International Journal of Geo-Information

MDPI

Article

The Influence of Point Cloud Accuracy from Image Matching on Automatic Preparation of Training Datasets for Object Detection in UAV Images

Paulina Zachar *, Wojciech Ostrowski, Anna Płatek-Żak and Zdzisław Kurczyński

Department of Photogrammetry, Remote Sensing and Spatial Information Systems, Faculty of Geodesy and Cartography, Warsaw University of Technology, Pl. Politechniki 1, 00-661 Warsaw, Poland
* Correspondence: paulina.konarzewska.dokt@pw.edu.pl

Abstract: The dynamic development of deep learning methods in recent years has prompted the widespread application of these algorithms in the field of photogrammetry and remote sensing, especially in the areas of image recognition, classification, and object detection. Still, one of the biggest challenges in this field is the low availability of training datasets, especially regarding applications of oblique aerial imagery and UAV data. The process of acquiring such databases is labor-intensive. The solution to the problem of the unavailability of datasets and the need for manual annotation is to automate the process of generating annotations for images. One such approach is used in the following work. The proposed methodology for semi-automating the creation of training datasets was applied to detect objects on nadir and oblique images acquired from UAV. The methodology includes the following steps: (1) the generation of a dense 3D point cloud by two different methods: UAV photogrammetry and TLS (terrestrial laser scanning); (2) data processing, including clipping to objects and filtering of point clouds; (3) the projection of cloud points onto aerial images; and (4) the generation of bounding boxes bounding the objects of interest. In addition, the experiments performed are designed to test the accuracy and quality of the training datasets acquired in the proposed way. The effect of the accuracy of the point cloud extracted from dense UAV image matching on the resulting bounding boxes extracted by the proposed method was evaluated.

Keywords: training datasets; object detection; deep learning; point clouds; oblique imagery; UAV

Citation: Zachar, P.; Ostrowski, W.; Płatek-Żak, A.; Kurczyński, Z. The Influence of Point Cloud Accuracy from Image Matching on Automatic Preparation of Training Datasets for Object Detection in UAV Images. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 565. https://doi.org/10.3390/ijgi11110565

Academic Editors: Fabio Remondino, Joep Crompvoets, Norbert Haala and Wolfgang Kainz

Received: 31 August 2022
Accepted: 5 November 2022
Published: 10 November 2022

1. Introduction

With the development of deep learning and the continuous increase in GPU performance, convolutional neural networks (CNNs) have found wide applications in image recognition and object detection [1–4]. These algorithms are still being developed, and new deep learning models are being created. However, one problem that arises in this area is the requirement of a large number of ground-truth annotations to train the deep convolutional neural networks. Moreover, the process of acquiring such datasets is time-consuming. The acquisition of traditional training datasets involves data gathering and annotation. Although data are obtained all the time and new image data are provided every day, whether by aircraft, drones, mobile platforms, or sensors mounted on satellites, there are many reasons why these data may not be sufficient or helpful in training deep learning models. Still, one of the biggest challenges is preprocessing and preparing the acquired data so the model can be taught based on it.

The largest training databases for CNN consist of natural scenes. These are large datasets made available for public use. ImageNet [5], PASCAL VOC 2012 [6], and MSCOCO [7] are a few examples of learning datasets [8]. However, the limitation is that they can only be applied to specific scenes or as an input to pretrain models. The challenge is also that objects are mapped differently in aerial photographs compared with those in natural ground scenes. The low diversity of object classes, variability of scale, orientation, and

shape of objects on the Earth's surface are also limitations that make using these existing and well-archived datasets impossible in specific tasks. As for photogrammetry and remote sensing applications, manual annotation is usually unavoidable, which requires a lot of time and labor.

However, with the rise of data-intensive deep learning methods, the number of solutions and effective strategies for generating ground-truth data is increasing. One methodology that is also worth mentioning here is that proposed by Laupheimer and Haala, 2021 [9]. It involves transferring labels from the manually-annotated point clouds to the mesh and from that place to the image space for the semantic scene analysis task. This is one of the examples of methods and approaches that partially solve the problem of dataset inaccessibility and manual work. Several other examples are briefly described in Section 2 with a review of related works.

The following work also uses a process to semi-automate the creation of training datasets for detection in nadir and oblique aerial images. For this purpose, orthophotomaps and point clouds are used as a starting point for generating ground-truth bounding boxes on images. A more detailed description of the methodology is described in Section 4.

Although such an approach does not eliminate manual work, it reduces the effort. Creating a point shapefile layer and marking objects using an orthophotomap is much less time-consuming than labeling all photos in a project and marking all bounding boxes surrounding objects.

The main contribution in the following work, on the other hand, is the aspect of the accuracy and quality of the training datasets acquired in this way. The research aims to assess the influence of point cloud accuracy extracted from dense UAV imagery matching on the resulting bounding boxes extracted using the proposed method.

The rest of this paper consists of the following parts. Section 2 reviews the availability of training datasets and describes methods for automating the process of generating annotations for images. Section 3 describes the data used. Section 4 is a presentation of the methodology adopted in this work. Section 5 contains the results and an assessment of the accuracy. Finally, Section 6 is a summary of the work and includes conclusions.

2. Related Works

The spread of deep learning methods in Earth observation has resulted in the creation and availability of some datasets for training models on aerial and satellite images. However, in the case of training datasets consisting of aerial photographs or satellite scenes, their abundance is much smaller than, for example, the ImageNet. Moreover, these collections are rather characterized by a low diversity of object categories. The most frequent datasets include objects such as cars: TAS set [10], VEDAI [11], UCAS-AOD [12], the 3K-DLR-Munich [13]; ships: RSOD [14], HRSC2016 [15]; and buildings: the SZTAKI-INRIA dataset [16]. The two largest datasets for object detection in the Earth observation domain are DOTA [17], which consists of 15 categories of objects and 2806 aerial images, and DIOR [18], which contains 23,463 images and 192,472 instances, covering 20 object classes. Nevertheless, even such multiclass, mentioned datasets among these dozen or so classes/categories of objects may not contain those desired in a particular case. Sometimes it happens that these objects are particular for specific applications.

However, their availability of oblique aerial images is still low regarding training datasets for object detection. The potential that oblique photogrammetry brings is quite significant. Firstly, it provides a data source with distinct advantages: multiple views from different perspectives and significantly different image scales [19]. In addition, oblique photogrammetry carries the possibility of obtaining information about the location of an object in a terrain system and the use of the multitemporal feature [20].

The scientific community's interest in using oblique aerial photographs has made the advantages of this technique obvious. This is evidenced by the appearance of publications and scientific studies on object detection [1,21].

UAV photogrammetry is a cost-effective and flexible data acquisition approach that provides a data source of nadir and oblique images [22]. Aerial and drone photogrammetry also brings high-accuracy imagery of the same object several times in different photos due to the side and forward overlap in a block of images. Moreover, the use of oblique images gives the additional advantage that a given object is imaged from several directions at different angles, which further increases the size and diversity of the dataset. Using data acquired from UAVs and photogrammetric products such as point clouds or orthophotomaps enables the detection of objects using algorithms based on deep learning frameworks [23]. Due to the fast high-resolution data acquisition ability, a UAV-based system could be used in many fields. Such solutions can be applied in the inventory and modeling of technical and transport infrastructure objects. The interest in using deep learning algorithms for object detection has grown in such industries as railroads, power generation, and road construction [24–28]. Drone-based solutions are also applied to inspect solar panels [29].

The presented examples outline the potential of using UAV-acquired data to detect objects of interest using deep learning methods. The review papers [30,31] summarize the previous ones regarding the fundamentals of deep learning applied in UAV-based imagery. Ramachandran and Sangaiah highlighted the mentioned problem with the availability of the datasets in their work [30], and it was pointed out that it is essential for the progress of research in this field to create a large benchmark dataset dedicated to the problem of object detection by UAVs.

Similar to aerial and satellite imagery, publicly available training datasets for UAV applications most often include classes such as cars [21,32–34]. Training datasets for engineering infrastructure [35] or tree detection [36] are also starting to appear, but class size and diversity are still disappointing.

Therefore, research teams spend a lot of time creating such datasets or using other solutions such as fine-tuning-based approaches or other transfer learning methods. Alternative methods can also be used to speed up the manual process—weak supervision or Semi-Supervised Learning (SSL) [37,38].

Another approach to tackle the lack of training data is to automate the process of generating annotations for images. Such solutions are used in both image segmentation and object detection. In their work, Ros et al., 2016 [39], proposed to generate synthetic images with pixel-level annotations. A further proposition to solve this problem is using a LiDAR point cloud or 3D reconstruction of the scene to lift the semantic instance labeling task from 2D into 3D [40]. The authors in [41] proposed the automatic generation of annotations on images. The method consists of three steps: (1) manual labeling of one or two aerial images; (2) transferring the pixel labels to multiple UAV images via the UAV point cloud; (3) refining the generated annotations using a densely-coupled CRF model and naive Bayes classifier. In their study, Zachar et al., 2022 [20], addressed the lack of training data for the model and proposed a methodology where manual labor is replaced by the use of existing resources for transferring references to new databases for training models for detecting objects on oblique aerial images. Similar solutions with transferring references and adopting deep learning-based algorithms in natural scene images to detect objects in UAV images have already been proposed [42–45].

3. Dataset Description

For the experiments in the following work, photogrammetric data (aerial nadir and oblique images) were acquired with a DJI Phantom 4 RTK drone with the camera FC6310R (resolution: 5472 × 3648, focal length: 8.8 mm; pixel size: 2.41 µm), and products (point clouds and orthophotomaps) were used. The study area over which the photogrammetric data were acquired was a railroad section near Czestochowa, Poland (Herby). The data were acquired in March 2021. Oblique and nadir aerial images were acquired in multivariate photogrammetric missions. Different flight heights and forward and side overlaps of the photos were tested. Data were processed in Pix4D Mapper software for a section of railroad infrastructure in the test area.

Data acquired in multiple variations were used to generate point clouds. The selection of photogrammetric mission parameters was related to UAV data acquisition methodology experiments with input data requirements for neural networks. Experiments on this topic were not studied in the above publications. Nadir images were acquired at two heights (54 m and 90 m), while oblique images were acquired at 40 m with a tilt of camera angle by 45°. In addition, the variants differed in their forward and side overlap, which affects not only the resulting point cloud but also the processing time and economic aspect.

Due to different mission flight parameters (heights, coverages) and different parameter settings in the software, products with other characteristics were obtained. Fourteen scenarios for image acquisition and processing were prepared for the study area. The variants also differed in the use of vertical and oblique photos and their combinations. Not all variants used all acquired photos, and this was modified by excluding, for example, every second row or every second photo. Table 1 shows all the variants' summaries and basic parameters. As a result, 14 different point cloud variants were obtained.

Table 1. The summary of each variant's mission flight and processing parameters. The variants differ in flight altitude and images' overlap. The overlap between images varies according to the mission plan, and also by managing the inputs to the data processing. All of these parameters affect average ground sampling distance (GSD) and the number of images covering a given area. The last column indicates the point cloud generation settings.

No.	Type of Images Used (N—Nadir, O—Oblique) and Flight Altitude	Forward Overlap	Side Overlap	Average Ground Sampling Distance (GSD)	Number of Photos	Image Scale for Dense Image Matching
1.	N: 54 m	90%	90%	1.50 cm	146	half
2.						full
3.	N: 54 m	90%	80%	1.50 cm	74	half
4.	N: 54 m	90%	70%	1.50 cm	73	half
5.	N: 54 m	90%	90%	1.50 cm	147	half
6.	N: 90 m	90%	90%	2.46 cm	151	half
7.						full
8.	N: 90 m	90%	80%	2.46 cm	100	half
9.	O: 40 m	80%	90%	1.97 cm	92	half
10.	N: 54 m O: 40 m	N: 90% O: 80%	N: 80% O: 90%	1.50 cm	48	half
11.	N: 54 m O: 40 m	N: 90% O: 80%	N: 80% O: 90%	1.50 cm	87	half
12.						full
13.	N: 54 m O: 40 m	N: 90% O: 80%	N: 90% O: 90%	1.50 cm	114	half
14.						full

The resulting alignment accuracies are shown in the tables below (Tables 2 and 3). The average RMS error ranges for control points from 0.2 cm to 3.6 cm, while for check points, it ranges from 1.3 cm to 3.6 cm. The best accuracies were obtained for variant no. 3 (Flight Altitude: 54 m; OF: 90%; OS: 80%; GSD: 1.50 cm) comparing accuracies on control points. Variant no. 1 (Flight Altitude: 54 m; OF: 90%; OS: 90%; GSD: 1.50 cm) showed the best accuracy on check points. Bundle Block Adjustment Details are also indicated by the average of the reprojection error in pixels (Table 4).

Table 2. Summary of accuracy on control points for variants developed in Pix4D software.

Variant (Number Related to Table 2)	Control Points			
	RMS Error —X (cm)	RMS Error —Y (cm)	RMS Error —Z (cm)	RMSE (cm)
1. and 2.	0.7	0.2	0.4	0.48
3.	0.7	0.2	0.2	0.44
4.	0.8	0.8	0.4	0.69
5.	0.9	0.9	0.6	0.81
6. and 7.	0.7	1.0	0.6	0.79
8.	0.9	1.0	0.9	0.93
9.	1.4	0.3	0.7	0.94
10.	1.4	0.4	0.6	0.93
11. and 12.	0.8	0.5	0.3	0.57
13. and 14.	1.5	0.4	0.6	0.96

Table 3. Summary of accuracy on check points for variants developed in Pix4D software.

Variant (Number Related to Table 2)	Check Points			
	RMS Error —X (cm)	RMS Error —Y (cm)	RMS Error —Z (cm)	RMSE (cm)
1. and 2.	1.5	1.7	2.1	1.78
3.	1.7	1.6	2.2	1.85
4.	2.3	1.5	2.0	1.96
5.	1.8	1.5	2.1	1.82
6. and 7.	2.1	1.9	3.5	2.60
8.	2.2	2.0	3.6	2.70
9.	3.3	1.3	2.9	2.65
10.	2.1	1.6	1.9	1.88
11. and 12.	2.5	1.5	2.8	2.33
13. and 14.	2.1	1.6	1.9	1.88

Table 4. Bundle Block Adjustment Details. The average of the reprojection error in pixels.

Variant (Number Related to Table 2)	Average Ground Sampling Distance (GSD)	Mean Reprojection Error (Pixels)
1. and 2.	1.50 cm	0.132
3.	1.50 cm	0.186
4.	1.50 cm	0.127
5.	1.50 cm	0.129
6. and 7.	2.46 cm	0.193
8.	2.46 cm	0.184
9.	1.97 cm	0.133
10.	1.50 cm	0.125
11. and 12.	1.50 cm	0.178
13. and 14.	1.50 cm	0.125

For the experiment's performance, terrestrial laser scanning (TLS) point clouds were used as the ground truth (the reference data to which the results would be compared).

An important aspect from a research perspective was that the data were acquired on the same day because it was a railroad station reconstruction site. The dynamic changes that can occur from day to day in such an environment could make it impossible to compare results from data taken at different times. Then, changes in land cover—such as the placement of a new building—would additionally need to be verified. In the case of the experiments above, the data were confirmed in this regard, but it is a crucial point to keep in mind.

The area for which TLS data was acquired was smaller than the area covered by the UAV data. Therefore, the area for UAV data was also limited to the site for which TLS data was acquired. As a result, this area had 20 traction poles and 15 railroad gates. These objects were analyzed.

The terrestrial laser scanning (TLS) data collection was conducted using a Leica RTC360 scanner. For the study area, 115 scans were taken using the medium settings, corresponding to a point resolution of 6 mm at a distance of 10 m. The data were geo-referenced into the PL-1992 terrain coordinate system using ground control points. For this purpose, 13 points were measured by the RTK method, using the national reference network (ASG-EUPOS) with a measurement accuracy of 0.03 m (horizontal) and 0.05 m (vertical). Registration of the scans was performed using the Cloud-to-Cloud method in Leica Cyclone REGISTER 360 software. The accuracy of the C2C fit for the bundles was 1 cm. The average error of matching the point clouds acquired from different scanner locations to ground control points was < 10 cm. The accuracy values are shown in the Table 5.

Table 5. The accuracy of TLS point clouds alignment.

Overall Quality	Value
Bundle Error	0.010 m
Cloud-to-Cloud	0.010 m

An important aspect investigated in the present experiments is the accuracy of the generated point clouds from DIM (dense image matching). The alignment accuracies are presented above. However, in addition to the analysis of the alignment reports of the images of each variant, the point cloud densities and the visual analysis to evaluate the noise of the point clouds are also compared. A summary of the average point density per m^3 is shown in Table 6. The highest densities are indicated by the variants for which the point cloud was generated on high settings.

Table 6. Comparison of point cloud densities per m^3.

Variant (Number Related to Table 2)	GSD (cm)	Image Scale for Dense Image Matching	Average Density (per m^3)
1.	1.50	half image scale	1009.1
2.	1.50	full image scale	4289.32
3.	1.50	half image scale	879.69
4.	1.50	half image scale	902.52
5.	1.50	half image scale	1357.23
6.	2.46	half image scale	178.18
7.	2.46	full image scale	453.87
8.	2.46	half image scale	183.23
9.	1.97	half image scale	407.53

Table 6. *Cont.*

Variant (Number Related to Table 2)	GSD (cm)	Image Scale for Dense Image Matching	Average Density (per m^3)
10.	1.50	half image scale	644.97
11.	1.50	half image scale	904.32
12.	1.50	full image scale	973.78
13.	1.50	half image scale	418.37
14.	1.50	full image scale	1533.09

By subjecting all variants to visual analysis, it can be seen that a more significant number of elements were mapped into the point cloud for the variants generated on the high setting. However, point clouds from these variants are noisier. This is particularly evident for the traction poles. The following figures show examples of a gate (Figure 1) and a traction pole (Figure 2) for all point cloud variants.

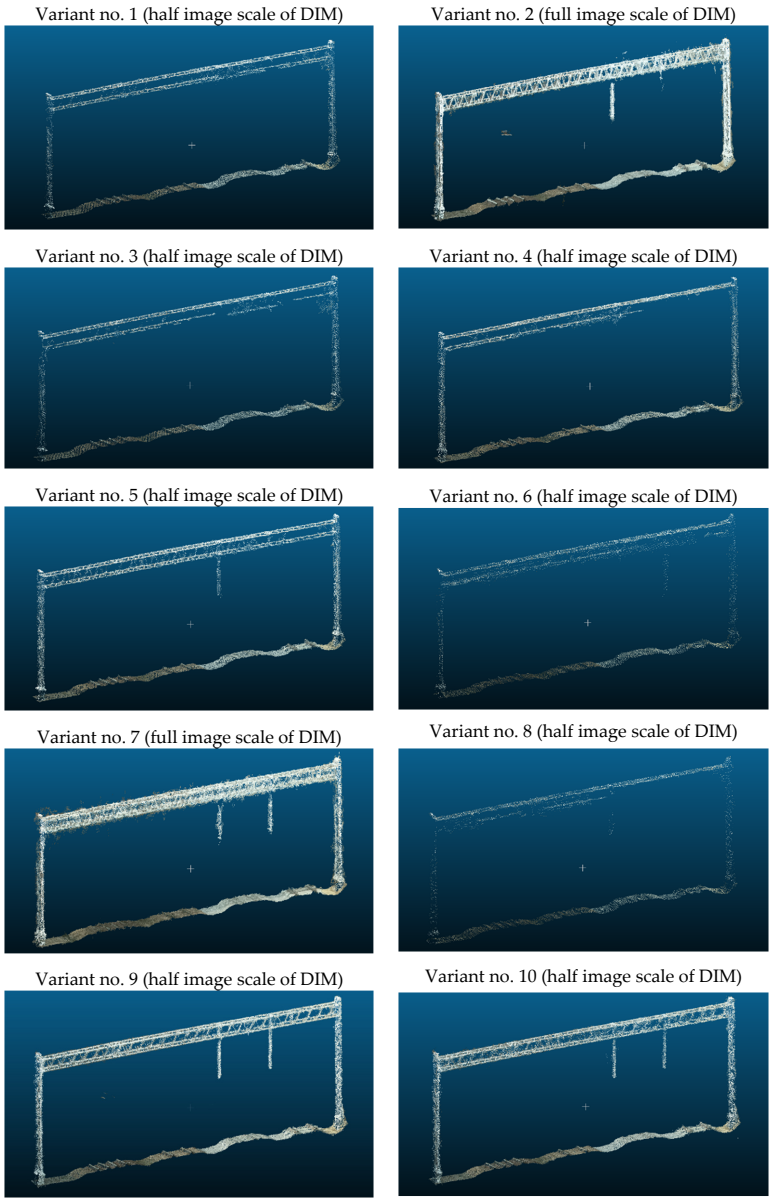


Figure 1. *Cont.*

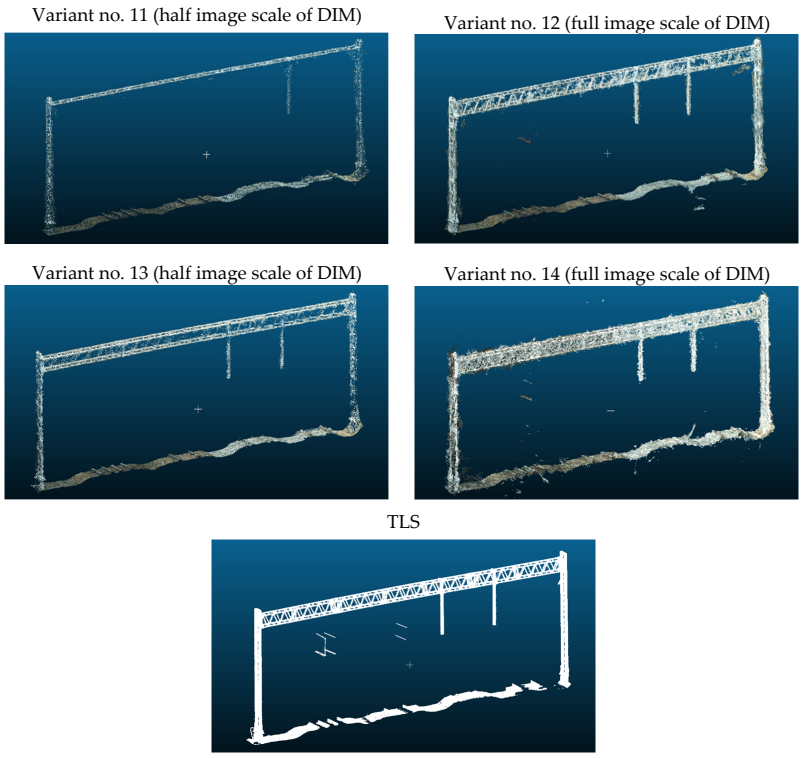


Figure 1. Point cloud fragments clipped to the gate object for all analyzed variants.

When comparing even the most accurate point cloud variants from dense image matching with terrestrial laser scanning data, it is apparent that some elements have not mapped into the dense point cloud. Furthermore, as more details are mapped and the density of the point cloud increases, noise increases. Such an effect is not desirable, especially when the next step is to project the points into pixel coordinates of the image to determine each object's precise location (bounding box).

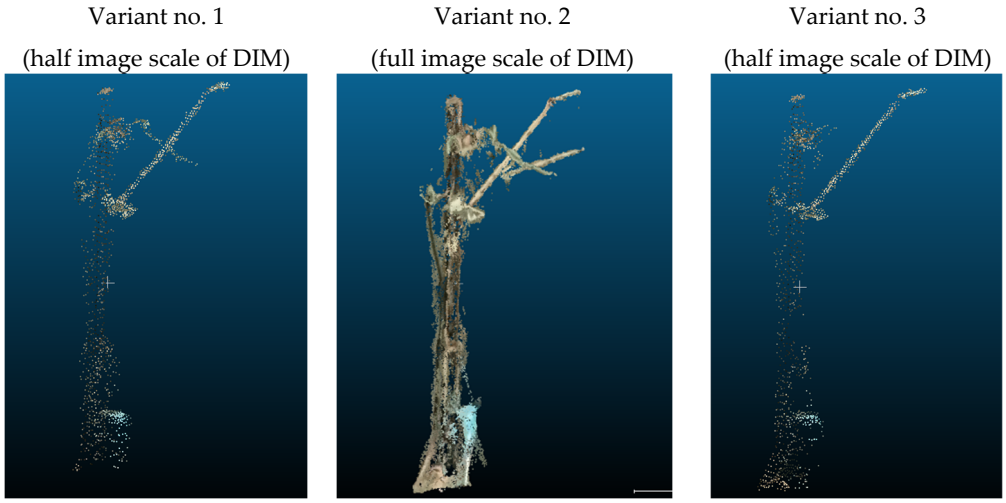


Figure 2. *Cont.*

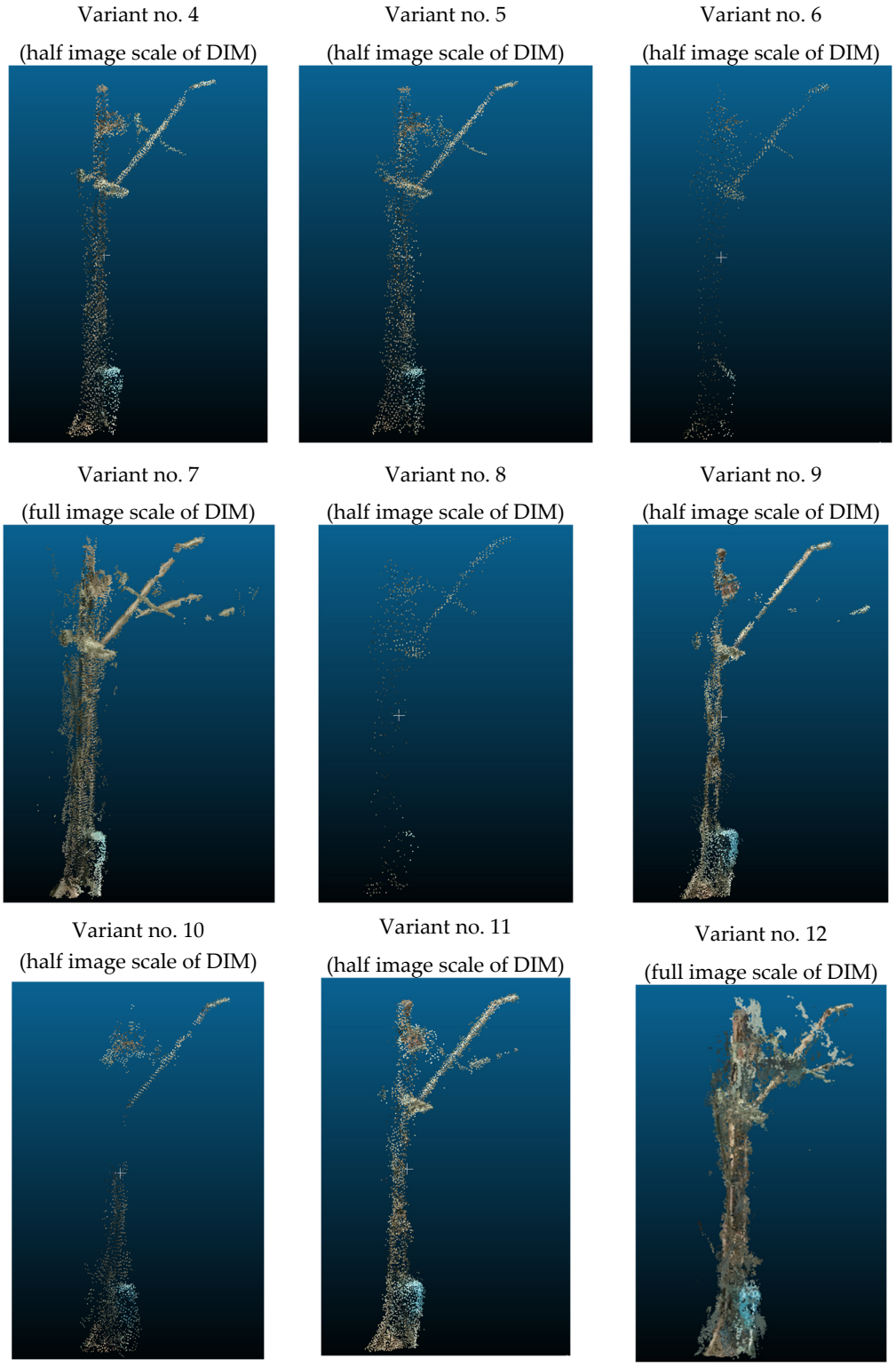


Figure 2. *Cont.*

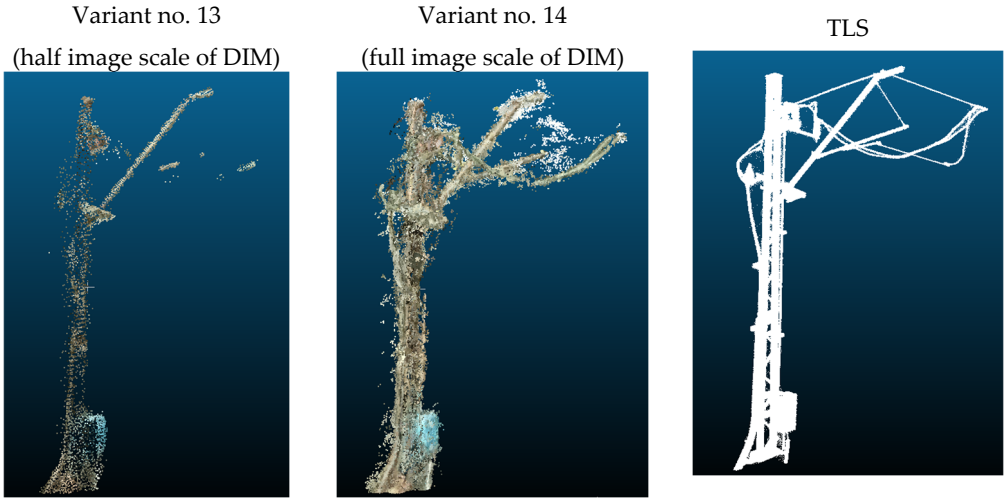


Figure 2. Point cloud fragments clipped to the traction pole object for all analyzed variants.

Taking into account all indicated aspects, experiments were conducted. The aim was to verify how the point cloud accuracy affects the resulting bounding boxes. As a result of research, recommendations have been made based on which the parameters of a photogrammetric mission flight should be used to automate the creation of high-accuracy training datasets. In addition to the mission flight parameters, essential elements are the data processing settings and parameters with which the point clouds should be generated in such a way that they are sufficient for the specified purpose.

In the following part of the paper, we describe in detail the particular steps of the research, the methodology adopted, and the results.

4. Methodology and Experimental Setup

The processing of UAV-acquired images produces various photogrammetric products, such as point clouds and orthophotomaps. These products were used as source data as part of the methodology to support the preparation of training datasets.

The first essential step was to process the image blocks; that is, to orient the data. The results from this step were described in an earlier section, where the alignment accuracies for each variant were also presented. As part of the data processing, point clouds and orthophotomaps for all variants were created in Pix4D.

The next part was the preparation of files containing information on the location of objects of interest. This step was necessary because of the need to have the terrain coordinates of each object so that, based on them, the point clouds from dense image matching could be clipped to the cloud fragments containing the object. Two different approaches were used for gates and traction poles. Based on the orthophotomap taken from one of the image variants, a point layer was created in ArcGIS Pro. The traction poles were marked with points on the orthophotomap, which made it possible to capture the information of the pole's X, Y, and Z terrain coordinates (Figure 3a). As for the gates, it was decided to create a polygon layer due to the different characteristics of the object. An example of a vectorized gate can be seen in Figure 3b.

The resulting terrain coordinates of all the objects served as input information for cutting the sections from the point cloud that contained points belonging to the object (Figure 4). In the case of gates, the point clouds were cropped with a polygon, while for traction poles, a buffer from a point with a radius of 5 m was used, so there was assurance that all traction pole elements would be mapped including the booms.

(**a**) (**b**)

Figure 3. Examples of annotation data for (**a**) traction poles annotated as points and (**b**) gates labelled as polygons based on orthophotomap from UAV images.

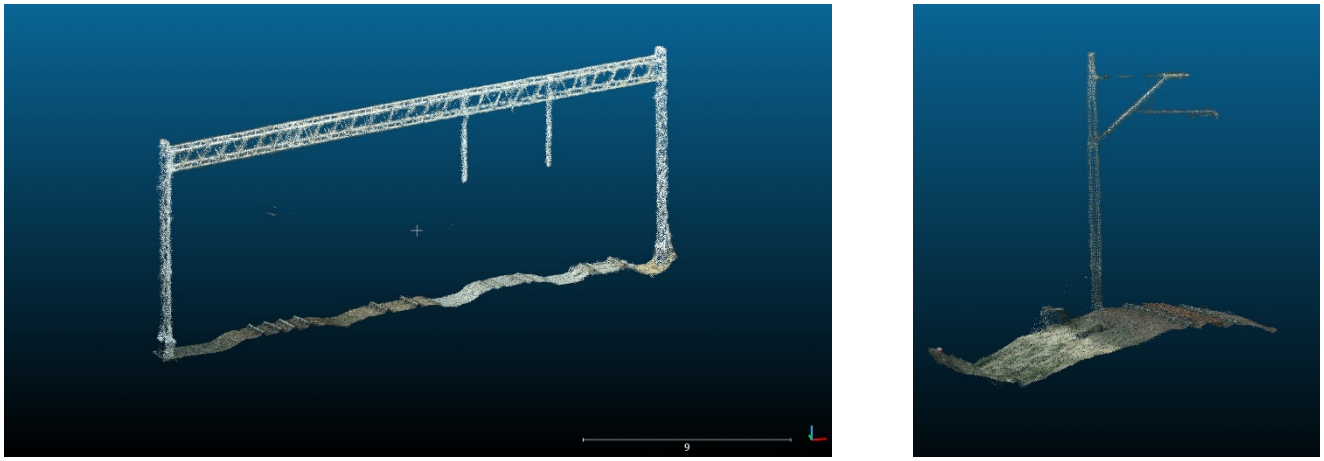


Figure 4. Examples of objects cut from a point cloud.

As can be seen in Figure 4., the cropped point clouds still contain elements that do not belong to the objects of interest (including points belonging to the ground class). Thus, ground filtering proved necessary for the bounding boxes surrounding the object to be well represented. The Cloth Simulation Filter (CSF) method, used to extract ground points in clouds and process LiDAR data, was decided upon. However, after testing (initially in CloudCompare software) and verifying the results, it was found that for the purposes of the above experiments, this method is also relevant for point clouds from dense image matching, as shown in Figure 5. Therefore, the CSF filter [46] implementation provided on GitHub (https://github.com/jianboqi/CSF, accessed on 1 November 2022) was used and added as a component of our algorithm.

In addition to filtering out ground points, it was also necessary to filter out points that were noise and did not belong to the object of interest. First, "isolated points" noise filtering was applied using Open3D library functions. These methods examine the neighborhood in the point cloud and reject outliers on this basis. Two methods were used:

- statistical outlier removal—removes points that are further away from their neighbors compared to the point cloud average;
- radius outlier removal—removes points that have few neighbors in a given surrounding space.

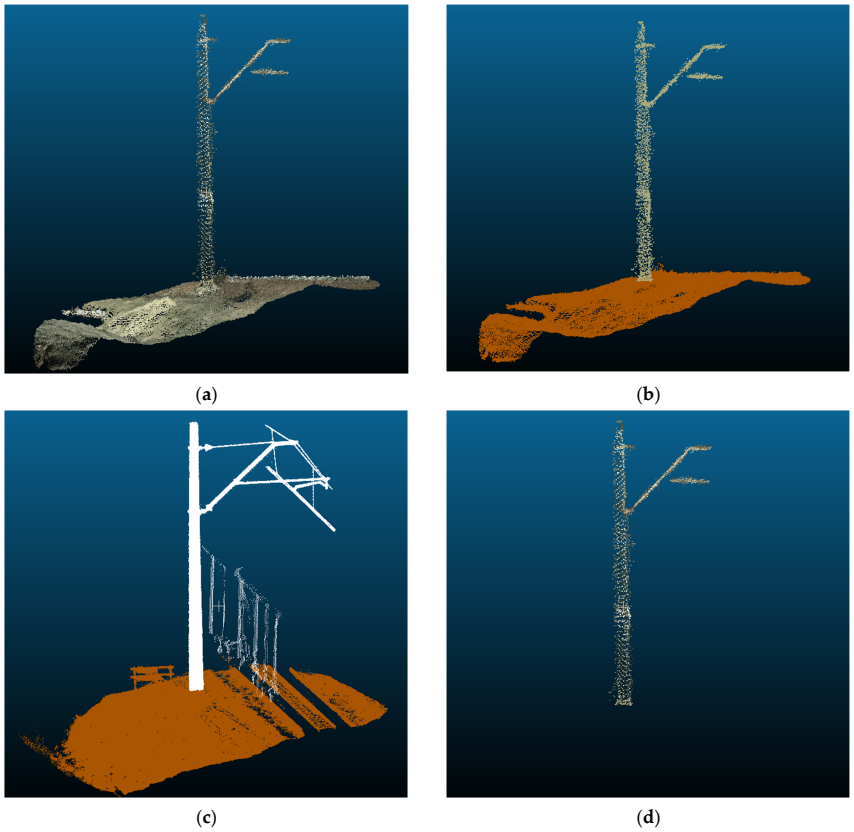


Figure 5. Cloth Simulation Filter (CSF) ground filtration examples: (**a**) an example of the source cloud, (**b**) the same object after ground filtration colored by classes—ground and non-ground, (**c**) an example of ground filtration for a point cloud with TLS, (**d**) the final segment of the point cloud passed to the next stage of the algorithm.

An example of outlier filtering for gates is shown in Figure 6, where the red points represent noise.

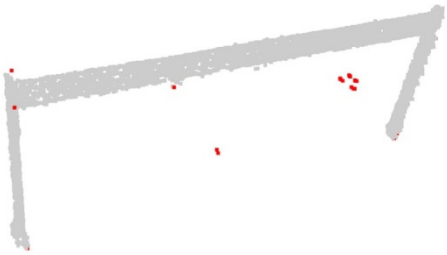


Figure 6. An example of noise filtering for a gate, where the red points are noise and the gray points are the object.

While the aforementioned filters worked well for gates, which generally had less noise, manual filtering was necessary for traction poles, which turned out to be more complex objects. An example of such a case is shown in Figure 4c, where objects that do not belong to the object of interest (the cloud points on the right) have also been mapped in the cut-out fragment of the point cloud (in the buffer of 5 m from the pole location point). It was decided that, despite the initial filtering using Open3D library functions, all objects would be verified for each variant, and unnecessary objects would be removed since no optimal tool was found that would automatically remove such fragments of the cloud. Manual verification and editing (filtering) were applied to both TLS point clouds and point clouds from UAV image matching. After making sure that all the point clouds for the objects

were correctly prepared, the final step occurred. This consisted of projecting the cloud points onto the image, basically converting the terrain coordinates into pixel coordinates of the images.

Based on the parameters of the camera's internal and external orientation, as well as information about the camera's distortion model, it was possible to make a precise transition between the coordinates in the field reference system (XYZ) and the pixel coordinates of the image (uv). Using this information and performing calculations using the files created as a result of the alignment in Pix4D, based on the mathematic model explained on the developer's website (https://support.pix4d.com/hc/en-us/articles/202559089-How-are-the-Internal-and-External-Camera-Parameters-defined; accessed on 1 November 2022 https://support.pix4d.com/hc/en-us/articles/202977149-What-does-the-Output-Params-Folder-contain, accessed on 1 November 2022), the result shown in Figure 7 was obtained.

Figure 7. An example of projecting a point cloud into a photo.

Having the cloud points transformed into pixel coordinates of the photo, it was possible to carry out the last part; that is, to calculate the coordinates of the bounding box. These values were estimated based on the maximum and minimum values of the projected points (umax, vmax, umin, and vmin). This is how the final result was created. The bounding box surrounding the object was obtained by transferring the point cloud to the images (Figure 8).

The methodology (Figure 9) thus developed involves using photogrammetric data and products as source data for object annotation. Then all the steps described in this section are carried out. The final result is numerous training datasets consisting of images and information about the object's position in the image by means of a bounding box saved in a text file according to the requirements under ML detectors and models.

For example, the resulting training databases could be used to train detectors such as YOLO or Fast R-CNN. On the other hand, an important issue is evaluating the accuracy of the resulting bounding boxes. This part related to the evaluation of the results is addressed in the next section, where the overlap between the reference (ground truth) and the bounding box, which is the result obtained by transferring the point cloud to the images, is examined.

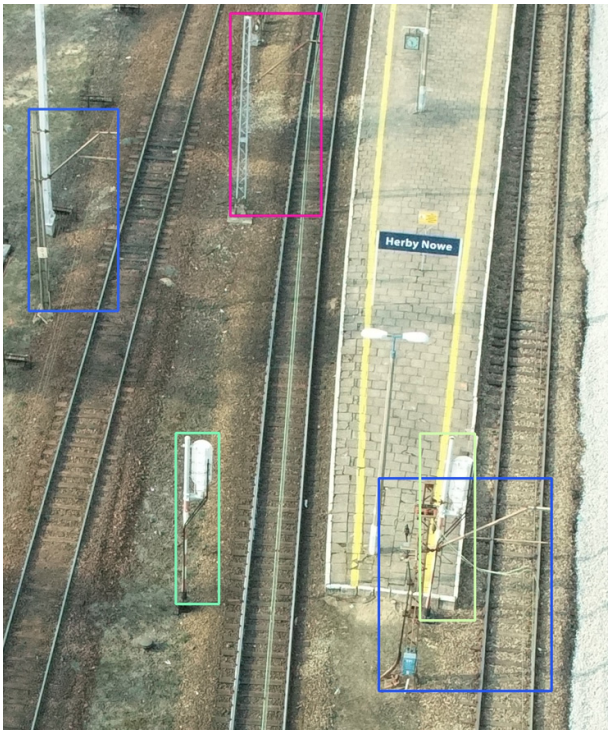


Figure 8. The final result is an example of training dataset for traction pole detection.

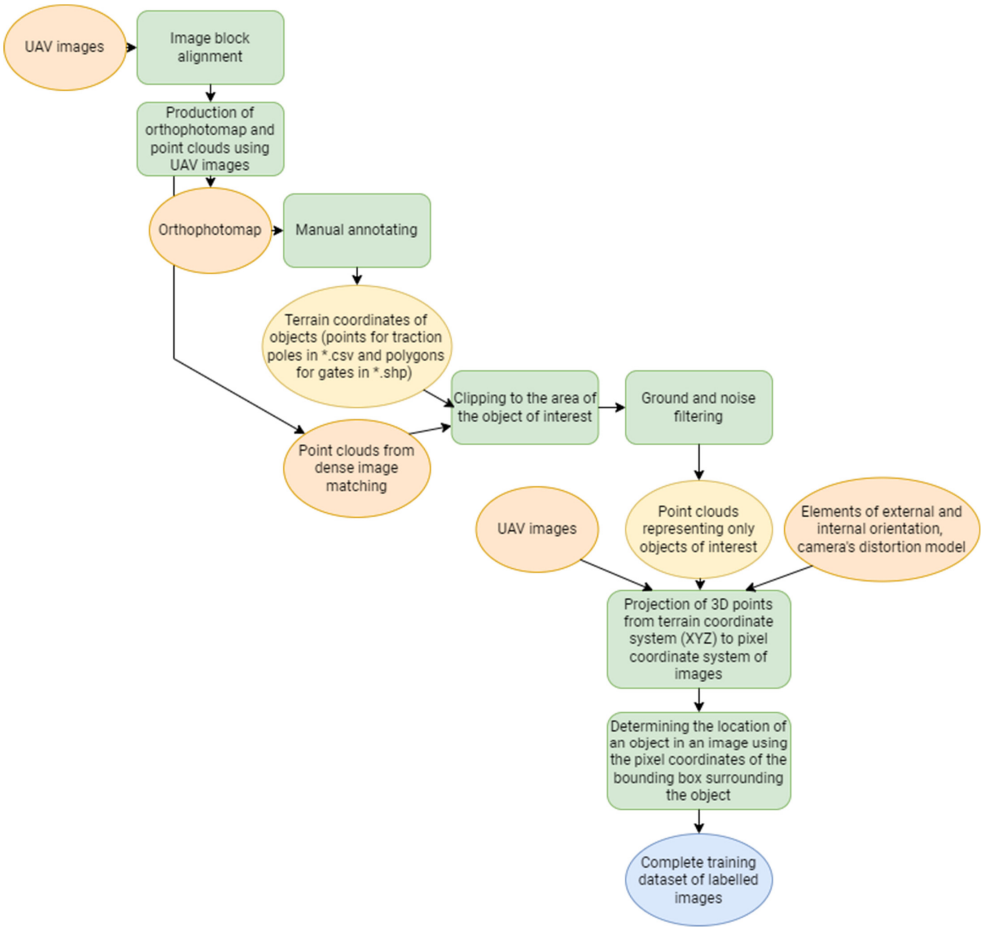
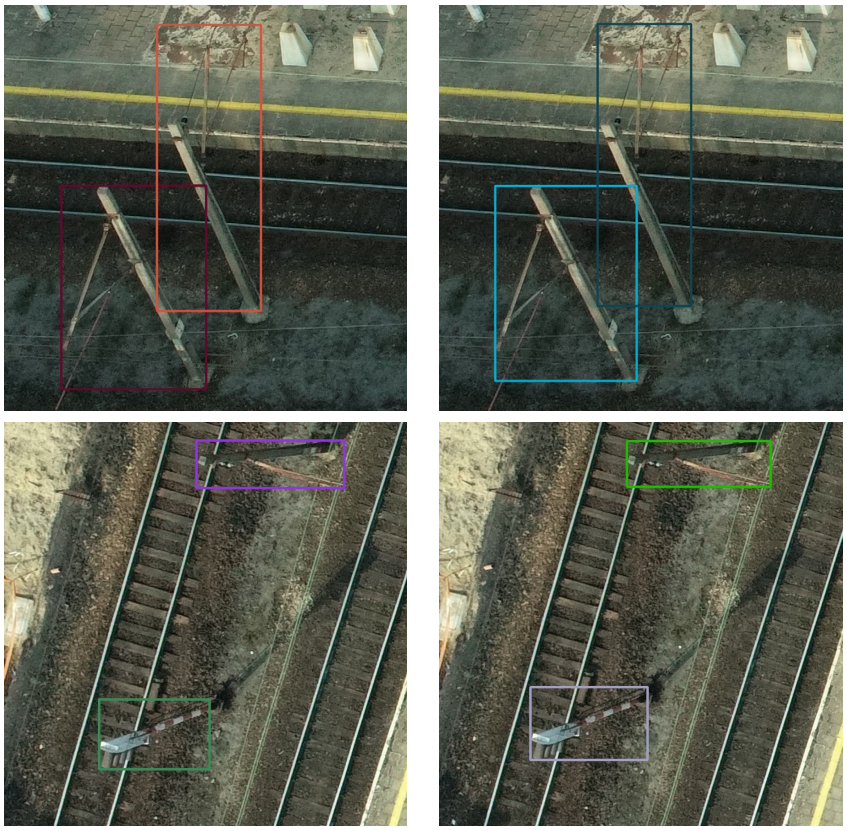


Figure 9. Workflow to semi-automate the process of preparing training datasets for object detection.

As mentioned earlier, the experiments were conducted on two objects of railroad infrastructure—gates and traction poles. The following section shows sample results—bounding boxes plotted on the images and bounding objects of interest. Visual inspection of the results also formed part of the analyses.

Figure 10 shows examples of the results for the traction poles. The right side shows results for point clouds from TLS, and the left side shows bounding boxes generated from point clouds from dense image matching. A similar visual comparison was made for the second object analyzed—gates (Figure 11).



(a) (b)

Figure 10. Example results for traction poles: (**a**) bounding boxes obtained from point cloud from DIM, (**b**) ground-truth bounding boxes obtained from point cloud from TLS.

The resulting bounding boxes obtained in this way can constitute a training dataset. Thus, the input to the network will be an image or a fragment of a photo containing the object of interest, and a file, for example, a text file containing information about the object's position in the pixel coordinate system. If a photo involves more than one object, all objects should be included in the file containing bounding box information to complete the collection. An important point to emphasize here is that by marking an object on an orthophotomap at one time, it is possible to obtain numerous training datasets as a given object may be visible in up to a dozen images. This depends on the parameters of the flight mission. This is described more extensively in the next section, including the number of bounding boxes obtained on all the images of a variant. The presented methodology was used to conduct experiments, the results of which are described in the next section.

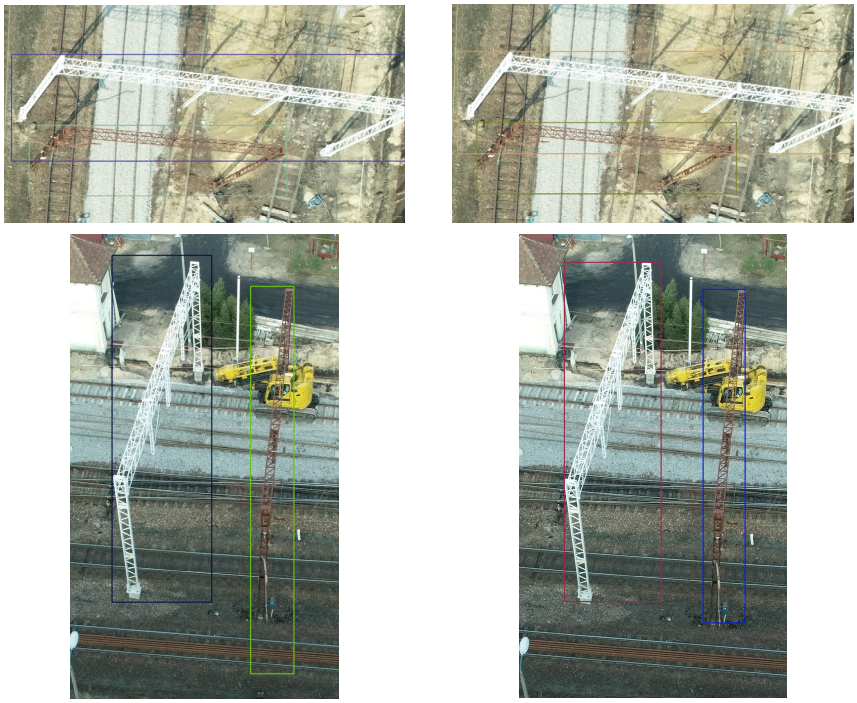


Figure 11. Example results for gates: (**a**) bounding boxes obtained from point cloud from DIM, (**b**) ground-truth bounding boxes obtained from point cloud from TLS.

5. Results

An important issue in evaluating the accuracy of deep learning models is the quality and accuracy of the training dataset. As is well known, the accuracy of an object detection model depends on the quality and number of training samples, input images, model parameters, and the required accuracy threshold. Therefore, the accuracy, correctness, and completeness of the bounding boxes for training the model are crucial when automating the training dataset generation process.

When interpreting the deep learning model results for object detection, the accuracy is evaluated using different metrics. One of them is the Intersection over Union (IoU) factor. This is a metric used as a threshold for determining whether a predicted result is a true positive or a false positive. This coefficient determines the overlap between the predicted bounding box around the object and the ground-truth bounding box. In this way, IoU can be used as a metric to evaluate the accuracy of an object detector on a particular dataset by comparing the results from the model to the reference data.

As previously mentioned, the main contribution of this paper is to analyze the impact of the accuracy of point clouds extracted from the nadir and oblique image matching on the resulting bounding boxes extracted automatically. Thus, in order to be able to evaluate the influence of the accuracy of point clouds extracted from image matching on the resulting bounding boxes, in addition to the visual assessment, the metric Intersection over Union was used (Figure 12).

This metric is used for the accuracy assessment of object detectors on a given set of data. Each detector, which provides bounding boxes as an output from the model, can be evaluated with the IoU metric. To apply this metric, it is necessary to have:

- ground-truth bounding boxes (labeled bounding boxes from the test dataset, which specify the location of the object on a pixel coordinate system of the image);
- predicted bounding boxes (output bounding boxes from the model). Based on these values, it is possible to evaluate the overlap between the reference (ground truth) and the bounding box, which is the result obtained from projecting cloud points onto images. IoU is therefore measured as the area of intersection of the ground-truth

bounding box with the output bounding box divided by the area of the combination of the two.

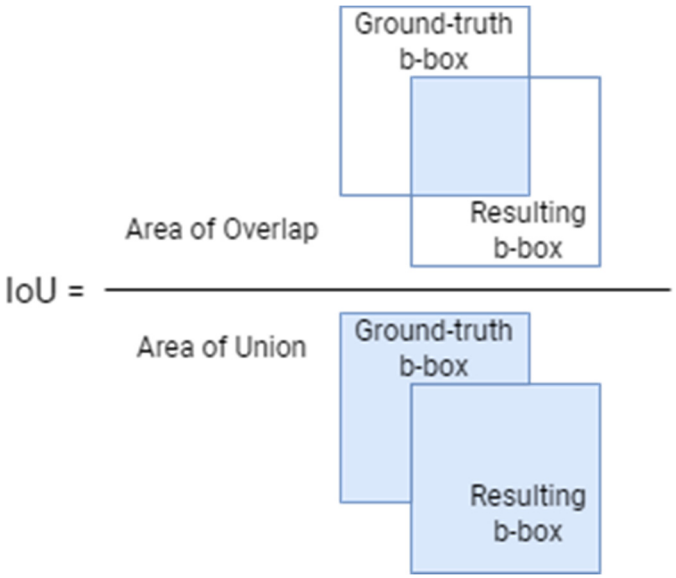


Figure 12. Intersection over Union.

In the numerator, there is the area of intersection of the predicted bounding box and ground-truth bounding box, and the denominator is the total area of the predicted bounding box and ground-truth bounding box. Dividing the area of overlap by the area of union yields the final result—Intersection over Union. The IoU value ranges from 0 (no overlap) to 1 (the boxes are identical).

Due to the fact that no object detector was used in the above experiments, the ground-truth bounding boxes were treated as a reference (obtained based on point clouds from the terrestrial laser scanning), while the resulting bounding boxes (obtained based on point clouds from dense image matching) were treated as "predicted". Both ground-truth and the resulting bounding boxes were obtained using a semi-automation script, as described in the section above.

As a result, all 14 variants were compared with ground-truth bounding boxes generated from TLS.

5.1. Gates

The first evaluated objects were traction gates. The experiment results are presented in Table 7, where the accuracies for all variants are summarized. The best results were obtained for the variant marked no. 5, for which the IoU value was 92.42%. The results obtained for the rest of the variants also showed relatively high accuracies.

The lowest IoU value was obtained for the variant marked no. 14—77.06%. A regularity that can be observed from the results is that for the variants for which the point cloud was generated with the "high" settings, lower values were obtained than for the variants with the default settings. After analyzing the experiment results, it can be concluded that the higher noise, which was obtained for the dense point clouds generated with "high point density" settings, significantly influenced the results.

Based on the results, there is a possibility to indicate a mission flight variant, for which resulting bounding boxes demonstrated the highest accuracies relative to references. Furthermore, the threshold that is used to distinguish a valid result can be defined (more precisely, what is (or is not) a "good" match). In that case, the threshold metric is the IoU value. Changing the score threshold allows the false positive and true positive rates to be distinguished to create a high-quality training dataset. By doing so, it is possible to discard

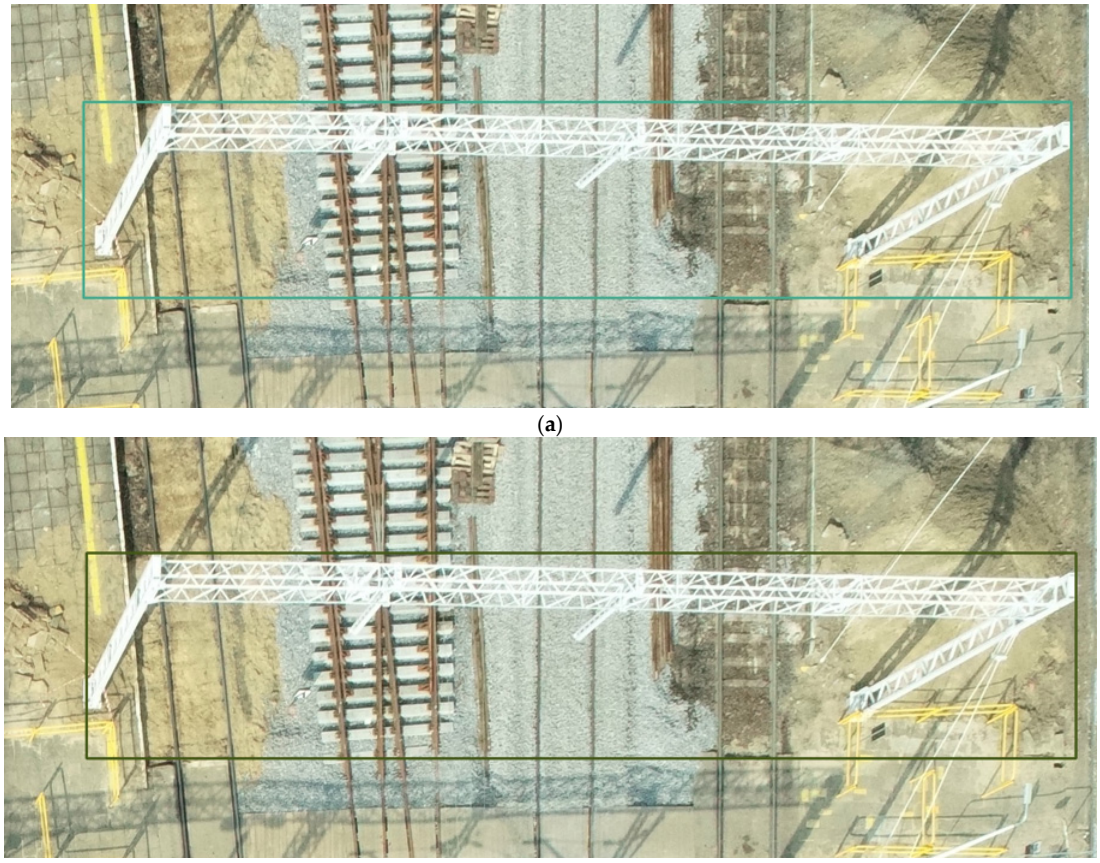
erroneous results and highlight those that possibly need manual correction before being passed to the model. More about the threshold value is described in Section 5.3.

Table 7. Accuracies of the resulting bounding boxes for the gates for fourteen point cloud variants. The table values are the average IoU (Intersection over Union) multiplied by 100%, which is the evaluation metric.

No.	Type of Images Used (N—Nadir, O—Oblique) and Flight Altitude	Image Scale for Dense Image Matching	Forward Overlap	Side Overlap	Average Ground Sampling Distance (GSD)	Number of Instances of Objects in the Images	IoU
1.	N: 54 m	half	90%	90%	1.50 cm	505	92.09%
2.	N: 54 m	full	90%	90%	1.50 cm	512	87.58%
3.	N: 54 m	half	90%	80%	1.50 cm	348	91.49%
4.	N: 54 m	half	90%	70%	1.50 cm	490	92.03%
5.	N: 54 m	half	90%	90%	1.50 cm	1150	**92.42%**
6.	N: 90 m	half	90%	90%	2.46 cm	476	91.44%
7.	N: 90 m	full	90%	90%	2.46 cm	486	83.31%
8.	N: 90 m	half	90%	80%	2.46 cm	312	88.49%
9.	O: 40 m	half	80%	90%	1.97 cm	2797	87.39%
10.	N: 54 m O: 40 m	half	N: 90% O: 80%	N: 80% O: 90%	1.50 cm	2015	88.73%
11.	N: 54 m O: 40 m	half	N: 90% O: 80%	N: 80% O: 90%	1.50 cm	2026	86.52%
12.	N: 54 m O: 40 m	full	N: 90% O: 80%	N: 80% O: 90%	1.50 cm	1802	80.83%
13.	N: 54 m O: 40 m	half	N: 90% O: 80%	N: 90% O: 90%	1.50 cm	385	79.37%
14.	N: 54 m O: 40 m	full	N: 90% O: 80%	N: 90% O: 90%	1.50 cm	391	**77.06%**

Thus, it can be stated that high point cloud density settings gave slightly worse results. Moreover, such a point cloud requires up to four times more processing time and RAM than optimal density.

Below are examples of the results obtained as a result of the experiments carried out—applied to the images' bounding boxes that surround the objects. Examples of correct results with high IoU values (Figure 13) and worse results (Figure 14) that would require possible manual improvement before inclusion in the training of the model are shown.



(**a**)

(**b**)

Figure 13. An example of a training dataset for gates with IoU = 0.90; (**a**) the bounding box generated from DIM point clouds; (**b**) ground-truth bounding box from TLS point cloud.

(**a**)

Figure 14. *Cont.*

(**b**)

Figure 14. An example of a training dataset for gates with IoU = 0.73; (**a**) the bounding box generated from DIM point clouds; (**b**) ground-truth bounding box from TLS point cloud.

5.2. Traction Poles

As for the results for the second object, they differ from those obtained for the gates. The traction poles turned out to be more complex objects that mapped less well to the point clouds than the gates. As for the gates, the accuracies for traction poles are summarized in the Table 8.

Table 8. Accuracies of the resulting bounding boxes for the traction poles for fourteen point cloud variants. The table values are the average IoU (Intersection over Union) multiplied by 100%, which is the evaluation metric.

No.	Type of Images Used (N—Nadir, O—Oblique) and Flight Altitude	Image Scale for Dense Image Matching	Forward Overlap	Side Overlap	Average Ground Sampling Distance (GSD)	Number of Instances of Objects in the Images	IoU
1.	N: 54 m	half	90%	90%	1.50 cm	611	69.72%
2.	N: 54 m	full	90%	90%	1.50 cm	613	68.83%
3.	N: 54 m	half	90%	80%	1.50 cm	413	70.43%
4.	N: 54 m	half	90%	70%	1.50 cm	552	73.15%
5.	N: 54 m	half	90%	90%	1.50 cm	1360	76.99%
6.	N: 90 m	half	90%	90%	2.46 cm	626	**68.64%**
7.	N: 90 m	full	90%	90%	2.46 cm	627	69.05%
8.	N: 90 m	half	90%	80%	2.46 cm	414	69.33%
9.	O: 40 m	half	80%	90%	1.97 cm	3930	**87.54%**
10.	N: 54 m O: 40 m	half	N: 90% O: 80%	N: 80% O: 90%	1.50 cm	2687	81.79%
11.	N: 54 m O: 40 m	half	N: 90% O: 80%	N: 80% O: 90%	1.50 cm	2711	78.22%
12.	N: 54 m O: 40 m	full	N: 90% O: 80%	N: 80% O: 90%	1.50 cm	2488	84.96%
13.	N: 54 m O: 40 m	half	N: 90% O: 80%	N: 90% O: 90%	1.50 cm	486	73.38%
14.	N: 54 m O: 40 m	full	N: 90% O: 80%	N: 90% O: 90%	1.50 cm	483	79.29%

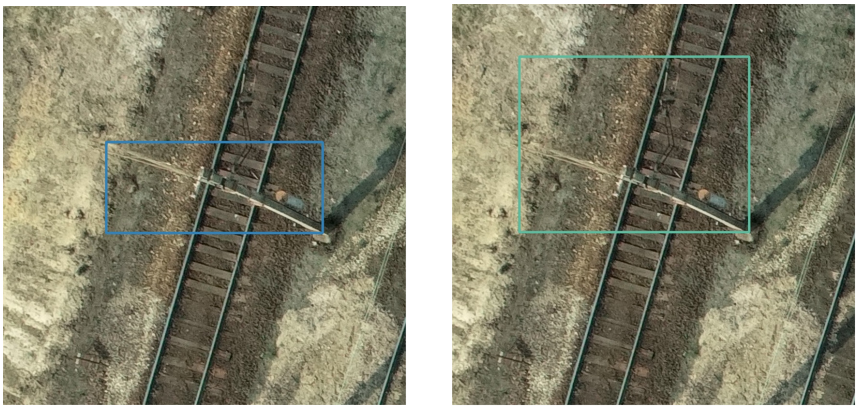
The best results were obtained for variant marked no. 9, for which the IoU value was 87.54%. For the rest of the variants, the results showed accuracies of 0.75. The lowest IoU value was obtained for variant marked no. 6—68.64%. The rule that was noticed in the case of the experiments for gates, that for point clouds generated on high settings, worse values were obtained, did not occur in the case of traction poles. Here, higher accuracies were obtained for three out of four variants on high settings.

Analyzing the accuracy results for the traction poles, it can also be noticed that for variants where in addition to the nadir photos the oblique images were used, much better results were obtained. Thus, based on these results, it can be concluded that for more complex objects extending above the terrain, such as traction poles, it is appropriate to include oblique photos when generating point clouds.

A similar summary to that for gates was presented for traction poles. The correct result had an IoU of 0.94, where the differences in bounding boxes are almost invisible to the eye (Figure 15). An example where part of the traction pole was probably not mapped to the point cloud from the dense image matching resulted in an IoU of only 0.48 (Figure 16).

(**a**) (**b**)

Figure 15. An example of a training dataset for traction poles with IoU = 0.94; (**a**) the bounding box generated from DIM point clouds; (**b**) ground-truth bounding box from TLS point cloud.



(**a**) (**b**)

Figure 16. An example of a training dataset for traction poles with IoU = 0.48; (**a**) the bounding box generated from DIM point clouds; (**b**) ground-truth bounding box from TLS point cloud.

5.3. General Considerations

For both types of analysed objects, no correlation or significant effect of forward and side overlaps and other mission flight parameters (such as flight altitude) were observed.

As stated before, IoU values are usually expressed in percentages; the most used threshold values are 50% and 75% [47]. However, such values are used as detection evaluation metrics to quantify the performance of detection algorithms in different areas and

fields. Thus, the question arises about what threshold to apply to the above methodology. Obviously, the training dataset obtained should automatically be as accurate as possible. Therefore, to define an IoU threshold for individual objects, a more precise evaluation would have to be carried out, aiming to indicate from which value the result should still be manually checked and corrected. Such a study could be an extension of the above experiments.

The overall conclusion is that it can be seen that the influence of mission flight type, including image acquisition parameters and processing parameters, is significant to the quality of point clouds.

An important achievement is the size of the training dataset. A summary of the instances of objects in the images for all variants can be seen in the above tables (Tables 7 and 8). A total of 35 objects (20 traction poles and 15 gates) were vectorized, and the final count of the set of instances in the photos for some variants was even about three thousand. Thus, in general, it is possible to use the methodology to automate the creation of training datasets as much as possible.

Figures 17 and 18 show cases for the whole high-resolution images with ground-truth annotations and the resulting bounding boxes obtained from the process.

Figure 17. An example of comparing bounding boxes for gates obtained using automatic generation from ground-truth for the entire image. From the top, IoU values: 0.9935, 0.9698, 0.9629.

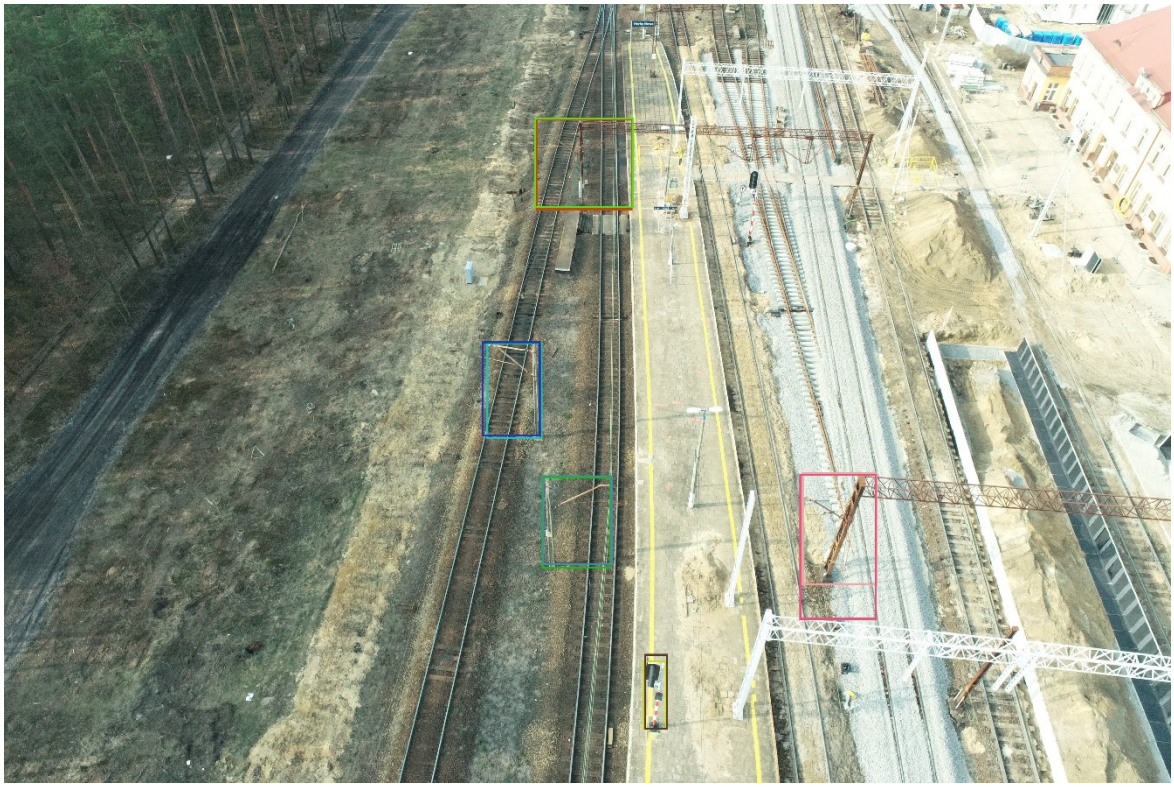


Figure 18. An example of comparing bounding boxes for traction poles obtained using automatic generation from ground-truth for the entire image. From the top, IoU values: 0.9748, 0.9116, 0.9675, 0.7411, 0.8496.

6. Conclusions

Annotating the ground-truth data and gathering the training datasets have a crucial relevance for training models, supervised object detectors, and deep learning approaches. This is tedious, time-consuming, and labor-intensive work, resulting in a lack of ground-truth datasets. The necessity to train models with a vast amount of independent annotated data (usually on the order of tens of thousands) is still an open problem that limits the use of the potential CNN carry. This is noticeable in many applications, and also for datasets to train models to detect objects in UAV images.

Exploiting the potential of available deep learning methods for object detection is the reason for developing the topic of automating the acquisition of training datasets. Thus, the need to apply the proposed approach is apparent, and the above work highlights the utility of automating this process as much as possible.

Adopting the strategy proposed in this paper makes data labeling easier. The method makes it possible to create bounding boxes based on points indicating only the location of the object, and thanks to the fact that it is common in photogrammetry to use overlaps in a block of images, the object transfer is performed on a few or a dozen images representing the object. Although the described approach does not exclude manual annotation at the beginning of the process, it significantly reduces time. Moreover, there is nothing to prevent the locations of existing objects from being used from already created databases, which would overcome manual work. Open tools for manual annotation exist and are quite widespread (for example, LabelMe), but preparing training datasets in this way is very time-consuming.

The above method, or others described in Chapter 1 using semi-automation processes, can help overcome the problem of acquiring training datasets. This process minimizes the manual effort involved in generating ground truth and significantly supports and promotes the training of deep learning algorithms.

An important point to highlight is the number of training datasets created using the above methodology. Using both vertical and oblique images, we obtain a dataset that maps the same area and the same objects but from different perspectives. Therefore, by vectorizing the object once on the orthophotomap, we obtain a collection consisting of a larger number of images using the more significant methodology described above. For example, for twenty traction poles for the variant numbered 9, for which the highest accuracies were obtained, a total of 3930 instances were obtained in the images.

This approach, of course, could also be applied using an orthophotomap cut into smaller tiles instead of images. However, the size of such a collection would be much smaller than if all the images in the block were taken since each object would be imaged only once. However, it is worth keeping in mind that with few changes in the code, this method, after some adaptations, can also be applied for orthophotomaps, the use of which has many advantages.

A particularly important issue in the context of the automatic creation of training datasets is the accuracy of the input data. This approach should pay attention to the accuracy of the point clouds included in the process, as it affects the resulting bounding boxes, essentially the prepared training dataset.

However, it has been shown that even with the use of slightly noisy point clouds, it is possible to achieve a reasonably accurate set, possibly requiring few manual corrections. Thus, it can be concluded that this method can be used analogously to the labeling methods mentioned above, where automatic data annotation is performed first, and then verification and possible improvement are performed later. The lowest accuracies obtained on the test dataset were about 70%, and for some variants, over 90%.

Based on the experiments, it can be said that the proposed approach is optimal in terms of performance and processing speed. It provides a semi-automated way to achieve consistent labels for all images in a block, reducing the effort of manually labeling each object in the photos. This is accomplished by starting with labeling the objects once on the orthophotos and then proceeding by clipping the point cloud to a fragment of the object of interest and projecting the points belonging to the object onto the pixel coordinates of the photo; the finished result is a bounding box surrounding the object. It is worth pointing out again that the step related to the manual marking of the location of objects using an orthophotomap is one of the options. We can start from terrain points, but extracted, for example, from a database of topographic objects or using BIM data. Manual labeling of bounding boxes on thousands or hundreds of thousands of images is much more time-consuming. In addition, point clouds from image matching or orthophotos are products often generated in the production process, which marks this strategy's potential. Naturally, the proposed methodology can also be adapted to other types of data (not only from UAV). The approach could also use point clouds from dense matching of aerial images, or point clouds acquired from laser scanning from different ceilings (both aerial and ground). Admittedly, the methods used are not a discovery but simple calculations, but the added value here is the ordering of strategy and accuracy analysis experiments.

In summarizing, based on the above results, it can be concluded that it is reasonable to use the developed methodology to semi-automate the process of creating datasets for training deep learning models for object detection in nadir and oblique images acquired from UAVs.

Author Contributions: Paulina Zachar proposed the methodology and designed the experiments. Wojciech Ostrowski was responsible for supporting the elaboration of the concepts. The data were collected by Wojciech Ostrowski and processed by Anna Płatek-Żak. All experiments were performed by Paulina Zachar. Zdzisław Kurczyński was responsible for reviewing and editing the paper. The publication was written entirely by Paulina Zachar. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by grants from the National Centre for Research and Development, grant no POIR.01.01.01-00-0980/20.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study have not yet been made available.

Acknowledgments: The data used in this paper were acquired for the project "R&D works in inventory and modelling of key technical and transport infrastructure objects in BIM technology using AI tools in the process of drone data processing", financed by the National Centre for Research and Development, grant no POIR.01.01.01-00-0980/20. The project is being carried out by an interdisciplinary research team by SkySnap Sp. z o.o. Company and Faculty of Geodesy and Cartography of Warsaw University of Technology.

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

References

1. Yang, C.; Zhang, F.; Gao, Y.; Mao, Z.; Li, L.; Huang, X. Moving Car Recognition and Removal for 3D Urban Modelling Using Oblique Images. *Remote Sens.* **2021**, *13*, 3458. [CrossRef]
2. Chu, W.T.; Chao, Y.C.; Chang, Y.S. Street sweeper: Detecting and removing cars in street view images. *Multimed. Tools Appl.* **2015**, *74*, 10965–10988. [CrossRef]
3. Felzenszwalb, P.F.; Girshick, R.B.; McAllester, D.; Ramanan, D. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1627–1645. [CrossRef] [PubMed]
4. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
5. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
6. Everingham, M.; Eslami, S.A.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes challenge: A retrospective. *Int. J. Comput. Vis.* **2015**, *111*, 98–136. [CrossRef]
7. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 740–755.
8. Wu, H.; Liu, Q.; Liu, X. A review on deep learning approaches to image classification and object segmentation. *Comput. Mater. Contin* **2019**, *60*, 575–597. [CrossRef]
9. Laupheimer, D.; Haala, N. Juggling with representations: On the information transfer between imagery, point clouds, and meshes for multi-modal semantics. *ISPRS J. Photogramm. Remote Sens.* **2021**, *176*, 55–68. [CrossRef]
10. Heitz, G.; Koller, D. Learning spatial context: Using stuff to find things. In Proceedings of the European Conference on Computer Vision, Marseille, France, 12–18 October 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 30–43.
11. Razakarivony, S.; Jurie, F. Vehicle detection in aerial imagery: A small target detection benchmark. *J. Vis. Commun. Image Represent.* **2016**, *34*, 187–203. [CrossRef]
12. Zhu, H.; Chen, X.; Dai, W.; Fu, K.; Ye, Q.; Jiao, J. Orientation robust object detection in aerial images using deep convolutional neural network. In Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP), Quebec City, QC, Canada, 27–30 September 2015; pp. 3735–3739.
13. Liu, K.; Mattyus, G. Fast multiclass vehicle detection on aerial images. *IEEE Geosci. Remote Sens. Lett.* **2015**, *12*, 1938–1942.
14. Xiao, Z.; Liu, Q.; Tang, G.; Zhai, X. Elliptic Fourier transformation-based histograms of oriented gradients for rotationally invariant object detection in remote-sensing images. *Int. J. Remote Sens.* **2015**, *36*, 618–644. [CrossRef]
15. Liu, Z.; Yuan, L.; Weng, L.; Yang, Y. A high resolution optical satellite image dataset for ship recognition and some new baselines. In Proceedings of the International Conference on Pattern Recognition Applications and Methods, Porto, Portugal, 24–26 February 2017; Volume 2, pp. 324–331.
16. Benedek, C.; Descombes, X.; Zerubia, J. Building development monitoring in multitemporal remotely sensed image pairs with stochastic birth-death dynamics. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *34*, 33–50. [CrossRef] [PubMed]
17. Xia, G.S.; Bai, X.; Ding, J.; Zhu, Z.; Belongie, S.; Luo, J.; Datcu, M.; Pelillo, M.; Zhang, L. DOTA: A large-scale dataset for object detection in aerial images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 3974–3983.
18. Li, K.; Wan, G.; Cheng, G.; Meng, L.; Han, J. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS J. Photogramm. Remote Sens.* **2020**, *159*, 296–307. [CrossRef]
19. Mao, Z.; Zhang, F.; Huang, X.; Jia, X.; Gong, Y.; Zou, Q. Deep neural networks for road sign detection and embedded modeling using oblique aerial images. *Remote Sens.* **2021**, *13*, 879. [CrossRef]
20. Zachar, P.; Kurczyński, Z.; Ostrowski, W. Application of machine learning for object detection in oblique aerial images. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2022**, *43*, 657–663. [CrossRef]
21. Heo, W.Y.; Kim, S.; Yoon, D.; Jeong, J.; Sung, H. Deep learning based moving object detection for oblique images without future frames. In Proceedings of the Automatic Target Recognition XXX 2020, Online, 27 April–8 May 202; Volume 11394, p. 1139403.

22. Jiang, S.; Jiang, W.; Huang, W.; Yang, L. UAV-based oblique photogrammetry for outdoor data acquisition and offsite visual inspection of transmission line. *Remote Sens.* **2017**, *9*, 278. [CrossRef]
23. Singh, A.K.; Dwivedi, A.K.; Nahar, N.; Singh, D. Railway Track Sleeper Detection in Low Altitude UAV Imagery Using Deep Convolutional Neural Network. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium IGARSS, Brussels, Belgium, 11–16 July 2021; pp. 355–358.
24. Mammeri, A.; Siddiqui, A.J.; Zhao, Y. UAV-assisted Railway Track Segmentation based on Convolutional Neural Networks. In Proceedings of the 2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring), Helsinki, Finland, 25–28 April 2021; pp. 1–7.
25. Singh, A.K.; Swarup, A.; Agarwal, A.; Singh, D. Vision based rail track extraction and monitoring through drone imagery. *Ict Express* **2019**, *5*, 250–255. [CrossRef]
26. Sahebdivani, S.; Arefi, H.; Maboudi, M. Rail track detection and projection-based 3D modeling from UAV point cloud. *Sensors* **2020**, *20*, 5220. [CrossRef]
27. Hu, Q.; Yang, B.; Khalid, S.; Xiao, W.; Trigoni, N.; Markham, A. Towards semantic segmentation of urban-scale 3D point clouds: A dataset, benchmarks and challenges. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4977–4987.
28. Bojarczak, P.; Lesiak, P. UAVs in rail damage image diagnostics supported by deep-learning networks. *Open Eng.* **2021**, *11*, 339–348. [CrossRef]
29. Henry, C.; Poudel, S.; Lee, S.W.; Jeong, H. Automatic detection system of deteriorated PV modules using drone with thermal camera. *Appl. Sci.* **2020**, *10*, 3802. [CrossRef]
30. Ramachandran, A.; Sangaiah, A.K. A review on object detection in unmanned aerial vehicle surveillance. *Int. J. Cogn. Comput. Eng.* **2021**, *2*, 215–228. [CrossRef]
31. Osco, L.P.; Junior, J.M.; Ramos, A.P.M.; de Castro Jorge, L.A.; Fatholahi, S.N.; de Andrade Silva, J.; Pistori, H.; Gonçalves, W.N.; Li, J. A review on deep learning in UAV remote sensing. *Int. J. Appl. Earth Obs. Geoinf.* **2021**, *102*, 102456. [CrossRef]
32. Yang, F.; Fan, H.; Chu, P.; Blasch, E.; Ling, H. Clustered object detection in aerial images. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 8311–8320.
33. Kyrkou, C.; Plastiras, G.; Theocharides, T.; Venieris, S.I.; Bouganis, C.-S. DroNet: Efficient convolutional neural network detector for real-time UAV applications. In Proceedings of the 2018 Design, Automation Test in Europe Conference Exhibition (DATE), Dresden, Germany, 19–23 March 2018; pp. 967–972.
34. Li, C.; Yang, T.; Zhu, S.; Chen, C.; Guan, S. Density Map Guided Object Detection in Aerial Images. *arXiv* **2020**, arXiv:2004.05520.
35. Zhang, H.; Yang, W.; Yu, H.; Zhang, H.; Xia, G.-S. Detecting Power Lines in UAV Images with Convolutional Features and Structured Constraints. *Remote Sens.* **2019**, *11*, 1342. [CrossRef]
36. Dos Santos, A.A.; Marcato Junior, J.; Araújo, M.S.; Di Martini, D.R.; Tetila, E.C.; Siqueira, H.L.; Aoki, C.; Eltner, A.; Matsubara, E.T.; Pistori, H.; et al. Assessment of CNN-based methods for individual tree detection on images captured by RGB cameras attached to UAVs. *Sensors* **2019**, *19*, 3595. [CrossRef] [PubMed]
37. Rasmussen, C.B.; Kirk, K.; Moeslund, T.B. The Challenge of Data Annotation in Deep Learning—A Case Study on Whole Plant Corn Silage. *Sensors* **2022**, *22*, 1596. [CrossRef]
38. Zhang, D.; Han, J.; Cheng, G.; Yang, M.H. Weakly supervised object localization and detection: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**. [CrossRef]
39. Ros, G.; Sellart, L.; Materzynska, J.; Vazquez, D.; Lopez, A.M. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3234–3243.
40. Xie, J.; Kiefel, M.; Sun, M.T.; Geiger, A. Semantic instance annotation of street scenes by 3d to 2d label transfer. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3688–3697.
41. Zhuo, X.; Fraundorfer, F.; Kurz, F.; Reinartz, P. Building detection and segmentation using a CNN with automatically generated training data. In Proceedings of the IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium, Valencia, Spain, 22–27 July 2018; pp. 3461–3464.
42. Kapania, S.; Saini, D.; Goyal, S.; Thakur, N.; Jain, R.; Nagrath, P. Multi object tracking with UAVs using deep SORT and YOLOv3 RetinaNet detection framework. In Proceedings of the 1st ACM Workshop on Autonomous and Intelligent Mobile Systems, Bangalore, India, 11 January 2020; pp. 1–6.
43. Kellenberger, B.; Volpi, M.; Tuia, D. Fast animal detection in UAV images using convolutional neural networks. In Proceedings of the 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Worth, TX, USA, 23–28 July 2017; pp. 866–869.
44. Bazi, Y.; Melgani, F. Convolutional SVM networks for object detection in UAV imagery. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 3107–3118. [CrossRef]
45. Wang, X.; Cheng, P.; Liu, X.; Uzochukwu, B. Fast and accurate, convolutional neural network based approach for object detection from UAV. In Proceedings of the IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society, Washington, DC, USA, 21–23 October 2018; pp. 3171–3175.

46. Zhang, W.; Qi, J.; Wan, P.; Wang, H.; Xie, D.; Wang, X.; Yan, G. An easy-to-use airborne LiDAR data filtering method based on cloth simulation. *Remote Sens.* **2016**, *8*, 501. [CrossRef]
47. Padilla, R.; Passos, W.L.; Dias, T.L.; Netto, S.L.; Da Silva, E.A. A comparative analysis of object detection metrics with a companion open-source toolkit. *Electronics* **2021**, *10*, 279. [CrossRef]