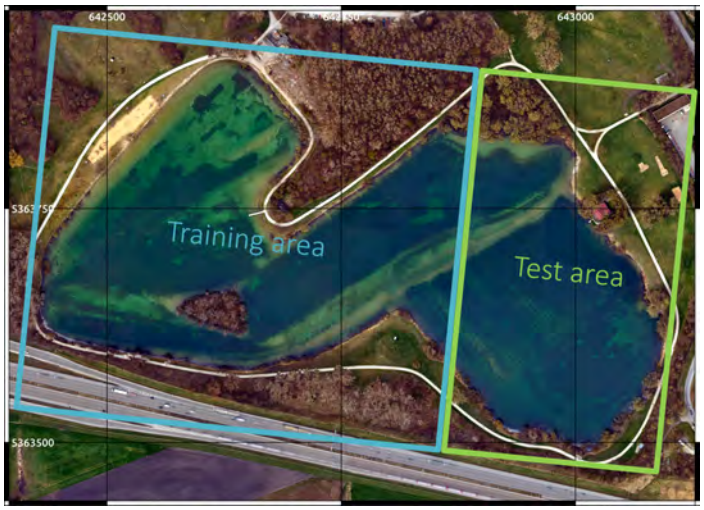


Figure 3.5: Distribution of training and testing area observed in the images.

nor for testing. In total 41 images are deployed in the training phase and 24 in the testing phase. However, it must be noted that the amount of data in each image is depending on the number of pixels showing the waterbody. The structures in the chosen areas differ rather strongly, so that it is possible to evaluate if the network is overfitting to the training area, or if it is learning characteristics that may be transferable also to other waterbodies.

4 Results and Discussion

4.1 Applied CNN for combined RGB and Coastal Blue band

While training the net, an indicator for the quality is the loss plot, which refers to the Root Mean Square Error (RMSE) of predictions with the current net for each training epoch (Figure 4.1). For the training phase, the used images are again divided in two parts. The training images are then used to train and adjust the weights and a smaller set of images is intended to validate these adjustments. Considering that the training loss should decrease constantly, as well as the validation loss with a small offset, the loss plot shows the expected behavior. Though, the importance should not be overestimated.
Applying the trained net to previously unseen data provides an independent performance test of the net. This data consists of a subset of all images, marked as test images. Thus it can be verified how well the net really learned certain characteristics instead of just memorizing the training data. An example for the prediction of a test image compared to the reference data can be seen in Figure 4.2. Despite the area on the upper right, in which the slanted underwater distances are predicted as too large, the predicted values seem to match the reference. Besides, there is no major discrepancy considering the trend

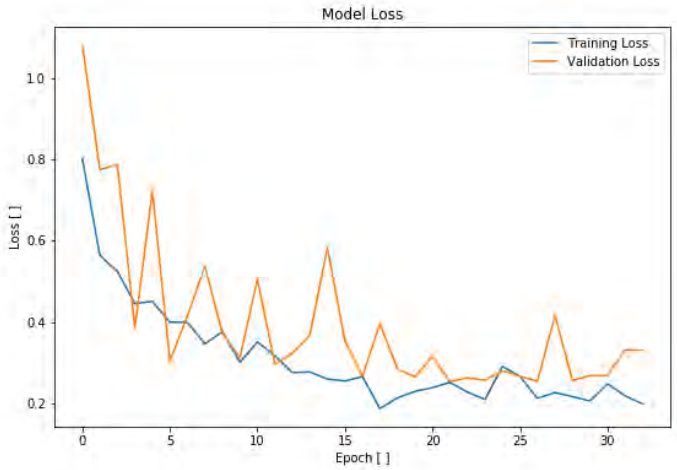
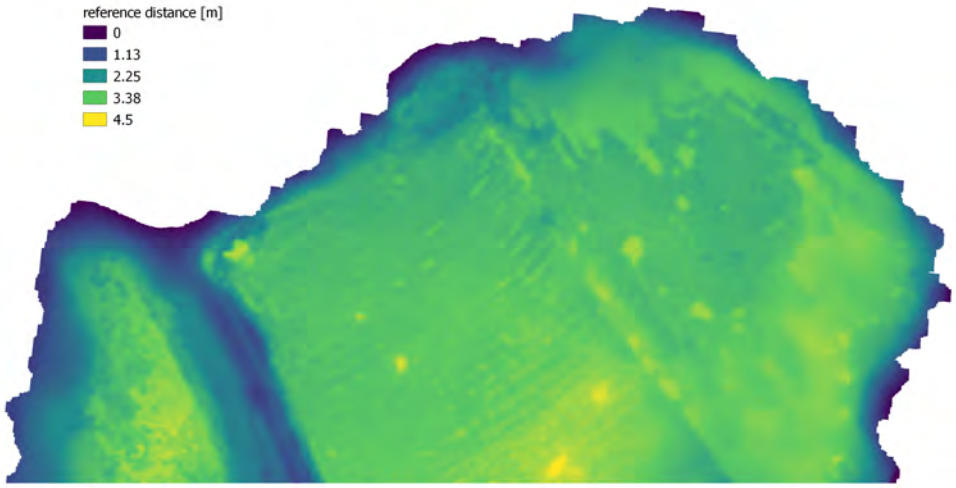


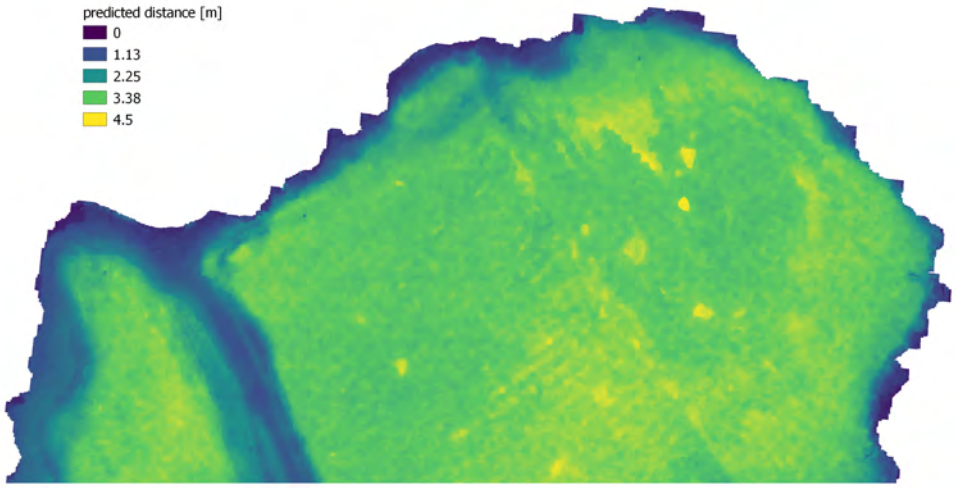
Figure 4.1: Loss plot with RMSE for each epoch during training.

of the water distances. What can be observed however, is a certain noise that may be caused by the camera sensor or by dynamics of the water surface.

After all test images are predicted, per-pixel distance deviations can be calculated by subtracting the predicted distance from the reference distance. By merging the deviations for all pixels of all test images, a histogram over all depth deviations (Figure 4.3) can be obtained (OPALS, 2019). It is noted that only water pixels are taken into account whereas all pixels in vegetation and on dry land are masked.



(a) Reference distances of test image



(b) Predicted distances of test image

Figure 4.2: Slanted water distances of test image.

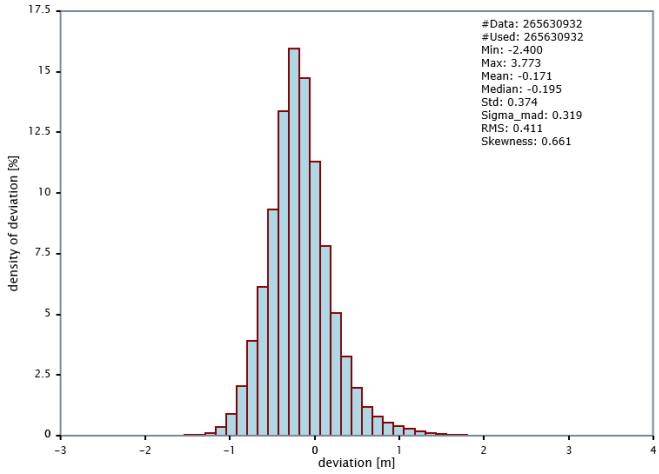


Figure 4.3: Histogram of deviations of predicted under-water distances compared to reference distances for 24 test images.

The histogram is showing an offset of one to two decimeters in negative direction, meaning that the predicted distances are larger than the reference (i.e. over estimation of water depth). It is nearly normally distributed with a median absolute deviation of 31.9 cm. The standard deviation is higher but the value is not as robust, considering outliers.

In Figure 4.4 a heatmap is showing the distribution of predicted under-water distances relating to the reference distances. The deviations enlarge with increasing distances. Furthermore, for reference distances near zero the net tends to predict longer distances. This partly might be ascribed to shaded areas at the shore line, where darkness implies greater depths. A common issue for aerial images containing water is sunglint caused by sunlight directly reflected into the sensor's field of view. This is producing bright spots in the images, in which there is no possibility to extract features from the bottom of

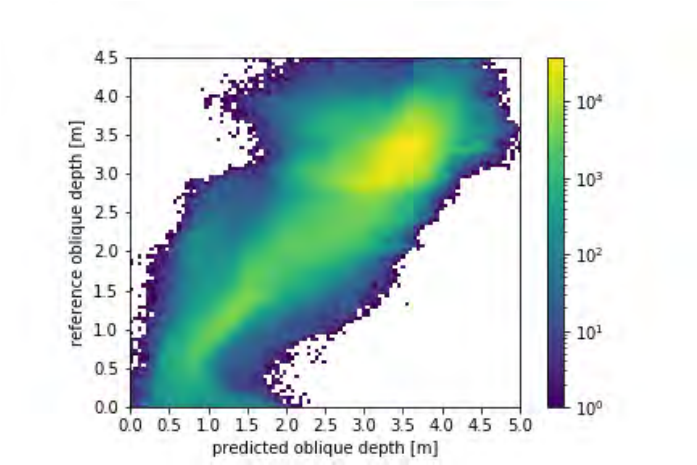
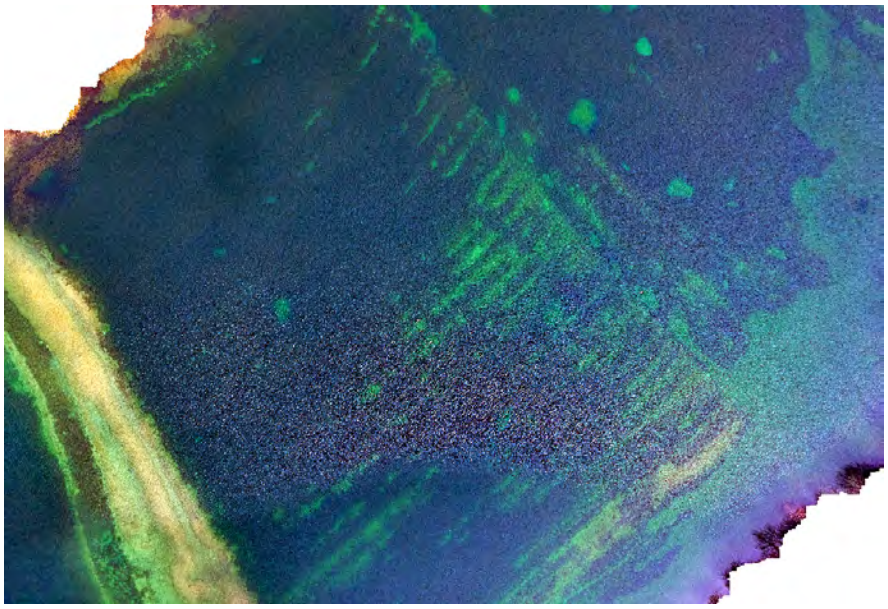
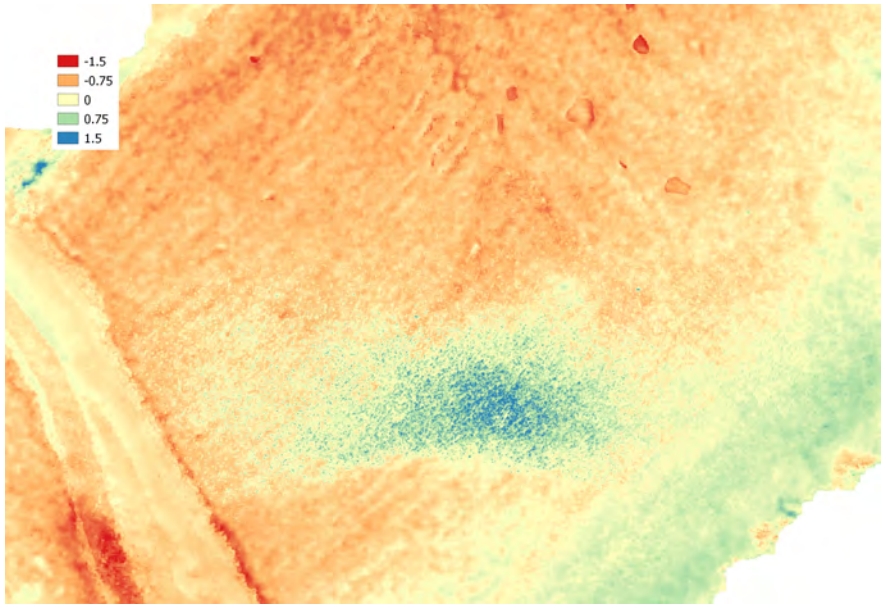


Figure 4.4: Heatmap of predicted and reference under-water distances.

the water. For approaches that benefit from spectral information this can cause large errors (Lyzenga et al., 2006). In Figure 4.5a an excerpt of a test image containing sunglint is shown. The deviation image for the predicted slanted under-water distances is shown in Figure 4.5b, in which large deviations up to 3 meters are occurring especially in the sunglint area. The slanted under-water distances in this area are predicted as close to zero because of the learned inverse proportionality of brightness to depth. Therefore the deviations are dependent on the reference distance for the particular pixels, which also is an explanation for the high maximum deviation and the positive skewness in Figure 4.3. This behavior is also reflected in the heatmap as larger deviations from the diagonal towards smaller predicted distances.
Furthermore, on the lower right of the image in Figure 4.5 a small area with unmasked vegetation above the water surface, as well as resulting shadows are observable, which are also causing errors. When examining the smaller bright patches on the upper right, a weakness of CNNs becomes apparent.



(a) RGB image



(b) Deviation image

Figure 4.5: Sunglint example test image.

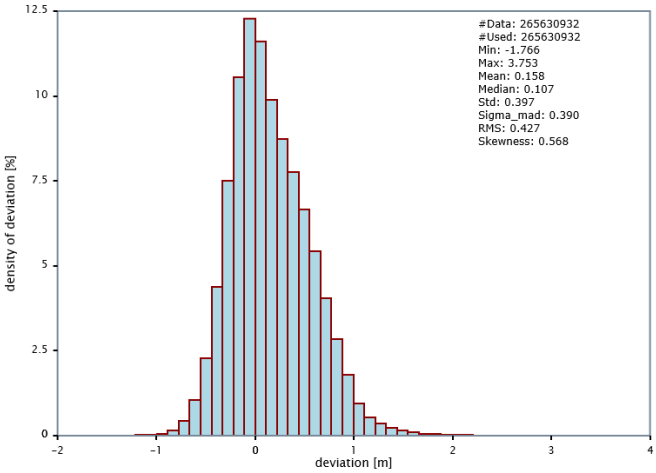


Figure 4.6: Histogram of deviations of predicted under-water distances without usage of the Coastal Blue band compared to reference distances.

While convolution kernels are taking information from surrounding pixels into account, they tend to blur strong edges. Thus, large deviations can be found at transitions from vegetation (dark) to bare soil (bright).

4.2 Applied CNN for RGB without Coastal Blue band

In order to evaluate the added-value of the additional Coastal Blue band for CNN-based deoth estimation, the net was also trained on the RGB images only. Looking at the difference histogram in Figure 4.6, this version seems to have a smaller positive bias, but a higher standard deviation, median absolute deviation, and root mean square compared to the histogram in Figure 4.3.

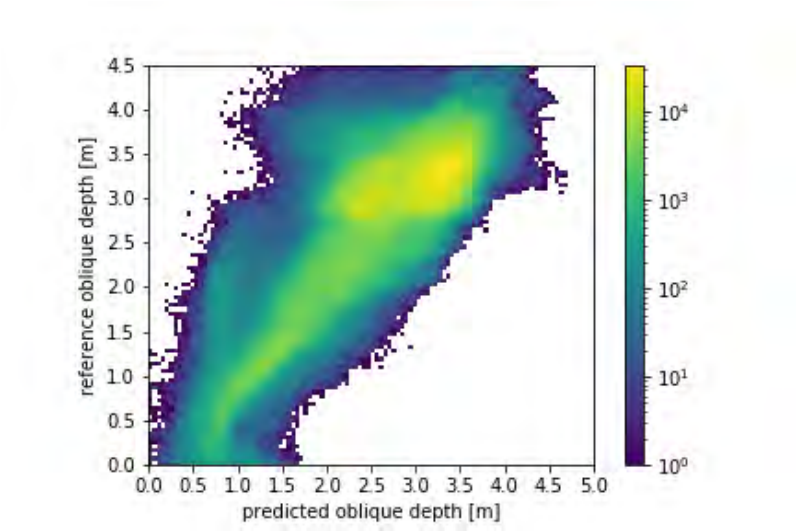
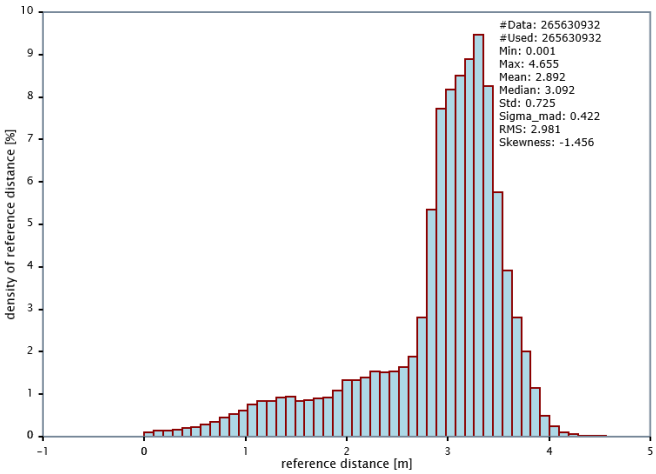


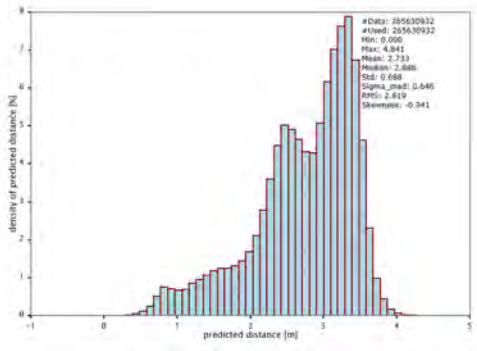
Figure 4.7: Heatmap of predicted and reference under-water distances without usage of Coastal Blue band.

4.3 Comparison

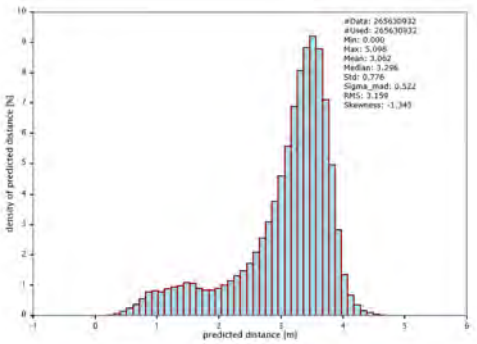
Figure 4.7 shows a heatmap of the distribution of predicted under-water distances derived without using the Coastal Blue band in relation to the reference distances. The deviations show a similar behavior to the results including the Coastal Blue band (Figure 4.4). However, for greater distances the heatmap is having two isolated bright areas. This is an indication that the Coastal Blue band provides useful information especially in deeper areas.
Also, when comparing the distribution of the under-water reference distances in Figure 4.8a to both approaches, the trend of the histogram with the Coastal Blue band (Figure 4.8c) seems to adapt better than the one without use of the Coastal Blue band (Figure 4.8b).



(a) Reference distances



(b) Predicted distances without Coastal Blue band



(c) Predicted distances with Coastal Blue band

Figure 4.8: Histograms of predicted distances of test images compared to reference distances.

5 Conclusion and Outlook

Considering that the area in the test images is unknown to the model, the predictions are consistent. If the desired accuracy has to be more precise than decimeter range, the method of choice would still be SoNAR or LiDAR. If not, advantages of CNN based bathymetry estimation over the stereo photogrammetric and linear regression approach are shown in this thesis. Because of the different ground covers of the lake, a more complex model than linear regression is required. Besides, looking back on the photogrammetric approach from Mulsow et al. (2019), the result is much smoother for homogeneous areas. It is to say that there are multiple possibilities for improvement. For example vegetation above the water surface as well as sunglint areas are causing major errors, which are having an effect on both, training and testing. If this is taken into account, the results should be more precise.
A common issue when trying to predict features is the lack of data. The major advantage from remaining in the image system instead of projecting into a global system is shown here. It results in the possibility of using the whole dataset with all overlapping areas without reducing it. This also is the reason, why it was possible to reject the images that covered both, the training and testing area, so that there was no connection.

To see how well the net is performing for alike datasets it is reasonable to

apply or transfer it to another lake or shallow waterbody. Ideally, the net can be used without any changes. Otherwise the pre-trained weights could be adapted by training with new reference under-water distances. In that case less training data should be necessary. When thinking about the advantages in terms of effort it would furthermore be interesting to see how well it performs for satellite imagery, probably after applying atmospheric corrections.
Proceeding with this method, the next logical step would be to derive a 3D point cloud from the predicted under-water distances in the images. For this purpose, an estimated water surface model and the orientations of the camera for each slanted distance image would be required. By doing so, it is possible to analyze the overlapping areas of consecutive images, to see how well they fit. Furthermore, outliers that only occur in single images, for example because of sunglint, could be rejected by calculating the median in a certain area when creating a DTM. This could avoid the necessity of introducing further postprocessing steps to mask out sunglint.
Another consideration is to introduce the second class "non water" into the training to avoid the currently necessary preprocessing step of masking out dry land areas. This was not done here, because the availability of reference data for each image automatically provided a non-water classification as side product. But especially when transferring it to another waterbody without reference data this should be implemented as well, especially considering that the water area would have to be segmented for all images.
Since, at least for lakes, a maximum depth is often known, another approach would be to use the pre-trained net as it is. The distances could then be determined relatively and afterwards scaled on the maximum depth. If a good accuracy could be achieved, it would be possible to derive depth models for lakes without using opulent methods like laser scanning or sonar.

Bibliography

Abadi, Martín et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: `https://www.tensorflow.org/` (visited on 09/15/2019).

Agarwal, Anubhav, CV Jawahar, and PJ Narayanan (2005). "A survey of planar homography estimation techniques." In: *Centre for Visual Information Technology, Tech. Rep. IIIT/TR/2005/12*.

Chollet, François et al. (2015). *Keras: The Python Deep Learning library*. `https://keras.io`. (Visited on 09/15/2019).

Costa, BM, TA Battista, and SJ Pittman (2009). "Comparative evaluation of airborne LiDAR and ship-based multibeam SoNAR bathymetry and intensity for mapping coral reef ecosystems." In: *Remote Sensing of Environment* 113.5, pp. 1082–1100.

Doneus, Michael et al. (2015). "Airborne laser bathymetry for documentation of submerged archaeological sites in shallow water." In: *Underwater 3D Recording and Modeling (ISPRS TC V, CIPA)*. Vol. 40. ISPRS, pp. 99–107.

Dumoulin, Vincent and Francesco Visin (2016). "A guide to convolution arithmetic for deep learning." In: *arXiv preprint arXiv:1603.07285*.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep learning*. MIT press.

Irish, Jennifer L and TE White (1998). "Coastal engineering applications of high-resolution lidar bathymetry." In: *Coastal engineering* 35.1-2, pp. 47–71.

Kotowski, Rüdiger (1988). "Phototriangulation in multi-media photogrammetry." In: *International Archives of Photogrammetry and Remote Sensing* 27.B5, pp. 324–334.

Kraus, Karl and Peter Waldhäusl (1996). *Photogrammetrie, Band 2, Verfeinerte Methoden und Anwendungen*. Dümmler.

Legleiter, Carl J, Dar A Roberts, and Rick L Lawrence (2009). "Spectrally based remote sensing of river bathymetry." In: *Earth Surface Processes and Landforms* 34.8, pp. 1039–1059.

Lyzenga, D. R., N. P. Malinas, and F. J. Tanis (Aug. 2006). "Multispectral bathymetry using a simple physically based algorithm." In: *IEEE Transactions on Geoscience and Remote Sensing* 44.8, pp. 2251–2259. ISSN: 0196-2892. DOI: 10.1109/TGRS.2006.872909.

Mandlburger, G et al. (2018). "INVESTIGATING THE USE OF COASTAL BLUE IMAGERY FOR BATHYMETRIC MAPPING OF INLAND WATER BODIES." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*.

Masnadi-Shirazi, Mohammed A et al. (1992). "Differential phase estimation with the SeaMARCII bathymetric sidescan sonar system." In: *IEEE Journal of Oceanic Engineering* 17.3, pp. 239–251.

MATLAB (2018). *version 9.5.0 (R2018b)*. Natick, Massachusetts: The MathWorks Inc.

Mulsow, Christian et al. (2019). "Vergleich von Bathymetriedaten aus luftgestützter Laserscanner-und Kameraerfassung." In:

Nielsen, Michael A (2015). *Neural networks and deep learning*. Vol. 25. Determination press San Francisco, CA, USA:

Nwankpa, Chigozie et al. (2018). "Activation functions: Comparison of trends in practice and research for deep learning." In: *arXiv preprint arXiv:1811.03378*.

OPALS (2019). *OPALS - Orientation and Processing of Airborne Laser Scanning data*. URL: https://opals.geo.tuwien.ac.at/html/stable/index.html (visited on 08/30/2019).

Pfeifer, Norbert et al. (2014). "OPALS–A framework for Airborne Laser Scanning data analysis." In: *Computers, Environment and Urban Systems* 45, pp. 125–136.

Riegl (2019). *RIEGL VQ-880-G: Fully Integrated Topo-Hydrographic Airborne Laser Scanning System*. URL: http://www.riegl.com/uploads/tx_pxpriegldownloads/Infosheet_VQ-880-G_2016-05-23.pdf (visited on 10/13/2019).

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-net: Convolutional networks for biomedical image segmentation." In: *International Conference on Medical image computing and computer-assisted intervention*. Springer, pp. 234–241.

Stanford University (2019). *CS231n Convolutional Neural Networks for Visual Recognition*. URL: http://cs231n.github.io/ (visited on 08/25/2019).

Wang, Yanhong et al. (2019). "Bathymetry Model Based on Spectral and Spatial Multifeatures of Remote Sensing Image." In: *IEEE Geoscience and Remote Sensing Letters*.

Hannes Nübel wurde für seine Bachelorarbeit, die er am Institut für Photogrammetrie der Universität Stuttgart verfasst hat, mit dem DHyG Student Excellence Award 2020 ausgezeichnet.
In seiner Arbeit beschäftigte er sich damit, wie sich Gewässertiefen aus multispektralen Bildern mit Hilfe von Convolutional Neural Networks bestimmen lassen. Insbesondere bei hochauflösenden Weitwinkel-Luftbildern kommt es darauf an, die Strahlengeometrie und die Brechungskorrektur streng zu berücksichtigen. Durch den Einsatz von Deep Learning lassen sich komplexe Zusammenhänge zwischen Bildradiometrie und Gewässertiefe modellieren.