



Article

Object Recognition of a GCP Design in UAS Imagery Using Deep Learning and Image Processing—Proof of Concept Study

Denise Becker  and Jörg Klonowski * 

i3mainz—Institute for Spatial Information and Surveying Technology, Mainz University of Applied Sciences,
Lucy-Hillebrand-Str. 2, D-55128 Mainz, Germany

* Correspondence: joerg.klonowski@hs-mainz.de

Abstract: Image-based unmanned aircraft systems (UASs) are used in a variety of geodetic applications. Precise 3D terrain surface mapping requires ground control points (GCPs) for scaling and (indirect) georeferencing. In image analysis software (e.g., Agisoft Metashape), the images can be generated to a 3D point cloud using Structure-from-Motion (SfM). In general, the conventional GCP design for UAS flights is a checkerboard pattern, which is provided in the software and used for automatic marker detection in each image. When changing the pattern, manual work would be required by picking the GCP individually by hand. To increase the level of automation in the evaluation, this article aims to present a workflow that automatically detects a new edge-based GCP design pattern in the images, calculates their center points, and provides this information to the SfM software. Using the proposed workflow based on deep learning (DL) and image processing, the quality of the resulting 3D model can be equated to the result with GCP center points picked by human evaluator. Consequently, the workload can be accelerated with this approach.

Keywords: unmanned aircraft system; ground control point; deep learning; object detection



Citation: Becker, D.; Klonowski, J. Object Recognition of a GCP Design in UAS Imagery Using Deep Learning and Image Processing—Proof of Concept Study. *Drones* **2023**, *7*, 94. <https://doi.org/10.3390/drones7020094>

Academic Editor: Arianna Pesci

Received: 21 December 2022

Revised: 22 January 2023

Accepted: 27 January 2023

Published: 30 January 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Today, unmanned aircraft systems (UASs) [1,2] are used in a wide range of applications [2,3]. In 2019, a market study by the unmanned aerial vehicle (UAV) association conducted a survey on the industrial use of UASs in Germany. According to the survey, the systems are primarily used in the geodetic context. Applications include surveying (79%), inspection (53%), and mapping and monitoring (33%) [4]. From a technological point of view, the compactness and accuracy of data acquisition systems are rapidly increasing [2,5]. Usually, UAVs are equipped with lightweight and high-resolution cameras. The flight area is captured photogrammetrically and reconstructed to a 3D point cloud using Structure-from-Motion (SfM) in a postprocess [6].

Ground control points (GCPs) with fixed geodetic coordinates are essential for high-accuracy sensing of the environment. The inclusion of these measurements as additional observations results in a more stable, scaled, and (indirectly) georeferenced 3D SfM model [1]. To increase the quality of the resulting 3D point cloud and to achieve more flexibility, e.g., by reducing the number of GCPs [7], real-time kinematic (RTK) dual-frequency GNSS receivers become inevitable and enable direct georeferencing [8,9]. However, it is not recommended to avoid GCPs completely but to stabilize the triangulation (bundle block adjustment) by at least one observation to obtain accurate results in the end [10]. Furthermore, if the navigation system (direct georeferencing) cannot be used due to the localities because the signal is severely limited (building shadowing in inner cities, valley or forest areas, etc.) or cannot be received at all (e.g., indoor flight), the image-based approach must rely on GCPs with local or global coordinates in the orientation phase [1]. With the increasing use of different acquisition technologies, the demand for sensor combinations is also growing. The use of GCPs as identical points in environmental acquisition can be useful for both fusion and stabilization of point clouds.

In geodesy, terrestrial laser scanning (TLS) has been used for decades to capture the environment with a rotating laser beam. Compared to the more recent UAS-based SfM technology, TLS achieves more detailed results but is insufficient in some aspects, such as facades and reconstruction of roofs. For this reason, several studies aim to combine TLS- and UAS-based point clouds [11–13] with appropriate approaches [14]. The idea is to use uniform control points to merge the resulting point clouds, improve the results, and increase the information context. Currently, there is no standardized point design pattern that can be used for both techniques. For this reason, we decided to use a design pattern from the TLS domain, which is being tested for UAS applications. The Institute of Geodesy and Geoinformation at the University of Bonn (IGG) has developed a new TLS target design [15] for the precise registration of single scans and accurate georeferencing. Based on the new IGG TLS target, we developed a new GCP design pattern for the UAS domain. The 2D target is square and contains eight straight lines running from the outside to the inside. The areas between the lines are alternately colored black and white, like a star pattern. The classic checkerboard pattern or specially coded markers are used as reference points and are implemented in the software Agisoft Metashape (1.8.0) so that the recognition during the marker search is automated. Since other appearances or newly developed GCP patterns are not provided from scratch, more manual time is involved in picking the center points. In this article, we intend to increase the automation level of GCP detection in UAS images by using a workflow that incorporates statistical and deterministic methods such as deep learning (DL) and image processing operators. Using DL, the GCPs are localized in the images. After the detection pipeline, digital image processing is used to calculate the GCP center points automatically. Optical text recognition is integrated to assign a unique ID to each recognized GCP. Finally, the calculated results are embedded in the Agisoft Metashape routine.

To locate the GCPs in the images, we used the object detection approach based on DL [16] by recognizing the repeating pattern in the UAS images. Convolutional neural networks (CNNs) or deep neural networks (DNNs) [17] represent state-of-the-art methods to learn complex features without manual intervention in an end-to-end learning based on deep architectures. Since the proposal of regions with CNN features (R-CNN) [18], a series of extended and improved models have been researched. The concept of R-CNNs based on two-stage detectors that first generate the region proposals and second perform refinements for each region. Among them, the Faster R-CNN represents a fast model for real-time object detection, which aims to jointly optimizes the classification and bounding box regression tasks [19]. You only look once (YOLO) is also a state-of-the-art CNN but based on a one-stage detector that performs object detection using fixed-grid regression [20]. At least, YOLO is more recent and faster than Faster R-CNN [21]. In addition, the single-shot detector (SSD) [22] is used for multireference and multiresolution, which improve the detection accuracy for smaller objects [23]. Basically, one-stage detectors are faster and simpler, but in most cases have less accurate performance than the two-stage regional proposal detectors. To overcome this problem, the class imbalance was identified as the main obstacle for one-stage object detectors, and RetinaNet was proposed [24], which is also applied in this study. When RetinaNet is trained with focal loss, it achieves the same speed as existing one-stage detectors while outperforming the accuracy of existing two-stage detectors [24]. The design is characterized by many similarities with previous dense detectors and, in particular the use of feature pyramids as in SSDs [22] and Feature Pyramid Networks (FPNs) [25]. A simple network architecture is built with an FPN backbone (multilevel features) linked to a feedforward ResNet architecture [26] to generate a rich, multiscale convolutional feature pyramid. Two subnets are added to the backbone, one for classifying anchor boxes and one for regressing anchor boxes on ground truth object boxes [24]. Since we did not train a CNN RetinaNet from scratch, we employed transfer learning to use a pretrained network to initialize the feature extractor kernels. In this process, knowledge is transferred from one domain to another. TensorFlow is used as the backend for object detection.

The results are evaluated by an independent UAS flight using the new GCP design pattern. This article does not discuss the specifications of the UAS, flight constellations, acquisition setup, recording configuration, distribution of GCP, camera calibration, or the basic settings of the SfM process in Agisoft Metashape. Instead, it presents a proof of concept for detecting individually designed edge-based GCP patterns in UAS images using DL and image processing operators.

2. Materials and Methods

In this chapter, we present a new GCP design pattern and the study area where the UAS flight will be conducted. Additionally, we explain the workflow from image acquisition to evaluation. The methodologies of object detection and digital image processing are integrated into the workflow. For object detection, the data involved in both the training and evaluation phase are outlined, and the model configuration is described.

2.1. GCP Design and Study Area

In UAS flights, GCPs for indirect georeferencing are usually represented as a black-and-white square checkerboard pattern or a circular disk. Based on the recently developed TLS target from IGG [15], called BOTAS, we printed the new GCP pattern for the UAS domain on 25 cm \times 25 cm plates of low thermal expansion coefficient aluminum composite. The identification of the GCP center is significant in UAS imagery because the model receives global information for postprocessing. The high contrast to the background makes it clearly recognizable. The eight-fold division of the GCP character ensures precise determination of the center point. The results of the presented workflow for automatic GCP detection and evaluation are demonstrated using a small UAS flight. For this purpose, a fixed point field of 20 m \times 25 m is set up on a model airfield. Five GCPs are placed in the four corners and one in the center for appropriate distribution. In the center, the GCP is mounted on a tripod to vary the height, while the other ones are placed on the ground. In addition, each GCP is assigned a unique number (ID) that allows for later identification. In this study, the numbers 2, 3, 5, 6, and 8 are used. Figure 1 shows the study area and the GCP design both schematically and during the UAS flight at a flight altitude of 30 m.



Figure 1. (Left) Study area with five placed GCP numbered 2, 3, 5, 6, and 8. (Top right) Image section of GCP from nadir flight 30 m above the ground with 4.2 mm GSD. (Bottom right) Eight-fold GCP design pattern.

The UAV is a self-developed system on which cameras can be mounted modularly. The take-off weight is limited to 4.9 kg. Automated flight planning is prepared and executed

using the Mission Planner software (1.3.70). The image flight arrangement at 30 m height is extended with a cross flight with 20% height difference (36 m). During the whole flight, the speed of 2 m/s is maintained. The image acquisition is realized with the Sony Alpha 7R camera. The resolution is 7360×4912 pixels, and the size of a pixel is $4.9 \mu\text{m}$. The camera constant is specified as 35.0 mm. The camera settings used on the study area are shown in Table 1.

Table 1. Sony Alpha 7R settings for the study area.

Setting	Value
Exposure time	1/1000 s
ISO value	200
Hyperfocal distance	6 m
Aperture	7.1

At 30 m altitude, a ground sample distance (GSD) of 4.2 mm is achieved. Longitudinal and transverse overlap is selected as 80% and 60% to provide stable bundle block adjustment in the postprocess. A total of eleven images (JPG format) in nadir view are used. The specifications of the UAS are not described in detail. The focus in this study is the detection of the pattern in the captured nongeoreferenced images and is considered independent of the equipment and flight conditions. Only the camera settings and image qualities are relevant for the precise calculations.

2.2. Methodology

We developed a workflow for GCP detection using DL and image processing to increase the automation level in the SfM routine (Figure 2). In the beginning, the UAS images are captured (step 1). After preparing the data (step 2), the DL-based detector predicts the object localization (step 3). The detected bounding boxes are used in image processing to calculate the center point of each GCP (step 4), which are then provided for evaluation in Agisoft Metashape (step 5). In the following, we describe the methodology of data preparation, object detection, and image processing.

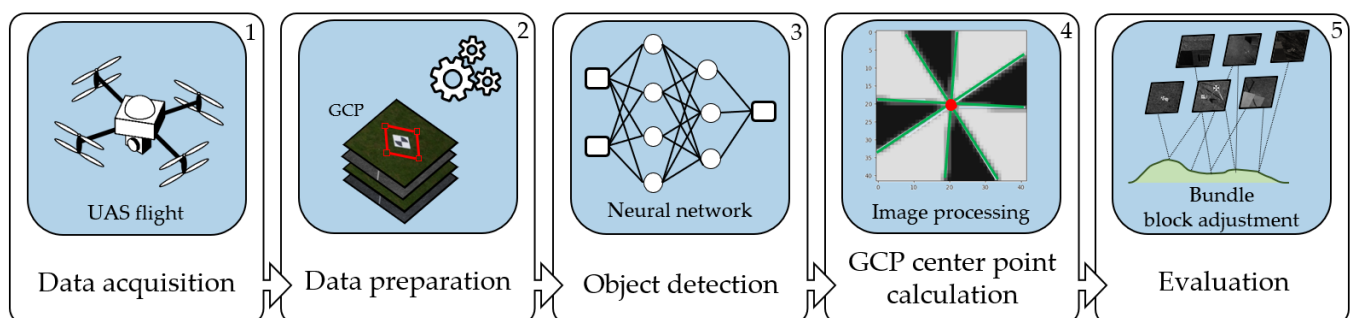


Figure 2. Workflow of automatic GCP detection. The process chain starts from UAS-based image acquisition, data processing, object recognition, and GCP center point calculation in Python 3.8 to evaluation and 3D point cloud generation in Agisoft Metashape 1.8.0.

2.2.1. Dataset Preparation, Augmentation, and Split

DL-driven object detection requires a large amount of data for the training, validation, and test phases. For this reason, we conducted additional flights using the UAS dataset described above. Thereby, we used the same UAS, the same camera, and approximately the same camera settings. The image datasets are characterized by nadir flights, different environments, and various flight altitudes between 20 m and 35 m. To reliably detect the pattern on different backgrounds, the GCPs are placed in the field, in forested, and in urban environments. In addition, shadow and brightly exposed areas are considered different lighting conditions and are thus included. The number of GCPs can also vary in the images,

so the datasets differ in the frequency of occurrence of the patterns. The captured UAS images that do not contain at least one GCP are excluded, essentially reducing the original size of the datasets. Under these conditions, we collected the UAS datasets from different flights. Figure 3 shows an example of the datasets for each flight, where the GCPs are highlighted by red bounding boxes.



Figure 3. Examples of six different UAS datasets with GCP locations highlighted in red. A total of 246 images were captured.

Three main tasks are addressed in the data preparation step. First, the images are manually annotated using the YOLO BBox Annotation Tool (Ybat) [27] and saved in the commonly known PASCAL (pattern analysis, statistical modeling, and computational learning) VOC (visual object classes) format [28]. Each image is linked to an extensible markup language (XML) file that contains the annotations (ground-truth). Second, the image resolution is resized to 25% relative to the original image size, resulting in 1840×1228 pixels. This task has a massive impact on the data volume and is necessary for graphics processing unit (GPU) training for object detection. Third, to increase the amount of training data, we used data augmentation. It is important to maintain the shape of the GCP pattern. The augmentation operations are presented in Table 2. The number of images in the augmented dataset is $3 \times 246 = 984$, which are used for the training and validation phase.

Table 2. Augmentation operations.

Operation	Value	Description
Flip horizontally	-	Flip the image horizontally
Flip vertically	-	Flip the image vertically
Rotation	180°	Rotate the image

Finally, the augmented dataset is split into two subsets: training and validation, with a ratio of 80:20. The training subset is used for optimizing the weights, while the validation subset is used for optimizing the hyperparameter to prevent overfitting. The test subset is utilized to evaluate the model after training independently. At this point, the dataset of the

UAS flight from the study field (a total of eleven images) is used. For the object detection pipeline, we converted the subsets into three disjunct TFRecord files (data format) and created a label map, which maps the class (GCP) to an integer value.

2.2.2. Object Detection Using RetinaNet50

The DL model selected in this work is SSD ResNet50 V1 FPN 1024×1024 (RetinaNet50) with an initial mAP (mean average precision) of almost 40% (Table 3). The SSD-based DL model was collected from the TensorFlow 2 detection model zoo [29] and trained using the open-source framework TensorFlow Object Detection Application Programming Interface (API) [30,31]. We applied transfer learning to accelerate performance improvement due to pretrained weights based on the Microsoft Common Objects in Context (MS COCO) dataset [32].

Table 3. RetinaNet50 from TensorFlow 2 detection model zoo [29].

Model Name	Speed (ms)	COCO mAP	Output
SSD ResNet50 V1 FPN 1024×1024 (RetinaNet50)	87	38.3	Boxes

We implemented object detection with TensorFlow 2.9.2 on Python 3.8. To accelerate the training phase, we used high-performance computing (HPC) resources called “El-wetritsch” at the University of Kaiserslautern-Landau (RPTU) in Germany. Specifically, we allocated an NVIDIA Tesla V100 GPU with 128 GB of memory. Based on the predefined pipeline of the used model [29], the hyperparameter setting was chosen as follows: The batch size was set to two. For the learning rate, 0.04 was used, and the warm-up learning rate was given as 0.0133. The weight of the L2 regularization was 0.0004. The training process stopped when the loss curve no longer converged. In this study, 40,000 steps seemed to be sufficient.

For characterizing the performance of a model, object detection evaluation metrics are used. The commonly used metrics were based on confusion matrix elements such as True Positives (TP), i.e., correctly recognized objects, false positives (FP), i.e., incorrectly recognized objects, and false negatives (FN), i.e., incorrectly unrecognized objects. To indicate what were true and false objects, the intersection of union (IoU, also known as Jaccard Index [33]) was used to measure the overlap between two bounding boxes. In object detection, the area of the labeled bounding boxes (ground-truth) is intersected with the predicted bounding boxes. From these concepts, two metrics called precision (P) and recall (R) can be used for the final model evaluation of object detection tasks [34]. Precision (1) indicates the accuracy of the predictions by determining the percentage of positive identifications that were correct. Recall (2) is a measure of success rate and represents the percentage of correct positive predictions that were correctly detected.

$$P = \frac{TP}{TP + FP} = \frac{TP}{\text{all detections}} \quad (1)$$

$$R = \frac{TP}{TP + FN} = \frac{TP}{\text{all ground truths}} \quad (2)$$

To present the result for the RetinaNet50 model we used multiple metrics from the MS COCO, such as the average precision (AP, AP50, AP75) and the average recall (AR1, AR10), which is determined using multiple IoU values over all images [32]. AP was averaged over multiple IoU values (ten IoU thresholds) from 50% to 95% with a step size of 5% (0.50:0.05:0.95). AP50 and AP75 were calculated for single IoU at 50% and 75%. The AR1 and AR10 scores were the average recalls across all images with one and ten detections per image over all classes and IoU thresholds.

2.2.3. Image Processing

Image processing means the extraction of information from image data using conventional operators. After the GCPs have been localized and highlighted by the detected bounding boxes as a paired coordinate tuple $[y_min, x_min, y_max, x_max]$ (step 3), the center points can be deterministically determined using common image operations. The aim is to detect significant edges between the black and white areas and to describe these edges parametrically. Based on the edges, the center of the GCP is calculated via line intersection. For the SfM approach, the GCP must be uniquely identified. For this reason, we placed numbers close to the GCPs. Figure 4 shows the image processing steps based on a bounding box section. In this process, only the images of the test dataset (study area) were used in their original size. The calculation of the center point is conducted individually for each bounding box.

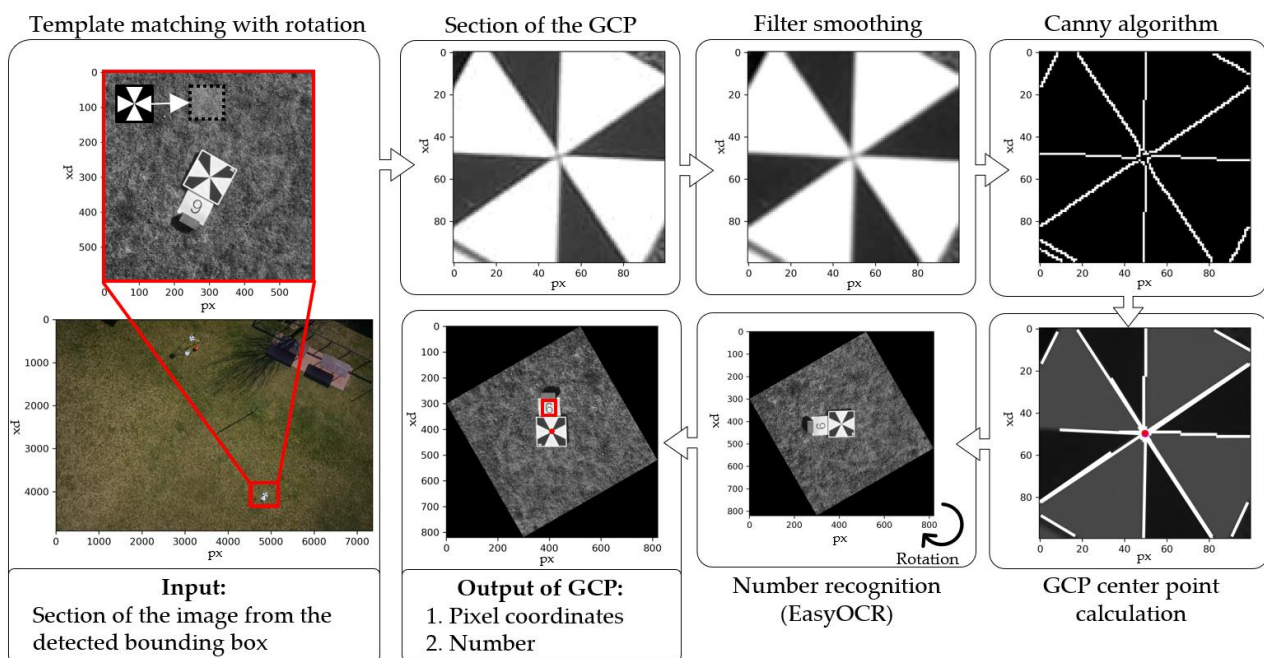


Figure 4. Steps of image processing for center point calculation and number recognition.

The region of a bounding box is enlarged by 150 pixels to ensure the recognition of the GCP numbers. Using an iterative template matching approach, the GCP design is found within the image section limited by the enlarged bounding box. For the match with a given pattern template (synthetically created image), a correlation of 80% is predefined. In the first step, the template is shifted natively at 0° rotation angle over the image section. If no correlation reaches the threshold, the template is rotated 10° , and the process is repeated for a total of eight times. For a better interpretation of the image information, the GCP sections are cropped and converted based on gray values. A linear smoothing filter (Gaussian filter) is applied to reduce the noise. The Canny edge detection algorithm [35] is used to detect significant gray value gradients. For the recognition of objects which consist predominantly of straight lines, the Hough transform [36] is particularly suitable [37]. The lines are constructed using the edge pixels based on the Hough line transform. The operations of the smoothing filter, Canny algorithm, and Hough transform are implemented using OpenCV. The GCP center point is calculated using straight line intersections.

To identify the GCP ID in the images, optical character recognition is used. For this purpose, EasyOCR (easy optical character recognition) is integrated into the workflow. If the OCR method fails, the GCP section is rotated with the corresponding template orientation, and the operation is run again. In case the number is still missing, the image is rotated in 90° steps up to 360° , and OCR is applied each time.

For the Agisoft Metashape routine, the GCP center points (pixel coordinates) and ID information are structurally stored in a text format and then automatically read in via the Agisoft Metashape Python API.

3. Results and Discussion

To validate the workflow and the final result in Agisoft Metashape (Section 3.3), we present the intermediate results of object detection (Section 3.1) and digital image processing (Section 3.2). In object detection, the performance of the RetinaNet 50 model is evaluated using the test dataset. The detected bounding boxes are provided as an image section for further image processing. Template matching is applied to locate the GCPs in the image section, and image operators are employed to determine the center points. The evaluation of the UAS flight of the study area is performed in Agisoft Metashape.

3.1. Object Detection

The hyperparameter optimization of the RetinaNet50 model is based on the validation dataset. For each step carried out in the training, the performance is determined by using the metrics below. Training and validation losses converged uniformly while 40,000 weight adaptations were performed. Table 4 shows the result of the validation and test dataset at the last checkpoint.

Table 4. Performance of the trained RetinaNet50 model for validation and test dataset.

COCO (%)	AP	AP50	AP75	AR1	AR10
Validation dataset	76.2	98.0	89.3	53.2	83.6
Test dataset	40.1	78.1	42.7	26.0	61.5

The result of the test dataset (AP = 40.1% and AR10 = 61.5%) shows worse results compared to the validation dataset (AP = 76.2% and AR10 = 83.6%). One reason is the number that were placed out during the UAS flights of the training and validation dataset. The similarities between the GCP design and the ID plate (black number on a white background) are partly too great for correct separation. If the numbers are very close to the GCPs, the algorithm has fewer complications. In the case of the elevated GCPs placed on the tripod, the positions of the GCP and the ID plate differ, resulting in two detected objects. To increase the model performance, the aspect of the number plates must be considered in future studies. Another reason is that part of a bright table and bench is found as a GCP in an image. This falsely detected GCP (FP) also affects the performance. The results of object recognition are shown in four examples in Figure 5. In total, all GCPs were found with adequate probabilities. Compared to the other points, GCP number five has the disadvantage that the nearby tree obstructs the view in some images. In general, image-based evaluations require that sufficient visibility exists. Due to the absence of leaves in the tree, it is possible to identify the pattern in its basic structure. For this reason, the datasets with tree objects should be enlarged so that the GCP pattern is trained even with visual obstructions. The GCPs located in the shaded areas (numbers three and five) show significantly lower scores (69% and 44%). The percentage of shaded areas can also be expanded in the training and validation dataset so that GCP is found more reliably in dark areas. In addition, it is recommended to add different contrast settings to the augmentation operations so that the dark areas are better represented. Finally, the performance of the RetinaNet50 model trained on non-UAS datasets such as MS COCO [32] is satisfactory for GCP detection on the existing datasets with the help of data augmentation. If the training success of the GCP is increased with extended datasets from different environments and further augmentation operations, the scope of detected objects can be raised so that the nonrelevant objects drop below the predefined threshold.

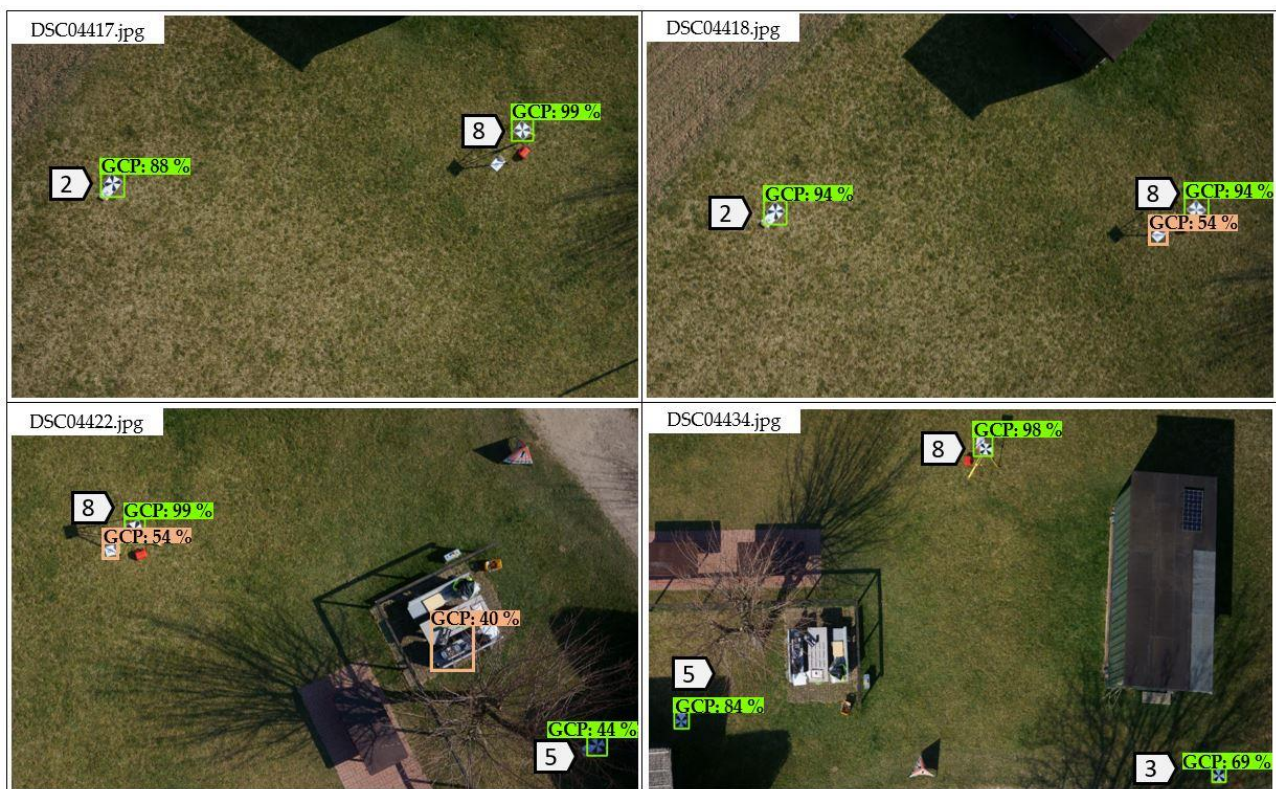


Figure 5. Results of the test dataset of four images with correctly detected GCP (green, *TP*) and incorrectly detected GCP (orange, *FP*).

In addition, other object detection models can be tried to find out which model performs best for GCP detection. The results obtained with the RetinaNet50 model are sufficient for this study.

For the further step in image processing, the bounding boxes of the *FP* detections are not critical, because number plates or other objects, such as a table or a bench, are ignored during the template matching process. When multiple predicted bounding boxes of a GCP are found within an image, nonmaximal suppression (NMS) is applied to select the main entity from many overlapping entities. Bounding boxes with a score above 40% are carried over to the next step.

3.2. Calculation of the GCP Center Point and Recognition of the ID

In this step, the GCP center points are calculated. The previously detected bounding boxes on the test dataset and the template of the GCP design pattern are used as data basis. The duration of the Python script runs approximately 5:55 min without using GPU on a standard notebook (Intel i7 4 × 2.50 GHz with 16 GB of memory). The EasyOCR method takes the most time to complete the process. Optionally, GCP center points can also be indicated without finding the numbers and reduce the processing time to 1:54 min. On average, the process takes about 45 s for one image with two GCPs including ID search.

To evaluate the results, we compare the calculated center points with manually determined integer pixel values (picked GCP). Figure 6 shows for comparison the manually picked pixel coordinate in the original image and the calculated point from the image processing.

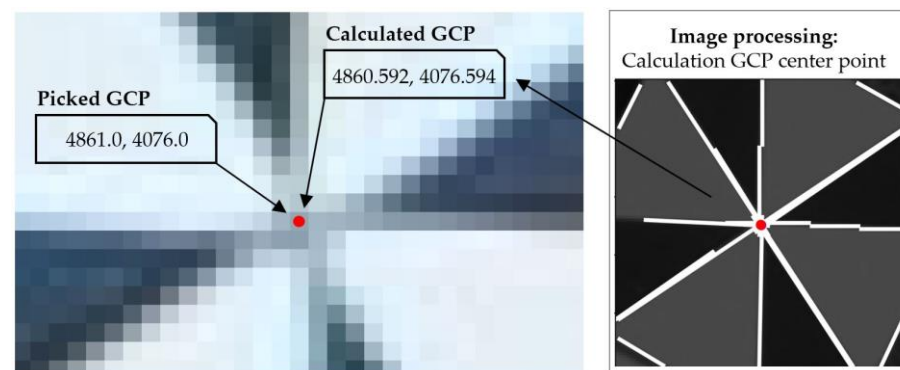


Figure 6. Example of the GCP center point (x- and y-axis in pixels) of number six in image DSC04416.jpg. **(Left)** Manually picked GCP with 2300% zoom. **(Right)** Calculated GCP from straight line intersections of image processing.

The numerical comparison of the coordinates and the result of the number recognition are presented in Table 5. Using template matching, all center points of the 20 GCPs can be calculated. The mean differences between the picked and calculated GCPs are -0.06 px in the x- and 0.18 px in the y-axis of the image. Based on the manual picking of integer pixel values, an uncertainty due to rounding errors of at least 0.5 px was expected. Most of the differences are less than 0.5 pixel. At points three and eight, some values are above 1.0 px. The maximum value of 1.06 px is at point eight in image DSC04422.jpg. Due to the vegetation state of the tree, the basic structure of the pattern can be recognized, as already mentioned in the previous section, so that the center point calculation is possible. For oblique images, it is necessary to check the applicability of the template matching. If the given section deviates geometrically from the reference, the method can no longer be performed reliably with high correlation [37].

Table 5. Results of the pixel differences in the x- and y-axis between manually picked and calculated GCP center points and success of number identification (noN = no detected number).

Image Name	Detected Number	Target Number	$\Delta x, \Delta y$ (px)
DSC04416.jpg	6 noN	6 8	$-0.41, 0.59$ $-0.15, 0.16$
DSC04417.jpg	noN noN	2 8	$0.00, -0.57$ $0.09, 0.2$
DSC04418.jpg	2 noN	2 8	$-0.17, 0.23$ $-0.32, 0.04$
DSC04419.jpg	3	3	$0.27, 0.39$
DSC04420.jpg	2 3	2 3	$-0.16, -0.46$ $-0.82, -0.16$
DSC04421.jpg	noN noN	2 8	$0.24, -0.23$ $-0.3, 0.32$
DSC04422.jpg	noN noN	5 8	$0.02, 0.30$ $-0.26, 1.06$
DSC04423.jpg	6	5	$-0.2, 0.41$
DSC04426.jpg	noN noN	3 8	$0.75, -0.14$ $-0.49, 0.96$
DSC04427.jpg	3	3	$-0.35, 0.46$
DSC04434.jpg	3 5 noN	3 5 8	$0.29, 0.21$ $0.29, 0.15$ $0.49, -0.35$
Total	9	20	

The numbers belonging to the corresponding GCP represent an unsatisfactory result. Nine numbers can be recognized; the others are classified with no detected number (noN). In addition to the numbers, probabilities can be specified. On average, the values for the correctly found numbers are between 80–100%. Point eight has a bad condition for number recognition due to the tripod shadow. The view of the number at point five is complicated by the tree, and not every image due to the change of perspective is favored. In image DSC04423.jpg, the number five is mistakenly recognized as six (probability 50%). Another problem for the weak results of the number recognition could be the heavy objects on the edge of the number plates, which were used for fixing. The objects may be too

close to the numbers so that the unambiguous number recognition is disturbed. In the future, the number plates will be fixed in the ground with long needles so that this factor is eliminated. To increase the hit rate, the image rotation can be divided into several smaller rotation angles so that the algorithm recognizes the number in the approximately correct orientation. However, this leads to a significantly longer processing time. For the test dataset, the runtime of about five minutes seems manageable. However, this would scale up exorbitantly for a UAS flight of several 100 images with GCPs. ID recognition should be improved. Another approach could be the integration of another DL pipeline at this point [38,39]. Alternatively, the MNIST dataset [40], which contains handwritten numbers, can be used to train a number-based model that is used for ID recognition of the GCPs [41].

The name of the images, the detected GCPs with calculated pixel coordinates, and the GCP numbers are stored and transferred to Agisoft Metashape via the Python plugin (“Run Python Script”).

3.3. Evaluation in Agisoft Metashape

For the evaluation of the final result at the end of the workflow, two projects were created in Agisoft Metashape, where the images of the study area were added. In the first project, every single GCP is hand-picked as a marker. In the second project, the calculated center points are provided as markers within a few seconds with the plugin. Since some GCPs are not numbered (noN), the markers with the name “no GCP number” must be manually replaced with the existing numbers and deleted (Figure 7). The duration of the process is about two minutes.

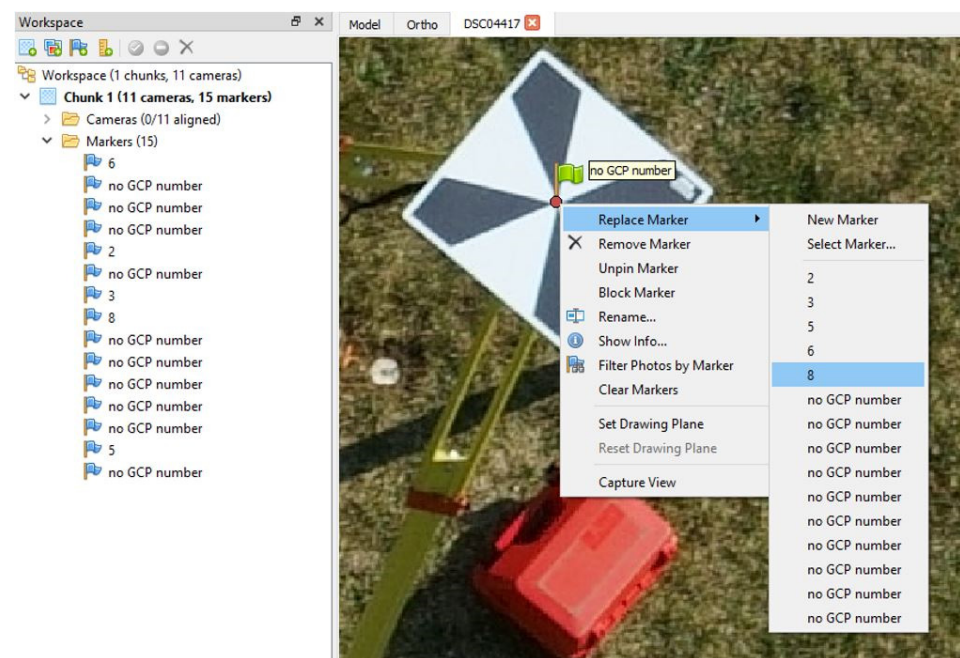


Figure 7. Result of the calculated GCPs in Agisoft Metashape. Missing numbers of the GCPs are changed to the correct numbers with “Replace Marker”.

In the project based on manually picked markers, the process takes about five minutes. After the images are aligned with the setting “high”, global coordinates are added to the GCPs with an accuracy of 1.0 cm. This value is also used as a weighting in the bundle block adjustment, which adjusts the preresult of the alignment. The results in both projects are shown in Table 6.

Table 6. SfM-Results of the image-based (px) and absolute error (cm) in Agisoft Metashape for the two projects (manually picked GCP and calculated GCP).

GCP	Projections	Picked GCP (px)	Picked GCP (cm)	Calculated GCP (px)	Calculated GCP (cm)
2	4	0.608	1.44	0.592	1.47
3	5	1.333	0.72	1.359	0.70
5	3	0.451	1.23	0.460	1.25
6	1	0.146	1.65	0.143	1.61
8	7	0.899	2.41	0.754	2.49
Total error		0.912	1.59	0.874	1.61

In the first project, the center points are selected manually in the images (2D), and in the second project, the calculated GCP are loaded. For this reason, the error in pixels is most important. We listed the absolute error (cm) to show the effect from the image plane to the 3D global point. The image-based and the absolute error after triangulation are similarly accurate for both methods and do not differ significantly. The comparison between the image-based error before and after importing the global coordinates varies marginally. The GCPs of number three are mostly at the edges of the images, which means that the image quality suffers, and the point cannot be determined as accurately as the others. Therefore, the image-based errors have larger values at these points. When the outliers of the GCPs of number three are excluded in both projects, the accuracy of the SfM result on the image plane increases to about 0.6 px. GCP number six can generally be neglected because the point has no redundancy. In summary, the same SfM results are achieved with the loaded calculated GCP center points as with the manually picked GCPs determined by a human evaluator.

Another comparison can be carried out using the point clouds of both projects. For this purpose, the dense point cloud with accuracy level “high” was generated in Agisoft Metashape. Using GCP eight as an example, the global coordinates were hand-picked from the point cloud depending on the point density and distribution and compared to the reference coordinate (Figure 8). The differences are similar for both approaches and are significantly better in the location than in the height component.

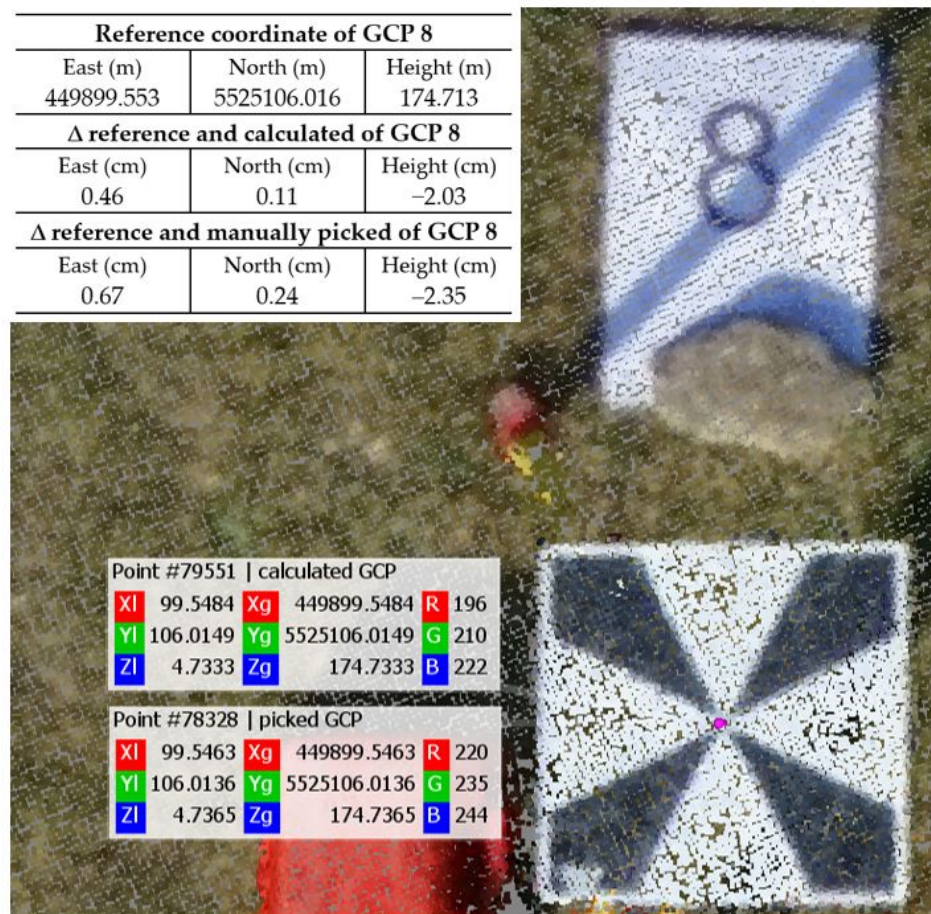


Figure 8. Results of the coordinates of GCP 8 from the generated point clouds from both projects (visualized in CloudCompare (2.12.2)) and difference determination to the reference coordinate (top left in the table).

4. Conclusions and Future Work

In this paper, we presented a proof-of-concept study focusing on image-based detection of a new GCP design pattern in UAS image data using DL specifically object detection with the RetinaNet50 model and image processing operators. Based on the result of a UAS flight in Agisoft Metashape, we can show that the degree of automation in the evaluation phase of marker identification is increased and the SfM results between the projects with calculated and manually picked GCP center points have equal accuracies. When evaluating additional flights, the workflow is shortened by skipping data preparation and object detection. The trained network can be applied to subsequent image data to locate the bounding boxes around the GCPs and calculate their center points.

In the future, further studies could be carried out. The accuracy comparison between the new GCP pattern and, e.g., the conventional checkerboard pattern for UAS flights, should be conducted to determine quality differences. In addition, the application of this pattern could be adapted to other technologies, such as investigating the use of calibration processes, and the possibility of combining different technologies could be tested to increase the potential and synergies. It would be interesting to evaluate the application of point cloud fusion between TLS, including scanning technology on a flying platform (ALS, airborne laser scanning) and image-based UAS, which we did not investigate in this article. To apply the workflow to the full range of UAS applications, studies based on UAS flights with oblique imagery and various altitudes are necessary.

Author Contributions: Conceptualization, D.B. and J.K.; methodology, D.B.; software, D.B.; validation, D.B.; formal analysis, D.B.; investigation, D.B.; resources, D.B.; data curation, D.B.; writing—

original draft preparation, D.B.; writing—review and editing, D.B. and J.K.; visualization, D.B.; supervision, J.K.; project administration, J.K.; funding acquisition, J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Carl-Zeiss-Foundation and is part of the project “BAM—Big-Data-Analytics in environmental and structural monitoring” at Mainz University of Applied Sciences (P2017-02-003).

Data Availability Statement: Not applicable.

Acknowledgments: We thank Bastian Plaß for technical support during the introducing of HPC and for debugging the DL network used in this study. Further thanks go to the University of Kaiserslautern-Landau (RPTU) in Germany for providing the high-performance cluster “Elwetritsch”. We thank Sven Kaulfersch for his technical support during the UAS flights. The authors would like to thank the model flight club Nierstein/Oppenheim (registered association in Germany) for providing the study area. We also thank Svenja Schwerdtfeger, Linda Staiger, David Kernstock, and Jan Emrich for their assistance with data collection and analysis. We also thank the anonymous reviewers who gave valuable feedback.

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

Abbreviations

The following abbreviations are used in this manuscript:

ALS	Airborne laser scanning
AP	Average precision
API	Application programming interface
AR	Average recall
CNN	Convolutional neural network
DL	Deep learning
DNN	Deep neural network
FN	False negative
FP	False positive
FPN	Feature pyramid network
GCP	Ground control point
GPU	Graphics processing unit
GSD	Ground sample distance
HPC	High-performance computing
IoU	Intersection of union
NMS	Nonmaximal suppression
noN	No detected number
mAP	Mean average precision
MS COCO	Microsoft Common Objects in Context
OCR	Optical character recognition
P	Precision
R	Recall
R-CNN	Region-based convolutional neural network
RTK	Real-time kinematic
SfM	Structure-from-Motion
SSD	Single-shot detector
TLS	Terrestrial laser scanning
TP	True positive
UAS	Unmanned aircraft system
UAV	Unmanned aerial vehicle
XML	Extensible markup language
YOLO	You only look once

References

- Nex, F.; Remondino, F. UAV for 3D mapping applications: A review. *Appl. Geomat.* **2014**, *6*, 1–15. [\[CrossRef\]](#)
- Colomina, I.; Molina, P. Unmanned aerial systems for photogrammetry and remote sensing: A review. *ISPRS J. Photogramm. Remote Sens.* **2014**, *92*, 79–97. [\[CrossRef\]](#)
- Hassanalian, M.; Abdelkefi, A. Classifications, applications, and design challenges of drones: A review. *Prog. Aerosp. Sci.* **2017**, *91*, 99–131. [\[CrossRef\]](#)
- Analyse des Deutschen Drohnenmarktes. Available online: https://www.bdli.de/sites/default/files/global_upload_upload/Analyse%20des%20deutschen%20Drohnenmarktes.pdf (accessed on 5 December 2022).
- Neitzel, F.; Klonowski, J. Mobile 3D mapping with a low-cost UAV system. In Proceedings of the ISPRS Zurich 2011 Workshop, Zurich, Switzerland, 14–16 September 2011; Volume XXXVIII-1/C22, pp. 39–44.
- Küng, O.; Strecha, C.; Fua, P.; Gurdan, D.; Achtelek, M.; Doth, K.-M.; Stumpf, J. Simplified building models extraction from ultra-light uav imagery. In Proceedings of the ISPRS Zurich 2011 Workshop, Zurich, Switzerland, 14–16 September 2011; Volume XXXVIII-1/C22, pp. 217–222.
- Przybilla, H.-J.; Bäumker, M. Untersuchungen zur Qualität des Realtime Kinematic GNSS Systems der DJI Phantom 4 RTK. In Proceedings of the 40th Wissenschaftlich-Technische Jahrestagung der DGPF, Stuttgart, Germany, 4–6 March 2020; pp. 47–61.
- Mian, O.; Lutes, J.; Lipa, G.; Hutton, J.; Gavelle, E.; Borghini, S. Direct georeferencing on small unmanned aerial platforms for improved reliability and accuracy of mapping without the need for ground control points. In Proceedings of the International Conference on Unmanned Aerial Vehicles in Geomatics, Toronto, Canada, 30 August–2 September 2015; Volume XL-1/W4, pp. 397–402.
- Gerke, M.; Przybilla, H.J. Accuracy analysis of photogrammetric UAV image blocks: Influence of onboard RTK-GNSS and cross flight patterns. *Photogramm. Fernerkund. Geoinf.* **2016**, *1*, 17–30. [\[CrossRef\]](#)
- Przybilla, H.J.; Bäumker, M.; Luhmann, T.; Hastedt, H.; Eilers, M. Interaction between direct georeferencing, control point configuration and camera self-calibration for RTK-Based UAV photogrammetry. *ISPRS Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2020**, *43*, 485–492. [\[CrossRef\]](#)
- Aicardi, I.; Dabove, P.; Lingua, A.M.; Piras, M. Integration between TLS and UAV photogrammetry techniques for forestry applications. *Iforest Biogeosci. For.* **2016**, *10*, 41–47. [\[CrossRef\]](#)
- Son, S.W.; Kim, D.W.; Sung, W.G.; Yu, J.J. Integrating UAV and TLS Approaches for Environmental Management: A Case Study of a Waste Stockpile Area. *Remote Sens.* **2020**, *12*, 1615. [\[CrossRef\]](#)
- Abdelazeem, M.; Elamin, A.; Afifi, A.; El-Rabbany, A. Multi-sensor point cloud data fusion for precise 3D mapping. *Egypt. J. Remote Sens. Space Sci.* **2021**, *24*, 835–844. [\[CrossRef\]](#)
- Zang, Y.; Yang, B.; Li, J.; Guan, H. An Accurate TLS and UAV Image Point Clouds Registration Method for Deformation Detection of Chaotic Hillside Areas. *Remote Sens.* **2019**, *11*, 647. [\[CrossRef\]](#)
- Janßen, J.; Medic, T.; Kuhlmann, H.; Holst, C. Decreasing the Uncertainty of the Target Center Estimation at Terrestrial Laser Scanning by Choosing the Best Algorithm and by Improving the Target Design. *Remote Sens.* **2019**, *11*, 845. [\[CrossRef\]](#)
- Zhao, Z.-Q.; Zheng, P.; Xu, S.-T.; Wu, X. Object Detection with Deep Learning: A Review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3212–3232. [\[CrossRef\]](#) [\[PubMed\]](#)
- Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [\[CrossRef\]](#)
- Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
- Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
- Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
- Li, Y.; Dong, H.; Li, H.; Zhang, X.; Zhang, B.; Xiao, Z. Multi-block SSD based on small object detection for UAV railway scene surveillance. *Chin. J. Aeronaut.* **2020**, *33*, 1747–1755. [\[CrossRef\]](#)
- Liu, W.; Anguelov, D.; Erhan, D.; Christian, S.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In *Computer Vision—ECCV 2016*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 21–37.
- Walambe, R.; Marathe, A.; Kotecha, K. Multiscale Object Detection from Drone Imagery Using Ensemble Transfer Learning. *Drones* **2021**, *5*, 66. [\[CrossRef\]](#)
- Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2999–3007.
- Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
- He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
- Ybat—YOLO BBox Annotation Tool. Available online: <https://github.com/drainingsun/ybat> (accessed on 9 December 2022).

28. Everingham, M.; Gool, L.V.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [[CrossRef](#)]
29. TensorFlow 2 Detection Model Zoo. Available online: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md (accessed on 28 November 2022).
30. TensorFlow Object Detection API. Available online: https://github.com/tensorflow/models/tree/master/research/object_detection (accessed on 28 November 2022).
31. Fleury, D.; Fleury, A. Implementation of Regional-CNN and SSD Machine Learning Object Detection Architectures for the Real Time Analysis of Blood Borne Pathogens in Dark Field Microscopy. *Preprints* **2018**, 2018070119. [[CrossRef](#)]
32. Lin, T.Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C.L.; Dollár, P. Microsoft COCO: Common Objects in Context. In *Computer Vision—ECCV 2014*; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 740–755.
33. Jaccard, P. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bull. De La Soc. Vaud. Des. Sci. Nat.* **1901**, *37*, 547–579.
34. Padilla, R.; Netto, S.L.; Da Silva, E.A.B. A Survey on Performance Metrics for Object-Detection Algorithms. In Proceedings of the International Conference on Systems, Signals and Image Processing (IWSSIP), Niteroi, Brazil, 1–3 July 2020; pp. 237–242.
35. Canny, J. A Computational Approach to Edge Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *6*, 679–698. [[CrossRef](#)]
36. Hough, P.V.C. Method and Means for Recognizing Complex Patterns. U.S. Patent US3069654A, 18 December 1962.
37. Luhmann, T.; Robson, S.; Kyle, S.; Boehm, J. *Close-Range Photogrammetry and 3D Imaging*; De Gruyter: Berlin, Germany, 2019.
38. Laroca, R.; Barroso, V.; Diniz, M.; Gonçalves, G.; Schwartz, W.; Menotti, D. Convolutional Neural Networks for Automatic Meter Reading. *J. Electron. Imaging* **2019**, *28*, 13023. [[CrossRef](#)]
39. Sarika, N.; Sirisala, N.; Velpuru, M.S. CNN based Optical Character Recognition and Applications. In Proceedings of the 6th International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 20–22 January 2021; pp. 666–672.
40. Le Cun, Y.; Cortes, C.; Burges, C.J.C. The MNIST Database of Handwritten Digits. 2012. Available online: <http://yann.lecun.com/exdb/mnist/> (accessed on 8 December 2022).
41. Baldominos, A.; Saez, Y.; Isasi, P. A Survey of Handwritten Character Recognition with MNIST and EMNIST. *Appl. Sci.* **2019**, *9*, 3169. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.