remote sensing

MDPI

Article

Deep Learning and Adaptive Graph-Based Growing Contours for Agricultural Field Extraction

Matthias P. Wagner * and Natascha Oppelt

Earth Observation and Modelling, Department of Geography, Kiel University, 24118 Kiel, Germany; oppelt@geographie.uni-kiel.de

* Correspondence: m.p.wagner@gmx.de; Tel.: +49-176-5788-6714

Received: 20 May 2020; Accepted: 18 June 2020; Published: 21 June 2020

Abstract: Field mapping and information on agricultural landscapes is of increasing importance for many applications. Monitoring schemes and national cadasters provide a rich source of information but their maintenance and regular updating is costly and labor-intensive. Automatized mapping of fields based on remote sensing imagery may aid in this task and allow for a faster and more regular observation. Although remote sensing has seen extensive use in agricultural research topics, such as plant health monitoring, crop type classification, yield prediction, and irrigation, field delineation and extraction has seen comparatively little research interest. In this study, we present a field boundary detection technique based on deep learning and a variety of image features, and combine it with the graph-based growing contours (GGC) method to extract agricultural fields in a study area in northern Germany. The boundary detection step only requires red, green, and blue (RGB) data and is therefore largely independent of the sensor used. We compare different image features based on color and luminosity information and evaluate their usefulness for the task of field boundary detection. A model based on texture metrics, gradient information, Hessian matrix eigenvalues, and local statistics showed good results with accuracies up to 88.2%, an area under the ROC curve (AUC) of up to 0.94, and F_1 score of up to 0.88. The exclusive use of these universal image features may also facilitate transferability to other regions. We further present modifications to the GGC method intended to aid in upscaling of the method through process acceleration with a minimal effect on results. We combined the boundary detection results with the GGC method for field polygon extraction. Results were promising, with the new GGC version performing similarly or better than the original version while experiencing an acceleration of 1.3× to 2.3× on different subsets and input complexities. Further research may explore other applications of the GGC method outside agricultural remote sensing and field extraction.

Keywords: field extraction; field boundary detection; deep learning; multilayer perceptron; active contours; growing snakes; graph-based growing contours

1. Introduction

Field mapping is a topic of increasing relevance. Knowledge of field sizes, shapes, and distributions is valuable for a wide variety of topics, ranging from precision agriculture to biodiversity mapping, agricultural policy, yield estimation, and cropland classification [1–3]. Today, monitoring is mostly based on field campaigns, administrative maps, and manual airborne or satellite imagery interpretation [1]. However, such approaches are time-consuming and costly, and require manual interpretation. Regularly updated and standardized national cadasters, such as the Land Parcel Identification Systems or the Integrated Administration and Control Systems promoted by the European Union, can mitigate this but are still prone to subjective interpretation in the mapping process and difficulty of evaluating information [1,4–6]. Furthermore, financial and labor limitations often hamper timely updates.

Remote sensing promises to alleviate these issues through regular large-scale observations of agricultural landscapes and much lower cost thanks to increasing amounts of freely accessible data. As a result, earth observation data has seen extensive use in agricultural applications over the years, covering topics such as crop type classification, plant health monitoring and stress detection [7–12]). The topic of field boundary detection and field extraction, however, has received relatively little interest so far.

Field boundary detection is commonly tackled through edge detection, clustering, and segmentation techniques, as well as machine learning classifications such as random forest [13–16]. Other approaches include the use of image characteristics such as texture and statistical modeling [2,17]. Frequently, separation of individual fields is performed via segmentation of pixels or clusters of pixels, ultimately limiting the accuracy of extracted fields.

Simultaneously, artificial intelligence methods, such as artificial neural networks (NNs), have frequently proven their merits in the context of edge detection, classification, and segmentation [12,18–20]. Recently, research has focused on convolutional NNs that were also applied to field boundary detection [21–24]. Neural networks have the advantage of being capable of extracting highly non-linear relationships and have proven to be highly capable in different classification tasks. However, a drawback often noted is the difficulty of transferring results obtained in one region or on a specific dataset to another.

In a previous publication, we presented a full workflow combining agricultural field boundary detection and polygon extraction steps [25]. Starting from image pre-processing and enhancement, we combined edge detection, contour detection, and, finally, field polygon extraction. We used a newly developed, modified growing contours method, called graph-based growing contours (GGC), to extract complex networks of contours present in agricultural field boundaries. This method is flexible regarding the input information used and proved to be promising in extracting field boundaries, even in rather complex environments.

However, this process is naturally dependent on the quality of the field boundary map used as an input. Despite extensive pre-processing and image enhancing, contours in the previous study were often heterogeneous in strength, and it proved to be difficult to create a reliable boundary map for the subsequent field polygon extraction steps. This necessitated optimization of local parameter settings to achieve comparable results in different parts of an image. Further, we had to resort to small step sizes in the extraction procedure, resulting in longer processing time and hampering scale-up to larger scenes.

To address these drawbacks and enable large-scale applications of our methodology, in this study we explore an improved boundary detection approach. We decided to use multilayer perceptron neural networks (MLP-NNs) and aimed to develop a structure with a high potential of performing well under different environments and in different locations [26]. Following an extensive literature review, we selected a large number of potentially useful input features for successful agricultural boundary detection using MLP-NNs. We explored the usefulness of different features based on various image characteristics and developed a boundary detection model based on study areas located in the east of the state of Schleswig-Holstein, Germany. By limiting our analysis only to the red, green, and blue (RGB) bands of Sentinel-2 (S-2), we wanted to demonstrate the use of only the most basic imagery, making our approach largely independent of the sensor.

We further introduce an improved version of the previously presented GGC algorithm with scalability in mind. We apply both versions of the GGC algorithm to the obtained boundary maps and analyze results of the boundary detection and the extracted polygons, as well as processing speed of the different GGC algorithms.

The paper is structured as follows. In Section 2, we describe the study area and the data we used. Section 3 presents the methodology, including image feature preparation, input selection, and model training. Section 4 shows results of the boundary detection and performance of subsequent polygon extraction. Section 5 discusses results and Section 6 summarizes the most important findings.

2. Data and Materials

2.1. Study Areas

We obtained data from two study areas in the state of Schleswig-Holstein, Germany (see Figure 1). Both were predominantly used for agricultural purposes. Field sizes were rather heterogeneous in both regions, ranging from below 1 ha up to about 75 ha. Fields, especially in study area 1, were often irregular in shape. The landscape structure is a mixture of agricultural areas with small forests, grasslands, and smaller urban agglomerations.

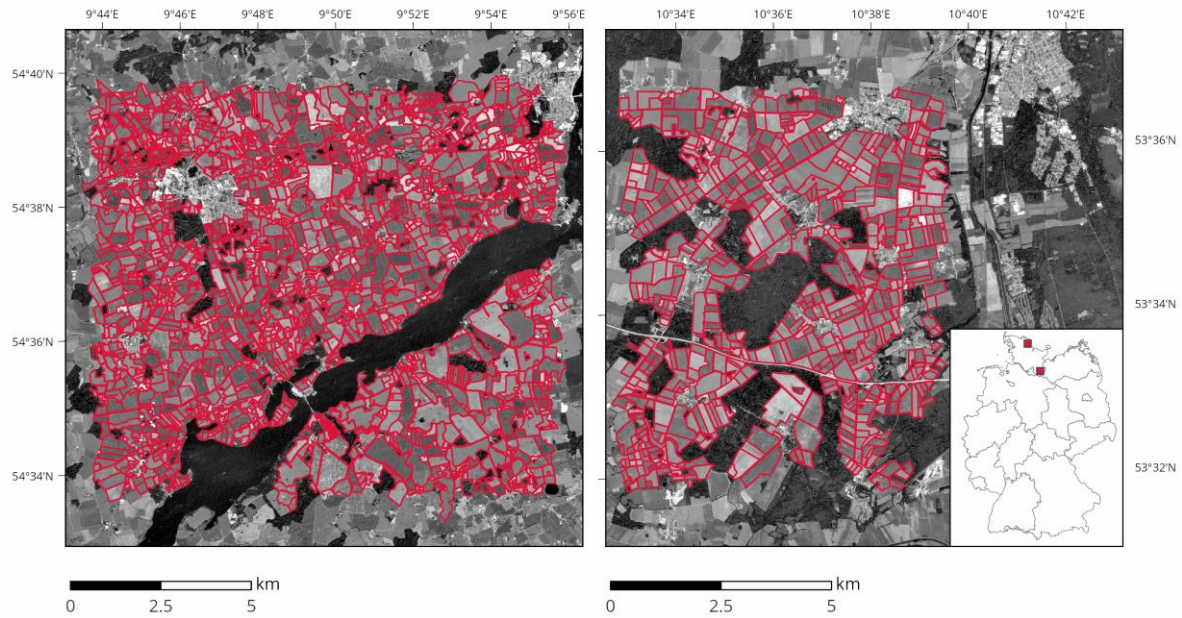

Figure 1. Maps of study areas 1 (**left**) and 2 (**right**).

The study areas are located in the so-called "Schleswig-Holsteinisches Hügelland", a hilly but low-elevation region along the coast of the Baltic Sea in northern Germany that forms one of three large landscape zones in the state of Schleswig-Holstein. The climate is temperate/oceanic with warm summers and wet winters (Cfb in the Koeppen–Geiger climate classification) [27].

Farming in the region is rain-fed and highly industrialized. Common agricultural crops include cereals such as wheat and barley but also maize and rapeseed [28].

2.2. Satellite Imagery

We used Sentinel-2 Level-2A atmospherically corrected imagery. For each study area, we considered observations over the 2019 growing season (March to October) and selected those with no cloud cover over the respective study area. We considered three observation times in early (March), mid (June), and late (September) growing season. To allow for high spatial detail and independence of very specific wavelength bands, we used only 10-m-resolution red, green, and blue (RGB) bands.

Pre-processing included filtering of each band to reduce noise and remove undesired features. Considering our interest in field boundaries, we opted for using bilateral filtering rather than traditional Gaussian smoothing. The bilateral filter is an edge-preserving filter balancing similarity in the spatial and spectral domains to smooth the image while still conserving strong edges [29]. The filtered images were then transformed from RGB to CIELAB color space to separately observe information on luminosity (l) and color (a, b) in the images.

2.3. Reference Data

We used cadastral data as a reference [30]. This data, however, represented purely administrative borders and needed manual updating based on field surveys and recent satellite imagery to accurately represent conditions in 2019. We omitted fields that were not clearly discernable in S-2 imagery, e.g., very small or thin and elongated fields. Table 1 contains some general information about the study areas.

Table 1. Field statistics in the two regions of interest based on the updated reference datasets.

Study Area	Total Extent	Field Sizes	Field Count
1	~13.5 × 12.4 km^2	~0.9 to 75 ha	~1400
2	~9.2 × 7.5 km^2	~0.8 to 50 ha	~600

3. Methodology

The flowchart in Figure 2 demonstrates the structure of our methodology. The pre-processing (green) was discussed in Section 2.2. In this chapter, we start with describing the image features we obtained in Section 3.1 (blue), followed by the boundary detection model development, including feature selection and hyperparameter tuning (yellow), in Section 3.2. Finally, we describe the modified GGC method and the field polygon extraction process (grey) in Section 3.3.

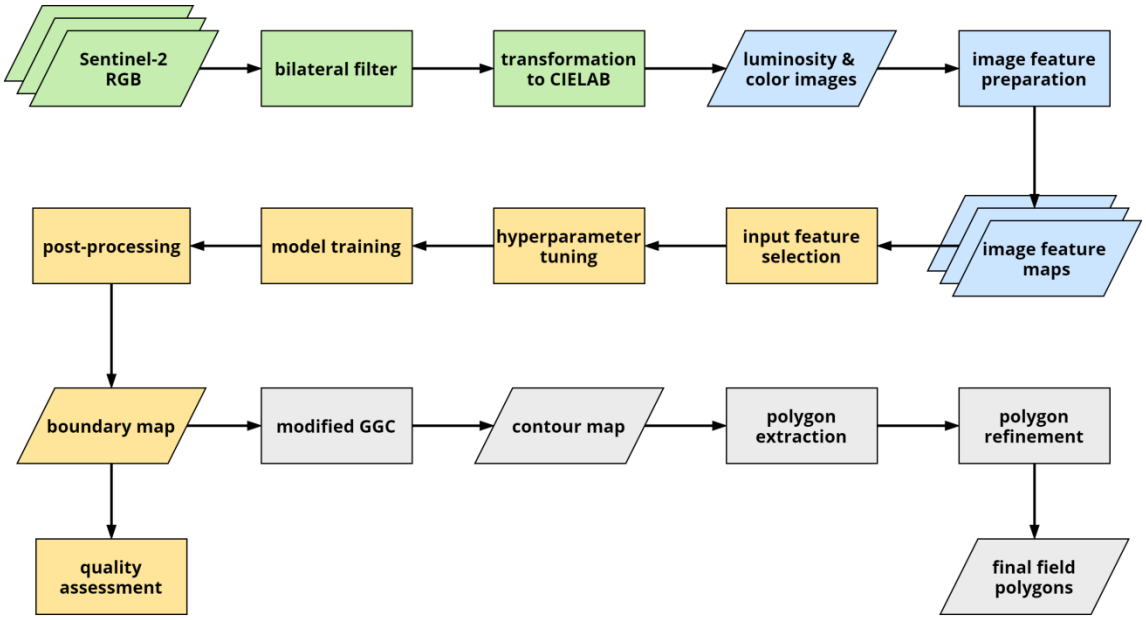

Figure 2. Flowchart of the methodology. Different stages highlighted in colors: pre-processing in green, image feature preparation in blue, boundary detection model development in yellow, and contour detection and field polygon extraction in grey.

3.1. Image Feature Preparation

We undertook an extensive literature review on topics of edge detection, boundary detection, and segmentation, and selected a set of potentially useful input features for field boundary detection. We distinguished applications on the luminosity (l) and the color channels (a, b) in the CIELAB color space. Table 2 provides an overview of all inputs we considered. In the following sub-sections, we briefly describe the concepts behind the features and how we obtained them. In the text, we also refer to features and feature groups by their numbers in Table 2 as F1, F2, G1, G2, etc. All inputs were scaled linearly to a range of 0 to 1 based on the overall minima and maxima observed.

Table 2. Overview of features and respective feature groups considered for boundary detection.

Feature Group		Feature	
G1	image gradient (l)	F1	gradient magnitude (l)
G2	image gradient (a, b)	F2	gradient magnitude (a, b)
G3	local statistics (l)	F3	variance (l)
		F4	skewness (l)
		F5	kurtosis (l)
G4	local statistics (a, b)	F6	variance (a, b)
		F7	skewness (a, b)
		F8	kurtosis (a, b)
G5	Hessian matrix (l)	F9	eigenvalue along ridge (l)
		F10	eigenvalue across ridge (l)
G6	Hessian matrix (a, b)	F11	eigenvalue along ridge (a, b)
		F12	eigenvalue across ridge (a, b)
G7	texture metrics (l)	F13	contrast (l)
		F14	correlation (l)
		F15	asm (l)
		F16	homogeneity (l)
G8	texture metrics (a, b)	F17	contrast (a, b)
		F18	correlation (a, b)
		F19	asm (a, b)
		F20	homogeneity (a, b)
G9	angular dispersion (l)	F21	modified angular dispersion (l)
G10	angular dispersion (a, b)	F22	modified angular dispersion (a, b)
G11	homogeneity measures (l)	F23	PEG (l)
		F24	QEG (l)
G12	homogeneity measures (a, b)	F25	PEG (a, b)
		F26	QEG (a, b)
G13	local cues (l)	F27	brightness gradient (l)
		F28	texture gradient (l)
G14	local cues color (a, b)	F29	color gradient (a, b)

3.1.1. Image Gradient

The image gradient is the first derivative of the image and highlights discontinuities. To obtain gradient magnitude (F1, F2), we first applied the Sobel operator in both x- and y-directions:

$$K_x = \begin{Bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{Bmatrix}, \quad K_y = \begin{Bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{Bmatrix} \tag{1}$$

We then obtained gradient magnitude g based on the following formula:

$$g = \sqrt{I_x^2 + I_y^2} \tag{2}$$

where I_x and I_y are the gradient images in x- and y-direction. For the application on color layers (F2), we summed up the individual gradient images I_x and I_y before calculating magnitude and direction from the result.

3.1.2. Local Statistics

We calculated local statistics for each pixel based on a 5 × 5 kernel neighborhood (F3-F8). We considered three statistical measures: variance and the third and fourth standardized moments, skewness and kurtosis [31].

The variance is defined as the spread of values in a (normal) distribution:

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2 \tag{3}$$

where $x_i, \ldots, x_n$ are the sampled values, μ is the mean of the sample, and N is the sample size.

Skewness refers to the asymmetry of a distribution. A negative skewness means the tail of the distribution to the left is longer and the distribution "leans to the right"; a positive skewness represents a longer tail to the right and the distribution "leans to the left". We used the Fisher–Pearson coefficient formula to obtain skewness in the local sample [31]:

$$\widetilde{\mu}_3 = \frac{\sum_{i=1}^{N}\frac{(x_i-\mu)^3}{N}}{\sigma^3} \tag{4}$$

where σ is the standard deviation.

Kurtosis describes the shape of a distribution in terms of its "curvedness", i.e., how convex or concave the shape of the underlying distribution is. We used the following formula to calculate kurtosis [31]:

$$\widetilde{\mu}_3 = \frac{\sum_{i=1}^{N}\frac{(x_i-\mu)^4}{N}}{\sigma^4} \tag{5}$$

In the case of application to color channels, we considered the maximum of the two results (F6-F8).

3.1.3. Hessian Matrix

Following concepts described in Meijering et al., Ando, and Sato et al., we used the Hessian matrix of the image to obtain information on ridge-like structure strength and orientation [32–34].

The Hessian matrix of a 2D image $I(x)$ is given by:

$$\nabla^2 I(x) = \begin{bmatrix} \frac{\partial^2}{\partial x^2} & \frac{\partial^2}{\partial x \partial y} \\ \frac{\partial^2}{\partial y \partial x} & \frac{\partial^2}{\partial y^2} \end{bmatrix} \tag{6}$$

where $\frac{\partial^2}{\partial x^2}$, etc. are the partial second derivatives of the image I. Following the descriptions by Sato et al., we obtained the eigenvectors $e_1(x)$ and $e_2(x)$ of $\nabla^2 I(x)$ and their corresponding eigenvalues λ_1 and λ_2 [34]. The eigenvector with the higher eigenvalue represents the direction in which the second derivative is maximal. In the case of a line-like structure, the eigenvalue across the line (or across ridge) is high and the eigenvalue along the line is low.

We use the eigenvalues along and across the ridge (F9–F12) as potential inputs to the model.

3.1.4. Second-Order texture Metrics

The concept of the grey-level co-occurrence matrix (GLCM) and the derived second-order texture metrics was first introduced by Haralick et al. as a means to quantify the texture of imagery for classification [35]. The GLCM refers to a matrix capturing the spatial relationship of values in a raster

by counting how often two values occur together ("co-occurrence") considering different directions (horizontal, vertical, two diagonals) and different distances (steps apart).

At first, a quantization is performed in which the image values are assigned to a fixed number of gray levels. We performed quantization error minimization to reduce the data to 32 grey levels. We considered four second-order texture metrics: contrast, correlation, angular second momentum (ASM), and homogeneity ("inverse difference moment" in the original paper) (F13–F20):

$$Contrast = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} P_{i,j} \, (i-j)^2 \tag{7}$$

$$Correlation = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} P_{i,j} \left[\frac{(i-\mu_i)\left(j-\mu_j\right)}{\sqrt{\left(\sigma_i^2\right)\left(\sigma_j^2\right)}} \right] \tag{8}$$

$$ASM = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} P_{i,j}^2 \tag{9}$$

$$Homogeneity = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{P_{i,j}}{1+(i-j)^2} \tag{10}$$

where i and j are two quantization values (or coordinates in the GLCM), $P_{i,j}$ is the probability value of the cell (i, j), N is the number of columns and rows in the GLCM (i.e., the number of grey levels), and μ_i, μ_j, σ_i, σ_j are the means and standard deviations of the marginal distributions of $P_{i,j}$.

We considered all four possible angles (0°, 45°, 90° and 135°) with a maximum of two steps in the 5×5 neighborhood. Final texture metrics for each pixel were obtained as the mean of all outputs of the process. In case of color layers, we took the mean of the resulting metrics in the two channels.

3.1.5. Angular Dispersion

Following the concept of angular dispersion introduced by Gregson and the fractional anisotropy described in Aganj et al., we calculated a modified angular dispersion metric similar to the local anisotropy we used for seed point selection in our previous study [25,36,37]. The concept is based on the observation that gradient direction tends to be homogeneous along edges or boundaries in an image while they are heterogeneous ("dispersing") if no clear boundary is present or boundary directions are changing, for example, at crossings of multiple edges or along sharp curves.

To approximate this, we consider the mean direction of gradient angles and the normal to it. By projecting all gradient direction vectors onto the two main directions, we calculate angular dispersion as follows:

$$I_{disp} = 1 - \left(\frac{\min\left(\sum_0^I \| proj_{v_{mean}} v_i \|, \ \sum_0^I \| proj_{v_{norm}} v_i \| \right)}{\max\left(\sum_0^I \| proj_{v_{mean}} v_i \|, \ \sum_0^I \| proj_{v_{norm}} v_i \| \right)} \right) \tag{11}$$

where v_{mean} and v_{norm} represent the mean directional and normal vectors, v_i describes a given local gradient direction vector, and I is the number of all sampled vectors. If the total magnitude of vectors projected in the mean and the normal direction is identical, we obtain a value of 0 (low dispersion); if the two total magnitudes differ largely, we obtain a value close to 1 (high dispersion).

We consider gradient directions in a 3×3 pixel neighborhood (F21-F22).

3.1.6. Homogeneity Measures

Ando presented an approach for edge and corner detection based on gradient covariance [33]. The cross-correlation matrix is given as:

$$\begin{bmatrix} f_x \\ f_y \end{bmatrix} \begin{bmatrix} f_x & f_y \end{bmatrix} = \begin{bmatrix} S_{xx} & S_{xy} \\ S_{yx} & S_{yy} \end{bmatrix} \tag{12}$$

where S_{xx}, S_{xy}, etc. refer to the cross-correlations of the image derivatives f_x and f_y. Based on these cross-correlations, he defines two homogeneity measures:

$$P_{EG} = \frac{\left(S_{xx} - S_{yy}\right)^2 + 4S_{xy}^2}{\left(S_{xx} + S_{yy}\right)^2 + \sigma_{EG}^4} \tag{13}$$

$$Q_{EG} = \frac{4\left(S_{xx}S_{yy} - S_{xy}^2\right)}{\left(S_{xx} + S_{yy}\right)^2 + \sigma_{EG}^4} \tag{14}$$

with σ_{EG}^4 being a small constant to avoid division by zero at completely flat areas in the image. The two measures show different behaviors: while P_{EG} reaches 1 where grayness varies one-dimensionally, i.e., near edges or ridges in the image, Q_{EG} reaches 1 where grayness variation is omnidirectional, i.e., at centers of circular symmetry or corners (F23–F26). For color images, we used the maximum of the two color results.

3.1.7. Local Cues

Martin et al. presented a set of local cues based on brightness, color, and texture information to observe boundaries in images [38]. Their reasoning is that different types of boundaries exist in images. Traditional image gradient approaches only observe sharp changes in image intensities, so their techniques aim to provide gradient information on color and texture as well (F27–F29).

They achieve this by transforming RGB images to the CIELAB color space and separating the luminosity (l) from the color bands (a, b). To obtain local gradient information, luminosity values in a local disk centered on each pixel and divided into two halves along a diameter are sampled. A dissimilarity between the two halves indicates the presence of a strong gradient and thereby a possible boundary. First, histograms of values in each disk half are created by sampling a kernel density estimate of the sampled values. The gradient is then obtained as the χ^2 distance between the two histograms:

$$\chi^2(P,\, Q) = \frac{1}{2} \sum_{i=1}^{N} \frac{(p_i - q_i)^2}{p_i + q_i} \tag{15}$$

with P and Q referring to the two distributions and p_i and q_i being the sampled probabilities. Local color gradients are obtained in the same fashion by applying the same procedure to both color channels and then adding the two marginal gradients to a joint gradient.

In the case of texture, the approach is based on what the authors refer to as "textons". At first, a filter bank of 13 different filters is applied to the neighborhood of each pixel. The filters contain a difference of Gaussian filters, as well as a set of elongated, oriented even- and odd-symmetric Gaussian second derivative filters. The responses of all filters are then clustered via k-means to obtain the so-called textons. The rest of the procedure is similar to the brightness and color gradients: textons are assigned to bins in a histogram and the local texture gradient is derived as the χ^2 distance between the two histograms of the disk halves.

In all three cases (brightness, color, and texture) the final gradient is obtained by taking the maximum gradient observed in 16 orientations in the range $[-\pi, \pi]$. As we did not train a separate

classifier as in the original paper, we computed textons on individual image subsets. Due to the limited amount of data, we observed that a reduced number of 32 clusters (versus 64 in the original paper) was preferable.

3.2. Boundary Detection

3.2.1. Dataset Preparation

To create the training data, we considered areas in proximity to our reference data as a possible input. Boundary samples were taken from the rasterized polygon boundaries of the reference data and non-boundary samples were considered from a distance of up to 8 pixels from the boundaries. We avoided selecting non-boundary samples in the vicinity of the boundaries within a two-pixel-wide buffer to enhance the sample set. Datasets were balanced to contain 50% boundary and 50% non-boundary samples and then randomly split into calibration (60%, 310,928 samples), validation (20%, 103,642 samples) and test (20%, 103,642 samples) sets.

3.2.2. Model Development Setup

We used fully-connected MLP-NNs for boundary detection. We considered the task of boundary detection as a binary classification problem, distinguishing any type of field boundary pixel as represented by the reference data from any type of non-boundary pixels. This results in a rather heterogeneous non-boundary class, as agricultural fields, nearby forests, urban agglomerations, streets, etc. may end up being selected as non-boundary examples. Using additional classes by distinguishing different kinds of land cover may have been advantageous in terms of classification accuracy but was not considered in order to achieve a result that can serve as a direct input to the subsequent field extraction steps. The output node was scaled using a sigmoid function.

We used the stochastic gradient descent optimizer (SGD) with momentum of 0.9 to train the model using the log loss function [39,40]:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log(p(y_i)) + (1-y_i) \cdot \log(1-p(y_i)) \quad (16)$$

where $p(y_i)$ is the predicted probability (output value between 0 and 1) and y_i is the class label of the given sample point (0 or 1). The smaller the resulting log loss value, the better the prediction. Theoretically, log loss can become infinitely large when the prediction is worse than a purely random guess.

3.2.3. Input Feature Selection

As seen in Table 2, we obtained a total of 29 input features. We categorized them into 14 groups based on concept (see Section 3.1) and on image information used (luminosity vs. color). For each feature, we considered values from the three observation times in April, June, and September.

Feature selection was performed through quality ranking. We first trained a default model of topology x-30-1 with each set of input feature groups for 10,000 epochs and ranked results from best (1) to worst (max) based on validation loss. The overall rank of each input feature group was obtained by summing up the ranks of all test runs involving the respective feature group. For example, if the test run ranked 1 was based on groups 2, 3, and 7, then the total score of all three groups would be increased by 1. As a result, we obtained an overall ranking that represents the relative performance of each group in all test runs with the lowest total rank number representing the best-performing group.

We started with single groups as inputs, then pairs, triplets, quadruplets, and, finally, quintuplets. At each stage, we dropped the two lowest ranking feature groups. Results are presented in Section 4.1.1 as relative importance, i.e., the ratio of the lowest total score achieved in the corresponding step to the scores of each group, resulting in importance scores between 0 and 1, where the best-performing group of each step achieves an importance of 1.

3.2.4. Hyperparameter Tuning and Model Training

We considered five hyperparameters for tuning: number of hidden layers, number of nodes per hidden layer, type of activation function in hidden layers, learning rate and dropout rate (random dropout probability) per hidden layer. Table 3 lists the considered settings.

Table 3. Hyperparameters considered (selected parameters in bold text).

Hyperparameter	Considered Values
Number of hidden layers	2, 3, **4**, 5
Number of nodes per hidden layer	15, 20, 25, 30, **35**, 40, 45, 50
Type of activation function	Sigmoid, **ReLU**
Learning rate	0.01, 0.001, **0.0001**, 0.00001
Dropout rate per hidden layer	20%, 30%, 40%, 50%, 60%, 70%, **80%**, 90%

We first performed a manual pre-testing to obtain some insights into the model behavior and the effects of different hyperparameter values to reduce the number of settings that need to be considered in the tuning stage. Our pre-testing showed that a single hidden layer was insufficient to obtain best results, although larger numbers of hidden layers beyond five also showed no advantage. Results for the number of hidden nodes were inconclusive, with no clear tendency towards larger or smaller numbers. Learning rates of 0.01 and below proved beneficial, as well as dropout rates of 20 % or higher, especially to avoid overfitting late in the training process.

We performed the tuning via a random search. We trained each model for 10,000 epochs and selected the one with the lowest validation loss as the parameter set for final model training. We trained and validated the model on the full calibration and validation datasets, respectively. The final model (four hidden layers of 35 hidden nodes each, rectified linear unit (ReLU) activation function, learning rate of 0.0001, and dropout rate of 80%) was trained for 100,000 epochs [41].

3.2.5. Boundary Map Post-Processing

To further improve quality of the output, we transformed the probability map into a binary boundary map by setting a threshold of 0.5. We further excluded non-agricultural areas using CORINE land cover data of 2018 and the Land Cover DE product of the German Aerospace Center (DLR) and highlighted the boundaries of those areas to allow for the contour extraction to detect them [42,43]. To further refine the boundaries, we calculated the boundary strength as the share of boundary pixels in a 3×3 pixel neighborhood. This helps guide the contour extraction towards the center of the boundaries (see Section 3.3).

3.2.6. Quality Assessment

For final model evaluation, we employed multiple accuracy metrics. First, we used sensitivity (true positive rate), specificity (true negative rate), accuracy, and F_1 score as general indicators of model performance, calculated as follows [39]:

$$Sensitivity = \frac{TP}{TP + FN} \tag{17}$$

$$Specificity = \frac{TN}{FP + TN} \tag{18}$$

$$Accuracy = \frac{TP + TN}{N} \tag{19}$$

$$F_1 = \frac{2\,TP}{2\,TP + FP + FN} \tag{20}$$

where *TP* is the number of samples correctly classified as positive (true positive), *TN* is the number of samples correctly classified as negative (true negative), *FP* is the number of samples falsely classified as positive (false positive), and *FN* is the number of those falsely classified as negative (false negative). *N* represents the whole sample size. All four metrics range from 0 to 1 and 0% to 100%, respectively. A score of 1 or 100% indicates a perfect result.

Further, we obtained the area under the receiver operating characteristic (ROC) curve as a probability-based indicator of model performance. The ROC curve represents the trade-off between the true positive rate and false positive rate by plotting the two measures against each other for multiple thresholds. The area under the curve (AUC), i.e., its integral, is used as an indicator for a model's capacity to separate two classes in a binary classification [39]. An AUC of 1 represents a perfect distinction, while an AUC of 0.5 indicates a purely random classification.

3.3. Field Extraction

3.3.1. The Graph-Based Growing Contours Method

For field polygon extraction, we employ an active contours algorithm based on growing contours (or snakes) that we presented recently [25,44]. Active contours in general were introduced for object delineation and are based on energy minimization [45]. An initial spline or polyline follows an external and an internal energy, representing information from the image (e.g., gradient magnitude or local intensities) and the shape of the contour. While the former guides the snake towards the contour of interest, the latter influences curvature and the size of the snake, preventing it from creating discontinuous shapes and collapsing or expanding indefinitely.

However, agricultural field boundaries are rather complex networks of contours that are sometimes only loosely interconnected and can be quite irregular in shape. Therefore, using shape priors or any previous assumptions may limit performance of an algorithm in different environments with varying characteristics (field sizes, shapes, locations, etc.).

The growing snakes concept introduced by Velasco and Marroquin addressed common drawbacks of many active contour models regarding sensitivity to initiation, i.e., initial position and shape priors, as well as difficulties in extracting discontinuous contours of many separate or highly irregularly shaped objects [44]. To achieve this, the growing snakes are initiated on seed points on the contour of interest and grow along it. As the growth follows local maxima, it tends to create irregular shapes and requires repeated smoothing.

In our applications to field boundaries, however, we observed that growing snakes tended to get "off track" easily. As they follow local maxima of the input image, any irregularities, especially near corner points or intersections of multiple boundaries, may easily move them away from the actual boundary. Further, we observed limitations in representing sharp turns.

Following these insights, we developed a modified growing contours model called graph-based growing contours (GGC). Our main goals were to (a) improve the quality of extracted contours and reduce the need for repeated smoothing; (b) introduce adaptive behavior allowing the tracing of contours of different shapes (straight lines, curves, sharp changes in direction, etc.); (c) enable automatic branching to explore multiple parts of a boundary network automatically and represent crossings as well as T-junctions; (d) reduce "dead ends"; and (e) extract an interconnected network rather than separate contour segments.

To achieve this, we replaced the growth step of the original growing snakes with a more sophisticated approach based on graph theory principles. Instead of simply searching for a local maximum in a set of possible steps, we generate a local graph by sampling around each current end point at each growth step. The GGC method works at sub-pixel accuracy. We used a barycentric transform to obtain values at a sub-pixel level [46].

The graph is defined by an inner radius r_{min}, an outer radius r_{max}, the number of vertices on the innermost circle n_i, and the number of evenly spaced circles n_c. The parameter n_e defines the number

of connections of each vertex to vertices on the next-larger circle. Starting from the innermost circle, the number of points doubles with each circle to obtain a homogeneous pattern, in which each vertex on a circle has the same number of neighbors on the next circle (see Figure 3). Vertices in the direction of the previous step are masked to avoid retracing of contours.

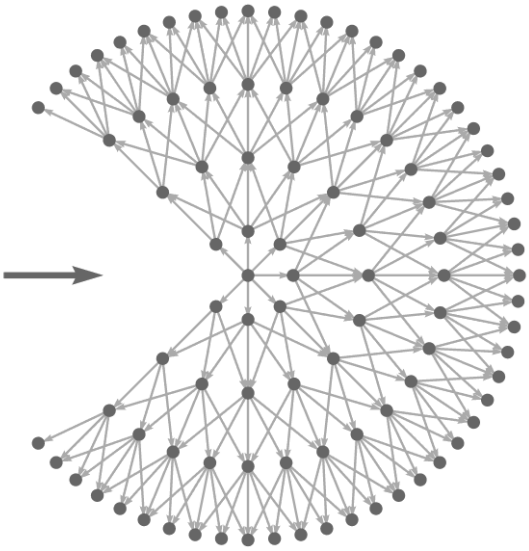

Figure 3. Example of local graph in graph-based growing contours. The large horizontal arrow indicates the direction of movement.

The sampled points are used as vertices in a directed, weighted graph $G = (V, E)$ as shown in Figure 3, where each edge (except for those originating at the center point) is weighted based on:

$$w_{i,j} = \frac{D_{i,\,j}}{g_j} \tag{21}$$

where $w_{i,j}$ is the weight of the edge connecting vertices v_i and v_j, $D_{i,\,j}$ is the Euclidian distance between the two vertices, and g_j is the gradient magnitude (or any other boundary strength indicator used) at vertex v_j. The resulting weights favor shorter paths (smaller Euclidian distance) and paths along boundaries in the image (higher boundary strength).

The different settings in constructing the graph (r_{min}, r_{max}, n_i, n_c, n_e) may be regarded as parameters for adjusting step size, precision, capacity to represent sharper directional changes, and complexity of the graph.

At each iteration, each current end point is explored for possible branches. Shortest path searches from the center vertex (source) to each vertex of the outermost circle (sinks) are initiated. Firstly, the single shortest path defines the first branch. The graph is then separated into two subsections at $+\frac{1}{2}\pi$ and $-\frac{1}{2}\pi$ from the first branch (see Figure 4). The shortest path in each segment is then considered as a further branch. Any branch exceeding a certain maximum length L_{max} is ignored to quickly omit possible "dead ends". If no valid path is found or the end point is near an already existing contour point or the image boundaries, the end point is terminated. As such, the number of end points changes with each iteration, dynamically exploring various contours.

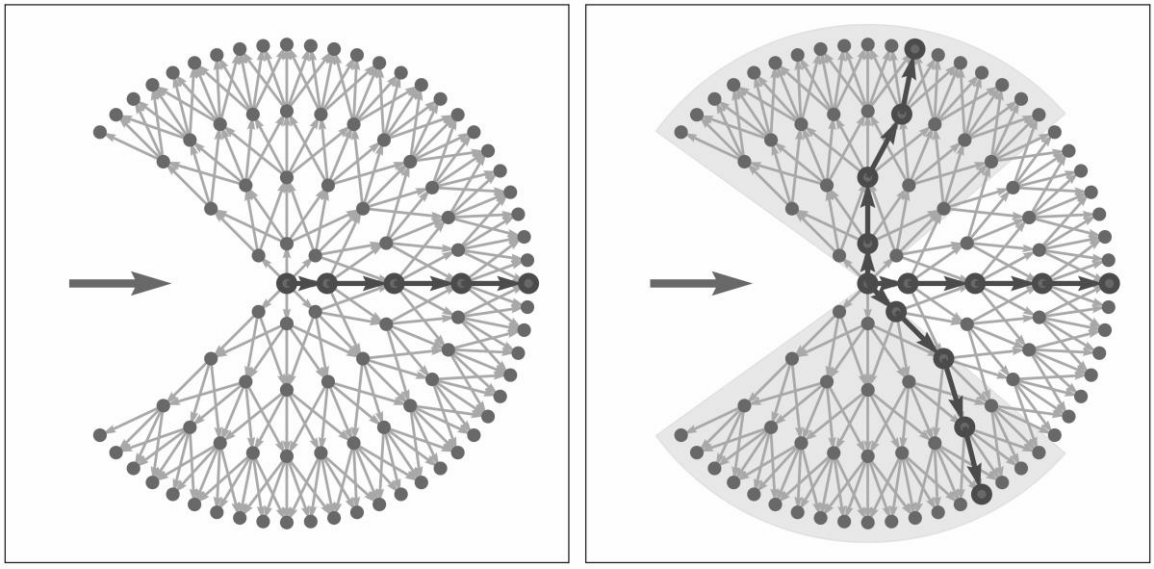

Figure 4. Main steps of movement in local graph: determine overall shortest path in graph (**left**); select shortest paths in subsections (**right**).

3.3.2. Modifications and Adaptive Masking

In our previous study, we observed two main drawbacks of the GGC method. Firstly, there is a significant trade-off between precision of the contour extraction and complexity of the local graph as expressed by the number of circles n_c per unit distance in a given step size r_{max}. Higher complexity dramatically increases the number of vertices and edges considered and consequently slows down the process. Secondly, different quality of boundary inputs to the GGC method requires a different behavior. If the boundary map provided is inconsistent or accuracy is low, following the provided magnitude values may result in "zig-zag" curves and irregular shapes, even when considering distances (see Equation (21)).

To address the complexity issue, we acknowledge that it is not necessary to consider the entire local graph at a given position. In fact, only those parts of the graph that cover actual boundary structures are of interest. Therefore, it would be desirable to exclude certain parts of the graph and thereby reduce the number of possible paths to explore in the movement step. We achieve this by first sampling the local magnitudes at all points of the graph. We then apply Otsu's method to obtain an optimal threshold, separating high and low values in the local neighborhood histogram [47]. By removing all sample points below the threshold, we effectively drop those parts of the graph that do not correspond to any boundary structures (see Figure 5).

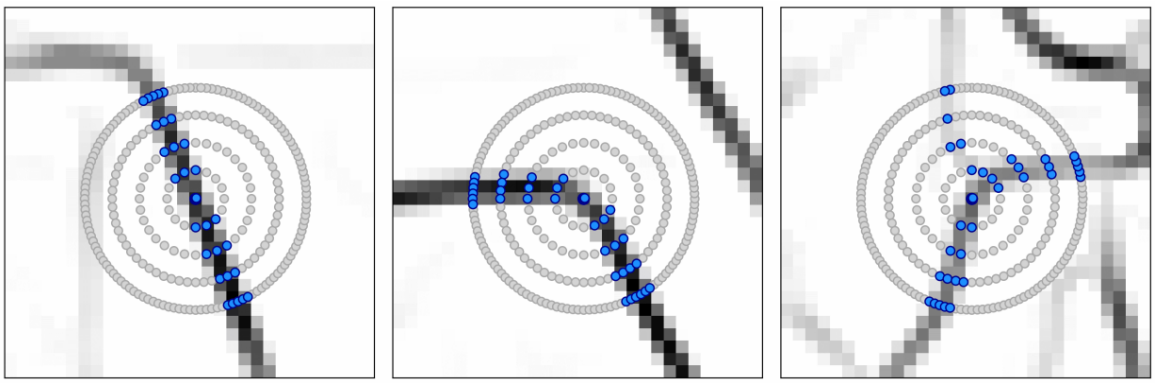

Figure 5. Examples for adaptive masking of vertices in the graph. Light grey points indicate removed points of the local graph, blue points represent those retained after masking.

The aspect of input quality can be addressed by modifying the weighting in Equation (21). If the input map is less reliable, a higher emphasis on shorter (and smoother) paths would be advantageous. If the input map is very accurate, however, the model may more closely follow the provided boundaries to create a more accurate result. Therefore, we introduce the factor β that can be used to increase or decrease the relevance of magnitudes relative to the distance between vertices:

$$w_{i,j} = \frac{D_{i,\,j}}{\beta \cdot g_j} \tag{22}$$

To ensure consistency, L_{max} is scaled by $\frac{1}{\beta}$ to allow using comparable values of L_{max}, regardless of the β used.

3.3.3. Field Polygon Extraction

Using the output of the GGC model as an undirected network or graph of contour points, we intend to produce actual field polygons. Theoretically, finding cycles in the graph would be a possible approach to distinguishing different fields. The large size of the resulting graph, however, makes this task computationally demanding. Further, even a few missing connections may result in incorrectly created polygons or cycles not found.

Following our previous methodology, we therefore transform the contour output of the GGC method into a binary edge map [25]. We use a flood fill algorithm initiated on local maxima of a Euclidian distance transform from the binary boundaries to create a segment representing the rough extent of a field as described by the extracted contours. We then only consider those vertices close to the segment and a new over-connected unweighted and undirected graph is produced in which each vertex is connected to its four nearest neighbors. The field polygon is then obtained using the longest cycle in this network. Finally, polygons are refined using adaptive Gaussian filtering and the Ramer–Douglas–Peucker algorithm [48–50].

4. Results

We provide statistical and visual results for the boundary detection as well as the subsequent field polygon extraction steps. For the sake of brevity, we focus visual comparisons on study area 2.

4.1. Boundary Detection

4.1.1. Input Feature Importance

The step-wise feature importance ranking resulted in relative importance values listed in Table 4. As step five (quintuplets) did not allow for much further distinction in quality, we ultimately arrived at the six feature groups (in order of importance): G8, G4, G14, G1, G6, and G5 and the corresponding 13 features F17–F20, F6–F8, F29, F1, F11–F12, and F9–F10. Multiplying by the three time steps, this resulted in a total of 39 inputs for the boundary detection model.

This demonstrates texture metrics (G8), local statistics (G4), local cues (G14), gradient magnitude (G1), and Hessian matrix eigenvalues (G5 and G6) to be the most relevant for boundary detection. Overall, it seems that color information was significantly more useful than luminosity, as in most feature types, those applied on color channels were preferred over luminosity ones. The only exception in this respect is image gradient.

Texture metrics on color images were particularly useful and continuously achieved the best ranks in all steps except the first one. This initial underperformance may be explained by similar results across all feature groups in the initial tests, except for the two "outliers", G9 and G10 (angular dispersion), which showed very little merit with validation loss barely better than a purely random guess. Further, Hessian matrix eigenvalues proved relevant in terms of both luminosity and color, while local statistics and local cues were again most useful in terms of color.

Table 4. Results of feature importance ranking. Values are given as relative importance values (ratio to best achieved for any feature group in all test runs of the respective step).

	Round 1	Round 2	Round 3	Round 4	Round 5
G1	0.20	0.19	0.30	0.49	0.83
G2	0.17	0.19	0.29	0.47	-
G3	0.25	0.19	0.28	-	-
G4	1.00	0.54	0.48	0.69	0.94
G5	0.33	0.22	0.29	0.51	0.75
G6	0.50	0.35	0.39	0.53	0.79
G7	0.10	0.18	-	-	-
G8	0.11	1.00	1.00	1.00	1.00
G9	0.07	-	-	-	-
G10	0.08	-	-	-	-
G11	0.09	0.18	0.26	-	-
G12	0.13	0.22	0.30	0.48	-
G13	0.08	0.17	-	-	-
G14	0.14	0.20	0.28	0.55	0.88

4.1.2. Performance Metrics

We evaluated the boundary detection model on the test dataset: once on the full dataset and once only on those samples that were actually agricultural according to the land cover maps. We made this distinction to analyze potential effects of basic "post-processing" based on prior knowledge of results.

As seen in Table 5, accuracies tend to be slightly better in study area 2 compared to 1. Nevertheless, results for both areas are similar, indicating a comparable performance of the model. Overall accuracies are reduced by a lower sensitivity while specificity is generally high. The high AUC values of 0.92 and greater indicate a very reliable distinction of the classes. Masking of non-agricultural areas slightly increases performance in all metrics, reaching accuracies of up to 88.2% and AUC of 0.94.

Table 5. Accuracy metrics of the field boundary detection for the test dataset.

	Full Dataset			Only Non-Agricultural Areas		
Study area	**1 & 2**	**1**	**2**	**1 & 2**	**1**	**2**
Sensitivity	77.9%	77.3%	79.1%	81.6%	81.3%	82.5%
Specificity	92.1%	92.9%	93.2%	93.5%	93.2%	94.4%
Accuracy	83.8%	83.4%	84.8%	87.2%	87.8%	88.2%
F_1	0.85	0.85	0.86	0.87	0.87	0.88
AUC	0.92	0.92	0.93	0.94	0.94	0.94

4.1.3. Visual Comparison

Figure 6 provides side-by-side comparisons of the predicted boundary probability map, the masked binary map, the boundary strength map, and the reference boundary map for three subsets.

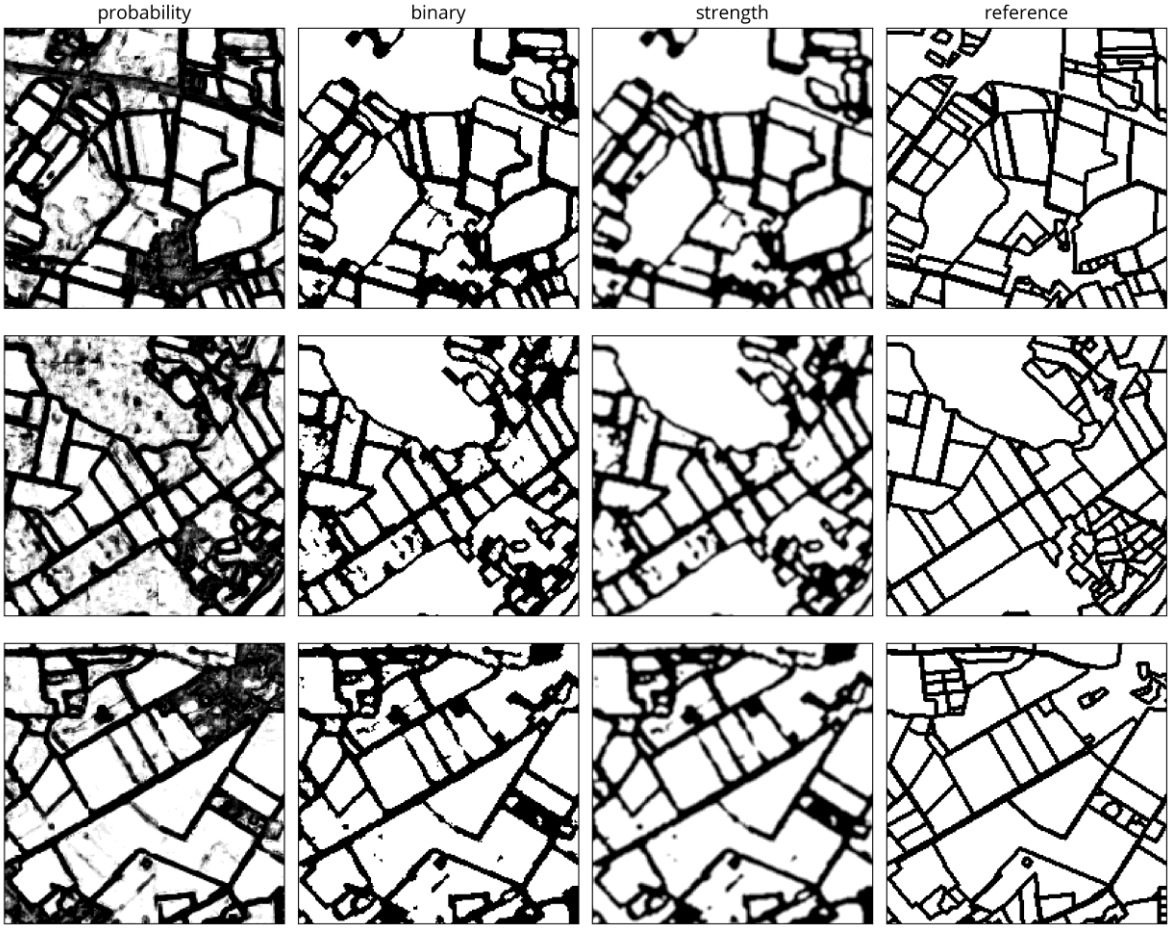

Figure 6. Comparison of boundary detection results in study area 2. Columns represent the model probability output, the binary output after land cover masking, the final boundary strength map, and the final reference dataset (boundaries thickened for visualization). Rows represent different subsets.

The comparison shows that most boundaries are correctly detected and overall appearance is matched quite well, although extracted boundaries tend to be thicker than those drawn in the reference dataset. Problems, however, occur near urban areas (first example top right, third example top right). The heterogeneous structure of these built-up areas is frequently confused with actual field boundaries when they have not been masked using the two land cover maps. Similar issues occur near groups of small fields where the model struggles to separate multiple boundaries in close proximity. In some cases, the land cover based masking excluded some structures that may have been considered agricultural field boundaries (first example top center).

Further, there are cases where the model originally seemed to detect some weaker boundary patterns within the fields that were not part of the reference set (third example bottom left), or it missed some smaller boundary segments (first example center right).

4.2. Field Extraction Results

We compare the default version of the GGC method as presented in our previous publication and the new adaptive version [25]. Settings were selected through manual testing of different settings and visual interpretation of results. According to the results of manual testing, we chose the following setting: $r_{max} = 12$, $n_c = 8$, $L_{max} = 20$, and $\beta = 1.25$.

4.2.1. Visual Comparison

As seen in Figures 7 and 8, distribution and general shapes of extracted polygons often closely resemble those of the reference polygons. However, shapes tend to be a bit smoother and edges more rounded (e.g., Figures 7 and 8 bottom center). Smaller fields, in particular, were occasionally missed or merged with larger fields (e.g., Figures 7 and 8 top center). This can be explained in part by issues in the prior boundary detection step, e.g., inconsistent boundaries or some falsely detected boundaries within the fields. Other fields were also missed by the process, again partly due to the boundary detection step (e.g., Figures 7 and 8 bottom left).

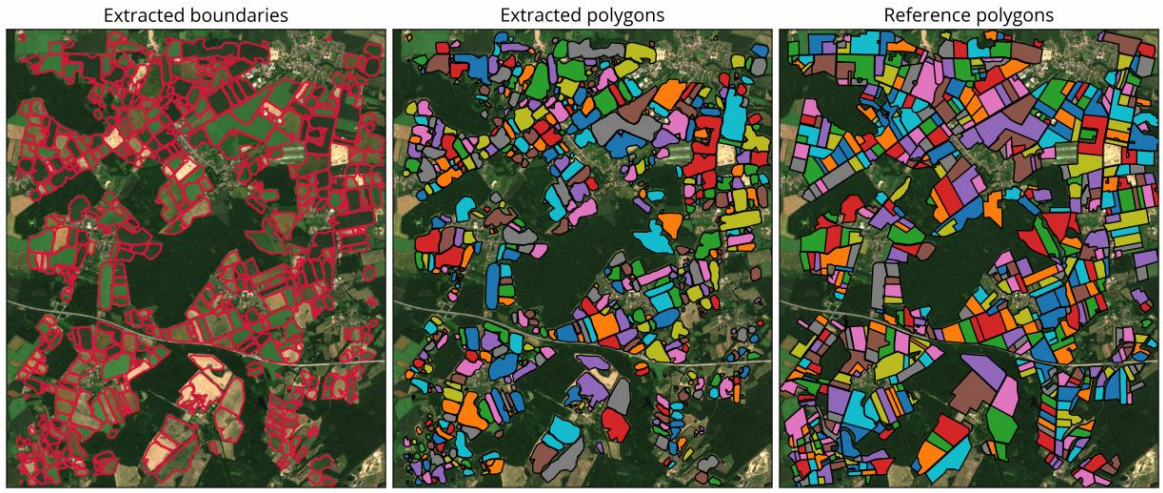

Figure 7. Result of the adaptive graph-based growing contours (GGC) field polygon extraction.

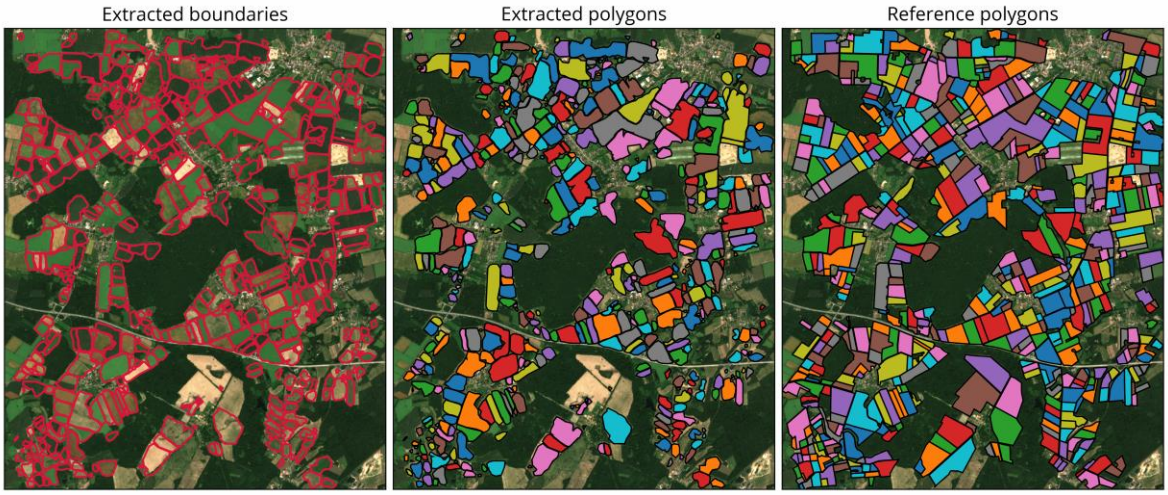

Figure 8. Result of the non-adaptive GGC field polygon extraction.

In particular, clusters of small fields pose problems (see Figures 7 and 8 bottom left). As mentioned in Section 4.1.3, boundaries are sometimes difficult to delineate here and the subsequent extraction may be confused if multiple adjacent boundaries are merged. Additionally, large gaps within fields due to larger non-agricultural areas, such as clusters of trees, occasionally led to errors in the extraction (e.g., Figures 7 and 8 bottom center), confusing the extraction process.

When comparing the adaptive and non-adaptive version, the results look very similar. Differences are visible in some polygons being detected by only one of the versions (e.g., Figures 7 and 8 center and bottom center). In many cases, the adaptive version seems to be a bit less likely to miss fields entirely.

4.2.2. Processing Speed and Upscaling

Our testing revealed the actual processing time to be highly dependent on the implementation. Therefore, the results in this section are presented as a relative comparison of the adaptive and non-adaptive versions rather than a reference for actual processing time. The test runs were performed on an Intel Core i7-4790K CPU with four cores at 4.0 Ghz (4.4 Ghz turbo boost) in single-core execution and 32 GB of memory. Table 6 shows an overview of execution speed and the number of contour points extracted as an indication of the complexity of the given scene.

Table 6. Comparison of processing speed and number of extracted contour points in adaptive vs non-adaptive GGC for differently sized areas. Processing acceleration indicates the reduction in processing time of the adaptive version in relation to the non-adaptive version.

Settings	r_{max} = 6, n_c = 4, L_{max} = 14		r_{max} = 12, n_c = 8, L_{max} = 20	
	Acceleration	Points extracted	Acceleration	Points extracted
2.5 × 2.5 km^2 subset	1.3×	~5500	2.2×	~3800
5.0 × 5.0 km^2 subset	1.3×	~20,000	2.2×	~14,000
10.2 × 7.5 km^2 scene	1.3×	~55,000	2.3×	~40,000

Results demonstrate the adaptive version achieved a substantial acceleration compared to the non-adaptive version at all subset sizes and settings. The advantage is more substantial for larger graphs.

5. Discussion

Analysis of the selected input features revealed that most features obtained from color channels outperformed those based on luminosity. Sequentially dropping the lowest performing groups highlighted texture metrics, eigenvalues of the Hessian matrix, local statistics, local cues, and gradient magnitude as useful. Homogeneity measures, however, showed poorer performance, and angular dispersion did not provide enough information to obtain good boundary detection. It may be interesting for future applications to further explore the effect on performance if the groups dropped were added back into the input set or comparisons were performed on an even more granular level evaluating each individual feature.

The MLP-NN model proved capable of accurately detecting field boundaries. Hyperparameter selection showed multiple layers to be advantageous, combined with a high dropout rate of 80% and a learning rate of 0.0001. Results of the boundary detection model were good, with high accuracies that could be further increased by removing non-agricultural areas from the output. This is particularly promising when considering the input being solely comprised of RGB channels. For both study areas combined, we obtained an accuracy of 87.2%, F_1 score of 0.87, and AUC of 0.94 after removing non-agricultural areas. These results also seem competitive when compared to another deep learning approach recently presented by Masoud et al. [21]. They proposed a field boundary detection based on convolutional NNs trained on S-2 10-m and 20-m resolution imagery of multiple test sites in the Netherlands. They reported an overall best F_1 score of 0.67 and an average F_1 score of 0.63 for all test sites. Further, the overall best sensitivity and specificity values were 0.69 and 0.66, respectively. However, a direct comparison may be difficult, as they used more bands, including near-infrared channels, and merged different spatial resolutions, but did not consider multi-temporal imagery in their approach. This may partly explain the difference in performance. Another approach based on convolutional NNs was presented by Persello et al. [23]. They reported performance similar to that obtained here with F_1 scores of up to 0.830 when using WorldView-2/3 data for field delineation in smallholder farms in Nigeria and Mali.

Visual comparisons further revealed that the boundary detection sometimes observed boundaries that were not part of the reference dataset, while it missed other parts. In some of these cases, it is

debatable if results are objectively wrong, as interpreting certain structures can be difficult if they only occur at some time of the growing season, e.g., due to field management or cropping patterns. The boundaries produced tended to be broader than a precisely hand-drawn delineation and may appear "blurred". This is mostly a result of the appearance of many of the input features, such as gradient magnitude, that do not provide a sharp distinction but rather a gradual highlighting of discontinuities in the image. As the model predictions, however, were intended as an input to the following GGC method, they are not required to be flawless. The β value > 1 selected as best for the extraction step further indicates a high quality of the boundary detection result. For other applications, further post-processing or modifications to the modeling step may be required.

Another advantage of the boundary detection approach presented here is that it is scalable and should be relatively easy to transfer to other regions and landscapes. The inputs to the model are universal image features and exclusively based on regular RGB imagery, enabling an adaptation to other satellite, unmanned aerial vehicle (UAV), or airborne sensors. Further research may also explore the use of higher resolution imagery that offers more accurate field delineations and helps resolve more detail to effectively extend the process to even smaller field sizes.

Field extraction results were also promising and proved to work well when given a consistent, high quality input map. This also allowed the use of a much larger step size than in our previous study. The new adaptive version of GGC obtained very similar results as the non-adaptive one, while drastically reducing processing time. It is therefore more suited for larger-scale applications. A further advantage could be that settings such as L_{max} may not be as crucial as in the previous version because the automatic masking step removes much of the ambiguity of a heterogeneous environment.

Nonetheless, despite the improved detection step, groups of very small fields or very heterogeneous structures posed problems in field polygon extraction. Fields were occasionally merged or missed entirely. While it also did not extract all fields accurately as represented in the reference set, the adaptive version seemed to suffer slightly less from those issues. One reason for this may be that masking less relevant parts of the graph also avoids following erroneous paths. For example, cases in which there are small spots of high boundary strength within a field due to false detections can result in a path being selected that attempts to connect to this point, even if other vertices on the way should clearly not be considered boundaries. By masking weaker vertices, the adaptive version may effectively avoid any possible path to such falsely detected boundary spots.

Another observation was that extracted fields tended to be more rounded in shape and sharp 90° angles were occasionally missed. This is in part due to the field extraction steps following the GGC application (see Section 3.3.3). The over-connected nature of the graph used as a basis for polygon creation can result in rounded shapes when connections "cut out" the actual corner points. Therefore, further research may be needed to improve the transferring of the GGC output to polygons.

Furthermore, it may be interesting to explore other applications of the GGC methodology outside of field delineation. As it is fundamentally a growing active contour technique and independent of the context or the type of input given, its application may extend outside its original use case.

6. Conclusions

We demonstrated a new field boundary detection approach based on deep learning and explored a large number of image features obtained from S-2 RGB imagery. We merged it with the GGC method for field polygon extraction and further introduced an improved version of the GGC approach that increases scalability through automatic masking of the local graph to reduce complexity without affecting quality of the output.

The analysis of a large number of possible input features revealed interesting insights regarding their usefulness for this particular task. While some features such as local statistics, texture metrics, image gradient, Hessian matrix eigenvalues, and local cues were valuable inputs, others such as angular dispersion and homogeneity measures were less useful. Color information was more valuable

than luminosity in most cases. The use of these universal image features may also facilitate adapting the methodology to other sensors and resolutions.

The boundary detection model obtained high accuracies but tended to produce broader boundary lines. The quality of the output map benefited from post-processing based on land cover masking. Limitations of the results were, to a large extent, successfully mitigated by the subsequent GGC method. Nonetheless, issues remain regarding performance on non-agricultural areas in case there is no land cover mask available. Further research may be needed, including closer analysis of input features (settings, types of features to use, etc.), as well as exploring different types of classification models. The use of different time steps and the relevance of multi-temporal compared to single image classification may also be further investigated.

The new, adaptive version of GGC proved to be much faster than the original version without compromising quality of the extraction result. A logical next step may be to apply the technique at an even larger scale, e.g., at county or state level. Beyond that, the nature of the GGC approach also allows exploring further applications outside of field extraction and agricultural remote sensing.

Author Contributions: Concept and design of the study were conceived by M.P.W. in collaboration with N.O. Methodology design, implementation, optimization and analysis of results were carried out by M.P.W., M.P.W. wrote the majority of the manuscript with N.O. contributing in reviewing and finalizing the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: Financial support for publication fees was provided by DFG within the funding programme Open Access Publizieren.

Acknowledgments: We thank the students and student assistants who carried out in-situ mapping campaigns and helped produce and update the reference data.

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

References

1. García-Pedrero, A.; Gonzalo-Martín, C.; Lillo-Saavedra, M. A machine learning approach for agricultural parcel delineation through agglomerative segmentation. *Int. J. Remote Sens.* **2017**, *38*, 1809–1819. [CrossRef]
2. Rahman, M.S.; Di, L.; Yu, Z.; Yu, E.G.; Tang, J.; Lin, L.; Zhang, C.; Gaigalas, J. Crop Field Boundary Delineation using Historical Crop Rotation Pattern. In Proceedings of the 8th International Conference on Agro-Geoinformatics (Agro-Geoinformatics), Istanbul, Turkey, 16–19 June 2019; pp. 1–5. [CrossRef]
3. Turker, M.; Kok, E.H. Field-based sub-boundary extraction from remote sensing imagery using perceptual grouping. *ISPRS J. Photogramm. Remote Sens.* **2013**, *79*, 106–121. [CrossRef]
4. Sagris, V.; Devos, W. *LPIS Core Conceptual Model: Methodology for Feature Catalogue and Application Schema*; OPOCE: Luxembourg, 2008.
5. Inan, H.I.; Sagris, V.; Devos, W.; Milenov, P.; van Oosterom, P.; Zevenbergen, J. Data model for the collaboration between land administration systems and agricultural land parcel identification systems. *J. Environ. Manag.* **2010**, *91*, 2440–2454. [CrossRef]
6. European Court of Auditors. *The Land Parcel Identification System: A Useful Tool to Determine the Eligibility of Agricultural Land—But Its Management Could Be further Improved*; European Union: Luxembourg, 2016; ISBN 9789287259677.
7. Jégo, G.; Pattey, E.; Liu, J. Using Leaf Area Index, retrieved from optical imagery, in the STICS crop model for predicting yield and biomass of field crops. *F. Crop. Res.* **2012**, *131*, 63–74. [CrossRef]
8. Verrelst, J.; Rivera, J.P.; Leonenko, G.; Alonso, L.; Moreno, J. Optimizing LUT-based RTM inversion for semiautomatic mapping of crop biophysical parameters from sentinel-2 and -3 data: Role of cost functions. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 257–269. [CrossRef]
9. Liu, J.; Pattey, E.; Miller, J.R.; McNairn, H.; Smith, A.; Hu, B. Estimating crop stresses, aboveground dry biomass and yield of corn using multi-temporal optical data combined with a radiation use efficiency model. *Remote Sens. Environ.* **2010**, *114*, 1167–1177. [CrossRef]
10. Belgiu, M.; Csillik, O. Sentinel-2 cropland mapping using pixel-based and object-based time-weighted dynamic time warping analysis. *Remote Sens. Environ.* **2018**, *204*, 509–523. [CrossRef]

11. Kussul, N.; Lemoine, G.; Gallego, F.J.; Skakun, S.V.; Lavreniuk, M.; Shelestov, A.Y. Parcel-Based Crop Classification in Ukraine Using Landsat-8 Data and Sentinel-1A Data. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2016**, *9*, 2500–2508. [CrossRef]
12. Peña, J.M.; Gutiérrez, P.A.; Hervás-Martínez, C.; Six, J.; Plant, R.E.; López-Granados, F. Object-based image classification of summer crops with machine learning methods. *Remote Sens.* **2014**, *6*, 5019–5041. [CrossRef]
13. Watkins, B.; van Niekerk, A. A comparison of object-based image analysis approaches for field boundary delineation using multi-temporal Sentinel-2 imagery. *Comput. Electron. Agric.* **2019**, *158*, 294–302. [CrossRef]
14. Yan, L.; Roy, D.P. Automated crop field extraction from multi-temporal Web Enabled Landsat Data. *Remote Sens. Environ.* **2014**, *144*, 42–64. [CrossRef]
15. Tiwari, P.S.; Pande, H.; Kumar, M.; Dadhwal, V.K. Potential of IRS P-6 LISS IV for agriculture field boundary delineation. *J. Appl. Remote Sens.* **2009**, *3*, 033528. [CrossRef]
16. Debats, S.R.; Luo, D.; Estes, L.D.; Fuchs, T.J.; Caylor, K.K. A generalized computer vision approach to mapping crop fields in heterogeneous agricultural landscapes. *Remote Sens. Environ.* **2016**, *179*, 210–221. [CrossRef]
17. Da Costa, J.P.; Michelet, F.; Germain, C.; Lavialle, O.; Grenier, G. Delineation of vine parcels by segmentation of high resolution remote sensed images. *Precis. Agric.* **2007**, *8*, 95–110. [CrossRef]
18. El-Sayed, M.A.; Estaitia, Y.A.; Khafagy, M.A. Automated Edge Detection Using Convolutional Neural Network. *Int. J. Adv. Comput. Sci. Appl.* **2013**, *4*, 11–17. [CrossRef]
19. Kemker, R.; Salvaggio, C.; Kanan, C. Algorithms for semantic segmentation of multispectral remote sensing imagery using deep learning. *ISPRS J. Photogramm. Remote Sens.* **2018**, *145*, 60–77. [CrossRef]
20. Längkvist, M.; Kiselev, A.; Alirezaie, M.; Loutfi, A. Classification and segmentation of satellite orthoimagery using convolutional neural networks. *Remote Sens.* **2016**, *8*, 329. [CrossRef]
21. Masoud, K.M.; Persello, C.; Tolpekin, V.A. Delineation of Agricultural Field Boundaries from Sentinel-2 Images Using a Novel Super-Resolution Contour Detector Based on Fully Convolutional Networks. *Remote Sens.* **2019**, *12*, 59. [CrossRef]
22. Waldner, F.; Diakogiannis, F.I. Deep learning on edge: Extracting field boundaries from satellite images with a convolutional neural network. *Remote Sens. Environ.* **2020**, *245*, 111741. [CrossRef]
23. Persello, C.; Tolpekin, V.A.; Bergado, J.R.; de By, R.A. Delineation of agricultural fields in smallholder farms from satellite images using fully convolutional networks and combinatorial grouping. *Remote Sens. Environ.* **2019**, *231*. [CrossRef]
24. Kamilaris, A.; Prenafeta-Boldú, F.X. Deep learning in agriculture: A survey. *Comput. Electron. Agric.* **2018**, *147*, 70–90. [CrossRef]
25. Wagner, M.P.; Oppelt, N. Extracting Agricultural Fields from Remote Sensing Imagery Using Graph-Based Growing Contours. *Remote Sens.* **2020**, *12*, 1205. [CrossRef]
26. Bishop, C. *Pattern Recognition and Machine Learning*, 1st ed.; Joran, M., Kleinberg, J., Schölkopf, B., Eds.; Springer: New York, NY, USA, 2006; ISBN 9780387310732.
27. Kottek, M.; Grieser, J.; Beck, C.; Rudolf, B.; Rubel, F. World Map of the Köppen-Geiger climate classification updated. *Meteorol. Zeitschrift* **2006**, *15*, 259–263. [CrossRef]
28. Statistisches Amt für Hamburg und Schleswig-Holstein. *Die Bodennutzung in Schleswig-Holstein*; Statistisches Amt für Hamburg und Schleswig-Holstein: Hamburg, Germany, 2019.
29. Tomasi, C.; Manduchi, R. Bilateral filtering for gray and color images. In Proceedings of the Sixth International Conference on Computer Vision, Bombay, India, 4–7 January 1998; pp. 839–846. [CrossRef]
30. Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland (AdV). *Produktspezifikation für ALKIS-Daten im Format Shape*; AdV: Berlin, Germany, 2016.
31. NIST. NIST/SEMATECH e-Handbook of Statistical Methods. Available online: http://www.itl.nist.gov/div898/handbook/ (accessed on 15 May 2020).
32. Meijering, E.; Jacob, M.; Sarria, J.-C.F.; Steiner, P.; Hirling, H.; Unser, M. Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry* **2004**, *58A*, 167–176. [CrossRef] [PubMed]
33. Ando, S. Image field categorization and edge/corner detection from gradient covariance. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 179–190. [CrossRef]

34. Sato, Y.; Nakajima, S.; Shiraga, N.; Atsumi, H.; Yoshida, S.; Koller, T.; Gerig, G.; Kikinis, R. Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. *Med. Image Anal.* **1998**, *2*, 143–168. [CrossRef]
35. Haralick, R.M.; Shanmugam, K.; Dinstein, I. Textural Features for Image Classification. *IEEE Trans. Syst. Man. Cybern.* **1973**, *SMC-3*, 610–621. [CrossRef]
36. Gregson, P.H. Using Angular Dispersion of Gradient Direction for Detecting Edge Ribbons. *IEEE Trans. Pattern Anal. Mach. Intell.* **1993**, *15*, 682–696. [CrossRef]
37. Aganj, I.; Lenglet, C.; Jahanshad, N.; Yacoub, E.; Harel, N.; Thompson, P.M.; Sapiro, G. A Hough transform global probabilistic approach to multiple-subject diffusion MRI tractography. *Med. Image Anal.* **2011**, *15*, 414–425. [CrossRef]
38. Martin, D.R.; Fowlkes, C.C.; Malik, J. Learning to detect natural image boundaries using brightness and texture. *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 530–549. [CrossRef]
39. Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning*, 2nd ed.; Springer Science+Business Media: New York, NY, USA, 2009.
40. Robbins, H.; Monro, S. A Stochastic Approximation Method. *Ann. Math. Stat.* **1951**, *22*, 400–407. [CrossRef]
41. Nair, V.; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel, 21–24 June 2010.
42. Weigand, M.; Staab, J.; Wurm, M.; Taubenböck, H. Spatial and semantic effects of LUCAS samples on fully automated land use/land cover classification in high-resolution Sentinel-2 data. *Int. J. Appl. Earth Obs. Geoinf.* **2020**, *88*, 102065. [CrossRef]
43. Büttner, G.; Kosztra, B.; Soukup, T.; Sousa, A.; Langanke, T. *CLC2018 Technical Guidelines*; European Environment Agency: Vienna, Austria, 2017.
44. Velasco, F.A.; Marroquin, J.L. Growing snakes: Active contours for complex topologies. *Pattern Recognit.* **2003**, *36*, 475–482. [CrossRef]
45. Kass, M.; Witkin, A.; Terzopoulos, D. Snakes: Active contour models. *Int. J. Comput. Vis.* **1988**, *1*, 321–331. [CrossRef]
46. Yiu, P. The uses of homogeneous barycentric coordinates in plane Euclidean geometry. *Int. J. Math. Educ. Sci. Technol.* **2000**, *31*, 569–578. [CrossRef]
47. Otsu, N. A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. Syst. Man Cybernet.* **1979**, *9*, 62–66. [CrossRef]
48. Deng, G.; Cahill, L.W. An Adaptive Gaussian Filter For Noise Reduction and Edge Detection. In Proceedings of the IEEE Conference Record Nuclear Science Symposium and Medical Imaging Conference, 31 October–6 November 1993, San Francisco, CA, USA; pp. 1615–1619.
49. Douglas, D.H.; Peucker, T.K. Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature. *Cartogr. Int. J. Geogr. Inf. Geovisualization* **1973**, *10*, 112–122. [CrossRef]
50. Ramer, U. An iterative procedure for the polygonal approximation of plane curves. *Comput. Graph. Image Process.* **1972**, *1*, 244–256. [CrossRef]