VEHICLE POSE AND SHAPE ESTIMATION IN UAV IMAGERY USING A CNN

S. El Amrani Abouelassad*, M. Mehltretter, F. Rottensteiner

Institute of Photogrammetry and GeoInformation, Leibniz University Hannover, Germany
(elamrani, mehltretter, rottensteiner)@ipi.uni-hannover.de

KEY WORDS: Object detection, object reconstruction, pose estimation, shape estimation, autonomous driving.

ABSTRACT:

Vehicle reconstruction from single aerial images is an important but challenging task. In this work, we introduce a new framework based on convolutional neural networks (CNN) that performs monocular detection, pose, shape and type estimation for vehicles in UAV imagery, taking advantage of a strong 3D object model. In the final training phase, all components of the model are trained end-to-end. We present a UAV-based dataset for the evaluation of our model and additionally evaluate it on an augmented version of the Hessingheim benchmark dataset. Our method presents encouraging pose and shape estimation results: Based on images of 3 cm GSD, it achieves median errors of up to 5 cm in position and 3° in orientation, and RMS errors of ± 7 cm and ± 24 cm in planimetry and height, respectively, for keypoints describing the car shape.

1. INTRODUCTION

Detecting and retrieving the pose and shape of object instances in a scene based on aerial imagery is still a challenging task in photogrammetry and computer vision. Information about the pose and the shape of objects is crucial in many applications requiring vision-based scene understanding, such as autonomous driving and traffic monitoring. In this paper, traffic surveillance at critical points is the focus application, where UAVs are assumed to be part of the infrastructure that monitors those scenes. Such UAVs can exchange information with other vehicles or infrastructure for autonomous driving, so that traffic agents can use it to know about road conditions, traffic congestion, accidents, etc. These applications usually use monocular camera setups because of their ease of use compared to more sophisticated set-ups such as stereo or multi-view systems.

However, object reconstruction based on a single image is very challenging due to the loss of information caused by the dimensionality reduction that takes place when projecting a three dimensional (3D) object to a 2D image plane. Traditional methods for 3D reconstruction of objects rely on the use of explicit prior shape models for specific object types, but they often require assumptions that might not be valid for real applications. Recently, the use of deep learning for object reconstruction has become an active research topic (Li et al., 2021; Alidoost et al., 2019). In deep learning based approaches, neural networks are trained to learn the relationship between a 2D image and a 3D object. Although they have achieved remarkable results thanks to their powerful learning capability, deep learning methods still face many problems, which include handling the shape complexity of objects, the selection of the optimum shape representation, and the completion of parts of objects that are not visible in the image (Ahmed et al., 2018; Chabra et al., 2020; Reddy et al., 2019). We address these challenges while reconstructing vehicles based on UAV imagery using a deep learning method together with a strong 3D object model learned from data.

In this paper, we introduce a methodology for the estimation of the pose and shape parameters of vehicles using a single UAV image with known orientation parameters. The method is based on a strong object model similar to (Zia et al., 2013). The following major contributions are made: (1) We introduce a new method to jointly estimate the shape, pose and type of a vehicle in 3D from a single UAV image. It is based on (El Amrani Abouelassad and Rottensteiner, 2022), which extends the Faster R-CNN model for object detection (Ren et al., 2017) to predict rotated bounding boxes, but goes beyond that work by additional branches predicting the vehicle type and the shape parameters of the 3D model. The new model is trained end-to-end, extending previous work by two variants of a regression loss for shape parameter estimation. (2) We introduce a new dataset for 3D vehicle pose, shape, and type estimation from UAV imagery including reference information for all these tasks[1]. (3) We evaluate the new method using the new dataset and an augmented variant of the Hessingheim benchmark (Kölle et al., 2021) and show that the method achieves promising results for shape and pose estimation.

2. RELATED WORK

In this section, we discuss related work on the estimation of the pose and shape of objects for 3D object reconstruction.

2.1 Object Pose Estimation

There has been an increasing use of deep learning for the estimation of object poses in 2D and 3D using a single image. Requiring detection, localisation, and prediction of the correct orientation of multiple objects in an image, pose estimation is a harder task than localisation. Methods suggested to solve this problem differ by the object representation that is used. Some works use keypoint based approaches, whereas others use bounding boxes for the task. Bi et al. (2019) introduce a 6D pose estimation framework with a segmentation stream and a 2D keypoint locations stream. The segmentation stream predicts the label of the object observed at each pixel while the regression stream predicts the 2D keypoint locations for that object. The pose candidates derived from the keypoints are combined into a set of 3D-to-2D correspondences, from which the

* Corresponding author

[1] The dataset will be published at `https://data.uni-hannover.de/organization/i-c-sens`; up to then it can be obtained from the main author on request.

pose is estimated. Reddy et al. (2019) also estimate the pose of vehicles in street view images by determining 2D/3D locations of visible and occluded keypoints. They use a region of interest feature map from a detector as input for a multi-layer convolutional block to generate heatmaps with a confidence score for visible keypoints. These confidence scores are passed through a graph encoder-decoder network trained to localise occluded 2D keypoints. The output of this network is passed through a 3D encoder to predict the shape basis and the camera orientation for vehicles. García López et al. (2019) adapt a method for human pose estimation (Moreno-Noguer, 2017) to determine vehicle poses. A vehicle with N keypoints is represented by 2D and 3D poses using $N \times N$ distance matrices. The method uses a two-stage network architecture (Newell et al., 2016) to first predict semantic 2D keypoints from vehicles and then convert them into 3D world coordinates by 2D-to-3D distance matrix regression. While these methods show good results, they are computationally expensive due to the generation of multiple outputs (keypoints) for a large number of object proposals.

The prediction of 3D bounding boxes has been pursued thanks to the success of deep learning networks in detecting 2D bounding boxes for target objects. Using monocular street view images, Ku et al. (2019) use object detectors to generate 2D bounding boxes for the classes $Vehicles$ and $Pedestrians$ in image space. These 2D boxes are used to generate image crops that are passed to an encoder for feature map extraction. The feature map is used to determine instance-centric 3D proposals for 3D bounding boxes, regress offsets of the proposals to get more precise bounding boxes, and predict instance point clouds using a CNN that was trained using LiDAR points. Such points may not be readily available for training, and the quality of the representation by 3D points predicted from a single image may be doubted. Tekin et al. (2018) use a single-shot deep CNN to regress a 6D object pose in one stage using 3D bounding boxes, while Sundermeyer et al. (2018) detect bounding boxes for object crop generation first and then predict object rotations using a representation learned from rendered 3D model views. Although these methods have shown good results, they represent the target objects as 3D bounding boxes, which leads to an estimation of the object size, but only to a very coarse representation of the shape. In applications such as tracking and re-identification, representing vehicles by fine-grained shapes can be very helpful. In this work, we are interested in a better representation of the vehicle shape. Therefore, we will estimate oriented bounding boxes for vehicles along with their shapes.

2.2 Object Shape Estimation

Recovering the shape of an object from a single image is a challenging task, as it requires both, powerful object detection methods and prior shape knowledge. Object shape priors are crucial information for 3D reconstruction from images, particularly if the object is observed only partially, as such priors act as robust regularisation terms and support the shape reconstruction also for occluded parts of the object. In general, shape representations can be categorised into implicit ones, e.g. signed distance and occupancy functions, and explicit ones, e.g. voxels, point clouds, and meshes (Xiao et al., 2020).

Using an implicit representation, the shape of a 3D object is described by an implicit function, for example, a signed distance function (SDF) or an occupancy function (OF). For any 3D point, a SDF delivers its minimal distance from the object surface, whereas an OF delivers the probability of this point being located inside the object. In the past, implicit representations were commonly approximated by sampling from the implicit function in a regular manner and storing the results in a 3D voxel grid, whereas values for points between voxel centres were obtained via interpolation. Following this approach, Engelmann et al. (2016) propose to use a truncated SDF for estimating the pose and shape of detected vehicles from stereo images by energy minimization. In more recent work, implicit functions have commonly been approximated using deep neural networks. Still relying on the concepts of a SDF (Park et al., 2019; Xu et al., 2019) or an OF (Mescheder et al., 2019; Chibane et al., 2020; Peng et al., 2020), such deep learning based methods estimate the implicit representation of an object directly. In addition, such methods have the ability to reconstruct the shape of objects from a single image (Xu et al., 2019) and to represent various topologies (Chen et al., 2022) as well as large-scale scenes (Peng et al., 2020) at arbitrary resolutions. However, the representation complexity and the level of detail of the models depend on the representative power of the employed neural network and thus on its size, which often characterises these approaches as computationally heavy. Also, implicit representations only carry information about the surface of an object, but lack semantic keypoint or edge locations, which increases the difficulty of model fitting. Lastly, the complexity of such approaches increases the risk for overfitting and for the occurrence of artifacts when reconstructing objects that differ from those seen during training. Consequently, such methods often deliver good results on synthetic data, but perform poorly on sparse and noisy real-world data, which is particularly true for the reconstruction of shapes from a single image. Duggal et al. (2022) tried to overcome these limitations by learning to embed more robust shape priors into a neural network via a discriminator function. While this approach improves the performance under real-world conditions, it requires a complex training strategy together with an optimisation at test time.

Active shape models (ASMs) (Cootes and Taylor, 1992) have the advantage of being a less complex method to represent the geometry of an object class that has great variability in shape, size, and appearance. They consist of flexible sets of labelled points representing an object, from which neighbouring point statistics are evaluated using several training shapes to learn a low-dimensional parameterisation (Cootes and Taylor, 1992; Chen and Medioni, 1991). ASMs have been used in object pose and shape estimation applications by extracting object features from images or point clouds and using these observations to predict the parameters of the ASM, e.g., Zhou et al. (2010); Busch (2019). Combining deep learning methods and ASMs to exploit both of their advantages is interesting, yet current work commonly uses ASMs either to get a coarse estimate of the object region (Nguyen et al., 2022) or as a refinement in a separate step applied to get a coarse object shape and position (Shi et al., 2021). Shi et al. (2021) use an ASM to determine the 3D shape of the object on the basis of 3D keypoint detections, while a CNN is applied to estimate the 2D object locations in the input images. While the method shows encouraging results when using images along with depth as input, when using monocular images only, its performance is very sensitive to the quality of the detected keypoints. To the best of our knowledge, no CNN based method has been proposed yet to directly estimate the shape parameters of an object (specifically vehicles) represented as ASM using a single image. We believe that the use of deep learning to estimate vehicle shapes following a model based approach to keep the number of parameters describing the shape small is an interesting path to investigate.

3. METHODOLOGY

3.1 Overview

The goal of the method presented in this paper is to extract vehicles from aerial near-nadir view images and at the same time determine each vehicle's pose and 3D shape. This goal is achieved by extending the Faster R-CNN method (Ren et al., 2017) so that it predicts rotated bounding boxes rather than boxes aligned with the image coordinate system in a way similar to (El Amrani Abouelassad and Rottensteiner, 2022). Whereas the latter work is based on Mask R-CNN (He et al., 2017) and thus includes a branch for instance segmentation, no such branch is considered in this paper. However, the network is extended so that it additionally predicts the vehicle type and the parameters required to describe the 3D shape of the vehicle. The resultant network architecture is shown in Figure 1.

Similarly to (El Amrani Abouelassad and Rottensteiner, 2022), the network uses a ResNet50 backbone to extract features from the image. This is followed by a rotated region proposal network (RRPN), proposing a limited number of rotated regions of interest (RoIs) that are to be considered candidate regions for containing vehicles. Feature maps of a fixed size are extracted within these rotated RoIs, and they are presented to four network branches. Two of them were already considered in (El Amrani Abouelassad and Rottensteiner, 2022), namely a classification branch, predicting whether a RoI contains a vehicle or not, and a bounding box regression branch that predicts improved parameters of the bounding boxes enclosing the detected vehicle instances. The other two branches, which also take the feature maps extracted in the RoIs, correspond to the major modification to the architecture proposed in this paper. The first of these new branches, referred to as the type prediction branch, predicts the vehicle type, where we differentiate a set of $n_T = 6$ vehicle types $T \in \{$`Estate Car, Compact Car, Sports Car, Sedan, SUV, Van`$\}$. The second new branch, referred to as the shape prediction branch, regresses the parameters describing the 3D shape of the car. In this context, and in order to be able to reconstruct the 3D shape from a single image, we use a strong object model; we represent cars by an ASM, learning the parametrisation of vehicle models from CAD models of cars (Zia et al., 2013) so that only a very small set of shape parameters needs to be predicted by the network. Our method predicts the pose of the vehicle in image space. However, if a Digital Terrain Model (DTM) is available, it can be used to deliver the height component if the pose of the vehicle is to be determined in 3D object space. In the following subsections, we describe the main parts of our methodology and the training procedure in more detail.

3.2 Vehicle Representation

In this work, every instance of a vehicle is represented by a rotated bounding box $\mathbf{b}$ indicating its pose and extents in image space, its type according to the class structure given earlier, and a vector of parameters $\mathbf{s}$ describing its shape. The bounding box of a vehicle is represented by 5 parameters, namely the image coordinates of the vehicle centre (r, c), the angle $\theta \in (0, 2\pi]$ representing the orientation of the main vehicle axis relative to the x axis of the image coordinate system, and the lengths l_1 and l_2 of the longer and the shorter semi-axes of the oriented rectangular box, respectively. Thus, $\mathbf{b} = (r, c, l_1, l_2, \theta)$.

Learning the ASM requires vehicle models consisting of triangulated irregular networks (TIN), so that the structure of the TIN is identical for each 3D model, i.e. the models essentially have the same number and structure of keypoints as well as the same definition of meshes based on these keypoints. These keypoints being given in a car-specific coordinate system with its origin in the car centre, a vertical z-axis and an x-axis pointing into the driving direction, one can collect all keypoint coordinates in one vector and determine the mean vehicle model $\mathbf{m}$ as the mean of all of these vectors. A principal component analysis (PCA) of the matrix of second central moments of these vectors delivers Eigenvalues σ_s^2 and corresponding Eigenvectors $\mathbf{e}_s$, which form the basis of the ASM. Using the mean model, the Eigenvalues and Eigenvectors, the shape of a car can be represented by a set of parameters $\gamma^{(s)}$, computing the vector $M(\gamma)$ of keypoint coordinates according to

$$M(\gamma) = \mathbf{m} + \sum_{s=1}^{n_s} \gamma^{(s)} \sigma_s \mathbf{e}_s. \quad (1)$$

It is sufficient to restrict the sum in Eq. 1 to the Eigenvectors corresponding to the first n_s largest Eigenvalues; we use $n_s = 3$ in our experiments, i.e. the 3D shape of a car is represented by a vector $\mathbf{s} = (\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$. The model $M(\gamma)$ is given in the local coordinate system defined above; in order to get the keypoint coordinates in object space, the bounding box has to be projected to the object space and the model has to be shifted and rotated so that it is situated in the bounding box and aligned with the driving direction of the car.

3.3 Region Extraction

The purpose of the RRPN is to obtain region proposals that are likely to contain an object. This part of the network uses the architecture proposed in (El Amrani Abouelassad and Rottensteiner, 2022). First, a 3×3 convolutional layer is used to process the feature map produced by the ResNet50 backbone, resulting in a 256-dimensional feature vector for every pixel. Each pixel is considered to be a potential centre of a region that is represented by an anchor, i.e. a rectangular window with a given rotation angle, size, and aspect ratio. The RRPN contains a classification branch that predicts a score for every pixel and every anchor on the basis of the feature vectors just determined, indicating whether a window corresponding to the anchor and centered at that pixel is likely to correspond to an object or not. A regression branch determines improved values for the parameters of the rotated bounding box. After non-maxima suppression, all remaining windows are ordered by the confidence scores of the classification branch of the RRPN, and the N_{prop} (set to $N_{prop} = 1000$ in our experiments) windows having the highest confidence values are selected to be the region proposals used for further processing. To do so, feature maps of a fixed size $n_x \times n_y$ (we use $n_x = 10$ and $n_y = 20$ in our experiments, which is consistent with the default aspect ratio of the anchors) aligned with the rotated region proposal windows are extracted from the output of the ResNet50 backbone, which is done by the rotated RoI pooling (RRoI) layer (cf. Figure 1).

As described earlier, the anchors used by RRPN have different sizes and orientations. As we are only interested in vehicles, the used anchors only have one aspect ratio ($l_1 : l_2 = 2 : 1$) and two scales, but 12 rotation angles in order to be able to predict orientations. Differently from (El Amrani Abouelassad and Rottensteiner, 2022), we first apply bi-linear interpolation to generate a grid at the resolution of the original feature map that is aligned with the main direction of the rotated bounding

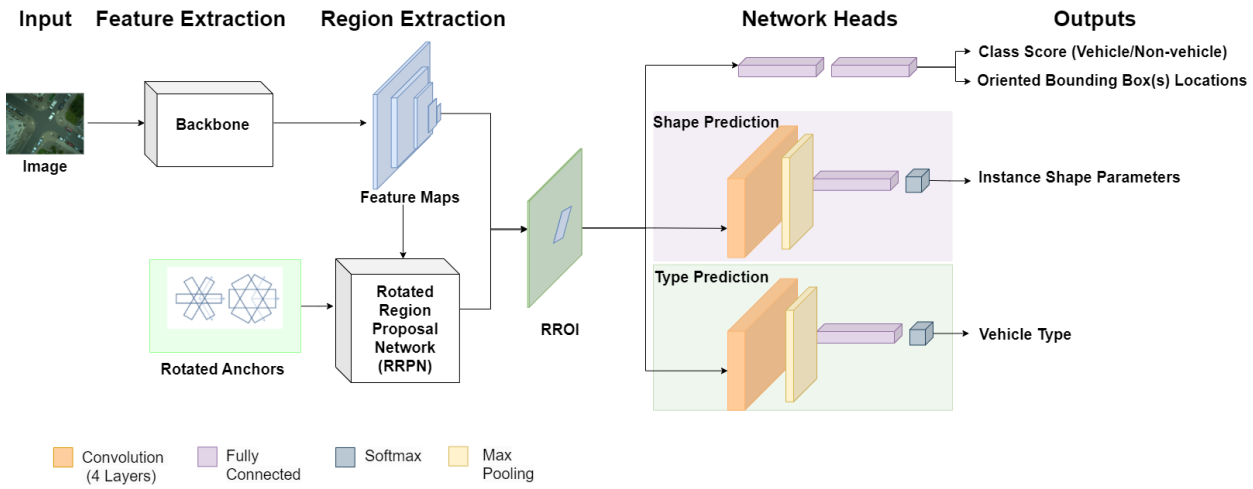

Figure 1. The architecture of the proposed method.

box; after that, maximum pooling is applied, considering all interpolated feature vectors inside one mesh of the output feature map (Ding et al., 2019).

3.4 Classification and Bounding Box Regression

The feature map generated by the RRoI pooling layer is processed by a classification head and the rotated bounding box regression head. The architectures of these heads are largely identical to those proposed in (Ren et al., 2017), consistng of a sequence of fully connected layers. Whereas the classification head predicts the class label of the object inside the region proposal (*Vehicle* vs. *Non-vehicle*), the regression branch predicts five real-valued numbers for improving the bounding box parameters (r, c, l_1, l_2, θ) (El Amrani Abouelassad and Rottensteiner, 2022). The only difference compared to (Ren et al., 2017) is the number of output nodes in the regression branch, because the rotation angle also has to be predicted.

3.5 Prediction of the Vehicle Type and Regression of the Shape Parameters

These new branches take a feature map generated by RRoI pooling, having a size of $n_x \times n_y$, as input. To predict the vehicle type, the feature map is processed by four convolutional layers with ReLU activations, followed by max pooling operations. The size of the filter matrices is 3×3 in these layers, the number of filters is 256, and max pooling is based on windows of size 2×2 and stride 2. Afterwards, two fully connected layers are applied, the first one delivering another 256-dimensional output and the other one producing class scores for the vehicle types (cf. Section 3.1) using the softmax function.

The shape prediction branch also processes the feature map generated by the RRoI pooling layer. Its architecture is identical to the one of the type prediction branch, with the exception of the output layer. Here, the number of nodes in the output layer is identical to the number of parameters n_s used to represent the vehicle shape (cf. Section 3.2), and each of them predicts a real-valued number encoding one of the shape parameters $\gamma^{(s)}$.

3.6 Training

The training data consist of images with known orientation parameters and rotated bounding boxes enclosing vehicles, vehicle type, and vehicle shape parameters. The rotated bounding boxes should be given in both, the image and the object coordinate system; however, if a DTM is available, it is easy to determine one from the other. In principle, we can follow a stratified training method that is also used in (Ren et al., 2017), but using a modified loss function. Using this strategy, one component is trained after the other before performing a joint training step at the end. In the experiments reported in this paper, we initialise the parameters of the network components that were also used in (El Amrani Abouelassad and Rottensteiner, 2022) by pre-trained values that were used in that publication, having been determined using the strategy just described. The parameters of the new type and the shape prediction branches are initialised by random numbers, which is followed by a series of training iterations for determining the parameters of these branches on the basis of the results of the RRPN, freezing the parameters of the other components of the network and minimising the sum of two loss functions $L_t + L_{sh}$ (one for the output of each branch; see below). Having thus obtained good initial values for all parameters, a final end-to-end training step is applied in which all parameters are fine-tuned using a combined loss function considering all intermediate and final outputs. The overall loss L_{total} to be minimised in training consist of three terms:

$$L_{total} = L_b + L_t + L_{sh}, \tag{2}$$

where the loss L_b already has three components:

$$L_b = L_{RRPN} + L_{cls} + L_{reg}. \tag{3}$$

The loss terms in Eqs. 2 and 3 will be explained in the subsequent subsections.

3.6.1 Loss L_b: This loss consist of three terms, namely the RRPN loss L_{RRPN}, the classification loss L_{cls}, and the rotated bounding box head regression loss L_{reg}; they were already defined in (El Amrani Abouelassad and Rottensteiner, 2022). The classification loss L_{cls} is a standard softmax cross entropy loss, whereas for the loss L_{reg}, a regression loss based on the Huber loss function, applied to the difference of the predicted and the given parameters, is used (Ren et al., 2017). The RRPN loss is the sum of a classification and a bounding box regression loss of the two branches inside the RRPN that are modelled in a way similar to L_{cls} and L_{reg}, respectively. For a detailed description of the loss terms constituting L_b, the reader is referred to (El Amrani Abouelassad and Rottensteiner, 2022).

3.6.2 Vehicle Type Loss L_t: The vehicle type loss is computed from the softmax output of the type prediction head. We use a standard cross-entropy loss function for this task:

$$L_{cls} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{n_T} c_j^i \cdot log(p_j^i), \tag{4}$$

where i is the index of a reference vehicle, N is the number of reference vehicles, j is the index of a vehicle type, and n_T is the number of vehicle types ($n_T = 6$; cf. Section 3.2). c_j^i is an indicator showing whether the i^{th} sample corresponds to vehicle type C_j (c_j^i=1) or not (c_j^i=0). Finally, p_j^i is the softmax output for vehicle i to correspond to the vehicle type C_j. This loss will push the parameters of the network towards producing softmax outputs close to 1 for the correct vehicle type.

3.6.3 Vehicle Shape Loss L_{sh}: This loss is computed on the basis of the real-valued output of the shape estimation branch. We propose two variants for that loss function, leading to two variants (V_{L1} and V_{L2}) of the method presented in this paper.

The first variant, V_{L1}, uses an L_2 loss for the regression task:

$$L_{sh}^1 = \frac{1}{N}\sum_{i=1}^{N}\sum_{s=1}^{n_s}(\hat{\gamma}_i^{(s)} - \gamma_i^{(s)})^2, \tag{5}$$

where i and N are as defined in Eq. 4 and s and n_s as in Eq. 1. The parameters predicted by the network are denoted by $\hat{\gamma}_i^{(s)}$, whereas $\gamma_i^{(s)}$ represents the corresponding reference value.

The second variant, V_{L2}, uses a loss that compares the distance between the shape $M(\hat{\gamma}_i^{(s)})$ obtained from the estimated shape parameters $\hat{\gamma}_i^{(s)}$ and the shape $M(\gamma_i^{(s)})$ derived from the reference parameters $\gamma_i^{(s)}$:

$$L_{sh}^2 = \frac{1}{N \cdot N_K}\sum_{i=1}^{N}\sum_{k=1}^{N_K} ||M(\hat{\gamma}_i^{(s)})_k - M(\gamma_i^{(s)})_k||_2, \tag{6}$$

where i and N are as defined in Eq. 4, k is the index of a keypoint in the ASM, N_K is the number of such keypoints, and $||\cdot||_2$ denotes the L_2 norm of a vector. $M(\hat{\gamma}_i^{(s)})_k$ and $M(\gamma_i^{(s)})_k$ are the three-vectors corresponding to the k^{th} keypoint of the predicted and the reference shapes, respectively.

The loss L_{sh}^1 directly compares the parameters determined by the shape prediction branch to the reference, but these differences are abstract quantities having unclear units. The loss L_{sh}^2 is easier to interpret (it is the average metric distance of keypoints of the predicted model and the reference). The results achieved by the network variants based on the two loss functions will be compared in the experiments.

4. DATASETS

The datasets required to train our framework consist of UAV images and a reference consisting of a rotated bounding box, a vehicle type, and shape parameters for vehicles depicted in these images. As the interpretation of the vehicle shape parameters depends on the ASM, the latter has to be generated even before the determination of the reference for training (and testing) in the way described in Section 3.2. For that purpose, a set of 36 different CAD models representative for the differentiated vehicle types was collected from Google's 3D Warhouse[2]. A total of 144 3D keypoints were manually labelled in these CAD models (Coenen and Rottensteiner, 2021), and the corresponding 3D coordinates were the basis for the ASM. In our experiments, two datasets are used for training and evaluating the proposed method. The first one was acquired by us in a dedicated measurement campaign involving image acquisition using a UAV; it is referred to as the UAV dataset (Section 4.1). The second one is an augmented version of the Hessigheim dataset (Kölle et al., 2021) (Section 4.2).

4.1 UAV Dataset

The UAV dataset is the product of a dedicated measurement campaign. In that campaign UAV equipped with a camera hovered over a street intersection and acquired images having a ground sampling distance (GSD) of about 3 cm at a frequency of 10 Hz. This corresponds to a scenario relevant for autonomous driving in which the UAV, considered to be infrastructure, supports the localisation of the cars in its field of view at critical points. The dataset consists of 1400 images of 2592×2048 pixels each, showing both parked and moving cars in the intersecting roads, and the task to be solved is the precise localisation and shape reconstruction of all visible cars. In addition to the images, their orientation parameters and a DTM of the street surface are known, though these data are not necessarily required for the evaluation. More importantly, the required reference data were generated by manual annotation. The dataset contains the required data for 2156 vehicle instances.

To generate a reference for the rotated bounding boxes, the outlines of all visible cars were digitised in all UAV images, and the 2D rotated bounding boxes were derived by calculating a minimum bounding rectangle for each of these outlines. A human operator inspected the resultant orientations and corrected them manually so that they always point into the driving direction. Furthermore, each car instance was annotated with a vehicle type according to the class structure defined in Section 3.1. Using the available DTM and the orientation parameters, the bounding boxes could be transformed into object space.

To obtain a reference for the shape parameters of vehicles, 24 keypoints of the ASM were used. These keypoints were identified manually in several images in which they were visible. For that purpose we could also use stereo image pairs acquired from moving cars at street level with a base line of 1 m for which orientation parameters were known. Thus, for static cars, the 3D coordinates of keypoints in object space could be determined by forward intersection from multiple images (one stereo pair and one UAV image), and these points could be transformed to a local vehicle coordinate system using the 3D bounding boxes. For moving cars, we used the corners of the 3D bounding boxes as tie points to relate the local car coordinate systems to each other; then, the poses of the UAV images from which the bounding boxes were determined were transformed into the local car coordinate system. Even though there was only a small baseline between different UAV images, the fact that the car had moved between different acquisition meant that there was a somewhat larger baseline between the projection centres in this local car coordinate system, which also allowed the determination of the 3D coordinates in the local car coordinate system by forward intersection. Finally, the parameters $\gamma^{(s)}$ of the reference car models were determined by fitting the ASM according to Eq. 1

[2] `https://3dwarehouse.sketchup.com`

to the available subset of 3D keypoints. The resultant reference consists of 2156 vehicle instances for which the shape and pose parameters are given.

4.2 Hessingheim Dataset

The Hessingheim dataset (Kölle et al., 2021) provides four epochs of UAV-based LiDAR measurements and stereo images, together with a reference for several tasks, though not for 3D car reconstruction. In this paper, we use an unpublished dataset, referred to as H3D, from a measurement campaign for a fifth epoch in the same area provided to us by the benchmark organizers on request. The H3D dataset was acquired using a UAV equipped with a LiDAR system and cameras. We were provided with images of 14204 x 10652 pixels each and a GSD of 2-3 cm. We also received the laser points, the orientaion parameters of the images, and a DTM.

Here, the generation of the reference was based on the LiDAR points. First, car instances were identified in the point cloud, using the images as auxiliary information. Afterwards, a human operator annotated the instance with a car type and selected one of the 36 CAD models used to learn the ASM (the one most similar to the car instance); that CAD model was manually shifted and rotated until it fitted well to the point cloud, using the software $CloudCompare$[3]. The pose of the CAD model along with its extent could be used to define the 2D bounding box and the orientation of the car in the object coordinate system, using the height of the lowest points of the CAD model (wheels) as the Z component. The parameters $\gamma^{(s)}$ of the reference car models were determined by fitting the ASM according to Eq. 1 to the keypoints that had been annotated in the CAD models (see above). In this case, the 2D poses in image space are determined by backprojecting the 3D vehicle model to the image and computing the minimum bounding rectangle, considering the correct driving direction for the orientation. In total, the reference consists of 220 distinct vehicle instances in this dataset.

5. EXPERIMENTS AND RESULTS

5.1 Test Setup

For the evaluation, we use the two datasets described in Section 4. In both cases, the available images are split into tiles of 1024×1024 pixels each. The images of the UAV dataset show an intersection of two roads. We first divide the image tiles and their respective reference data into two subsets. The first one contains the first road with the intersection area; this subset is used for training and validation according to a 80%:20% split. The second subset contains the second road without the intersection area, and it corresponds to the test set. After augmentation, we get 6536, 2176, and 2178 vehicle instances in the UAV dataset for training, validation and test, respectively. The H3D dataset is also split into disjunct training, validation and test sets with shares of 60%, 20%, and 20%, respectively. After augmentation, we get 1260, 420, and 420 vehicle instances for training, validation and testing, respectively.

For both datasets, we train two variants of our method (V_{L1} and V_{L2}), differing by the loss function used for learning the prediction of the shape parameters (cf. Section 3.6.1). In both cases, we use 24 anchors with one aspect ratio (2:1), two scales (64 and 128 for l_1, respectively), and 12 rotation angles (0°, 30°, 60°, 90°, 120°, 150°, 180°, 210°, 240°, 270°, 300°, 330°). The training of our method is based on stochastic gradient descent with a learning rate of 0.1, momentum of 0.9 and weight decay of 0.0001 for optimisation. We also apply data augmentation by applying random crops, scales, and rotations. We follow the training strategy described in Section 3.6, initialising the parameters of the backbone, the RRPN, and the classification and rotated bounding box regression heads using the pre-trained weights from (El Amrani Abouelassad and Rottensteiner, 2022), initialising and improving the new branches, and finally fine-tuning the entire network in an end-to-end manner by optimising the joint loss in Eq. 2. The final end-to-end training requires 80 epochs, where for each epoch, one image tile is used during each iteration.

To evaluate the quality of the detection results, we compare the predicted oriented bounding boxes to the reference in image space. If the IoU score of a predicted bounding box with a reference bounding box is above 50%, the predicted bounding box is considered to be a true positive (TP); otherwise, it is a false positive (FP). A reference bounding box is considered to be a false negative (FN) if there is no predicted bounding box such that the IoU of the boxes is larger than 50%. Based on the numbers of TP, FP and FN instances, we calculate the precision (percentage of detected boxes that correspond to a reference box), the recall (percentage of reference boxes that were detected), and the F1 score (the harmonic mean of precision and recall).

To evaluate the pose, we determine the Euclidean distance d_t^i of the predicted vehicle position from the reference for every detected vehicle i. Similarly, the differences d_θ^i of the predicted vehicle orientations from the reference values are calculated. We report the median ϵ_{mae}^t of the absolute position errors and the median of the absolute orientation error ϵ_{mae}^θ, as well as the median absolute deviations of the position and orientation errors, σ_{MAD}^t and σ_{MAD}^θ (Hampel et al., 2005):

$$\epsilon_{mae}^m = median(|d_m^i|), \tag{7}$$

$$\sigma_{MAD}^m = 1.4826 \cdot median(|d_m^i - \epsilon_{mae}^m|), \tag{8}$$

where $m \in \{t, \theta\}$. We also present the cumulative histograms of the absolute errors $|d_\theta^i|$ of orientations for the UAV dataset.

For the evaluation of the vehicle type branch, we compare the predicted vehicle types with the reference and analyse the overall accuracy (OA-1), i.e. the percentage of correct decisions. As some of the classes related to vehicle type are not well defined (because they have a very similar appearance in the data), we additionally report another metric (OA-2), considering a prediction to be correct if the reference class is among the two classes with the highest class scores.

In order to evaluate the shape prediction, we use the predicted vehicle shape parameters to determine keypoint positions and compare them to the keypoint positions determined using the ASM parameters from the reference. For every keypoint j and every vehicle i, we compute the 2D distance D_{ij}^{2D} of the predicted and the reference positions as well as the absolute value of the height difference, ΔH_{ij}. Using these values, we compute the root mean square (RMS) errors $E_{D^{2D}}$ and $E_{\Delta H}$ of the distances and the height differences, respectively:

$$E_M = \sqrt{\frac{1}{Kp \cdot N_v} \sum_{i=1}^{N_v} \sum_{j=1}^{Kp} M_{ij}^2}, \tag{9}$$

[3] https://www.danielgm.net/cc/

where N_v is the number of vehicles, Kp is the number of keypoints, and $M \in \{D^{2D}, \Delta H\}$. Additionally, we compare the dimensions of the reconstructed vehicle shapes with the reference. For this purpose, we calculate the minimum 3D bounding boxes enclosing the estimated and the reference models, respectively. We report the RMS errors of the length, width, and height (l, w, h) differences of these 3D bounding boxes:

$$E_d = \sqrt{\frac{1}{N_v}\sum_{i=1}^{N_v} \Delta d_i^2}, \tag{10}$$

where $d \in \{l, w, h\}$, Δd_i is the difference of dimension d for vehicle i, and N_v is the number of vehicles.

5.2 Results and Discussion

The results of the evaluation of the two variants of our method are described separately for the aspects of detection, pose, type and shape estimation in the subsequent sections.

5.2.1 Detection: Table 1 presents the quality metrics achieved for the task of vehicle detection for both datasets and both variants of our method. The results show that variant V_{L1} performs better than V_{L2} in detecting vehicles in both datasets in all metrics. The table also shows that both models perform better on the dataset UAV than on H3D. A reason for that behaviour could be that the UAV dataset contains many more training samples than the H3D training dataset. In general, the detection performance is encouraging, but there is still room for improvement. For instance, the recall of V_{L1} for the UAV dataset indicates that about every fourth car in the test set is not detected. The main reason for missing vehicles is truncation of vehicles at the tile boundaries.

Data	Model	Precision [%]	Recall [%]	F_1 [%]
UAV	V_{L1}	81.6	76.5	78.9
	V_{L2}	79.2	73.6	76.3
H3D	V_{L1}	75.1	68.9	71.8
	V_{L2}	72.3	66.4	69.2

Table 1. Precision, Recall, and F_1 score for vehicle detection on the two test datasets.

5.2.2 Vehicle Pose: Table 2 shows the quality metrics for both the position and orientation estimates as defined in Section 5.1. The error metrics for position were converted to [m] by using the average GSD of the corresponding dataset. Again, the results for variant V_{L1} are better than those achieved using V_{L2}, and both variants perform better for the UAV dataset than for H3D. In general, these results are quite encouraging. The median error ϵ^t_{mae} for position achieved by V_{L1} on the UAV dataset is only 5 cm (2-3 pixels), which would be considered to be sufficient for applications such as autonomous driving. In H3D, it is slightly worse, but still below 10 cm. Something similar can be said about median error ϵ^θ_{mae} for orientation; an error of 2.8° in orientation corresponds to a mis-alignment of 12 cm at a distance of 2.5 m (half the length of a car). It would seem that the pose estimates are quite good for the best variant.

Figure 2 shows the cumulative histogram of the absolute differences between the predicted and the reference angles for V_{L1} and V_{L2} on the UAV test data. The figure show that both variants can predict the majority of the angles correctly, but yet again that V_{L1} is to be preferred over V_{L2}. V_{L1} predicts the orientations with an error smaller than 10° in about 85% of the cases, whereas using V_{L2} the corresponding number is about 75%. However, the figure also shows that there is a considerable percentage of outliers: whereas errors between 20° and 160° are very rare, in about 5% of the cases the orientation error is close to 180°, indicating that the orientation of the car was predicted to correspond to the opposite of the driving direction. This may be due to the difficulty of both variants to differentiate between the front and the rear of some vehicles. It has to be noted that in a near-nadir view, some vehicle types, e.g., Sedans, do indeed have an almost symmetrical appearance, which makes the prediction of the correct orientation difficult.

Data	Model	ϵ^t_{mae} [m]	σ^t_{MAD} [m]	ϵ^θ_{mae} [°]	σ^θ_{MAD} [°]
UAV	V_{L1}	0.05	0.07	2.8	2.7
	V_{L2}	0.10	0.09	3.1	2.9
H3D	V_{L1}	0.09	0.08	3.6	3.5
	V_{L2}	0.11	0.10	4.9	4.3

Table 2. Quantitative position and orientation error metrics for the two test datasets.

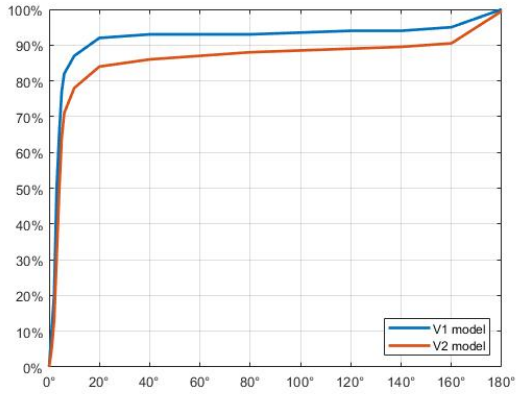

Figure 2. Cumulative histogram of absolute differences between estimated and reference angles for V_{L1} and V_{L2} for the UAV dataset. The abscissa gives the absolute value of the angle difference in [°].

5.2.3 Vehicle Type: Table 3 shows the class-specific F1 scores achieved by vehicle type prediction in variants V_{L1} and V_{L2}, and Table 4 shows the overall accuracies. Again, V_{L1} outperforms V_{L2} in both datasets, and again most of the quality metrics are better for the UAV dataset than for H3D. The F1 scores in Table 3 show that there is a considerable variation between the car types. One of the reasons for this is class imbalance in the training data: the most frequent classes in the training set are Estate Car, Compact Car, Sedan and Van. For the first three of these classes, the F1 scores are larger than 64%, whereas for the others they are below 45%. Analysing the overall accuracy OA-1 in Table 4, one can see that the best variant (V_{L1}) only predicts the correct car type in about 66% of the cases in the UAV dataset and about 53% of the cases in H3D. This relatively low performance of the type classification can be explained by the difficulty of the task: some car types look so similar that it is even difficult for a human annotator to differentiate them in the images. Thus, there are many confusions between certain pairs of car types that look very similar. This can also be inferred from the error rates OA-2 in Table 4, which correspond the percentages of cases in which the true la-

bel is within the top-2 class scores. For instance for the best performing variant, in about 13% of the cases, the correct class achieves the second highest class score for the UAV dataset; for H3D, where in general there is a much smaller number of training samples for all classes, this number is even larger (22%). We take this as an indication that the model has difficulties in differentiating the most similar car types.

Data	Model	EC	CC	SC	Sed.	SUV	Van
UAV	V_{L1}	72.2	64.1	41.5	70.2	31.4	36.9
	V_{L2}	71.5	63.9	39.8	68.8	30.3	33.4
H3D	V_{L1}	69.7	67.7	32.6	65.3	36.7	30.7
	V_{L2}	68.7	64.8	32.4	64.4	36.4	29.8

Table 3. F1 scores [%] for the categories of the vehicle type branch on both datasets by both models. EC, CC, SC, Sed.: Estate Car, Compact Car, Sports Car, Sedan.

Data	Model	OA-1 [%]	OA-2 [%]
UAV	V_{L1}	65.8	78.1
	V_{L2}	60.9	71.7
H3D	V_{L1}	52.6	74.6
	V_{L2}	48.3	69.9

Table 4. Overall accuracies of the vehicle type prediction on both datasets by both variants.

5.2.4 Vehicle Shape: Table 5 shows the RMS errors for the car dimensions (E_l, E_w, E_h), as well as the RMS errors of the differences between keypoints in planimetry ($E_{D^{2D}}$) and height ($E_{\Delta H}$). Again, V_{L1} achieves better results than V_{L2} and the results obtained on UAV are better than those for H3D.

Analysing the results for the car dimensions (E_l, E_w, E_h), it is obvious that the planimetric dimensions (length and width) are predicted quite accurately, with RMS errors in the order of ± 10 cm. Although the planimetric dimensions were derived from keypoint coordinates (cf. Section 5.1), this is essentially related to the capability of the model to predict the vehicle dimensions accurately. The planimetric RMS errors of the keypoints ($E_{D^{2D}}$) are also quite good (± 7 cm for V_{L1} on the UAV dataset). This error metric is more susceptible to the actual shape of the car than the metrics for length and width, because 24 key points of the ASM models are considered. Although no keypoint locations are estimated in the images, the planimetric positions of the keypoints are determined rather well, which indicates the high potential of the method. The RMS errors of vehicle (E_h) and keypoint height ($E_{\Delta H}$) are considerably larger (± 21 cm and ± 24 cm, respectively, for V_{L1} on the UAV dataset). Even in the best variant, the method has difficulties in predicting the third dimension accurately. However, one has to note that this has to be expected, because only monocular and near-nadir images are used for 3D reconstruction. For instance, a method solely aiming at a reconstruction of keypoints in 3D would not work at all given the available data; the only reason why the vehicle models can be reconstructed in 3D is the fact that we use a strong model, the ASM. Another aspect is that the lowest parts of the vehicles are hardly visible in near-nadir images, as shown in Figure 3, which might additionally hamper a more accurate determination of the height components of the keypoints.

Figure 3 shows some qualitative results obtained with variant V_{L1} tested on the UAV dataset. The wireframe fits the vehicles rather well. In these examples, the frontal parts of the vehicles are rather different from their backs, so that in these cases, in addition to the shape, the orientation can also be predicted well.

Data	Model	E_l [m]	E_w [m]	E_h [m]	$E_{D^{2D}}$ [m]	$E_{\Delta H}$ [m]
UAV	V_{L1}	0.11	0.08	0.21	0.07	0.24
	V_{L2}	0.12	0.10	0.26	0.08	0.26
H3D	V_{L1}	0.12	0.09	0.29	0.09	0.26
	V_{L2}	0.13	0.12	0.31	0.10	0.28

Table 5. Error metrics for shape estimation for both test datasets.

Figure 3. Qualitative results of V_{L1} on two UAV test images. The backprojected shapes of the reconstructed vehicles are represented by red wireframes.

6. CONCLUSION

In this paper, we have proposed a CNN-based method for the detection of vehicles from monocular UAV images, simultaneously predicting the vehicle pose, type and shape. The method is based on a final end-to-end training of all branches of the CNN. Additionally, we have presented a new dataset for evaluating vehicle detection and reconstruction methods. The main limitation of detection is related to truncated vehicles, which could not be detected well. The pose prediction results were very encouraging, with median errors in the range of 5 cm for the position and of less than 3° in orientation. However, in some cases, errors of about 180° in orientation were observed, mainly due to nearly symmetric vehicle shapes. The results of type prediction were not very accurate, mainly because of the similar appearance of some vehicle types. The evaluation of shape estimation has shown that the dimensions and the shapes of the vehicles can be determined very accurately in planimetry, with 2D RMS errors of keypoints in the order of ± 7 cm in images of about 3 cm GSD. The height component is less accurate (RMS errors of about ± 24 cm), which is to be expected when using near-nadir monoscopic images. Comparing two different variants of the loss minimised in training, our results show that the variant using a loss based on shape parameters (V_{L1}) is a better choice than the alternative.

In future work, we want to use the results of our method to build an approach for collaborative tracking of vehicles over time, considering not only the UAV images, but also street level images acquired from stereo cameras mounted on moving cars (Coenen and Rottensteiner, 2021). In this context, beyond the predicted poses, the shape parameters will be particularly useful in order to combine information acquired from images with such different viewing directions. Assuming the UAV and the cars to be able to communicate with each other, this would be the basis for collaborative positioning of the vehicles. This could be useful for autonomous driving, in particular in GNSS-denied areas.

ACKNOWLEDGEMENTS

This work was supported by the German Research Foundation (DFG) as a part of the Research Training Group i.c.sens [GRK2159]. Special thanks go to Norbert Haala, Michael Kölle and the entire team of the Hessigheim dataset for providing us with additional data that we used for our evaluation.

References

Ahmed, E., Saint, A., Shabayek, A. E. R., Cherenkova, K., Das, R., Gusev, G., Aouada, D., Ottersten, B. E., 2018. A survey on Deep Learning Advances on Different 3D Data Representations. *arXiv: Computer Vision and Pattern Recognition*.

Alidoost, F., Arefi, H., Tombari, F., 2019. 2D Image-To-3D Model: Knowledge-Based 3D Building Reconstruction (3DBR) Using Single Aerial Images and Convolutional Neural Networks (CNNs). *Remote Sensing*, 11(19).

Bi, S., Chai, Z., Liu, C., Xiong, Z., 2019. A segmentation-driven approach for 6d object pose estimation in the crowd. *International Conference on Advanced Intelligent Mechatronics*, 19–24.

Busch, S., 2019. Active shape model precision analysis of vehicle detection in 3D LiDAR point clouds. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-2/W13, 21–26.

Chabra, R., Lenssen, J., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R., 2020. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. *European Conference on Computer Vision (ECCV)*, 608–625.

Chen, W., Lin, C., Li, W., Yang, B., 2022. 3PSDF: Three-Pole Signed Distance Function for Learning Surfaces With Arbitrary Topologies. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 18522–18531.

Chen, Y., Medioni, G., 1991. Object modeling by registration of multiple range images. *IEEE International Conference on Robotics and Automation (ICRA)*, 2724–2729 vol.3.

Chibane, J., Alldieck, T., Pons-Moll, G., 2020. Implicit Functions in Feature Space for 3D Shape Reconstruction and Completion. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6970–6981.

Coenen, M., Rottensteiner, F., 2021. Pose estimation and 3D reconstruction of vehicles from stereo-images using a subcategory-aware shape prior. *ISPRS Journal of Photogrammetry and Remote Sensing*, 181, 27-47.

Cootes, T. F., Taylor, C. J., 1992. Active shape models—'smart snakes'. *BMVC92: Proceedings of the British Machine Vision Conference*, 266–275.

Ding, J., Xue, N., Long, Y., Xia, G.-S., Lu, Q., 2019. Learning roi transformer for oriented object detection in aerial images. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2844–2853.

Duggal, S., Wang, Z., Ma, W.-C., Manivasagam, S., Liang, J., Wang, S., Urtasun, R., 2022. Mending Neural Implicit Modeling for 3D Vehicle Reconstruction in the Wild. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 1900–1909.

El Amrani Abouelassad, S., Rottensteiner, F., 2022. Vehicle instance segmentation with rotated bounding boxes in uav images using cnn. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-1-2022, 15–23.

Engelmann, F., Stückler, J., Leibe, B., 2016. Joint object pose estimation and shape reconstruction in urban street scenes using 3d shape priors. *German Conference on Pattern Recognition*, 219—230.

García López, J., Agudo, A., Moreno-Noguer, F., 2019. Vehicle pose estimation via regression of semantic points of interest. *International Symposium on Image and Signal Processing and Analysis*, 209–214.

Hampel, F., Ronchetti, E., Rousseeuw, P., Stahel, W., 2005. *Robust Statistics: The Approach Based on Influence Functions*. Wiley & Sons.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn. *IEEE International Conference on Computer Vision (ICCV)*, 2980–2988.

Ku, J., Pon, A., Waslander, S., 2019. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 11859–11868.

Kölle, M., Laupheimer, D., Schmohl, S., Haala, N., Rottensteiner, F., Wegner, J. D., Ledoux, H., 2021. The Hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV LiDAR and Multi-View-Stereo. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, 1, 11.

Li, W., Meng, L., Wang, J., He, C., Xia, G.-S., Lin, D., 2021. 3d building reconstruction from monocular remote sensing images. *IEEE International Conference on Computer Vision (ICCV)*, 12528–12537.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A., 2019. Occupancy Networks: Learning 3D Reconstruction in Function Space. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 446–4470.

Moreno-Noguer, F., 2017. 3d human pose estimation from a single image via distance matrix regression. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1561–1570.

Newell, A., Yang, K., Deng, J., 2016. Stacked hourglass networks for human pose estimation. *European Conference on Computer Vision (ECCV)*, 483–499.

Nguyen, D., Duy, N., Truong, M., Bao, P., Nguyen, B., Nguyen, T., 2022. ASMCNN: An efficient brain extraction using active shape model and convolutional neural networks. *Information Sciences*, 591, 25-48.

Park, J. J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S., 2019. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 165–174.

Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A., 2020. Convolutional Occupancy Networks. *European Conference on Computer Vision (ECCV)*, 523–540.

Reddy, N. D., Vo, M., Narasimhan, S. G., 2019. Occlusion-net: 2d/3d occluded keypoint localization using graph networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7318–7327.

Ren, S., He, K., Girshick, R., Sun, J., 2017. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), 1137-1149.

Shi, J., Yang, H., Carlone, L., 2021. Optimal pose and shape estimation for category-level 3d object perception. *Robotics: Science and Systems XVII*, Robotics: Science and Systems Foundation.

Sundermeyer, M., Marton, Z., Durner, M., Brucker, M., Triebel, R., 2018. Implicit 3d orientation learning for 6d object detection from rgb images. *European Conference on Computer Vision (ECCV)*, 712–729.

Tekin, B., Sinha, S., Fua, P., 2018. Real-time seamless single shot 6d object pose prediction. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 292–301.

Xiao, Y.-P., Lai, Y.-K., Zhang, F.-L., Li, C., Gao, L., 2020. A survey on deep geometry learning: From a representation perspective. *Computational Visual Media*, 6, 113-133.

Xu, Q., Wang, W., Ceylan, D., Mech, R., Neumann, U., 2019. DISN: Deep Implicit Surface Network for High-quality Single-view 3D Reconstruction. *Advances in Neural Information Processing Systems*, 32.

Zhou, D., Petrovska-Delacrétaz, D., Dorizzi, B., 2010. 3d active shape model for automatic facial landmark location trained with automatically generated landmark points. *International Conference on Pattern Recognition*, 3801–3805.

Zia, M. Z., Stark, M., Schiele, B., Schindler, K., 2013. Detailed 3D representations for object recognition and modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11), 2608–2623.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-1-W1-2023-935-2023 | © Author(s) 2023. CC BY 4.0 License.