GLL

Geomatics, Landmanagement and Landscape No. 4 • 2024, 171–188

ISSN 2300-1496 https://doi.org/10.15576/GLL/194414

Research paper

Received: 14.09.2024
Accepted: 9.10.2024
Published: 31.12.2024

Algorithmic image analysis – building detection in aerial photos

Szczepan Budkowski[1], ✉ 0000-0002-1806-1173
Agata Basak[2] 0000-0001-5014-8802

[1] Department of Geodesy, University of Agriculture in Krakow
[2] University of Life Sciences in Lublin
✉ Corresponding author: szczepan.budkowski@urk.edu.pl

Summary

The article presents the results of research comparing edge detection methods in digital images and verifying their usefulness in the context of the automatic vectorization process. As part of the experiment, well-known edge detection algorithms based on the analysis of derivatives of image quality functions (Sobel, Canny, Kirch) were implemented. The research problems of the article in the case of building detection basically boil down to the identification of homogeneous areas, the detection of edges or points in a digital image. The original program developed in the Matlab environment made it possible to obtain a description of the edges and their approximation with straight lines, as well as to analyze the quality of the obtained results. In addition, the validity of using neural networks was also analyzed in this context. The neural networks used an algorithm obtained from the GitHub hosting website and implemented as a plug-in for QGIS 3.26. Another attempt at algorithmic image analysis was based on the use of the GAN technique, i.e. the use of a generative network architecture that acts as an algorithm using the potential of two mutually opposed networks whose task is to generate a synthetic result. Under this assumption, one network is the so-called data generator and the other is the discriminator, critically assessing the generating network for authenticity. For each algorithm, the accuracy of vectorization of the detected edges was calculated. The most promising in this respect was an artificial intelligence algorithm using the technique of generative adversarial networks.

Keywords

GIS • geographic information system • machine learning • deep learning • neural networks • artificial intelligence

1. Introduction – research issues

Issues related to digital image processing have been known and used for a long time. Over time, the way in which space is recorded changes, and therefore also the way in which the terrain is represented. Increasing field resolution and the ability of recording subsequent spectral channels provide much higher quality data while maintaining the current workload [ERDAS 2023]. The increase in data quality results primarily from changes related to the development of technology, the use of better quality equipment, and software that enables advanced analyzes to be performed in a short time [Parker 1996]. The use of software is becoming more and more intuitive, which also affects the availability and universality of the performed analyzes [Kraus 1997]. Advanced algorithms free the operator from the need to constantly supervise processes, increasing his efficiency and allowing him to focus on selected elements of the process [Bai et al. 2020]. However, the existence of multiple analysis techniques does not exempt the operator from the need to possess knowledge of the activities performed, such as the preparation of input data, samples, or patterns of individual classes, thanks to which the process will provide satisfactory results [Xu et al. 2018, Wang et al. 2019].

The process of identifying objects in aerial photos can be significantly automated and accelerated thanks to existing tools for algorithmic image analysis, often providing decent results while saving time [Huang et al. 2019, Harirchian 2021, Xu and Noh 2021]. Attempts to automate production stages related to digital image processing also apply to low-level photogrammetry. Feature detection is used in many algorithms for automatic vectorization of objects in digital images or in the recently increasingly popular object-based image classification [Liu et al. 2020].

Research issues related to building detection basically come down to the identification of homogeneous areas, the detection of edges or points of a digital image [Rottensteiner 2001]. Of course, in the available literature one can find another concept leading to the use of a three-level processing system, which takes into account the types of detected objects [Fuchs 1998]. The three-level processing system described by Fuchs [1998] involves a hierarchical approach to detecting objects based on their types. At the lowest level, basic features such as edges, textures, or color gradients are extracted from raw data. The middle level processes these features in order to form intermediate structures or patterns, like shapes or regions, without yet identifying specific objects [Li and Dong 2022]. In the highest level, these patterns are analyzed in a context which is necessary to fully recognize and classify the objects [Liegang et al. 2021]. This gradual increase in complexity at each stage improves the accuracy of object detection. The developed algorithms contributed to simplifying or completely solving many practical tasks [Shon and Dowman 2001]. In the 1990s, the construction of wireframe models was used for architectural inventory purposes. Later methods for building detection and their visualization focused on edge extraction with sub-pixel accuracy [Jachimski and Mikrut 1998].

In the presented experiments, known edge detection algorithms based on the analysis of derivatives of brightness image functions were implemented, as well as modified solutions in the form of machine learning [Rasti 2021, Weider 1995]. The first part was implemented in Matlab, while the machine learning solutions were implemented in the form of Python code. Such division of labor was forced by the desire to confront classic edge detection algorithms with solutions made possible by the use of artificial intelligence in GIS tools.

Contemporary algorithmizations increasingly refer to the use of deep learning methods [Prabhakara and Grag 2023], while most publications refer to extraction using the highest resolution photos [Xiaoming et al. 2023, Wen et al. 2021, Wu et al. 2023, Liu et al. 2020].

2. Materials and methods

Orthophotomap sheets covering the Krakow county and the city of Krakow were used for testing. The orthophotomaps developed differed in spatial scope and terrain resolution. In addition to the above-mentioned material, the data was supplemented based on our own orthophotomaps recorded with a drone, showing objects that were not present on the orthophotomap due to changes that occurred after the photos were obtained. These images differed in quality, which require the experiment to be carried out on diverse research material.

The tests began with the use of classic algorithms based on the analysis of image function discontinuities. Initially, known filters were applied to observe differences in edge detection in photos. The tasks were carried out both on RGB images and on images after the conversion to grayscale. The tasks also included cleaning the images from noise. In total, several dozen operations were performed, changing the order of operations, photos and the method of obtaining the contours, ultimately using the most optimal solutions. The experiments were performed with the following ones: image denoising, image thresholding, erosion, dilation and object segmentation. Known filters were used independently and in combination with each other, and analyzed in terms of the results obtained depending on the materials being developed.

Sobel's algorithms [Parker 1996] belong to a group of simple edge detection methods based on patterns, i.e. filter masks detecting lines with a specific direction, either vertical, horizontal or diagonal. The Sobel algorithm uses two filtering masks: horizontal S_x – determining the gradient value towards the rows, and vertical S_y – determining the gradient value towards the columns.

The algorithm is implemented according to formulas (1, 2):

$$G_{mag} = \sqrt{\left(S_x\right)^2 + \left(S_y\right)^2} \tag{1}$$

$$G_{dir} = \arctan\left(\frac{S_y}{S_x}\right) \tag{2}$$

In the case of the Kirch algorithm, image analysis is performed with the support of eight filter masks. The coefficients of individual masks were selected in such a way as to detect the difference in the grayscale in each of the eight possible turns in the vicinity of the examined pixel. The algorithm therefore allows you to analyze pixel brightness changes in every possible direction. Pixels that are similar are connected to each other, and the boundary is a step change in the brightness parameters of individual pixels. The response value is the maximum value obtained for a given pixel, expressed by the formula (3):

$$G_{mag} = \max \{k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7\} \tag{3}$$

Edge orientation in the algorithm is performed according to formula (4):

$$G_{dir} = \frac{\pi}{4 \cdot i} \tag{4}$$

where:

i – the number of the mask for which the value was obtained.

In the case of the Canny algorithm, the first stage is the necessary noise reduction with a Gauss filter, and then determining the first derivative of the image as the derivative of this function. The operations are performed according to equation (5):

$$G_0 (x)' = \left(- \frac{x^2}{6^2}\right) \cdot e^{-\frac{x^2}{26^2}} \tag{5}$$

where:

6 – the standard deviation,

x – the position of the pixel in the mask in the direction of the rows.

In the case of solutions for a horizontal mask, the formula is analogous, but implemented with respect to a different axis. The result of the image convolution in gradient components in two directions perpendicular to each other.

The last stage of the experiment was to vectorize the buildings with an artificial intelligence algorithm. For this purpose, a building detection algorithm available on Github was adopted. The availability of such a complex algorithm required a change of software to Python software, which was implemented into the QGIS program. Automatic analysis of the Deep Learning raster technique can be divided into several stages:

a. Labeling the sample objects is undoubtedly the stage that engages the operator the most. In the case of deep learning, the selection of appropriate sample data was the most time-consuming. It is important to take into account the use of a wide set of data from the Krakow county and the city of Krakow. This selection of the database meant that the sample buildings obtained were objects with various shapes. In addition to the classic buildings, which usually have a cuboid shape (rectangular in orthogonal projection), occurring mainly in the district, objects with slightly more modernist shapes were also identified.
b. Training the deep learning model – in this stage everything outside the labeled buildings was treated as a background value, and therefore as an undesirable ele-

ment. This is a fully automated stage, which involves the operator defining training parameters such as sample size, thresholding value or pixel classification.

c. The third stage is the classification of images on the target raster. A well-conducted stage of labeling and training the model results in the detection of similar objects (in this case buildings) in other rasters.
d. Analysis and evaluation of the results delivered by the model. This stage is again characterized by the need for the operator to intervene in the process, whose main task is to verify the correctness of the vectorization of the buildings from the orthophotomap by comparing it with the effects of the algorithm's work.
e. The last stage is to complete the patterns and re-train the model. This stage is worth attention in case a situation arises in which the current model does not fulfill its function. In this case, one should go back to the labeling the example buildings again.

The generative modeling approach for deep learning algorithms is supported by the GAN (Generative Adversarial Networks) technique. It involves learning regularities, automatic discoveries, or patterns as input. Generative network architecture works as an algorithm that uses the potential of two mutually opposing networks to generate a synthetic result. Under this assumption, one network is the so-called data generator and the other is the discriminator, critically assessing the generating network for authenticity. This means that the discriminator decides whether the generated result it evaluates belongs to the actual training data set. Due to the huge amount of data required for this technique, existing shapefile layer buildings obtained from OpenStreetMap were used for the purpose of the experiment.

The methodology includes the use of the Paddle library and the COCO format, as presented in the following fragment of the algorithm (Fig. 1).

The main method does all the work of the plugin. It opens a dialog box (SplitRSDataDialog) for the user, then after clicking the 'OK' button, it processes raster and vector data, creates a directory structure, generates a PaddlePaddle dataset, creates a COCO dataset, etc. The algorithm:

- Retrieves information about the currently selected raster and vector layers.
- Creates a directory structure to store PaddlePaddle and COCO data.
- Processes raster and vector data, creating a rasterized file and dividing it into images and labels.
- In the case of the 'Instance Segmentation', generates segmentation maps.
- For the 'PaddlePaddle' option, generates file lists for the PaddlePaddle dataset.
- For the 'COCO' option, creates a dataset in the COCO format.

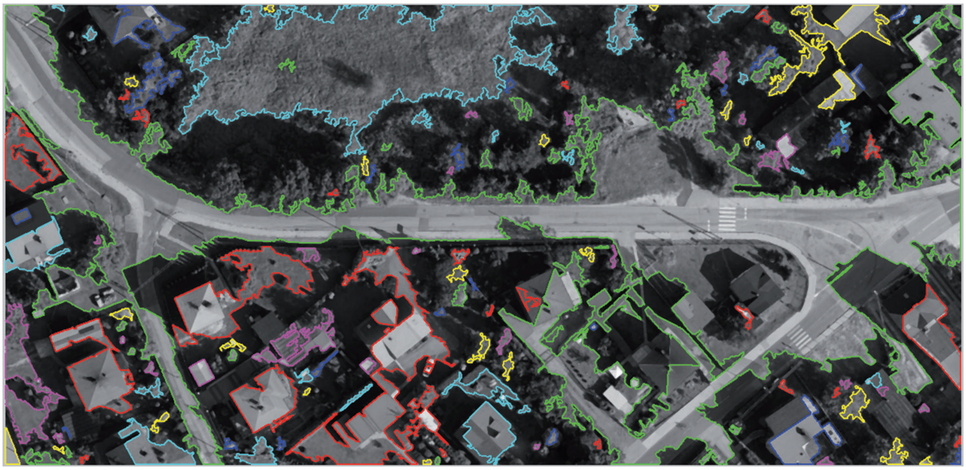
```
#PaddlePaddle _ Dataset Paths
dataset_paddle = osp.join (dataset_path, " PaddlePaddle ")
mkdir_p (dataset_paddle)

Ras_Paddle_path = osp.join (dataset_paddle, " rasterized /")
output = osp.join (
   Ras_Paddle_path, currentrasterlay + " rasterized " + ".tif "
) # Output Rasterized File
image_Paddle_path = osp.join (dataset_paddle, "image/")
label_Paddle_path = osp.join (dataset_paddle, " label /")
InSeg_Paddle_path = osp.join (dataset_paddle, " inseg /")
mkdir_p (Ras_Paddle_path)
mkdir_p (image_Paddle_path)
mkdir_p (label_Paddle_path)
mkdir_p (InSeg_Paddle_path)

feedback = QgsProcessingFeedback ()
feedback.pushInfo ("Raster Path : " + ras_path)
feedback.pushInfo (" Vector Path : " + vec_path)
feedback.pushInfo (" Output Rasterized Path : " + output)
feedback.pushInfo (" Imge Splitting Size : " + str (SplittingSize))

# TODO: if shp in memory, it can't work

rasterize (ras_path, vec_path, output)
iface.messageBar (). pushMessage (
   " You Will find the rasterized file in " + output,
   level = Qgis.Info,
   duration = 5,
```

Source: Authors' own study

Fig. 1. Fragment of the algorithm

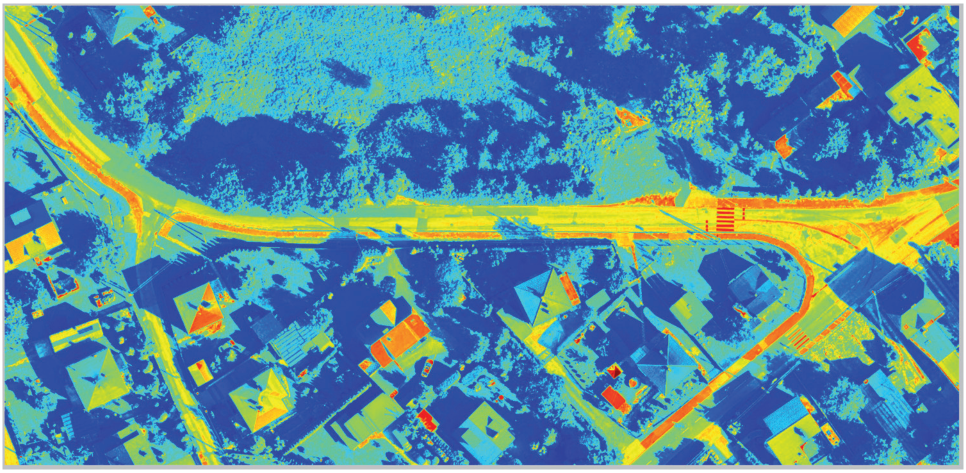
3. Results

The ongoing research on building detection focused mainly on edge detection. In the first stage, the image was converted to grayscale in order to reduce the time needed for calculations. The next step was to perform median filtering, which allowed for the reduction of image noise. Due to the fact that the main axis of activities focused on the detection of edges in the image, after thresholding, all edges that the algorithm was able to find were presented (not only buildings). In the developed image (Fig. 2), it can be seen that the algorithm coped very well with the detection of edges, the identification of which was strongly correlated with the change in pixel brightness. This can be seen, for example, in the shadows cast by the buildings. The lack of clarity in the identification of buildings was partly due to the failure to apply the condition requiring the building to be recognized as a homogeneous area, usually characterized by specific shapes (square or rectangular) and angular values between the edges that are correlated with the shape and side ratios. Taking into account the conditions of homogeneity and shape in the algorithmization process improved the quality of identified areas (warm colors), but still did not ensure that only objects buildings were detected (Fig. 3).



Source: Authors' own study

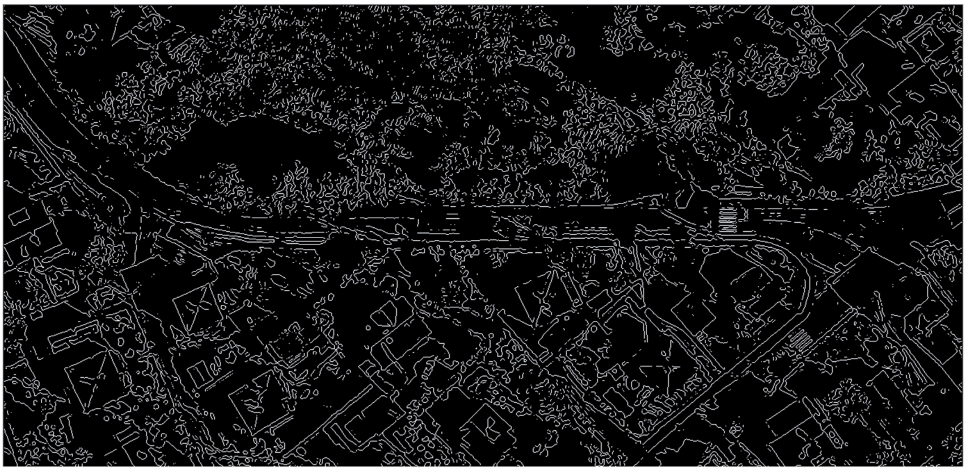
Fig. 2. Detection of the edges of an orthophotomap image



Source: Authors' own study

Fig. 3. Identification of areas that meet the shape conditions

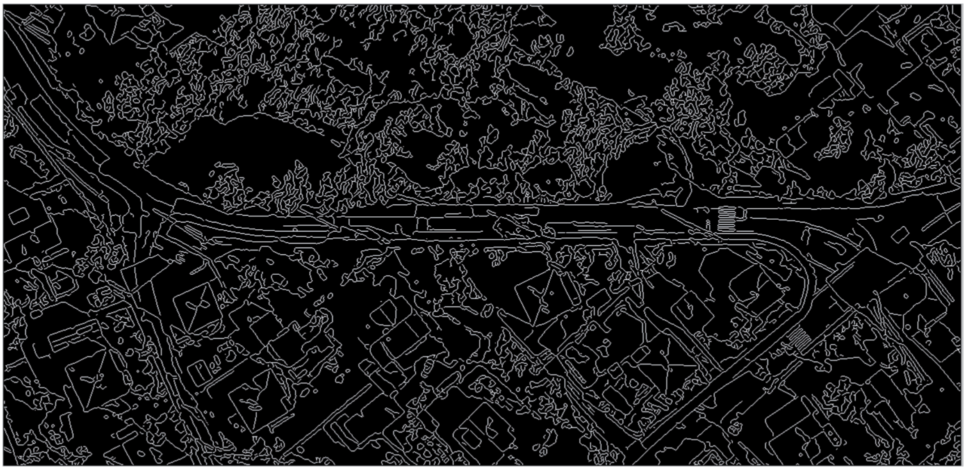
Subsequent tests of edge identification were based on the Kirch, Sobel and Canny algorithms preceded by converting photos to shades of gray and using a median filter to reduce noise. In each of the analyzed cases, the identification of the buildings was not clear. The algorithms differed between each other. The Sobel and Canny algorithms provide the most accurate representations of the space, while the Kirch algorithm renders the least detail. The large number of details in the case of the first two algorithms, on the one hand, provides detail but, on the other hand, makes it difficult to correctly identify the elements and to algorithmize them. Figures 4, 5 and 6 show the edges detected using the algorithms discussed above. The figures show all edge detection. In order to accurately identify buildings, it is necessary to impose additional conditions that remove background elements by detecting irregular lines, as in the previous cases, as well as conditioning the permissible limit of angles close to right angles. This method of conditioning guarantees the removal of redundant background elements, but it does not cope with the detection of buildings with shapes other than the default ones (rectangular). Regardless of the edge detection filter used, the brightness of the terrain pixels is also important, as it creates or disturbs the homogeneity of the areas. The development of the area, where elements (such as roads) cause reflection or refraction of light rays, is also important. Due to this fact, in sensitive algorithms based on the difference in pixel brightness, they determine the detection of edges and, when applying dilation and erosion operations, also structures. This causes, for example, the road to be presented as a uniform, homogeneous structure (except for darkened areas), with a regular edge that resembles a rectangle.



Source: Authors' own study

Fig. 4. Edge identification using the Sobel algorithm

The last test involved the artificial intelligence algorithm for vectorization of buildings. The algorithm was obtained from the GitHub hosting website and implemented as a plug-in for QGIS 3.26. The most labor-intensive stage was the labeling stage, where the main task was to identify and label the buildings. The necessity to cover a large number of buildings in order to forma a basis for the deep learning algorithm followed the need to train the model exhaustively. In this case, to efficiently conduct the training, it was necessary to use a computer equipped with a high-performance graphics card and processor. High hardware requirements were caused by the need to analyze high-resolution raster materials. The system performance depends on the data set, the number of objects in the photos, disk performance, and the workstations themselves.



Source: Authors' own study

Fig. 5. Edge identification using the Canny algorithm

Source: Authors' own study

Fig. 6. Identification of edges using the Kirch algorithm

The result of the labeling was a vector layer containing the location of identified objects saved as building polygons. This layer was the ‘base’ for the algorithm, which selected objects from orthophotomaps in the identification process on the target raster. It was important for this study to carry out the labeling stage both in the city of Krakow and in the county. This selection of objects was dictated by the fact that the buildings located within the county were usually characterized by a shape close to cuboid. However, buildings located in the city of Krakow often deviated from these shapes in favor of more modernist ones.

Two areas with different characteristics were selected for testing. The first area was the town of Goszcza in the Krakow county, while the second area was the city center of Krakow. Areas differ from each other not only in the number of objects, but also in their shape and density in a given area.

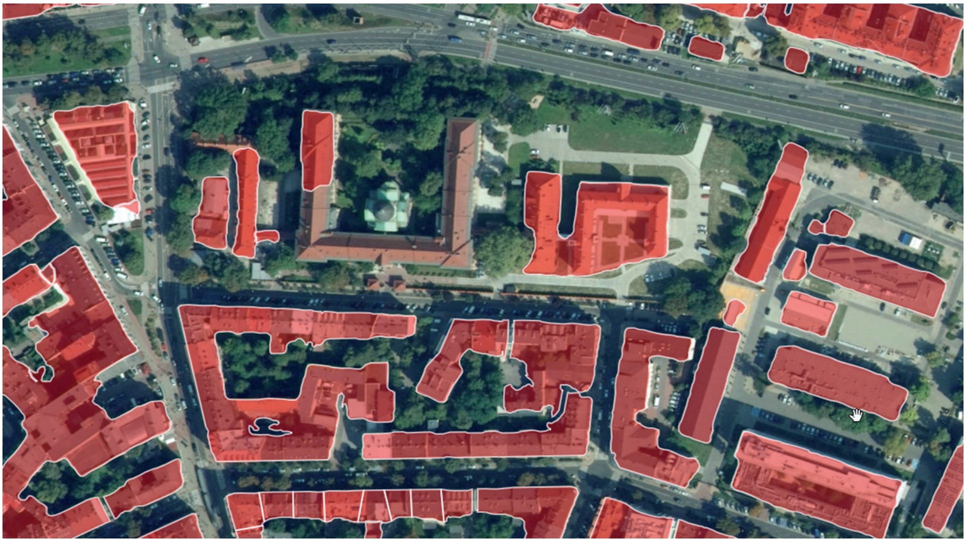
The analysis and evaluation of the results were quite satisfactory. The algorithm coped relatively well with detecting the buildings in the Goszcza municipality. Despite that, not all buildings were properly vectorized An example of this is the garage building marked with a red circle in the central part of the photo (Fig. 7). Of the 1,043 buildings targeted for identification, the detection efficiency was over 97%, and only 21 buildings were missed. In addition, 9 buildings, despite correct identification, were not vectorized properly because the algorithm did not cope well with buildings located in the immediate vicinity, sometimes showing two twin buildings as one.

Source: Authors’ own study

Fig. 7. Vectorized buildings in the town of Goszcza, Kraków county

The algorithm performed much worse when there was a large number of objects to detect, as was the case with the city of Krakow, an example of which is shown in

Figure 8. The first big error of the algorithm was an incorrect identification of the object. Unfortunately, there were situations in which objects were not detected or were detected in places where they should not have been detected. In addition, objects were detected fragmentarily, or detected as one object if located in close proximity of each other. Since such cases were not isolated, it was decided to return to the labeling stage and multiply the number of objects on the basis of which the algorithm can detect.



Source: Authors' own study

Fig. 8. Vectorized buildings in Krakow

Source: Authors' own study

Fig. 9. Vectorized buildings using deep learning in Krakow

Additionally, for a better final effect (Fig. 9), it was decided to use the technique of generative adversarial networks (GAN), which allowed for supervised learning with two sumodels.

In order to perform quantitative and qualitative statistical research, the vectorized building edges were compared with objects (building edges) revealed in the OpenStreetMap database. The algorithms were tested with different parameter settings (mask size, 6, binarization threshold). Accuracy analysis reveals that building edges were vectored correctly (P) to the ratio of all building edges (6) in an area covering the entire Goszcza municipality and the 12 square kilometers of the central part of the city of Krakow (Fig. 10).

$$\text{Correctness rate} = \frac{P}{\left(P + PO + NP\right)} \tag{6}$$

where:

P - correctly identified building edges,
PO - omitted building edges,
NP - edges incorrectly identified.

```python
import cv2
import numpy as np

# Wczytaj obraz
image = cv2.imread('obraz_lotniczy.jpg', cv2.IMREAD_GRAYSCALE)

# Zastosuj segmentację obrazu
_, segmented_image = cv2.threshold(image, 128, 255, cv2.THRESH_BINARY)

# Wykorzystaj analizę złączności
_, labels, stats, _ = cv2.connectedComponentsWithStats(segmented_image)

# Oblicz stopień złączenia
degree_of_connectivity = len(stats) - 1  # -1, aby uwzględnić tło jako

print("Stopień złączenia:", degree_of_connectivity)
```

Source: Authors' own study

Fig. 10. Fragment of the algorithm code

In addition to the number of buildings in individual photos, the following is also important:

a) Building density, which is proposed to be expressed using the formula:

$$\text{Density} = \frac{\text{Building area}}{\text{Total area of the area}} \tag{7}$$

The degree of connection, expressed as the number and size of areas representing buildings (6, 7, 8, 9),

b) Form factor expressed as the ratio of the perimeter to the area of buildings,

c) The distance between buildings is calculated as the average distance between the centers of gravity of individual buildings

$$\text{Average distance} = \frac{\sum \text{distance between the centers of gravity of buildings}}{\text{number of buildings}} \tag{8}$$

d) Area congestion factor, expressed as:

$$\text{Congestion factor} = \frac{\text{number of buildings}}{\text{area}} \tag{9}$$

Table 1. Results of quantitative and qualitative analysis of the tested algorithms

Correctness coefficient value										
	Sobel		Canny		Kirch		Neuron network		"Neural network - GAN technique"	
Name	**Goszcza**	**Kraków**	**Goszcza**	**Kraków**	**Goszcza**	**Kraków**	**Goszcza**	**Kraków**	**Goszcza**	**Kraków**
Foto 1	0.430	0.127	0.653	0.293	0.277	0.110	0.910	0.771	1.000	0.993
Foto 2	0.567	0.202	0.515	0.322	0.488	0.456	0.959	0.640	1.000	0.998
Foto 3	0.720	0.141	0.948	0.413	0.704	0.119	0.980	0.684	1.000	0.954
Foto 4	0.369	0.807	0.417	0.968	0.181	0.410	0.966	0.615	1.000	0.996
Foto 5	0.496	0.012	0.817	0.088	0.388	0.022	0.978	0.737	1.000	0.984
Foto 6	0.549	0.125	0.942	0.289	0.446	0.048	0.973	0.825	0.998	0.965
Foto 7	0.421	0.214	0.749	0.757	0.386	0.243	0.955	0.784	0.997	1.000
Foto 8	0.549	0.239	0.612	0.363	0.363	0.415	0.957	0.820	0.985	1.000
Foto 9	0.597	0.354	0.984	0.389	0.264	0.206	0.991	0.795	0.968	1.000
Foto 10	0.391	0.878	0.791	0.322	0.232	0.491	0.967	0.754	0.977	1.000

Table 1. cont.

Correctness coefficient value										
	Sobel		Canny		Kirch		Neuron network		"Neural network - GAN technique"	
Name	Goszcza	Kraków	Goszcza	Kraków	Goszcza	Kraków	Goszcza	Kraków	Goszcza	Kraków
Foto 11	0.460	0.546	0.800	0.549	0.025	0.400	0.983	0.794	1.000	1.000
Foto 12	0.541	0.421	0.893	0.326	0.899	0.343	0.987	0.693	1.000	1.000
Foto 13	0.601	0.548	0.317	0.357	0.268	0.311	0.964	0.867	1.000	0.994
Foto 14	0.715	0.842	0.792	0.538	0.407	0.468	0.982	0.797	1.000	0.965
Foto 15	0.605	0.311	0.911	0.292	0.039	0.321	0.996	0.844	1.000	0.929
Foto 16	0.509	0.184	0.322	0.274	0.231	0.175	0.974	0.501	1.000	0.985
Foto 17	0.586	0.183	0.447	0.242	0.339	0.364	0.990	0.789	1.000	0.945
Foto 18	0.482	0.117	0.599	0.810	0.214	0.189	0.982	0.830	1.000	0.932
Mean	0.533	0.347	0.695	0.422	0.342	0.283	0.972	0.752	0.996	0.980

4. Discussion of the results

The first phase of the experiment focused on the analysis of known algorithms for edge detection. The Sobel, Canny and Kirch algorithms were compared (Table 1). Regardless of the algorithm used, detecting all the buildings required additional conditioning. The statistical tests presented in the third section leave no doubt that the Canny algorithm is the most accurate in terms of edge detection (average result 0.695 for rural areas, 0.422 for urban areas). Although this algorithm, like the Sobel's algorithm, detects the most details, it is in line with the expectations, and conducted research [Czechowicz and Mikrut 2006, Cui et al. 2008] also detects a large amount of noise. This is due to the relationship between the binarization threshold and the detection quality. In the case of the Canny algorithm, it should be stated that the optimal solution is to use small masks, e.g. 3×3 pixels with any value of the sigma parameter, which allows for cleaning the images from noise.

The next phase of the experiment was based on the use of neural network analysis methods. Here, an extremely important part was labeling, which determined the final result. A large amount of input data was necessary to properly test the model, which is especially visible in the differences in the detection and vectorization of buildings in rural and urban areas. Of the 1,043 buildings targeted for identification, the detection efficiency was over 98%, and only 21 buildings were missed. In addition, 9 buildings, despite correct identification, were not vectorized properly. The algorithm, which had a 97% success rate in detecting a building in the village of Goszcza in the Krakow county, was unable to perform in the city. Only the use of the GAN technique brought the expected results. Although, in this respect, the presented research is generally consistent with the available literature [Dalka 2014, Dalka et al. 2013, Mikrut 2003], it sheds new light on the possibilities offered by algorithmic image analysis, especially considering its use in the context of building a modern space description system [Budkowski et al. 2022]. The conducted experiments could potentially improve the quality of modernization works [Budkowski and Gniadek 2019, 2020]. Automating the checks performed between the buildings existing in the orthophotomaps and those in the database of land and building records could be a development factor giving an impulse to the creation of a modern cadastre, which would undoubtedly have the capacity to be updated almost in real time. Solutions that take into account the use of neural networks in the GAN technique are consistent with the idea of creating the SMART cadastre system known from the literature [Budkowski and Litwin 2019, 2022, Budkowski 2021]. It should also be noted with certainty that (Fig. 8) in the last image the edges are orthogonal, but they are not properly aligned and sometimes overlap. Building trace extraction [Touzani and Granderson 2021] can therefore be developed for future research work. The topic of algorithmic image analysis is not limited to narrow applications in geodesy [Adamiak et al. 2021, Adamiak et al. 2021]. Edge detection algorithms and, above all, neural networks are used in modern cars, medicine, aviation and many other fields [Mahlik 2015, Petryniak 2011]. Such a wide range of implementations makes the publication extremely valuable and provides a basis for further research.

5. Conclusions

This study concerns the issue of building detection, with particular emphasis on edge detection algorithms. The problem was developed using Matlab software, and the part involving the use of neural networks was implemented using Python code, available on GitHub hosting, which was integrated into the QGIS 3.26 program.

The research carried out allows for ranking the tested algorithms in terms of detection quality in the following order from best to worst: neural networks using the GAN technique, neural networks, Canny, Sobel, Kirch. The results were systematized based on the average value of the correctness coefficient developed during the analyses, which guaranteed qualitative analysis and supported the implementation of statistical research. The average values of the obtained coefficients differ from each other, depending not only on the algorithm used, but also on the various orthophotomaps and areas with different degrees of urbanization. The algorithms coped much better with less urbanized areas, achieving an average correctness rate of 0.342 – for the Kirch algorithm, and 0.996 for neural networks using the GAN technique. In urban spaces, the algorithms faced much more problems. There were situations where objects were not detected or were detected in places where they should not have been detected. In addition, objects were detected fragmentarily or, in the case of those located in close proximity, detected as a single object. These errors had direct impact on the reduction of the correctness rates of individual photos, as well as their average value. Therefore, at this point it is not yet possible to use this method to update the real estate cadastre.

In the view of the great potential of the conducted research, the obtained results should not be regarded as definitive. The chosen research direction gives hope for further development of extraction methods. Further research will allow the authors to focus on increasing the effectiveness of the solutions used by optimizing them and preparing digital images in advance by using adaptive filters. The presented methods of algorithmic image analysis can also be used to modify existing procedures and recommendations, for example in the context of modernization works.

Financed by a subsidy from the Ministry of Education and Science for the University of Agriculture in Krakow for 2024.

References

Adamiak M., Będkowski K., Majchrowska A. 2021. Aerial Imagery Feature Engineering Using Bidirectional Generative Adversarial Networks: A Case Study of the Pilica River Region, Poland, Remote Sensing, 13(2), 306.

Adamiak M., Jażdżewska I., Nalej M. 2021. Analysis of Built-Up Areas of Small Polish Cities with the Use of Deep Learning and Geographically Weighted Regression. Geosciences, 11(5), 223.

Bai T., Pang Y., Wang J., Han K., Lou J., Wang H., Lin J., Wu J., Zhang H. 2020. An Optimized Faster r-cnn Method Based on Drnet and Roi Align for Building Detection in Remote Sensing Images. Remote Sensing, 12, 762.

Budkowski S. 2021. Multidimensional cadastre as an element of participation in modern space management. Geomatics, Landmanagement and Landscape, 4, 17–26.

Budkowski S., Gniadek J. 2019. Selected problems pertaining to modernization of land and building records – determination of boundaries of real estate plots. Geomatics, Landmanagement and Landscape, 3, 17.

Budkowski S., Gniadek J. 2020. Boundaries in the real estate cadastre – establishment of boundaries in the land and building records. Geomatics, Landmanagement and Landscape, 2, 7.

Budkowski S., Gniadek J., Litwin U. 2022. Multidimensional cadastre – a modern way of describing space. Publishing House of the University of Agriculture in Krakow.

Budkowski S., Litwin U. 2019. Synergy in space – the use of GIS tools on the example of premises records in the LoD4 model. Acta Scientarum Polonorum, Formatio Cirucumiectus, 18(1), 127–135.

Budkowski S., Litwin U. 2022. The Polish cadastre 3D. FIG Conference 2022, Warsaw.

Cui S., Yan Q., Liu Z., Li M. 2008. Building Detection and Recognition from Hight Resolution Remotely Sensed Imagery. Proceedings of the XXIst ISPRS Congress, 37, 411–416.

Czechowicz A., Mikrut S. 2006. Analysis of the usefulness of edge detection algorithms in short-range photogrammetry applications. Archive of Photogrammetry, Cartography and Remote Sensing.

Dalka P. 2014. Methods of algorithmic video image analysis for road traffic monitoring applications. Doctoral thesis. Gdańsk University of Technology.

Dalka P., Ciarkowski A., Szczuko P., Czyżewski A. 2013. Open standards-based communication system for distributed intelligent surveillance solution. Telecommunications Review, 8–9, 777–785.

ERDAS Imagine. Geospatial Modeling & Visualization. https://gmv.cast.uark.edu/photogrammetry/ [accessed: 15.02.2023].

Fuchs K., Heuel S. 1998. Feature extraction. Third Course in Digital Photogrammetry. Institute for Photogrammetry, University Bonn.

Harirchian E., Hosseini S.E.A., Jadhav K., Kumari V., Rasulzade S., Işık E., Wasif M., Lahmer T. 2021. A Review on Application of Soft Computing Techniques for the Rapid Visual Safety Evaluation and Damage Classification of Existing Buildings. J. Build. Eng.

Huang J., Zhang X., Xin Q., Sun Y., Zhang P. 2019: Automatic bulding extraction from high-resolution aerial images and LiDAR data using gated residual refinement network. ISPRS Journal of Photogrammetry and Remote Sensing, 151, 91–105. https://doi.org/10.1016/j.isprsjprs.2019.02.019.

Jachimski J., Mikrut S. 1998. An attempt at sub-pixel localization of contour lines using the second derivative of a digital image. Archive of Photogrammetry, Cartography and Remote Sensing, 8, 25/1–25/7.

Kraus K. 1997. Photogrammetry, Ferd. Dummlers Verlag, Bonn, 4th ed.

Li Z., Dong J. 2022. A Framework IntegratingDeeplabV3+, Transfer Learning, Active Learning, andIncremental Learning for Mapping Building Footprints. Remote Sensing, 14(19). https://doi.org/10.3390/rs14194738

Liegang X., Xiongbo Z., Zhang J., Yang H., Chen T. 2021. Building Extraction from Very-High-Resolution Remote Sensing Images Using Semi-Supervised Semantic Edge Detection. Remote Sensing, https://doi.org/10.3390/rs13112187

Liu J., Li T., Xie P., Du S., Teng F., Yang X. 2020. Urban big data fusion based in deep learning: An overview. Information Fusion, 53, 23–133. https://doi.org/10.1016/j.inffus.2019.06.016

Mahlik M. 2015. Fire detection in images. Engineering thesis. Warsaw University of Technology.

Mikrut S. 2003. The influence of scanning and compression according to the JPEG standard on the detection of linear and point objects in digital images. PhD thesis. AGH, Krakow.

Parker J.R. 1996. Algorithms for Image Processing and Computers Vision. Wiley.

Petryniak R. 2011. Algorithms for computer detection of large objects in images with a high level of noise and heterogeneity. Doctoral thesis. AGH, Krakow.

Prabhakara D., Grag P.K. 2023. Building edge detection from very high-resolution remote sensing imagery using deep learning. The International Archives of the Photogrammetry Remote Sensing and Spatial Information Sciences. https://doi.org/10.5194/isprs-archives-XLVIII-M-3-2023-189-2023

Rasti B., Ghamisi P. 2021. Remote sensing image classification using subspace sensor Fusion. Information Fusion, 64, 121–130. https://doi.org/10.1016/j.inffus.2020.07.002

Rottensteiner F. 2001. Semi-automatic extraction of buildings based on hybrid adjustment using 3D surface models and management of building data in a TIS. PhD thesis, Institute of Photogrammetry and Remote Sensing, University of Vienna.

Sohn G., Dowman I. 2001. Extraction of buildings from high-resolution satellite data. A.A. Balkema Publishers, Lisse, 345–355.

Touzani S., Granderson J. 2021. Open Data and Deep Semantic Segmentation for Automated Extraction of Building Footprints. Remote Sensing, 13(13), 2578. https://doi.org/10.3390/rs13132578

Wang X., Li P. 2019. Extraction of urban building damage using spectra, height and corner information from VHR satellite images and airborne LiDAR data. ISPRS Journal of Photogrammetry and Remote Secsing, 159, 322–336.

Weider U., Förstner W. 1995. Towards Automatic Building Extraction from High-Resolution Digital Elevation Models. ISPRS, Journal of Photogrammetry and Remote Sensing, 50, 38–49.

Wen X., Li X., Zhang C., Han W., Li E., Liu W., Zhang L. 2021. Me-net: A multi-scale erosion network for crisp building edge detection from very high resolution remote sensing imagery. Remote Sensing, 13(19). https://doi.org/10.3390/rs13193826

Wu Y., Xu L., Wang L., Chen Q., Chen Y., Clausi D.A. 2023. Multi-Task Edge Detection for Building Vectorization from Aerial Images. IEEE Geoscience and Remote Sensing Letters, 20, 1–5. https://doi.org/10.1109/LGRS.2023.3244131

Xiaoming L., Dan Z., Ye Z., Wuyang H., Weixi W., Jizhe X., Zhihan L. 2023. Evolutionary computation-based machine learning for Smart City high-dimensional Big Data Analytics. Applied Soft Computing, 133, January 2023. https://doi.org/10.1016/j.asoc.2022.109955

Xu S., Noh H.Y. 2021. PhyMDAN: Physics-informed knowledge transfer between buildings for seismic damage diagnosis through adversarial learning. Down. System Signal process.

Xu Y., Wu L., Xie Z., Chen Z. 2018. Building extraction in Very High Resolution Remote Sensing Imagery Using Deep Learning and Guided Letters. Remote Sensing.

https://github.com/microsoft/seismic-deeplearning [accessed: 10.09.2024]